



ФКН

ТРУДЫ МОЛОДЫХ УЧЕНЫХ ФАКУЛЬТЕТА КОМПЬЮТЕРНЫХ НАУК ВГУ

ВЫПУСК 4

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет компьютерных наук

**ТРУДЫ МОЛОДЫХ УЧЁНЫХ
ФАКУЛЬТЕТА КОМПЬЮТЕРНЫХ
НАУК ВГУ**

В ы п у с к 4

Под редакцией кандидата технических наук Д. Н. Борисова

Воронеж
2024

УДК 004.65 + 004.438.5
ББК Ч481(2)22
Т78

*Рекомендовано к опубликованию
Ученым советом факультета компьютерных наук ВГУ*

Т78 **Труды молодых учёных факультета компьютерных наук ВГУ. Выпуск 4 / под ред. Д. Н. Борисова ; Воронежский государственный университет. – Воронеж, 2024. – 588 с.**

В сборник включены научные работы студентов бакалавриата и магистратуры факультета компьютерных наук ВГУ, выполненные в 2023-2024 гг. под руководством преподавателей факультета, представленные в виде докладов и рекомендованные к опубликованию оргкомитетами студенческих научных конференций.

УДК 004.65 + 004.438.5
ББК Ч481(2)22

© Воронежский государственный
университет, 2024

Статъи студентов бакалавриата

Разработка веб-приложения для генерации документов, сопутствующих государственной регистрации объектов интеллектуальной собственности

Е. А. Бабкина

Студент бакалавр

Е. А. Копытина

Старший преподаватель

Введение

Подача заявки на регистрацию объектов интеллектуальной собственности сопровождается большим количеством стандартных документов. Задача обработки этих документов является сложной и рутинной. Связанные с ней действия требуют внимательности авторов объектов интеллектуальной собственности и сотрудников отдела защиты интеллектуальной собственности Воронежского государственного университета, большого количества времени и чреваты ошибками, так как документы заполняются не автоматически, а вручную. Дополнительные, зачастую избыточные действия, а также участие людей в процессе повышает шансы ошибок и замедляет ведение дел, связанных с результатами интеллектуальной деятельности.

Автоматизированный подход к выполнению повторяющихся действий является наиболее современным и эффективным решением подобных проблем, как показано в [1-5].

Проект, описанный в данной статье, является крайне актуальным, так как в современном мире тема документооборота, связанного с интеллектуальной собственностью, особенно востребована. Кроме того, разработанное веб-приложение способно устранить вышеописанные сложности, проблемы и уменьшить риски, сопутствующие ручному заполнению документов.

1. Постановка задачи

К описываемому в статье веб-приложению были выдвинуты следующие функциональные требования:

- Возможность введения данных пользователем;

- Автоматическая генерация документов, сопутствующих государственной регистрации объектов интеллектуальной собственности, на основе введенных данных;
- Возможность скачивания сгенерированных документов.

2. Реализация логики

Веб-приложение имеет клиент-серверную архитектуру. Клиентская и серверная части взаимодействуют по принципам REST API посредством обмена HTTP-запросами.

Серверная часть веб-приложения написана на языке Java. Для её реализации был использован фреймворк Spring Boot. Данная часть приложения состоит из следующих модулей, представленных в программе в виде отдельных пакетов с необходимыми классами:

- Application, где находится главный класс приложения с аннотацией `@SpringBootApplication`, откуда запускается приложение;
- Controller, где находится единственный класс контроллера `DocumentsController`, отвечающий за обработку HTTP-запросов и взаимодействие с клиентом (браузером);
- Config, где находится единственный класс `SecurityConfig`, в котором настроены основные компоненты Spring Security для обеспечения безопасности серверной части приложения;
- Service, где находятся общий класс сервиса `DocumentsService` и классы, относящиеся к каждому документу, которые содержат логику приложения, связанную с генерацией документов;
- Entity, где находятся классы всех документов, которые хранят данные, необходимые для заполнения этих документов;
- DTO, где хранится единственный класс `UserDataDto`, который хранит все данные, полученные от пользователя, и в процессе программы преобразуется в сущности, соответствующие документам;
- Mapper, где содержатся классы для каждого документа, описывающие логику преобразования DTO в сущности.

Клиентская часть приложения разработана на языке JavaScript с использованием фреймворка React.js и представляет собой SPA (single page application). Особенностью разработки приложения на React является разбиение на компоненты. Компоненты представляют собой функции, возвращающие определённый HTML код.

Данная часть приложения состоит из следующих модулей, представленных в программе в виде отдельных папок:

- Inputs, где находятся файлы с расширением .js, соответствующие полям ввода на сайте (например, ProgramName.js для ввода названия проекта);
- Forms, где находится файл AuthorForm.js, объединяющий все поля ввода, относящиеся к данным об авторах, а также файл GeneratorForm.js, объединяющий все поля ввода в единую форму;
- Pages, где находится единственный файл GeneratorPage.js, формирующий страницу веб-приложения, на которой расположена форма, а также вступительная информация к ней;
- Styles, где расположен единственный файл с CSS стилями.

3. Реализация интерфейса

Перед вводом данных в форму пользователю предлагается ознакомиться со вступительной информацией, в том числе с правилами заполнения формы. Данный раздел представлен на рис. 1.



Рис. 1. Вступительная информация

Далее пользователя встречает раздел, связанный с общей информацией о проекте. Перед каждым полем описывается, что именно необходимо в него ввести, а при наведении на поле высвечивается подсказка с примером ввода. В этом разделе доступен выбор папки с исходным кодом приложения через открывающийся проводник. Вид первого раздела представлен на рис. 2.

1

Информация о проекте

Введите название программы для ЭВМ (или базы данных)

Название проекта

Введите аннотацию к программе для ЭВМ (или базе данных). Вы можете кратко описать, для чего предназначена программа, где она будет использоваться

Аннотация

Кратко опишите, что было реализовано в программе (методы, паттерны, алгоритмы и т.д.). Если Вы хотите упомянуть несколько реализованных идей, используйте нумерацию

Что реализовано

Пример: Java

Введите язык программирования, который использовался при разработке

Язык программирования

Введите объем программы с указанием единицы измерения объема информации

Объем программы

Выберите основную папку с исходным кодом программы (src)

Выбрать папку

Выберите вид результата интеллектуальной деятельности:

☒ Программа для ЭВМ

☐ База данных, государственная регистрация которой осуществляется в соответствии с пунктом 4 статьи 1259 Кодекса

☐ База данных, государственная регистрация которой осуществляется в соответствии с пунктом 3 статьи 1334 Кодекса

Рис. 2. Раздел «Информация о проекте»

Второй раздел посвящен информации об авторах проекта. В этом разделе изначально доступно заполнение полей для одного автора, однако в любой момент можно добавить поля для любого количества дополнительных авторов или удалить эти поля. Ввод дат доступен через открывающийся календарь. Вид части второго раздела представлен на рис. 3.

2 Информация об авторах

Добавить автора

Удалить автора

Автор №1

Введите фамилию, имя, отчество автора без сокращений

ФИО

Введите дату рождения автора

Дата рождения

Введите адрес места жительства автора: страна, (аббревиатура страны), почтовый индекс, населенный пункт, улица, дом, квартира

Адрес

Рис. 3. Раздел «Информация об авторах»

Третий раздел посвящен дополнительной информации о проекте, связанной с факультетом и обоснованием регистрации объекта интеллектуальной собственности. Вид данного раздела представлен на рис. 4.

3 Дополнительная информация

Выберите факультет, на котором велась разработка программы для ЭВМ или базы данных

Выберите факультет

Введите полное описание авторов, включая их ученую степень, должность, звание (при наличии). Дополните предложение: "Результат интеллектуальной деятельности разработан ..." (кем?)

Результат интеллектуальной деятельности разработан ...

Введите дополнительное обоснование для получения свидетельства о государственной регистрации. Возможные причины: защита ВКР, получение стипендии и т.д. Дополните предложение: "Свидетельство о государственной регистрации результата интеллектуальной деятельности требуется для ..."

Свидетельство о государственной регистрации требуется для ...

Сгенерировать документы

Рис. 4. Раздел «Дополнительная информация»

При нажатии на кнопку «Сгенерировать документы» генерируются следующие документы: листинг программы, реферат, согласие на обработку персональных данных (для каждого автора), согласие на использование сведений об авторе, указанных в заявлении (для каждого автора), обоснование рекомендации от факультета. Все документы помещаются в архив, и для выбора места скачивания архива открывается проводник. Пример сгенерированного согласия на обработку персональных данных представлен на рис. 6.

В Федеральную службу
по интеллектуальной собственности
Бережковская наб., д. 30, корп. 1,
г. Москва, Г-59, ГСП-3, 125993,
Российская Федерация

Название программы для ЭВМ или базы данных: «Веб-приложение для генерации документов сопутствующих государственной регистрации объектов интеллектуальной собственности»

№ заявки _____
(указывается при наличии регистрационного номера заявки)

Согласие на обработку персональных данных

Ф. И. О. субъекта персональных данных Иванов Иван Иванович

Адрес места жительства Россия, (Ru), 395042, г. Воронеж, ул. Ленина, д. 100, кв. 9

Документ, удостоверяющий личность субъекта персональных данных, дата его выдачи и выдавший орган паспорт: серия 2010 номер 010101, 20.02.2020, выдан ГУ МВД России по Воронежской области

Подтверждаю согласие на обработку моих персональных данных, предусмотренную частью 3 статьи 3 Федерального закона от 27 июля 2006 г. № 152-ФЗ «О персональных данных», в целях предоставления Федеральной службой по интеллектуальной собственности государственной услуги в соответствии с Федеральным законом от 27 июля 2010 г. № 210-ФЗ «Об организации предоставления государственных и муниципальных услуг».

Мне известно, что в случае отзыва согласия на обработку персональных данных Федеральная служба по интеллектуальной собственности вправе продолжить обработку персональных данных без моего согласия в соответствии с частью 2 статьи 9, пунктом 4 части 1 статьи 6 Федерального закона от 27 июля 2006 г. № 152-ФЗ «О персональных данных».

Подпись _____
(Ф. И. О. субъекта персональных данных)

Дата _____

Рис. 5. Документ «Согласие на обработку персональных данных»

Заключение

В данной статье была рассмотрена разработка веб-приложения для генерации документов, сопутствующих государственной регистрации объектов интеллектуальной собственности. Все поставленные в начале разработки приложения функциональные требования и задачи были

выполнены. Приложение является актуальным, востребованным и полностью готовым к использованию.

Список литературы

1. Васянкин, А. А. Модификация понятия «электронный документ» в системе электронного документооборота / А. А. Васянкин, Э. В. Кузьмина // Кайгородовские чтения : Материалы региональной научно-практической конференции, Краснодар, 29 апреля 2010 года / Краснодарский государственный университет культуры и искусств. – Краснодар: Краснодарский государственный университет культуры и искусств, 2010. – С. 75-79.

2. Свидетельство о государственной регистрации программы для ЭВМ № 2023661037 Российская Федерация. Приложение для автоматизации документооборота дополнительного образования : № 2023618813 : заявл. 03.05.2023 : опубл. 25.05.2023 / А. С. Голышева, Е. А. Копытина ; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Воронежский государственный технический университет». – EDN NBVMM.

3. Шорохова, Т. А. Разработка проекта «List_of_documents» для реализации программ дополнительного образования ФГБОУ ВО «Воронежского государственного университета» / Т. А. Шорохова, Е. А. Копытина // Труды молодых учёных факультета компьютерных наук ВГУ : Сборник статей. – Воронеж : Воронежский государственный университет, 2023. – С. 284-289. – EDN MMGXTK.

4. Претков, А. И. Разработка программного комплекса для автоматизации формирования документов основных профессиональных образовательных программ высшего образования факультета компьютерных наук Воронежского государственного университета / А. И. Претков, Е. А. Копытина // Информатика: проблемы, методы, технологии : материалы XXIII Международной научно-практической конференции им. Э.К. Алгазина, Воронеж, 15–17 февраля 2023 года. – Воронеж: Воронежский государственный университет, 2023. – С. 907-912. – EDN ENWODO.

5. Претков, А. И. Автоматизации формирования документов основных профессиональных образовательных программ высшего образования факультета компьютерных наук Воронежского государственного университета / А. И. Претков, Е. А. Копытина // Труды молодых учёных факультета компьютерных наук ВГУ : Сборник статей. – Воронеж : Воронежский государственный университет, 2023. – С. 94-107. – EDN HKXMIS.

Моделирование распространения сигнала в городских условиях

Д. С. Бадалян

Студент бакалавр

Д. Н. Борисов

Доцент

Введение

Распространение радиоволн в городской среде является важной задачей в области телекоммуникаций и беспроводных сетей передачи данных. Распространение радиосигналов в городских условиях имеет ряд особенностей, которые необходимо учитывать для эффективного планирования, развертывания и оптимизации беспроводных сетей связи.

Точное моделирование распространения радиоволн в городской среде является сложной задачей из-за наличия многочисленных препятствий: зданий, деревьев, транспортных средств и других объектов, которые могут влиять на распространение сигнала. Существующие модели распространения радиоволн, такие как модель Хата и COST-231, широко используются для планирования сетей.

Выбор темы исследования обусловлен необходимостью всестороннего изучения и анализа моделей распространения радиосигналов в городской среде, а также оценки их точности и применимости. Результаты исследования могут быть использованы для повышения эффективности планирования и развертывания беспроводных сетей связи.

Основной целью данного исследования является всестороннее изучение и анализ моделей распространения радиосигналов в городских условиях, а также оценка их применимости для планирования и оптимизации беспроводных сетей связи.

1. Модель Окумуры-Хаты

Модель Хата – модель распространения радиосигналов для прогнозирования потерь на пути передачи сотовой связи с учетом внешних условий. Используется для частот от 150 до 1500 МГц и является эмпирической формулировкой, основанной на результатах исследований Окумуры (ее также называют моделью Окумуры–Хаты).

Модель развивает исследования Окумуры и учитывает эффекты дифракции, отражения и рассеяния радиоволн на городских объектах. Кроме того, модель Хата учитывает особенности распространения радиоволн в пригородных и сельских условиях [1, 2, 3].

Модель Хата используют в следующих условиях:

- Рабочая частота: 150-1500 МГц;
- Высота антенны мобильной станции: 1-10 м;
- Высота антенны базовой станции: 30-200 м;
- Протяженность трассы : 1-10 км.

Среднее затухание радиосигнала в городских условиях рассчитывается по эмпирической формуле:

$$L_U = 69.55 + 26.16 \log_{10} f - 13.82 \log_{10} h_B - C_H + \\ + [44.9 - 6.55 \log_{10} h_B] \log_{10} d, \quad (1)$$

где:

CH – поправочный коэффициент для высоты антенны АС;

LU – Потери сигнала в версии для городских условий (dB);

f – Рабочая частота в МГц;

hB – Высота подъема антенны базовой станции (m);

hM – Высота подъема антенны мобильной станции (m);

d – Дальность связи (km).

2. Модель COST-231

Для диапазона частот $f = 1,5 \dots 2$ ГГц широко используется модель COST231- Хата, которая является модифицированным вариантом модели Хата [1, 2, 3].

Модель COST231-Хата применяют при изменении значений параметров в пределах:

- Рабочая частота: 1500-2000 МГц;
- Высота антенны мобильной станции: 1-10 м;
- Высота антенны базовой станции: 30-200 м;
- Протяженность трассы: 1-20 км.

Формула для расчета среднего затухания в городе (дБ) записывается в виде:

$$L_b = 46.3 + 33.9 \log_{10} f - 13.82 \log_{10} h_B - a(h_R, f) + \\ + [44.9 - 6.55 \log_{10} h_B] \log_{10} d + C_m, \quad (2)$$

3. Моделирование модели Хата в Matlab

Для модели Хата рассмотрим влияние несущей частоты антенны на затухание сигнала. Для этого в программе Matlab построим зависимости

затухания сигнала от расстояния между антеннами базовой и мобильной станций [4].

На рис. 1 изображен график, который позволяет оценить затухание сигнала – оно увеличивается с увеличением частоты.

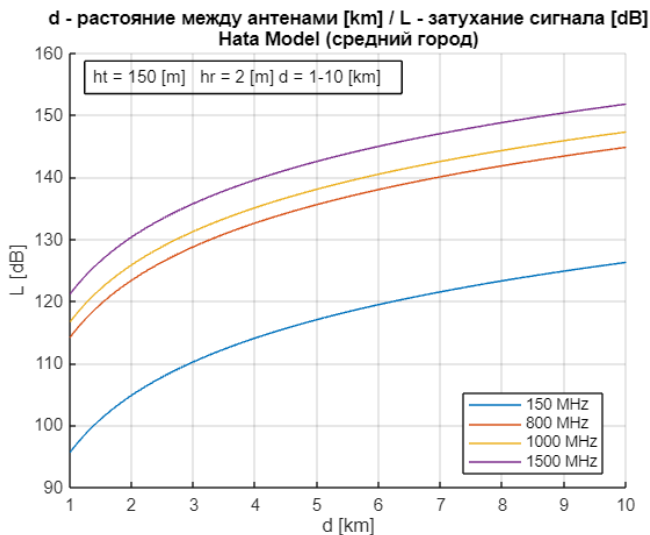


Рис. 1. График затухания сигнала для разных частот

4. Моделирование модели COST-231в Matlab

Чтобы понять, как влияет размер города на поведение сигнала, воспользуемся моделью COST-231. С помощью модели COST-231 следует рассчитать потери сигнала для различных значений расстояния между антеннами базовой и мобильной станций. В крупном городе потери сигнала больше, чем в среднем (рис. 2). Это может быть обусловлено большей плотностью застройки, дифракцией и усиленным искажением сигнала [4].

5. Сравнение моделей Хата и COST-231

Проанализируем результаты, полученные по моделям Хата и COST-231. На рис. 3 видно, что при одинаковых условиях, которые подходят для обеих моделей, потери сигнала у модели COST-231 выше чем у модели Хата.

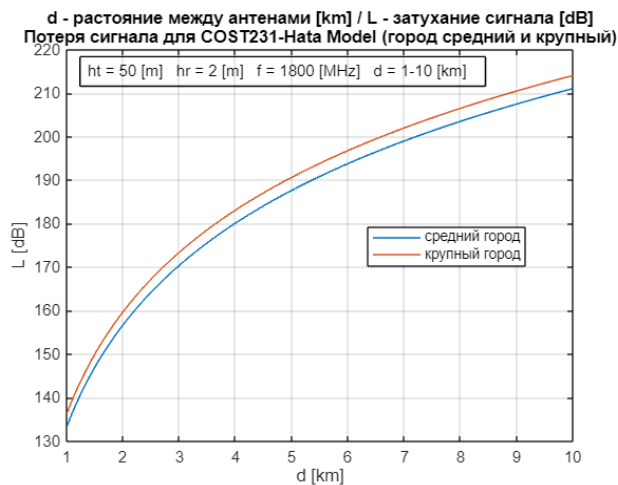


Рис. 2. График затухания сигнала для городов разных размеров

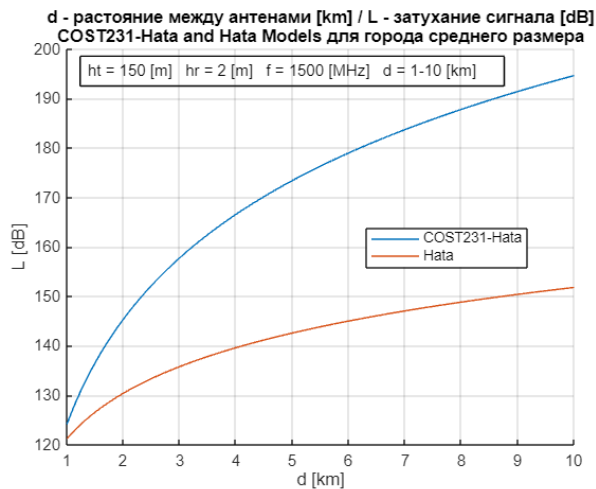


Рис. 3. График затухания сигнала для двух моделей

Так как модель COST-231 основана на модели Хата и по-сути дополняет ее, то она учитывает большее количество факторов, чем модель Хата.

6. Построение модели распространения сигнала в RadioPlanner

Рассмотренные модели – усреднение экспериментальных исследований для различных городов. Для получения результатов конкретной местности воспользуемся алгоритмами численных расчетов, реализованных в программе RadioPlanner [5]. RadioPlanner позволяет рассчитать зону покрытия базовых станций согласно модели Окумуры-Хаты с учетом карты местности.

В качестве города для которого будем проводить оценку выберем г. Воронеж, а именно территория, прилегающая к Воронежскому государственному университету. Необходимые данные для задания характеристик базовых станций возьмем на сайте реестра санитарно-эпидемиологических заключений на проектную документацию [6].

На рис. 4 показано, как выглядит карта, с размещенными базовыми станциями оператора МТС.

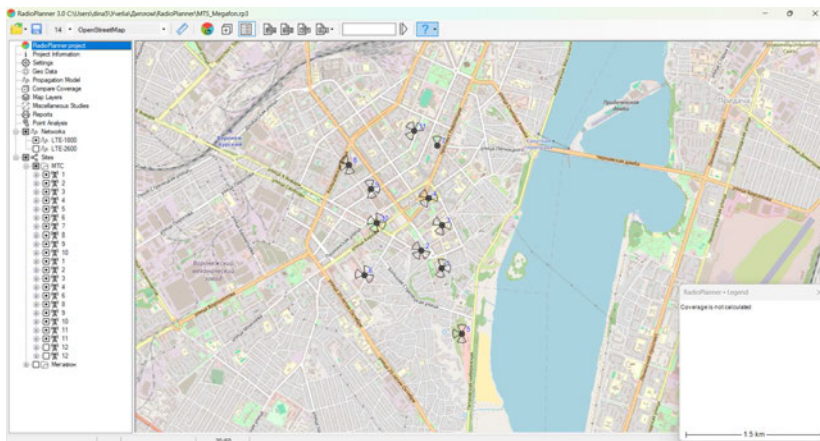


Рис. 4. Карта Воронежа с размещенными базовыми станциями в программе RadioPlanner

Моделирование будем проводить для стандарта LTE-1800 (диапазон частот 1710-1880 МГц, модель распространения – модель Окумуры-Хаты). Антенны используются в основном трехсекторные, с заданными характеристиками по [6]. На рис. 5 изображена карта покрытия и принимаемая мощность сигнала для антенны моильной станции высотой $h=3$ м. Разные цвета отображают уровень принимаемой мощности сигнала.

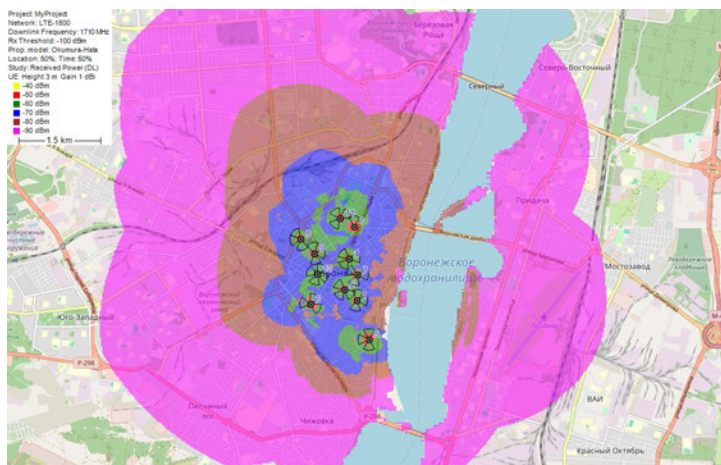


Рис. 5. Карта покрытия сигнала базовыми станциями МТС в Воронеже

В качестве альтернативного решения может быть предложен следующий вариант размещения базовых станций, изображенный на рис. 6. Такое расположение базовых станций позволяет повысить уровень принимаемого сигнала в области ВГУ Воронежа, что положительно скажется на качестве связи.

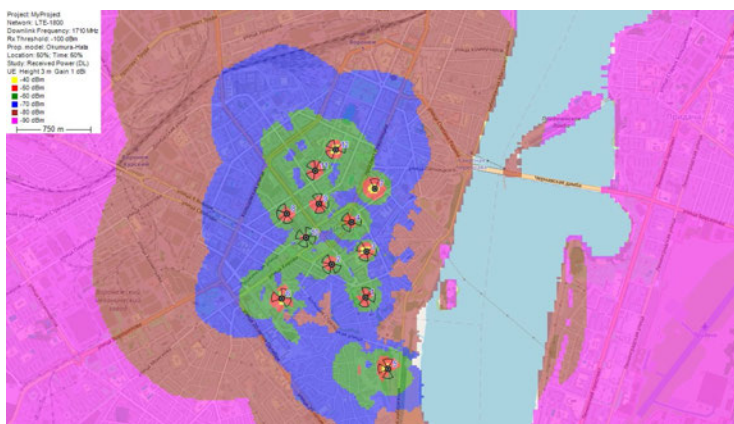


Рис. 6. Карта альтернативного покрытия сигнала базовыми станциями МТС в Воронеже

Заключение

Влияние высоты антенн, несущей частоты и мощности базовых станций играет ключевую роль в планировании сетей и оценки затухания сигнала в точке приема. Для расчета потерь сигнала в городских условиях были использованы модели COST-231 и Хата с использованием программы MATLAB. Модель COST-231 в среднем показывает большие потери сигнала по сравнению с моделью Хата. В условиях метрополитенов и крупных городов потери сигнала выше, чем в менее застроенных областях.

Моделирование зоны покрытия с использованием инструментов, таких как RadioPlanner, позволяет точно оценить влияние различных факторов на качество сигнала в заданной местности. При этом на нижних частотах наблюдаются меньшее затухание сигнала и больший радиус покрытия, а увеличение несущей частоты приводит к потере мощности и радиуса покрытия. Для качественного покрытия в городской среде необходимо тщательно планировать размещение базовых станций, а также учитывать особенности местности и плотность застройки.

Таким образом, комплексный подход к моделированию и анализу позволяет принимать обоснованные решения при планировании мобильной сети, улучшать качество обслуживания в различных условиях.

Список литературы

1. Макоевеева, М.М. Системы связи с подвижными объектами: учеб. пособие для вузов / М.М. Макоевеева, Ю.С. Шинаков. – М.: Радио и связь, 2002. – 440 с.
2. Федоров, С. М. Системы и сети связи с подвижными объектами: учеб. пособие для вузов / С.М. Федоров, И.А. Черноиваненко. – Воронеж: ФГБОУ ВО «Воронежский государственный технический университет», 2021. – 84 с.
3. Печаткин А.В. Системы мобильной связи: учеб. пособие / А.В. Печаткин. – Рыбинск: РГТА, 2008. – 122 с.
4. MathWorks [Электронный ресурс]: программный продукт. – Режим доступа : <https://www.mathworks.com/products/matlab.html>
5. RadioPlanner 3.0 Частотно-территориальное планирование подвижной радиосвязи, ТВ- и радиовещания [Электронный ресурс]: программный продукт. – Режим доступа : <https://www.ctt-group.ru/radioplanner>
6. Реестр санитарно-эпидемиологических заключений на проектную документацию [Электронный ресурс]: программный продукт. – Режим доступа : <http://fp.crc.ru/doc/?oper=s&uinz=%291K%28-J4YC%28%3D&pdk=on&pril=on>

Разработка информационной системы, автоматизирующей генерацию шаблонов документов, сопутствующих прохождению учебных практик ФКН ФГБОУ ВО «ВГУ»

Е. О. Бордюжа

Студент бакалавр

Е. А. Копытина

Старший преподаватель

Н. С. Богданова

Ассистент

Введение

Университеты и другие образовательные организации всё чаще сталкиваются с необходимостью создания и хранения большого количества документов, связанных с учебным процессом. Это может быть трудоёмкой задачей, которая займет значительное количество времени, особенно если эти документы имеют стандартный формат, но при этом должны быть заполнены уникальными данными для каждого студента. В статье рассматривается разработка информационной системы (веб-приложения), которая позволяет автоматизировать процесс генерации шаблонов документов прохождения учебной практики в ФГБОУ ВО «ВГУ». Аналогичные подходы к решению такого рода задач были подробно рассмотрены в [1-6].

Цель исследования заключается в создании веб-приложения под названием «DocFormer», способного автоматизировать процесс генерации документов, используемых в учебной деятельности. В данной статье представлено описание процесса разработки такой системы и анализ её функциональных возможностей.

В заключении сделан вывод о том, насколько эффективно разработанное приложение для автоматизации генерации шаблонов документов, сопутствующих прохождению учебных практик ФКН ФГБОУ ВО «ВГУ», а также рассмотрены перспективы его использования.

1. Постановка задачи

К описываемому в статье веб-приложению «DocFormer» были выдвинуты следующие требования:

- Реализовать авторизацию пользователя;
- Создать механизм, позволяющий скачивать заполненные шаблоны документов с автоматически сгенерированными данными для каждого студента;
- Реализовать возможность заполнения БД списком студентов из excel-файла;
- Спроектировать понятный и удобный для каждого пользователя интерфейс веб-приложения;
- Предоставить возможность преподавателю создавать задания с необходимыми полями для заполнения и прикреплением файлов;
- Реализовать возможность оценивать прикрепленные к заданиям работы студентов;
- Предоставить возможность получать оповещения о присланных ответах студента по почте;
- Для студента реализовать возможность просматривать задания, добавлять ответ на них с прикреплением нужных файлов и получать уведомления об оценке преподавателя;
- Предоставить возможность получать оповещение по почте и в самом приложении об оценённых работах от преподавателя.

2. Реализация логики

Разработанное приложение имеет клиент-серверную архитектуру. Клиент и сервер взаимодействуют посредством обмена HTTP-запросов. Для серверной части выбран объектно-ориентированный язык программирования Python [7]. Для реализации были использованы следующие модули:

- Pandas;
- Django;
- Docxtpl.

Управление базой данных осуществляется в открытой системе PostgreSQL, которая позволяет хранить и обрабатывать большие объемы данных. Документирование API производится через Swagger.

Для реализации генерации шаблонов документов были разработаны следующие функции:

- `def convert_excel_to_csv(excel_file_path, output_csv_path);`
- `def create_students(line, username, password);`
- `def filling_document_report(user, assignment, temp_dir);`
- `def download_docs(report_filename, diary_filename, temp_dir).`

Процесс генерации начинается с загрузки Excel файла, содержащего данные о студентах. Этот файл проходит через функцию `convert_excel_to_csv`, в котором каждая строка и столбец разбираются на составные части, после чего данные заносятся в базу с помощью функции `create_students`. Затем происходит процесс заполнения шаблонного doc-файла через модуль `filling_document_report`. После проведения описанных действий пользователи могут скачивать автоматически сгенерированные шаблоны с помощью функции `download_docs`.

В основном модуле `UserController.py` были реализованы функции-запросы, взаимодействующие с клиентом. Для реализации клиентской части был выбран язык программирования TypeScript, имеющий статическую типизацию, и библиотека React для создания пользовательских интерфейсов.

3. Реализация интерфейса

Работа в веб-приложении начинается с авторизации пользователя. Вид окна авторизации представлен на рис. 1.

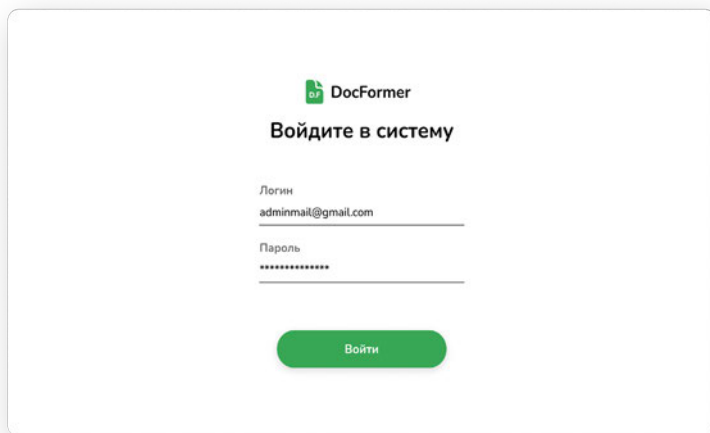


Рис. 1. Экран авторизации пользователя

После успешной авторизации пользователь переходит на главную страницу. Вид главной страницы зависит от типа пользователя: преподаватель или студент. Для преподавателя открывается экран, на

котором он сразу может приступить к созданию задания, нажав на кнопку «Создать», как показано на рис. 2.

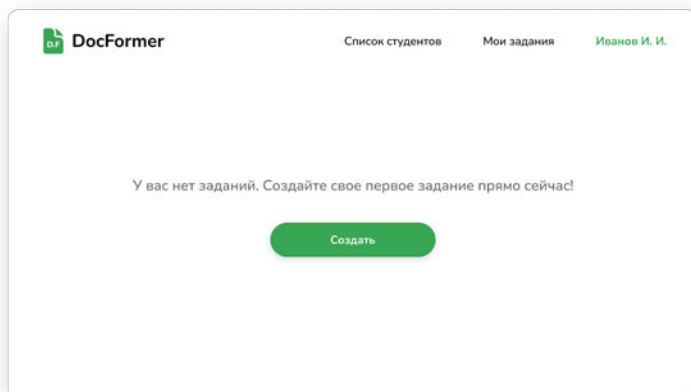


Рис. 2. Главный экран преподавателя

После чего открывается форма по созданию задания (рис. 3).

The screenshot displays the 'Создание задания' (Create Task) form. The title 'Создание задания' is followed by a close icon 'X'. The form contains four main input areas: 'Название' (Name) with a single-line text field, 'Описание' (Description) with a multi-line text area, 'Крайний срок сдачи' (Deadline) with a date picker showing 'ДД.ММ.ГГГГ', and 'Прикрепить файлы' (Attach files) with a dashed border, a file upload icon, and the text 'Максимальное количество файлов: 3'. A green 'Создать' button is located at the bottom right of the form.

Рис. 3. Форма для создания задания

Созданные задания отображаются во вкладке «Мои задания», где их можно просматривать, а также скачивать прикрепленные документы,

автоматически заполняющиеся сгенерированными данными индивидуально для каждого студента. Студент может дать ответ на задание, прикрепив необходимые файлы (рис. 4).

DocFormer Уведомления Мои задания Петров П. П.

Шаблоны документов для производственной практики ИСнТУП X

Необходимо заполнить прикреплённые шаблоны документов в соответствии с производственной практикой на вашем предприятии. Получить все подписи у руководителей практик.

Крайний срок сдачи: 28.05.2024

Прикреплённые файлы: Шаблон отчета по практике.docx
Шаблон дневника по практике.docx

Добавить ответ на задание

Комментарий

Прикрепить файлы Максимальное количество файлов: 3

Отправить

Рис. 4. Просмотр задания для студента

Далее преподавателю придет по почте сообщение о том, что студент прикрепил ответ на задание. После чего он может оценить работу обучающегося и дать свой отзыв на нее (рис. 5).

DocFormer Список студентов Мои задания Иванов И. И.

Ответ на задание "Шаблоны документов для производствен..." X

Студент: Петров П. П.

Отправлено: 26.05.2024

Комментарий: Проходил практику на предприятии "НойНейн". Прикреплено необходимые документы.

Прикреплённые файлы: Петров Ответ по практике.docx
Петров Дневник по практике.docx

Отзыв

Преподаватель: Иванов И. И.

Оценка: 100/100

Оценено: 27.05.2024

Комментарий: Выполнено верно. Ваша производственная практика зачтена.

Перезагрузить

Рис. 5. Ответ на задание от студента и отзыв преподавателя

После того, как преподаватель оценит работу, студенту придет оповещение по почте и в самом веб-приложении в разделе «Уведомления», откуда он может перейти на отзыв преподавателя и просмотреть оценку (рис. 6-7).

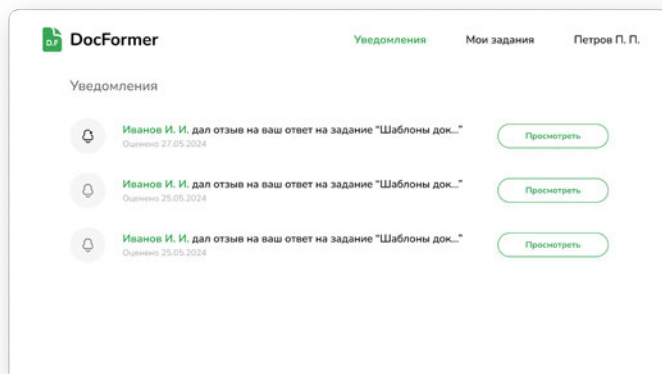


Рис. 6. Вкладка «Уведомления» у студента

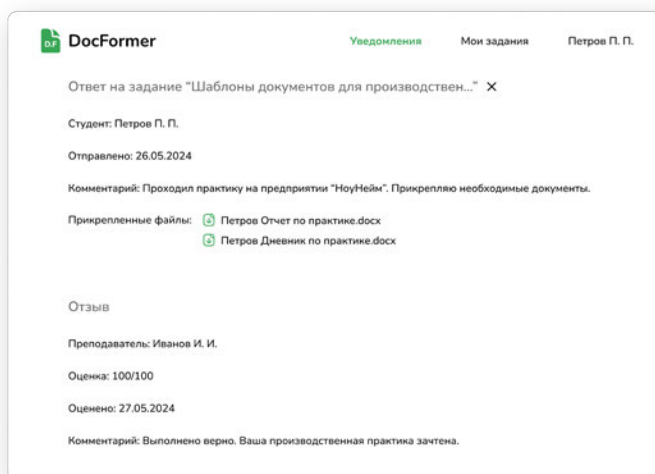


Рис. 7. Просмотр отзыва преподавателя для студента

Заключение

В статье была рассмотрена разработка информационной системы, автоматизирующей генерацию шаблонов документов, сопутствующих прохождению учебных практик ФКН ФГБОУ ВО «ВГУ», которая представляет собой важную и актуальную задачу, способствующую оптимизации оформления документов студентами и преподавателями. Созданное веб-приложение, основанное на данной системе, обеспечивает улучшение качества оформления документов, сокращение времени на подготовку и уменьшение количества ошибок.

Перспективы развития включают интеграцию с другими системами управления учебным процессом и расширение функционала для использования в других учебных заведениях, что позволит значительно упростить процесс управления документами и повысить эффективность учебного процесса. Все поставленные в начале разработки приложения функциональные требования были выполнены. Веб-приложение является востребованным и полностью работоспособным.

Список литературы

1. Копытина, Е. А. Разработка веб-сервиса для формирования горячего коммерческого предложения ООО «ЧерноземАгроماش» / Е. А. Копытина, Ф. А. Десятириков, А. В. Копытин // Информатика: проблемы, методы, технологии : Материалы XXII Международной научно-практической конференции им. Э.К. Алгазина, Воронеж, 10–12 февраля 2022 года / Под редакцией Д.Н. Борисова. – Воронеж: Общество с ограниченной ответственностью "Вэлборн", 2022. – С. 990-996.
2. Свидетельство о государственной регистрации программы для ЭВМ № 2023661037 Российская Федерация. Приложение для автоматизации документооборота дополнительного образования : № 2023618813 : заявл. 03.05.2023 : опубл. 25.05.2023 / А. С. Гольшева, Е. А. Копытина ; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Воронежский государственный технический университет». – EDN NBUVMM.
3. Претков, А. И. Автоматизации формирования документов основных профессиональных образовательных программ высшего образования факультета компьютерных наук Воронежского государственного университета / А. И. Претков, Е. А. Копытина // Труды молодых учёных факультета компьютерных наук ВГУ : Сборник статей. – Воронеж : Воронежский государственный университет, 2023. – С. 94-107. – EDN НКXMIS.
4. Новохатский, Н. М. Разработка мобильного приложения для медицинских клиник "Медангел" / Н. М. Новохатский, А. В. Копытин,

Е. А. Копытина // Сборник студенческих научных работ факультета компьютерных наук ВГУ : Сборник статей. В 2-х частях / Под редакцией Д.Н. Борисова. Том Выпуск 13. Часть 1. – Воронеж : Воронежский государственный университет, 2019. – С. 150-155.

5. Проектная деятельность в дополнительном образовании на примере практик российских ВУЗов / Е. А. Копытина, Л. А. Гладкова, М. Е. Семенов [и др.] // Информатика: проблемы, методы, технологии : Материалы XXII Международной научно-практической конференции им. Э.К. Алгаинова, Воронеж, 10–12 февраля 2022 года / Под редакцией Д.Н. Борисова. – Воронеж: Общество с ограниченной ответственностью "Вэлборн", 2022. – С. 1342-1355.

6. Подольский, К. Д. Формирование предварительного коммерческого предложения ООО «ЧерноземАгромаш» средствами MS Excel / К. Д. Подольский, Я. Э. Головин, Е. А. Копытина // Сборник студенческих научных работ факультета компьютерных наук ВГУ : Сборник научных работ. В 2-х частях / Под редакцией Д.Н. Борисова. Том Выпуск 14. Часть 1. – Воронеж : Воронежский государственный университет, 2020. – С. 143-147.

7. Официальная документация Python [Электронный ресурс]. – Python Software Foundation. – Режим доступа: <https://docs.python.org>. – Заглавие с экрана. – (Дата обращения: 18.03.2024). DOI: 10.5281/zenodo.1234567

GPU-реализация машинно-обучаемых потенциалов для молекулярной динамики

А. С. Будников

Студент бакалавр

А. В. Романов

Старший преподаватель

Введение

Результатом развития современных вычислительных систем стал перенос части математических расчетов на графические процессоры, которые позволяют проводить массивные параллельные вычисления с производительностью, кратно превышающей производительность центральных процессоров. Отражением этих тенденций стал рост числа суперкомпьютеров с гибридной архитектурой, т.е. с использованием графических сопроцессоров. Более того, современные видеокарты пользовательского уровня предоставляют возможности ранее доступные только в формате больших машин.

Широко распространенным типом таких вычислений является моделирование методом молекулярной динамики в биохимии и материаловедении. Возможности выполнения таких расчетов представлены в ряде продвинутых программных пакетов, например, LAMMPS и Gromacs. Тем не менее, несмотря на высокую производительность данных пакетов при использовании видеокарт существует проблема выбора межатомных потенциалов. В классической молекулярной динамике межатомные потенциалы выбираются на основании физически мотивированной математической формы с использованием параметров для адаптации потенциала под результаты квантово-химического или физического эксперимента. Разработка подобных потенциалов носит сложный и длительный характер.

С развитием машинного обучения (Machine Learning – ML) и, в частности, нейронных сетей (Neural Network – NN) стало возможным преодоление этого ограничения, переложив работу по поиску межатомного потенциала на вычислительную машину. На сегодняшний день представлены различные программные пакеты, использующие

данный подход, например, Amp [1], ænet [2], DeePMD-kit [3], PROPhet [4], SIMPLE-NN [5] и т.д.

В данной работе представлена программный пакет для работы с машинно-обучаемыми потенциалами на основе атомно-центрированных симметричных функций Бехлера-Парринелло, их обучением и использованием на графических процессорах NNMD [6] (Neural Networks for Molecular Dynamics – NNMD).

Статья организована следующим образом: в первой части представлены основные идеи использования машинно-обучаемых потенциалов для молекулярной динамики; во второй части дано описание дескрипторов – атомно-центрированных симметричных функций Бехлера-Парринелло; в третьей части рассмотрена модель нейронной сети высокой размерности; в четвертой части показаны результаты работы пакета для тестовой системы; в пятой части сделаны выводы об использовании машинно-обучаемых потенциалов и дальнейших путях развития программного пакета.

1. Описание метода

Машинно-обучаемые потенциалы – это регрессионная модель зависимости между конфигурацией атомов и полной энергией системы. В общем случае, такая модель может работать с фиксированным количеством атомов, потому что длина обучающего вектора в стандартной нейронной сети постоянна и не изменяется во времени. Для того, чтобы преодолеть данное ограничение, сеть сначала вычисляет потенциалы на атомах, а затем их суммирует, получая тем самым полную энергию системы. Локальное окружение каждого атома описывается специальным вектором-дескриптором, который является входным вектором для используемой модели. В данной работе используется нейронная сеть высокой размерности [7,8] (High-Dimensional Neural Network – HDNN) в качестве модели и атомно-центрированные симметричные функции [9] в качестве дескрипторов.

2. Дескрипторы: атомно-центрированные симметричные функции

Точность выбранной модели критически зависит от дескриптора, который описывает локальное окружение каждого атома в системе. Использование декартовых координат не подходит для подобного описания, так как перемещение и вращение группы атомов системы не приводит к изменению энергии, но приводит к изменению декартовых координат, что означает неинвариантность полученной энергии относительно операций перемещения и вращения.

Помимо корректного описания поверхностей потенциальной энергии симметричные функции должны выполнять несколько условий:

во-первых, функции и их первые производные должны удовлетворять условию непрерывности, поскольку они используются при вычислении сил на атомах; во-вторых, при увеличении межатомного расстояния эти функции должны стремиться к нулю для ускорения обучения ML-потенциалов.

Перед непосредственным описанием функций, удовлетворяющим вышеописанным условиям, введем специальную функцию – функцию отсечки f_c , которая задается формулой:

$$f_c = \begin{cases} \frac{1}{2} \cos\left(\frac{\pi R_{ij}}{R_c}\right), R_{ij} \leq R_c \\ 0, R_{ij} > R_c \end{cases} \quad (1)$$

где R_{ij} – расстояние между атомами i и j , R_c – радиус отсечки. Ключевым преимуществом использования функции заключается в устремлении симметричных функций и их производных к нулю в случае увеличения межатомного расстояния, тем самым исключая связи, которые не вносят существенного вклада в энергию, из вычислений.

В представленной работе используются симметричные функции Бехлера-Парринелло, описываемыми следующими формулами:

$$G_1^i = \sum_j f_c(R_{ij}) \quad (2)$$

$$G_2^i = \sum_j e^{-\eta(R_{ij}-R_s)} f_c(R_{ij}) \quad (3)$$

$$G_3^i = \sum_j \cos(kR_{ij}) f_c(R_{ij}) \quad (4)$$

$$G_4^i = 2^{1-\xi} \sum_{j,k \neq j} \left(1 + \lambda \cos \theta_{ijk}\right)^\xi e^{-\eta(R_{ij}^2 + R_{ik}^2 + R_{jk}^2)} f_c(R_{ij}) f(R_{ik}) f(R_{jk}) \quad (5)$$

$$G_5^i = 2^{1-\xi} \sum_{j,k \neq j} \left(1 + \lambda \cos \theta_{ijk}\right)^\xi e^{-\eta(R_{ij}^2 + R_{ik}^2)} f_c(R_{ij}) f(R_{ik}) \quad (6)$$

где i – индекс центрального атома, j и k – индексы соседних атомов, R_{ij} , R_{ik} , R_{jk} – расстояния между ними, θ_{ijk} – угол между векторами от i -го атома к j -му и от i -го атома к k -му, η , R_s , k , λ , ξ – параметры функций Бехлера-Парринелло.

Набор симметричных функций $G_i(R_i)$ (далее, G_i для удобства) описывает среду вокруг i -го атома, где R_i – координаты i -го атома.

3. Нейронные сети высокой размерности

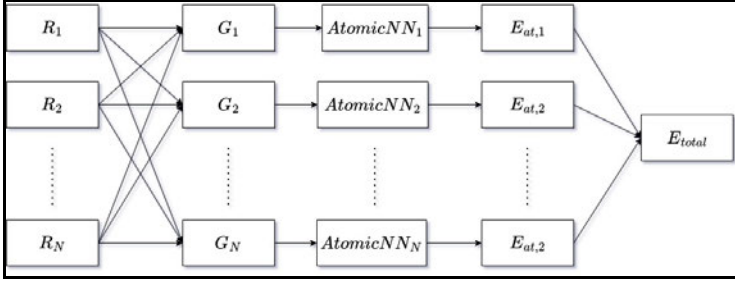


Рис. 1. Схема нейронной сети высокой размерности

Нейронные сети высокой размерности (HDNN) имеют структуру, показанную на рис. 1. Она состоит из атомных нейронных сетей (далее, AtomicNN для удобства) для каждого атома системы. В данной работе в качестве примера используются наборы из атомов одного элемента, однако, такая архитектура позволяет вычислять необходимые энергии и силы и для более сложных атомных систем.

AtomicNN использует вектора G в качестве входного и на его основе вычисляет энергию. По своей сути, AtomicNN – сеть прямого распространения с архитектурой многослойного перцептрона, в качестве функции активации для всех слоев, кроме выходного, используется сигмоидальная функция.

Значение полной энергии системы получается путем суммирования потенциалов на атомах:

$$E = \sum_{i=1}^{N_{at}} E_{at}(G_i) \quad (7)$$

где E_{at} – энергия i -го атома, вычисленная с помощью нейронной сети, N_{at} – это количество атомов системы.

После получения значений энергии происходит вычисление сил на атомах путем численного дифференцирования полной энергии по компонентам координат каждого атома:

$$F_{k,\alpha} = -\frac{\partial R}{\partial R_{k,\alpha}} = -\sum_{i=1}^{N_{at}} \sum_{j=1}^{N_s} \frac{\partial E_{at,i}}{\partial G_{i,s}} \frac{\partial G_{i,s}}{\partial R_{k,\alpha}} \quad (8)$$

где α – это x , y или z компонента соответствующих декартовых координат, $F_{k,\alpha}$ и $R_{k,\alpha}$ – α -компонента вектора силы и радиус-вектора атома k соответственно, N_s – это количество симметричных функций-компонент вектора G .

Функция потерь, которую требуется минимизировать, является суммой среднеквадратичных ошибок полных энергий E_i и сил на атомах F_{ij} , полученных с помощью теории функционала плотности и с помощью машинно-обучаемых потенциалов:

$$\Gamma = \frac{1}{M} \sum_{i=1}^M \left(\frac{E_i^{dfi} - E_i^{nm}}{N_i} \right)^2 + \frac{\mu}{3 \sum_{i=1}^M N_i} \sum_{i=1}^M \sum_{j=1}^{N_i} |F_{ij}^{dfi} - F_{ij}^{nm}| \quad (9)$$

$$\Gamma = RMSE^2(E) + \frac{\mu}{3} RMSE^2(F) \quad (10)$$

где M – это количество конфигураций атомов в обучающем наборе, N_i – это количество атомов в i -ой конфигурации, μ – параметр важности ошибки по силам относительно ошибки по энергии.

4. Программный пакет

В данном разделе приведены основные принципы разработки программного пакета NNMD (Neural Networks for Molecular Dynamics), а также примеры кода.

Большая часть программного пакета написана на языке программирования Python, наиболее затратные операции, например, вычисление сил на атомах и вычисление симметричных функций, написаны на C++ с применением технологии CUDA.

Для реализации нейронных сетей (HDNN и AtomicNN) используется open-source фреймворк PyTorch [9, 10], который дает возможность реализовывать различные алгоритмы машинного или глубокого обучения как для CPU, так и для GPU.

PyTorch позволяет реализовывать тензорные операции на C++ и создавать биндинги – связки между C++ и Python путем подключения специальной библиотеки Pybind11 [11], механизмы которого также присутствуют в дистрибутиве фреймворка для C++ – Libtorch. Такое решение приносит выгоду в скорости выполнения, при этом не ломая программную логику встроенных частей фреймворка.

NNMD разрабатывается как пакет с возможностью использования уже имеющихся обученных моделей для вычислительных задач MD симуляции. Рассмотрим процедуру обучения, реализованной в пакете и показанной в виде блок-схемы на рис. 2. Перед обучением формируется обучающий датасет, состоящий из значений симметричных функций для атомов системы, которые предварительно нормализуются, заданными параметрами функций Бехлера-Парринелло, декартовыми координатами атомов, а также целевых значений энергии и сил на атомах.

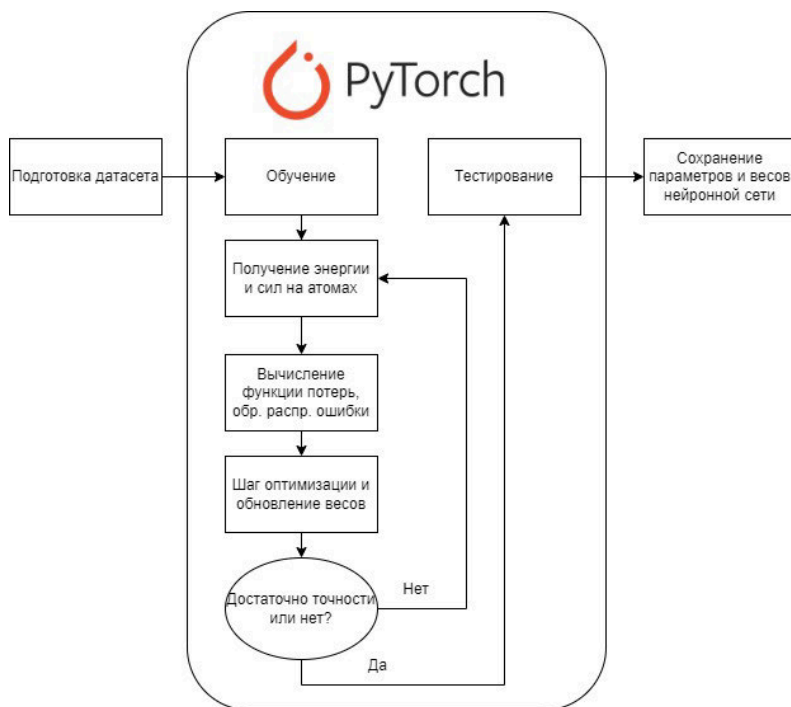


Рис. 2. Блок-схема работы пакета при обучении

Далее после получения подготовленного датасета начинается процесс обучения: HDNN будет оптимизироваться с помощью PyTorch, пока среднеквадратичная ошибка по энергии и по силам не будет минимальной или не стабилизируется. Параметры обученной нейронной сети сохраняются в специальные файлы, которые можно загрузить, чтобы использовать уже готовую модель в дальнейшем.

В качестве тестовых данных взяты результаты квантово-химического моделирования с помощью пакета GPAW для 45 атомов меди. Эпох обучения – 180, оптимизатор – Adaptive Momentum Estimation (Adam), параметры симметричных функций выбраны следующими: $\eta = 0.01$, $R_s = 0.5$, $k = 1$, $\lambda = -1$, $\xi = 3$, $R_c = 12 \text{ \AA}$. Обучение проводилось с использованием графического процессора NVIDIA GeForce RTX 3070. В результате получен следующий график функции потерь, представленный на рис. 3.

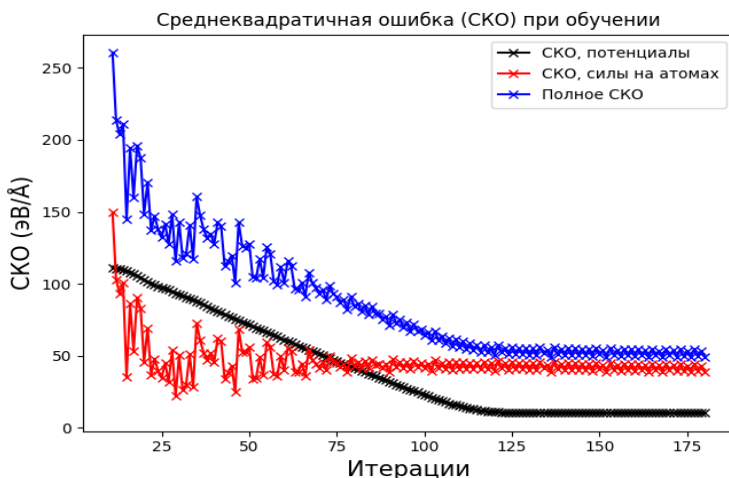


Рис. 3. График среднеквадратичной ошибки при обучении HDNN

Как видно из этого графика, потенциалы поначалу вносят основной вклад в ошибку, но постепенно ошибка по энергии уменьшается. Ошибка по силам колеблется, однако, после 120 эпох ошибка стабилизировалась по обеим величинам и существенно не изменялась. Это объясняется тем, что вычисление сил практически целиком зависит от энергий, полученных при помощи подсетей, а поскольку значения сил малы по сравнению со значениями энергий, то малейшее изменение энергий влечет изменение сил.

Заключение

В рамках данной работы был исследован и реализован метод получения машинно-обучаемых потенциалов, позволяющий проводить молекулярно-динамическое моделирование с достаточно большой точностью.

Был реализован программный пакет, который проводит вычисления с использованием GPU и технологии CUDA, получая в результате значительное ускорение.

В качестве дальнейшего направления работы может быть рассмотрена идея применения CUDA-aware MPI для получения большего ускорения работы.

Список литературы

1. Khorshidi A., Amp: A modular approach to machine learning in atomistic simulations / A. Khorshidi, A. A. Peterson // *Computer Physics Communications*. – 2016. – Vol. 207. – PP. 310-324.
2. Artrith N., An implementation of artificial neural-network potentials for atomistic materials simulations: Performance for TiO₂ / N. Artrith, A. Urban // *Computational Materials Science*. – 2016. – Vol. 114. – PP. 135-150.
3. Han W., DeePMD-kit: A deep learning package for many-body potential energy representation and molecular dynamics / W. Han, Z. Linfeng, H. Jiequn, E Weinan // *Computer Physics Communications*. – 2018. – Vol. 228. – PP. 178-184.
4. Kolb B., Discovering charge density functionals and structure-property relationships with PROPhet: A general framework for coupling machine learning and first-principles methods / B. Kolb, L. Lentz, A. Kolpak // *Scientific Reports*. – 2017. – Vol. 7.
5. Kyuhyun L., SIMPLE-NN: An efficient package for training and executing neural-network interatomic potentials / L. Kyuhyun, Y. Dongsun, J. Wonseok, H. Seungwu // *Computer Physics Communications*. – 2019. – Vol. 242. – PP. 95-103.
6. NNMD [Электронный ресурс]. Режим доступа: <https://github.com/ded-otshelnik/nnmd>
7. Behler J., Constructing High-Dimensional Neural Network Potentials: A Tutorial Review / J. Behler // *International Journal of Quantum Chemistry*. – 2015. – Vol. 115. – Issue 16.
8. Behler J., Generalized Neural-Network Representation of High-Dimensional Potential-Energy Surfaces / J. Behler, M. Parrinello // *Physical Review Letters*. – 2007. – Vol. 98. – Issue 14.
9. Behler J., Atom-centered symmetry functions for constructing high-dimensional neural network potentials / J. Behler // *The Journal of Chemical Physics*. – 2011. – Vol. 134. – Issue 3.
10. Paszke A. PyTorch: An Imperative Style, High-Performance Deep Learning Library. / A. Paszke, S. Gross, F. Massa [et al.] // 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.
11. PyTorch documentation [Электронный доступ]. – Режим доступа: <https://pytorch.org/docs/stable/index.html>
12. Pybind11 documentation [Электронный доступ]. – Режим доступа: <https://pybind11.readthedocs.io/en/stable/index.html>

Разработка информационной системы факультета компьютерных наук ФГБОУ ВО «Воронежский государственный университет»

С. И. Бутовецкая

Студент бакалавр

Е. А. Копытина

Старший преподаватель

Введение

В современном образовании информационные технологии играют значительную роль в оптимизации учебного процесса. Развитие цифровых технологий позволяет создавать информационные системы, которые улучшают качество образования и предоставляют студентам и преподавателям доступ к необходимой и актуальной информации, как показано в [1-6]. Данная статья описывает разработку информационной системы для факультета компьютерных наук ФГБОУ ВО «ВГУ», которая предоставляет доступ к информации о текущих оценках и посещаемости студентов. Разработка такой системы позволит студентам, преподавателям и администрации учебного заведения взаимодействовать более эффективно и оперативно.

1. Постановка задачи

К описываемой в статье информационной системе были выдвинуты следующие требования:

- Аутентификация пользователя.
- Регистрация пользователя с ролью студент, преподаватель или администратор.
- Возможность пользователя с ролью студента просматривать баллы за аттестации и итоговые оценки по дисциплинам текущего семестра.
- Возможность пользователя с ролью студента просматривать посещаемость дисциплин текущего семестра.
- Возможность пользователя с ролью преподавателя выставять баллы за аттестации и итоговые оценки группам студентов по дисциплинам текущего семестра.

2. Реализация логики

Основной код веб-приложения написан на объектно-ориентированном языке Java с использованием следующих технологий:

- Apache Maven – фреймворк для автоматической сборки проекта с описанием зависимостей в файле POM.
- PostgreSQL – объектно-реляционная система управления базами данных (СУБД).
- Spring – фреймворк для создания веб-приложений и Spring Boot – модуль Spring, который осуществляет автоматическую настройку веб-приложений и предоставляет встроенные HTTP-серверы (Tomcat). Spring Security – фреймворк для реализации регистрации и авторизации пользователей веб-приложения.
- Hibernate – библиотека для языка программирования Java, цель которой связать объектно-ориентированное программирование и реляционную базу данных, а также сопоставить типы данных Java с типами данных SQL. Hibernate является реализацией JPA.
- Thymeleaf – серверный механизм Java-шаблонов для веб-приложений, способный обрабатывать HTML.
- HTML – язык гипертекстовой разметки документов для веб-страниц. CSS – формальный язык для описания стиля документа HTML.

Разработанное веб-приложение имеет клиент-серверную архитектуру. Клиент и сервер взаимодействуют посредством обмена сообщений типа GET и POST протокола HTTP.

Проект собран Maven с нужными зависимостями для реализации логики приложения.

База данных department-db состоит из следующих главных сущностей: Student, Lecturer, Speciality, Subject, Marks. Соответствующие классы реализованы и в Java-проекте.

Для каждой сущности базы данных созданы JPA-репозитории – интерфейсы, которые можно определять для получения доступа к данным в нужном виде.

За авторизацию и регистрацию пользователей отвечают следующие классы проекта, которые находятся в папке department/main/java:

- User, реализация интерфейса UserDetails (нужно для взаимодействия со SpringSecurity),
- Role,
- UserService,
- UserRepository,
- RoleRepository,
- RegistrationController,

- SecurityController,
- WebSecurityConfiguration.

В основных модулях StudentMainController, LecturerMainController и AdminMainController были реализованы функции-запросы, взаимодействующие с клиентской частью приложения. За запуск веб-приложения отвечает модуль DepartmentApplication.

Вышеописанным образом реализована серверная часть веб-приложения.

Клиентская часть представлена следующими HTML файлами, содержащимися в папке department/main/resources/templates:

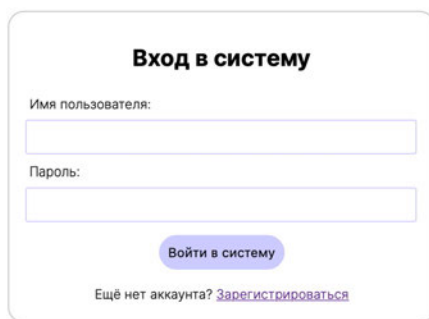
- login.html,
- registration.html.

Также были разработаны следующие представления пользовательского интерфейса, находящиеся в папке department/main/resources/templates:

- home.html,
- header.html,
- marks.html,
- attendance.html.

3. Реализация интерфейса

Работа с программой начинается с авторизации пользователя. Вид окна авторизации представлен на рис. 1.



The image shows a web form for user login. At the top, the title "Вход в систему" (Login) is centered. Below it, there are two input fields: "Имя пользователя:" (Username) and "Пароль:" (Password). A blue button labeled "Войти в систему" (Login) is positioned below the password field. At the bottom, there is a link that says "Ещё нет аккаунта? Зарегистрироваться" (Don't have an account? Register).

Рис. 1. Страница авторизации пользователя

Чтобы авторизоваться предварительно необходимо зарегистрироваться в информационной системе при помощи уникального логина. Роль пользователя и его уникальный логин устанавливаются администратором системы. Вид окна регистрации представлен на рис. 2.

Регистрация в системе

Имя пользователя:

16211062

Пароль:

Зарегистрироваться

Уже есть аккаунт? [Авторизоваться](#)

Рис. 2. Страница регистрации пользователя

После успешной авторизации пользователь с ролью STUDENT переходит на страницу /stud/main. Вид главной страницы студента представлен на рис. 3.

Домашняя страница

Оценки

Посещаемость

Выйти

Карточка студента

Номер студенческого билета	16211062
Фамилия	Бутовецкая
Имя	Софья
Отчество	Игоревна
Дата Рождения	2004-04-07
Курс	3
Группа	15
Номер специальности	09.03.03

Рис. 3. Главная страница студента

Для просмотра оценок за текущий семестр пользователю с ролью STUDENT нужно перейти на вкладку «Оценки». После чего откроется страница /stud/marks, на которой будет представлена таблица с данными о текущих баллах студента. Вид страницы «Свод оценок студента» представлен на рис. 4.

Свод оценок студента

Название дисциплины	ФИО преподавателя	Отчетность	Аттестация 1	Аттестация 2	Аттестация 3	Итоговая оценка
Проектирование баз данных	Соломатин Д. И.	Дифференцируемый зачет	25	46	34	4
Технология программирования	Тарасов В. С.	Зачет	35	31	50	4
Базы данных	Савченко С. В.	Дифференцируемый зачет	46	40	29	4
Компьютерная геометрия и геометрическое моделирование	Ермаков М. В.	Дифференцируемый зачет	50	33	35	4
Современные технологии программирования (Си)	Решетова О. О.	Экзамен	35	46	39	4
Лингвистические основы информатики	Гончаров Н. И.	Дифференцируемый зачет	24	48	41	2
Технологии и методы программирования	Кургалин С. Д.	Экзамен	33	33	36	3

Рис. 4. Вид страницы «Свод оценок студента»

Для того чтобы пользователь с ролью STUDENT мог просмотреть посещаемость за текущий семестр, нужно перейти на вкладку «Посещаемость». На странице /stud/attendance нужно выбрать дисциплину из списка и загрузится таблица с отметками посещаемости. Вид страницы «Посещаемость студента» представлен на рис. 5.

Посещаемость студента

Дисциплина:

Выберите дисциплину
⌵
Выбрать

Дата	Номер	ФИО преподавателя	Тип	Форма	Отметка посещаемости	Комментарий
2024-02-01	2	Соломатин Д. И.	Практическое занятие	О	true	-
2024-02-02	3	Соломатин Д. И.	Практическое занятие	О	true	-
2024-02-02	4	Соломатин Д. И.	Лекционное занятие	О	true	-

Рис. 5. Вид страницы «Посещаемость студента»

После успешной авторизации пользователь с ролью LECTURER переходит на страницу /lect/main. Вид главной страницы преподавателя представлен на рис. 6.

Карточка преподавателя

Фамилия	Соломатин
Имя	Дмитрий
Отчество	Иванович
Должность	Преподаватель

Рис. 6. Главная страница преподавателя

Для того чтобы выставить оценки группе студентов, пользователь с ролью LECTURER должен перейти на вкладку «Оценки». Откроется страница /lect/marks, на которой после выбора нужных параметров – дисциплина, курс, группа – откроется окно редактирования оценок. На рис. 7 представлены параметры для дальнейшего выставления оценок.

Выставить баллы студентам

Дисциплина:

Курс:

Группа:

Выбрать

Рис. 7. Вид параметров для выставления оценок

На рис. 8 представлен вид таблицы редактирования оценок студентов.

Заключение

В данной статье была рассмотрена разработка информационной системы факультета компьютерных наук ФГБОУ ВО «ВГУ». Все поставленные в начале разработки приложения функциональные требования были выполнены. Приложение является востребованным и полностью работоспособным.

Номер студ билета	ФИО студента	Аттестация 1	Аттестация 2	Аттестация 3	Итоговая оценка
16211062	Бутовецкая С. И.	25	46	50	5
16211074	Ермаков И. С.	41	42	30	4
16211065	Каплий В. В.	32	49	30	4
16211068	Маньков И. А.	46	24	22	2
16211069	Скогорева М. А.	29	27	28	3
16211070	Фомичев Д. В.	25	25	25	3
16211071	Харин А. В.	27	38	40	4
16211061	Братышев Т. Д.	40	42	38	4
16211066	Кожевников А. В.	30	25	27	3

Сохранить

Рис. 8. Вид окна редактирования оценок

Список литературы

1. Свидетельство о государственной регистрации программы для ЭВМ № 2023661037 Российская Федерация. Приложение для автоматизации документооборота дополнительного образования : № 2023618813 : заявл. 03.05.2023 : опубл. 25.05.2023 / А. С. Голышева, Е. А. Копытина ; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Воронежский государственный технический университет». – EDN NBUVMM.

2. Быков, М. Н. Разработка парсера для автоматизации работы с учебными планами факультета компьютерных наук Воронежского государственного университета / М. Н. Быков, Е. А. Копытина // Информатика: проблемы, методы, технологии : материалы XXIII Международной научно-практической конференции им. Э.К. Алгаинова, Воронеж, 15–17 февраля 2023 года. – Воронеж: Воронежский государственный университет, 2023. – С. 768-772. – EDN HSGNJG.

3. Свидетельство о государственной регистрации программы для ЭВМ № 2023617539 Российская Федерация. Программа расчета коэффициента облучения при лучистом теплообмене прямоугольного обогревателя с элементарной площадкой : № 2023616247 : заявл. 28.03.2023 : опубл. 11.04.2023 / С. В. Чуйкин, Е. А. Копытина, Е. С.

Аралов, С. О. Харин ; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Воронежский государственный технический университет». – EDN DLOVWK.

4. Погорелов, Р. И. Адаптация тиражируемого решения от фирмы 1С (1С: Управление нашей фирмой) и разработка приложения на мобильной платформе 1С / Р. И. Погорелов, А. В. Копытин, Е. А. Копытина // Труды молодых учёных факультета компьютерных наук ВГУ : Сборник статей / Под редакцией Д.Н. Борисова. Том Выпуск 1. – Воронеж : Общество с ограниченной ответственностью "Вэлборн", 2021. – С. 121-125. – EDN EOZHFN.

5. Претков, А. И. Разработка программного комплекса для автоматизации формирования документов основных профессиональных образовательных программ высшего образования факультета компьютерных наук Воронежского государственного университета / А. И. Претков, Е. А. Копытина // Информатика: проблемы, методы, технологии : материалы XXIII Международной научно-практической конференции им. Э.К. Алгаинова, Воронеж, 15–17 февраля 2023 года. – Воронеж: Воронежский государственный университет, 2023. – С. 907-912. – EDN ENWODO.

6. Копытина, Е. А. Дополнительное образование школьников в сфере информационных технологий на примере курса «Основы программирования на языке Python» Яндекс.Лицея / Е. А. Копытина, А. В. Копытин, Ю. В. Шишко // Информатика: проблемы, методы, технологии : Материалы XX Международной научно-методической конференции, Воронеж, 13–14 февраля 2020 года / Под редакцией А.А. Зацаринного, Д.Н. Борисова. – Воронеж: "Научно-исследовательские публикации" (ООО "Вэлборн"), 2020. – С. 1830-1834. – EDN ZIFIQY.

Разработка системы аутсорсинга краткосрочных работ на предприятиях

М. Н. Быков

Студент бакалавр

Е. А. Копытина

Старший преподаватель

Введение

Современное деловое окружение характеризуется постоянными изменениями и необходимостью быстрой адаптации к новым условиям. В связи с этим, компании все чаще обращаются к аутсорсингу краткосрочных работ для удовлетворения временных потребностей в персонале. Аутсорсинг краткосрочных работ – это процесс передачи выполнения определенных задач или проектов внешнему исполнителю на определенный период времени. Это может быть связано с различными задачами, начиная от IT-разработки и маркетинговых исследований до временного увеличения рабочей нагрузки в период пикового спроса, как показано в [1, 2].

В то же время, для работников становится все важнее иметь возможность находить кратковременные подработки с гибким графиком, соответствующим их индивидуальным потребностям. Это может быть особенно актуально для студентов, молодых родителей или фрилансеров, которым нужна дополнительная работа на определенный период времени.

Актуальность данной темы обусловлена необходимостью поиска новых форм организации труда в условиях быстро меняющейся экономической среды. [3-7]. Кроме того, данная работа имеет связь с другими научно-исследовательскими работами в области управления персоналом, организации труда и аутсорсинга. Изучение системы для аутсорсинга краткосрочных работ имеет большое значение для понимания тенденций развития рынка труда и создания более гибких и эффективных моделей работы как для компаний, так и для работников.

Целью данного исследования является разработка системы для аутсорсинга краткосрочных работ на предприятиях.

1. Постановка задачи

К описываемой в статье системе аутсорсинга краткосрочных работ были выдвинуты следующие функциональные требования:

Для работодателя:

- публикация объявлений о подработке;
- просмотр резюме потенциальных работников.

Для работника:

- просмотр вакансий по отобранным категориям и фильтрам;
- публикация резюме с описанием навыков и пожеланий к работе;
- отклик на подходящие объявления.

Для администратора:

- верификация информации о компаниях и предприятиях;
- модерация недопустимого контента в резюме и вакансиях.

2. Реализация логики

Приложение написано на языке Java с использованием фреймворка Spring Boot. Данные, необходимые для корректной работы приложения хранятся в базе данных PostgreSQL, взаимодействие с сервером осуществляется посредством модуля Spring JPA.

Архитектура приложения REST API следует принципам MVC (Model-View-Controller) и включает в себя следующие компоненты:

- контроллеры (Controllers): отвечают за обработку HTTP-запросов и возвращение HTTP-ответов. Они содержат методы, аннотированные аннотациями @RequestMapping или @GetMapping, @PostMapping, @PutMapping, @DeleteMapping и другими, для обработки различных типов запросов.

Контроллеры взаимодействуют с сервисами для выполнения бизнес-логики и с репозиториями для доступа к данным.

- сервисы (Services): содержат бизнес-логику приложения. Они выполняют операции над данными, обрабатывают бизнес-правила и могут взаимодействовать с репозиториями для доступа к данным. Сервисы могут быть инъектированы в контроллеры и другие сервисы.
- репозитории (Repositories): отвечают за доступ к базе данных или другому источнику данных. содержат методы для выполнения операций CRUD (Create, Read, Update, Delete) над данными. Репозитории могут использовать JPA или другие технологии для взаимодействия с базой данных.
- мапперы (Mappers): Мапперы отвечают за преобразование данных между различными моделями внутри приложения. Они могут использоваться для преобразования сущностей JPA в DTO

(Data Transfer Object) или наоборот. Реализован с использованием библиотеки MapStruct.

Такая архитектура позволяет разделить ответственности между различными компонентами приложения, обеспечивая лучшую читаемость, поддержку и масштабируемость кода.

Начало работы с приложением осуществляется с запуском сервера.

После этого необходимо авторизоваться под учётной записью работодателя или работника, либо создать новый аккаунт. Авторизация реализована с использованием модуля Spring Boot и технологии JWT (JSON Web Token). Авторизация устроена следующим образом:

- пользователь отправляет запрос на аутентификацию, предоставляя свои учетные данные (например, логин и пароль);
- сервер проверяет предоставленные учетные данные и, если они верны, создает JWT токен. Токен содержит информацию об аутентифицированном пользователе и может включать дополнительные данные, такие как срок действия;
- сервер отправляет JWT токен обратно пользователю;
- пользователь включает JWT токен в заголовок каждого последующего запроса к защищенным ресурсам приложения;
- сервер проверяет действительность JWT токена при получении запроса. Если токен действителен, сервер разрешает доступ к защищенным ресурсам;
- если JWT токен истек или недействителен, сервер отклоняет запрос пользователя.

После авторизации в роли «Работодатель» пользователь получает доступ к модулю «Модерация вакансий», в котором он может:

- создать новую вакансию с помощью метода `addVacancy()`;
- отобразить список созданных вакансий с помощью метода `getVacancy()`;
- обновить содержимое выбранной вакансии с помощью метода `updateVacancy()`;
- просмотреть отклики на выбранную вакансию с помощью метода `displayResumeByVacancy()`;
- одобрить отклик соискателя и закрыть вакансию с помощью метода `confirmResumeByVacancy()`;
- закрыть вакансию без отклика соискателя с помощью функции `removeVacancy()`.

Для того, чтобы открыть данный модуль, пользователю с ролью «Работодатель» необходимо зарегистрировать компанию с помощью метода `registerCompany()`, в котором необходимо указать все необходимые реквизиты (ИНН, КПП, адрес) и документы. После

исполнения метода необходимо дождаться верификации компании пользователем с ролью «Администратор».

После авторизации в роли «Соискатель работы» пользователь получает доступ к модулям «Конструктор резюме» и «Просмотр вакансий». В модуле «Конструктор резюме» можно:

1. создать новое резюме с помощью метода `addResume()`;
2. отобразить список созданных резюме с помощью метода `getResume()`;
3. обновить содержимое выбранного резюме с помощью метода `updateResume()`;
4. удалить неактуальное резюме с помощью функции `removeResume()`.

В модуле «Просмотр вакансий» пользователю доступен выбор вакансий в выбранном пользователем городе, либо онлайн по всему часовому поясу. Доступен поиск как подработок на месте, так и удалённых подработок, реализованы фильтрация и пагинация. После выбора интересующей вакансии пользователь может отправить на неё отклик, выбрав подходящее резюме.

При входе на учётную запись администратора, пользователю открывается доступ к модерации существующих в системе компаний, вакансий и резюме. Это необходимо для соблюдения пользователями приложения правил платформы, и отсутствия на ней запрещённого контента.

Взаимодействие с базой данных PostgreSQL происходит с помощью Java Persistence API. Настройка параметров подключения к базе данных в файле `application.properties`. Для выполнения операций с базой данных, таких как сохранение, поиск, удаление и обновление, используются репозитории (`Repository`). Контроль версий базы данных происходит с помощью библиотеки `Flyway`.

Для хранения необходимых данных была создана база данных «`jobilee`», в которой были созданы 4 таблицы:

1. «`company`» – хранение сведений о компаниях;
2. «`resume`» – хранение сведений о резюме;
3. «`vacancy`» – хранение сведений о вакансиях;
4. «`user`» – хранение сведений о пользователях;
5. «`role`» – хранение сведений о ролях пользователей («Администратор», «Работодатель», «Соискатель работы»).

3. Реализация интерфейса

Работа с приложением начинается с авторизации пользователя. Вид окна входа в приложение представлен на рис. 1.

Jobilee

Добро пожаловать!
Введите данные для входа

Электронная почта

Пароль

Вход

Регистрация

Рис. 1. Страница авторизации пользователя

Чтобы авторизоваться в приложении, необходимо зарегистрировать соответствующую учётную запись. Вид окна регистрации представлен на рис. 2.

Jobilee

Добро пожаловать!
Введите данные для входа

Имя

Фамилия

Электронная почта

Пароль

Повторите пароль

Регистрация

Вход

Рис. 2. Страница регистрации пользователя

После успешной авторизации пользователь переходит на главную страницу. Вид главной страницы представлен на рис. 3.

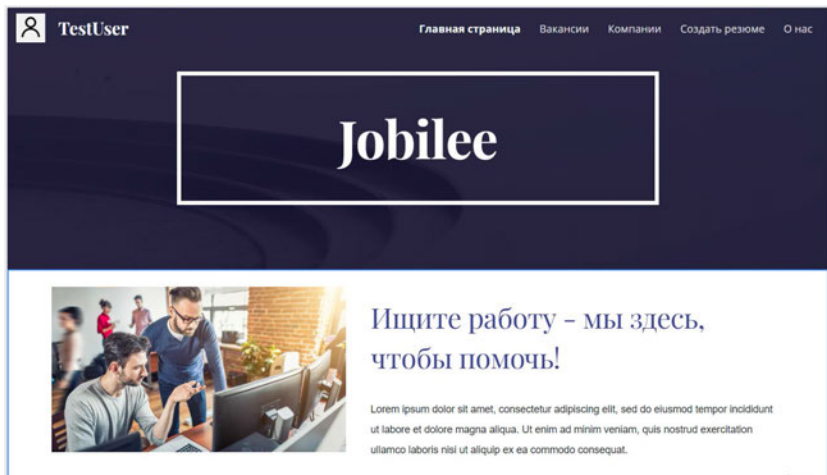


Рис. 3. Главная страница приложения

В данный момент приложение обладает пятью разделами: «Главная страница», выводящая ознакомительную информацию о приложении, раздел «Вакансии», отображающий вакансии доступные пользователю, представленный на рис. 4:

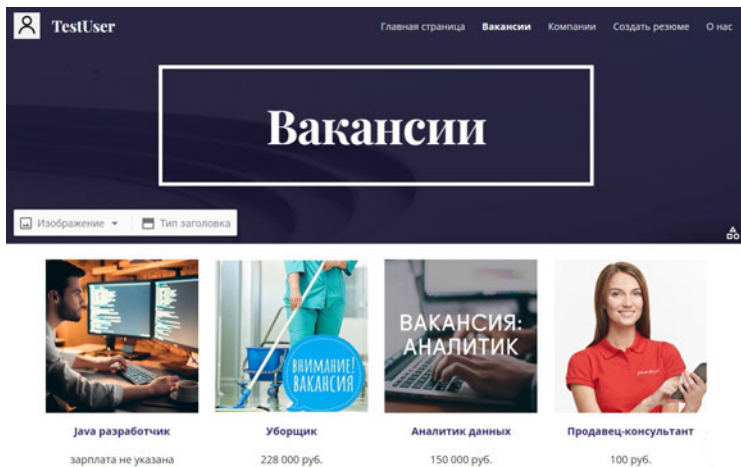


Рис. 4. Страница «Вакансии» приложения

Страница «Компании», на которой пользователь может ознакомиться с компаниями, сотрудничающими с приложениями (рис. 5).

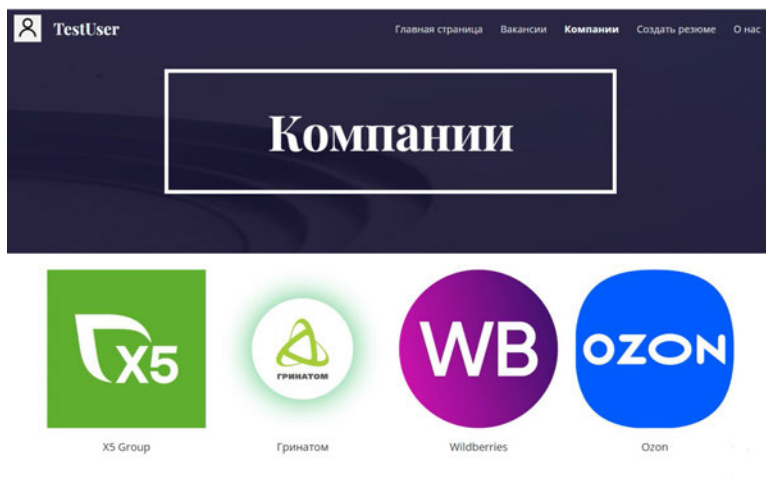


Рис. 5. Страница «Компании» приложения

Страница создания резюме, на которой пользователь, ищущий работу, может опубликовать новое резюме (рис. 6).

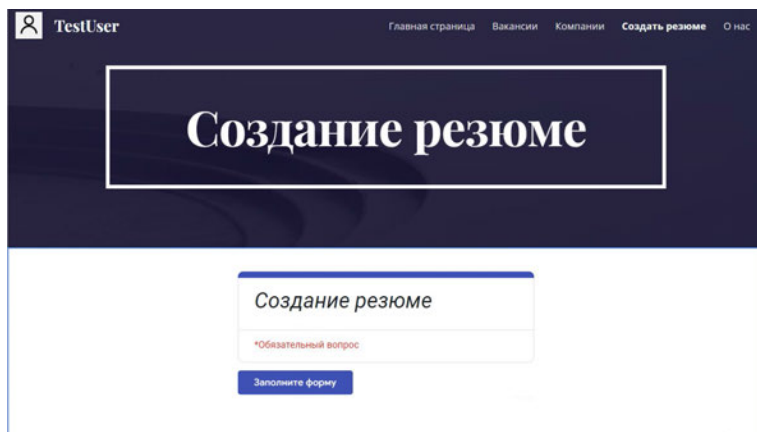


Рис. 6. Страница «Создание резюме» приложения

Страница «О нас» (рис. 7), на которой расположена основная информация о приложении, а также контактная информация создателей приложения.

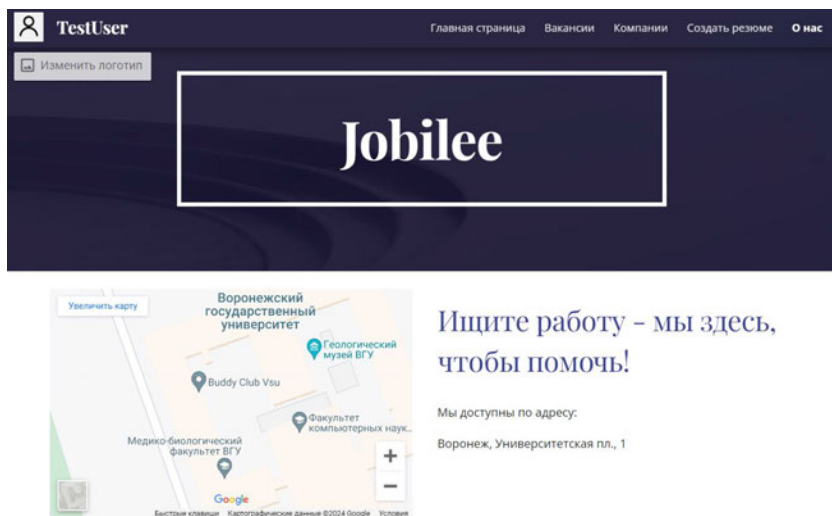


Рис. 7. Страница «О нас» приложения

Заключение

В данной статье была рассмотрена реализация системы аутсорсинга краткосрочных работ на предприятиях. На данный момент приложение обладает базовой функциональностью (авторизация, создание вакансии, создание резюме, отклик на вакансию). В дальнейшем в процессе развития проекта планируется добавить дополнительные возможности для более качественного пользовательского опыта (например, отображение вакансий на интерактивной карте).

Список литературы

1. Немешаев, А. А. Специфика применения аутсорсинга и ИТ-аутсорсинга как инструмента повышения эффективности деятельности / А. А. Немешаев, Г. В. Исмагилова // Весенние дни науки : Сборник докладов международной конференции студентов и молодых ученых, Екатеринбург, 20–22 апреля 2023 года. – Екатеринбург: ООО Издательский Дом "Ажур", 2023. – С. 177-180. – EDN SRDVJN.
2. Заяц, А. А. Механизм аутсорсинга. Аутсорсинг логистических функций и бизнес-процессов / А. А. Заяц, Е. Ю. Козлов, Н. А. Вакулич //

Логистика – евразийский мост : Материалы XIII Международной научно-практической конференции, Красноярск, 25–29 апреля 2018 года. Том Часть 2. – Красноярск: Красноярский государственный аграрный университет, 2018. – С. 89-93. – EDN XYXFED.

3. Манукян, С. Ш. Автоматизация взаимоотношений с клиентами в SEO- компании / С. Ш. Манукян, Е. А. Копытина // Труды молодых ученых факультета компьютерных наук ВГУ. Том Выпуск 2. – Воронеж : ООО «ВЭЛБОРН», 2022. – С. 108-113. – EDN PUNHZU.

4. Землянухин, Д. В. Разработка системы контроля и управления параметрами производственных процессов с GSM-каналом телеметрии / Д. В. Землянухин, Е. А. Копытина, А. А. Головкин // Сборник студенческих научных работ факультета компьютерных наук ВГУ : Сборник статей. В 2-х частях / Под редакцией Д.Н. Борисова. Том Выпуск 13. Часть 1. – Воронеж : Воронежский государственный университет, 2019. – С. 78-84. – EDN ACDOWK.

5. Карпов, В. Д. Разработка мобильного приложения "Ruparking" / В. Д. Карпов, А. В. Копытин, Е. А. Копытина // Сборник студенческих научных работ факультета компьютерных наук ВГУ : Сборник статей. В 2-х частях / Под редакцией Д.Н. Борисова. Том Выпуск 13. Часть 1. – Воронеж : Воронежский государственный университет, 2019. – С. 92-97. – EDN ETMHBV.

6. Другалев, А. И. Разработка модуля постановки и контроля задач на платформе 1С: Предприятие 8.3 с применением технологии SMART / А. И. Другалев, Е. А. Копытина // Информатика: проблемы, методология, технологии : материалы XVI международной научно-методической конференции, Воронеж, 11–12 февраля 2016 года. Том Секции 5-6. – Воронеж: Научно-исследовательские публикации, 2016. – С. 74-79. – EDN WNTMJ.

7. Копытина, Е. А. Разработка мобильного приложения на платформе 1С Предприятие 8.3 / Е. А. Копытина, А. И. Другалев // Сборник студенческих научных работ факультета компьютерных наук ВГУ / Воронежский государственный университет; Под редакцией Д.Н. Борисова. Том Выпуск 9. – Воронеж : Издательский дом ВГУ, 2015. – С. 115-120. – EDN ULMHGD.

Сравнительный анализ эффективности SAST/DAST решений в CI пайплайнах

И. А. Воробцов

Студент бакалавр

М. В. Ермаков

Старший преподаватель

Введение

Существующая на данный момент методология разработки безопасного ПО закреплена в ГОСТ Р 56939-2016 [1] и подразумевает наличие таких этапов, как статическое (SAST) и динамическое (DAST) тестирование безопасности приложений, в связи с чем, применение инструментов, основанных на этих методах, является стандартом. Более того, используемая повсеместно методология непрерывной интеграции и доставки (CI/CD) [2], позволяет всецело внедрять большинство инструментов SAST и DAST непосредственно в конвейер разработки и автоматизировать сканирование продуктов на уязвимости.

Перед разработчиками всё чаще встает вопрос о том, какой из методов анализа безопасности применить и какие программные компоненты для этого выбрать. Недостаточное понимание различий между этими подходами и их применимостью в контексте CI/CD может привести к неправильному выбору инструментов безопасности и их комбинаций, что в свою очередь способствует увеличению риска возникновения уязвимостей в разрабатываемом программном обеспечении.

В данной работе рассматривается задача анализа эффективности существующих SAST и DAST инструментов в контексте непрерывной интеграции и доставки.

1. Методология проводимого исследования

Методология проведения исследования основывается на выборе OWASP Benchmark в качестве стандарта для оценки точности сканеров уязвимостей. Этот выбор обоснован значительными различиями в производительности между доступными эталонами [3].

OWASP Benchmark – полностью функциональное открытое веб-приложение, написанное на широко используемом языке

программирования. Оно предоставляет стандартизированный подход к формированию результатов на основе следующих объективных метрик:

True Positive (TP): инструмент правильно обнаруживает реальную уязвимость.

False Negative (FN): инструмент не обнаруживает реальную уязвимость.

True Negative (TN): инструмент правильно игнорирует ложную уязвимость.

False Positive (FP): инструмент ошибочно считает ложную уязвимость реальной.

True Positive Rate (TPR): доля верно обнаруженных уязвимостей в приложении. Вычисление происходит согласно следующей формуле:

$$TPR = \frac{TP(\text{True Positive})}{TP(\text{True Positive}) + FN(\text{False Negative})} \quad (1)$$

False Positive Rate (FPR): доля ложных срабатываний сканера. Вычисляется согласно формуле:

$$FPR = \frac{FP(\text{False Positive})}{FP(\text{False Positive}) + TN(\text{True Negative})} \quad (2)$$

Для вычисления индивидуальной оценки OWASP Benchmark использует индекс Юдена, который рассчитывается путем вычитания единицы из общего числа специфичности и чувствительности теста. Чувствительность (Sensitivity) равна True Positive Rate (TPR), а специфичность (Specificity) равна единице минус False Positive Rate (FPR). Более высокое значение индекса Юдена указывает на хорошую производительность сканера.

$$\text{Yoden Index} = (\text{Sensitivity} + \text{Specificity}) - 1 \quad (3)$$

Исходя из этого, оценка эффективности каждого отдельного инструмента сводится к верному или ложному выявлению им какой-либо уязвимости.

2. Выбор тестируемых инструментов безопасности

Для проведения тестирования безопасности были выбраны следующие инструменты с открытым исходным кодом:

- SAST: SonarQube, Semgrep и ShiftLeft Scan;
- DAST: OWASP ZAP, Arachni и Wapiti.

Выбор обоснован прежде всего моделью их распространения, заявленной эффективностью и необходимым функционалом для их успешной интеграции и использования.

3. Формирование тестовой среды

Для проведения сравнительного анализа эффективности SAST и DAST решений была создана специализированная тестовая среда, включающая в себя GitLab CI для хранения репозитория с кодом OWASP Benchmark и кластер Kubernetes с развёрнутым GitLab Runner для выполнения CI/CD задач.

Взаимодействие между компонентами осуществляется через основную GitLab CI пайплайн, который запускает дочерние пайплайны с инструментами тестирования безопасности. Динамическое тестирование (DAST) проводится на контейнере с работающим приложением OWASP Benchmark, а статическое (SAST) – непосредственно на исходном коде в репозитории GitLab.

Результаты сканирования сохраняются в GitLab как артефакты пайплайна для последующего анализа эффективности и точности каждого инструмента.

Подробная схема интеграции описанного процесса представлена на рисунке ниже.

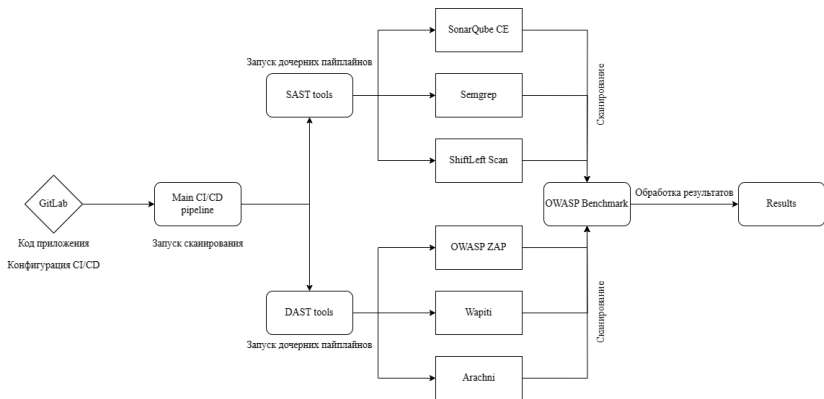


Рис. 1. Схема интеграции

4. Результаты исследования SAST инструментов

После применения инструментов статического тестирования безопасности на тестовых сценариях OWASP Benchmark были получены следующие результаты.

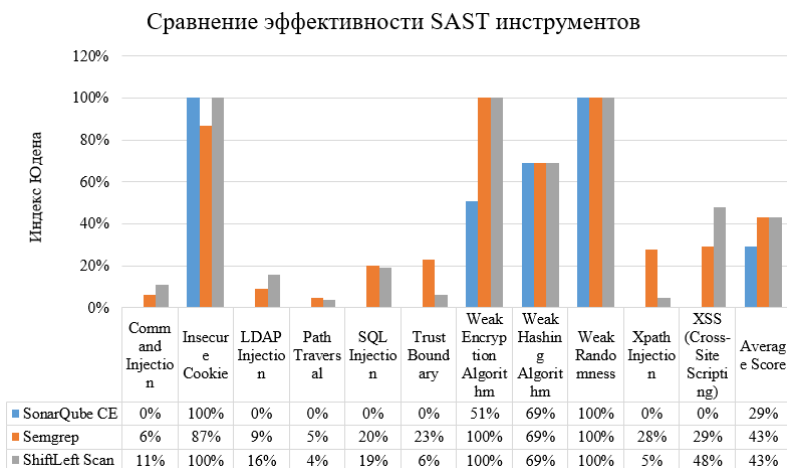


Рис. 2. Сравнение эффективности SAST инструментов по категориям уязвимостей

На основе приведённой выше диаграммы, можно провести полноценное сравнение выбранных инструментов SAST. Согласно ей, все анализаторы справляются с распознаванием слабой случайности, Insecure Cookie и слабых алгоритмов хэширования.

В совокупности – SonarQube CE показал худший результат среди всех, но уровень ложных срабатываний заметно ниже других инструментов, что явно можно отнести к сильным его сторонам.

Semgrep и ShiftLeft Scan показали примерно одинаковый уровень точности, однако высокая доля ложных срабатываний заставляет задуматься о целесообразности их использования.

Эффективность распознавания различных категорий уязвимостей SAST инструментами зависит от их уникальных характеристик и особенностей. Некоторые категории, такие как Insecure Cookie, Weak Encryption Algorithm и Weak Randomness часто имеют характерные признаки в коде и легко обнаруживаются поиском по шаблонам. В то время как другие, к примеру, Command Injection и SQL Injection сложны для распознавания из-за их контекстно-зависимой природы кода и требуют более сложных и ресурсозатратных методов анализа (к примеру Taint-анализ, который отслеживает, откуда данные приходят, как они распространяются по приложению и куда попадают).

5. Результаты исследования DAST инструментов

После применения инструментов динамического тестирования безопасности на тестовых сценариях OWASP Benchmark были получены следующие результаты. На графике представлены только те категории уязвимостей, которые хотя бы один DAST-сканер смог распознать успешно.

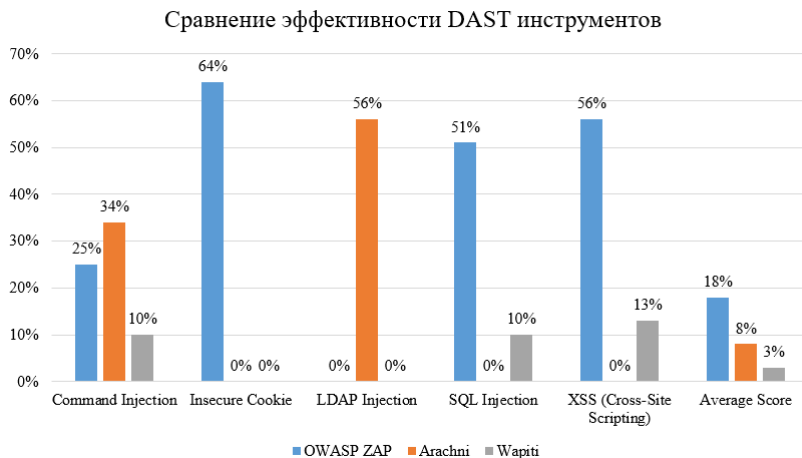


Рис. 3. Сравнение эффективности DAST инструментов по категориям уязвимостей

Исходя из представленных результатов, можно провести полноценное сравнение выбранных инструментов DAST. Согласно графику, все инструменты показывают разную эффективность в обнаружении тех или иных категорий уязвимостей.

OWASP ZAP демонстрирует хороший уровень точности в целом и является лучшим кандидатом для использования. Особенно хорошо показывает себя в категориях Insecure Cookie и XSS (межсайтовый скриптинг), где его показатели составляют 64% и 56% соответственно.

Arachni имеет неплохую точность в категориях Command Injection и LDAP Injection, с показателями 34% и 56% соответственно. Однако имеет низкие показатели в других категориях, что снижает его общую эффективность.

Wapiti демонстрирует наименьшую производительность среди всех инструментов. Общая производительность оценивается на 3%, что свидетельствует о его ограниченных возможностях в обнаружении уязвимостей.

Различия в эффективности распознавания тех или иных категорий уязвимостей DAST инструментами обусловлены спецификой их работы. Уязвимости, такие как Insecure Cookie часто легче обнаружить, потому что они проявляются в сетевом взаимодействии между клиентом и сервером или в содержимом веб-страниц. С другой стороны, уязвимости, связанные с слабыми алгоритмами хэширования, могут требовать анализа именно программного кода, что для DAST анализаторов является непосильной задачей.

Заключение

В ходе исследования был проведён сравнительный анализ эффективности инструментов SAST и DAST в контексте CI-пайплайнов. Результаты показали, что оба подхода имеют свои преимущества и недостатки, и наиболее эффективным является использование обоих методов тестирования вместе.

Одним из ключевых выводов исследования является то, что ни один сканер не может быть рассмотрен как универсальное средство для обнаружения всех видов уязвимостей в веб-приложениях. Результаты демонстрируют, что сканеры проявляют разную эффективность с различными категориями уязвимостей.

Для обеспечения практичности и эффективности тестирования безопасности в пайплайнах CI/CD, выявлены следующие рекомендации:

- при интеграции инструментов динамического сканирования безопасности в CI-пайплайны следует учитывать необходимость оптимизации времени его выполнения;
- развертывание сканеров на отдельных машинах с достаточным количеством ресурсов и управление ими с помощью пайплайнов позволит оптимизировать процесс сканирования и уменьшить его воздействие на производительность приложения;
- использование как статических (SAST), так и динамических (DAST) методов тестирования вместе позволит выявить больше уязвимостей и обеспечить более высокий уровень безопасности приложений;
- параллельное выполнение задач в пайплайне ускоряет его выполнение, включая тестирование безопасности на разных уровнях абстракции;
- тестирование каждого коммита позволяет раньше выявлять уязвимости и быстрее исправлять ошибки в сборках;
- сборка только необходимых компонентов способно сократить общее время выполнения пайплайна;
- подробная отчетность о найденных уязвимостях способно обеспечить их быстрое обнаружение и устранение.

Список использованных источников

1. ГОСТ Р 56939-2016. Защита информации. Разработка безопасного программного обеспечения. Общие требования [Текст]. – Введ. 2017-06-01. – М. : Стандартиформ, 2016. – IV, 19 с. : ил. ; 29 см.
2. Continuous Integration and Continuous Delivery Pipeline Automation for Agile Software Project Management [Электронный ресурс]. – Режим доступа : https://www.researchgate.net/publication/326406017_Continuous_Integration_and_Continuous_Delivery_Pipeline_Automation_for_Agile_Software_Project_Management
3. Evaluation of Web Vulnerability Scanners Based on OWASP Benchmark [Электронный ресурс]. – Режим доступа : https://www.researchgate.net/publication/331038733_Evaluation_of_Web_Vulnerability_Scanners_Based_on_OWASP_Benchmark

Автоматический контроллер температуры и влажности для домашних питомцев на базе Arduino

А. М. Гурьева

Студент бакалавр

С.А. Зуев

Доцент

Введение

Во все времена люди заводили домашних питомцев. Любой ответственный хозяин стремится создать условия, наиболее благоприятные для своего животного. Устройства на базе Arduino – популярный способ автоматизации действий, регулярно повторяющихся в повседневной жизни. Благодаря широкому функционалу, малому энергопотреблению, наличию множества модулей и деталей для Arduino и большому количеству теоретических материалов в Сети, становится легко выполнимой задача по созданию системы автоматического контроля температуры и влажности, которая поможет поддерживать оптимальные условия для содержания любого животного. В этой работе будет рассмотрено применение микропроцессора, датчика температуры и влажности и вспомогательных элементов для создания удобного и эффективного устройства по контролю параметров окружающей среды.

1. Цель работы

Целью работы является создание и исследование на практике функционала модели автоматического контроллера температуры и влажности для домашних питомцев.

К разрабатываемой модели предъявляются следующие требования:

- Точное измерение температуры и влажности с допустимой погрешностью 5% для влажности, 2 °С для температуры, обусловленной аппаратными особенностями датчика температуры и влажности [1];
- Наличие простого, интуитивно понятного пользователю интерфейса для контроля данных;
- Система оповещения пользователя о выходе температуры и влажности за рамки заданных диапазонов при помощи светового и звукового уведомления;

- Возможность изменения режима работы устройства для разных животных, задания пользовательского режима, перевода устройства в беззвучный режим.

2. Средства реализации

Аппаратное обеспечение:

- Микропроцессор Arduino Nano 3.0
- Датчик температуры и влажности DTH11
- LED-светодиод F5 GS MIN SL2 красного цвета
- Модуль зуммера Arduino

Программное обеспечение:

- Arduino IDE 2.3.2
- IDE IntelliJ IDEA Ultimate 2023.3.2

3. Аппаратная часть

Работа с аппаратным обеспечением производилась на беспаячной макетной плате для большей наглядности результата. К макетной плате подключаются микропроцессор, модуль зуммера. Светодиод подключается через токоограничивающий резистор 220 Ом, датчик DTH11 – через подтягивающий резистор 10 кОм [1].

Схема соединения элементов представлена на рис. 1.

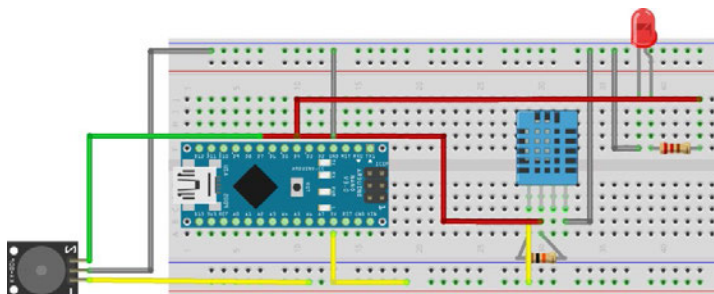


Рис. 1. Схема соединения элементов

Собранное устройство представлено на рис. 2.

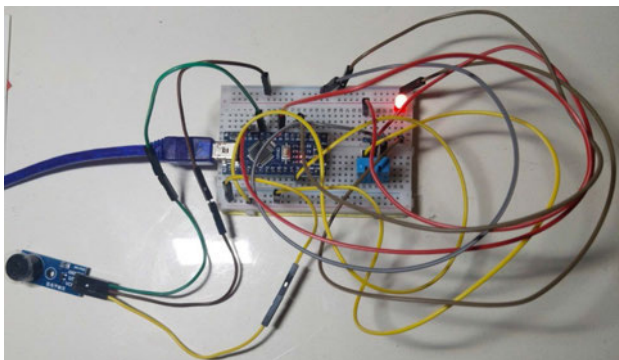


Рис. 2. Контроллер температуры и влажности

4. Программная часть

Для программирования устройства написана программа на языке Arduino C, который представляет собой C++ с фреймворком Wiring [2].

Создано приложение, в котором пользователи могут отслеживать значения температуры и влажности, выбирать диапазоны значений в зависимости от нужд животного, включать и отключать беззвучный режим. Для написания приложения был выбран язык программирования Java, являющийся наиболее удобным вариантом благодаря включению в Java Development Kit библиотеки для создания графического интерфейса Swing и наличию пользовательских библиотек для работы с портом, таких как jSerialComm.

Со стороны Arduino прописана логика отправки данных, работы с данным из Java и с элементами устройства. Сначала идет обработка данных о включении или выключении беззвучного режима и диапазоне влажности и температур, которые передаются из Java в виде строки в байтовом виде, содержащей значения, разделенными пробелами. Если в силу какой-либо ошибки данные не передаются (например, если приложение не запущено), светодиод мигает с задержкой 0,5 секунды. В ином случае программа выполняется дальше и считывает значения с датчика DHT11.

Arduino проверяет принадлежность каждого значения указанному для него диапазону. Когда хотя бы одно из них не удовлетворяет рекомендуемым условиям, на пин светодиода подается значение HIGH, и светодиод горит, пока условия не придут в норму, а если беззвучный режим выключен, то, кроме того, на пин зуммера подается сигнал !HIGH, при котором зуммер издает звук в силу особенностей конструкции модуля, в течение 1 секунды. В противном случае звуковой

сигнал не будет подан. Для правильной работы приложения на Java данные о реальной температуре и влажности каждую секунду передаются в последовательный порт. Также в коде предусмотрена подача сигнала HIGH для включения встроенного в микропроцессор светодиода при активации бесшумного режима. Это позволит пользователю убедиться в том, что устройство настроено правильно.

В программе используется функция `delay()` для мигания светодиодом и запуска зуммера на 0,5 секунд. Несмотря на то, что она приостанавливает выполнение кода на указанное время [3], функция `delay()` является оптимальным вариантом, ведь она не перегружает код и не вызывает ошибок из-за переполнения переменных, что происходит в методах с функцией `milis()` [3] и позволяет зажигать светодиод и включать зуммер нужно через произвольное, зависящее от окружающей среды и пользователя количество времени.

В приложении на Java прописана логика пользовательского интерфейса и работы с данными порта и отправки данных.

Связь приложения на Java и устройства на Arduino обеспечивается библиотекой `JSerialComm`. Средствами библиотеки определяется и прослушивается порт, в который подключается контроллер.

Данные в порт из Arduino отправляются в формате «`h t`», где `h` и `t` – считанные с датчика и разделенные пробелом значения влажности и температуры соответственно. Приложение получает данные из порта в виде массива байтов [4], которые далее переводятся в строковый тип, обрабатываются и вносятся в массив двух целочисленных значений. Эти целочисленные значения выводятся в интерфейсе приложения в колонке с настоящими значениями температуры и влажности.

Из приложения в порт отправляется переведенная в биты строка [4] вида «`s h N t T`», где `s` – код включения или выключения беззвучного режима, `h`, `t` – нижние границы, `N`, `T` – верхние границы значений влажности и температуры соответственно. Передача в порт каждого диапазона помогает устройству быстро реагировать на переключение режима и правильно реагировать на параметры окружающей среды.

В приложении есть код для обработки бесшумного режима, слайдеры для ввода времени работы этого режима и флажок «Постоянный бесшумный режим».

Передача данных через порт должна производиться постоянно во время работы приложения. Для этого используется класс `Timer` и `TimerTask()`. В теле метода `run()` выполняется проверка на включение или выключение беззвучного режима и соответствие температуры и влажности заданному режиму, вывод состояния среды в строку состояния, отправка и получение данных. Если установить соединение

через порт не удалось, например, если приложение было запущено перед подключением контроллера, в методе run() совершается попытка соединения. Действия производятся каждую секунду, как и вывод значений температуры и влажности в Arduino.

Необходимость проверять оптимальность реальных параметров на стороне Java и на стороне Arduino обусловлена тем, что передача данных через порт не всегда является стабильной. Иногда из-за особенностей работы таймера и функции loop() нарушается синхронизация между выводом данных из Arduino и их считыванием в Java. В такие моменты в приложение могут быть переданы заведомо неправильные данные. Для уменьшения ошибок нужно передавать минимальное количество параметров от Arduino в Java. Чтобы защититься от подобных сбоев, значения температуры и влажности каждую секунду сохраняются в целочисленный массив, и, если данные из порта заведомо ошибочны, в приложение выводятся данные, полученные секунду назад.

Вид приложения представлен на рис. 3.

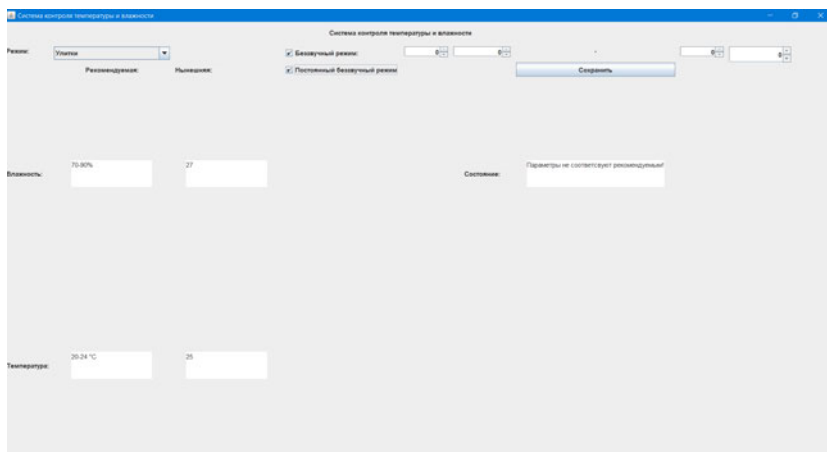


Рис. 3. Приложение для системы контроля температуры и влажности

Для удобства пользователей в приложении есть 4 предустановленных режима для самых распространенных питомцев, требующих особых условий: улитки, хомяки и морские свинки, птицы, ящерицы.

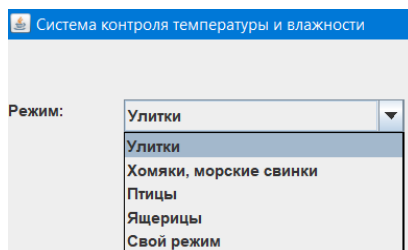


Рис. 4. Выбор режима работы

Пользователь может задать собственный режим, выбрать границы температуры и влажности из допустимого диапазона.

Приложение защищено от ошибок при вводе собственного режима, таких как ввод нечисловых значений или диапазонов, в которых значение нижней границы больше или равно значению верхней.

Ознакомиться с кодом на Arduino и Java можно по ссылке [5].

5. Практическое применение

Чтобы убедиться в точности измерений и в том, что погрешность не превышает указанных в документации датчика DTH11 значений [1], нужно сравнить значения, получаемые датчиком, со значениями другого устройства для измерения температуры и влажности.

В качестве второго устройства будет использована портативная метеостанция Xiaomi Mijia Bluetooth Hygrothermograph 2. По заявлениям производителя, погрешность при измерении температуры – 0.1 °C, влажности – 1%.

Оба устройства располагаются рядом. Через некоторое время, после стабилизации, измерения параметров окружающей среды становятся более точными.

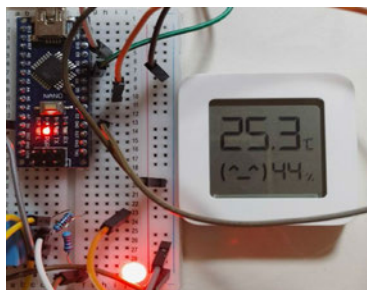


Рис. 5. Измерение температуры метеостанцией и датчиком DTH11

Из рис. 6 видно, что результаты измерений датчиком отличаются от данных метеостанции не более чем на 2 °С и 5%, что соответствует заявленной погрешности DTH11 и позволяет эффективно использовать устройство на практике.

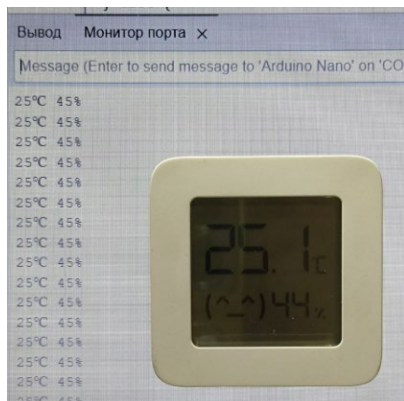


Рис. 6. Сравнение результатов измерений метеостанции и датчика DTH11

Заключение

В ходе выполнения работы была сконструирована система автоматического контроля температуры и влажности, состоящая из устройства на базе Arduino и пользовательского приложения на базе Java. Проведено исследование, которое выяснило, что модель полностью соответствует выдвинутым требованиям: определяет параметры окружающей среды с погрешностью не более 2 °С для температуры и 5% для влажности, позволяет пользователю с удобством следить за данными и менять настройки под собственные нужды. В дальнейшем планируется доработка устройства и приложения для более комфортного использования: создание корпуса из пластика, добавление возможности сохранения пользовательских режимов и смены формата времени при задании бесшумного режима.

Список литературы

1. Документация DTH11 [Электронный ресурс]. – Режим доступа : <https://roboparts.ru/upload/iblock/b15/b15064e95c6560e891be4addf580390b.pdf>
2. Как программируют Arduino [Электронный ресурс]. – Режим доступа : <https://thecode.media/arduino-code/>

3. Монк С. Програмируем Arduino: Основы работы со скетчами / С. Монк. – Санкт-Петербург : Питер, 2016. – 176 с.
4. Документация библиотеки jSerialComm [Электронный ресурс]. – Режим доступа : <https://fazecast.github.io/jSerialComm/javadoc/index.html>
5. Исходный код [Электронный ресурс]. – Режим доступа : <https://github.com/anya-ananasss/arduinoProject>

Организация канала связи клиент-сервер на базе модулей NRF24L01 и двух микроконтроллеров Arduino

С. В. Долгих

Студент бакалавр

С. А. Зуев

Доцент

Введение

В современном мире информационных технологий развитие систем связи играет ключевую роль в обеспечении эффективного обмена данными. В этой гонке за связью и информацией технологии должны гарантировать не только быструю и стабильную передачу данных, но и обеспечивать безупречную надежность в самых разнообразных условиях использования. В этом контексте наша работа сосредотачивается на создании инновационного канала связи между клиентами и сервером, основанного на модулях NRF24L01 и двух микроконтроллерах Arduino.

Исследование базируется на глубоком анализе технических характеристик и возможностей модулей NRF24L01 и микроконтроллеров Arduino. Модуль NRF24L01, разработанный компанией Nordic Semiconductor, представляет собой высокопроизводительный радиочастотный приемопередатчик, способный обеспечивать надежную передачу данных на большие расстояния. Микроконтроллеры Arduino, в свою очередь, являются популярным выбором для реализации различных проектов благодаря их простоте использования и обширной экосистеме.

Применение созданного канала связи может быть широким и разнообразным. Например, в области интернета вещей (IoT), модули NRF24L01 и Arduino могут быть использованы для создания сети умных домов, где различные устройства, такие как датчики, умные розетки и освещение, могут обмениваться данными и управляться из центральной системы. В области мониторинга и управления, эти модули могут быть использованы для создания систем мониторинга окружающей среды, контроля за производственными процессами или даже для управления беспилотными аппаратами.

1. Модуль NRF24L01

NRF24L01 – это однокристалльный приемопередатчик с частотой 2,4 ГГц и встроенным механизмом протокола основной полосы частот (Enhanced ShockBurst), подходящий для беспроводных приложений со сверхнизким энергопотреблением. Модуль разработан для работы в широком во всем мире диапазоне частот ISM 2.400 – 2.4835 ГГц.

Чтобы спроектировать радиосистему с NRF24L01, необходим микроконтроллер и несколько внешних конечных пассивных компонентов.

Управление NRF24L01 и его настройка осуществляется через последовательный периферийный интерфейс (SPI). Карта регистров, доступная через SPI, содержит все регистры конфигурации в NRF24L01 и доступна во всех режимах работы чипа.

Встроенный механизм протокола основной полосы частот Enhanced ShockBurst (ESB) основан на пакетной связи и поддерживает различные режимы – от ручного управления до расширенной автономной работы протокола. Внутренние FIFO обеспечивают бесперебойный поток данных между радиоприемником и микроконтроллером системы. ESB снижает стоимость системы за счет выполнения всех операций высокоскоростного канального уровня.

Интерфейс радиостанции использует модуляцию GFSK. Он имеет настраиваемые пользователем параметры, такие как частотный канал, выходная мощность и скорость передачи данных в эфир. NRF24L01 поддерживает скорость передачи данных по воздуху 250 кбит/с, 1 Мбит/с и 2 Мбит/с.

Высокая скорость передачи данных по воздуху в сочетании с двумя режимами энергосбережения делают NRF24L01 очень подходящим для устройств со сверхнизким энергопотреблением.

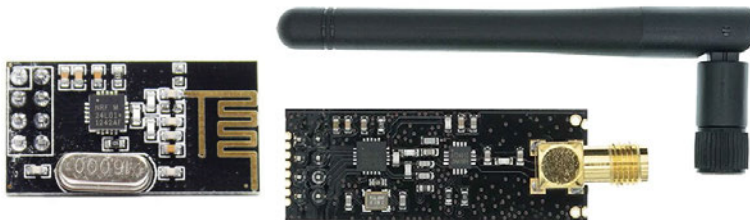


Рис. 1. Модуль NRF24L01

Модуль бывает в нескольких исполнениях: стандартный, мини, модуль с внешней антенной и с внешней антенной и усилителем. В

данной работе будут использоваться стандартные модули и с внешней антенной и усилителем.

Основные характеристики и спецификации этого модуля:

Частота работы: NRF24L01 работает на частоте 2.4 ГГц, что делает его совместимым с многими другими устройствами, использующими этот диапазон частот.

Модуляция: Модуляция сигнала происходит с помощью метода GFSK (Гауссовская частотная модуляция с частотной конволюцией), что обеспечивает хорошую устойчивость к помехам.

Скорость передачи данных: Модуль поддерживает настраиваемую скорость передачи данных от 250 Кбит/с до 2 Мбит/с, что позволяет регулировать энергопотребление и обеспечивать высокую скорость передачи информации.

Дальность передачи: Дальность передачи зависит от условий окружающей среды и может достигать от нескольких десятков до нескольких сотен метров при использовании в типичных условиях.

Множество каналов: NRF24L01 поддерживает работу на 126 каналах, что позволяет избегать перегрузок на занятых частотах и обеспечивает возможность работы в многоканальных средах.

Многопользовательская сеть: Модуль может быть использован для создания многопользовательских сетей, что делает его подходящим для проектов с большим количеством устройств. С помощью него до семи приборов объединяются в общую радиосеть топологии звезда на частоте 2,4 ГГц. Один прибор в радиосети ведущий, остальные ведомые.

Низкое энергопотребление: NRF24L01 спроектирован с учетом низкого энергопотребления, что делает его подходящим для использования в устройствах с ограниченным источником питания.

Гибкий интерфейс: Модуль имеет SPI интерфейс для взаимодействия с микроконтроллерами и поддерживает различные режимы работы, включая передачу данных, прием данных и многое другое.

2. Подключение к микроконтроллеру

Микроконтроллеры Arduino являются популярным выбором для реализации различных проектов благодаря их простоте использования и обширной экосистеме, именно они и будут использоваться в данной работе.

NRF24L01 требует качественного питания 3.3 В, в этом случае выхода Arduino ему не хватает по току. Поэтому для подключения использовались готовые адаптеры питания (рис. 3), которые питаются от 5 В и подают на модуль стабильные 3.3 В.

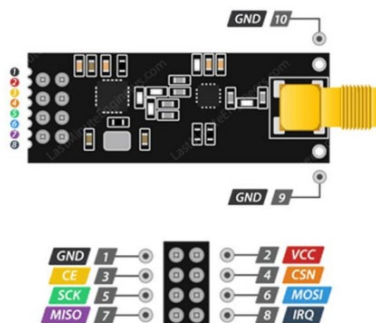


Рис. 2. Назначение контактов подключения NRF24L01 с внешней антенной и усилителем

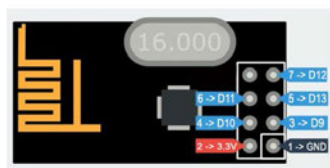


Рис. 3. Назначение контактов подключения NRF24L01

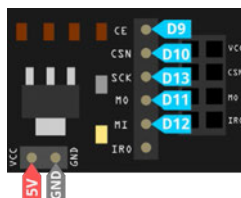


Рис. 4. Назначение контактов подключения адаптера питания

Модуль имеет следующие контакты:

2. GND – общий провод;
3. VCC – питание 3,3 В;
4. CE – включение радиотракта микросхемы высоким уровнем;
5. CSN – Chip Select Not, активный низкий уровень. Если установлен низкий уровень, то модуль отвечает на SPI команды. Это более важный сигнал выбора MC чем сигнал CE;
6. SCK – тактирование шины SPI, до 10 МГц;
7. MOSI – используется для передачи данных от микроконтроллера к устройству;

программы передатчика TX.ino

```

#include <SPI.h>           // библиотека для работы с шиной SPI
#include "nRF24L01.h"      // библиотека радиомодуля
#include "RF24.h"          // ещё библиотека радиомодуля

RF24 radio(9, 10); // "создать" модуль на пинах 9 и 10
byte address[][6] = {"1Node", "2Node", "3Node", "4Node",
"5Node", "6Node"}; //возможные номера труб
byte counter;

void setup() {
  Serial.begin(9600);           // открываем порт для связи с ПК
  radio.begin();                // активировать модуль
  radio.setAutoAck(1);          // режим подтверждения приёма, 1
вкл 0 выкл
  radio.setRetries(0, 15);      // (время между попыткой
достучаться, число попыток)
  radio.enableAckPayload();      // разрешить отсылку данных в
ответ на входящий сигнал
  radio.setPayloadSize(32);      // размер пакета, в байтах

  radio.openWritingPipe(address[0]); // мы - труба 0,
открываем канал для передачи данных
  radio.setChannel(0x60);        // выбираем канал (в
котором нет шумов)

  radio.setPALevel (RF24_PA_MAX); // уровень мощности
передатчика. На выбор RF24_PA_MIN, RF24_PA_LOW, RF24_PA_HIGH,
RF24_PA_MAX
  radio.setDataRate (RF24_250KBPS); // скорость обмена. На
выбор RF24_2MBPS, RF24_1MBPS, RF24_250KBPS

  radio.powerUp();              // начать работу
  radio.stopListening();        // не слушаем радиоэфир, мы
передатчик
}

void loop() {
  Serial.print("Sent: ");
  Serial.println(counter);
  radio.write(&counter, sizeof(counter));
  counter++;
  delay(10);
}

```

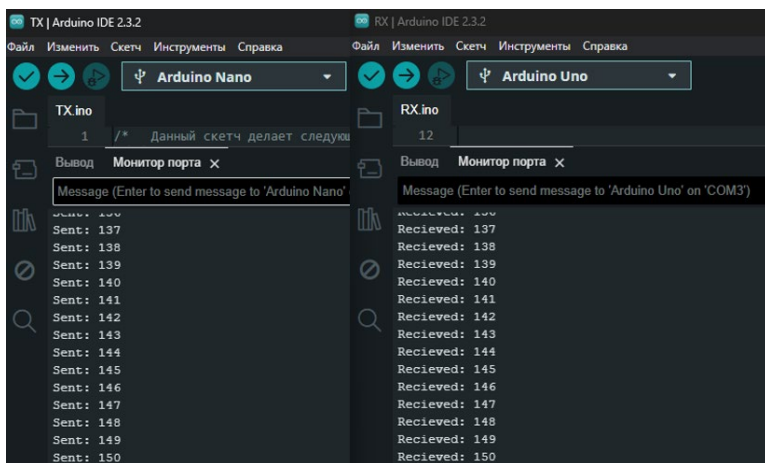


Рис. 6. Мониторы порта с результатом работы программ TX.ino и RX.ino

Ведем наблюдение за процессом передачи данных с помощью монитора порта. По полученным данным можно судить, что передатчик отправляет счетчик, а приемник успешно его получает, что и требовалось реализовать.

Заключение

В данной научной работе мы поставили перед собой цель создания эффективного канала связи между клиентами и сервером на основе модулей NRF24L01 и микроконтроллеров Arduino. Этот канал должен обеспечивать высокую степень защиты данных, надежную передачу в реальном времени с минимальной задержкой и высокую стабильность.

Для достижения поставленной цели мы изучили спецификации и возможности модулей NRF24L01 и микроконтроллеров Arduino, разработали эффективный алгоритм передачи данных, настроили параметры связи и провели тщательное тестирование созданного канала связи на его стабильность, скорость и надежность передачи данных.

Полученные результаты подтверждают, что созданный канал связи на базе модулей NRF24L01 и микроконтроллеров Arduino полностью соответствует поставленным требованиям. Он обеспечивает надежную и стабильную передачу данных между клиентами и сервером, что делает его идеальным выбором для различных приложений, требующих высокой эффективности и защищенности передачи информации.

Список литературы

1. Чистяков Н. И. Радиоприемные устройства / Н. И. Чистяков. – М.: Современное радио, 1978. – 152 с.
2. Андреев В. А. Усилительные устройства: учеб. пособие для вузов по направлению "Радиосвязь, радиовещание и телевидение" и спец. "Телекоммуникация" / В. А. Андреев [и др.]; под ред. О. В. Головина. – М. : Радио и связь, 1993. – 352 с.
3. Розов А. К. Алгоритмы последовательного обнаружения сигналов / А. К. Розов. – СПб. : Политехника, 1992. – 234 с.

Моделирование итеративной кодовой модуляции с побитовым перемежением с последовательным составным свёрточным кодеком

Д. А. Загреба

Студент бакалавр

А. В. Стромов

Старший преподаватель

Введение

В представленной работе рассматривается совместная работа итеративной кодовой модуляции с побитовым перемежением с комбинацией двух последовательных свёрточных кодеков, причём демодуляция и декодирование выполняются итеративно. Задача данной работы – оценить помехоустойчивость рассматриваемой комбинации методов в зависимости от количества итераций в каждом из них.

В [1] описана схема итеративной кодовой модуляции с побитовым перемежением (BICM-ID) с использованием декодера апостериорной вероятности (АПВ) и частотного демодулятора с «мягкими» решениями и обратной связью, учитывающей априорную вероятность демодулируемых данных, реализованной в соответствии с [2]. Схема, рассмотренная в [1] приведена на рис. 1, где Π – перемежитель, Π^{-1} – деперемежитель.

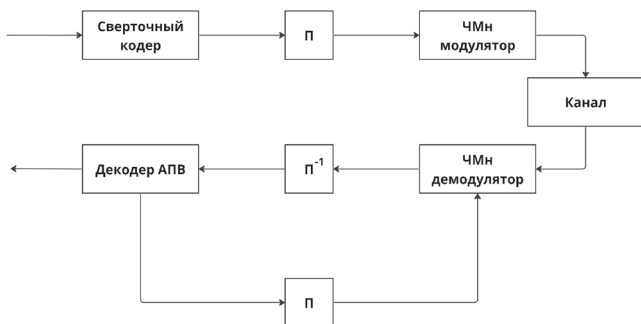


Рис. 1. Схема системы кодовой модуляции с побитовым перемежением и с итеративным декодированием



Рис. 2. Схема системы итеративного декодирования последовательно составного сверточного кода (SCCC)

В [3] описана схема итеративного декодирования последовательно составного свёрточного кода (SCCC). В схеме (рис. 2) используются декодеры апостериорной вероятности.

2. Модель

Построена модель, совмещающая схемы, рассмотренные в [1] и [3] (рис. 3).

Для наглядности назовём обратную связь, передающую априорные оценки для демодулятора через перемежитель Π_2 внешним контуром, а связь, соединяющую декодеры АПВ через перемежитель Π_1 –

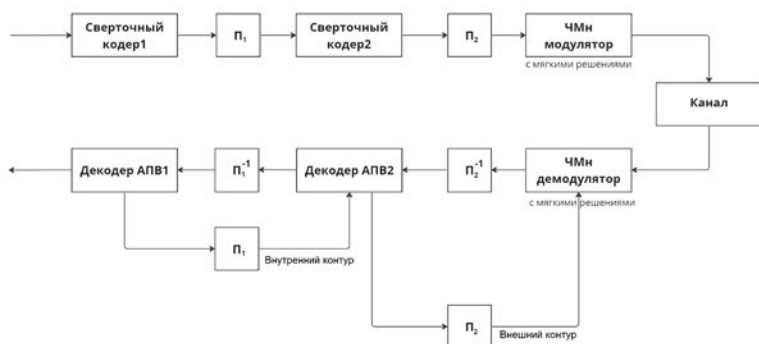


Рис. 3. Схема итеративной кодовой модуляции с побитовым перемежением в связке с последовательным составным сверточным кодом

внутренним контуром. В модели использовались мягкий 4-позиционный частотный демодулятор с некогерентным приёмом и обратной связью с использованием априорных решений, реализованной по формуле (6) из [1], а также АПВ декодеры с точным подсчётом мягких решений.

В ходе моделирования были выполнены численные эксперименты с различным количеством итераций по каждому контуру, результаты которых приведены далее. Использовалась модель канала с аддитивным белым гауссовским шумом.

3. Результаты моделирования

Рассмотрим сначала результаты работы при итеративном использовании только внутреннего контура (рис. 4). При этом частотный демодулятор задействуется однократно, без использования обратной связи.

Далее рассмотрим случай, когда на каждой итерации производится по одному проходу по каждому из контуров, т. е. за одну итерацию демодулятор и каждый из декодеров обработают по 2 раза.

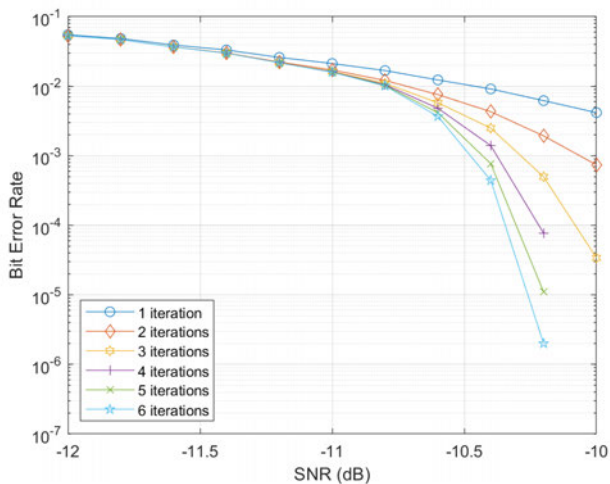


Рис. 4. Частота битовой ошибки при итерациях только по внутреннему контуру.

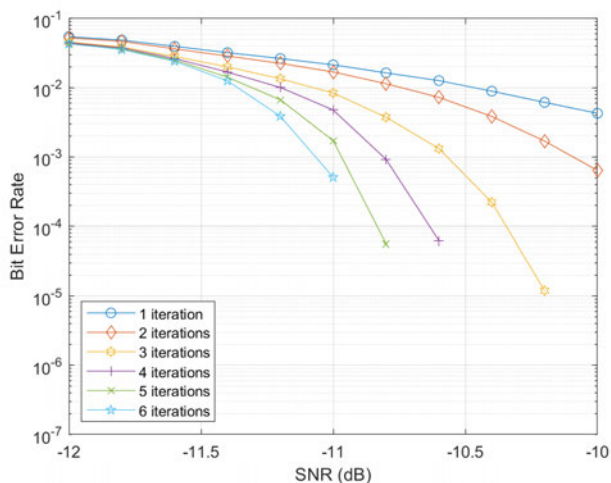


Рис. 5. Частота битовой ошибки при одиночных проходах по внешнему и внутреннему контурам на каждой итерации.

Ниже приведены результаты моделирования, при которых на каждой итерации выполнялось два прохода по внешнему контуру и один по внутреннему (рис. 6), и один по внешнему и два по внутреннему соответственно (рис. 7).

Дополнительно был проведён численный эксперимент, при котором на один проход внешнего контура приходилось три прохода внутреннего (рис. 8).

4. Анализ результатов моделирования

Анализ полученных данных свидетельствует о том, что из всех рассмотренных вариантов наилучшие результаты по соотношению количества выполненных циклов работы частотного демодулятора и декодеров АПВ к достигнутой помехоустойчивости в белом шуме соответствуют случаю двух проходов внутреннего контура на один проход внешнего за итерацию.

При таком соотношении при вычислительных затратах, соответствующих четырём циклам каждого АПВ декодера и двум циклам демодулятора получаем выигрыш в 0,4 дБ по отношению сигнал/шум в канале по уровню частоты ошибки 10^{-4} по сравнению со случаем, в котором внешний контур не задействован. В то же время добавление ещё одной итерации по внутреннему контуру улучшает результат незначительно (рис. 8).

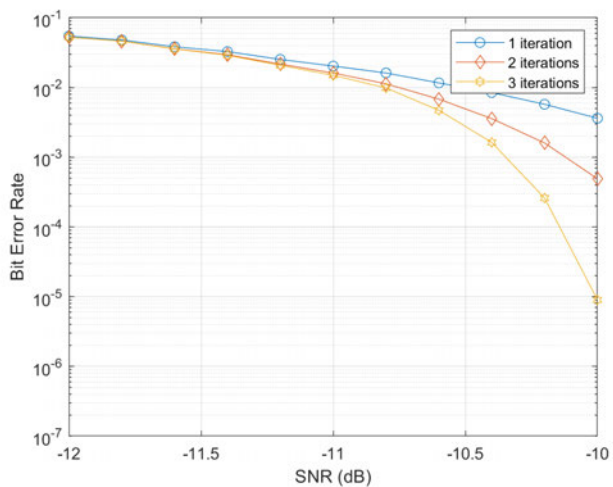


Рис. 6. Частота битовой ошибки при двух проходах по внешнему контуру и одному проходу по внутреннему на каждой итерации

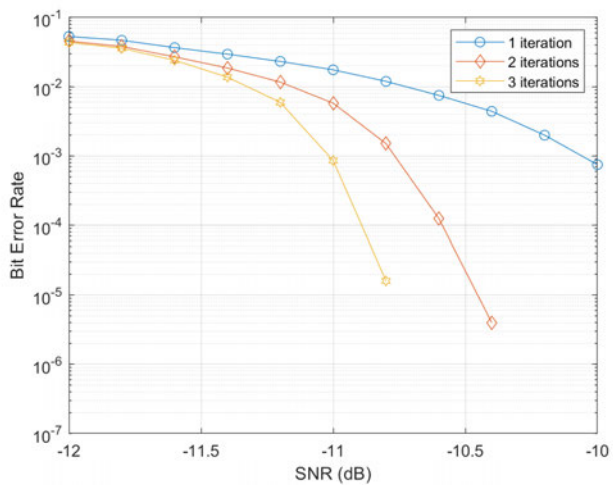


Рис. 7. Частота битовой ошибки при двух проходах по внутреннему контуру и одному проходу по внешнему на каждой итерации

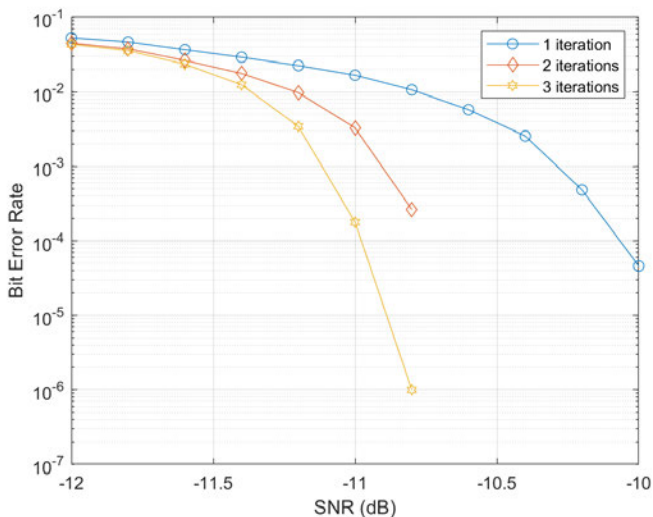


Рис. 8. Частота битовой ошибки при трёх проходах по внутреннему контуру и одному проходу по внешнему на каждой итерации

Заключение

В ходе изучения схемы кодирования и модуляции, составленной из схем BICM-ID и SCCC, смоделирована помехоустойчивость рассматриваемой комбинации схем при различном соотношении количества циклов по каждой из схем. Соотношение два цикла схемы SCCC на один цикл схемы BICM-ID показало наилучшее соотношение помехоустойчивости к вычислительным затратам из рассмотренных вариантов.

Список литературы

1. Valenti M. C. and Shi Cheng, Iterative demodulation and decoding of turbo-coded M-ary noncoherent orthogonal modulation, IEEE Journal on Selected Areas in Communications, vol. 23, no. 9, pp. 1739-1747, Sept. 2005
2. S. Benedetto, G. Montorsi, D. Divsalar, and F. Pollara, Soft-input soft-output modules for the construction and distributed iterative decoding of code networks, Eur. Trans. Telecommun., vol. 9, pp. 155-172, Mar.-Apr. 1998.
3. US Patent 6023783 (Expired), Dariush Divsalar & Fabrizio Pollara, Hybrid concatenated codes and iterative decoding, issued 2000-02-08

Разработка алгоритма сбора информации о товарах с маркетплейсов

Д. А. Змаев

Студент бакалавр

П. С. Лысачев

Старший преподаватель

Введение

В условиях современной глобализации рынков и стремительного развития цифровых технологий, тема автоматизации процессов в электронной коммерции становится особенно актуальной. По мере того, как всё большее количество потребителей и компаний обращаются к онлайн-платформам для покупки и продажи товаров, важность своевременной информации о продуктах увеличивается многократно [1].

Среди множества задач, стоящих перед исследователями в области электронной коммерции, одной из наиболее сложных и критически важных является задача сбора и анализа данных о товарах с маркетплейсов. Данные о товарах – это основа для анализа рыночных трендов, принятия решений о ценообразовании, оптимизации ассортимента и разработки маркетинговых стратегий. Эффективный сбор данных помогает компаниям лучше понять потребности своих клиентов, адаптировать продукты и услуги под текущие рыночные условия и увеличить свою конкурентоспособность.

Однако процесс сбора данных с маркетплейсов наталкивается на ряд препятствий. Во-первых, большинство крупных маркетплейсов, таких как Ozon, SberMarket, Wildberries, используют различные методы защиты, чтобы предотвратить автоматический сбор данных, что делает техническую реализацию таких систем сложной. Во-вторых, данные о товарах часто представлены в неоднородных форматах, что усложняет их анализ и интеграцию в существующие системы управления данными.

В данной статье представлен алгоритм, разработанный специально для сбора данных с российского маркетплейса Ozon. Алгоритм автоматизирует процесс сбора данных, начиная с формирования поискового запроса и заканчивая извлечением и сохранением

информации о товарах в структурированном виде. Это позволяет значительно ускорить процесс анализа данных и повысить его точность.

1. Постановка задачи

Целью данной работы является создание алгоритма парсинга [2-3] маркетплейсов для автоматизации сбора информации о товарах. В качестве конкретного примера для исследования выбран маркетплейс Ozon.

2. Проблема

Основной проблемой, является отсутствие доступа к официальным интерфейсам программирования приложений (API), предоставляемых крупными маркетплейсами для извлечения данных о товарах. Во многих случаях, как например на маркетплейсе Ozon, эта проблема становится критичной, так как API либо не предоставляется вовсе, либо доступ к нему ограничен.

В таких условиях единственным возможным методом извлечения данных является парсинг веб-страниц, то есть автоматизированный сбор информации непосредственно из HTML-кода страниц.

3. Основная задача

Основная задача, которую решает данный алгоритм – это возможность получения данных о товаре с маркетплейсов, таких как название товара, ссылка на него, ссылка на изображение и его цена.

Решение зависит от выбора инструментов, при помощи которых будет произведен парсинг. Алгоритм парсинга маркетплейса Ozon реализован на языке программирования Python с использованием библиотек Selenium [4], BeautifulSoup [5].

4. Описание работы алгоритма

На первом этапе алгоритма происходит инициализация объекта класса ItemScraper, отвечающего за парсинг конкретного товара. При инициализации объекта передается поисковый запрос, формирующийся из названия товара или его артикула. На основе поискового запроса формируется URL-адрес страницы с результатами поиска на маркетплейсе Ozon.

Далее алгоритм настраивает веб-драйвер для работы в режиме headless, что позволяет выполнять парсинг без отображения графического интерфейса браузера. Затем веб-драйвер переходит по сформированному URL-адресу и ожидает загрузки страницы.

После загрузки страницы алгоритм получает ее исходный код и передает его в объект класса BeautifulSoup для последующего парсинга. С помощью методов класса BeautifulSoup алгоритм извлекает из

исходного кода страницы информацию о товарах, соответствующих поисковому запросу.

Информация о товарах включает в себя название товара, ссылку на товар, цену, изображение товара и дополнительные параметры, такие как наличие бонусов при покупке. Алгоритм обрабатывает полученную информацию и формирует объект класса `OzonItem`, содержащий все необходимые данные о товаре.

Диаграмма последовательности алгоритма представлена на рис. 1.

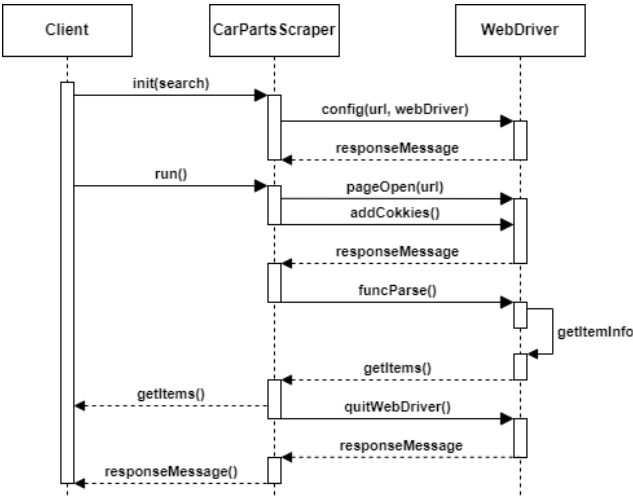


Рис. 1. Диаграмма последовательности алгоритма сбора информации с маркетплейса

5. Пример выполнения алгоритма

Для демонстрации работы разработанного алгоритма парсинга маркетплейса Ozon, будет рассмотрен следующий сценарий:

Для начала необходимо инициализировать поисковый запрос, например, «toyota+corolla+e180+свечи» который представляет собой запрос на поиск свечей для автомобиля Toyota Corolla E180.

```
...
search = 'toyota+corolla+e180+свечи'
...
```

После инициализации алгоритм запускается, иницируя экземпляр класса `ItemScraper` с указанным поисковым запросом. Затем алгоритм автоматически загружает страницу маркетплейса Ozon, используя веб-драйвер Selenium. Наконец, происходит сам парсинг и отображение

результатов. Результаты парсинга, представленные на рис. 2, включают информацию о найденных товарах, их ценах и изображениях, они могут быть выведены на экран или сохранены в **xlsx** файл для дальнейшего анализа.

Name	Link	Image Link	Last Price
Свечи зажигания для TOYOTA MAZDA LEXUS Денсо Iri	https://www.ozon.ru/product/	https://ir.ozone.ru/s3/mu1305	
Toyota Прокладка свечного колодца комплект 4 шт.	https://www.ozon.ru/product/	https://ir.ozone.ru/s3/mu783	
Ремкомплект ограничителей на 4 двери Toyota COR	https://www.ozon.ru/product/	https://ir.ozone.ru/s3/mu621	
Свеча зажигания NGK 4912 ILKAR7B-11, 1 шт (Toyota A	https://www.ozon.ru/product/	https://ir.ozone.ru/s3/mu426	
Свечи зажигания / Комплект свечей зажигания/Aver	https://www.ozon.ru/product/	https://ir.ozone.ru/s3/mu1373	
Свеча зажигания Toyota 9091901275 для Toyota Auris	https://www.ozon.ru/product/	https://ir.ozone.ru/s3/mu407	
AMD Свеча зажигания AMD.PL79	https://www.ozon.ru/product/	https://ir.ozone.ru/s3/mu440	

Рис. 2. **xlsx** файл, с результатом выполнения алгоритма

Заключение

Таким образом, разработанный алгоритм парсинга маркетплейса Ozon позволяет автоматизировать сбор информации о товарах с веб-страниц маркетплейса, что может быть полезно для аналитики рынка и принятия решений в сфере электронной коммерции, а также может быть полезен разработчикам, которым необходима информация о товарах с маркетплейса Ozon.

Список литературы

1. Гаврилов, Л. П., Электронная коммерция / Л. П. Гаврилов. – 6-е изд., перераб. и доп. – М. : Юрайт, 2024. – 579 с.
2. A Practical Introduction to Web Scraping in Python [Электронный ресурс] : сайт. – Режим доступа : <https://realpython.com/python-web-scraping-practical-introduction/>
3. Web Scraping With Python [Электронный ресурс] : сайт. – Режим доступа : <https://brightdata.com/blog/how-tos/web-scraping-with-python>
4. Selenium with Python [Электронный ресурс] : сайт. – Режим доступа : <https://selenium-python.readthedocs.io/>
5. beautifulsoup [Электронный ресурс] : сайт. – Режим доступа : <https://pypi.org/project/beautifulsoup4/>

Анализ топологии сервиса с помощью технологии Annotation processing

С. П. Иванников

Студент бакалавр

Н. К. Самойлов

Старший преподаватель

Введение

В современной информационной эпохе, когда объемы данных и сложность информационных систем неуклонно растут, особую важность приобретает вопрос организации и анализа структуры сложных программных решений. Одним из перспективных направлений в этой области является применение технологии Annotation processing в контексте анализа топологии сервисов. Annotation processing – мощный инструмент в арсенале современного программиста, позволяющий обрабатывать аннотации в коде и генерировать дополнительные исходные, классовые файлы или другие артефакты в процессе компиляции. Аннотации в Java, как и во многих других языках программирования, предоставляют метаданные, которые могут быть использованы для различных целей: от документирования и конфигурирования до изменения поведения программы во время выполнения. Annotation processor расширяет эти возможности, позволяя разработчикам влиять на процесс компиляции, анализируя и модифицируя структуру кода в соответствии с заданными аннотациями. Применение annotation processor для анализа топологии сервиса открывает новые горизонты в понимании и управлении сложными системами. Анализируя аннотации, разработчики могут автоматически генерировать документацию, карты зависимостей, графы вызовов и другие полезные артефакты, которые обеспечивают более глубокое понимание внутренней структуры и взаимодействий внутри приложения или микросервисной архитектуры.

1. Annotation processor

Annotation processing в Java – это мощный механизм, позволяющий автоматически генерировать код и выполнять проверки на этапе компиляции. Эта технология предоставляет разработчикам возможность

добавлять метаданные в код, которые затем обрабатываются специальными инструментами для создания дополнительных файлов и проверок.

Annotation processing начинается на этапе компиляции. Когда компилятор запускается, он сканирует путь к классам (classpath) и путь к исходным файлам (sourcepath) в поисках файлов конфигурации процессоров аннотаций. Эти конфигурационные файлы находятся в директории resources/META-INF/services и содержат имена классов, реализующих интерфейс javax.annotation.processing.Processor [1]. Пример приведен в листинге 1.

Листинг 1

Пример регистрации процессора аннотаций

```
ru.cs.processor.CustomProcessor1  
ru.cs.processor.CustomProcessor2
```

Компилятор анализирует исходный код и собирает информацию о всех аннотациях, присутствующих в коде. На основании этой информации он определяет, какие процессоры аннотаций должны быть запущены. Каждый процессор аннотаций указывает, какие аннотации он поддерживает, используя аннотацию @SupportedAnnotationTypes или метод getSupportedAnnotationTypes [2].

Для создания процессора аннотаций в проекте необходимо создать аннотации, которые должны будут обрабатываться препроцессорами, так как annotation processor отрабатывает на этапе компиляции, аннотации могут иметь параметр @Retention равный Retention.Source. Также необходимо создать классы процессора аннотаций, эти классы должны быть унаследованы от AbstractProcessor и имплементировать метод process. Общий вид кода представлен в листинге 2.

Листинг 2

Переопределение метода process

```
@Override  
public boolean process(  
    Set<? extends TypeElement> annotations,  
    RoundEnvironment roundEnv) {...}
```

Процессор может генерировать новые файлы исходного кода, классы и ресурсы, используя интерфейс javax.annotation.processing.Filer. Например, процессор может создать новый класс, основанный на аннотации, или добавить методы в существующий класс [3].

Annotation processing может проходить в несколько этапов (раундов). На каждом этапе компилятор повторно анализирует вновь созданные файлы и вызывает процессоры аннотаций для обработки

новых аннотаций. Этот процесс продолжается до тех пор, пока не будут обработаны все аннотации и не будут сгенерированы все необходимые файлы. Данная схема представлена на рис. 1.

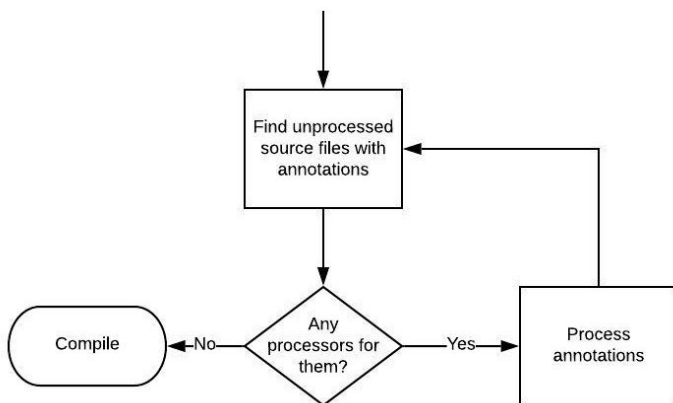


Рис. 1. Схема работы процессора аннотаций

Одним из главных преимуществ Annotation processing является возможность автоматической генерации кода. Это позволяет значительно сократить количество шаблонного кода и уменьшить вероятность ошибок, связанных с ручным написанием однотипных конструкций. Например, библиотеки, такие как Lombok, используют Annotation processing для автоматического создания методов доступа (геттеров и сеттеров), конструкторов, методов toString, equals и hashCode [4].

Annotation processing позволяет выполнять сложные проверки на этапе компиляции. Это помогает обнаружить ошибки на ранней стадии разработки, что снижает затраты на их исправление и повышает надежность программного обеспечения. Процессоры аннотаций могут проверять корректность использования аннотаций, соответствие кода заданным архитектурным требованиям и многое другое.

Автоматическая генерация кода и выполнение проверок на этапе компиляции могут значительно повысить производительность разработки. Это позволяет сократить время на написание, тестирование и отладку кода. Кроме того, генерируемый код обычно оптимизирован и содержит меньше ошибок, что положительно сказывается на общем качестве приложения.

2. Проблемы микросервисной архитектуры в контексте анализа топологии

Микросервисная архитектура является одним из ключевых подходов к построению современных распределенных систем. Этот подход предполагает разделение приложения на множество мелких, изолированных сервисов, каждый из которых выполняет отдельную функцию и взаимодействует с другими через стандартизированные API [5]. Такое деление позволяет улучшить масштабируемость, гибкость и независимость разработки, что особенно важно для крупных и динамично развивающихся проектов. Однако, несмотря на очевидные преимущества, микросервисная архитектура несет в себе ряд сложных проблем, особенно в контексте анализа топологии системы. Важно понимать, что топология микросервисов – это не просто схема взаимодействия между сервисами, но и сложная сеть зависимости, маршрутизации и координации, требующая тщательного управления и мониторинга.

Одной из ключевых проблем является сложность управления микросервисами. В отличие от монолитной архитектуры, где изменения в коде влияют на одно целое приложение, в микросервисах каждая отдельная часть системы может развиваться и деплоиться независимо. Это приводит к увеличению числа версий сервисов и сложностей в их согласовании. Поддержание согласованности интерфейсов и контрактов между сервисами становится критически важным аспектом.

Мониторинг и наблюдаемость микросервисной системы – еще одна значительная проблема. В условиях множества изолированных сервисов традиционные методы мониторинга, которые применяются к монолитным приложениям, становятся неэффективными. Необходимо внедрение распределенных систем логирования и трассировки для отслеживания запросов по всей системе. Такие инструменты, как Prometheus, Grafana и Jaeger, помогают в этом, но требуют дополнительной настройки и интеграции.

Микросервисная архитектура, безусловно, предоставляет значительные преимущества в плане гибкости и масштабируемости приложений. Однако, она также приносит с собой множество сложностей, связанных с управлением, мониторингом, отладкой и безопасностью системы. Анализ топологии микросервисной системы требует тщательного подхода и использования специализированных инструментов и методов для обеспечения стабильности, и безопасности всей архитектуры.

3. Роль Annotation Processor в анализе топологии

Annotation processor может значительно упростить анализ и оптимизацию структуры микросервисов. Он позволяет автоматически собирать информацию о топологии системы, проверять целостность и согласованность конфигураций и выявлять потенциальные проблемы. При этом процессор аннотаций может анализировать аннотации, описывающие зависимости и взаимодействия между микросервисами, и на основе этой информации создавать визуальные представления системы. Это помогает разработчикам лучше понимать структуру и взаимодействия в системе.

4. Применение на практике

В рамках данного кейса была создана система управления информацией о программных компонентах для крупного программного продукта. Система должна была обеспечивать следующие функциональные возможности:

- организация централизованного хранилища для метаданных компонентов, включая их идентификаторы, названия, типы и связи с другими компонентами;
- поддержка возможности обновления информации о компонентах, включая изменение их атрибутов и связей;
- обеспечение функциональности безопасного удаления данных компонентов, с учетом целостности связей в графовой базе данных;
- предоставление механизма для выполнения сложных запросов, позволяющих извлекать данные о компонентах и их связях на основе различных критериев.

Библиотека обеспечивает сохранение информации о компонентах в графовую базу данных. Она предоставляет удобный интерфейс для работы с данными компонентов, позволяя разработчикам легко сохранять и обновлять информацию о компонентах. Данный пример представлен на рис. 2.

Для обновления данных компонентов используются методы библиотеки, которые принимают измененные данные компонента и обновляют соответствующие записи в базе данных. Библиотека позволяет работать с данными компонентов в виде объектов, что обеспечивает удобную и интуитивно понятную работу с ними. Удаление данных компонентов осуществляется с помощью методов библиотеки, которые обеспечивают корректное удаление записей из базы данных и соответствующих связей между компонентами. Используется транзакционное управление для обеспечения целостности данных.

Описанный кейс показал высокую применимость и полезность в реальных условиях разработки программного обеспечения. Это

подтверждает её значимость и перспективность для дальнейшего использования и развития.

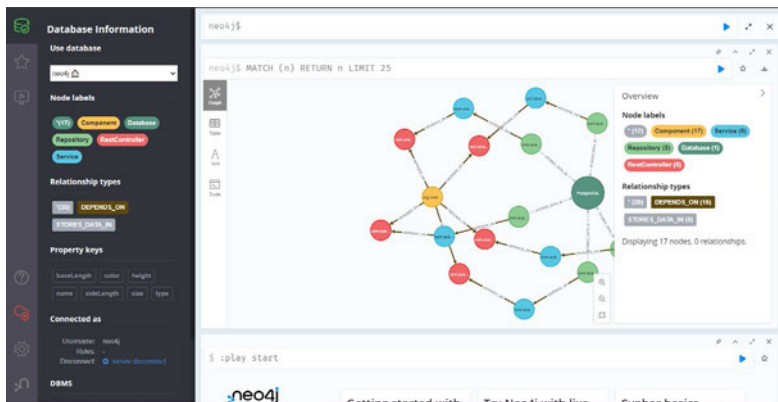


Рис. 2. Интерфейс для работы с данными в графовой базе данных Neo4j

Заключение

В результате работы была разработана библиотека, которая позволяет автоматизировать анализ топологии сервиса с использованием аннотационного процессора. Данная библиотека собирает данные о компонентах и их зависимостях, сохраняет их в базе данных Neo4j и предоставляет возможности для визуализации топологии сервиса. Это решение способствует улучшению прозрачности и управляемости сложных распределенных систем.

Сформированы перспективные направления дальнейшего развития проекта:

- добавление новых видов анализа, таких как выявление циклических зависимостей и автоматический рефакторинг приложений для улучшения архитектуры;
- добавление возможности интерактивного графического представления топологии с поддержкой фильтрации и поиска по компонентам;
- разработка интеграций с популярными DevOps инструментами и системами CI/CD для автоматизации анализа на этапе сборки и развертывания;
- добавление поддержки новых внешних баз данных для хранения данных анализа.

Список литературы

1. Java Annotation Processing and Creating a Builder [Электронный ресурс] : сайт. – Режим доступа: <https://www.baeldung.com/java-annotation-processing-builder>
2. Annotation Processing 101 [Электронный ресурс] : сайт. – Режим доступа: <https://hannedorfmann.com/annotation-processing/annotationprocessing101/>
3. Customize Java Annotation with Examples [Электронный ресурс] : сайт. – Режим доступа: <https://www.geeksforgeeks.org/customize-java-annotation-with-examples/>
4. Project Lombok [Электронный ресурс] : сайт. – Режим доступа: <https://projectlombok.org/>
5. Microservices: a definition of this new architectural term [Электронный ресурс] : сайт. – Режим доступа: <https://martinfowler.com/articles/microservices.html>

Обучение искусственному интеллекту для пошаговой стратегии

Д. А. Карташов

Студент бакалавр

А. В. Акимов

Старший преподаватель

Введение

Идея создания искусственного интеллекта волнует человеческое сознание, наверное, с момента появления первых вычислительных машин. Более того, количество задач, исполнение которых, хотелось бы поручить именно искусственному интеллекту растет с каждым днем.

Сегодня многие игры так или иначе имеют реализованный искусственный интеллект. Таким образом, компьютер, обладая алгоритмом, на основе которого он выбирает свои действия, может играть в игру и составлять очень серьезную конкуренцию для человека. Например, шахматный движок Stockfish вполне способен обыгрывать мировых чемпионов [1].

Но прогресс не стоит на месте и на смену ИИ, реализованному с помощью алгоритмов, логика принятия решений которого заранее прописана разработчиками, приходит ИИ, который использует нейронные сети и обучается сам, не используя в этом процессе человеческий опыт.

В 2019 году компания DeepMind выпустила модель под названием MuZero, которую научили, в числе прочих игр, играть в шахматы. Отличительной особенностью этой модели стало то, что она основана только на поиске по дереву Монте-Карло и нейронных сетях, для которых используется обучение с подкреплением [2].

Так как MuZero на данный момент является одним из лучших решений для тренировки ИИ в играх [2], то она и была выбрана для реализации ИИ для пошаговой стратегии, в которой нескольким командам персонажей необходимо перемещаться по игровому полю, искать монеты и собирать их на базе. Персонажи из разных команд могут атаковать друг друга. При этом, на поле существует несколько разных типов клеток, каждый из которых имеет свои особенности, а

также существуют несколько различных типов персонажей с разными способностями и видами атак и передвижения.

1. Поиск по дереву Монте-Карло

Одной из важнейших вещей в MuZero является использование алгоритма поиска по дереву Монте-Карло. Изначально в дереве есть лишь начальный узел, с начальным состоянием игры. Далее же алгоритм можно разделить на четыре стадии: выбор, расширение, симуляция, обратное распространение.

На стадии выбора происходит поиск листа дерева, которым является один из конечных узлов. Поиск такого узла может основываться на верхней доверительной границе

$$U = \frac{w_i}{n_i} + c \sqrt{\frac{\ln(t)}{n_i}},$$

где U – верхняя доверительная граница, w_i – количество побед после i -го хода, n_i – количество симуляций после i -го хода, c – параметр исследования, t – общее количество симуляций для родительского узла.

На стадии расширения добавляется новый узел с игровым состоянием, следующим за игровым состоянием в предыдущей стадии.

Далее происходит симуляция игры до конца из созданного состояния.

На последней стадии алгоритм, дошедший до конца игры, оценивает итоговое состояние. В зависимости от результата игры, происходит корректировка параметров всех узлов выше по дереву.

Пример работы алгоритма поиска по дереву Монте-Карло показан на рис. 1.

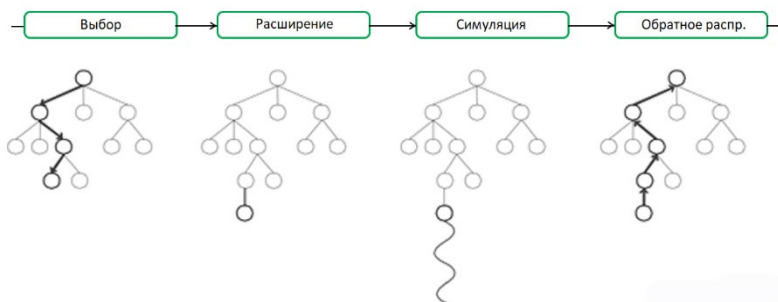


Рис. 1. Пример алгоритма поиска по дереву Монте-Карло

2. Нейронные сети, использующиеся в MuZero

Помимо поиска по дереву Монте-Карло, MuZero совмещает в себе три нейронные сети: нейросеть представления, динамическая нейросеть и нейросеть прогнозирования [3].

Нейросеть представления предназначена для формирования начального состояния игры $s^0 = h_\theta(o_1, \dots, o_t)$ на основе обработки прошлых наблюдений, где s^0 – начальное состояние игры, o_i – наблюдения, а h_θ – функция представления.

Динамическая нейросеть служит для имитации динамики окружения среды игры. Имея на входе состояние игры s^{k-1} и выбранное действие a^k , она выдает переход в следующее состояние $s^k = g_\theta(s^{k-1}, a^k)$ и соответствующее вознаграждение r^k .

Нейросеть прогнозирования необходима для предсказания вероятности p^k и ценности действия $v^k = f_\theta(s^k)$ на основе полученного состояния.

Схема взаимодействия этих нейросетей между собой представлена на рис.2.

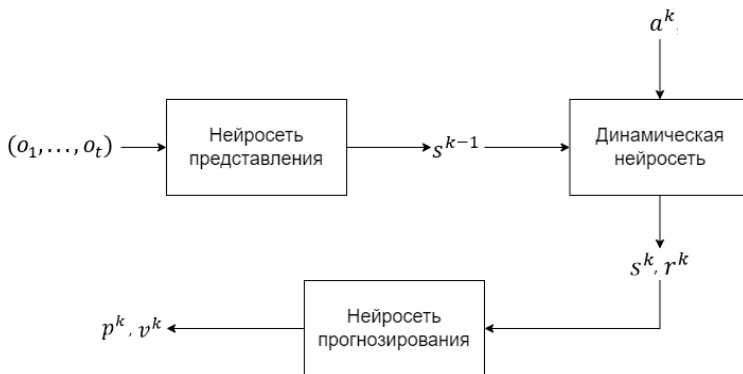


Рис. 2. Взаимодействие нейросетей MuZero между собой

3. Этапы работы MuZero

Изначально в дереве существует только один узел – стартовый. Он означает начальное состояние игры. Далее, в начале каждой итерации обучения модель ищет в дереве терминальный узел, основываясь на верхней доверительной границе. После этого работу MuZero можно

разбить на несколько основных этапов: выбор действия, расширение дерева, обратное распространение [3].

На первом этапе происходит выбор подходящего узла дерева поиска и выбор действия, основываясь на верхней границе доверия, используя следующую формулу

$$U = \operatorname{argmax}(Q) + P \frac{\sqrt{\sum_b N}}{1 + N} \left[c1 + \log \left(\frac{\sum_b N + c2 + 1}{c2} \right) \right],$$

где N – количество посещений узла, Q – расчетное среднее значение вознаграждения, P – вероятность стратегии, а $c1$ и $c2$ – константы, которые регулируют относительный вес P и Q .

На этапе расширения выбранное действие выполняется. После нейросеть представления формирует соответствующее игровое состояние, таким образом происходит расширение дерева. Далее происходит рекурсивный процесс, динамическая нейросеть проигрывает игру до конца, то есть пока не будет выполнено условие победы, формируя таким образом новые состояния. Таким образом выполняется третий этап работы алгоритма поиска по дереву Монте-Карло. Нейросеть прогнозирования же, на каждом шагу пытается предсказать связанное с ними вознаграждение r^j и значения P^j и v^j .

На последнем этапе происходит обратное распространение результатов работы вверх по дереву. Рассчитывается совокупное вознаграждение в узле, добавленном в начале поиска по дереву на основе значения v^j в каждом узле симуляции. Далее Q и N обновляются по пути поиска в дереве. После достижения корневого узла s^0 , устанавливается и нормализуется количество посещений N .

Данные, которые нужны нейросетям для обучения, необходимо генерировать кодом. Таким образом, на каждом ходу, включая начальное состояние, формируются матрицы, демонстрирующие позиции агента, которым является персонаж, управляемый моделью, и целевой точки. Они нужны, чтобы нейросеть представления смогла наиболее корректно сформировать состояние игры, чтобы отдать его динамической нейросети.

Также на каждом ходу формируется маска действий, в которой помечаются действия, возможные к выполнению агентом. Эта информация также нужна для работы динамической нейросети [3].

Далее, когда агент достигает целевой точки, модели выдается вознаграждение, что помогает нейросети прогнозирования из раза в раз лучше предсказывать необходимые значения.

4. Создание авторской среды для тренировки модели

Для отработки взаимодействия с MuZero была выбрана подзадача обучения поиска пути персонажем. Подзадачей является нахождение кратчайшего пути от стартовой до целевой точки. Средой работы модели является квадратное поле размером 7 на 7 полей. Модель управляет агентом, который начинает двигаться из стартовой точки с координатами $(0,0)$ и может двигаться только по вертикали или горизонтали по одной клетке за ход. Целевая точка каждый раз генерируется случайным образом.

Для задания начального состояния была переопределена функция инициализации среды `reset()`. В ней генерируется координата целевой точки, формируется первая маска действий агента, о которой было сказано в предыдущей части статьи и где помечены те действия, которые агент может выполнить в данный момент игры, и создаются матрицы, которые описывают состояние среды: положение агента и положение целевой точки.

Для организации движения агента была переопределена функция действия агента `step(action)`, где `action` – индекс действия, которое было выбрано для агента. После этого, как и в функции для задания начального состояния, формируются матрицы состояния среды и маска действий агента. Далее происходит проверка, выполнено ли условие конца игры после очередного действия. Если условие выполнено, то модели выдается награждение.

При обучении возникала проблема, при которой агент заикливался на одних и тех же действиях и не мог таким образом окончить очередную симуляцию, чтобы продолжить обучение. В итоге она была решена, путем увеличения параметра температуры, что подтолкнуло модель с большей вероятностью выбирать действия, которые использовались меньшее количество раз [3].

5. Тестирование модели после тренировки

В итоге, после тренировки, модель научилась находить кратчайший путь до целевой точки, где бы та ни находилась. Примеры поиска кратчайшего пути агентом показаны на рис. 3.

Здесь агент – черный X, стартовая точка – В, целевая точка – О, а клетки, по которым двигался агент помечены красным X. Агент не мог ходить по диагонали. В итоге в каждом из примеров он дошел до целевой точки за 12, 5 и 10 шагов соответственно. Эти пути являются кратчайшими.

Необходимо отметить, что во время обучения, агент не получал никакой информации о том, каким образом наиболее правильно

действовать в той или иной ситуации. Вся информация, которой он обладал – возможные ходы и награда при достижении целевой точки.

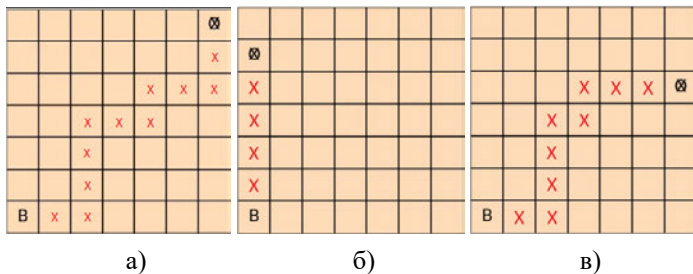


Рис. 3. Примеры поиска кратчайшего пути агентом

Таким образом, можно сделать вывод, что модель научилась сама анализировать игровую ситуацию и выработала наилучшую стратегию для решения поставленной задачи.

Заключение

Данная статья посвящена обучению искусственного интеллекта в пошаговой стратегии. В рамках нее освещена работа модели MuZero, ее внутренних нейронных сетей, этапы ее работы и алгоритм поиска по дереву Монте-Карло. Помимо этого, была разработана среда для тренировки модели играть в пошаговую стратегию и организовано ее обучение, в результате которого, она научилась выполнять поставленную задачу и делать это наиболее качественно, не используя во время обучения человеческий опыт.

Далее планируется усложнять среду для обучения ИИ, увеличивать количество монет на поле, увеличивать размер поля, добавлять различные типы клеток.

Список литературы

1. Киргизбаев, С. П. Эволюция искусственного интеллекта: от шахматных движений до сверхинтеллекта / С. П. Киргизбаев, В. П. Киргизбаев // Студенческие научные исследования: сборник статей XIX. – 2024. – С. 24.
2. LightZero: A Unified Benchmark for Monte Carlo Tree Search in General Sequential Decision Scenarios / Niu Y. [et al.] // Advances in Neural Information Processing Systems. – 2024. – Т. 36.
3. Mastering atari, go, chess and shogi by planning with a learned model / Schrittwieser J. [et al.] // Nature. – 2020. – Т. 588. – №. 7839. – С. 604-609.

Двухкомпонентные системы равномерных сдвигов, порожденные функцией Гаусса и контуром Лоренца

А. И. Кобзарь

Студент бакалавр

Д. А. Татьяна

Студент бакалавр

Е. А. Киселев

Доцент

Введение

В настоящее время однокомпонентные системы равномерных сдвигов достаточно хорошо изучены. На практике широко применяются базисные сплайны, а также оконные системы, порожденные функцией Гаусса [1, 2]. В последнем из указанных случаев речь идет фактически об одновременном сдвиге во временной и частотной области.

Когда используется система сдвигов, сигнал предполагается состоящим из компонент одинаковой формы, но разной амплитуды и возникающих в различные моменты времени. Тем не менее, как показывают результаты исследований, часто складывается ситуация, когда компоненты различаются также и по другим характеристикам, например, по ширине. Подобного рода поведение демонстрируют сигналы электрокардиограммы (ЭКГ) электроэнцефалограммы (ЭЭГ) [3, 4]. В таких случаях логично использовать многокомпонентные системы сдвигов.

Целью данной работы является построение биортогональных систем для двухкомпонентных систем сдвигов, порожденных функцией Гаусса и контуром Лоренца. Функция Гаусса в настоящее время успешно используется для моделирования сигналов ЭКГ и ЭЭГ [3, 4], а контур Лоренца применяется в атомной спектроскопии и ядерной физике [5, 6].

1. Теоретическая часть

Скалярное произведение в пространстве $L_2(\mathbb{R})$ и норму определим стандартным образом

$$(f, g) = \int_{-\infty}^{\infty} f(x) \overline{g(x)} dx, \|f\| = \sqrt{(f, f)}. \quad (1)$$

Прямое и обратное преобразование Фурье зададим в следующей форме

$$\hat{f}(\xi) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x) e^{-ix\xi} dx, f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{f}(\xi) e^{ix\xi} d\xi. \quad (2)$$

Нам понадобятся два его свойства

$$(f(x-a))^{\wedge}(\xi) = \hat{f}(\xi) e^{-ia\xi}, \quad (3)$$

$$(f, g) = (\hat{f}, \hat{g}). \quad (4)$$

Исследуется двухкомпонентное семейство равномерных сдвигов, состоящее из функций $\phi_1(x-k)$, $\phi_2(x-k)$, $k \in \mathbb{Z}$. Фактически рассматриваются целочисленные сдвиги, однако это не ограничивает общности, поскольку шаг сдвига заменой переменных всегда может быть сделан равным единице.

Данная работа посвящена построению биортогональных систем. Дадим им определение, причем сразу адаптированное для двухкомпонентных систем.

Определение 1. Два набора функций $\phi_1(x-k)$, $\phi_2(x-k)$, $k \in \mathbb{Z}$ и $\psi_1(x-m)$, $\psi_2(x-m)$, $m \in \mathbb{Z}$ образуют биортогональную систему, если выполнены следующие соотношения ортогональности

$$\begin{aligned} (\phi_1(x-k), \psi_1(x-m)) &= \delta_{km}, (\phi_2(x-k), \psi_1(x-m)) = 0, \\ (\phi_1(x-k), \psi_2(x-m)) &= 0, (\phi_2(x-k), \psi_2(x-m)) = \delta_{km}. \end{aligned} \quad (5)$$

Если биортогональная система построена, то коэффициенты какой-либо конечной линейной комбинации

$$f(x) = \sum_k c_{1,k} \phi_1(x-k) + \sum_k c_{2,k} \phi_2(x-k) \quad (6)$$

могут быть найдены следующим образом

$$c_{1,k} = (f(x), \psi_1(x-k)), c_{2,k} = (f(x), \psi_2(x-k)). \quad (7)$$

Для построения биортогональной системы в случае однокомпонентных систем сдвигов имеются хорошо известные общие формулы [1, 2]. В рамках данной работы мы рассматриваем более сложный случай двухкомпонентных систем.

Будем искать $\psi_1(x)$ в виде линейной комбинации

$$\psi_1(x) = \sum_k d_{1,k} \phi_1(x-k) + \sum_k d_{2,k} \phi_2(x-k) \quad (8)$$

с неопределенными коэффициентами $d_{1,k}$ и $d_{2,k}$. Используя свойства (3) и (4), запишем условия ортогональности для $\psi_1(x)$ в образах Фурье

$$\begin{aligned} \sum_k d_{1,k} \int_{-\infty}^{\infty} \left| \hat{\phi}_1(\xi) \right|^2 e^{i(m-k)\xi} d\xi + \sum_k d_{2,k} \int_{-\infty}^{\infty} \hat{\phi}_2(\xi) \overline{\hat{\phi}_1(\xi)} e^{i(m-k)\xi} d\xi &= \delta_{0m}, \\ \sum_k d_{1,k} \int_{-\infty}^{\infty} \hat{\phi}_1(\xi) \overline{\hat{\phi}_2(\xi)} e^{i(m-k)\xi} d\xi + \sum_k d_{2,k} \int_{-\infty}^{\infty} \left| \hat{\phi}_2(\xi) \right|^2 e^{i(m-k)\xi} d\xi &= 0. \end{aligned} \quad (9)$$

Выполнив следующее преобразование во всех интегралах

$$\int_{-\infty}^{\infty} f(x) dx = \sum_{n=-\infty}^{\infty} \int_{2\pi n}^{2\pi(n+1)} f(x) dx = \int_{-\infty}^{\infty} \left(\sum_{n=-\infty}^{\infty} f(x + 2\pi n) \right) dx, \quad (10)$$

после несложных, но громоздких выкладок приходем к набору соотношений

$$\begin{aligned} \int_0^{2\pi} (D_1(\xi) p_{11}(\xi) + D_2(\xi) p_{12}(\xi)) e^{im\xi} d\xi &= \delta_{0m}, \\ \int_0^{2\pi} (D_1(\xi) p_{21}(\xi) + D_2(\xi) p_{22}(\xi)) e^{im\xi} d\xi &= 0, \end{aligned} \quad (11)$$

где

$$\begin{aligned} D_1(\xi) &= \sum_k d_{1,k} e^{-ik\xi}, \quad D_2(\xi) = \sum_k d_{2,k} e^{-ik\xi}, \\ P(\xi) &= \begin{pmatrix} p_{11}(\xi) & p_{12}(\xi) \\ p_{21}(\xi) & p_{22}(\xi) \end{pmatrix} = \\ &= \begin{pmatrix} 2\pi \sum_{k=-\infty}^{\infty} \left| \hat{\phi}_1(\xi + 2\pi k) \right|^2 & 2\pi \sum_{k=-\infty}^{\infty} \hat{\phi}_1(\xi + 2\pi k) \overline{\hat{\phi}_2(\xi + 2\pi k)} \\ 2\pi \sum_{k=-\infty}^{\infty} \hat{\phi}_2(\xi + 2\pi k) \overline{\hat{\phi}_1(\xi + 2\pi k)} & 2\pi \sum_{k=-\infty}^{\infty} \left| \hat{\phi}_2(\xi + 2\pi k) \right|^2 \end{pmatrix}. \end{aligned} \quad (12)$$

Отсюда следует, что

$$\begin{aligned} D_1(\xi) p_{11}(\xi) + D_2(\xi) p_{12}(\xi) &= 1, \\ D_1(\xi) p_{21}(\xi) + D_2(\xi) p_{22}(\xi) &= 0. \end{aligned} \quad (13)$$

При условии, что матрица $P(\xi)$ является невырожденной, из системы уравнений (13) можно легко найти $D_1(\xi)$ и $D_2(\xi)$. Как показали расчеты, именно такая ситуация и складывается в рассматриваемых ниже частных случаях. Разложив $D_1(\xi)$, $D_2(\xi)$ в ряд Фурье, можно

получить и коэффициенты $d_{1,k}$, $d_{2,k}$. Аналогично строится и $\psi_2(x)$.
Перейдем к конкретным примерам.

2. Результаты расчетов

Пусть

$$\phi_1(x) = \exp\left(-\frac{x^2}{2\sigma_1^2}\right), \phi_2(x) = \exp\left(-\frac{x^2}{2\sigma_2^2}\right), \sigma_2 > \sigma_1 > 0. \quad (14)$$

На рис. 1-4 приведены примеры графиков функций $\psi_1(x)$ и $\psi_2(x)$.

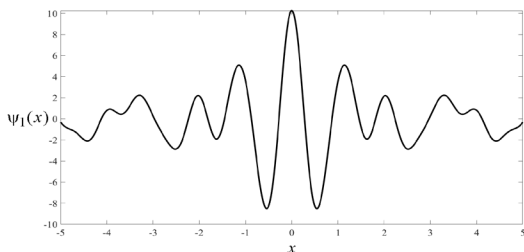


Рис. 1. График функции $\psi_1(x)$ при $\sigma_1 = 0.4$, $\sigma_2 = 0.6$

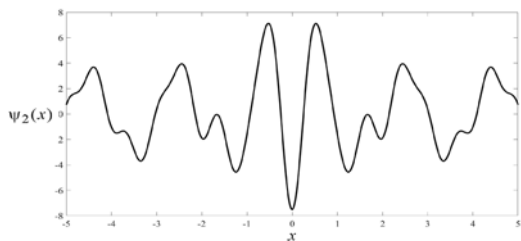


Рис. 2. График функции $\psi_2(x)$ при $\sigma_1 = 0.4$, $\sigma_2 = 0.6$

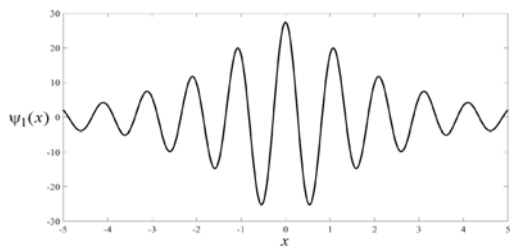


Рис. 3. График функции $\psi_1(x)$ при $\sigma_1 = 0.5$, $\sigma_2 = 1.0$

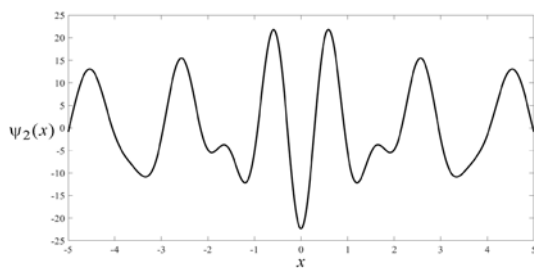


Рис. 4. График функции $\psi_2(x)$ при $\sigma_1 = 0.5$, $\sigma_2 = 1.0$

Пусть

$$\phi_1(x) = \frac{\sigma_1^2}{\sigma_1^2 + x^2}, \phi_2(x) = \frac{\sigma_2^2}{\sigma_2^2 + x^2}, \sigma_2 > \sigma_1 > 0. \quad (15)$$

На рис. 5–8 приведены примеры графиков функций $\psi_1(x)$ и $\psi_2(x)$.

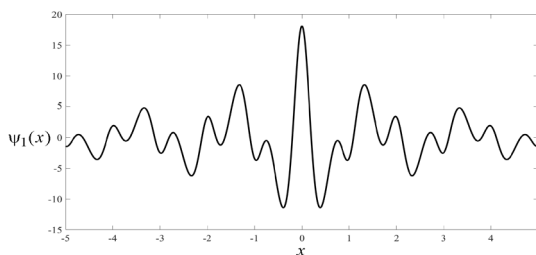


Рис. 5. График функции $\psi_1(x)$ при $\sigma_1 = 0.4$, $\sigma_2 = 0.6$

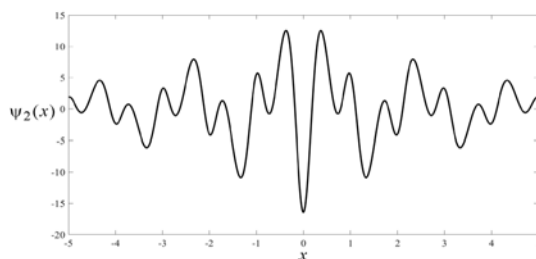


Рис. 6. График функции $\psi_2(x)$ при $\sigma_1 = 0.4$, $\sigma_2 = 0.6$

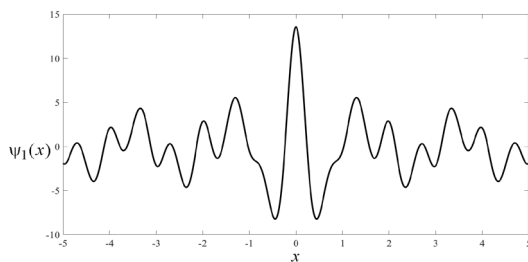


Рис. 7. График функции $\psi_1(x)$ при $\sigma_1 = 0.5$, $\sigma_2 = 1.0$

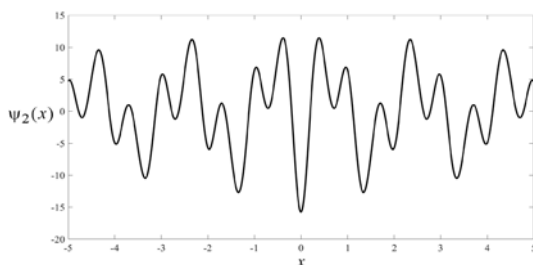


Рис. 8. График функции $\psi_2(x)$ при $\sigma_1 = 0.5$, $\sigma_2 = 1.0$

Заключение

Данная статья посвящена построению биортогональных систем для двухкомпонентных систем сдвигов. Как выяснилось, задача сводится к нахождению и обращению матрицы (12). Это подсказывает прямой путь к обобщению формул на случай многокомпонентных систем: потребуется матрица схожей структуры, но большего размера.

В перспективе планируется провести вычислительные эксперименты с реальными сигналами, используя построенные биортогональные системы, а также исследовать их поведение в зависимости от параметров σ_1 и σ_2 .

Список литературы

1. Christensen O. An introduction to frames and Riesz bases / O. Christensen. – Basel : Birkhauser/Springer, 2016. – 704 p.
2. Чуи Ч. Введение в вейвлеты / Ч. Чуи ; пер. с англ. Я. М. Жилейкина. – М. : Мир, 2001. – 412 с.
3. Никифоров П. Л. Модель электрокардиографического сигнала на основе совокупности колокольных импульсов / П. Л. Никифоров // Вестник молодых ученых. Серия техническая, 1998. – № 1. – С. 64-68.

4. Turovskiy Ya. A. New Algorithms of Generation of Mathematical Models of Electroencephalogram Signals / Ya. A. Turovskiy, E. A. Kiselev, S. V. Borzunov // 15th International Conference Management of large-scale system development (MLSD'2022). – М. : IEEE, 2022. – P. 1-4.
5. Lang K. R. Astrophysical Formulae / K. R. Lang. – New York : Springer Verlag, 1980. – 784 p.
6. Мухин К. Н. Экспериментальная ядерная физика. В 3-х т. Т. 2 / К. Н. Мухин. – СПб. : Лань, 2009. – 336 с.

Семантический разбор деловой переписки

А. А. Лазуткина

Студент бакалавр

В. В. Гаршина

Доцент

Введение

В современном мире деловая переписка играет ключевую роль в повседневной деятельности предприятий и организаций. Общение через электронную почту является неотъемлемой частью коммуникации в бизнесе. Она используется для обмена информацией между коллегами, партнёрами и клиентами.

С ростом объёма электронной почты становится сложнее обрабатывать все поступающие сообщения. На один адрес электронной почты могут приходиться сотни писем, которые необходимо разделять по темам, оперативно отвечать и направлять другим сотрудникам для дальнейшей обработки. Зачастую разбор и поиск нужных писем или авторов совершаются вручную, однако этот процесс, очевидно, можно автоматизировать и значительно ускорить, тем самым освободив сотрудников для других задач. Поэтому возникает потребность в разработке инструментов, способных автоматически обрабатывать и классифицировать письма.

В данной работе объектом исследования является электронная почта научного журнала «Вестник ВГУ. Серия: Системный анализ и информационные технологии» [1]. Поставленная задача разработки инструмента для разбора деловой переписки успешно решается при помощи методов машинного обучения. Однако для обучения алгоритмов требуется большой набор данных, который необходимо получить из архивов электронной почты журнала, изучить и разметить.

Целью данного исследования является семантический анализ электронной почты и формирование набора данных для обучения.

1. Анализ существующих решений

При анализе предметной области и поиске уже существующих решений было обнаружено, что, несмотря на распространённость проблемы, универсального решения придумано не было. Сложность

задачи заключается в том, что для каждой предметной области существуют свои категории сообщений, и необходимо иметь большое количество размеченных данных, чтобы обучить алгоритм под конкретную задачу.

Известны эффективные решения, созданные для корпоративных целей внутри компании. Например, в 2022 году была запущена система интеллектуального семантического анализа в «Ренессанс страхование», которая автоматически распределяет письма по 5 категориям и направляет заявки в соответствующую службу [2].

Также существуют инструменты, использующие искусственный интеллект для помощи в управлении электронной почтой. Один из них – сервис SaneBox [3]. Он распределяет письма по пяти стандартным папкам, куда направляются: сообщения, не требующие срочного внимания; новостные и информационные письма; спам; письма, ожидающие ответа; отложенные письма.

Однако функции подобных инструментов достаточно общие и не решают поставленной задачи разбора электронной почты научного журнала. Поэтому было решено разработать собственное решение, специализирующееся на данной предметной области и обученное на конкретных примерах писем, приходящих на электронную почту научного журнала «Вестник ВГУ. Серия: Системный анализ и информационные технологии».

2. Предобработка текста

Для формирования набора данных с электронной почты журнала были выгружены сообщения с февраля 2018 года по ноябрь 2019 года. Всего было собрано около 800 писем.

Сложность создания датасета заключается в том, что при стандартной предобработке данных – удалении пунктуации, специальных символов, стоп-слов и переводе текста в нижний регистр могут быть утеряны важные признаки. Зачастую письма содержат, помимо отправленного сообщения, и историю предыдущей переписки. Так как классифицируется именно приходящее сообщение, необходимо отделять старые ответы от нового письма. На листинге 1 показан пример письма с историей переписки, где, если удалить знаки переноса и пунктуационные символы, будет затруднительно отделить сообщения.

Следовательно, для анализа необходимо из сообщения выделить последний ответ, используя разделители в виде заголовка «Original Message» и знаков переноса строки, и только затем проводить остальную очистку текста.

Пример текста письма

Добрый день!\r\nпосылаю рецензию\r\n\r\nс уважением,\r\nИванов Иван Иванович\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\nmailto:
iii@vsu.ru\r\n\r\nтел.: +7(123)3456789\r\n\r\nмоб.тел.: +7-
1233456789\r\n\r\nskype: iii-vsu\r\n\r\n\r\n\r\n\r\n-----Original Message-----\r\n\r\nFrom: Vestnik VSU [mailto:cs-it@cs.vsu.ru] \r\n\r\nSent: Friday, February 9, 2018 5:36 PM\r\n\r\nTo: iii@rcnit.vsu.ru\r\n\r\nSubject: статья Вестник САИТ\r\n\r\n\r\nУважаемый Иван Иванович!\r\n\r\nПросьба отрецензировать статью - ОПТИМИЗАЦИЯ СТРУКТУРЫ БАЗЫ ДАННЫХ ИНФОРМАЦИОННОЙ СИСТЕМЫ КОНТРОЛЯ ОБРАЗОВАТЕЛЬНЫХ ДОСТИЖЕНИЙ УЧАЩИХСЯ А.С. Платонова Прикрепляю также, новый вариант шаблона рецензии.\r\n\r\nАндреева В.В.\r\n\r\n

Ситуация также осложняется тем, что формат подобных заголовков может отличаться. На листинге 2 показаны различные варианты блоков заголовка, присутствующие в письмах.

Листинг 2

Варианты заголовков писем

1. "-----Original Message-----\r\nFrom: Vestnik VSU [mailto:cs-it@cs.vsu.ru] \r\nSent: Friday, February 9, 2018 5:36 PM\r\nTo: iii@rcnit.vsu.ru\r\nSubject: статья Вестник САИТ\r\n\r\n"
2. "----- Пересылаемое сообщение -----\r\n>> От кого: Илья Сидоров <sidorov@mail.ru>\r\n>> Кому: cs-it <cs-it@cs.vsu.ru>\r\n>> Дата: Воскресенье, 4 февраля 2018, 12:31 +04:00\r\n>> Тема: Статья Сидоров И.Л. Комплект документов в журнал "Вестник ВГУ.\r\n>> Серия: Системный анализ и информационные технологии "\r\n"
3. "15 февраля 2018 г., 22:11 пользователь Vestnik VSU <cs-it@cs.vsu.ru>\r\n\r\nнаписал: \r\n\r\n"
4. "On 19 Feb 2018, at 21:42, Vestnik VSU <cs-it@cs.vsu.ru> wrote:\r\n\r\n"
5. "Пятница, 9 февраля 2018, 17:33 +04:00 от Vestnik VSU <cs-it@cs.vsu.ru>:\r\n\r\n" (95)
6. "On Thu, 15 Feb 2018 15:57:32 +0300 \r\nВладимир Дмитриевич Федоров <m@qmm.spbstu.ru> wrote: \r\n\r\n"
7. "15.02.2018, 14:50, "Vestnik VSU" <csit@cs.vsu.ru> "

Так, для каждого варианта были написаны регулярные выражения, выделяющие данные заголовки, а также выражения для выделения личных подписей.

Для семантического анализа текста была произведена предобработка. Она осуществлялась в несколько последовательных этапов:

1. Удаление пустых записей (None, NaN и прочее);

2. Удаление возврата каретки, специальных символов, незначащих пунктуационных символов;
3. Замена нескольких переносов строк одним;
4. Удаление заголовков пересылаемых сообщений;
5. Удаление личных подписей;
6. Удаление почтовых адресов и url-адресов;
7. Приведение текста в нижний регистр и удаление стоп-слов(предлоги, местоимения и частицы, а также иностранные слова);
8. Удаление цифр, пунктуации, лишних пробелов.
9. Лемматизация (приведение слов к нормальной форме).

3. Семантический анализ данных

Проведем анализ полученных данных, чтобы в дальнейшем использовать полученную информацию при классификации.

Был проведен частотный анализ встречаемости слов в тексте. На рис. 1 представлена визуализация частоты распределения слов в тексте, были выведены 30 самых встречаемых слов.

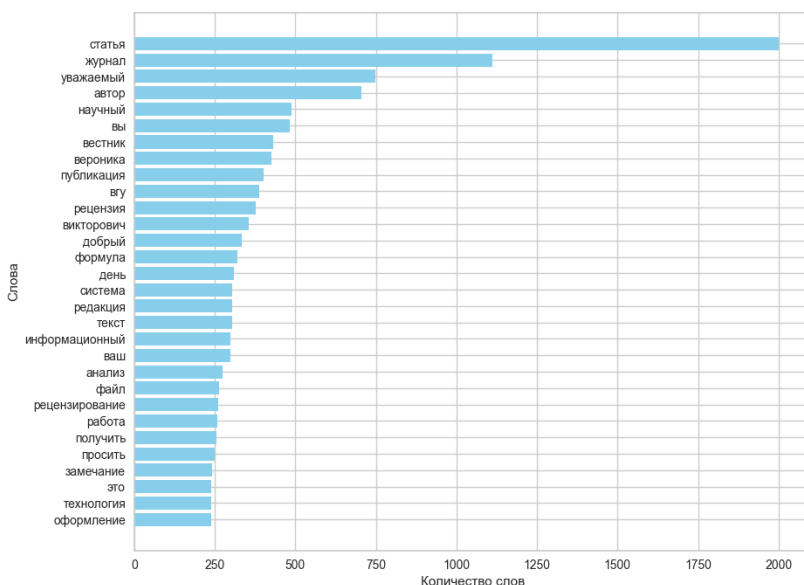


Рис. 1. Частотный анализ

Можно заметить, что самые частые слова – «статья», «журнал», «уважаемый» и «автор» не несут в себе полезной нагрузки для решения

После удаления самых частых и малозначащих слов в тексте, с помощью библиотеки WordCloud [4] было сформировано облако слов для наглядного представления текстовых данных. Визуализация представлена на рис. 2.

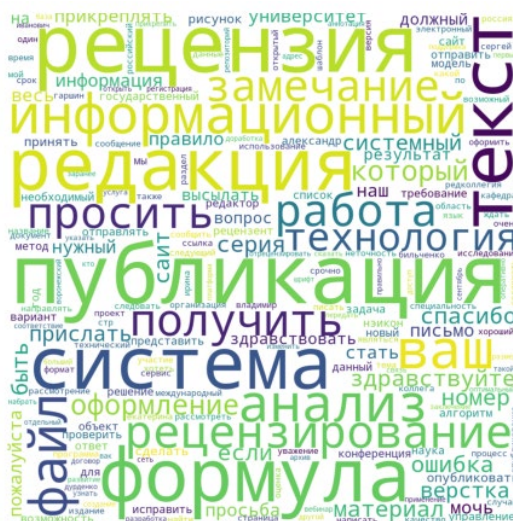


Рис. 2. Облако слов

С помощью библиотеки `Natasha` [5] и `Yargy`-парсера на основе выделенных слов текст письма относился к той или иной категории. Фрагмент кода для определения категории показан на листинге 3.

Была использована библиотека Naves из проекта Natasha для получения эмбедингов слов для последующей классификации. С помощью метода снижения размерности t-SNE [6], использующегося при визуализации многомерного набора данных, эмбединги слов из предоставленных текстов были отображены на графике, показанном на рис. 3.

4. Формирование набора данных

Для решения поставленной задачи необходимо сформировать размеченный набор данных. Из атрибутов каждого письма было выбрано несколько признаков: отправитель письма, дата отправления, тема письма, текст письма, название и тип приложенных файлов, если они есть. С помощью семантического анализа из текста письма выбирались именованные сущности, такие как личные подписи, обращения, имена собственные и аббревиатуры. Проанализировав данные, было решено в качестве признаков для обучения оставить тему письма, название приложенных файлов и текст письма, очищенный от именованных сущностей, заголовков и подписей, так как имена и названия могут служить помехой при классификации.

Так как зачастую при подаче статьи или доработке может отсутствовать непосредственно текст письма, важно это учитывать и использовать в таком случае другие атрибуты, такие как тема и название вложений.

Разметка собранного набора данных проводилась с использованием парсинга, регулярных выражений и частично вручную, для более сложных случаев.

При публикации научной статьи в журнале сообщение проходит несколько этапов:

- Подача статьи – первое сообщение с просьбой принять статью к публикации.
- Проверка материалов – проверка правильности оформления и содержания статьи.
- Доработка – в случае, если в статье были найдены ошибки, она отправляется обратно авторам на доработку и исправление.
- Готово к рецензированию – статья оформлена верно и может быть направлена рецензенту.
- Рецензирование – рецензент изучил статью и направил рецензию.
- Принята – на статью была написана положительная рецензия и работа может быть опубликована в журнале.
- Отклонена – статья не прошла рецензирование, либо авторы отказались от участия.

Помимо этого, на почту журнала приходят различные вопросы от авторов, и письма, не относящиеся напрямую к публикации статей, к примеру, приглашения на конференции, реклама, обсуждение деловых вопросов.

Таким образом, данные были разделены на 9 категорий: «подача статьи», «проверка статьи», «доработка», «готово к рецензированию», «рецензирование», «отклонена», «публикация», «другое», «вопросы».

Заключение

В данной работе была исследована проблема формирования набора данных для классификации электронных писем научного журнала «Вестник ВГУ. Серия: Системный анализ и информационные технологии». Были выделены существенные и несущественные признаки, использующиеся при классификации текста писем. Семантический анализ показал связи между различными словами, используемыми при деловой переписке по электронной почте. В результате работы был сформирован размеченный набор данных, который в дальнейшем может быть использован для настройки и обучения алгоритмов классификации.

Список литературы

1. Вестник ВГУ. Серия: Системный анализ и информационные технологии [Электронный ресурс] : науч. журн. – Режим доступа: <https://journals.vsu.ru/sait>
2. Робот-сортировщик: как обучить модель, читающую 1500 писем в день [Электронный ресурс] : – Режим доступа: <https://vc.ru/marketing/773660-robot-sortirovshchik-kak-obuchit-model-chitayushchuyu-1500-pisem-v-den>
3. SaneBox [Электронный ресурс] : – Режим доступа: <https://vc.ru/marketing/773660-robot-sortirovshchik-kak-obuchit-model-chitayushchuyu-1500-pisem-v-den>
4. A Wordcloud in Python [Электронный ресурс] : – Режим доступа: <https://peekaboo-vision.blogspot.com/2012/11/a-wordcloud-in-python.html>
5. Natasha [Электронный ресурс] : – Режим доступа: <https://github.com/natasha/natasha>
6. sklearn.manifold.TSNE [Электронный ресурс] : – Режим доступа: <https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html>

Закон Ципфа для решения лингвистических и социально-экономических задач

С. С. Ламзин

Студент бакалавр

М. Е. Залыгаева

Старший преподаватель

Введение

Инфометрия – это наука об измерении количественных характеристик информации. Востребованность её в современном мире велика, так как в цифровую эпоху с развитием интернета объёмы информации и потребность в ней увеличились колоссально.

Одним из первых инфометрических законов стал закон Ципфа, описанный американским лингвистом Джорджем Ципфом. Им он показал закономерность распределения слов в английском языке.

В результате многочисленных эмпирических исследований оказалось, что данный закон может применяться не только в лингвистике, но и в других областях, занимающихся анализом статистических данных, например в экономической сфере, описывая распределение доходов людей, а также роста населения городов[1].

Целью данной работы является проверка выполнимости и применение закона Ципфа для решения лингвистических задач по изучению иностранных языков и анализ текстов на удобочитаемость, а также для задач экономического планирования на основе распределения населения городов.

1. Открытие закона Ципфа и его проявления в реальной жизни

Закон Ципфа – это частотная закономерность всякого семантически коррелированного сообщения. Процесс изучения языка в первую очередь включает в себя расширение словарного запаса, и Ципф искал способ оптимизировать данный процесс. Самым быстрым способом это сделать будет освоение самых часто встречаемых слов. Как правило это короткие слова, быстро и легко произносимые, хорошо воспринимаемые на слух.

В лингвистике закон Ципфа сыграл большую роль, так как был найден параметр, исследуя который можно было получать информацию

о внутреннем устройстве языка. Таким параметром стала частота встречаемости слов в тексте.

Французский математик Бенуа Мандельброт модифицировал изначальный закон, привлекая теорию оптимизации кодирования информации. Конечный закон получил название закон Ципфа-Мандельброта.

$$\rho(i) \cdot (B + i)^\gamma = K, \quad (1)$$

где $\rho(i)$ - относительная частота i -го слова в словарном списке, $K, B, \gamma = \text{const}, i = 1, 2, \dots$

В частности при $B = 0, \gamma = 1$, этот закон переходит в закон Ципфа.

При этом был обнаружен факт: закон Ципфа-Мандельброта хорошо согласуется с частотными данными отдельных литературных произведений с чёткой сюжетной линией и практически не выполняется для частотных данных по произвольным лексическим выборкам, не обладающих смысловой корреляцией. Объяснение при этом выглядит так. Язык – это средство коммуникации, и в ходе его функционирования действуют закономерности, справедливые для любого канала связи. То есть с одной стороны, увеличение количества слов при передаче информации удлиняет время коммуникации, с другой стороны, оно уменьшает вероятность ошибки при передаче сообщения, а значит сокращает время коммуникации за счёт того, что нет необходимости в повторной передаче.

Кроме того закономерности типа (1) были обнаружены в таких рангово-степенных распределениях как: распределения богатства и доходов у людей (В. Парето); распределение численности населения городов (Ф. Ауэрбах); количество публикаций учёных (А. Лотка); статистика чрезвычайных ситуаций с ранжировкой по числу погибших и так далее[2].

2. Математическая суть

Математически закон Ципфа можно объяснить так: если все слова языка, или достаточно длинного текста упорядочить по убыванию частотности их использования, то частотность i -го слова в таком списке окажется приблизительно обратно пропорциональной его порядковому номеру i , иначе рангу этого слова. То есть второе по используемости в тексте слово встречается примерно в два раза реже, чем первое, третье слово – в три раза реже, и так далее,

$$f_i = \frac{C}{\text{rank}(i)}, \quad (2)$$

где $C = \text{const}$, f_i - частота появления i -го слова, $\text{rank}(i)$ – ранг i -го слова при упорядочивании по убыванию частот, $i = 1, 2, \dots, m$, m – число разных слов.

Иначе,

$$f_i = \frac{n_i}{n}, \quad (3)$$

где n_i - число i -го слова в тексте, n – общее число слов в тексте, f_i - частота появления i -го слова.

Или,

$$f_i = \frac{C}{\text{rank}(i)^\alpha}, \quad (4)$$

где $C \approx 0,1$ и $\alpha \approx 1$ [3].

3. Использование в экономическом анализе

На современном этапе развития экономической науки закон Ципфа может использоваться как инструмент оценки городских иерархий. На практике это означает, что при равномерном развитии городов их размер следует за распределением Парето с коэффициентом, равным единице, то есть наибольший город должен быть примерно в два раза крупнее второго, в три раза третьего, и так далее.

При проведении социально-экономической политики в регионах важное значение имеет понимание процессов концентрации ресурсов, населения и других субъектов экономической деятельности, предприятий. В современном урбанизированном обществе эта концентрация чаще всего происходит в городах, местах сосредоточения экономической активности. Всё это усиливает дифференциацию регионального развития в стране. Причём экономическая активность означает не только размещение промышленного производства, сферы услуг и прочих видов деятельности, но и населения, трудовых ресурсов, инвестиций и благосостояния.

При принятии решения предпринимателем о размещении какого-либо производства в том или ином регионе и городе, в первую очередь рассматриваются факторы, которые обеспечили бы высокую рентабельность в будущем. Такими факторами являются доступность трудовых ресурсов, их качество и стоимость, уровень налогов и возможность получить субсидии, развитие инфраструктуры и транспорта, социальные объекты.

Исходя из этого региональные правительства со своей стороны должны прогнозировать качественные характеристики и динамику вышеперечисленных факторов, чтобы проводить такие мероприятия, которые смогли бы способствовать повышению привлекательности территории.

Для описания эмпирической зависимости между размером города и его рангом в иерархии городов региона или страны, Д. Ципф модифицировал первоначальную формулу, предложенную Ф. Ауэрбахом:

$$\log P = \log K - q \log R, \quad (5)$$

где q – коэффициент Ципфа (при $q = 1$ распределение размера города соответствует закону Ципфа), $K = \text{const}$, P – численность населения города, R – ранг города [4].

4. Анализ текстов

Для проверки выполнимости закона Ципфа на различных естественных языках были выбраны несколько крупных и известных литературных произведений. Анализ проводился с помощью написанной на языке Python программы, источник текстов – бесплатный архив "Gutenberg Project".

Первым примером будет роман Германа Мелвилла "Моби Дик", написанный на английском языке. Общее количество слов в тексте – 215864. Пример представлен на рис. 1.

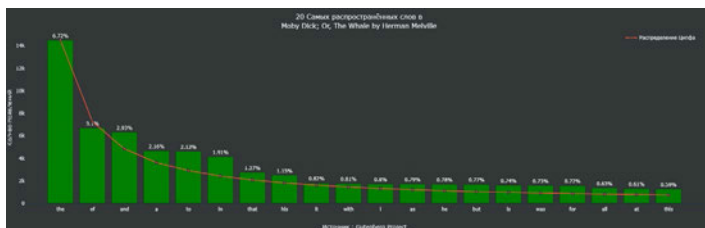


Рис. 1. Частотно-ранговое распределение слов в романе "Моби Дик"

Зелёная гистограмма – реальное частотное распределение слов в тексте, красная кривая – идеальное распределение Ципфа для данного текста.

Как видно из вышеперечисленного графика, закон Ципфа выполняется для первых двух слов почти точно, для остальных кривая лежит чуть выше идеальной кривой Ципфа, но со слова "it" кривая близко стремится к гистограмме. Также можно заметить, что самые

распространённые слова в тексте – это местоимения и артикли. К тому же такое близкое соответствие реального частотного распределения закону Ципфа говорит о том, что перед нами естественный текст и его удобочитаемость высока.

Второй пример – "Фауст" Иоганна Гёте, написан на немецком языке. Общее количество слов – 33709. Пример представлен на рис. 2.

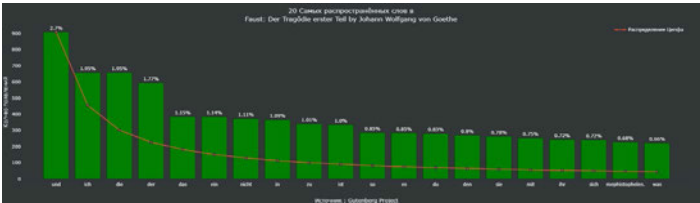


Рис. 2. Частотно-ранговое распределение слов в трагедии "Фауст"

Закон Ципфа довольно плохо выполняется в данном тексте. Связано это с особенностями немецкого языка и большим количеством часто употребляемых артиклей и частиц. Таким образом можно получить ясную картину наиболее важных слов для изучения немецкого языка и понимания смысла написанных на нём сообщений.

Третий пример – роман "Любящее созвездие" Нино Баэна, на искусственном языке эсперанто. Общее количество слов – 16963. Пример представлен на рис. 3.

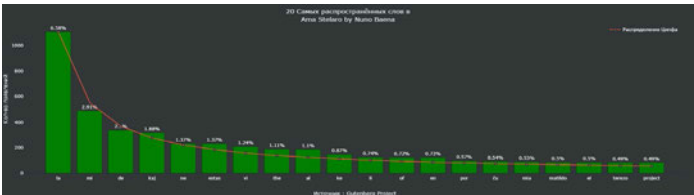


Рис. 3. Частотно-ранговое распределение слов в романе "Любящее созвездие"

Несмотря на относительно небольшое количество слов, закон Ципфа выполняется для данного произведения практически идеально, что свидетельствует о том, что эсперанто, будучи искусственным языком, использует простой синтаксис, соответствующий теории оптимизации кодирования информации. Само произведение также, скорее всего, обладает высокой удобочитаемостью.

5. Анализ распределений населения городов

Несколько модифицировав код предыдущей программы проведём анализ крупных городов в нескольких странах и проверим выполнение закона Ципфа в распределении населения этих городов.

Первый пример – невыполнение закона Ципфа характерно для населения городов Великобритании. Страна сильно централизована и Лондон поглощает большее количество квалифицированной рабочей силы, как из-за внутренней миграции, так и внешней. Из-за этого создаётся сильный дисбаланс в концентрации ресурсов и неравномерное развитие остальных городов, как субъектов экономики. Пример представлен на рис. 4.

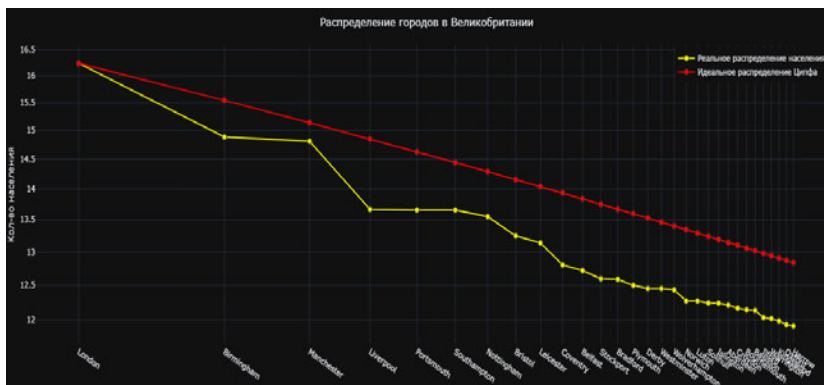


Рис. 4. Частотно-ранговое распределение населения городов в Великобритании в 2022 году

Красный график – идеальное распределение Ципфа, жёлтый график – реальное распределение. График представлен в логарифмическом масштабе.

Второй пример – частичное выполнение закона Ципфа характерно для городов Германии, так как здесь, в отличие от прошлой страны более сильны федеральные позиции в управлении, что показывает более приемлемые результаты. Кроме трёх крупных городов: Штутгарта, Мюнхена и Гамбурга закон Ципфа более менее выполняется. Пример представлен на рис. 5.

В качестве примера практически идеального выполнения закона Ципфа можно взять данные по городам США в 1950 году, до Иммиграционного акта 1965 года население городов, а значит и их экономическое развитие здесь проходило равномерно. Пример представлен на рис. 6.

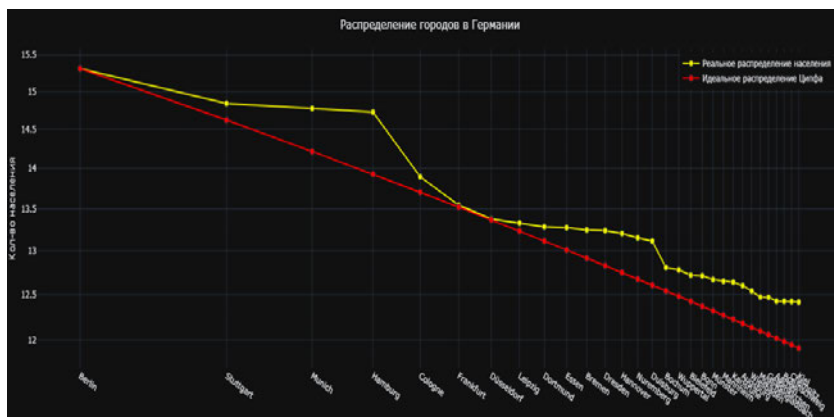


Рис. 5. Частотно-ранговое распределение населения городов в Германии в 2022 году

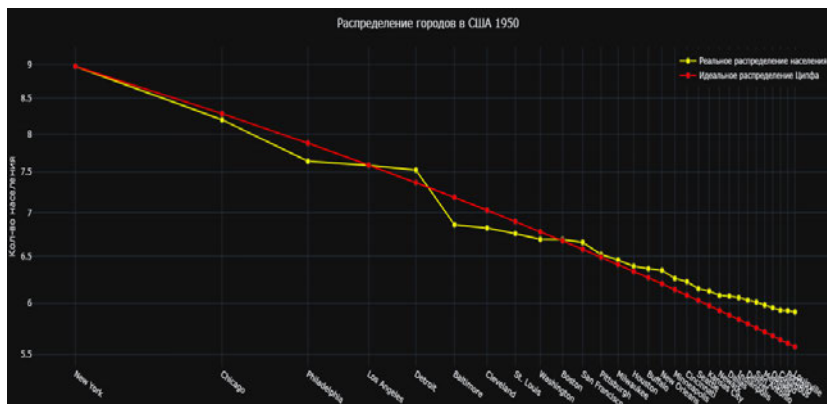


Рис. 6. Частотно-ранговое распределение населения городов в США в 1950 году

Заключение

Данная статья посвящена использованию закона Ципфа для решения некоторых лингвистических и социально-экономических задач. В результате была изучена теория частотно-рангового распределения и его применение в инфометрическом анализе, была выполнена программная реализация закона на языке Python. С его помощью была проверена выполнимость закона Ципфа на текстах разных языков, сделаны выводы о синтаксической сложности самих языков и

удобочитаемости приведённых текстов. Также был проведён анализ населения городов некоторых стран, а также целесообразность применение закона Ципфа для экономического планирования. В дальнейшем эту программу можно использовать и для анализа размеров городов по отдельным макрорегионам, например по федеральным округам.

Список литературы

1. Li W. Random texts exhibit Zipf's-law-like word frequency distribution / W. Li // IEEE Transactions on Information Theory. – 1992. – Issue 6. – P. 1842-1845.
2. Фирстов В. Е. Из истории стенографии и интерпретации закона Ципфа-Мандельброта в учебном процессе / В. Е. Фирстов, Р. А. Иванов // Ярославский педагогический вестник. – 2013. – № 4. – С. 14-20.
3. Растворцева С. Н. Закон Ципфа в городах России: анализ новых показателей / С. Н. Растворцева, И. В. Манаева // Экономика региона. – 2020. – № 3. – С. 935-947.
4. Манаева И. В. Закон Ципфа в экономической науке: методы оценки / И. В. Манаева // Экономический рост Республики Беларусь: глобализация, инновационность, устойчивость : материалы XI Междунар. науч.-практ. конф. (Минск, 17 мая 2018 г.) / отв. ред. В. Н. Шимов. – Минск : БГЭУ, 2018. – С. 286-288.

Статистический и нейросетевой подходы к прогнозированию временных рядов

А. Г. Линьков

Студент бакалавр

М. Е. Залыгаева

Старший преподаватель

Введение

Временной ряд – это последовательность наблюдений за определенным явлением во времени. Они широко распространены в инфографике – графическом представлении информации. Из этого следует востребованность и актуальность во многих современных сферах: от статистики до биржевых торгов.

Цель данной работы состоит в сравнении методов анализа и предсказания временных рядов, для достижения которой были поставлены такие задачи как изучение теории временных рядов, моделей SARIMAX и GRU, применение этих моделей к реальным финансовым данным для прогноза и последующего сравнительного анализа и интерпретации результатов.

1. Теория временных рядов

Структурно временной ряд можно представить следующей формулой:

$$Y(t) = f(t) + S(t) + C(t) + \epsilon(t), \quad (1)$$

где t – момент времени, $Y(t)$ – значение временного ряда, $f(t)$ – составляющая тренда, $S(t)$ – сезонная составляющая, $C(t)$ – циклическая составляющая, $\epsilon(t)$ – случайная составляющая (шум).

Для сглаживания временных рядов с целью выделения основных тенденций, циклов используют функции из семейства скользящих средних. Их значения в каждой точке определения равны некоторому среднему значению исходной функции за предыдущий период. Зачастую используется простое скользящее среднее (simple moving average), которое вычисляется по формуле:

$$SMA_t = \frac{1}{n} \sum_{i=1}^{n-1} p_{t-i}, \quad (2)$$

где n – размер окна, p – значение исходной функции.

Стационарность временного ряда (в широком смысле) означает, что он имеет неизменное во времени математическое ожидание (отсутствие тренда), а также ковариация между элементами будет зависеть только от лага (расстояния между элементами во времени).

Функция $\gamma(h)$ как функция от лаговой переменной называется автоковариационной функцией временного ряда и вычисляется по следующей формуле:

$$\gamma(-h) = \text{cov}(x_t, x_{t-h}) = \text{cov}(x_t, x_t) = \text{cov}(x_t, x_{t+h}) = \gamma(h), \quad (3)$$

где h – лаговая переменная, $\text{cov}(x, x)$ – ковариация между двумя элементами временного ряда.

С её помощью рассчитывается функция автокорреляции $\rho(h)$

$$\rho(h) = \frac{\text{cov}(x_t, x_{t+h})}{\sqrt{\text{Var}(x_t) \cdot \text{Var}(x_{t+h})}} = \frac{\gamma(h)}{\sqrt{\gamma(0) \cdot \gamma(h)}} = \frac{\gamma(h)}{\gamma(0)}, \quad (4)$$

2. Модель SARIMAX

Объединяя подходы авторегрессии (AR) и скользящего среднего (MA) получаем модель ARMA, которую в общем виде можно представить как:

$$x_t = \mu + \sum_{j=1}^p \phi_j x_{t-j} + u_t + \sum_{s=1}^q \theta_s u_{t-s}, \quad (5)$$

где x – значение элемента, p – лаг для AR, q – лаг для MA, u – среднее значение временного ряда, ϕ, θ – веса модели.

Реальные данные, в частности финансовые, редко бывают стационарными, а для применения модели необходима стационарность ряда. Для перехода к стационарному ряду можно продифференцировать ряд путём нахождения конечных разностей.

Операция дифференцирования или первая конечная разность обозначается:

$$\Delta x_t = x_t - x_{t-1} = (1 - L)x_t, \quad (6)$$

где x_t – значение ряда, L – лаговый оператор.

Таким образом, вместо исходного ряда будет рассмотрено его приращение за один период. Дифференцирование приводит ряд к

стационарному относительно приращений. Порядок конечных разностей, необходимых для того, чтобы ряд стал стационарным, даёт интегральную составляющую I , с помощью которой наша модель модифицируется до модели ARIMA.

Если учесть сезонную составляющую ряда S и экзогенные факторы X , влияющие на модель, но не учитываемые напрямую, получаем модель SARIMAX, которая и будет использоваться для дальнейшего анализа данных.

3. Модель Gated Recurrent Unit

Нейросети – математические модели, созданные по образу и подобию биологических нейронных сетей, представляющие собой систему соединённых искусственных нейронов, которые, обрабатывая информацию, взаимодействуют между собой и обучаются посредством изменения своих весовых параметров. Нейронные сети стремительно развиваются и уже вошли в повседневную жизнь, позволяя выполнять широкий спектр задач. Существует множество архитектур нейросетей, но под нашу задачу моделирования временного ряда необходима рекуррентная архитектура, отличающаяся наличием механизма обратной связи для учёта контекста и истории входных данных.

GRU (Gated Recurrent Unit) – разновидность рекуррентных нейронных сетей, разработанная для решения проблемы затухания градиента при обучении. Модель использует механизм вентилях. Помимо входного и выходного, в сети имеется вентиль сброса, позволяющий забывать устаревшую информацию, и вентиль обновления, регулирующий степень влияния прошлых состояний и актуальной входящей информации.

Вентили описываются следующими формулами:

$$z_t = \sigma_g(W_z x_t + U_z h_{t-1} + b_z), \quad (7)$$

$$r_t = \sigma_g(W_r x_t + U_r h_{t-1} + b_r), \quad (8)$$

$$h_t = z_t \circ h_{t-1} + (1 + z_t) \circ \sigma_h(W_h x_t + U_h (r_t \circ h_{t-1}) + b_h), \quad (9)$$

где x – входной вектор, h – выходной вектор, z – вектор вентиля обновления, r – вектор вентиля обновления, σ_g – сигмоидальная функция активации, σ_h – гиперболическая функция активации, W, U – матрицы весов, b – вектор весов.

На рис. 1 представлена архитектура сети GRU.

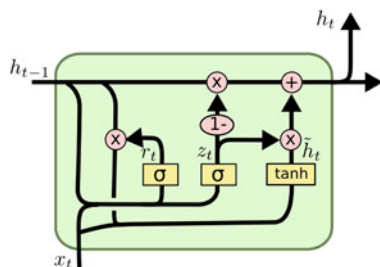


Рис. 1. Архитектура GRU

4. Прогнозирование временного ряда с помощью моделей SARIMAX и GRU

Для сравнительного анализа в качестве датасета были выбраны котировки акций компании Tesla. Анализ проводился посредством программы, написанной на языке Python.

Чтобы воспользоваться моделью SARIMAX, необходимы семь гиперпараметров. Первым из них будет сезонность. Поскольку рассмотрены биржевые показатели, можно выдвинуть гипотезу о том, что параметр равен длине недели – семи. Перед проверкой гипотезы нужно убедиться в наличии сезонности как таковой. На рис. 2 представлена сезонная декомпозиция временного ряда.

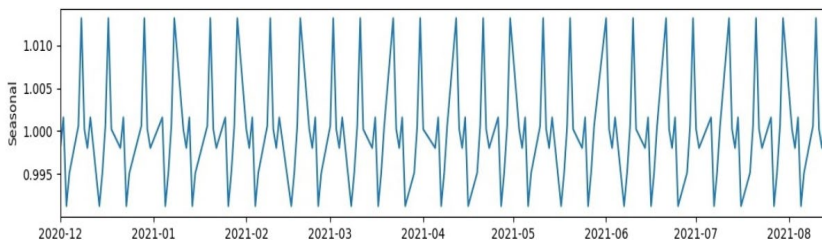


Рис. 2. Сезонная декомпозиция временного ряда

Четыре пика в месяц наталкивают на принятие гипотезы, но необходимо проверить точные значения с помощью функции автокорреляции, результат работы которой отражён на рис. 3.

Небольшой пик на значении семь позволяет принять гипотезу.

Оставшиеся шесть гиперпараметров будут найдены автоматическим перебором и отбором по наилучшей скорости работы в сочетании с информационным критерием Акаике с помощью функции. Для этого необходима обучающая выборка, которая была отделена от

тестовой незадолго до окончания восходящего тренда, чтобы проверить, уловит ли модель последующий нисходящий.

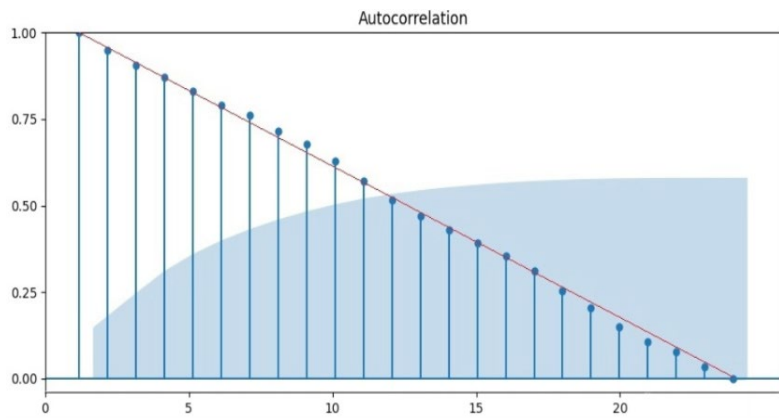


Рис. 3. График функции автокорреляции

После обучения модель была применена как к тестовым, так и к обучающим данным, чтобы в дальнейшем найти метрики, а также проведен прогноз ещё на 15 дней, что представлено на рис. 4.

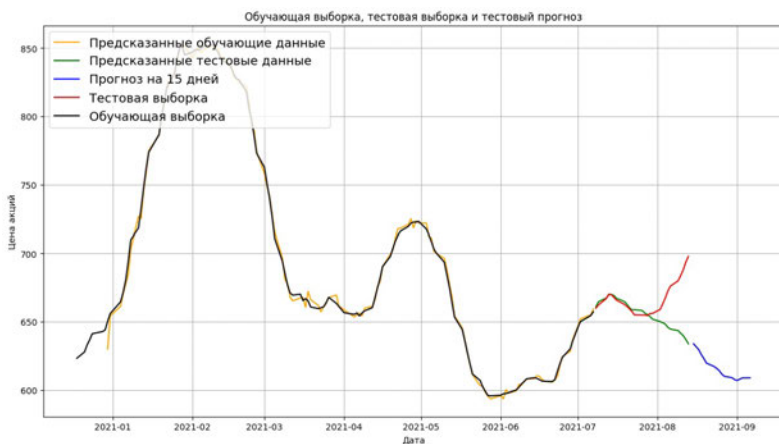


Рис. 4. Прогноз моделью SARIMAX

Прогноз обучающей выборки почти полностью совпадает с исходными данными, что свидетельствует о том, что интервал

смоделирован хорошо. Тестовый прогноз на протяжении 15-17 дней также близок, но далее происходит расхождение, в связи с неожиданным для модели восходящим трендом. Стоит отметить, что прогноз на 15 дней в биржевых торгах уже является отличным результатом.

Приступим к прогнозированию с помощью GRU. Предполагая гибкость и адаптивность рекуррентной сети, для исследования её предельных возможностей решено взять больший интервал с восходящим трендом после относительной стационарности. Также для проверки на прочность соотношение обучающих данных к тестовым взято 3:2 и опущено сглаживание временного ряда.

Опытным путём установлено, что лучше всего модель проявляла себя с тремя слоями по 32 нейрона. Итого в сети вышло порядка шестнадцати тысяч параметров. Обучение проводилось в двести эпох. Результат предсказания представлен на рис. 5.



Рис. 5. Прогноз моделью GRU

Прогноз обучающих данных вышел не столь аккуратным, как в случае модели SARIMAX, но для данных без сглаживания это очень хороший результат. Модель уловила все внезапные скачки. То же самое можно сказать и про тестовые данные, несмотря на их возросший объем. Однако дальнейший прогноз на тридцать дней получился очень

сглаженным и выглядит весьма условным, хоть интуитивно и следует тренду, которым хотелось бы продолжить график. Это может быть связано с переобучением, зависимостью от предыдущих данных и отдалённостью от обучающей выборки.

5. Сравнительный анализ результатов

Для сравнения моделей были найдены метрики среднее абсолютное отклонение (MAE), среднее квадратическое отклонение (MSE) и его квадратный корень (RMSE). Для тестового прогноза SARIMAX рассмотрено меньше значений, до расхождения графиков, чтобы результаты были сопоставимы с GRU. Результаты представлены в таблице.

Таблица

Метрики прогнозов

Модель	SARIMAX		GRU	
Выборка	Обучающая	Тестовая	Обучающая	Тестовая
MSE	11,21	1507,2	1074,68	413,55
RMSE	3,34	38,82	33,78	20,33
MAE	2,35	34,72	25,43	16,71

Подводя итоги исследования, можно подметить, что SARIMAX хорошо подходит для моделирования стационарных относительно приращений временных рядов, в связи с чем она не готова к внезапным изменением тренда. Рекуррентные нейросети являются адаптивным и гибким инструментом, что позволяет пользоваться ими при резких скачках трендов даже без сглаживания, но при этом дальнейшие прогнозы могут выходить весьма неточными в связи с переобучением.

Стоит отметить, что финансовая сфера является весьма непредсказуемой и переменчивой, претерпевающей на себе влияние множества факторов, которые сложно учесть. И, несмотря на это, обе рассмотренные модели проявляют себя достаточно неплохо при правильном подходе к подготовке подходящих данных.

Заключение

Данная статья посвящена реализации программы для анализа и прогнозирования временных рядов. Рассмотрены алгоритмы статистической модели SARIMAX и рекуррентной нейросетевой модели архитектуры GRU, процесс подготовки данных, на которых модели и были обучены. Модели были применены к реальным данным из финансовой сферы, на основании чего были получены удобоваримые прогнозы и был проведен сравнительный анализ результатов с помощью

статистических метрик. В дальнейшем предполагается расширение программы с целью добавления иных моделей: как статистических, так и нейросетевых.

Список литературы

1. Кизбикенов К. О. Прогнозирование и временные ряды : учеб. пособие / К. О. Кизбикенов. – Барнаул : АлтГПУ, 2017. – 113 с.
2. Тихонов Э. Е. Методы прогнозирования в условиях рынка : учеб. пособие / Э. Е. Тихонов. – Невинномысск : СевКавГТУ, 2006. – 221 с.
3. Бокс, Дж. Анализ временных рядов, прогноз и управление / Дж. Бокс, Г. М. Дженкинс. – М. : Мир, 1974. – 406 с.
4. Эконометрия : учебник / В. И. Суслов [и др.] ; отв. ред. Г. М. Мкртчян. – Новосибирск : СО РАН, 2005. – 744 с.
5. Гафаров, Ф. М. Искусственные нейронные сети и приложения : учеб. пособие / Ф. М. Гафаров, А. Ф. Галимянов. – Казань : Изд-во Казан. ун-та, 2018. – 121 с.

Виртуальные потоки в Java'21

К. П. Луговской

Студент бакалавр

Е. Д. Проскуряков

Студент магистр

Д. И. Соломатин

Старший преподаватель

Введение

В наше время нагрузка на приложения с каждым годом все возрастает. Для корректной работы систем используются все больше разных подходов. Основой многих из них являются потоки, которые помогают распределять эту нагрузку. При этом их реализация разнится в языках программирования и имеет собственные особенности. Настало время, когда многоядерные процессоры становятся достаточно недорогими для установки в настольных системах среднего уровня. И не случайно разработчики отмечают все больше сообщений об ошибках, связанных с многопоточностью в своих проектах [1].

Не так давно в Java появился новый вид потоков – виртуальные. Они имеют некоторые характерные отличия от стандартной реализации и могут служить для оптимизаций и переосмысления подходов к написанию кода.

Проблемы, которые призваны решать виртуальные потоки – это ограниченность количества их аналогов, занимаемая память, излишняя сложность написания некоторых асинхронных решений.

Рассматриваться данное нововведение будет на основании 21 версии Java, однако его реализация была и в более ранних версиях.

1. Происхождение виртуальных потоков

Для того чтобы понять, как устроены виртуальные потоки, необходимо рассмотреть стандартные аналоги. Все начинается с ядра процессора, на котором выполняются системные процессы, в них находятся системные потоки, а стандартные потоки Java оборачиваются вокруг системных, так что один поток Java соответствует одному системному. Ключевое отличие виртуальных потоков в том, что они не

ограничены отношением один к одному, то есть несколько виртуальных потоков могут делить пространство одного системного.

2. Реализация концепции многопоточности

Процессы имеют одно адресное пространство, а потоки могут совместно использовать адресное пространство родительского процесса. Системный процесс является экземпляром выполняющейся программы, это позволяет использовать несколько потоков внутри процесса для ускорения выполнения программы, тем самым реализуя подход многопоточности. Ядро может поддерживать одновременно выполнение небольшого количества потоков, обычно это число ограничивается десятками. Например, в архитектуре процессора Niagara (Oracle) это ограничение составляет 16 потоков или меньше, в зависимости от версии.

Однако для работы приложения могут потребоваться сотни и тысячи потоков для параллельного выполнения, в таком случае ядро берет на себя обязанности создать иллюзию параллелизма, которая будет правдоподобна. Делает оно это путем контекстного переключения. Суть в том, что ядро само решает, когда и какой поток выполняется, то есть производит переключение, но делает это очень быстро, а в приложении это выглядит как параллельное выполнение.

3. Аналоги в других языках

Стоит отметить, что виртуальные потоки не являются уникальным нововведением, существующим исключительно в Java, их реализации также присутствуют в:

- Go (Goroutines);
- Kotlin (Coroutines);
- Haskell (Haskell threads).

4. Сравнение ресурсоемкости

Факт того, что виртуальные потоки легковесней очевиден из того, что они, как и стандартный Java-поток делят пространство родительского процесса, но при этом их может быть несколько. Тесты показывают, что виртуальные потоки могут существенно масштабироваться до сотен тысяч или миллионов, тогда как Java-потоки, поддерживаемые ОС, могли масштабироваться только до пары тысяч [2].

На высоком уровне поток управляется и планируется операционной системой, в то время как виртуальный поток управляется и планируется виртуальной машиной [3]. Следовательно, создание

виртуальных потоков проходит легче и быстрее, так как не требует взаимодействий с системой.

5. Реализация в Java

В языке Java стало традицией поддерживать обратную совместимость своих разработок, поэтому и виртуальные потоки не стали исключением. Изменение обычных потоков на виртуальные происходит за счет минимальных изменений в коде. Поэтому многие решения будут легко переписываться на уровне кода.

Для обычных потоков в Java существуют пулы потоков, которые позволяют сразу создать несколько потоков, а затем нагружать их по необходимости, это необходимо т.к. создание обычного потока ресурсоемкая операция. Когда используется пул потоков, пишется параллельный код в виде параллельных задач и затем они отправляются на выполнение экземпляру пула потоков. Этот экземпляр управляет несколькими повторно используемыми потоками для выполнения этих задач [4]. В Java существует Executor, которому делегируется создание пулов.

Виртуальные потоки не имеют таких ограничений и создаются из обычных, однако все равно имеют свой Executor, который может создавать виртуальный поток на каждое задание, не используя при этом концепцию пулов.

Также одним из преимуществ виртуальных потоков является то, что они основаны на неблокирующей модели, в отличие от своих предшественников, использующих блокирующую модель. Это открывает новые возможности при решении задач, связанных с асинхронностью. При этом виртуальные потоки представляются более простым решением с точки зрения лаконичности кода, в отличие от других решений по асинхронности, но не могут стать однозначной их заменой, а лишь послужат еще одним удобным способом для таких задач.

В дополнении к этому важно сказать, что виртуальные потоки являются относительно новой технологией в Java, поэтому могут присутствовать проблемы с интеграцией с другими решениями. Также эта технология активно дорабатывается и еще не сформировалось четкое представления по ее использованию, а следовательно, требуется использовать ее с пониманием того, в чем она хороша.

6. Тестирование

Первым было проведено тестирование производительности, для этого использовалась библиотека JMH, которая предоставляет ряд полезных механизмов, в том числе обход оптимизаций компилятора,

которые могут повлиять на конечный результат. Тесты представляли собой имитирование бизнес-задач за счет искусственной загрузки процессора. Выполнение этих задач делегировалось на обычные потоки и виртуальные, при этом также были задействованы пулы потоков для честной конкуренции.

Результаты тестирования представлены на рис. 1-4, выводы можно сделать следующие:

- Виртуальные потоки создаются в несколько раз быстрее;
- На малых задачах преимущество за виртуальными потоками;
- На средних задачах разница становится все меньше;
- На тяжелых задачах разница и вовсе исчезает.

Benchmark	Mode	Cnt	Score	Error	Units
BenchmarkThreading.testCreateThread	avgt	4	13,373	± 9,837	ms/op
BenchmarkThreading.testCreateVirtualThread	avgt	4	2,385	± 10,702	ms/op

Рис. 1. Создание потоков

Benchmark	Mode	Cnt	Score	Error	Units
BenchmarkThreading.testCachedExecutorSmallTask	avgt	4	1,231	± 0,533	ms/op
BenchmarkThreading.testFixedExecutorSmallTask	avgt	4	1,703	± 0,405	ms/op
BenchmarkThreading.testVirtualExecutorSmallTask	avgt	4	0,050	± 0,003	ms/op

Рис. 2. Тестирование производительности на малых задачах

Benchmark	Mode	Cnt	Score	Error	Units
BenchmarkThreading.testCachedExecutorNormalTask	avgt	4	5,125	± 0,652	ms/op
BenchmarkThreading.testFixedExecutorNormalTask	avgt	4	5,175	± 0,624	ms/op
BenchmarkThreading.testVirtualExecutorNormalTask	avgt	4	4,231	± 2,409	ms/op

Рис. 3. Тестирование производительности на средних задачах

Benchmark	Mode	Cnt	Score	Error	Units
BenchmarkThreading.testCachedExecutorNormalTask	avgt	4	4015,944	± 1223,935	ms/op
BenchmarkThreading.testFixedExecutorNormalTask	avgt	4	4063,273	± 1942,217	ms/op
BenchmarkThreading.testVirtualExecutorNormalTask	avgt	4	4073,638	± 1424,138	ms/op

Рис. 4. Тестирование производительности на тяжелых задачах

Однако здесь стоит сделать оговорку, что загрузка процессора относится к типу задач, связанных с операционной системой, а виртуальные потоки не так хороши в этой области, т.к. управляются виртуальной машиной в отличие от стандартных.

Также стоит понимать, что бесконечное создание потоков не ведет к значительной оптимизации и с каждым новым потоком эффективность

уменьшается, т.к. количество ресурсов не является бесконечным. Переключение между потоками добавляет дополнительную нагрузку и в конце концов прирост производительности пойдет на убыль.

Далее было проведено нагрузочное тестирование с использованием такого инструмента как JMeter. Тест представлял собой моделирование процесса одновременного обращения к приложению 1000 пользователей в течении 100 миллисекунд. Результаты показали, что при использовании стандартных потоков, происходил скачок времени ответа от 1 миллисекунды до 5, это было вызвано тем, что в какой-то заканчиваются свободные потоки и требуется подождать пока часть из них освободится. Виртуальные же потоки не имеют значительных ограничений на количество, поэтому время ответа так и осталось в пределах 1 миллисекунды. На рис. 5-6 показаны эти результаты.

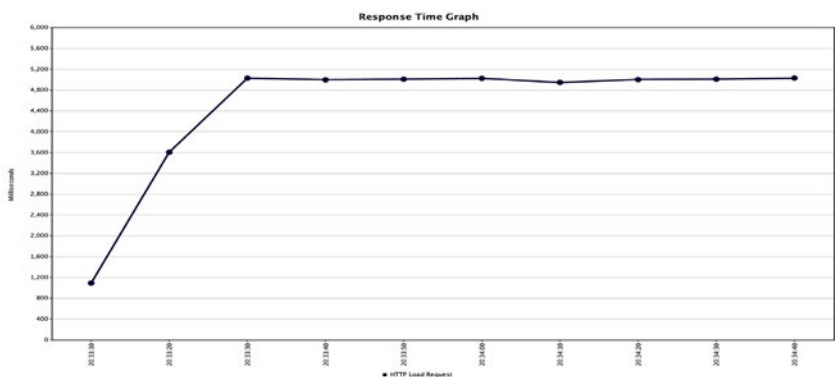


Рис. 5. Нагрузочное тестирование со стандартными потоками

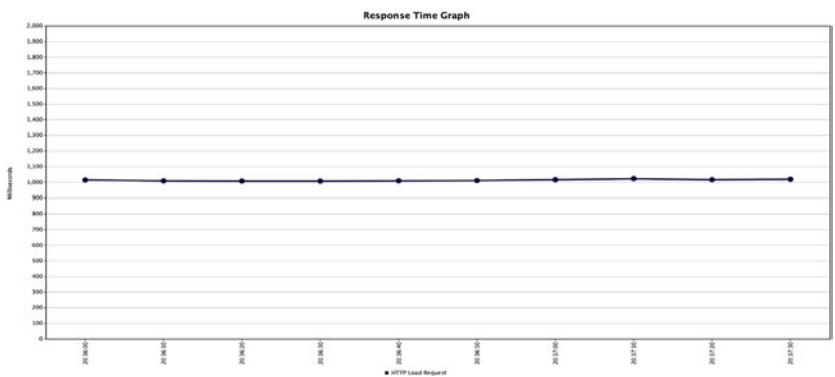


Рис. 6. Нагрузочное тестирование с виртуальными потоками

7. Выводы

1. Виртуальные потоки станут новым инструментом для работы с асинхронностью за счет неблокирующей модели, при этом предоставляя несложный подход для написания такого кода. Приложения, которые проводят много асинхронных операций ввода-вывода, такие как чтение, запись больших файлов, сетевой ввод-вывод, могут использовать виртуальные потоки для каждой операции, тем самым увеличивая производительность и эффективность потоков.

2. Эта технология применима в задачах, где количество потоков или затраты на их создание могут быть решающим фактором. Веб-серверы или приложения, управляющие множеством одновременных сетевых соединений, могут использовать виртуальные потоки для обработки каждого соединения. Это позволяет легко масштабироваться до тысяч или даже миллионов параллельных соединений, так как виртуальные потоки мало загружают систему.

3. Виртуальные потоки неэффективны при серьезных операциях, связанных с загрузкой процессора и предпочтительными для таких случаев остаются стандартные. Если задача требует интенсивной работы процессора, как тяжелые вычисления или обработка большого объема данных в памяти, то использование большого количества виртуальных потоков приведет к частым сменам контекста, что может снизить общую производительность системы.

Заключение

В данной статье было рассмотрено виртуальные потоки в Java, их возникновение с технической точки зрения, теоретическая основа, особенности их реализации. Были проведены различные виды тестирования с использованием сторонних библиотек. Сформулированы выводы относительно полезности этой технологии.

Список литературы

1. Java Concurrency in Practice : книга переведенная / Б. Гетц [и др.]. – СПб.: Питер, 2020. – 464 с.
2. Представление Project Loom в Java [Электронный ресурс] : Статья. – Режим доступа: <https://habr.com/ru/articles/698440/>
3. Различие между потоками и виртуальными потоками в Java [Электронный ресурс] : Статья. – Режим доступа: <https://www.baeldung.com/java-virtual-thread-vs-thread>
4. Введение в пулы потоков Java [Электронный ресурс] : Статья. – Режим доступа: <https://www.baeldung.com/thread-pool-java-and-guava>

Анализ модели графа свойств на примере языка Cypher

Д.В. Мамонов

Студент бакалавр

Н.К. Самойлов

Старший преподаватель

Введение

На сегодняшний день наиболее распространенными системами управления базами данных (СУБД) являются программные продукты, основанные на реляционной модели отображения данных, например, PostgreSQL, MySQL, Oracle. Несмотря на то, что они отлично справляются со структурированными данными и простыми отношениями, при работе с сильно взаимосвязанными данными или быстро меняющимися схемами они испытывают трудности. Поэтому во многих отраслях становятся популярными графовые базы данных, являющиеся NoSQL СУБД. Такие системы позволяют эффективно представлять и запрашивать сложные взаимосвязи между объектами данных. Узлы, ребра и свойства являются основными компонентами графовой базы данных, обеспечивая интуитивно понятную и наглядную модель данных. Но для таких СУБД язык запросов SQL, обычно используемый реляционными базами данных, неприменим в большинстве случаев. Поэтому разработчики придумали собственные варианты языков, один из которых – Cypher – и будет рассматриваться в данной работе.

1. История языка Cypher и попытки стандартизации

Cypher – специальный декларативный язык запросов, который был разработан инженером Андреасом Тейлором во время его работы в компании Neo4j, Inc в 2011 году. На этапе создания язык предполагался для взаимодействия только с графовой базой данных Neo4j, но в 2015 году появился проект openCypher, который позволил интегрировать его и с другими СУБД [1].

После создания проекта openCypher сообществом были предприняты попытки стандартизации языка в качестве базового языка обработки графовых данных, совместимого с языком запросов к реляционным СУБД SQL. Для этого были проведены 5 очных

совещаний разработчиков openCypher. Первое совещание состоялось в феврале 2017 года в штаб-квартире SAP в Вальдорфе (Германия) и совпало с заседанием Совета по тестированию связанных данных. Последнее OCIM состоялось в Берлине одновременно с семинаром W3C по веб-стандартам управления графическими данными в марте 2019 года. По итогу работы в сентябре 2019 года был создан проект по разработке GQL (Graph Query Language) (рис. 1), который был одобрен голосованием национальных органов по стандартизации, являющихся членами Объединенного технического комитета ISO/IEC. На текущий момент стандарт находится в разработке и еще не является общедоступным.

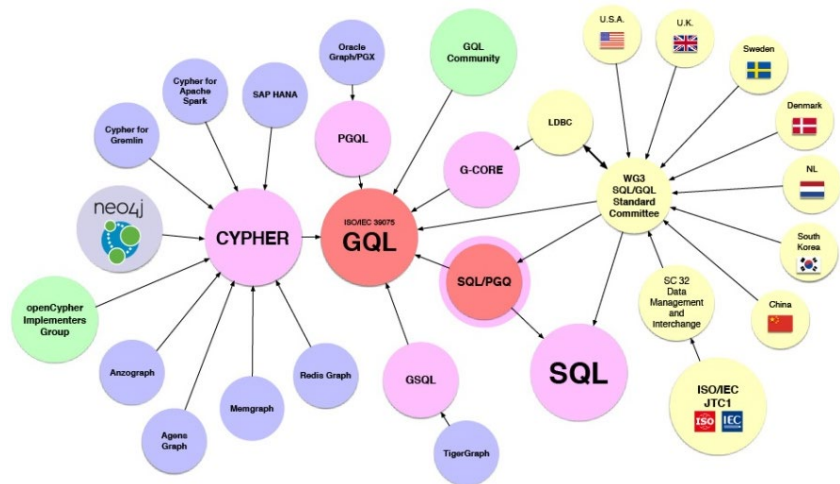


Рис. 1. Компоненты стандарта GQL

2. Модель графа свойств

Язык Cypher основан на использовании модели графа свойств (рис. 2). В теории графов такая структура данных может быть определена как ориентированный, с помеченными узлами и ребрами мультиграф, где ребра могут быть уникально идентифицированы. В графе свойств понятие узла отражает вершину, а понятие отношения – ребра [2].

Графовая модель включает в себя следующие понятия: сущность; путь в графе; токен; свойство.

Сущность – уникальный объект, который можно сравнить на идентичность с другими аналогичными объектами. Она содержит в себе

специальный контейнер свойств – словарь "ключ – значение". Каждое свойство (т.е. "ключ") является уникальным в пределах описания одной сущности. Сущность представлена узлом и отношением между узлами.

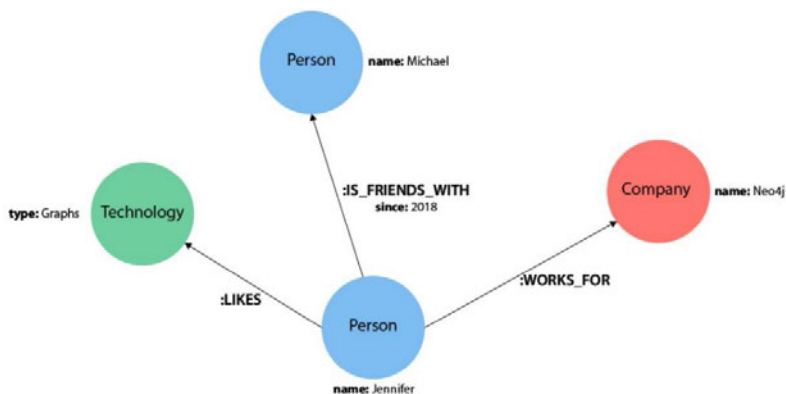


Рис. 2. Пример графа свойств

Узел – базовая сущность графа с уникальными атрибутами, которая может существовать независимо от других сущностей. Он может хранить в себе набор свойств, который не является обязательным. Узел может иметь входящие и исходящие отношения (либо вообще не иметь отношений, связывающих его с остальными узлами).

Отношение – сущность, обозначающая направленное соединение между двумя узлами (исходный и целевой узлы). Оно может быть исходящим и входящим. Исходящее отношение рассматривается со стороны исходного узла, входящее – со стороны целевого узла. Отношение может иметь только один тип (или метку).

Путь в графе отображает обход в графе свойств и состоит из набора узлов, соединяемых отношениями между собой. Он всегда начинается с определенного узла и заканчивается узлом. Самым кратчайшим путем считается один узел, являющийся и начальным, и конечным. Длина пути – характеристика, обозначающая количество отношений в нем.

Токен – непустая строка символов в кодировке Unicode. Токенами представлены метки, типы отношений и ключи свойств. Метка присваивается только узлам, тип отношений – отношениям. Ключи свойств – названия уникальных идентификаторов полей в наборе свойств.

Свойство – пара «ключ-значение», которая может быть инициализирована значением определенного типа или списком таких значений.

3. Описание DML в Cypher

Cypher использует ASCII-представление для описания запросов, т.е. информация должна быть наглядно визуализирована для пользователя. Это делает язык очень наглядным и удобным для чтения, поскольку он как визуально, так и структурно представляет данные, указанные в запросе. Запросы Cypher объединяются с шаблонами узлов и связей с любой указанной фильтрацией по меткам и свойствам для создания, чтения, обновления, удаления данных, найденных в указанном шаблоне [3].

Самое простое описание узла – его имя, заключенное в круглых скобках. Дополнительно при описании узла могут указываться его метка (например, при выборе определенных узлов) и тело свойств (например, при создании нового отношения), как представлено на листинге 1 и рис. 3.

Листинг 1

Описание узлов

```
(s) // 1. Простейшее описание
(s:Person) // 2. Описание с меткой
(s:Person{name: 'Tom', age: 18}) // 3. Описание со свойствами
```

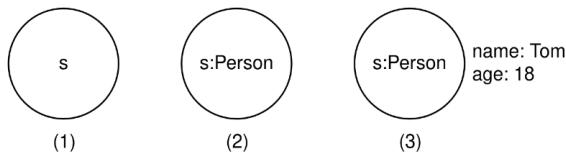


Рис. 3. Визуализация узлов графа

В случае если набор свойств отображен в выборке, то он накладывает дополнительное ограничение на ее результат. При выполнении запроса CREATE набор будет установлен по умолчанию в новых узлах и отношениях.

Для описания простейшего отношения между двумя узлами используется конструкция вида (a)-[]-(b). В этом случае такие отношения будут образовывать неориентированный граф. Для того чтобы указать направление отношения, используются знаки ->, представляющие собой «стрелочку». Как и узлы, отношения могут иметь имена и метки, обозначающие их тип (или класс, к которому они относятся). Описать отношение, которое имеет определенный тип, можно, указав в квадратных скобках идентификатор отношения и его тип через знак двоеточия. В отличие от узлов, при задании нового отношения или при построении шаблона имя отношения можно

опустить и использовать только название метки. Примеры задания отношений приведены на листинге 2, визуализация представлена на рис. 4.

Листинг 2

Описание отношений между узлами

```
// 1. Простейшее неориентированное отношение
(a) - [] - (b)
// 2. Простейшее ориентированное отношение
(a) - [] -> (b)
// 3. Отношение с указанием его типа
(a) - [r:REL_TYPE] -> (b)
// 4. Отношение с указанием его типа и с опущением
идентификатора
(a) - [:REL_TYPE] -> (b)
```

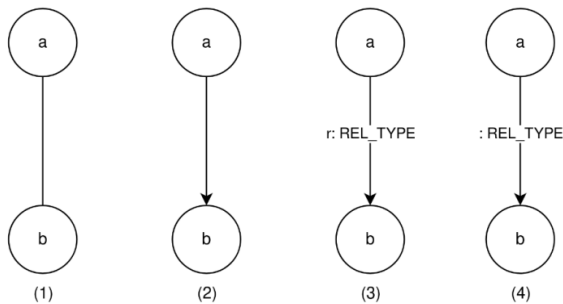


Рис. 4. Визуализация отношений в графе

4. Запросы выборки данных в Cypher

Подобно другим языкам запросов, Cypher содержит множество ключевых слов для задания шаблонов, фильтрации шаблонов и возврата результатов. Наиболее распространенными для построения запросов являются MATCH и RETURN (примеры приведены в листинге 3 и визуализированы на рис. 5).

Ключевое слово MATCH обозначает поиск существующих узлов, отношений, меток, свойств или шаблонов в СУБД. Оно является аналогичным слову SELECT в SQL. В запросах, построенных с MATCH, можно производить поиск всех меток и типов отношений, узлов и отношений на основе шаблонов [4].

Ключевое слово RETURN в Cypher указывает, какие значения или результаты можно вернуть из запроса. В качестве объектов, которые можно возвращать, могут использоваться узлы, отношения, свойства узлов и отношений, шаблоны. RETURN не требуется при выполнении запросов записи, но необходим для выборки. Названия переменных узла

и отношения становятся необходимыми при использовании RETURN. Чтобы вернуть узлы, отношения, свойства или шаблоны, в запросе MATCH должны быть указаны переменные для данных, которые надо вернуть.

Листинг 3

Примеры запросов на шаблонах с MATCH и RETURN

```
MATCH (p1:Person)-[:PARENT]->(p2:Person)
RETURN p1, p2
MATCH ans = (:Person {name: "Pam"})-[:PARENT]->(:Person {name:
"Bob"})
RETURN ans
```

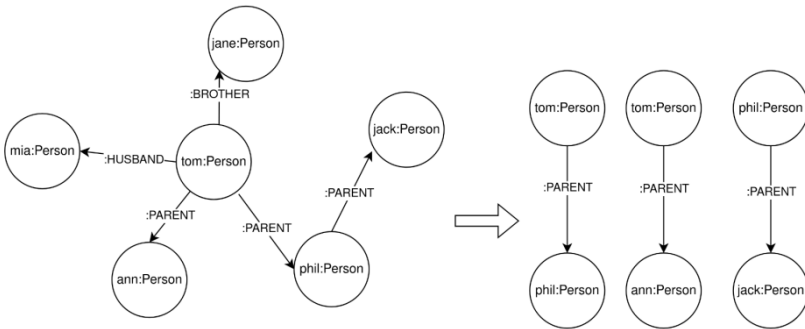


Рис. 5. Визуализация выполнения запроса выборки

Язык поддерживает фильтрацию выборки с помощью ключевого слова WHERE, которое имеет аналогичный принцип работы как в языке SQL (примеры приведены в листинге 4 и визуализированы на рис. 6).

Листинг 4

Примеры запросов с фильтрацией через WHERE

```
MATCH (p1:Person)-[:PARENT]->(p2:Person)
WHERE p1.name="Bob"
RETURN p2
MATCH (p:Person)
WHERE p.name STARTS WITH 'M'
RETURN p.name
```

5. Запросы изменения базы данных в Cypher

Редактирование базы данных (создание новых узлов и отношений, их изменение и удаление) обеспечивается при помощи ключевых слов CREATE, SET и DELETE [5].

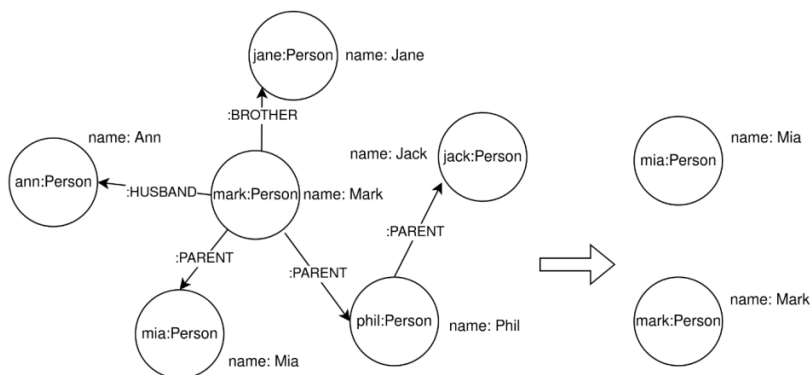


Рис. 6. Визуализации фильтрации выборки по свойству

Запрос CREATE является отдельно выполняемым запросом (как и INSERT в SQL), но при создании отношений может являться частью MATCH. Он позволяет создавать как отдельные узлы и отношения из уже имеющихся узлов, так и целый граф, в котором будут содержаться новые данные (примеры приведены в листинге 5 и визуализированы на рис. 7).

Листинг 5

Запросы на создание сущностей через CREATE

```
CREATE (n:Person)
CREATE (n:Person {name: 'Andres', title: 'Developer'})
MATCH (a:Person), (b:Person)
  WHERE a.name = 'Node A' AND b.name = 'Node B'
    CREATE (a)-[r:RELTYPE]->(b)
RETURN r
CREATE pam: Person{name: "Pam"},
  bob: Person{name: "Bob"},
  (pam: Person)-[:PARENT]->(bob: Person)
```

Запросы SET и DELETE выполняются только в теле запроса MATCH и могут проводиться как над всеми выбранными данными, так и над результатом их фильтрации. Узлы разрешается удалять только после того, как они не будут включены ни в одно отношение (примеры приведены в листинге 6).

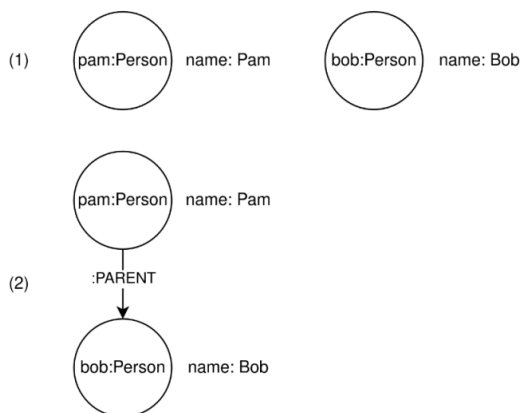


Рис. 7. Визуализация процесса создания сущностей в базе данных

Листинг 6

Примеры запросов на изменение данных в графе

```
MATCH (n {name: 'Andres'})
  SET n.surname = 'Taylor'
  RETURN n
MATCH (n:Useless)
  DELETE n
MATCH (p1:Person) - [r] -> (p2:Person)
  DELETE r
```

Заключение

В данной статье было представлено описание модели графа свойств и относящихся к ней понятий, таких как сущность, узел, отношение, токен, путь и свойство. Также была рассмотрена история разработки языка запросов Cypher и варианта стандартизации подобных языков для обращения к графовым базам данных GQL. Показаны базовые варианты определения в языке узлов, отношений, их свойств, меток и типов, а также шаблонов, связывающих компоненты в единое целое. В работе были приведены примеры полноценных запросов на выборку, создание и изменение данных.

Список литературы

1. Cypher: An Evolving Query Language for Property Graphs / N. Francis [et al.] // SIGMOD'18 Proceedings of the 2018 International Conference on Management of Data. – Houston, United States, June, 2018. – P. 2-3

2. Property Graph Model [Электронный ресурс]. – Режим доступа: <https://github.com/opencypher/openCypher/blob/master/docs/property-graph-model.adoc>
3. Cypher Query Language Reference Version 9 [Электронный ресурс]. – Режим доступа: <https://s3.amazonaws.com/artifacts.opencypher.org/openCypher9.pdf>
4. Anthapu, R. Graph Data Processing with Cypher / R. Anthapu – Birmingham: Packt Publishing, 2022. – 332 p. – ISBN 978-1804611074
5. Panzarino, O. Learning Cypher / O. Panzarino – Birmingham: Packt Publishing, 2014. – 254 p. – ISBN 978-380561089

Разделение вокала и музыкального сопровождения в песнях с помощью нейронных сетей

Н. В. Мерзляков

Студент бакалавр

А. В. Акимов

Старший преподаватель

Введение

В современном мире пристрастие к музыке у людей стало частью жизни. Каждый человек находит в прослушивании музыки что-то свое, уникальное. Кто-то таким образом самовыражается, а кто-то обретает источник вдохновения. И у каждого из людей музыка переплетается с различными аспектами повседневной жизни.

В современной музыкальной индустрии существует постоянная потребность в инструментах и методах для разделения аудиозаписей на составляющие части, такие как вокал и музыкальное сопровождение. Это актуально для многих областей, включая звукозапись, музыкальное производство, а также анализ и обработку аудиоданных. Нейронные сети представляют собой мощный инструмент в области обработки сигналов и анализа аудио, что делает их перспективным средством для решения проблемы разделения вокала и музыкального сопровождения.

В данной работе внимание будет уделено изучению и реализации методов разделения вокала от музыкального сопровождения в песнях, используя нейронные сети. Эта тема имеет огромное значение для аранжировщиков, звукорежиссеров и исследователей, так как обеспечивает новые возможности в творческом процессе и анализе музыкальных композиций.

1. Описание рабочей среды

Используемым основным интегрированным средством разработки (IDE) является PyCharm. PyCharm предоставляет широкий набор инструментов для эффективной разработки на языке Python. Благодаря поддержке различных фреймворков и библиотек, PyCharm позволяет эффективно работать над различными проектами, обеспечивая высокую производительность и комфортное программирование.

Для интерактивной разработки и документирования исследований использован Jupyter Notebook. Это мощное средство, которое позволяет создавать и выполнять код в виде отдельных блоков, интегрировать текстовые описания, визуализации и результаты выполнения кода. Jupyter используется для проведения экспериментов, анализа данных и создания отчетов, что делает работу более наглядной и понятной.

Для решения задач машинного обучения и анализа аудиоданных используются библиотеки Keras и Librosa. Keras предоставляет удобный интерфейс для создания и обучения нейронных сетей, позволяя быстро прототипировать и разворачивать модели глубокого обучения. С помощью Keras создается модель для разделения голоса и аккомпанемента.

Librosa, с другой стороны, является неотъемлемым инструментом для анализа аудиоданных. Librosa используется для извлечения признаков из звуковых файлов, спектрального анализа и обработки аудиосигналов.

2. Набор данных

Для обучения модели нейронной сети использован набор данных DSD100 (The Demixing Secrets Dataset). Датасет DSD100 представляет собой ресурс для исследований в области разделения вокала и музыкального сопровождения в аудиозаписях с применением методов глубокого обучения. Состоящий из 100 композиций различных музыкальных жанров, таких как рок, поп, джаз и др., данный датасет предоставляет разделенные аудиодорожки для вокала и музыкального сопровождения для каждой композиции. Это делает DSD100 ценным инструментом для обучения и оценки алгоритмов, разработанных для автоматического разделения аудиосигналов на составляющие части [1].



Рис. 1. Составляющие набора данных DSD100

3. Предварительная обработка данных

Первоначально производится загрузка датасета DSD100. В процессе предварительной обработки данных используется библиотека Librosa для аудиоанализа и преобразования аудиозаписей в спектрограммы. Спектрограмма, представляет собой визуальное изображение, которое демонстрирует распределение энергии звукового сигнала по частотам и во времени, таким образом можно «увидеть» звук и изучить его характеристики. Функция Short-Time Fourier Transform (STFT) из библиотеки Librosa позволяет разбить аудиосигнал на короткие участки и преобразовать каждый участок в его частотно-временное представление [1].

$$X(m, \omega) = \sum_{n=-\infty}^{\infty} x[n] \omega[n-m] e^{-i\omega n}, \quad (1)$$

где $x[n]$ – дискретный сигнал во временной области, $\omega[n]$ – оконная функция, m – дискретная величина (временной индекс), ω – непрерывная величина (частота) [2].

После получения стандартных спектрограмм, для дальнейшей оптимизации и работы с данными, выделяются компоненты амплитуды спектрограмм без учета фазы. Это позволяет снизить размерность данных и убрать ненужную информацию о фазе, сохраняя при этом основные характеристики спектрограммы.

Для обеспечения единообразия данных и упрощения последующей обработки, каждая спектрограмма приводится к стандартному формату с размерностью (N, 1025, 100), где N – количество разделений всего участка аудиозаписи, 1025 – количество частотных полос в каждом клипе (вдвое меньший размер, чем длина окна STFT), 100 – количество временных кадров в каждом клипе. Такой формат обеспечивает удобство работы с данными и готовность к дальнейшему этапу обучения нейронной сети.

Примеры спектрограмм смеси и вокала представлены на рис. 2.

4. Описание U-Net сверточной нейронной сети

Сверточные нейронные сети (CNN) представляют собой мощный инструмент в области глубокого обучения, который нашел широкое применение в различных областях, включая компьютерное зрение, распознавание речи, медицинская маркировка и другие. Они эффективно решают задачи классификации, детекции объектов, сегментации изображений и другие задачи, связанные с анализом визуальных данных.

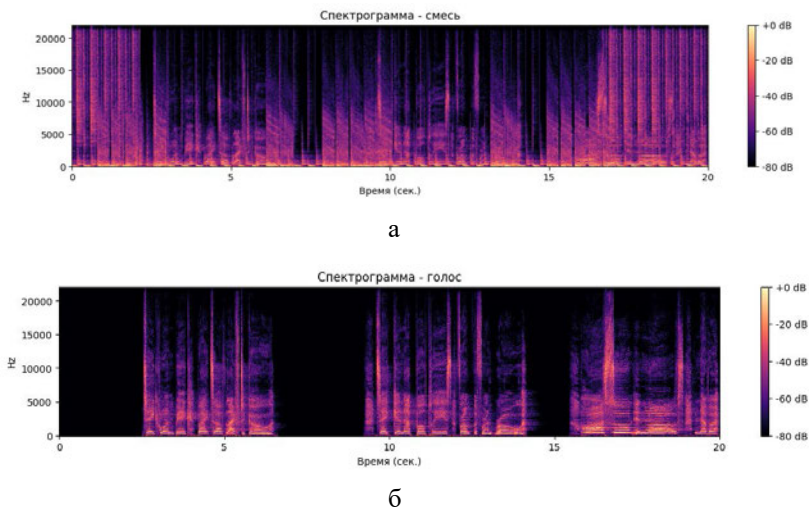


Рис. 2. Спектрограммы фрагмента аудиозаписи: а – спектрограмма смеси, б – спектрограмма голоса

Основная идея CNN заключается в том, что они способны автоматически извлекать признаки из входных данных без необходимости явного программирования этих признаков. Это достигается за счет применения операций свертки и пулинга, которые позволяют сети выделять важные пространственные и временные особенности в данных.

UNet – это глубокая нейронная сеть, которая была разработана для сегментации изображений, то есть для разделения изображения на отдельные сегменты или объекты. Название "UNet" происходит от его архитектуры, которая напоминает букву "U" из-за особенной структуры энкодера и декодера, так как они расположены на противоположных концах архитектуры.

Основной идеей UNet является комбинация пространственно-связанных признаков с более глобальной контекстуальной информацией, что позволяет ей точно и эффективно выделять объекты на изображениях. Это достигается за счет использования сверточных слоев в энкодере для извлечения признаков с различных уровней абстракции, а затем объединения этих признаков в декодере для создания детализированной сегментации.

Архитектура UNet состоит из двух основных частей: энкодера и декодера. Энкодер состоит из серии сверточных слоев, которые

постепенно уменьшают размер изображения и извлекают его признаки. Декодер, напротив, состоит из серии обратных сверточных слоев (иногда называемых де-конволюционными слоями), которые постепенно увеличивают размер изображения и объединяют признаки с разных уровней энкодера для получения итоговой сегментации.

В контексте разделения вокала и музыкального сопровождения в песнях с использованием нейронных сетей, архитектура UNet может быть применена для выполнения этой задачи. Обычно в музыкальном сигнале вокал и музыкальное сопровождение представлены различными частями спектра и имеют разные характеристики. Например, вокальные части обычно содержат высокие частоты и характеризуются переменной частотой, в то время как музыкальное сопровождение может иметь более широкий диапазон частот и более стабильные характеристики.

UNet может быть обучен для разделения этих двух компонентов путем обучения на большом наборе данных, содержащем песни с размеченными вокальными и музыкальными частями. В ходе обучения сеть будет изучать характеристики спектра, которые характеризуют каждую из этих компонент, и использовать их для эффективного разделения вокала и музыкального сопровождения в новых аудиозаписях [3].

Реализованная архитектура сети представлена на рис. 3.

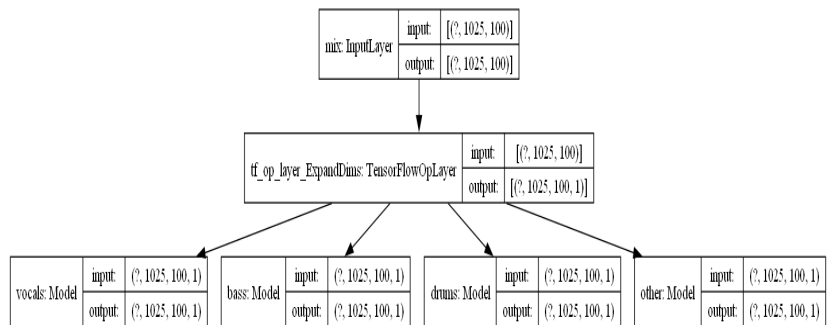


Рис. 3. Архитектура созданной нейронной сети.

На вход нейронной сети подаются обработанные спектрограммы с размерностью (N, 1025, 100). Далее добавляется дополнительная размерность с помощью блока TensorFlowOpLayer, чтобы размерность соответствовала ожидаемой размерности тензора, для последующей передачи в U-Net модель.

Каналы voice, bass, drums и other имеют свою модель нейронной сети, в них используется U-Net архитектура, представленная на рис. 4.

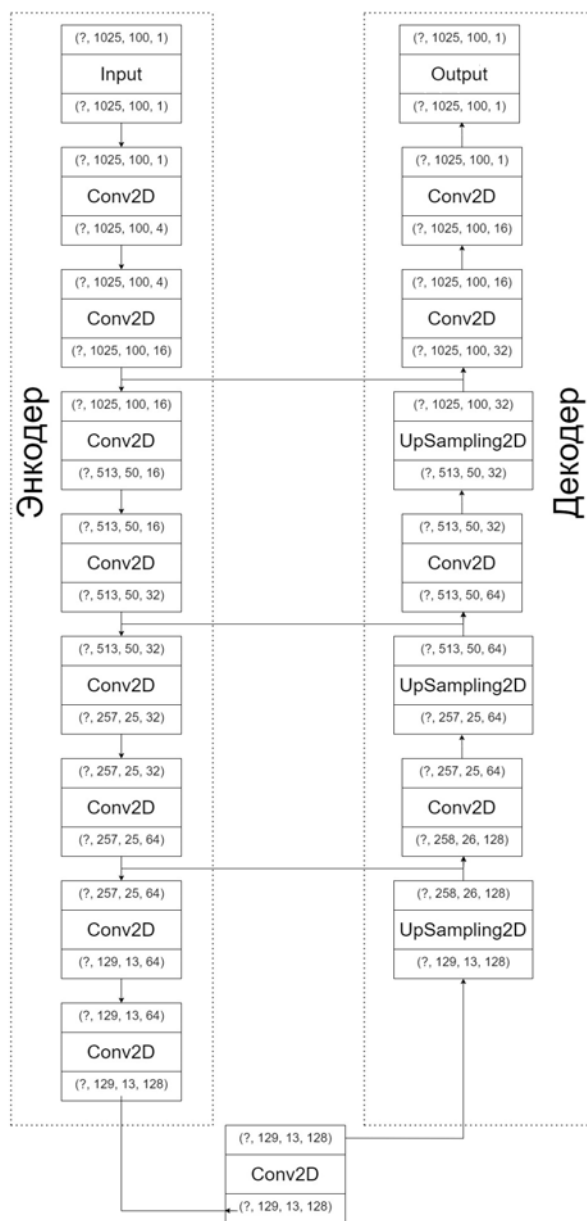


Рис. 4. Архитектура U-Net нейронной сети

InputLayer слой служит точкой входа для нейронной сети, принимая входные данные (спектрограммы) для дальнейшей обработки.

Conv2D сверточные слои являются основными строительными блоками сверточных нейронных сетей (CNN), и присутствуют в энкодере и декодере. Они выполняют свертки, которые включают в себя скольжение фильтра (ядра) по входным данным для извлечения признаков. После каждого из сверточных слоев применяется функция активации и нормализация по батчам.

Функции активации Rectified Linear Unit (ReLU) вводят нелинейность в сеть. Они выводят входное значение, если оно положительное, и ноль в противном случае. Это помогает сети изучать более сложные отношения между входными данными и желаемым выходом.

Нормализация по батчам помогает стабилизировать процесс обучения путем нормализации активаций каждого слоя по набору данных. Это помогает предотвратить внутренний ковариационный сдвиг, делая процесс обучения более эффективным и менее подверженным взрывающимся или исчезающим градиентам.

Слои повышения дискретизации (UpSampling2D) используются для увеличения пространственного разрешения карт признаков. Они выполняют интерполяцию ближайшего соседа, чтобы вставить новые значения между существующими. В данной архитектуре U-Net, карты признаков из энкодера увеличиваются в масштабе и объединяются с соответствующими картами признаков декодера для восстановления пространственных деталей.

TensorFlowOpLayer слой позволяет включать пользовательские операции TensorFlow в нейронную сеть. В данной архитектуре слой необходим чтобы размерность подаваемого на новый слой тензора соответствовала ожидаемой.

На рис. 4 можно заметить, что архитектура похожа на букву "U". В энкодере происходит уменьшение размеров изображения и извлечение его признаков, а в декодере – увеличение размеров изображения и объединение признаков с разных уровней энкодера. Таким образом происходит выделение нужного канала (voice, bass, drums или other) из спектрограммы.

5. Оценка точности разделения

Измерение результатов подхода к разделению источников является сложной задачей. В общем, существуют две основные категории оценки выходных данных подхода к разделению источников: объективная и субъективная. Объективные меры оценивают качество разделения, выполняя набор вычислений, которые сравнивают выходные сигналы

системы разделения с изолированными источниками в истинной форме. Субъективные меры включают оценку от человеческих экспертов, которые выставляют оценки для выходных данных системы разделения источников.

Для оценки работы нейронной сети были использованы объективные оценки: Source-to-Distortion Ratio (SDR), Source-to-Interference Ratio (SIR) и Source-to-Artifact Ratio (SAR). Все три метрики – SDR, SIR и SAR – измеряются в децибелах (дБ). Это позволяет компактно представить, чувствительно к изменениям и сравнимо с человеческим восприятием информацию о качестве разделения источников. Это делает децибелы универсальной и информативной единицей измерения для оценки эффективности систем разделения источников. На сегодняшний день эти метрики являются наиболее распространенными методами оценки качества выходного сигнала системы разделения источников [4].

Отдельные показатели эффективности рассчитываются для вектора каждого оцененного источника $\hat{s}^i$ путем сравнения его с заданным истинным вектором-источником s^i .

Вычисление критериев включает два последовательных шага. На первом этапе $\hat{s}^i$ разлагается следующим образом:

$$\hat{s}^i = s_{target} + e_{interf} + e_{noise} + e_{artif}, \quad (2)$$

где $s_{target} = f(s^i)$ – это версия s^i , модифицированная искажением $f \in F$, а e_{interf} , e_{noise} и e_{artif} – это соответственно компоненты-вектора ошибки, связанные с помехами, шумом и артефактами.

Для получения компонент ошибки используется алгоритм Independent Time Frequency Masking Transform. Сначала оценка каждого источника ($\hat{s}^i$) и его истинный сигнал (s^i) преобразуются в частотно-временное представление с помощью преобразования Фурье. Затем происходит маскирование, вычисляется маска для каждого частотно-временного элемента, она определяет, насколько данный элемент соответствует истинному сигналу. И для получения компонент, связанных с истинным сигналом, помехами, шумом и артефактами применяется обратное преобразование Фурье к замаскированным частотно-временным представлениям.

Ошибка, связанная с помехами (e_{interf}), представляет собой ту часть оценки сигнала, которая не является истинным сигналом и возникла из-за других источников. Это могут быть другие голоса в разговоре, фоновая музыка, шум улицы и т. д.

Ошибка, связанная с шумом (e_{noise}), представляет собой часть оценки сигнала, которая не является истинным сигналом и вызвана посторонним шумом. Это может быть шум от оборудования, электрический шум или фоновый шум.

Ошибка, связанная с артефактами (e_{artif}), представляет собой часть оценки сигнала, которая не является истинным сигналом и возникла в процессе разделения источников. Это могут быть искажения, шумы, возникающие из-за самого алгоритма разделения.

Source-to-Artifact Ratio (SAR) обычно интерпретируется как количество нежелательных артефактов в оценке источника относительно истинного источника. Хорошим считается значение от 10 дБ.

$$SAR = 10 \log_{10} \left(\frac{\|s_{target} + e_{interf} + e_{noise}\|^2}{\|e_{artif}\|^2} \right). \quad (3)$$

Source-to-Interference Ratio (SIR) обычно интерпретируется как количество других источников, которые можно услышать в оценке источника. Хорошим считается значение от 10 дБ.

$$SIR = 10 \log_{10} \left(\frac{\|s_{target}\|^2}{\|e_{artif}\|^2} \right). \quad (4)$$

Source-to-Distortion Ratio (SDR) обычно рассматривается как общая мера того, насколько хорошо звучит источник. Хорошим считается значение от 15 дБ.

$$SDR = 10 \log_{10} \left(\frac{\|s_{target}\|^2}{\|e_{artif} + e_{noise} + e_{artif}\|^2} \right). \quad (5)$$

Необходимо добиться более высоких значений метрик.

6. Обучение нейронной сети

Нейронная сеть обучалась на GPU около 12 часов и среднее используемое количество оперативной памяти было 9 гигабайт. Обучающая и тестовая выборки содержали по 50 музыкальных композиций, приведенных к спектрограммам. Валидационная выборка составляла 10% от общего объема обучающих данных. Это позволяло систематически проверять производительность модели в процессе обучения на независимых данных, не используемых для обучения. Как

оптимизатор был выбран Adam с коэффициентом скорости обучения (learning rate) равным 0.001. Размер пакета (batch size) составлял 16.

После завершения обучения модели были получены следующие метрики:

- SAR = 10.8012861 дБ,
- SIR = 20.6501875 дБ,
- SDR = 10.8012861 дБ.

Заключение

Данная статья посвящена разделению вокала и музыкального сопровождения в песнях с помощью нейронных сетей. Описана рабочая среда, набор данных для обучения нейронной сети, его предварительная обработка, архитектура сети, U-Net часть сети и средства оценки обученной сети. В итоге данного исследования и обучения нейронной сети получен работоспособный инструмент. Данное решение позволяет получать вокал из музыки. Одним из возможных способов использования решения является передача в сервисы распознавания речи, для автоматического получения текстов песен. Это позволит получить лучший результат, так как сервисы будут работать с чистой речью. Также решение можно интегрировать в десктопное приложение или веб-сервис. В дальнейшем предполагается обучать модель на большем наборе данных и использовать дополнительные средства обработки результатов работы нейронной сети.

Список литературы

1. An overview of lead and accompaniment separation in music / Rafii Z. [et al.] // IEEE ACM Transactions on Audio, Speech, and Language Processing. – 2018. – Т. 26. – №. 8. – С. 1307-1335.
2. Белов, С.П. Метод частотно-временного анализа сигналов / С.П. Белов, Е.И. Прохоренко, А.С. Белов ; БелГУ // Научные ведомости БелГУ. Сер. История. Политология. Экономика. Информатика. – 2009. – №1, вып. 9/1. – С. 5-11.
3. Stoller, D. Wave-u-net: A multi-scale neural network for end-to-end audio source separation / D. Stoller, S. Ewert, S. Dixon // arXiv preprint arXiv:1806.03185. – 2018.
4. Vincent, E. Performance measurement in blind audio source separation / E. Vincent, R. Gribonval, C. Févotte // IEEE transactions on audio, speech, and language processing. – 2006. – Т. 14. – №. 4. – С. 1462-1469.

Разработка системы поддержки принятия решения при выборе логистического партнера

Е. Н. Михалевская

Студент бакалавр

Н. А. Алейникова

Доцент

Введение

Современная экономика характеризуется разнообразием логистических услуг и различными потребностями предприятий в их выборе. Возникает необходимость определения наилучшего логистического партнера в условиях множества альтернатив для крупных компаний, а также для предприятий малого бизнеса, которые нацелены на развитие эффективных и гибких цепочек поставок.

Принятие решения по выбору логистического партнера, как правило, происходит с учетом множества критериев, и возникает сложность в принятии обоснованного решения без применения математического моделирования. Кроме того, у компании часто нет четкого представления о том, насколько логистический партнер должен удовлетворять тем или иным критериям, не обязательно, чтобы партнер удовлетворял всем критериям по максимуму, тем более, что на практике этого невозможно достичь. Компанию может устроить логистический партнер, большинство характеристик которого находятся в определенных приемлемых диапазонах. Чем больше характеристик у потенциального логистического партнера превосходят аналогичные характеристики у других претендентов, тем лучше.

Таким образом, целью данной работы является разработка веб-сервиса, реализующего систему поддержки принятия решения, который в дальнейшем может стать существенным инструментом, позволяющим учитывать предпочтения в определении параметров транспортных компаний и облегчить процесс принятия решения при выборе логистического партнера для компаний.

1. Математическое обеспечение веб-сервиса для выбора логистического партнера

Для компании критерии при выборе партнера могут иметь разный вес (важность) при принятии решения.

По этой причине для определения весов критериев в данной работе используются методы экспертного оценивания, в том числе элементы метода анализа иерархий.

Метод анализа иерархий (МАИ) представляет собой систематический подход к решению задач, основанный на декомпозиции проблемы на более простые составляющие части и последующей обработке последовательности суждений лица, принимающего решение (ЛПР) с использованием парных сравнений. Этот метод позволяет выразить относительную степень важности параметров логистической компании в виде иерархии, что упрощает процесс принятия решений на основе множества критериев. В процессе МАИ суждения выражаются численно с применением шкалы Саати.

Процесс решения задачи с использованием метода анализа иерархий включает следующие этапы:

- определение и выделение проблемы: этап предполагает определение относительной степени важности характеристик логистической компании;
- декомпозиция проблемы в иерархию. На рис. 1 – элементы иерархии, – категории и характеристики.
- построение матриц парных сравнений элементов для каждого уровня относительно элементов иерархии более высокого уровня;
- вычисление локальных приоритетов элементов для каждого уровня иерархии относительно элементов более высокого уровня. Для этого используется наибольшее собственное значение матриц суждений и проверка индекса согласованности;
- вычисление глобальных приоритетов.

В результате, определяются локальные и глобальные относительно цели значения весов критериев, по которым принимается решение о выборе логистического партнера.

Для этого вводится понятие вектора глобальных весов критериев низшего уровня иерархии (2-го уровня схемы):

$$\lambda = (\lambda_1, \lambda_2, \dots, \lambda_n). \quad (1)$$

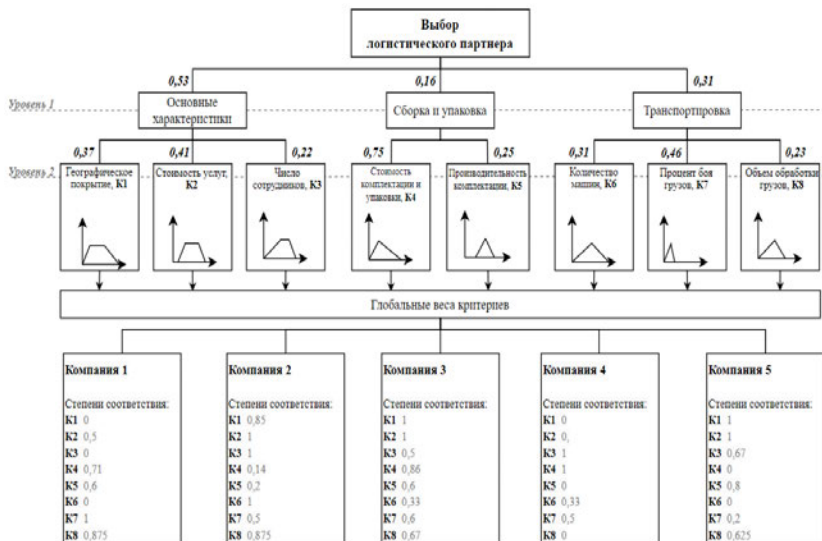


Рис. 1. Элементы иерархии весов критериев

При проведении экспертного сравнения часто возникают несогласованности в оценках. Для определения степени согласованности экспертных суждений используется показатель – индекс согласованности (ИС):

$$I = \frac{\lambda_{\max} - n}{n - 1}, \quad (2)$$

где для определения максимального собственного числа матрицы парных сравнений применяются приближенные методы. Например, максимальное собственное число и соответствующий собственный вектор можно вычислить по формулам:

$$p = \frac{Ae}{e^T Ae}, \quad (3)$$

$$\lambda_{\max} = e^T Ap, \quad (4)$$

где $e = (1, 1, \dots, 1)^T$ – единичный вектор-столбец размерности n ; A – матрица парных сравнений, элементами которой являются оценки отношений, полученные в соответствии со шкалой Саати.

В случае полной согласованности $I = 0$; удовлетворительная степень согласованности определяется неравенством $I \leq 0,1$; при $I \geq 0,1$ экспертные оценки нуждаются в корректировке.

В отличие от классического метода анализа иерархий, где альтернативы так же сравниваются между собой относительно критериев низшего уровня по шкале Саати, в поставленной задаче необходимо учитывать нечеткость требований компании к логистическим партнерам. Поэтому для формализации требований к выбору логистического партнера предлагается критерии низшего уровня задавать с помощью нечетких чисел или интервалов.

Нечеткое число – это нечеткое подмножество универсального множества действительных чисел, имеющее нормальную и выпуклую функцию принадлежности, т. е. такую, что существует значение носителя, в котором функция принадлежности равна единице, а также при отступлении от своего максимума влево или вправо функция принадлежности не возрастает (рис. 2) [1].

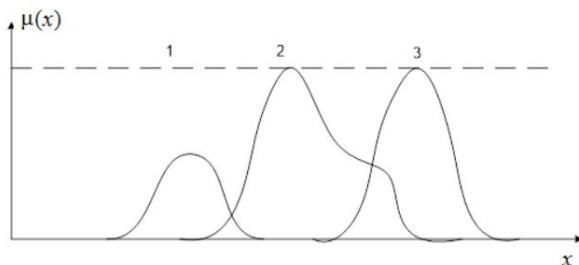


Рис. 2. Нечеткие числа: 1 – нечеткое выпуклое; 2 – нечеткое нормальное; 3 – нечеткое нормальное выпуклое

В рамках поставленной задачи было определено 2 вида нечетких чисел: треугольные и трапециевидные нечеткие числа (рис. 3).

Представлены они следующими функциями принадлежности, которые могут быть как симметричными, так и несимметричными в зависимости от особенностей моделируемой области:

– треугольное нечеткое число:

$$t_{ij} = (t_{ij}^L, t_{ij}^m, t_{ij}^R), \quad (5)$$

где t_{ij}^m – мода (наиболее вероятное значение); t_{ij}^L, t_{ij}^R – левая и правая граница нечеткого треугольного числа.

Выражение для функции принадлежности t_{ij} :

$$\mu_{t_{ij}} = \begin{cases} \frac{t_{ij} - t_{ij}^L}{t_{ij}^m - t_{ij}^L}; t_{ij} \in (t_{ij}^L; t_{ij}^m], \\ \frac{t_{ij} - t_{ij}^R}{t_{ij}^m - t_{ij}^R}; t_{ij} \in [t_{ij}^m; t_{ij}^R), \\ 0, \text{ в остальных случаях,} \end{cases} \quad (6)$$

– трапецевидное нечеткое число:

$$t_{ij} = (t_{ij}^L, t_{ij}^m, t_{ij}^n, t_{ij}^R). \quad (7)$$

Функция принадлежности t_{ij} :

$$\mu_{t_{ij}} = \begin{cases} \frac{t_{ij} - t_{ij}^L}{t_{ij}^m - t_{ij}^L}; t_{ij} \in (t_{ij}^L; t_{ij}^m), \\ 1; t_{ij} \in [t_{ij}^m; t_{ij}^n] \\ \frac{t_{ij} - t_{ij}^R}{t_{ij}^n - t_{ij}^R}; t_{ij} \in (t_{ij}^n; t_{ij}^R), \\ 0, \text{ если } t_{ij} \leq t_{ij}^L, t_{ij} \geq t_{ij}^R, \end{cases} \quad (8)$$

где t_{ij}^m, t_{ij}^n – диапазон наиболее вероятных значений; t_{ij}^L, t_{ij}^R – левая и правая граница нечеткого трапецевидного числа; $\mu_{t_{ij}}(t_{ij})$ – функция принадлежности нечеткого трапецевидного числа.

При рассмотрении трапецевидных чисел предположим, что $m \neq n$, так как в противном случае трапецевидное числовое множество превращается в треугольное.

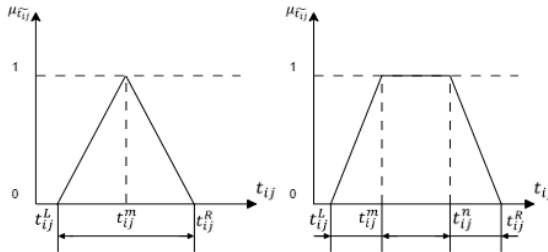


Рис. 3. Симметричная треугольная и трапецевидная функции принадлежности

Подставляя значения характеристик конкретного логистического партнера в соответствующие выражения для функций принадлежности нечетких критериев низшего уровня, будут получены локальные соответствия компании по каждому критерию.

Для свертки множества критериев оценки логистических партнеров в один критерий вводится понятие оператора агрегирования.

Оператором агрегирования называют функцию от n переменных (критериев, частных оценок) $\text{agg}: \bigcup_{j \in n} [0,1]^j \rightarrow [0,1]$,

удовлетворяющую ряду обязательных условий [2,3] на множестве произвольных $x, y \in [0,1]$:

1. $\text{agg}(x) = x$;
2. $\text{agg}(0, \dots, 0) = 0$ и $\text{agg}(1, \dots, 1) = 1$;
3. $\text{agg}(x_1, \dots, x_n) \leq \text{agg}(y_1, \dots, y_n)$, если $(x_1, \dots, x_n) \leq (y_1, \dots, y_n)$.

С помощью агрегирующего оператора можно получить комплексную оценку, которая характеризует объект в целом по всем критериям.

В случаях, когда наиболее важными являются значения частных оценок, используются OWA-операторы, агрегирующие компоненты векторной оценки, упорядоченные определенным образом. Один из вариантов оператора агрегирования OWA позволяет учесть глобальные веса критериев:

$$\text{agg}(g_1, \dots, g_n) = \text{OWA}(g_1, \dots, g_n) = \sum_{j=1}^n w_j g_{\sigma(j)}, \quad (9)$$

где σ – индекс упорядочивания по величине элементов, такой что $g_{\sigma(1)} \geq g_{\sigma(2)} \geq \dots \geq g_{\sigma(n)}$, $w = (w_1, \dots, w_n)^T$ – вектор весов, такой что

$$\sum_{j=1}^n w_j = 1, w_j \geq 0, j = \overline{1, n}.$$

Вектор весов OWA-оператора задается в виде:

$$w_j = \mathcal{Q}\left(\sum_{i=1}^j \lambda_{\sigma(i)}\right) - \mathcal{Q}\left(\sum_{i=1}^{j-1} \lambda_{\sigma(i)}\right). \quad (10)$$

В операторах OWA весовые коэффициенты не привязаны к конкретному элементу вектора оценок альтернативы. Они определяются на основании относительного значения элемента по сравнению с

другими объектами. Таким образом, наибольший элемент получает вес w_1 , следующий за ним – вес w_2 и т.д.

В контексте теории принятия решений, внутри оператора OWA используются нечеткие квантификаторы в ситуациях, когда существует множество критериев. Они позволяют формализовать такие понятия, как принятие решения «по большинству» критериев или «для как можно большего числа».

Квантификатор представляет собой приближенную оценку того количества агрегируемых величин, которые в значительной мере определяют значение обобщенной оценки. Квантификатор – нечеткая переменная, основой которой является доля частных оценок r , распределенная на интервале $[0;1]$.

Рассмотрим квантификатор, с помощью которого можно выразить понятие «для большинства» (рис. 4), функция принадлежности которого имеет вид:

$$Q(r) = \begin{cases} 0, & 0 \leq r \leq a, \\ \frac{r-a}{b-a}, & a < r \leq b, \\ 1, & b < r \leq 1. \end{cases} \quad (11)$$

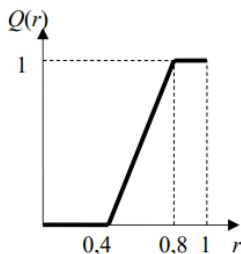


Рис. 4. Квантификатор «Для большинства»

Рассмотрим еще один квантификатор. Квантификатор «чем больше, тем лучше» или «для как можно большего числа» можно определить по формуле:

$$Q(r) = r. \quad (12)$$

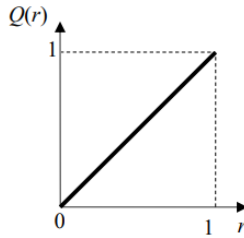


Рис. 5. Квантификатор «Чем больше, тем лучше»

2. Алгоритм осуществления ранжирования на основе оценки покомпонентных локальных соответствий

Рассмотрим поэтапно алгоритм осуществления ранжирования потенциальных логистических партнеров на основе покомпонентных локальных соответствий критериям.

Первый этап. Чтение и обработка файла с загруженными данными.

На данном этапе осуществляется чтение и обработка файла, содержащего список логистических компаний с их характеристиками, на основании которого требуется осуществить ранжирование с целью поиска наиболее подходящего варианта компании-партнера.

Второй этап. Определение относительной степени важности элементов 1-го и 2-го уровней иерархии.

На втором этапе с помощью метода анализа иерархий определяется относительная степень важности каждой характеристики, а также каждого вида категории параметров относительно глобальной цели – выбора лучшего партнера.

Третий и четвертый этапы. Задание пограничных значений выборки НЧ и Определение соответствующих потребностей.

На данных этапах для нескольких характеристик в рамках каждого блока (1-го уровня иерархии) происходит задание пограничных значений выборки каждой характеристики и диапазон наиболее подходящих ее значений/моды в зависимости от вида нечеткого числа, по которому было формализовано требование.

Пятый этап. Формирование векторного соответствия.

Подстановка значений компонент в соответствующие функции принадлежности и формирование векторного соответствия полученных результатов подстановки.

Шестой этап. Упорядочивание векторов.

На шестом этапе осуществляется упорядочивание соответствующих векторов для последующего вычисления весов характеристик с помощью оператора OWA.

Седьмой и восьмой этапы. Выбор квантификатора для OWA и Расчет агрегированного соответствия.

На седьмом этапе происходит выбор квантификатора «для большинства» или «для как можно большего числа» для расчета агрегированного соответствия в рамках каждого блока характеристик.

Девятый этап. Определение наилучшего совпадения.

Последний этап включает расчет соответствий характеристик логистических компаний и поиск наиболее подходящей на основе ранее определенных требований.

3. Реализация веб-сервиса

В соответствии с поставленными задачами реализован веб-сервис принятия решений с возможностью осуществления ранжирования параметров компаний по заданному требованию. С помощью данного приложения пользователь имеет возможность из списка интересующих их организаций получить информацию о наиболее подходящем варианте логистического партнера: сервис позволяет определить наилучшую компанию на основании всех характеристик, а также по отдельным категориям.

Для начала работы сервиса необходимо загрузить файл с датасетом, который содержит исходные данные с характеристиками логистических компаний.

Сервис считывает данные из файла, далее, используя их, находит наилучший вариант логистической компании по заданным требованиям пользователя и выбранному условию – «для большинства» или «для как можно большего числа» характеристик, и возвращает результат работы программы в отдельном всплывающем окне.

На рис. 6 – страница, позволяющая задать диапазоны требований к основным характеристикам логистической компании и выбрать способ, на основании которого будет осуществлена ранжировка компании: квантификатор «для большинства» или «для как можно большего числа».

Аналогичным образом выглядят страницы для осуществления выбора диапазона требований параметров и установки способа ранжирования по другим характеристикам деятельности логистической компании: процессам сборки и упаковки, транспортировки груза (рис. 7, 8).

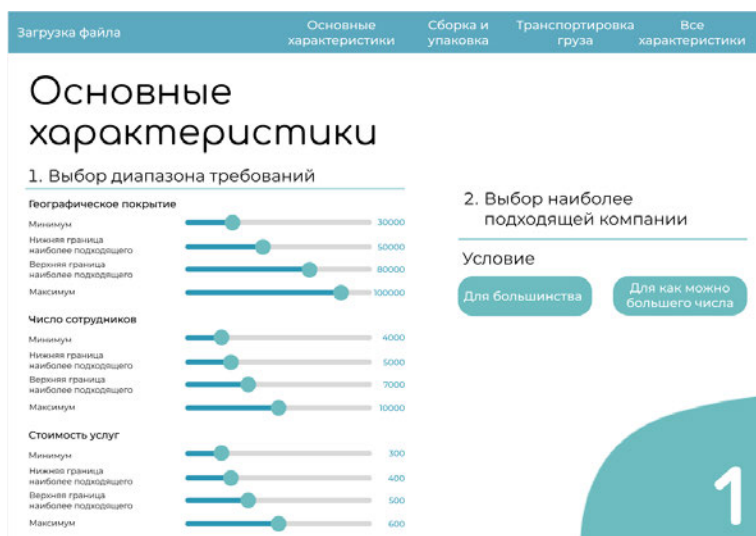


Рис. 6. Страница «Основные характеристики»

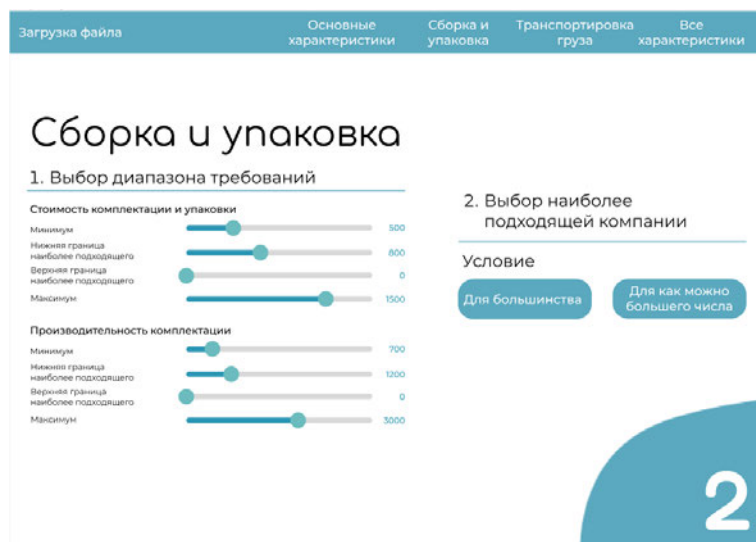


Рис. 7. Страница «Сборка и упаковка»

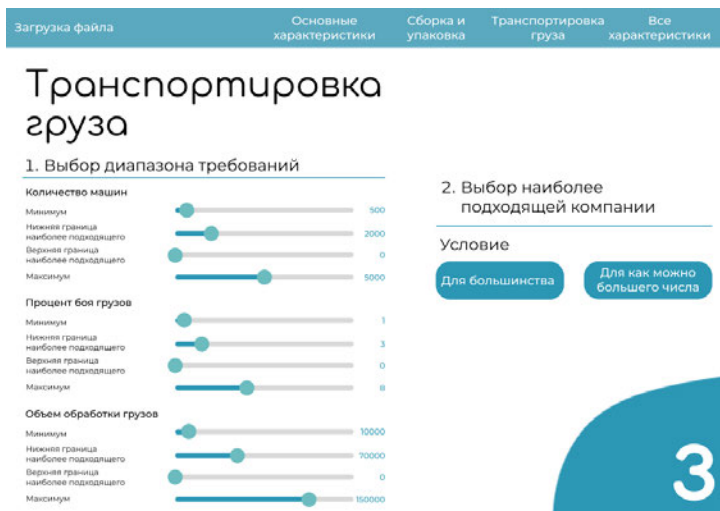


Рис. 8. Страница «Транспортировка груза»

Результат работы сервиса продемонстрирован на рис. 9, 10.

Всплывающее окно информирует пользователя о том, что наилучшим логистическим партнером «для большинства» параметров, относящихся к блоку основных характеристик, является компания «Байкал Сервис» (рис. 9). Но если для тех же данных применить условие «для как можно большего числа», то компания «DPD» в большей степени удовлетворяет заданным требованиям (рис. 10).

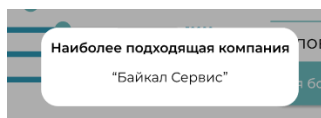


Рис. 9. Уведомление о наиболее подходящей компании с заданным квантификатором «для большинства»

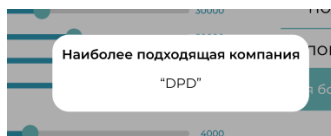


Рис. 10. Уведомление о наиболее подходящей компании с заданным квантификатором «для как можно большего числа»

Заключение

Данная статья посвящена разработке алгоритма ранжирования для принятия решения при выборе логистического партнера с использованием OWA-квантификаторов на основе оценки и агрегирования покомпонентных локальных соответствий. Продемонстрирован результат работы соответствующего веб-сервиса для поддержки принятия решения при выборе логистического партнера.

Сервис выделяется своей легкостью использования и полным отсутствием ограничений: пользователю не требуется проходить авторизацию или вносить сложные настройки, а также он абсолютно бесплатен. Это позволяет пользователям получать необходимую информацию без лишних затрат времени и средств.

Список литературы

1. Чернов, В. Г. Нечеткие множества. Основы теории и применения: учеб. пособие // Владим. гос. ун-т им. А. Г. и Н. Г. Столетовых. – Владимир: Изд-во ВлГУ, 2018. – 36 с.
2. Detyniecki, M. Mathematical aggregation operators and their application to video querying. / M. Detyniecki PhD dissertation. Docteur de l'Universite. // – Paris, – 2000. – 185 p.
3. Mesiar R. and M. Komornikova, 1997. Aggregation Operators. XI Conference on applied Mathematics "PRIM 96", Proceedings. Novi Sad. 193-211.

Разработка программного средства защиты отдельных видов информации конфиденциального характера от несанкционированного доступа путем внесения мешающих искажений во фрагменты графических файлов

К. Р. Нартова

Студент бакалавр

А. В. Швырева

Ассистент

Введение

В соответствии с законодательством Российской Федерации распространение информации конфиденциального характера является незаконным, поэтому при публикации документов в открытом доступе необходимо проводить обезличивание информации. Таким обезличиванием занимаются при публикации результатов судебных практик, например, на сайте судебного департамента или на справочной платформе КонсультантПлюс.

Целью данной работы является разработка программного средства, обеспечивающего защиту информации от несанкционированного доступа путем внесения мешающих искажений во фрагменты графических файлов, представляющих отдельные виды информации конфиденциального характера.

1. Описание предлагаемого алгоритма

Под отдельными видами информации конфиденциального характера в данной работе будем понимать информацию, которая может представлять собой коммерческую тайну, например, персональные данные (фамилия, имя, отчество), названия проектов/работ, наименование населенного пункта, числовые значения и другое. В качестве контейнеров конфиденциальной информации будем рассматривать pdf-файлы, используемые для хранения большей части корпоративных документов.

Идея предлагаемого алгоритма защиты представлена на рисунке 1 и заключается в выделении на документах фрагментов, содержащих интересующие виды информации, и внесении в выделенные фрагменты искажений, мешающих получению доступа к информации.

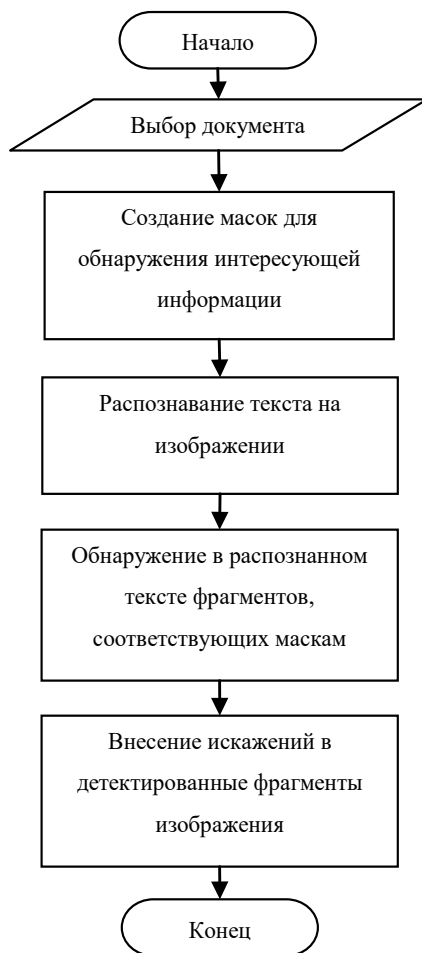


Рис. 1. Идея предлагаемого алгоритма

2. Создание масок для обнаружения интересующей информации

Информацию конфиденциального характера можно разделить на следующие виды: персональные данные, коммерческая тайна, профессиональная тайна и служебная тайна. Для каждого типа можно создать маски, задаваемые регулярными выражениями, для обнаружения соответствующих текстовых фрагментов.

В таблице представлены примеры используемых регулярных выражений и их область применения.

Используемые регулярные выражения

Регулярное выражение	Где используется
<code>[А-ЯЁ][а-яё]+\s+[А-ЯЁ][а-яё]+\s+[А-ЯЁ][а-яё]+(?:\s+[А-ЯЁ][а-яё]+)?</code>	ФИО в полном виде как с отчеством, так и без него
<code>[А-ЯЁ][а-яё]+\s*[А-ЯЁ][.][А-ЯЁ][.]</code>	ФИО инициалы
<code>\d{2}[\.]\d{2}[\.]\d{4}</code>	Дата, включая день, месяц и год
<code>[\w\.-]+@[\w\.-]+(?:\.[\w]+)+</code>	Почта
<code>\d{2}-?\s?\d{2}\s?№\s?\d{6}</code>	Серия и номер паспорта
<code>г[.]{2,24}ул[.]*д[.]*кв[.]\s*\d{1,3}</code>	Адрес, место жительства
<code>8\s?[(?)\d{3}()][?]\s?\d{3}</code> <code>[-]\s?\d{2}[-]\s?\d{2}</code>	Номер телефона
<code>\d{3}([-]\s)\d{2}([-]\s)\d{2}</code>	Номер домашнего телефона

3. Описание метода распознавания текста на изображении

Для выделения текста на изображении используются методы оптического распознавания символов (ОРС)[1]. Данный метод разбивает изображение на отдельные символы и сопоставляет их с набором известных символов, чтобы перевести изображение в текстовый формат, который может быть сохранен, отредактирован и использован. Задачи ОРС могут решаться с помощью технологий компьютерного зрения и машинного обучения [2].

В представленной работе распознавание текста реализуется с помощью инструмента искусственных нейронных сетей Google Tesseract OCR Engine. Выбор обуславливается высокой точностью и скоростью распознавания символов, поддержкой распознавания текста на более чем 100 языках, в том числе смешанного текста на нескольких языках. Стоит отметить, что Tesseract постоянно улучшается и обновляется, повышая качество распознавания [3].

4. Разработка программного средства

Реализацию программного обеспечения можно разбить на несколько этапов: предварительная обработка данных, распознавание текста, анализ текста, удаление информации конфиденциального характера.

Предварительная обработка данных заключается в конвертировании PDF файла в формат изображений, загрузке изображений и преобразовании цветовых пространств BGR в RGB. А также создание набора масок для поиска интересующей информации.

Этап распознавания текста предназначен для получения всех данных с документа.

На этапе анализа реализуется поиск слов или словосочетаний, которые могут нести конфиденциальную информацию в соответствии с существующими масками, вычисляются координаты обнаруженных слов.

Удаление информации конфиденциального характера производится в соответствии с координатами, полученными на предыдущем этапе, а на месте обнаруженной информации добавляется зашумленный графический элемент.

Для выделения потенциально конфиденциальной информации производим открытие изображения и выбираем интересующие области по переданным координатам, чтобы в последствии применить размытие к словам или словосочетаниям.

В качестве используемого языка разработки был выбран Python, а для работы с изображениями библиотека компьютерного зрения с открытым исходным кодом – OpenCV. OpenCV имеет множество функций и алгоритмов обработки изображений, а также поддерживает работу с различными типами изображений и видео (включая цветные и черно-белые, статические и видеофайлы).

На рис. 2 представлена диаграмма классов разработанного программного обеспечения.

Класс Processing занимается работой с файлами, предварительной обработкой данных.

Класс Analysis предназначен для получения всех данных с изображения и поиска информации конфиденциального характера. Распознавание текста на изображении осуществляется с помощью библиотеки PyTesseract [4]. Для поиска и вычисления координат интересующих подстрок применяются регулярные выражения от библиотеки re в соответствии со списком регулярных выражений, которые передаются из класса Regex. Список регулярных выражений хранится в файле .json (рис. 3).

Регулярные выражения описаны для конфиденциальной информации, которая может встретиться в документе в виде указанного формата: ФИО в полном формате, ФИО в сокращенном виде «через точку», дата, почта, серия и номер паспорта, свидетельство о рождении, адрес, номер, домашний номер.

Класс Blur используется для удаления интересующей информации путем внесения мешающих искажений с помощью метода размытия по среднему значению в соответствии с передаваемыми параметрами функции размытия и размером ядра.

5. Описание интерфейса

Главное окно графического пользовательского интерфейса для разработанного программного средства защиты представлено на рис. 4. В верхней строке меню расположены кнопки “Файл” и “Защита”. В подменю “Файл” реализуются возможности открытия файла и выхода из программы. Реализован просмотр всех страниц документа с помощью кнопок перелистывания изображений вперед и назад. Под страницами загруженного файла появляется информация о номере текущей страницы и о количестве страниц в документе.

В подменю “Защита” расположены кнопки, позволяющие как скрыть данные, так и шифровать весь документ, что является дополнительной функцией.

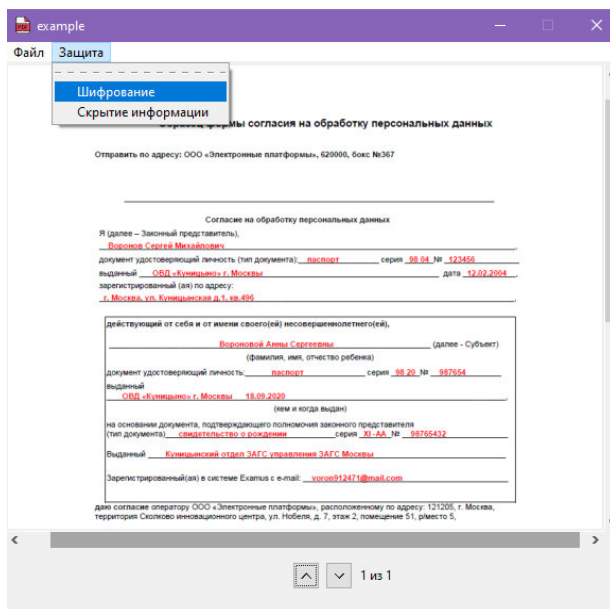


Рис. 4. Главная панель

В окне “Скрытие данных в файле” (рис. 5) нажав на кнопку “Файл” пользователь выбирает путь к файлу. После нажатия кнопки “Ок” программа приступает к обработке файла.

В окне “Шифрование и доступ к файлу” (рис. 6) требуется задать свой пароль для документа. Перейдя по кнопке “Справка”, пользователь может посмотреть дополнительную информацию и критерии задания пароля.

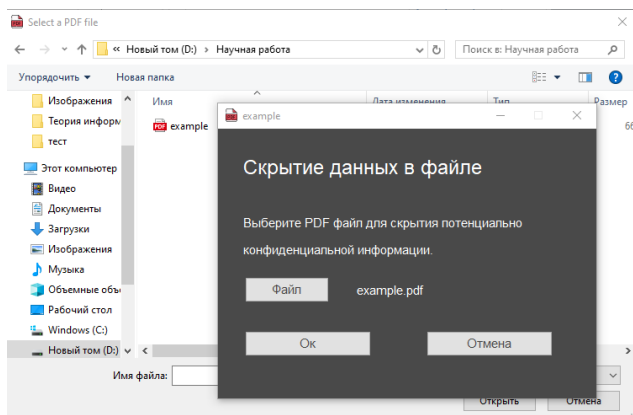


Рис. 5. Выбор документа в окне “Скрытие данных в файле”

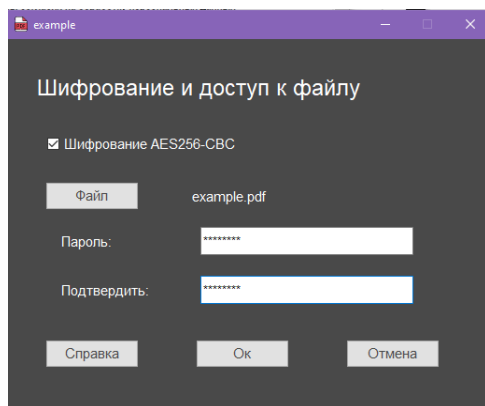


Рис. 6. Интерфейс окна “Шифрование и доступ к файлу”

6. Тестирование реализованной программы

Для тестирования функционирования программного обеспечения был выбран бланк формы согласия на обработку персональных данных, в которой содержатся персональные данные пользователя (рис. 7).

Использование разработанного программного обеспечения позволило скрыть информацию конфиденциального характера для возможности размещения документов в открытом доступе (рис. 8).

Образец формы согласия на обработку персональных данных

Отправить по адресу: ООО «Электронные платформы», 620000, бокс №367

Согласие на обработку персональных данных

Я (далее – Законный представитель),
Воронов Сергей Михайлович

документ удостоверяющий личность (тип документа): паспорт серия 98 04 № 123456
выданный ОВД «Кунцыно» г. Москвы дата 12.02.2004
зарегистрированный (ая) по адресу:
г. Москва, ул. Кунцынская д.1, кв.496

действующий от себя и от имени своего(ей) несовершеннолетнего(ей),
Вороновой Анны Сергеевны (далее - Субъект)
(фамилия, имя, отчество ребенка)
документ удостоверяющий личность: паспорт серия 98 20 № 987654
выданный ОВД «Кунцыно» г. Москвы 18.09.2020
(кем и когда выдан)
на основании документа, подтверждающего полномочия законного представителя
(тип документа): свидетельство о рождении серия XI-AA № 98765432
Выданный Кунцынский отдел ЗАГС управления ЗАГС Москвы
Зарегистрированный(ая) в системе Едактилс с e-mail: voron912471@mail.com

даю согласие оператору ООО «Электронные платформы», расположенному по адресу: 121205, г. Москва, территория Сколково инновационного центра, ул. Нобеля, д. 7, этаж 2, помещение 51, место 5,

на обработку с использованием средств автоматизации и без использования таких средств, включая сбор, запись, систематизацию, накопление, хранение, извлечение, использование, передачу (предоставление, доступ), блокирование, удаление, уничтожение,

своих персональных данных и персональных данных Учащегося, на следующих условиях:

1. Оператор осуществляет обработку персональных данных Субъекта исключительно в целях мониторинга процесса проведения контрольных испытаний.
2. Оператор осуществляет обработку персональных данных Законного представителя исключительно в целях получения согласия на обработку персональных данных Субъекта
3. Для достижения заявленных целей процесс обработки персональных данных может быть поручен следующим операторам:
 - ООО «Mail.ru», адрес места нахождения: 125167, г. Москва, Ленинградский пр-кт, д. 39, стр. 79,
 - Ассоциация «Национальная платформа открытого образования», ОГРН 1157700009136, адрес: г. Москва, Большой Чудов пер., д.8/1,а также образовательному учреждению, проводящему контрольное испытание.

1

Рис. 7. Форма согласия на обработку персональных данных

Заключение

Обеспечение безопасности информации конфиденциального характера от несанкционированного доступа является важной задачей в современном мире, особенно при условии необходимости публикации документов в открытом доступе. В данной статье был рассмотрен алгоритм и реализованное программное средство для скрытия информации конфиденциального характера в pdf-файлах по специальным маскам, и шифрования pdf-файла в качестве дополнительной меры защиты. Разработанное программное средства способствует соблюдению требований по безопасной публикации информации.

Образец формы согласия на обработку персональных данных

Отправить по адресу: ООО «Электронные платформы», 620000, бокс №367

Согласие на обработку персональных данных

Я (далее – Законный представитель): _____

документ удостоверяющий личность (тип документа): паспорт серия _____

выданный ОБД «Кутянецово, г. Москва _____ дата _____

зарегистрированный (ая) по адресу: г.Москва, ул.Новослободская, д.100/101

действующий от себя и от имени своего(их) несовершеннолетнего(их), _____ (далее – Субъект)

(фамилия, имя, отчество ребенка) _____

документ удостоверяющий личность: паспорт серия _____

выданный ОБД «Кутянецово, г. Москва _____

(имя и когда выдан) _____

на основании документа, подтверждающего полномочия законного представителя (тип документа): свидетельство о рождении серия _____

выданный Кутянецовский отдел ЗАГС управления ЗАГС Москвы

Зарегистрированный(ая) в системе Едактил с e-mail: _____

даю согласие оператору ООО «Электронные платформы», расположенному по адресу: 121205, г. Москва, территория Сколково инновационного центра, ул. Нобеля, д. 7, этаж 2, помещение 5/1, место 5,

на обработку с использованием средств автоматизации и без использования таких средств, включая сбор, запись, систематизацию, накопление, хранение, извлечение, использование, передачу (предоставление, доступ), блокирование, удаление, уничтожение,

своих персональных данных и персональных данных Учащегося, на следующих условиях:

1. Оператор осуществляет обработку персональных данных Субъекта исключительно в целях мониторинга процесса проведения контрольных испытаний.
2. Оператор осуществляет обработку персональных данных Законного представителя исключительно в целях получения согласия на обработку персональных данных Субъекта
3. Для достижения заявленных целей процесс обработки персональных данных может быть поручен следующим операторам:
 - ООО «Mail.ru», адрес места нахождения: 125167, г. Москва, Ленинградский пр-кт, д. 39, стр. 79.
 - Ассоциация «Индустриальная платформа открытого образования», ОГРН 1157700009136, адрес: г. Москва, Большой Чудов пер., д.9/1.а также образовательному учреждению, проводящему контрольное испытание.

1

Рис. 8. Результат обработки файла программой

Список литературы

1. Что такое оптическое распознавание символов [Электронный ресурс] : статья. – Режим доступа: <https://aws.amazon.com/ru/what-is/ocr/>
2. Сравнительный анализ методов машинного обучения для решения задачи классификации документов в научно-образовательного учреждения / М. Н. Краснянский [и др.] // Вестник ВГУ. Серия: Системный анализ и информационные технологии. – 2018. – №3. – С. 173-18.
3. Tesseract OCR: What Is It, and Why Would You Choose It in 2024? [Электронный ресурс] : статья. – Режим доступа: <https://www.klippa.com/en/blog/information/tesseract-ocr/>
4. The Config File [Электронный ресурс] : статья. – Режим доступа: <https://pytesseract.readthedocs.io/en/latest/config.html#general-options>
5. AES: Стандарт шифрования AES [Электронный ресурс] : статья. – Режим доступа: <https://electromicro.ru/resources/wiki/aes/aes1/>

Разработка информационной системы для автоматизации формирования отчетных документов аналитика Центрального Федерального Округа компании ООО «Партнер»

И. Д. Наумова

Студент бакалавр

Е. А. Копытина

Старший преподаватель

Введение

Работа аналитика в компании включает формирование отчетных документов в виде сводных Excel таблиц. Информация для отчетов выбирается из различных таблиц вручную и вставляется в другие документы с четкой структурой. Эта рутинная задача требует много времени и подвержена ошибкам, как показано в [1-6]. Автоматизация этого процесса актуальна и важна, так как позволяет экономить ресурсы, снижать ошибки и повышать производительность. Автоматизированные системы способны выполнять задачи быстрее и точнее, а также эффективно управлять большим объемом данных. Они легко адаптируются к изменениям в бизнес-процессах и требованиях компании, учитывая ее уникальные потребности и обеспечивая безопасность данных. Это делает автоматизацию ключевым элементом повышения эффективности и конкурентоспособности компании. Автоматизация также улучшает эффективность бизнес-процессов, позволяя сотрудникам фокусироваться на стратегически важных задачах. Создание программного обеспечения, адаптированного к уникальным потребностям компании, позволяет избежать лишней функциональности и уменьшить зависимость от сторонних поставщиков. Такие решения обеспечивают лучший контроль и безопасность данных, что крайне важно в чувствительных сферах бизнеса. Итак, автоматизация играет ключевую роль в улучшении операционной эффективности и обеспечении долгосрочного успеха компании.

1. Постановка задачи

К описываемому в статье приложению для формирования отчетных документов были выдвинуты следующие требования:

- Чтение данных из указанных пользователем рабочих таблиц Excel;
- Обработка, если необходимо, полученных данных;
- Генерация отчетной таблицы Excel в указанной предопределенной форме, на основе информации, полученной из рабочих таблиц;
- Проверка, если требуется, корректности полученных в отчете итоговых значений.

2. Реализация логики

Для написания десктопного приложения AutomAn использовался язык программирования Python.

В реализации также были использованы следующие библиотеки:

- PyQt5,
- openpyxl.

При разработке приложения использовался паттерн MVC (Model View Controller). Этот паттерн разделяет приложение на три компонента: модель, которая отвечает за данные и их обработку, представление, которое отображает данные пользователю и контроллер, который управляет взаимодействием между моделью и представлением.

Для разработки бизнес-логики приложения использовалась модульная архитектура. Данная архитектура подходит для параллельной разработки, так как модули чаще всего независимы, взаимозаменяемы и переиспользуемы, можно подключать и отключать модули по требованию.

Каталог controller содержит контроллеры, которые обрабатывают запросы от пользователя и взаимодействуют с моделью для обработки данных, а также с представлением для отображения результата пользователю.

Далее расположен каталог model. Здесь расположены модели, которые отвечают за бизнес-логику и обработку данных. В данном случае каталог разделен еще на три. Так, в модуле processor находятся обработчики данных из таблиц, в файлах данного модуля реализована основная логика обработки данных из отчетов. Модуль reader содержит файлы, в которых определяются методы для чтения Excel таблиц, а модуль writer – для записи данных в Excel таблицы.

Каталог view содержит представления, которые отвечают за отображение данных пользователю. В данном случае здесь содержатся

шаблоны окон, а также определяются методы для визуализации данных и взаимодействия с пользователем.

Каталог `utils` используется для размещения вспомогательных данных, которые не относятся напрямую к модели, представлению или контроллеру. Например, там располагается иконка приложения.

В корне проекта находится файл `main.py`, это основной файл проекта, который запускает приложение AutomAn. Структура проекта приложения показана на рис. 1.

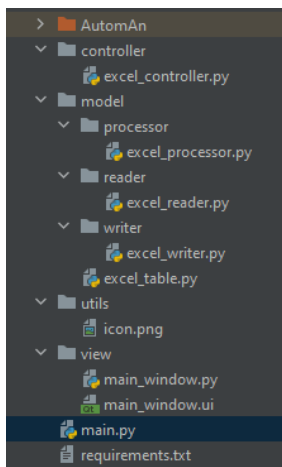


Рис. 1. Структура проекта

Также, в корне проекта находится файл `requirements.txt`. Это файл зависимостей, в котором перечислены все библиотеки, необходимые для работы приложения.

3. Реализация интерфейса

Для более быстрого и наглядного создания пользовательского интерфейса использовался вспомогательный инструмент Qt Designer. Это кроссплатформенная среда для разработки графических интерфейсов (GUI) для программ, использующих библиотеку Qt.

Пример разработки главного окна приложения представлен на рис. 2. Разработанный интерфейс сохраняется в файл с расширением `ui`, который подключается к разработанной программе с помощью специальных методов библиотеки Qt. Этот файл имеет `xml`-формат, и может, в случае необходимости, редактироваться в любом текстовом редакторе.

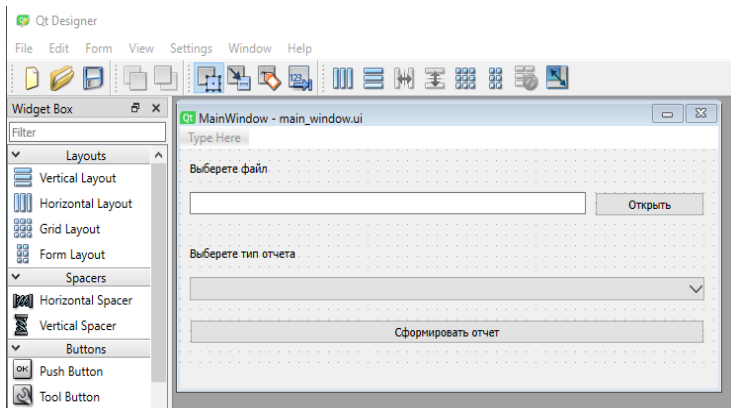


Рис. 2. Окно Qt Designer

При запуске приложения, пользователь видит окно, представленное на рис. 3.

В верхней части окна находится поле для ввода пути к исходному файлу и кнопка для его поиска в компьютере.

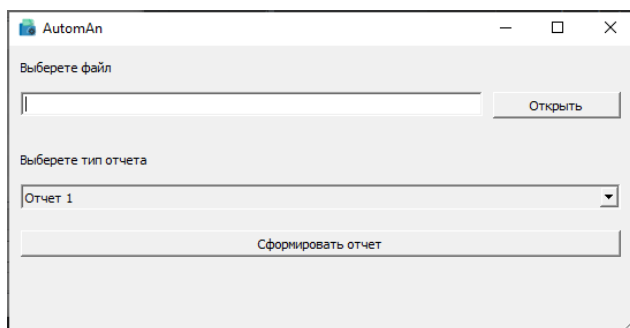


Рис. 3. Окно программы

При нажатии на кнопку «Открыть» открывается стандартное диалоговое окно поиска, представленное на рис. 4. При этом, как можно заметить на данном рисунке, выбрать можно только файлы формата Excel.

Когда входной файл выбран, можно переходить к выбору типа генерируемого отчета. Выбор делается в нижней части окна в выпадающем меню. Пример выбора представлен на рис. 5.

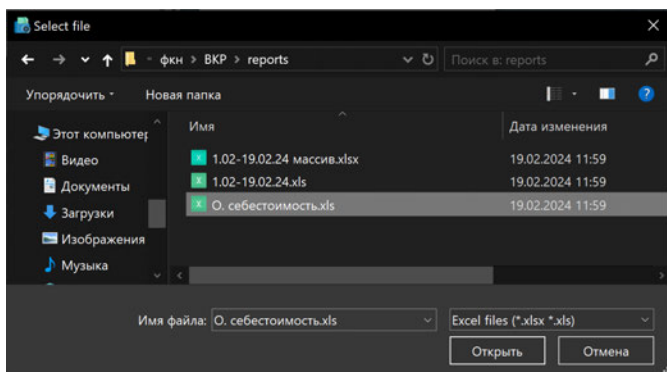


Рис. 4. Окно выбора входного файла

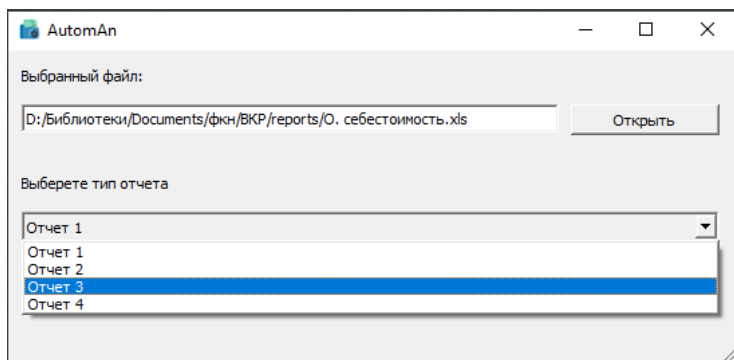


Рис. 5. Выбор типа отчета

После проделанных действий можно сформировать требуемый отчет. Для этого надо нажать на кнопку «Сформировать». По нажатию вызывается метод обработки входной таблицы.

Когда обработка завершается, появляется стандартное окно сохранения файла, часть которого представлена на рис. 6.

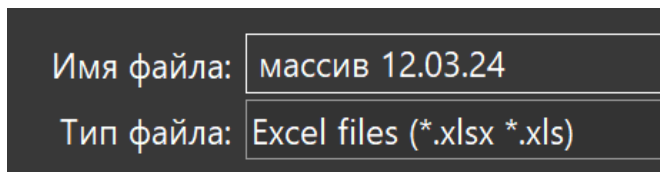


Рис. 6. Фрагмент окна сохранения

Здесь можно задать имя сформированному отчету. Доступные форматы файла ограничены форматами Excel.

После сохранения новый файл, с указанным именем и расположением, появится в системе. Результат сохранения показан на рис. 7.

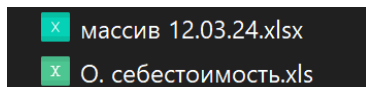


Рис. 7. Сформированный и исходный файл

Полученный с помощью приложения отчет представлен далее на рис. 8.

	A	B
1		
2		
3	Сумма по полю Сумма в ценах закупки без скидки	
4	сети	Итого
5	А Пиво	740,58
6	Алкобренд	50 919,28
7	БирМир	26 832,04
8	Газпром	56 266,22
9	Доставка	533 556,60
10	Идея	685,60
11	Лукойл	121 572,96
12	Партнер-Ориенталь	1 099,31
13	розница	187 408,33
14	Табачный ряд	10 671,31
15	Хмельбург	2 335,35
16	Хмельная миля	33 136,11
17	Центрторг	148 491,40
18	(пусто)	
19	Общий итог	1 173 715,09
20		
21		

Рис. 8. Сформированный файл Excel

Заключение

В данной статье была рассмотрена разработка информационной системы для автоматизации формирования отчетных документов аналитика компании ООО «Партнер». Разработанный программный продукт удовлетворяет всем заявленным функциональным требованиям, а также является актуальным, востребованным и работоспособным.

Список литературы

1. Свидетельство о государственной регистрации программы для ЭВМ № 2023661037 Российская Федерация. Приложение для автоматизации документооборота дополнительного образования : № 2023618813 : заявл. 03.05.2023 : опубл. 25.05.2023 / А. С. Голышева, Е. А. Копытина ; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Воронежский государственный технический университет». – EDN NBUVMM.

2. Быков, М. Н. Разработка парсера для автоматизации работы с учебными планами факультета компьютерных наук Воронежского государственного университета / М. Н. Быков, Е. А. Копытина // Информатика: проблемы, методы, технологии : материалы XXIII Международной научно-практической конференции им. Э.К. Алгазинова, Воронеж, 15–17 февраля 2023 года. – Воронеж: Воронежский государственный университет, 2023. – С. 768-772. – EDN HSGNJG.

3. Разработка программного комплекса для автоматизации деятельности Контрольно-счетной палаты Воронежской области / А. В. Порядин, И. И. Дмитриева, И. Р. Корнилов [и др.] // Информатика: проблемы, методы, технологии : материалы XXIII Международной научно-практической конференции им. Э.К. Алгазинова, Воронеж, 15–17 февраля 2023 года. – Воронеж: Воронежский государственный университет, 2023. – С. 901-906. – EDN GCRTDP.

4. Претков, А. И. Разработка программного комплекса для автоматизации формирования документов основных профессиональных образовательных программ высшего образования факультета компьютерных наук Воронежского государственного университета / А. И. Претков, Е. А. Копытина // Информатика: проблемы, методы, технологии : материалы XXIII Международной научно-практической конференции им. Э.К. Алгазинова, Воронеж, 15–17 февраля 2023 года. – Воронеж: Воронежский государственный университет, 2023. – С. 907-912. – EDN ENWODO.

5. Свидетельство о государственной регистрации программы для ЭВМ № 2022617688 Российская Федерация. Система анализа данных правонарушений наркоконтроля по Воронежской области методами

машинного обучения : № 2022617044 : заявл. 19.04.2022 : опубл. 25.04.2022 / Е. А. Копытина, А. Н. Кропачев ; заявитель федеральное государственное бюджетное образовательное учреждение высшего образования «Воронежский государственный университет». – EDN SEFGOS.

6. Копытина, Е. А. Разработка веб-сервиса для формирования горячего коммерческого предложения ООО «ЧерноземАгромаш» / Е. А. Копытина, Ф. А. Десятириков, А. В. Копытин // Информатика: проблемы, методы, технологии : Материалы XXII Международной научно-практической конференции им. Э.К. Алгаинова, Воронеж, 10–12 февраля 2022 года / Под редакцией Д.Н. Борисова. – Воронеж: Общество с ограниченной ответственностью "Вэлборн", 2022. – С. 990-996. – EDN SXFFFI.

О константах Рисса для систем целочисленных сдвигов, порожденных окнами Ханна и Хэмминга

Д. Р. Небогина

Студент бакалавр

Р. К. Рыкунов

Студент бакалавр

Е. А. Киселев

Доцент

Введение

В настоящее время активно продолжается создание и совершенствование алгоритмов цифровой обработки сигналов. Конец XX века и начало XXI века можно назвать эпохой широкого внедрения вейвлетов. Однако в последнее время многие исследователи вновь обратили внимание на системы равномерных сдвигов и оконные семейства функций, поскольку они обладают более простой структурой и ясной физической интерпретацией [1].

Для систем равномерных сдвигов разработана общая математическая теория [1, 2]. Однако в зависимости от выбора конкретной порождающей функции они могут сильно отличаться по своим свойствам. В качестве примера можно назвать системы равномерных сдвигов функций Гаусса и контура Лоренца, исследованные в статьях [3] и [4]. Авторы этих работ обнаружили качественные различия в поведении коэффициентов узловых функций, которые играют важную роль в задаче об интерполяции. Следовательно, выбор в пользу функций того или иного вида должен осуществляться с учетом специфики решаемой прикладной задачи.

Целью данной работы является изучение некоторых особенностей систем целочисленных сдвигов окон Ханна и Хэмминга. Эти функции множество раз успешно применялись в задачах цифровой обработки сигналов. Нами проводится численное исследование особенностей процедуры разложения сигналов по этим функциям во временной области, что в перспективе должно позволить упростить поиск оптимальных параметров для решения конкретных прикладных задач.

1. Теоретическая часть

Рассматривается пространство $L_2(\mathbb{R})$. Скалярное произведение и норма задаются обычным образом

$$(f, g) = \int_{-\infty}^{\infty} f(x) \overline{g(x)} dx, \|f\| = \sqrt{(f, f)}. \quad (1)$$

Прямое и обратное преобразование Фурье имеет вид

$$\hat{f}(\xi) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x) e^{-ix\xi} dx, f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{f}(\xi) e^{ix\xi} d\xi. \quad (2)$$

Исследуется семейство равномерных сдвигов $\phi(x - kh)$, $k \in \mathbb{Z}$, $h > 0$, где

$$\phi(x) = \begin{cases} \alpha + \beta \cos\left(\frac{2x}{\sigma}\right), & |x| \leq \frac{\sigma\pi}{2}, \\ 0, & |x| > \frac{\sigma\pi}{2}. \end{cases} \quad (3)$$

Параметр $\sigma > 0$ в формуле (3) задает характерную ширину $\phi(x)$. При $\alpha = \beta = 0.5$ функция $\phi(x)$ называется окном Ханна, а при $\alpha = 0.53836$, $\beta = 0.46164$ – окном Хэмминга. Это очень удобно, поскольку оба рассматриваемых случая описываются едиными формулами. Также для простоты в дальнейшем будем полагать, что $h = 1$. Это не ограничивает общности, поскольку на практике равномерный шаг дискретизации за счет замены переменной всегда может быть сделан равным единице.

Важными характеристиками, позволяющими оценить устойчивость неортогональных систем функций, являются константы Рисса [1, 2].

Определение 1. Функции $\phi_k(x)$, $k \in \mathbb{Z}$ образуют систему Рисса, если существуют константы $0 < A < B < \infty$ такие, что выполняется двусторонняя оценка

$$A \sum_{k=-\infty}^{\infty} |c_k|^2 \leq \left\| \sum_{k=-\infty}^{\infty} c_k \phi_k(x) \right\|^2 \leq B \sum_{k=-\infty}^{\infty} |c_k|^2, \quad (4)$$

где c_k – любой набор коэффициентов, удовлетворяющих условию

$$\sum_{k=-\infty}^{\infty} |c_k|^2 < \infty, \quad (5)$$

Наилучшие оценки для величины A и B в формуле (4) называются нижней и верхней константами Рисса, соответственно. Если

система функций $\phi_k(x)$, $k \in \mathbb{Z}$ ортонормированна, то $A = B = 1$. Для произвольных систем Рисса $B/A \geq 1$. Отношение B/A влияет на скорость сходимости итерационных алгоритмов разложения по $\phi_k(x)$, $k \in \mathbb{Z}$, а также на устойчивость процедуры нахождения коэффициентов c_k . Чем ближе B/A к единице, тем лучше скорость и надежность работы алгоритмов [1].

Для систем равномерных сдвигов известно следующее важное утверждение [1, 2].

Теорема 1. Функции $\phi(x-k)$, $k \in \mathbb{Z}$ образуют систему Рисса с константами $0 < A < B < \infty$, если почти всюду на отрезке $\xi \in [-\pi, \pi]$ выполняется двусторонняя оценка

$$A \sum_{k=-\infty}^{\infty} |c_k|^2 \leq \left\| \sum_{k=-\infty}^{\infty} c_k \phi_k(x) \right\|^2 \leq B \sum_{k=-\infty}^{\infty} |c_k|^2, \quad (6)$$

где

$$P(\xi) = 2\pi \sum_{k=-\infty}^{\infty} \left| \hat{\phi}(\xi + 2\pi k) \right|^2. \quad (7)$$

Теорема 1 означает, что для нахождения констант Рисса достаточно найти $\min P(\xi)$ и $\max P(\xi)$ на отрезке $\xi \in [-\pi, \pi]$.

Отметим, что для систем Рисса можно построить так называемую биортонормальную систему [1], которая является эффективным инструментом для нахождения коэффициентов разложения.

2. Вычислительные эксперименты

Ключевую роль в дальнейшем играет вспомогательная функция (7). Для ее расчета нам понадобится преобразование Фурье функции (3):

$$\hat{\phi}(\xi) = \sqrt{\frac{2}{\pi}} \frac{\sin(0.5\sigma\pi\xi)}{\xi} \left(\alpha - \beta + \frac{4\beta}{4 - \sigma^2\xi^2} \right). \quad (8)$$

Сумму в формуле (7), к сожалению, удалось найти только численно, поэтому все дальнейшие расчеты носят приближенный характер.

Как показали результаты вычислений, в диапазоне значений σ от 0.5 до 1.5 функция $P(\xi)$ достигает минимального значения в точке $\xi = \pi$, а максимального – в точке $\xi = 0$. График $P(\xi)$ при разных значениях σ представлен на рис. 1–3.

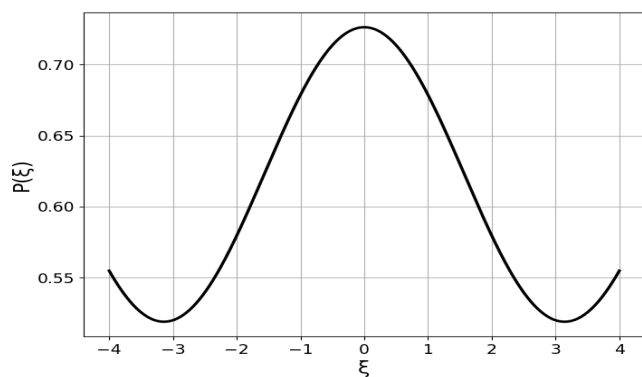


Рис. 1. График функции $P(\xi)$ при $\sigma = 0.5$

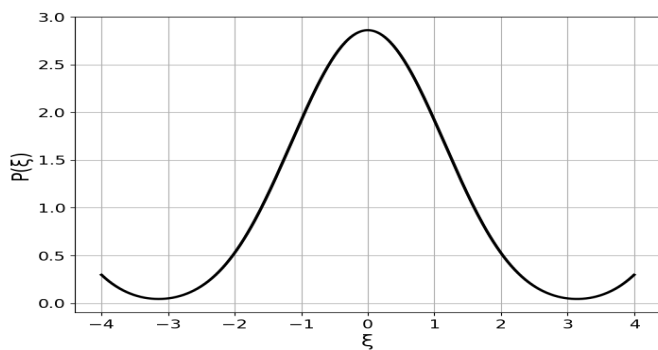


Рис. 2. График функции $P(\xi)$ при $\sigma = 1.0$

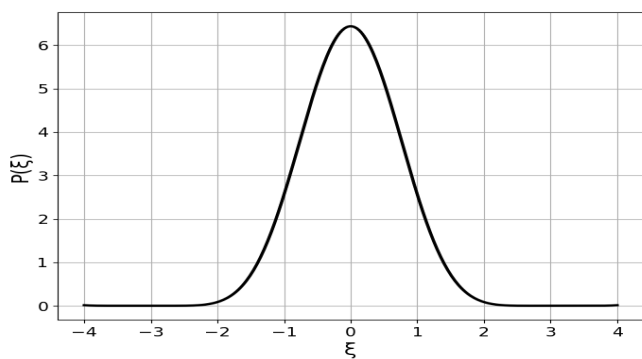


Рис. 3. График функции $P(\xi)$ при $\sigma = 1.5$

В таблице содержатся значения отношения констант Рисса для двух рассматриваемых систем функций при разных значениях параметра σ .

Таблица

Отношение констант Рисса B_1 / A_1 и B_2 / A_2 для систем сдвигов, порожденных окнами Ханна и Хэмминга, соответственно

σ	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2
B_1 / A_1	1.19	1.70	2.76	5.02	10.6	27.6	102.0	910.5
B_2 / A_2	1.40	2.22	4.02	8.34	20.7	65.1	298.8	$2.46 \cdot 10^3$

Заключение

Данная статья посвящена оценке устойчивости систем равномерных сдвигов, порожденных окнами Ханна и Хэмминга. Как видим отношение констант Рисса при тех же значениях σ несколько больше в случае окна Хэмминга, поэтому скорость работы алгоритмов разложения и их точность будет для этой функции несколько хуже в сравнении с окном Ханна. По-видимому, это связано с наличием разрывов у $\phi(x)$ в точках $x = \pm\sigma\pi/2$. Еще один вывод заключается в том, что в обоих случаях следует избегать значений $\sigma > 1.3$, поскольку отношение констант Рисса B/A при $\sigma = 1.3$ достигает величины порядка 10^4 и продолжает расти с увеличением σ .

Список литературы

1. Christensen O. An introduction to frames and Riesz bases / O. Christensen. – Basel : Birkhauser/Springer, 2016. – 704 p.
2. Чуи Ч. Введение в вейвлеты / Ч. Чуи ; пер. с англ. Я. М. Жилейкина. – М. : Мир, 2001. – 412 с.
3. О константах Рисса для некоторых систем целочисленных сдвигов / Е. А. Киселев, Л. А. Минин, И. Я. Новиков, С. М. Ситник // Математические заметки. – 2014. – Т. 96, № 2. – С. 239-250.
4. Минин Л. А. Поведение коэффициентов узловых функций, построенных из равномерных сдвигов функций Гаусса и Лоренца / Л. А. Минин, С. М. Ситник, С. Н. Ушаков // Научные ведомости БелГУ. Серия : Математика. Физика. – 2014. – Т. 183, № 12. – С. 214-217.

Детектирование метеоритных кратеров на спутниковых фотографиях

А. С. Новиков

Студент бакалавр

А. В. Акимов

Старший преподаватель

Введение

В настоящее время детектирование метеоритных кратеров является важной задачей в рамках исследований космических тел. Выявление метеоритных кратеров предоставляет важную информацию о поверхности космического тела и ее эволюции. Плотность кратеров коррелирует с возрастом поверхности, и количество кратеров используется для оценки возраста лунной поверхности [1]. Кратеры также могут дать информацию о направлении удара и популяции метеоритов. Предполагаемые временные изменения скорости образования кратеров используются для того, чтобы делать выводы о динамической истории Солнечной системы. Также изучение кратеров может дать информацию о свойствах поверхности и различных процессах, воздействующих на них. Детектирование кратеров необходимо для определения мест посадки и маршрутизации луноходов, не полагаясь на связь с наземными станциями и в отсутствие систем, подобных глобальной системе позиционирования Земли (GPS).

Традиционно обнаружение кратеров выполнялось вручную, что ограничивало эффективность и точность поиска. В связи с этим, а также с наличием объемных данных о планетных телах необходимость в автоматическом алгоритме стала неизбежна. Данная работа посвящена использованию сверточной нейронной сети для детектирования кратеров.

1. Задача детектирования на изображениях

Детектирование изображений – это задача идентификации объектов на данном изображении, а также локализации обнаруженных объектов с использованием ограничительных рамок и классификации объектов путем их маркировки.

До разработки модели AlexNet в 2012 году [2] обнаружение объектов выполнялось с использованием традиционных методов обработки изображений. Самые распространенные из них включали метод обнаружения объектов Виолы-Джонса, масштабно-инвариантные преобразования объектов (SIFT) и гистограмму ориентированных градиентов (HOG).

С успехом моделей глубокого обучения область обнаружения объектов получила огромный импульс, и начался новый этап, в котором эти модели используются для детектирования объектов. Глубокие нейронные сети превосходно справляются со сложными задачами классификации изображений и обнаружения объектов.

2. Архитектура модели

Для обнаружения кратеров была выбрана модель компьютерного зрения Faster R-CNN, архитектура которой представлена на рис.1. Она состоит из двух основных компонентов: сети предложений регионов (RPN) и сети классификации объектов.

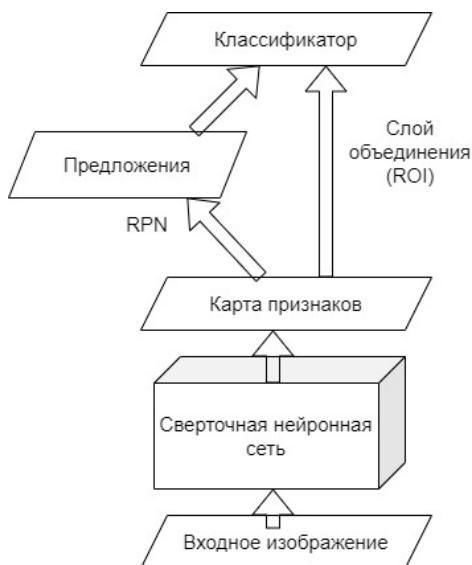


Рис. 1. Архитектура модели глубокого обучения на основе Faster R-CNN

Сначала входное изображение пропускается через сверточную нейронную сеть для извлечения карт признаков. На каждом из слоев свертки применяются фильтры для извлечения признаков из входного

изображения. Распространённой практикой является увеличение количества фильтров с каждым уровнем в сверточной сети. Когда изображение проходит через первый слой сети, сеть выделяет базовые черты. Затем на следующем слое она комбинирует эти базовые черты, чтобы выделить более сложные характеристики. Это требует большего количества фильтров, чтобы учесть разнообразие возможных признаков [3].

Основная идея сверточной нейронной сети заключается в том, чтобы позволить сети извлекать иерархические и более абстрактные признаки из входных данных, начиная с простых текстур и форм на низких уровнях иерархии и заканчивая сложными объектами и концепциями на более высоких уровнях.

Сеть предложений регионов затем принимает карты признаков последнего сверточного слоя в качестве входных данных и выдает набор регионов вероятного расположения объектов. В частности, RPN генерирует предложения регионов, накладывая небольшое окно на различные места входного изображения и подавая эти регионы в слой регрессии рамок и слой классификации рамок для уточнения местоположения объектов. Затем слой объединения (ROI) берёт предложения регионов и масштабирует их в карты признаков фиксированного размера, которые могут быть классифицированы сетью обнаружения объектов. Эта сеть выдает окончательную классификационную метку и дальнейшее уточнение положения ограничивающей рамки объекта.

В качестве основы для Faster R-CNN была выбрана остаточная нейронная сеть ResNet. До разработки ResNet, наращивание количества слоев для повышения глубины нейронной сети рано или поздно вызывало ограничение или даже быстрое ухудшение производительности сети. Это происходило из-за проблемы исчезающих градиентов, которая возникает из-за того, что множество функций активации сокращают большое входное пространство до гораздо меньшего выходного. Для решения этой проблемы в ResNet были введены остаточные блоки (residual blocks), представленные на рис. 2.

Этот остаточный блок представляет простую концепцию добавления промежуточных входных данных к выходным данным серии слоёв свертки. Выход остаточного блока $H(x) = F(x) + x$ где x – это входные данные для остаточного блока и выходные данные предыдущего уровня, а $F(x)$ – это часть сверточной сети, состоящая из нескольких слоев свертки [5].

То есть, слои, составляющие функцию, должны аппроксимировать не всю функцию целиком, а лишь её остаток. В остаточном блоке нам

достаточно обучить сеть воспроизводить не весь сигнал целиком, а лишь отличия входных значений для получения требуемой функции. Можно считать, что каждый блок теперь настраивает вывод предыдущего блока, и ему не приходится генерировать желаемый вывод с нуля.

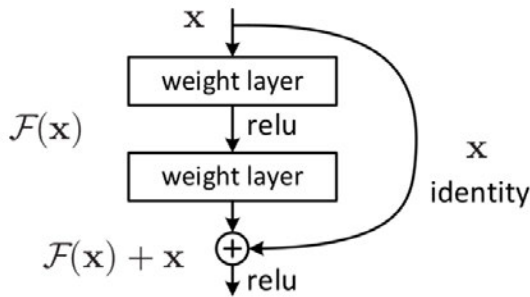


Рис. 2. Остаточный блок из [4]

3. Реализация и тестирование модели

Как и говорилось ранее, для этой работы был загружен Faster R-CNN с основой ResNet-50, который был предварительно обучен на наборе данных COCO.

Слой ROI был заменён на новый для обучения на нашем наборе данных. Он принимает на вход количество входных признаков (`in_features`) и имеет количество выходных слоев (`num_classes`), соответствующих количеству классов, которые модель должна определить.

Модель была обучена на наборе данных из [6]. В данном наборе 143 изображения, которые поделены на три части:

- 98 для обучающей выборки;
- 26 для валидационной выборки;
- 19 для тестовой выборки.

Результаты работы модели на тестовой выборке представлены в таблице 1 и на рис. 3.

Метрики $mAP@.5$ и $mAP@.5:.95$, которые представляют собой среднюю точность обнаружения объектов на определенных уровнях IoU (Intersection over Union). IoU – это показатель, который определяет степень перекрытия между предсказанной и реальной границей объекта.

Метрика $mAP@.5$ показывает среднюю точность обнаружения на уровне IoU 0.5, что означает, что объект считается правильно обнаруженным, если предсказанная граница пересекается с реальной на

50% или больше. Метрика $mAP@.5:.95$ показывает среднюю точность обнаружения на уровнях IoU от 50% до 95% с шагом 5%, что позволяет оценить производительность модели при разных пороговых значениях обнаружения.

Таблица 1

Метрики тестирования модели

Метрика	Значение
$mAP@.5$	0.668
$mAP@.5:.95$	0.470



Рис. 3. Результат работы модели на тестовой выборке

Синие прямоугольники ограничивают области реального расположения кратеров, красные – области обнаружения кратеров детектором. Можно заметить, что модель лучше справляется с кратерами больших размеров, в то время как некоторые небольшие области и объекты, не являющиеся кратерами, иногда ошибочно принимаются за кратеры.

В целом алгоритм показывает удовлетворительные результаты, что видно из метрик $mAP@.5 = 0.668$ и $mAP@.5:.95 = 0.470$. Для увеличения производительности модели, в том числе и для лучшей работы с кратерами меньшего размера, следует обучить её на большем наборе данных, чтобы нейронная сеть смогла извлечь больше признаков.

Заключение

В результате выполнения данной работы была разработана модель детектирования кратеров на спутниковых фотографиях. В качестве алгоритмов были рассмотрены и использованы детектор Faster R-CNN и нейронная сеть ResNet.

Также эти методы были реализованы программно, что позволило проверить их работу на тестовом наборе изображений. Была получена средняя точность обнаружения при пороге IoU 0.5 равная 66.8%.

Список литературы

1. Bland, P. Crater counting / P. Bland // *Astronomy & Geophysics*. – 2003. – Т. 44. – №. 4. – С. 4.21-4.21.
2. Krizhevsky, A. Imagenet classification with deep convolutional neural networks / A. Krizhevsky, I. Sutskever, G. E. Hinton // *Advances in neural information processing systems*. – 2012. – Т. 25.
3. Li, W. Recognizing terrain features on terrestrial surface using a deep learning model: An example with crater detection / W. Li // *Proceedings of the 1st Workshop on Artificial Intelligence and Deep Learning for Geographic Knowledge Discovery*. – 2017. – С. 33-36.
4. Deep residual learning for image recognition / He K. [et al.] // *Proceedings of the IEEE conference on computer vision and pattern recognition*. – 2016. – С. 770-778.
5. Распознавание сцен на изображениях с помощью глубоких свёрточных нейронных сетей [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/companies/vk/articles/359214/>
6. Mars/Lunar Crater Computer Vision Project [Электронный ресурс]. – Режим доступа: <https://universe.roboflow.com/lincoln-zhou/mars-lunar-crater>

Выделение именованных сущностей и отношений из предложений на естественном языке на основе контекста

Т. С. Покушалова

Студент бакалавр

В. В. Гаршина

Доцент

Введение

В современном информационном обществе огромное количество данных генерируется ежедневно, особенно в контексте текстовых материалов на естественных языках. Объем информации и её накопление в мире стремительно растут. Люди уже не могут обойтись без доступа к интернету. Однако при поиске нужной информации сталкиваются с рядом трудностей: запросы часто неоднозначны, многие источники информации слабо структурированы, а компьютеры не могут понять человеческий язык напрямую. В результате поиска чаще всего получается большой объем данных на похожие темы, а не конкретный ответ на вопрос. Именно поэтому анализ текстов стал важным направлением развития в различных областях.

Одной из ключевых задач в области обработки естественного языка является выделение информации из текста, а именно выделение именованных сущностей и отношений между ними. Это имеет важное значение для различных областей, таких как информационный поиск, извлечение знаний, анализ социальных сетей и биомедицинская область. Существующие методы выделения именованных сущностей и отношений все еще сталкиваются с вызовами, такими как многообразие языков, неоднозначность контекста и необходимость работать с огромным объемом данных. Поэтому совершенствование существующих методов и разработка новых подходов к решению этой задачи являются актуальными и перспективными направлениями исследований.

Данная работа посвящена проведению сравнения существующих методов извлечения именованных сущностей, а также сбору и анализу датасета для дальнейшего дообучения модели на предметную область.

1. Именованные сущности

Именованной сущностью считается слово или словосочетание, предназначенное для конкретного и вполне определенного предмета или явления, выделяющего этот предмет или явление из ряда однотипных предметов или явлений (конкретные объекты или понятия, которые можно однозначно определить и выделить из текста, например, имена людей, названия организаций, местоположения, даты и времена, товары и услуги и т. д.). Отношения между этими сущностями выражают связи и взаимодействия между ними, что позволяет лучше понять смысл текста и извлечь ценные знания из него. В области обработки естественного языка (NLP) и выделения информации (IE) задача выделения именованных сущностей (NER) и отношений между ними является важным направлением.

2. Применяемые известные подходы к решению задачи выделения именованных сущностей

В настоящее время существует несколько различных подходов для задачи извлечения именованных сущностей из текста. Например, это можно сделать с помощью нейросетевых моделей. Они представляют собой архитектуры нейронных сетей, обученные на размеченных данных для автоматического извлечения и классификации именованных сущностей в тексте. Они часто используются с различными вариантами рекуррентных нейронных сетей (RNN), сверточных нейронных сетей (CNN) или трансформеров. Нейросетевые модели для NER широко применяются во многих областях, включая обработку естественного языка, биомедицинские исследования, информационный поиск и другие. Они обычно требуют больших объемов размеченных данных для обучения и могут достичь высокой точности на новых данных, особенно если используются предварительно обученные модели.

Подходы на основе глубокого обучения, такие как рекуррентные нейронные сети (Recurrent Neural Networks, RNN) являются классом нейронных сетей, способных работать с последовательными данными, такими как текст. В контексте NER, RNN могут использоваться для моделирования последовательности слов в тексте и предсказания метки класса (сущности) для каждого слова. Они обладают способностью учитывать контекст и зависимости между словами в предложении, что делает их эффективными для задач с разнообразными типами сущностей и сложной структурой текста. RNN были введены в область обработки естественного языка еще в конце 20-го века, но их применение в NER стало широко распространено с развитием глубокого обучения в 2010-х годах.

Сверточные нейронные сети (Convolutional Neural Networks, CNN) изначально разработаны для обработки изображений, но они также могут быть применены к текстовым данным. CNN для NER также получили широкое распространение и показывают хорошие результаты, особенно в случаях, когда важны локальные зависимости в тексте или когда требуется эффективная обработка длинных последовательностей. CNN были разработаны для обработки изображений в 1990-х годах, и их применение к обработке текста стало активно исследоваться в последние десятилетия.

Трансформеры (Transformers) – это относительно новый класс моделей, разработанных специально для обработки последовательных данных, таких как текст. Они используют механизмы внимания для обработки длинных последовательностей и моделирования зависимостей между элементами в них. Трансформеры, особенно модель BERT (Bidirectional Encoder Representations from Transformers), получили огромную популярность в области обработки естественного языка. Они обычно демонстрируют высокую точность и способность к обучению на больших объемах данных, что делает их очень эффективными для NER. Трансформеры были представлены в работе "Attention is All You Need" в 2017 году и быстро стали доминирующими моделями в области обработки естественного языка.

Подходы на основе глубокого обучения, такие как RNN, CNN и трансформеры, обычно обладают высокой точностью и способностью к обобщению на новые данные. Они позволяют учитывать сложные зависимости в тексте и могут быть эффективными в различных задачах NER. Трансформеры, особенно модель BERT, часто считаются наиболее мощными моделями для NER в настоящее время, хотя их использование может потребовать больших вычислительных ресурсов.

Также существуют парсеры для работы с именованными сущностями, например Томита-парсер и Yargy-парсер.

Томита-парсер назван инструмент в честь японского ученого Масару Томита, автора алгоритма GLR-парсинга (Generalized left-to-right algorithm), на основе которого и был создан Томита-парсер. Еще в 1984 году он описал имплементацию этого алгоритма, поставив перед собой задачу эффективно и точно производить анализ текстов на естественном языке. В некотором роде GLR-алгоритм – это расширенная версия алгоритма LR-парсинга. Но LR-алгоритм предназначен для анализа текстов, написанных на достаточно строго детерминированных языках программирования, и с естественным языком работать не может. Томита решил эту проблему путем параллелизации стеков, что позволило рассматривать различные

трактовки тех или иных участков текста: как только возникает возможность различной трактовки, стек разветвляется. Таких последовательных разветвлений может быть несколько, но в процессе анализа ошибочные ветви отбрасываются, и результатом становится наиболее длинная цепочка. При этом алгоритм выдает результаты своей работы в режиме реального времени, по мере продвижения вглубь текста, другие алгоритмы обработки естественного языка такой особенностью не обладают. Парсер может позволить написать свою собственную грамматику, добавить свои словари и запустить их на текстах.

Yargy-парсер в свою очередь является частью большого проекта Natasha и, собственно, основой библиотеки natasha. Работа Yargy-парсера основана на контекстно-свободных грамматиках и словарях, в нем реализован алгоритм Эрли. Парсер использует словари и правила для извлечения структурированной информации из текстов на русском языке. Правила состоят из предикатов, многие из которых уже являются встроенными. Вместе с тем, парсер реализован на python, что делает его использование в проекте чаще всего более удобным, в сравнении с аналогами, такими, как Tomita-парсер от Yandex.

На диаграмме (рис. 1) представлена схема перечисленных решений:

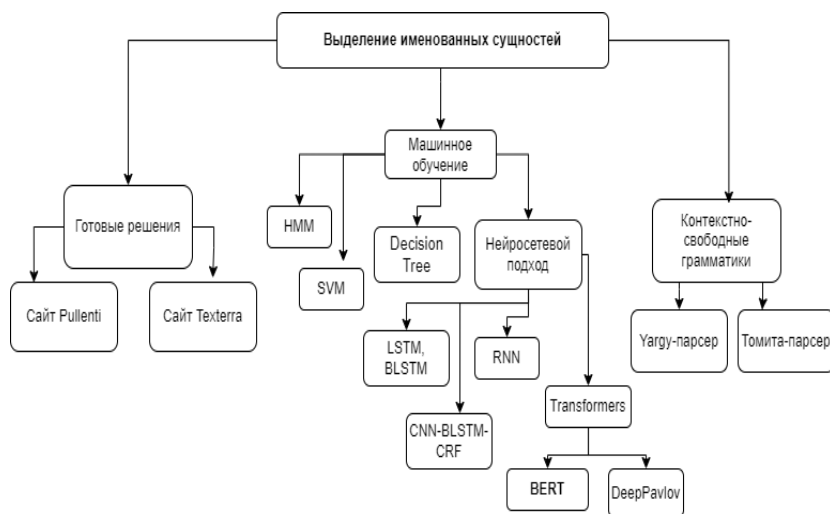


Рис. 1. Обзор систем и подходов к решению задачи

3. Подготовка набора данных

С помощью студентов РГФ был создан датасет, представляющий из себя почтовые сообщения о договорной деятельности организации (рис. 2):

1	sentence
2	проблема: не открывается АКТ сверки по СБС № Пакета документа 3794887, скан прилагаю
3	В случае выбора неверного шаблона просьба указать наименование корректного, по названиям иного
4	Вопрос/информация Добрый день! В ЗНВ 30000196281 в графе "Сумма комиссии" необходимо откорре
5	Вопрос/проблема: После выхода из отпуска с 01.08.16 меню в SAP УВХД стало короче, полагаю отсутств
6	В УВХД необходимо снять резервирование с ЗНС *5792502. М11 не дает поставить статус удалить - выд
7	Подскажите пожалуйста, какую Роль необходимо присвоить, для того, чтобы у сотрудника была возмо
8	Вопрос/Проблема: не вижу ведомости по инвентаризации
9	Добрый день. Проблема при создании спецификации. дог № 50001789222. Спецификация ссылочная.)
10	Вопрос/Проблема: Добрый день! есть права на доступ в УВХД завод автоматически присвоили 6700 Тис
11	Не осуществляется поиск по системному номеру договора, а так же любой поиск контрагента
12	Добрый день! При подписании сотруднику приказа на командирование вы дает ошибку. Пришлось ве
13	Добрый день! По сотруднику #ФИО ЕП (тн 344991) в SAP создана командировка 4400023766. По возвра
14	При выставлении запроса на командировку отсутствует строка для учета проживания.
15	Укажите подробную информацию. Принято по звонку. Приказ на командировку находится в статусе "н
16	Вопрос/проблема: Добрый день! Данные ЗНРС резервируют бюджет: 40001565577 -22 000,00 40001565!
17	Вопрос/информация Добрый день! При согласовании ЗНС программа выдает ошибку согласно вложе
18	Вопрос/Проблема: Просьба отбраковать акт 520518002411 в связи с ошибкой при выборе комиссии.
19	Добрый день! При попытке отправить ЗНС на согласование выдает ошибку. См вложение.
20	По сотруднику #ФИО #ФИО #ФИО (324885) необходимо сменить подписанта по заявке на командиров
21	Вопрос/проблема: пропишите пож-та схему как внести в УВХД -исполнение серийный номер по основ
22	Добрый день. В заявке, отправленной сотрудником, не указывается наименование. Не могу ввести командиро

Рис. 2. Пример данных из датасета

Для дообучения модели `ner_ontonotes_bert_mult` необходима обработка данного датасета. Для этого необходимо токенизировать предложения и разметить получившиеся токены `bio`-разметкой: где «В» обозначает начало сущности, «I» означает «внутри» сущности (все остальные токены, относящиеся к сущности после начального), а «О» – незначимая сущность. Были выбраны следующие теги для разметки датасета:

Ter	Описание
B-ORG, I-ORG	Название организации
B-MONEY, I-MONEY	Денежные суммы
B-CONTRACT	Номер договора
B-ADR, I-ADR	Адрес
B-DATE	Дата
B-PHONE	Номер телефона
O	Незначимые теги

Рис. 3. Выбранные теги для разметки датасета

Для предварительной разметки можно воспользоваться библиотекой `srasu` и моделью «`ru_core_news_sm`» для выделения некоторых именованных сущностей. В итоговом варианте датасет преобразован к следующему виду: к каждому токену прилагается номер предложения, в котором он состоит, а также его тег. Пример размеченного предложения из датасета

Column1	Column2	Column3
5	В	O
5	ЗНВ	O
5	30000196281	B-CONTRACT
5	в	O
5	графе	O
5	"	O
5	Сумма	O
5	комиссии	O
5	"	O
5	необходимо	O
5	откорректировать	O
5	сумму	O
5	вознаграждения	O
5	АО	B-ORG
5	"	I-ORG
5	РАД	I-ORG
5	"	I-ORG
5	за	O
5	продажу	O
5	объекта	O
5	на	O
5	80359.84	B-MONEY
5	руб.	I-MONEY

Рис. 4. Пример разметки предложения

Получившийся датасет планируется использовать для дообучения модели, что позволит использовать ее для определения выбранных именованных сущностей в дальнейших сообщениях.

Заключение

Данная работа посвящена проведению сравнения существующих методов извлечения именованных сущностей, а также сбору и анализу датасета для дальнейшего дообучения модели на предметную область.

Были описаны применяемые известный подходы к задаче выделения именованных сущностей, а также в итоге данной работы был получен датасет, который можно использовать для дообучения модели `ner_ontonotes_bert_mult`, что и планируется сделать в дальнейшей работе.

Список литературы

1. Using Machine Learning to Maintain Rule-based Named-Entity Recognition and Classification Systems / G. Petasis [et al.] // Proceedings of

the 39th Annual Meeting of the Association for Computational Linguistics. – 2001. – С. 426-433.

2. Березин, С. А. Выделение именованных сущностей из текстов распорядительных документов с помощью глубоких нейронных сетей / С. А. Березин, И. Ю. Бондаренко // Системная информатика. – 2020. – №16. – С. 137-148.

3. Томита-парсер. Руководство разработчика. [Электронный ресурс]. – Режим доступа : <https://tech.yandex.ru/tomita/doc/dg/concept/about-docpage/>

4. Сравниваем работу open source Python – библиотек для распознавания именованных сущностей. [Электронный ресурс] – Режим доступа: <https://habr.com/ru/articles/502366/>

5. Yargy-парсер и библиотека Natasha. Извлечения структурированной информации из текстов на русском языке. [Электронный ресурс] – Режим доступа: <https://habr.com/ru/articles/349864/>

Разработка сайта для купли-продажи авто

А. А. Полев

Студент бакалавр

А. А. Вахтин

Доцент

Введение

Быстрый рост интернета и цифровых технологий значительно повлиял на различные аспекты человеческой жизни, преобразив способы общения, работы и ведения бизнеса. Сфера, которая претерпела значительные изменения – автомобильная промышленность, особенно в части торговли автомобилями. В последние годы спрос на онлайн-площадки, облегчающие покупку и продажу автомобилей, резко вырос, так как они предоставляют удобство, более широкий выбор и эффективность во времени. В этой статье представлена разработка площадки для торговли автомобилями.

1. Постановка задачи

Целью данной статьи является создание сайта для купли-продажи автомобилей, который позволит пользователям удобно искать и приобретать транспортные средства, а также размещать объявления о продаже своих автомобилей. Сайт будет предоставлять возможность фильтрации объявлений по различным параметрам, таким как марка, модель, год выпуска, цена и другие, а также будет выставлять оценку качества каждому объявлению. Одной из особенностей сайта будет реализация алгоритма оценки автомобилей по их параметрам. Этот алгоритм будет учитывать такие факторы, как год выпуска, пробег, цена, количество владельцев и др.

2. Алгоритм оценки качества автомобиля

Данный алгоритм предназначен для оценки качества подержанных автомобилей на основе нескольких ключевых параметров. Он позволяет получить оценку, которая может помочь потенциальным покупателям принять обоснованное решение при выборе автомобиля на вторичном рынке.

Для оценки качества автомобиля будут использованы следующие параметры с соответствующими весовыми коэффициентами:

- пробег – w_1 ;
- цена – w_2 ;
- количество владельцев – w_3 ;
- год выпуска – w_4 ;
- тип коробки передач (КПП) – w_5 ;
- наличие ДТП – w_6 .

Для подбора оптимальных весовых коэффициентов нужно учитывать важность каждого параметра и как эти параметры влияют на общую оценку качества автомобиля. Далее представлены веса, основанные на общепринятых представлениях о значимости каждого аспекта автомобиля.

Все параметры должны быть преобразованы в единый диапазон от 0 до 1 для корректного сравнения и суммирования их влияния на итоговую оценку.

Итоговые коэффициенты:

- для пробега – 0.15;
- для цены – 0.2;
- для года выпуска – 0.2;
- для типа КПП – 0.1;
- для наличия ДТП – 0.25.

Расчёт итоговой оценки качества автомобиля (Q) рассчитывается по формуле:

$$Q = (n_1 w_1) + (n_2 w_2) + (n_3 w_3) + (n_4 w_4) + (n_5 w_5) + (n_6 w_6) * 100, \quad (1)$$

где n – нормализованное значение i -го параметра, w – весовой коэффициент i -го параметра.

Нормализация означает преобразование значений параметров к единому диапазону, в нашем случае от 0 до 1. Это позволяет сравнивать и комбинировать значения разных параметров, которые первоначально могут измеряться в разных единицах или иметь разные диапазоны.

Нормализованные значения для параметров пробега, стоимости, количества владельцев и возраста автомобиля будут вычисляться по формуле 2.

$$n_i = 1 - a / b, \quad (2)$$

где a – характеристики автомобиля и b – эталонные значения.

Нормализованное значение для КПП будет задано статически, а именно 1 для автоматической, 0.9 для механической и 0.8 для вариаторной.

Нормализованное значение для наличия ДТП у автомобиля будет тоже задано статически, а именно 1 для автомобилей без ДТП в истории, 0.6 для автомобилей с небольшими ДТП и 0 для автомобилей с сильными ДТП.

Пример расчёта оценки для автомобиля с пробегом 92000 км, ценой 750000 рублей, тремя предыдущими владельцами, годом выпуска 2009, автоматической коробкой передач и легким ДТП в истории, расчет выглядит следующим образом:

- пробег = 0.693;
- цена = 0.85;
- количество владельцев = 0.25;
- год выпуска = 0.35;
- тип коробки передач (КПП) = 1;
- наличие ДТП = 0.6.

Тогда, при таких коэффициентах, итоговая оценка автомобиля будет равняться 67 баллам.

3. Создание сайта

Для создания сайта были использованы технологии TypeScript, React, Java, Spring, PostgreSQL, Liquibase[1-5].

Ниже, на рис. 1, представлена ER диаграмма, иллюстрирующая архитектуру базы данных. Диаграмма состоит из восьми сущностей. Таблица объявлений – listing содержит в себе информацию об объявлениях. Сущность автомобиля – car содержит информацию об автомобилях. Сущность пользователя – users хранит необходимые данные о пользователях. Также есть и вспомогательные сущности, такие как марка, модель, тип топлива, тип привода и другие. Они содержат справочную информацию о возможных характеристиках автомобиля. Каждая из этих сущностей имеет идентификатор и название соответствующей характеристики. Таблица модели – model связана с маркой – mark и содержит информацию об идентификаторе модели – id, идентификаторе марки – mark_id и названии модели – model_name.

Связь избранных объявлений – user_favorite представляет собой связующую сущность между пользователем – users и объявлением – listing, позволяющую пользователям сохранять избранные объявления.

Связь "user_favorite" представляет собой связующую сущность между "users" и "listing", позволяющую пользователям сохранять избранные объявления.

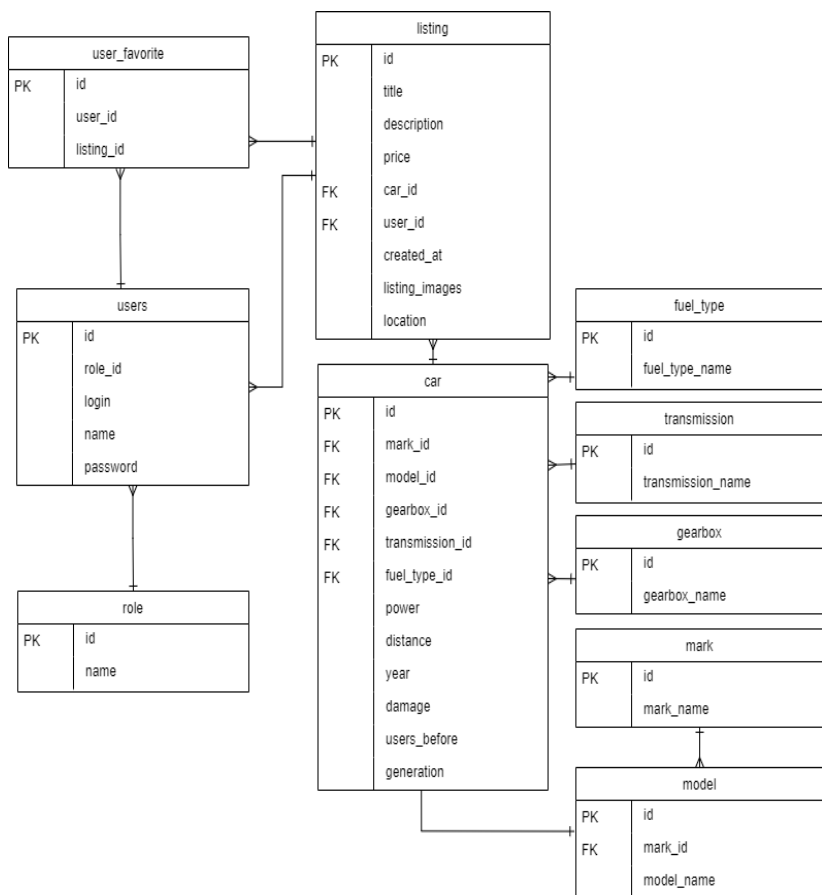


Рис. 1. ER диаграмма

Ниже, на рис. 2, изображена главная страница со списком объявлений и оценками за них. Если пользователю требуется найти какой-то конкретный автомобиль или автомобиль с конкретными характеристиками, то он может воспользоваться панелью с поиска в верхней части страницы. На рис. 3 демонстрируется пример результатов фильтрации по критерию марки автомобиля. В результате поиска остаются объявления, которые удовлетворяют критериям поиска и также на странице появляется счётчик с количеством найденных объявлений.

Выберите марку	Выберите модель	Выберите привод
Цена от	Цена до	Год от
Пробег от	Пробег до	Выберите тип КПП
		Найти  

	Lexus Характеристики: Пробег: 25000 Год выпуска: 2019 Владельцев: 1 Привод: FWD Трансмиссия: АТ Без ДТП	3500000 Р Оценка: 82	☆	Ростов
	Mitsubishi Lancer Характеристики: Пробег: 92000 Год выпуска: 2009 Владельцев: 3 Привод: FWD Трансмиссия: АТ Легкие ДТП	750000 Р Оценка: 67	☆	Москва
	Toyota Nashiroku Характеристики: Пробег: 320000 Год выпуска: 1992 Владельцев: 5 Привод: RWD Трансмиссия: МТ Сильный ПТП	400000 Р Оценка: 22	☆	Владивосток

Рис. 2. Страница с объявлениями

Toyota	Выберите модель	Выберите привод
Цена от	Цена до	Год от
Пробег от	Пробег до	Выберите тип КПП
		Найти  

Найдено объявлений: 1

	TOYOTA NASHIROKU Характеристики: Пробег: 320000 Год выпуска: 1992 Владельцев: 5 Привод: AWD Трансмиссия: МТ С ДТП	400000 Р Оценка: 22	☆	Владивосток

Рис. 3. Фильтрация объявлений по марке

Далее, на рис. 4, изображена страница создания объявления.

Марка	Модель
NISSAN	Модель вашего авто
Цена	INTEGRA
Цена	HACHIROKU
Коробка передач	CIVIC
Автоматическая	GTR
Привод	LANCER
Задний привод	RS200
Количество владельцев	CHARGER
Число владельцев:	FD30
	VESTA
	XSM
Мощность л.с.	Топливо
0	Бензин
Год выпуска	Пробег
Год выпуска вашего авто	(км)
Местоположение	
Город, улица	
Номер телефона	
+7(111)222-33-44	
Заголовок объявления	
Напишите заголовок для объявления	
Комментарий к объявлению	
Напишите подробнее о вашем автомобиле, добавьте информацию о состоянии, его особенности и преимущества	
Загрузите фотографии	
Выберите файл	Файл не выбран
← Создать объявление	

Рис. 4. Страница с формой создания объявления

На рис. 5 представлено окно профиля.

Окно профиля

Логин: admin

Имя: admin

Мои объявления

Администрирование

Выйти из профиля

Отмена

Сохранить изменения

Рис. 5. Страница с окном профиля

Заключение

В результате был создан сайт для купли-продажи авто. Особое внимание уделено алгоритму оценки качества автомобилей, который учитывает несколько ключевых параметров и помогает потенциальным покупателям принять обоснованное решение при выборе автомобиля на вторичном рынке. Реализация проекта с использованием современных технологий обеспечивает эффективность и удобство для пользователей.

Список литературы

1. TypeScript is JavaScript with syntax for types [Электронный ресурс] : сайт – Режим доступа : <https://www.typescriptlang.org/>
2. SpringOne at VMware Explore [Электронный ресурс] : сайт. – Режим доступа : <https://spring.io/>
3. Consta Дизайн-система для быстрой разработки интерфейсов [Электронный ресурс] : сайт – Режим доступа : <https://consta.design/>
4. React JavaScript-библиотека для создания пользовательских интерфейсов [Электронный ресурс] : сайт. – Режим доступа : <https://ru.legacy.reactjs.org/>
5. Run JavaScript Everywhere [Электронный ресурс] : сайт. – Режим доступа : <https://nodejs.org/en>

Детекция границ объектов на изображениях

А.А. Положенцев

Студент бакалавр

Н.К. Самойлов

Старший преподаватель

Введение

В настоящее время довольно остро стоит вопрос автоматизации рутинных процессов, т.к. постоянно повторяющиеся операции, выполняемые человеком, приводят к появлению необратимых ошибок. Объем рынка CV в 2021 г., по данным консалтинговой компании Tractica, увеличился на 32,9% и в 2022 г. составила \$48–50 млрд [1]. Существует два подхода для реализации компьютерного зрения: применение алгоритмов без анализа предыдущих результатов или применение нейросетей. Однако существует проблема скорости работы последних, что доказывают исследования 2021 года (табл. 1). Анализ данных позволяет сделать вывод, Cannydetector в 3.17 раза быстрее обрабатывает 1 линию дорожной разметки в сравнении с FasterRCNN, лучшей реализацией RCNN-подобных нейросетей. В некоторых ситуациях скорость детекции важнее незначительного снижения качества распознавания.

Таблица 1

Сравнение скорости детекции полос дорожного движения

** – рассматривается сопоставимая функциональность подходов*

*** – согласно исследованиям TexasInstruments, 2021*

Характеристики / название алгоритма	FilteredCann ydetector (OpenCV)	TensorFlow (FasterRCNN*)
Скорость распознавания (мс)	6	19
Скорость обработки видеоряда (320 × 160) (FPS, кадров/с)	25	19
Предельная скорость обработки видеоряда (1920 × 1080) (FPS, кадров/с) **	44	28

1. CannyDetector

Детектор границ Кэнни – алгоритм, который путём снижения количества шумов и подавления «слабых» границ, выделяет границы объектов толщиной в 1 пиксель. Является одним из самых эффективных детекторов среди существующих и, в отличие от детектора Харриса, не привязан к наличию углов у объекта.

Основные этапы работы детектора границ Кэнни:

- перевод изображения в серые оттенки, так как детекторы не работают с цветом;
- гауссовское размытие;
- поиск градиентов;
- подавление немаксимумов;
- двойная пороговая фильтрация. [2].

2. Реализованные задачи

Целью данной работы является реализация детектора Кэнни на языке Kotlin, ввиду дополнительных конвертаций изображения для работы с OpenCV мы можем сравняться по скорости, не теряя в качестве детекции границ объектов с эталонной реализацией.

Для успешного достижения цель разделена на задачи.

Во-первых, изучен каждый этап обработки изображения, а именно перевод фотографии в серые тона, применение размытия по Гауссу (“gaussianblur”), применение ядра Собеля, с последующим применением алгоритма Кэнни. На данном этапе найдём наиболее подходящее решение, с учётом требуемой высокой скорости и качества обработки изображения.

Во-вторых, реализован детектор Кэнни, с учётом ранее выбранных математических инструментов, на языке программирования Kotlin.

В-третьих, подключена библиотека OpenCV и проверена скорость обработки тестовых изображений и оценена качество детекции нашей реализации и эталонной.

3. Анализ предметной области для реализации Cannydetector

1. Перевод изображения в серые оттенки.

Изначально требуется перевести цветное входное изображение в оттенки серого. Произведём данное действие с помощью усреднение значений RGB:

$$Y' = (R + G + B) / 3, \quad (2)$$

где Y' - измененный пиксель изображения, R - градация красного цвета пикселя Y , G - градация зелёного цвета, B - градация синего цвета.

2. Сглаживание.

Размытие по Гауссу – фильтр, ядро которого вычисляется с использованием функции распределения Гаусса.

Существует две вариации функций для ядра применяемые для размытия изображения. Первый вариант предполагает использование двумерной функции для построения матрицы и однократное прохождение по изображению, смещая данную матрицу, а второй – использование одномерной функции (2) и прохождение сначала по горизонтали, а потом по вертикали.

$$G(x) = \frac{1}{\sqrt{2\pi}\delta^2} e^{\frac{-x^2}{2\delta^2}}, \quad (3)$$

где δ – стандартное отклонение нормального распределения, x – смещение от центра ядра свёртки.

Полученный массив используют дважды: сначала проходят вектором-строкой по горизонтали, потом по вертикали в виде вектора-столбца. В итоге получается то же самое изображение, что и после применения двумерного размытия Гаусса, однако временная сложность алгоритма вместо квадратичной $O(n^2)$ становится линейной $O(2n)$ [3].

3. Предварительная детекция границ. Поиск градиентов.

Для обнаружения границ используются аналоги производных первого и второго порядка. Вычисление первой производной цифрового изображения основано на различных дискретных приближениях двумерного градиента (3).

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}. \quad (4)$$

Важную роль при обнаружении контуров играют такие характеристики, как модуль этого вектора: I и направление вектора градиента: $W \times H$.

В нашей работе будет применено ядро Собеля (4), так как данный оператор чаще остальных применяется для анализа изображений. [4].

$$I^{(p)} \subset I \ w \times h. \quad (5)$$

4. Детектор границ Кэнни

Детектор Кэнни включает в себя все вышесказанные этапы, а также подавление немаксимумов и двойную пороговую фильтрацию, принцип работы подавления представлен на рис.1.

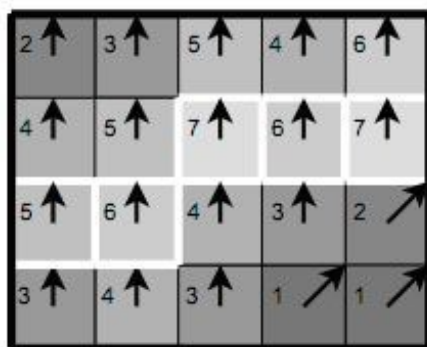


Рис. 1. Принцип подавления немаксимумов

Анализируя рисунок, можно сказать, что многие пиксели имеют ориентированные вверх градиенты, которые и считаются потенциальными границами объекта. Путём сравнения значений градиента в данных пикселях выделяем окончательные границы с локальными максимумами среди соседей (на рисунке обведены белым контуром), остальные – подавляются.

Далее применяется двойная пороговая фильтрация. В зависимости от поставленной задачи необходимо определить оптимальные значения фильтров, так как при снижении порогов число границ будет выше, однако такая детекция сильнее подвержена шуму, что снизит эффективность обнаружения, и, наоборот, при увеличении мы будем игнорировать «слабые» границы, но увеличим устойчивость к внешним помехам.

Итогом анализа предметной области является нахождение математических алгоритмов, отвечающие высоким требованиям по скорости работы, исполнения детектора Кэнни.

4. Анализ средств реализации

Рассмотрим архитектуру приложения подробнее (рис. 2).

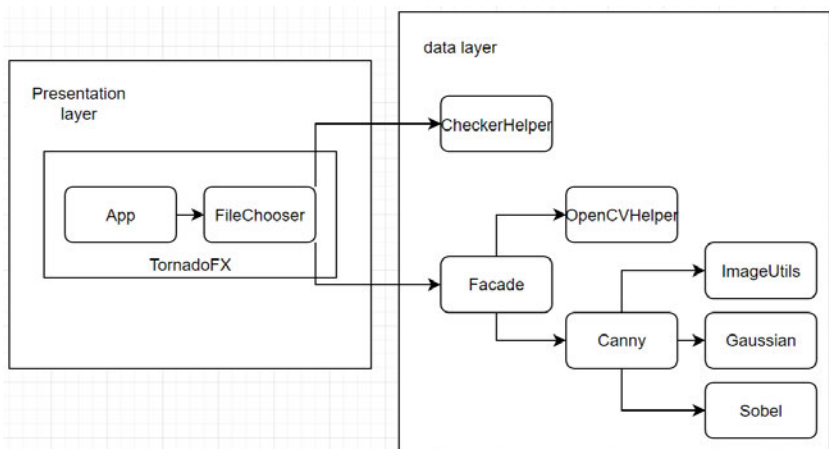


Рис. 2. Архитектура приложения

Приложение имеет многослойную архитектуру:

- слой представления (presentationlayer);
- слой бизнес-логики (datalayer).

Слой представления представлен двумя классами и отвечает за взаимодействие пользователя со слоем бизнес-логики.

Слой бизнес-логики состоит из семи классов, отвечающих принципу *single responsibility* – одному из основополагающих частей объектно-ориентированного подхода в программировании, путём реализации шаблона проектирования *facade*.

Рассмотрим особенности библиотеки **OpenCV**. Библиотека имеет несколько ограничений в работе.

1. Продукт не работает напрямую с изображением, а двумерное представление конвертируется в трёхмерное. После обработки требуется конвертация обратно в изображение.

2. Библиотека не поддерживает работу с кириллицей, что ограничивает её применение в русскоговорящих странах.

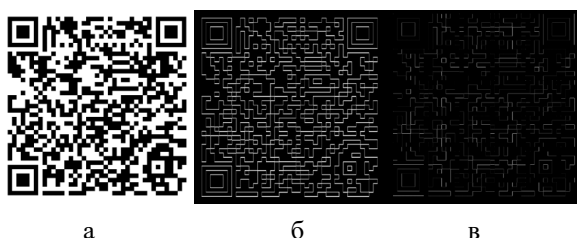
Для подключения **OpenCV** необходимо добавить соответствующую зависимость в код, а также написать 2 адаптера-конвертера (ввиду пункта 1) и вызвать соответствующую функцию преобразования изображения.

5. Сравнение производительности и качества обнаружения границ эталонной реализации детектора Кэнни OpenCV и написанной реализации

В рамках достижения поставленной цели проведено тестирование. Были выбраны девять изображений QR-кодов [5] и проведена оценка скорости работы и субъективная оценка качества.

Рассмотрим изображение с самым низким отставанием между реализациями и проведём анализ качества детекции (рис.3).

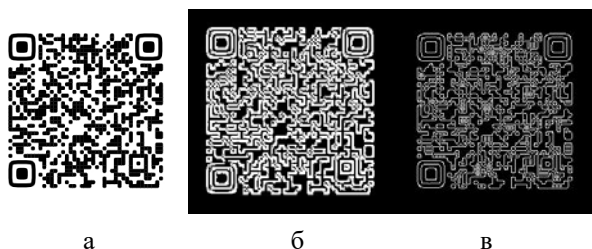
По результатам проведённого тестирования можно сказать, что скорость обработки изображения алгоритмом, реализованным на Kotlin, отстаёт от эталонной на 28,3%, однако качество обработки двух реализаций сопоставимо.



*а – исходное изображение, б – детекция Kotlin Canny detector (829 мс),
в – детекция OpenCV Canny detector (646 мс)*

Рис. 3. Детекция линий QR-кода (лучшее соотношение скорости Kotlin Canny detector относительно OpenCV Canny detector)

Рассмотрим наихудший случай по разнице в скорости между реализациями (рис.4).



*а – исходное изображение, б – детекция Kotlin Canny detector (51 мс),
в – детекция OpenCV Canny detector (28 мс)*

Рис. 4. Детекция линий QR-кода (худшее соотношение скорости Kotlin Canny detector относительно OpenCV Canny detector)

Скорость обработки нашей реализацией ниже на 82,1% по сравнению с эталонной.

Остальные показатели скорости приведены в табл. 2.

Таблица 2

Результаты тестирования реализаций Cannydetector (детекция QR-кодов)

№ Изображения	Canny detector (Kotlin) Время обработки (мс)	Canny detector (OpenCV) Время обработки (мс)	Отставание от реализации OpenCV	Разрешение
1	1258	869	+44,76%	1960×1960 $w \times W, h \times H$
2	1065	773	+37,77%	1960×1960 $I^{(1)}, I^{(2)}, I^{(3)}, I^{(4)} \subset I$
3	884	644	+31,06%	1960×1960 $I^{(p)}$
4	962	679	+41,69%	1960×1960 $I^{(p)} \cup I^{(1)} \cup I^{(2)} \cup I^{(3)} \cup I^{(4)} = I$
5	862	655	+31,6%	1960×1960 F
6	842	639	+31,77%	1960×1960 $\{I^{(i)}\}$
7	911	667	+36,58%	1960×1960 $i = \overline{1, 4}$
8	829	646	+28,33%	1980×1980 $I^{(p)}$
9	51	28	+82,14%	281×281 $I^{-(p)} = F(I^{(p)}, \{J^{(i)}\}, K)$ $J^{(i)} = G(I^{(i)}), i = \overline{1, 4}$

Согласно данным, исключая из расчёта наилучший (28,3%) и наихудший (82,1%) случай, среднее отставание по скорости обработки Kotlin Cannydetector от эталонной реализации составляет 37,4%.

Заключение

Подводя итоги, можно отметить качество детекции очень сильно зависит от входных данных, так QR-коды обрабатываются Kotlin реализацией лишь незначительно хуже, чем эталонной. Однако при обработке более сложных изображений с множеством мелких деталей растёт число aberrаций, снижается скорость обработки в ущерб качеству. С другой стороны, даже эталонная реализация не лишена проблем с неточностью детекции [5], но по качеству обработки заметно стабильнее нашей реализации. Скорость обработки является важным параметром, но если число aberrаций значительно, то высокая скорость перестаёт быть преимуществом программы в сравнении с аналогами.

Сформированы перспективные направления дальнейшего развития проекта:

- проверить оптимальность выбора оператора Собеля (3 G 3) с возможной заменой на аналогичное ядро размером 5 K 5 или на фильтр Шарра;
- получение параметров границ объектов в удобном виде для дальнейших манипуляций над объектами путём внедрения imagefiltering, morphologicaltransformations.

Список литературы

1. ИКС медиа: официальный сайт [Электронный ресурс]: статья. – Режим доступа: <https://www.iksmedia.ru/articles/5894791-Tendencii-razvitiya-kompyuternogo.html>
2. Волохов, В.В. Алгоритм автоматического сбора изображения типа «пазл» из фрагментов [Электронный ресурс]: статья. – Режим доступа: <https://nauchkor.ru/pubs/algoritm-avtomaticheskogo-sbora-izobrazheniya-tipa-pazl-iz-fragmentov-587d363f5f1be77c40d58a79>
3. Дипломная работа на тему «Оптическое распознавание символов» [Электронный ресурс]: статья. – Режим доступа: <https://bank.nauchniestati.ru/primery/diplomnaya-rabota-na-temu-opticheskoe-raspoznavanie-simvolov-imwp/>
4. Детектор границ Канни [Электронный ресурс]: статья. – Режим доступа: <https://habr.com/ru/articles/114589/>
5. Cannydetector [Электронный ресурс]: документация. – Режим доступа: <https://drive.google.com/drive/folders/107ZqsktJ5tHgngVWT64ZlgOoqq1N1SIo?usp=sharing>

Моделирование межатомного потенциала по результатам квантовой-химического расчёта с использованием симметричных функций Бехлера-Парринелло

А. Г. Потапов

Студент бакалавр

А. В. Романов

Старший преподаватель

Введение

Внедрение нейронных сетей в расчёты методом молекулярной динамики долгое время сдерживалось отсутствием адекватных способов описания геометрии многоатомных систем. Традиционные подходы с использованием декартовых или сферических координат вызывали сложности с масштабированием вследствие привязки энергии системы к сетке координат.

Для того чтобы преодолеть это ограничение, предложено несколько вариантов, таких как, например, функции Бехлера-Парринелло [1-2] или полиномы Чебышева [3-4]. В этом случае атомное окружение описывается набором простых функций, учитывающих межатомное расстояние и валентные углы. Группируя несколько функций, появляется возможность описания атомного окружения, по аналогии с базисными множествами в квантовой механике, однако выбор их типа и количества для каждой конкретной химической системы остаётся вопросом открытым.

В последние годы был сделан ряд успешных попыток применить данный подход к решению задач молекулярной динамики [5-6]. В данной статье будут рассмотрены особенности применения симметричных функций Бехлера-Парринелло для модельных систем с использованием потенциала Леннарда-Джонса [7] и результатов квантово-химического моделирования теорией функционала электронной плотности.

В первой части статьи будет дано описание симметричных функций Бехлера-Парринелло. Далее использование этих функций будет рассмотрено на примере димера и тримера с использованием межатомного потенциала Леннарда-Джонса. В третьей части статьи будет рассмотрена возможность использования функций Бехлера-Парринелло для обучения межатомного потенциала по результатам квантово-химического моделирования. В заключительной части будет

сделаны выводы касательно нюансов выбранного подхода в применении к реальным моделям.

1. Функции Бехлера-Парринелло

Главной особенностью функций Бехлера-Парринелло является то, что, в отличие от декартовых координат, они не имеют привязки к координатной сетке, а учитывают только атомное окружение, исходя из расстояний и валентных углов между атомами. Вследствие этого данные функции не подвержены проблемам декартовых координат, связанных с трансляционной и вращательной симметрией системы, что позволяет беспрепятственно масштабировать модель.

В первоначальном варианте было предложено пять видов функций Бехлера-Парринелло. Каждая из пяти функций описывает систему относительно конкретного заданного i -го атома в объёме сферы радиусом R_c , называемого радиусом отсечки. При этом функции G^1 , G^2 , G^3 описывают только изменение радиальной составляющей, в то время как G^4 и G^5 учитывают и угловую часть.

$$f_c(R_{ij}) = \begin{cases} 0,5 \cdot \left[\cos\left(\frac{\pi R_{ij}}{R_c}\right) + 1 \right] & \text{для } R_{ij} \leq R_c \\ 0 & \text{для } R_{ij} > R_c \end{cases} \quad (1)$$

$$G_i^1 = \sum_j f_c(R_{ij}) \quad (2)$$

$$G_i^2 = \sum_j e^{-\eta(R_{ij}-R_s)^2} \cdot f_c(R_{ij}) \quad (3)$$

$$G_i^3 = \sum_j \cos(\kappa R_{ij}) \cdot f_c(R_{ij}) \quad (4)$$

$$G_i^4 = 2^{1-\zeta} \sum_{j,k \neq i}^{all} (1 + \lambda \cos \theta_{ijk})^\zeta e^{-\eta(R_{ij}^2 + R_{ik}^2 + R_{jk}^2)} f_c(R_{ij}) f_c(R_{ik}) f_c(R_{jk}) \quad (5)$$

$$G_i^5 = 2^{1-\zeta} \sum_{j,k \neq i}^{all} (1 + \lambda \cos \theta_{ijk})^\zeta \cdot e^{-\eta(R_{ij}^2 + R_{ik}^2)} \cdot f_c(R_{ij}) \cdot f_c(R_{ik}) \quad (6)$$

Параметр η определяет ширину гауссиан и помогает регулировать эффективное радиальное расширение функций симметрии; R_s отвечает за сдвиг центра гауссиан на определённый радиус. Параметр κ настраивает период функции косинуса. Параметр θ описывает угол между тремя атомами. Параметр λ может принимать только значения ± 1 и влияет лишь на ориентацию графика $G(\theta)$. При $\lambda = +1$ значение G

обнуляется при $\theta = 180^\circ$, а иначе – при $\theta = 0^\circ$. Параметр ζ регулирует угловое разрешение, то есть плавность изменения значения функции G .

2. Моделирование двух- и трёхатомных систем с использованием потенциала Леннарда-Джонса

Чтобы продемонстрировать возможность описания геометрии многоатомных систем с использованием функций Бехлера-Парринелло, удобно рассмотреть простейший случай двух- и трёхатомных молекул. В качестве межатомного потенциала подойдет потенциал Леннарда-Джонса, как простейший и один из широко распространённых межатомных потенциалов. Форма потенциала Леннарда-Джонса, использованного в данном эксперименте, представлена ниже.

$$U(r) = 4\epsilon \left[\left(\frac{\sigma}{r} \right)^{12} - \left(\frac{\sigma}{r} \right)^6 \right] \quad (7)$$

Здесь r – расстояние между центрами частиц, ϵ – глубина потенциальной ямы, σ – расстояние, на котором энергия взаимодействия равна нулю.

На рис. 1 представлены результаты воспроизведения потенциала Леннарда-Джонса.

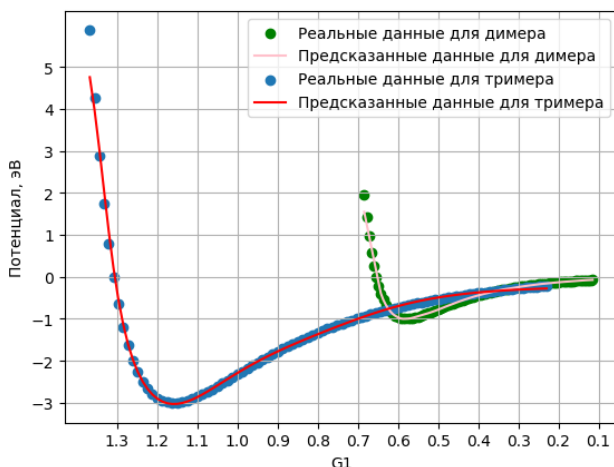


Рис. 1. Графики зависимости потенциала от функции G^1

Был рассмотрен простейший случай, когда изменение положения атомов происходило в плоскости и, в случае тримера, с одинаковыми углами. Для описания такой системы достаточно функции G^1 .

Далее с использованием Python-библиотеки Skikit-Learn был сконструирован многослойный перцептрон (MLP) с двумя скрытыми слоями и сигмоидальной функцией активации. Коэффициент обучения $\eta = 0.01$. Обученный на точных значениях потенциала Леннарда-Джонса, он должен уметь восстанавливать значение потенциала через значения G , не имея представления о декартовых координатах системы.

На рис. 2 представлено сравнение $U(r)$, полученного аналитически (точки) и с помощью MLP (линии). Как видно, графики почти полностью совпадают.

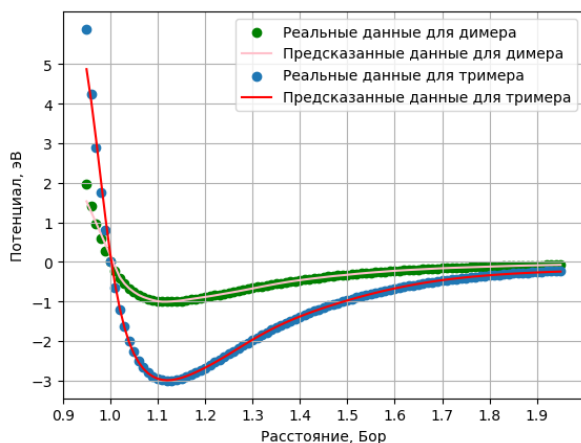


Рис. 2. Графики зависимости потенциала от расстояний $U(r)$

Теперь необходимо провести эксперимент, задействующий более сложные аспекты изменения системы и, как следствие, требующий более «продвинутых» симметричных функций для её описания.

Была рассмотрена следующая трёхатомная система (см. рис. 3)

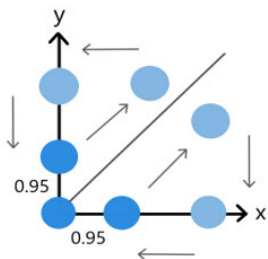


Рис. 3. Движение атомов с изменением расстояний и углов

В процессе перемещения траектории атомов не пересекаются. При этом два атома движутся симметрично относительно начала координат, что необходимо для уменьшения числа степеней свободы и наглядной визуализации результатов.

Для описания вышеприведённой системы была использована функция G^4 .

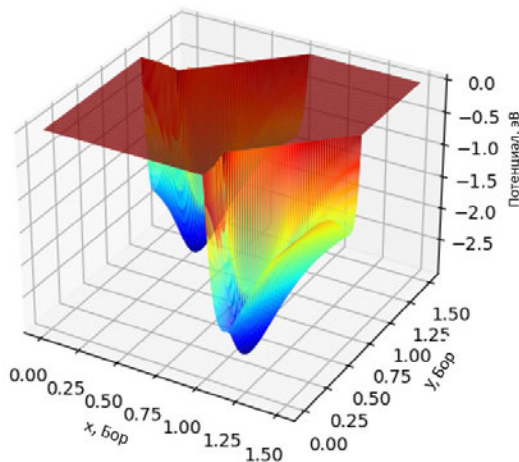


Рис. 4. Поверхность потенциальной энергии системы

В процессе расчёта был вычислен суммарный потенциал системы для каждой точки координатной сетки, а также сформирован двумерный массив значений функций G^4 . Это позволило наглядно отобразить поверхность потенциальной энергии системы (см. рис. 4).

На данном рисунке хорошо видны две потенциальные ямы, закономерно появившиеся на расстоянии $\sqrt{2}$ Бор от начала координат, что вполне согласуется с общим видом потенциала Леннарда-Джонса.

Кроме того, можно наблюдать резкий «взлёт» потенциала по мере приближения 2-го и 3-его атомов друг к другу. В связи с этим возникла необходимость ввести ограничение на предельный радиус сближения. В данном случае он равен 0,95 Бор, что и обуславливает наличие монотонной красной зоны значительных размеров. В данной области расчёты не проводились, а значение потенциала принято условно самым большим.

Наконец, анализ полученных данных нейросетью позволил сформировать поверхность потенциальной энергии системы на основе значений функции G^4 (см. рис. 5).

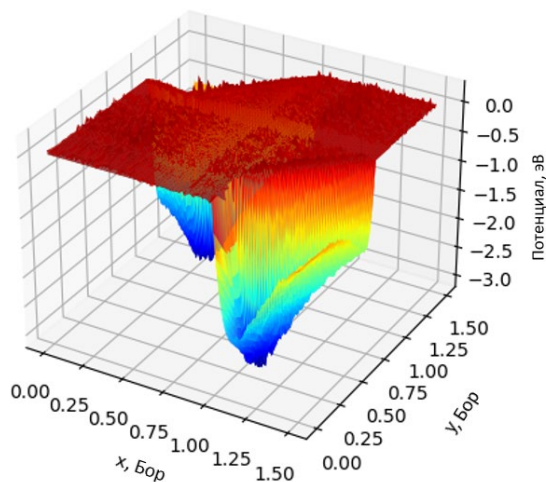


Рис. 5. Поверхность потенциальной энергии по MLP

Как видно, результат хорошо описывает исходную конфигурацию, что подтверждается значением среднеквадратичной ошибки $RMSE = 0,0497$ эВ при диапазоне изменения энергии $\Delta U = 2,5$ эВ. Тем не менее, на графике присутствуют возмущения, что свидетельствует о зависимости результата обучения нейросети от большого числа параметров, чем для рассмотренных ранее тривиальных случаев, и, как следствие, большей вероятности ошибки при их подборе.

3. Моделирование межатомного потенциала по результатам квантово-химического расчёта

Убедившись в работоспособности механизма конвертации декартовых координат в набор симметричных функций, можно приступить к моделированию на основе данных квантово-химического эксперимента. Набор функций G^1 хорошо показал себя для плоской системы атомов. Теперь требуется проверить его работоспособность в 3D-пространстве.

В рассмотрение был введён кристалл меди с ГЦК-структурой и ячейкой из четырёх атомов. Первоначальное расстояние между атомами равнялось $2,12 \text{ \AA}$, кристалл растягивался с шагом в $0,71 \text{ \AA}$ между атомами. Таким образом, были получены данные о характере межатомного потенциала для данной области межатомных расстояний.

Для каждого шага системы были рассчитаны значения функции G^1 относительно атома, находящегося в центре координат. Обучив уже

подготовленную ранее нейросеть на новых данных с функцией активации гиперболического тангенса и коэффициентом обучения $\eta = 0.04$, был построен график зависимости $U(G)$ (см. рис. 6).

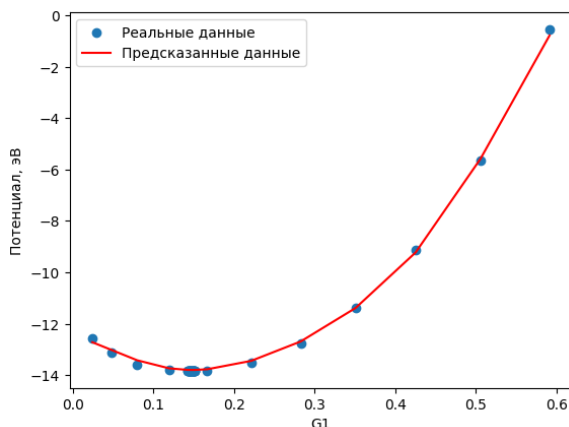


Рис. 6. Результат обучения MLP на реальных данных

В приведённом графике можно отметить две особенности. Во-первых, его общий вид зеркально отображён к графику $U(r)$, пример которого был приведён ранее. Это объясняется монотонно убывающим характером изменения функции G , что соответствует ожиданиям.

Во-вторых, на графике нет заметного «плато» энергии, к которому она должна стремиться по мере взаимного увеличения расстояний. Данная особенность является следствием того, что эта область отсутствует в данных квантово-химического эксперимента.

Главным же образом, результат моделирования демонстрирует возможность успешного применения функции G^1 для описания систем, изменяющих только радиальную составляющую и сохраняющих валентные углы между атомами. Для приближенного к реальности случая с изменением валентных углов, очевидно, недостаточно иметь только одну из функций Бехлера-Парринелло, что демонстрируют результаты расчета тримера с использованием функции G^4 .

Заключение

На основании проведённых экспериментов можно сделать следующие выводы. Во-первых, для описания простых систем с сохраняющимися углами достаточно учитывать только линейную часть. Для этого подходит функция G^1 .

Во-вторых, при тщательном подборе симметричных функций они с достаточной степенью точности описывают систему многих частиц, учитывая симметрию их взаимного расположения, что подтверждается экспериментами с нейросетью. При этом для описания более сложных конфигураций с изменяющейся угловой составляющей необходимо вводить функцию G^4 относительно зафиксированного атома.

В ходе дальнейших исследований планируется более детальное рассмотрение особенностей использования функций G^2 , G^3 , G^5 , а также комбинаций всех функций для реальных атомных структур.

Список литературы

1. Behler J., Constructing High-Dimensional Neural Network Potentials: A Tutorial Review / J. Behler // International Journal of Quantum Chemistry. – 2015. – Vol. 115. – Issue 16.
2. Behler J., Atom-centered symmetry functions for constructing high-dimensional neural network potentials / J. Behler // The Journal of Chemical Physics. – 2011. – Vol. 134. – Issue 3.
3. Lindsey, R. K., ChIMES: A force matched potential with explicit three-body interactions for molten carbon / R. K. Lindsey, L. E. Fried, N. Goldman // The Journal of Chemical Physics. – 2017. – Vol. 13. – Issue 12.
4. Lindsey, R. K., Active learning for robust, high-complexity reactive atomistic simulations / R. K. Lindsey, L. E. Fried, N. Goldman, S. Bastea // The Journal of Chemical Physics. – 2020. – Vol. 153. – Issue 13.
5. Kyuhyun L., SIMPLE-NN: An efficient package for training and executing neural-network interatomic potentials / L. Kyuhyun, Y. Dongsun, J. Wonseok, H. Seungwu // Computer Physics Communications. – 2019. – Vol. 242. – PP. 95-103.
6. Marx D., Ab initio Molecular Dynamics: Basic Theory and Advanced Methods / D. Marx, J. Hutter // Cambridge University Press. – 2009. – Vol. 63. – Issue 3.
7. Hod, S., Energy spectrum of the long-range Lennard-Jones potential / S. Hod // European Physical Journal Plus. – 2018. – Vol. 133. – Issue 11.

Разработка информационной системы управления контентом малого бизнеса на базе телеграм-бота

А. А. Пустовалов

Студент бакалавр

Е. А. Копытина

Старший преподаватель

Введение

В условиях прогресса в информационных технологиях и постоянно изменяющейся бизнес-среды для малых предприятий становится критически важным иметь эффективный и передовой подход к управлению контентом, как показано в [1-6]. Настоящая статья представляет инновационную разработку: систему управления контентом на базе телеграм-бота, направленную на оптимизацию процессов создания, редактирования и управления информацией. Такой подход отвечает вызовам современного бизнеса, где скорость принятия решений, и оперативная обработка данных играют решающую роль.

Современные технологии, особенно в области мессенджеров, активно внедряются в бизнес-процессы, предлагая новые возможности для автоматизации повседневных задач. Мессенджер «Телеграм» с его мощным API и разнообразными функциями становится предпочтительной платформой для разработки инновационных приложений, таких как боты, способных существенно улучшить управление информацией на малых предприятиях.

Цель данной работы заключается в предложении концепции и реализации программного комплекса под названием «ControlBot CMS», который не только обеспечивает возможности создания и редактирования контента, но и интегрируется в повседневные коммуникационные процессы бизнес-среды через мессенджер «Телеграм». Это предоставляет бизнесу гибкость и мобильность в управлении информацией, сокращая временные затраты на выполнение задач и повышая общую эффективность процессов.

1. Постановка задачи

Для системы управления контентом, описанной в статье и основанной на технологии телеграм-ботов, были сформулированы следующие ключевые требования:

- Разработка механизмов для создания и редактирования информации через телеграм-бот, обеспечивающих гибкость и оперативность процессов;
- Гарантированная совместимость с уже действующими информационными платформами для эффективного управления контентом;
- Создание механизмов интеграции телеграм-бота в ежедневные коммуникационные процессы бизнеса для уменьшения временных затрат на выполнение задач;
- Внедрение мер безопасности для защиты конфиденциальности информации, передаваемой и обрабатываемой в системе;
- Использование современных технологий, включая JavaScript, TypeScript, NestJS и Strapi для создания стабильной, масштабируемой и надежной системы;
- Реализация функциональных возможностей, направленных на повышение оперативности и эффективности управления информацией в условиях быстро меняющейся бизнес-среды.

2. Реализация логики

Проект реализуется посредством управления содержимым уже созданной системы, например, интернет-магазина через телеграм-бот.

Разработка логики системы началась с проектирования механизмов для создания и редактирования контента через телеграм-бот. С использованием технологий JavaScript и TypeScript были разработаны следующие функции, позволяющие администратору взаимодействовать с системой непосредственно из мессенджера:

- CreateContent,
- EditContent,
- DeleteContent.

Далее использовалось API, предоставляемое мессенджером «Telegram» и возможности конструктора API Strapi. Результатом стала разработка прослойки между функциями бота и API интернет-магазина – модуль «CmsIntegration», обеспечивающей эффективное управление контентом на различных платформах.

Для интеграции телеграм-бота в повседневные коммуникационные процессы бизнеса был создан дополнительный функциональный блок «TelegramBot», который сокращает временные затраты на выполнение

задач и обеспечивает более гладкое взаимодействие между ботом и бизнес-процессами.

Одновременно с этим применялись меры безопасности, такие как шифрование данных и авторизация, для защиты конфиденциальности информации.

С использованием современных технологий, таких как NestJS, Strapi и Telegraf, была разработана стабильная, масштабируемая и устойчивая система. TypeScript использовался для написания надежного и качественного кода, что упростило сопровождение системы.

3. Реализация интерфейса

Была спроектирована информационная панель бота внутри приложения «Телеграм». Вид информационной панели показан на рис.1.

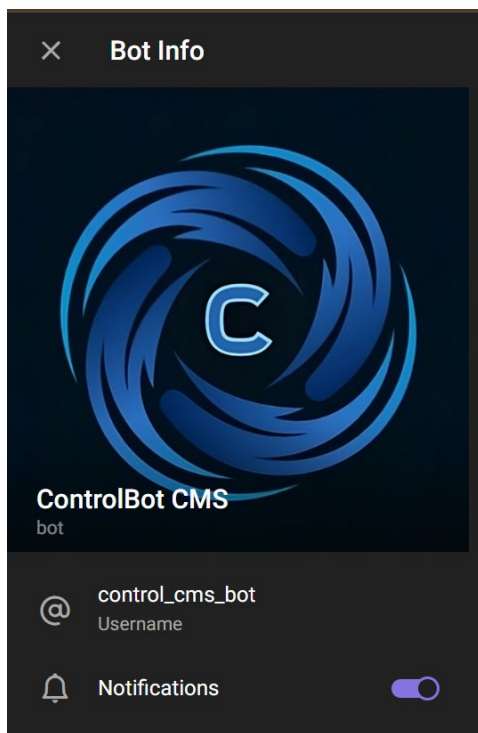


Рис. 1. Информационная панель бота

Работа в приложении начинается с запуска бота посредством команды /start, как проиллюстрировано на рис. 2.

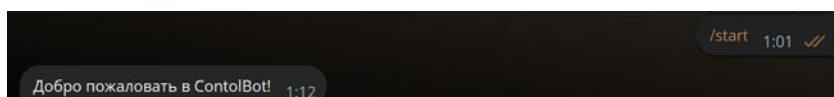


Рис. 2. Запуск чата с ботом

Далее происходит авторизация в вашу систему управления контентом (рис. 3).

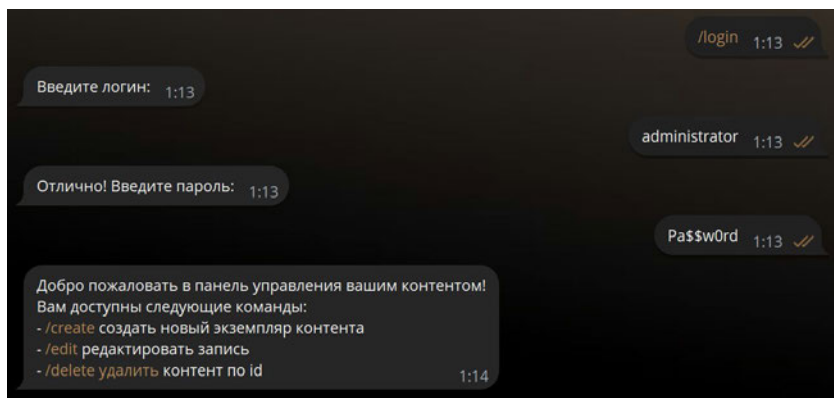


Рис. 3. Авторизация в CMS через бот

Создание контента происходит при помощи команды /create, как показано на рис. 4.

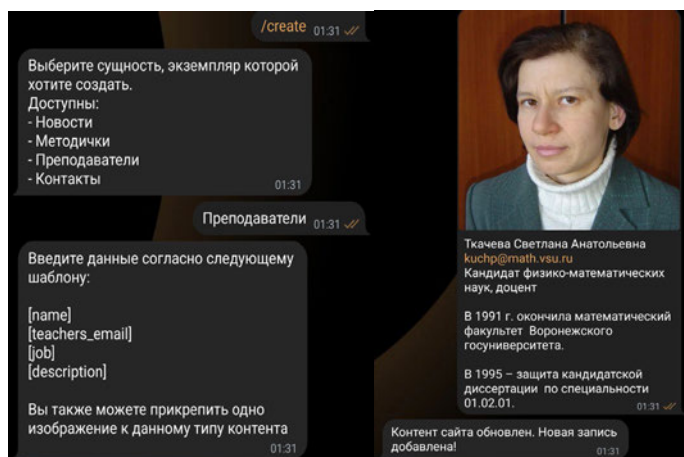


Рис. 4. Создание контента

Также возможно редактировать выбранную запись, данная возможность демонстрируется на рис. 5.

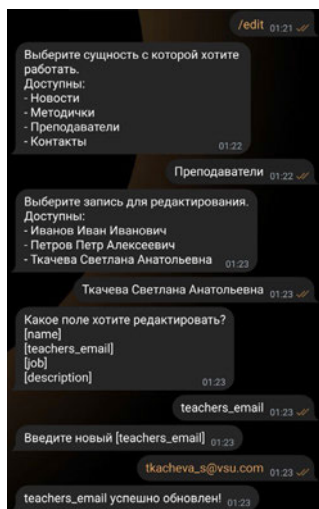


Рис. 5. Редактирование выбранной записи

Удаление записей тоже возможно и реализуется через текстовый интерфейс чат-бота, как показано на рис. 6.

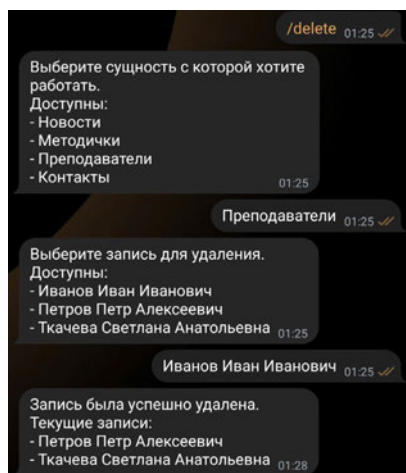


Рис. 6. Удаление выбранной записи

Заключение

В ходе данной статьи была создана система управления контентом, основанная на технологии телеграм-ботов, с целью оптимизации бизнес-процессов. Разработанные механизмы для создания и редактирования информации через мессенджер «Телеграм», включая интеграцию существующих платформ и обеспечение безопасности данных, успешно решают поставленные задачи.

Эта система не только обеспечивает эффективное управление информацией, но и приспособлена к динамичной бизнес-среде. Использование современных технологий, таких как JavaScript, TypeScript, NestJS, Strapi и Telegraf, придает ей стабильность и устойчивость.

Таким образом, все поставленные в начале разработки функциональные требования были выполнены. Система сокращает временные затраты на выполнение задач, повышает оперативность и общую эффективность управления информацией, является востребованной в сфере современного бизнеса и полностью работоспособной.

Список литературы

1. Манукян, С. Ш. Автоматизация взаимоотношений с клиентами в SEO- компании / С. Ш. Манукян, Е. А. Копытина // Труды молодых ученых факультета компьютерных наук ВГУ. Том Выпуск 2. – Воронеж : ООО «ВЭЛБОРН», 2022. – С. 108-113. – EDN PUNHZU.
2. Погорелов, Р. И. Адаптация тиражируемого решения от фирмы 1С (1С: Управление нашей фирмой) и разработка приложения на мобильной платформе 1С / Р. И. Погорелов, А. В. Копытин, Е. А. Копытина // Труды молодых учёных факультета компьютерных наук ВГУ : Сборник статей / Под редакцией Д.Н. Борисова. Том Выпуск 1. – Воронеж : Общество с ограниченной ответственностью "Вэлборн", 2021. – С. 121-125. – EDN EOZHFN.
3. Копытина, Е. А. Моделирование и реализация системы расчета заказов торговой сети / Е. А. Копытина, А. В. Копытин // Информатика: проблемы, методы, технологии : Материалы XXI Международной научно-методической конференции, Воронеж, 11–12 февраля 2021 года. – Воронеж: Общество с ограниченной ответственностью "Вэлборн", 2021. – С. 1240-1249. – EDN GUVONJ.
4. Новохатский, Н. М. Разработка мобильного приложения для медицинских клиник "Медангел" / Н. М. Новохатский, А. В. Копытин, Е. А. Копытина // Сборник студенческих научных работ факультета компьютерных наук ВГУ : Сборник статей. В 2-х частях / Под редакцией

Д.Н. Борисова. Том Выпуск 13. Часть 1. – Воронеж : Воронежский государственный университет, 2019. – С. 150-155.

5. Свидетельство о государственной регистрации программы для ЭВМ № 2023660153 Российская Федерация. Приложение для автоматизации деятельности персонала медицинских учреждений : № 2023618847 : заявл. 03.05.2023 : опубл. 18.05.2023 / А. А. Барышев, Е. А. Копытина ; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Воронежский государственный технический университет». – EDN IFYMGK.

6. Землянухин, Д. В. Разработка системы контроля и управления параметрами производственных процессов с GSM-каналом телеметрии / Д. В. Землянухин, Е. А. Копытина, А. А. Головкин // Сборник студенческих научных работ факультета компьютерных наук ВГУ : Сборник статей. В 2-х частях / Под редакцией Д.Н. Борисова. Том Выпуск 13. Часть 1. – Воронеж : Воронежский государственный университет, 2019. – С. 78-84. – EDN ACDOWK.

Численное интегрирование бигармонического уравнения в квадратной области

Е. М. Рогова

Студент бакалавр

М. И. Попов

Доцент

Введение

Бигармоническое уравнение возникает в механике сплошных сред при моделировании изгибов тонких жестких пластин под нагрузкой [1], а также при моделировании свободной конвекции в кондуктивно-ламинарном режиме [2]. Следует отметить, что в ряде исследований получены аналитические решения для некоторых постановок в виде рядов. Первое решение для свободно опертой прямоугольной пластины представил Навье в 1820 году в виде двойного тригонометрического ряда. Однако скорость сходимости полученного ряда и его производных оставляла желать лучшего. Решение для прямоугольной пластины, два противоположных края оперты, а два других имеют любое закрепление получено Морисом Леви. Оно представляет собой простой ряд с тригонометрическими и гиперболическим функциями. Скорость сходимости выше, чем в решении Нави, но не достаточная для точных вычислений. В дальнейшем получены решения в виде рядов и для других постановок. Плохая сходимость тригонометрических рядов заставляет учитывать большое количество членов ряда для приемлемой точности. Поскольку в выражения для коэффициентов ряда входят тригонометрические и гиперболические функции, в расчетах требуются сотни десятичных разрядов. Эти два обстоятельства существенным образом увеличивают время расчета. В связи с этим для решения подобных задач становится актуальным использование численных методов.

1. Постановка задачи

Рассмотрим тонкую жесткую пластину рис.1. Пластина закреплена по всему контуру и в каждой внутренней точке приложена сила [3]. Математическое описание прогиба $w(x, y)$ представляет собой

неоднородное бигармоническое уравнение с граничными условиями первого и второго рода в квадратной области $\Omega = [0;1] \times [0;1]$:

$$\frac{\partial^4 w}{\partial x^2} + 2 \frac{\partial^4 w}{\partial x^2 \partial y^2} + \frac{\partial^4 w}{\partial y^2} = \frac{q(x, y)}{D}. \quad (1)$$

$$w(0, y) = w(1, y) = w(x, 0) = w(x, 1) = 0, \quad (2)$$

$$\frac{\partial w(0, y)}{\partial x} = \frac{\partial w(1, y)}{\partial x} = \frac{\partial w(x, 0)}{\partial y} = \frac{\partial w(x, 1)}{\partial y} = 0 \quad (3)$$

Здесь $w(x, y)$ – прогиб пластины в точке (x, y) , $q(x, y)$ – поперечная нагрузка, D – цилиндрическая жесткость.

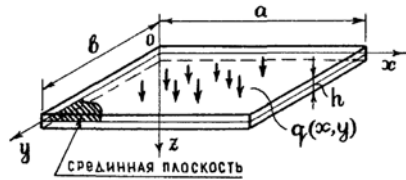


Рис. 1. Квадратная пластина пластина

2. Построение конечно-разностной схемы

Для построения конечно-разностной схемы заменим область Ω решения системы сеткой с шагами $\Delta x, \Delta y$, $\Omega_{\Delta x, \Delta y} = \{(x_i, y_j) = (i\Delta x, j\Delta y)\}, i = 0, \dots, n, j = 0, \dots, m$. Для построения конечно-разностного аналога частных производных $\partial^4 u / \partial x^4$ и $\partial^4 u / \partial y^4$ используем центрально-разностный оператор 2-го порядка примененный дважды по соответствующей переменной:

$$\left. \frac{\partial^4 w}{\partial x^4} \right|_{i,j} = \frac{w_{i+2,j} - w_{i+1,j} + 6w_{i,j} - 4w_{i-1,j} + w_{i-2,j}}{\Delta x^4} + O(\Delta x^2) \quad (4)$$

$$\left. \frac{\partial^4 w}{\partial y^4} \right|_{i,j} = \frac{w_{i,j+2} - w_{i,j+1} + 6w_{i,j} - 4w_{i,j-1} + w_{i,j-2}}{\Delta y^4} + O(\Delta y^2) \quad (5)$$

Конечно-разностный аналог смешанной производной:

$$\left. \frac{\partial^4 w}{\partial x^2 \partial y^2} \right|_{i,j} = \frac{1}{\Delta x^2 \Delta y^2} (w_{i+1,j+1} - 2w_{i+1,j} + w_{i+1,j-1} - 2w_{i,j+1} + 4w_{i,j} - 2w_{i,j-1} + \dots) \quad (6)$$

$$+ w_{i-1,j+1} - 2w_{i-1,j} + w_{i-1,j-1}) + O(\Delta x^2, \Delta y^2)$$

Подставим в уравнение (6) аппроксимации производных и отбросим слагаемые более высокого порядка малости, получим разностное уравнение:

$$\begin{aligned} & \frac{u_{i+2,j} - 4u_{i+1,j} + 6u_{i,j} - 4u_{i-1,j} + u_{i-2,j}}{\Delta x^4} + \frac{2}{\Delta x^2 \Delta y^2} (u_{i+1,j+1} - 2u_{i+1,j} + \\ & + u_{i+1,j-1} - 2u_{i,j+1} + 4u_{i,j} - 2u_{i,j-1} + u_{i-1,j+1} - 2u_{i-1,j} + u_{i-1,j-1}) + \\ & + \frac{u_{i,j+2} - 4u_{i,j+1} + 6u_{i,j} - 4u_{i,j-1} + u_{i,j-2}}{\Delta y^4} = f_{i,j} \end{aligned} \quad (7)$$

Граничные условия (7), накладываемые на искомую функцию приобретут вид:

$$w_{0,j} = w_{n,j} = w_{i,0} = w_{i,n} = 0 \quad (8)$$

Аналог граничных условий, накладываемых на частные производные первого порядка, находим из соотношения для конечно-разностного аналога первой производной 2 порядка точности:

$$\begin{aligned} & \frac{-3w_{0,j} + 4w_{1,j} - w_{2,j}}{2\Delta x} = \frac{3w_{n,j} - 4w_{n-1,j} + w_{n-2,j}}{2\Delta x} = \\ & = \frac{-3w_{i,0} + 4w_{i,1} - w_{i,2}}{2\Delta y} = \frac{3w_{i,n} - 4w_{i,n-1} + w_{i,n-2}}{2\Delta y} = 0. \end{aligned} \quad (9)$$

Таким образом, непрерывная краевая задача (1) – (3) заменена конечно-разностной схемой (7) – (9). Полученная схема имеет неявный характер. На рис. 2 изображена расчетная схема на равномерной сетке с шагами $\Delta x = \Delta y = h$.

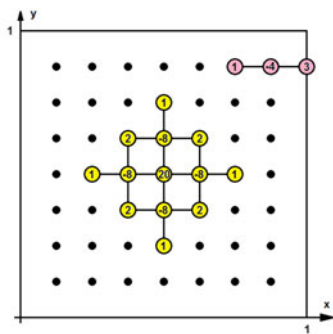


Рис. 2. Расчетная схема

3. Анализ решения

Подберем тестовую функцию таким образом, чтобы она удовлетворяла следующим условиям:

1. Для тонких и жестких пластин $1/100 \leq h/b \leq 1/5$, $w/h \leq 1/4 \Rightarrow w \leq b/20$, где h – толщина пластины, b – наименьший характерный размер пластины (ширина).

2. Выполнение граничных условий.

3. После двукратного применения оператора Лапласа должна получиться непрерывная и ограниченная функция.

В качестве тестовых подберем следующие функции: полиномиальная – $w(x, y) = x^2(1-x)^2 y^2(1-y)^2$ и синусоидальная – $w(x, y) = \frac{1}{\pi^4} \sin^2(\pi x) \sin^2(\pi y)$. Нетрудно увидеть, что условия 1 – 3 для тестовой функции выполняются.

В табл. 1 приведены расчеты абсолютной и относительной погрешностей для различных значений размера сетки для синусоидальной функции. 1 столбец – количество узлов по оси x , 2 – квадрат шага сетки h^2 , 3 – абсолютная погрешность в центральной точке (максимальная), 4 – относительная погрешность в центральной точке, 5 – увеличение количества узлов сетки, 6 – уменьшении шага, 7 – уменьшение относительной погрешности.

Таблица 1

Абсолютная и относительная погрешности синусоидальной функции

1	2	3	4	5	6	7
11	0,01	0,029	0,028			
21	0,0025	0,012	0,011	3,645	2	2,545
31	0,0011	0,006	0,0059	2,179	1,5	1,864
41	0,00063	0,0035	0,0035	1,749	1,33	1,686
51	0,0004	0,0023	0,0023	1,547	1,25	1,522
61	0,00028	0,0016	0,0016	1,431	1,2	1,438
71	0,0002	0,0012	0,0012	1,355	1,17	1,333
81	0,00016	0,00096	0,00096	1,301	1,14	1,25
91	0,00012	0,00076	0,00076	1,262	1,125	1,263
101	0,0001	0,00062	0,00062	1,232	1,111	1,226

В табл. 2 приведены расчеты относительной погрешностей для различных значений размера сетки для полиномиальной функции. 1 столбец – количество узлов по оси x , 2 – квадрат шага сетки h^2 , 3 – относительная погрешность в центральной точке, 4 – увеличение количества узлов сетки, 5 – уменьшении шага, 6 – уменьшение относительной погрешности.

Таблица 2

Относительная погрешность полиномиальной функции

1	2	3	4	5	6
11	0,01	0,11			
21	0,0025	0,025	3,645	2	4,4
31	0,0011	0,011	2,179	1,5	2,27
41	0,00063	0,0064	1,749	1,33	1,72
51	0,0004	0,0041	1,547	1,25	1,56
61	0,00028	0,0029	1,431	1,2	1,41
71	0,0002	0,0021	1,355	1,17	1,38
81	0,00016	0,0016	1,301	1,14	1,31
91	0,00012	0,0012	1,262	1,125	1,33
101	0,0001	0,001	1,232	1,111	1,2

Анализ данных таблиц 1 и 2 позволяет сделать следующие выводы. Во сколько раз увеличивается размерность сетки, во столько же раз уменьшается погрешность, т.е. погрешность обратно пропорциональна размерности сетки и прямо пропорциональна квадрату шага. Реальная погрешность становится в 6 раз больше порядка погрешности, что говорит об их согласованности.

Заключение

С помощью метода конечных разностей разработана неявная конечно-разностная схема для численного решения задачи об изгибе квадратной пластины. Получено решение тестовых примеров в виде сеточных функций для различных шагов сетки. Проанализирована погрешность полученная в ходе вычислительных экспериментов для тестовых примеров. Эта погрешность согласуется с порядком погрешности конечно-разностной схемы.

Список литературы

1. Монахов, В. А. Теория пластин и оболочек: учеб. пособие по направлению подготовки 08.05.01 «Строительство уникальных зданий и сооружений» / В. А. Монахов. – Пенза : ПГУАС, 2016. – 252 с.

2. Ряжских, В. И. О численном интегрировании нестационарного неоднородного бигармонического уравнения в задачах кондуктивной свободной конвекции / В. И. Ряжских, М. И. Попов // Вестник Воронежского государственного технического университета. – 2014. – Т. 10. – № 1. – С. 56–62.

3. Попов, М. И. Интегрирование бигармонического уравнения по неявной схеме / М. И. Попов // Вестник Воронежского государственного университета инженерных технологий. – 2018. – Т. 80, № 2(76). – С. 114–118.

Разработка портала центра развития карьеры и бизнес-партнерства ВГУ

И. Р. Саратов

Студент бакалавр

Е. А. Михайлюк

доцент

Введение

В Воронежском государственном университете обучается более 20000 студентов. Один из острых вопросов любого учебного заведения – это обеспечение прохождения практики студентами и аспирантами, а также трудоустройство выпускников.

Центр развития карьеры и бизнес-партнерства Воронежского государственного университета (ВГУ) предлагает студентам и выпускникам возможность участвовать в современных программах практик/стажировок и трудоустраиваться на передовые предприятия региона и нашей страны, связанных с любыми направлениями подготовки.

На данный момент Центр карьеры взаимодействует с пользователями в вопросах трудоустройства через цифровую карьерную среду «Факультетус», а также имеет google-таблицу для обработки индивидуальных заявок обучающихся. Этих инструментов не достаточно для качественной и оперативной работы по учету практик и стажировок. Поэтому главной причиной разработки портала стала необходимость написания собственной автоматической системы заявок для программ практик и стажировок.

Данное программное обеспечение разрабатывается в интересах Центра карьеры ВГУ в целях повышения эффективности его внутренних процессов. В целях более качественной работы сотрудников Центра карьеры необходимо внедрение специализированного программного обеспечения. Это позволит оптимизировать внутренние процессы, автоматизировать обработку заявок и улучшить качество предоставляемых услуг. Разработка нового программного продукта позволит повысить эффективность работы Центра карьеры и обеспечить более высокий уровень сервиса для студентов и выпускников вуза.

Также данная работа рассматривается как разработка небольшой полноценной CMS для сайта, которая может быть использована в ряде сценариев основных бизнес-процессов ЦРК и БП ВГУ.

1. Теория и постановка задачи

Перед началом разработки системы был проведен анализ предметной области и выявлены определенные требования к данному продукту.

Для заказчика необходимо разработать информационную систему, сочетающую в себе простоту использования, интеграцию максимального количества процессов Центра карьеры, а также отказоустойчивость.

Первый этап разработки требовал создания системы, где имеются статические информационные страницы, а также табличная система заявок.

На рис.1 указана диаграмма последовательностей относительно обработки заявок.

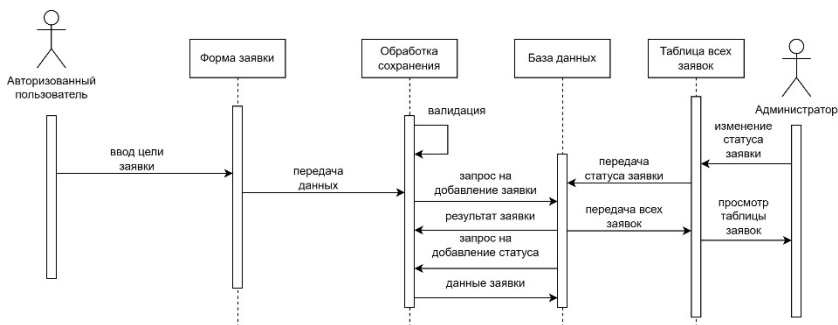


Рис. 1. Диаграмма последовательности для работы с заявками

На втором этапе разработки создается редактор статических страниц и новостей, опросник, автопостинг новостей и телеграм-бот.

2. Структура информационной системы

В самом начале разработки в качестве основы стека технологий был выбран язык программирования Python и фреймворк Django, схема которого указана на рис.2.

Фреймворк устроен по идеологии MVT (Model-View-Template) [1], где:

1. Model (модель) – отвечает за структуру данных в приложениях. Отдельные классы соответствует таблицам в базе данных.

2. View (представление) – отвечает за обработку взаимодействия с пользователем, работает как интерфейс взаимодействия с моделью.
3. Template (шаблон) – это компонент, который фактически просматривается пользователем. Он отвечает за взаимодействие с пользователем. Шаблон состоит из статических частей желаемого HTML-вывода, а также некоторого специального синтаксиса, описывающего, как будет вставляться динамический контент.

При этом представление в Django скорее выполняет функцию контроллера, а шаблон – представления, если сравнить это с классическим MVC подходом.

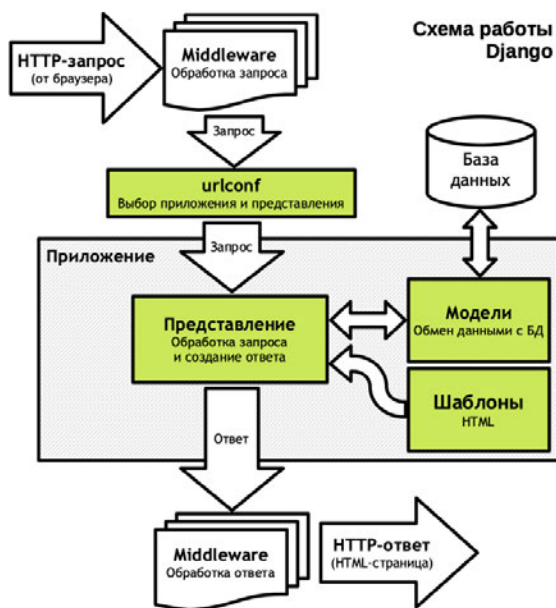


Рис. 2. Структура фреймворка Django

Данный стек технологий имеет следующие преимущества [2]:

1. Простота и скорость разработки
2. Удобство переноса проекта на сервер
3. Легкость установки внешних модулей и библиотек
4. Широкие возможности фреймворка
5. Удобное масштабирование кода

Данный фреймворк имеет встроенную систему объектно-реляционных отношений DjangoORM [3], которая позволяет быстро изменять структуру таблиц в базе данных, а также надежно экранирует запросы, что исключает SQL-инъекции. Является одним из самых эффективных решений по защите баз данных с открытым исходным кодом [4].

Фреймворк также обладает встроенными решениями по защите от XSS-уязвимостей и за счет токенизации защищает от CSRF-атак.

Недостатком выступает относительно большое потребление ресурсов, из-за того, что язык Python является интерпретируемым, утечками памяти, связанными с несовершенством некоторых модулей и библиотек, а также большим количеством слоев абстракций, возникающих в ходе изготовления готового проекта.

В качестве базы данных используется встраиваемая SQLite. Она является встраиваемой СУБД с собственным диалектом языка SQL, автоматически устанавливается в Django-проект при его создании, потребляет небольшое количество серверных ресурсов и обеспечивает потребности портала на данном этапе.

Для взаимодействия с внешними ресурсами используется DjangoRestFramework, позволяющий эффективно и быстро реализовать API по стандартам Rest [5].

Отдельный фреймворк и API для фронтенда не применяются, вместо этого используются Django-шаблоны.

На рис.3 указана структура проекта.

Входящие в его состав Django-приложения имеют следующее предназначение:

- MyCareer (пользователи и табличная система заявок);
- Survey (опросы);
- Blog (редактор новостей и статических страниц);
- Social_api(автопостинг для VKи Telegram);
- EmailBulletin(новостные рассылки).

3. Взаимодействие пользователя с системой

Посетителя сайта встречает стартовая страница, изображенная на рис.4, на которой имеются панель виджетов, панель навигации и информационная сводка.

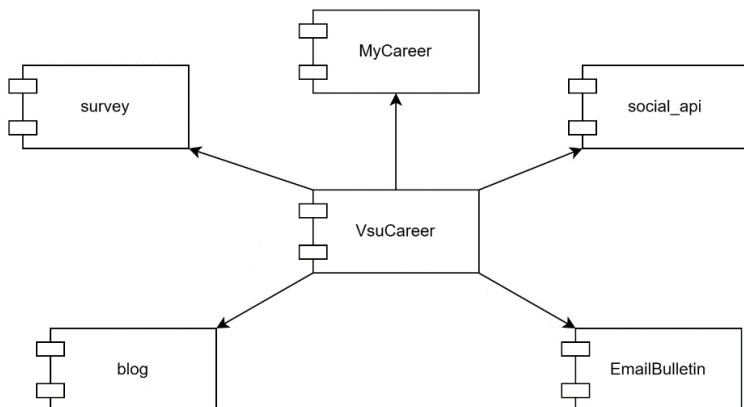


Рис. 3. Структура проекта

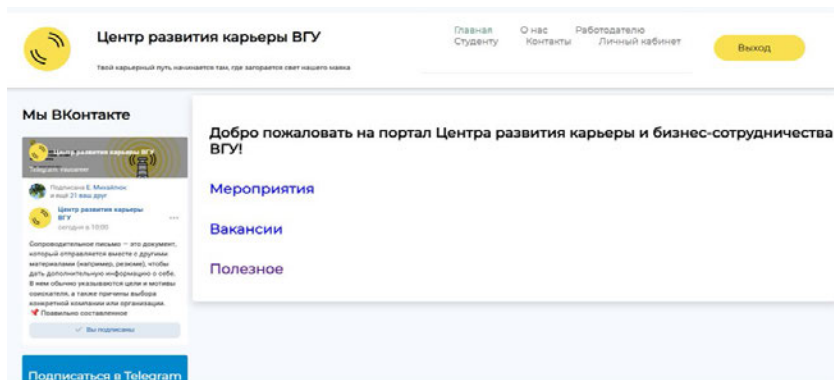


Рис. 4. Главная страница

Также обычный пользователь имеет доступ к подаче заявки и отслеживанию ее статуса.

Пользователям с правами администратора, которыми являются сотрудники Центра карьеры, доступно управление сайтом из админ-панели Django, указанной на рис. 5, в которой возможно управление любыми данными в БД.

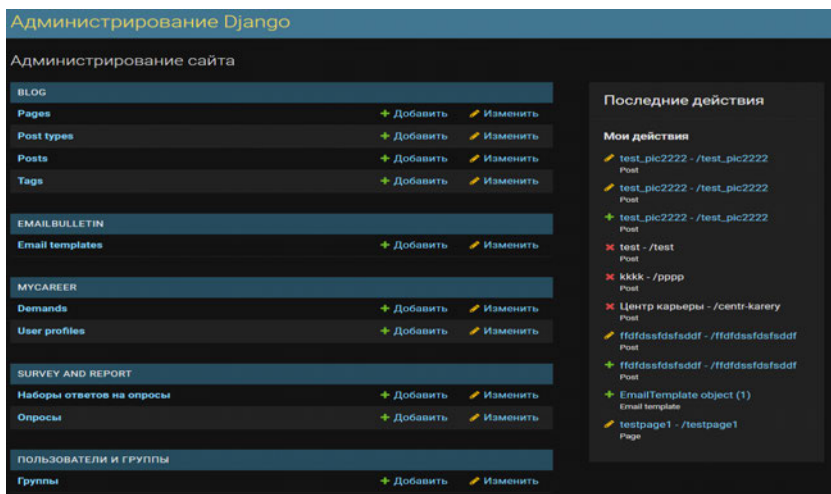


Рис. 5. Django Admin

В модуле автопостинга новостей используется взаимодействие с VK API и Telegram API. Администратор может создать новость в рамках имеющегося новостного функционала портала и автоматически запостить ее в официальные сообщества Центра карьеры в данных ресурсах.

При автопостинге формат RichText, в котором создаются новостные посты, разбивается на 4 составляющие:

1. Заголовок поста (формат text);
2. Текст поста без форматирования (формат text);
3. Теги поста (формат text);
4. Изображения поста (формат base64).

Это связано с тем, что API вышеописанных ресурсов не поддерживают работу с форматом RichText

В целях выполнения данной задачи администратор вводит действующий токен социальных сетей и ID канала/группы, а при постинге выбирает, куда отправить запись.

Модуль Survey реализован на базе библиотеки django-survey-and-report, дает администратору возможность создавать опросы и взаимодействовать с их результатами.

Разрабатываемый telegram-бот взаимодействует с порталом через Rest API, позволяет пользователю портала отслеживать статус поданной заявки, просматривать справочную информацию и получать новостные рассылки, аналогичные email-рассылкам.

Заключение

Получаемая в итоге разработки информационная система Центра карьеры ВГУ позволяет повысить эффективность работы сотрудников данного Центра при работе с индивидуальными заявками студентов.

Автоматизация заявок на партнерские программы, оказываемые при содействии с Центром, увеличивает привлекательность взаимодействия клиентов с Центром, а также уменьшает рутинный труд сотрудников.

Данная информационная система после ее внедрения будет иметь следующий эффект:

1. Увеличение привлекательности и авторитета Центра карьеры;
2. Рост объема услуг, оказываемых Центром;
3. Наличие единой точки для обработки данных, связанных с Центром;
4. Рост показателей трудоустройства и взаимодействия студентов и выпускников с партнерами университета и факультетов;
5. Потенциал развития партнерских программ Воронежского государственного университета;
6. Повышение эффективности труда сотрудников Центра развития карьеры.

Список литературы

1. Дронов, В.А Django 4. Практика создания веб-сайтов на Python [Текст] / В. А. Дронов. – СПб.: БХВ-Петербург, 2023. – 791 с.
2. Эффективный Django. Часть 1 [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/articles/240463/>
3. Радж, Ш. ORM Cookbook [Текст] / Ш. Радж, Я. Растоги. – Хайдарабад: Agiliq Info Solutions India Pvt Ltd, 2018 – 47 с.
4. An overview of the built-in security features of Django and best practices for keeping your application safe from attacks [Электронный ресурс]. – Режим доступа: <https://www.linkedin.com/pulse/overview-built-in-security-features-django-best-practices-shaikh>
5. Дэвис, Дж. The Django Book Django for Entrepreneurs [Текст] / Дж. Дэвис. – Портланд: Softcover, 2017. – 200 с.

Инвалидация кэшей с использованием Debezium

А. Р. Сашина

Студент бакалавр

А. Л. Гавшин

Старший преподаватель

Введение

Современные веб-приложения для обеспечения быстрой скорости получения ответа на запрос пользователя используют кэши. В данной работе под кэшем подразумевается промежуточный буфер данных, который хранит в себе часто используемую информацию из базы данных.

Зачастую в таких системах, которые используют кэши, возникает проблема инвалидации [1], то есть обновления или удаления устаревших данных. В данной работе будет рассмотрена проблема инвалидация кэшей и способы ее решения с использованием платформы Debezium.

1. Проблема инвалидации кэша с переменным временем обновления данных

В качестве примера будет рассмотрено приложение с архитектурой, которая приведена на рис. 1. Серверная часть (бэкенд) взаимодействует с базой данных и имеет кэш на уровне приложения, который хранит в себе часто используемую информацию из базы. Для системы с малой нагрузкой и среднем временем обновлении информации (данные обновляются не часто, но и не слишком редко) в базе данных такая архитектура является приемлемой. Однако, когда количество запросов к определенной информации становится больше или, наоборот, меньше, то для этого нужно изменять конфигурацию кэша.

Для решения вышеприведенной проблемы можно использовать evict-методы [2], которые очищают кэш. Существуют разные стратегии по удалению устаревших данных, а также различные реализации для разных фреймворков [3]. Например, при использовании Spring фреймворка можно для кэша задать TTL (Time To Live, то есть время, в течение которого данные считаются актуальными), и тогда кэш будет автоматически удалять ту информацию, у которой TTL истек. Также

можно при вызове определенного метода, например удаления сущности, автоматически убирать эту же сущность из кэша, в котором она хранится.

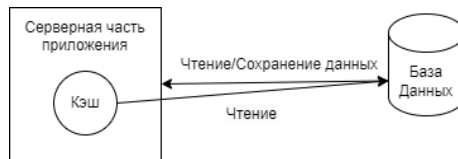


Рис. 1. Архитектура приложения с использованием кэша

Пусть количество запросов становится переменным, то есть в какое-то время происходит частое обновление данных, а в другое наоборот, очень редко происходит модификация. Тогда задание TTL становится сложной задачей, так как, выбирая максимальное значение, кэш будет хранить неактуальную информацию, а, выбирая среднее или минимальное значение, будет слишком часто делать лишние обновления. Та же проблема относится и к частоте вызова `evict`-методов.

2. Использование Debezium для актуализации данных в кэше

Debezium – это распределенная платформа, созданная как реализация паттерна захвата изменения данных [4]. Он использует журналы транзакций базы данных и создает потоки изменения данных. Debezium можно использовать как коннектор к базе данных, который отправляет сообщения в очередь, например, Apache Kafka, или использовать Debezium Server для отправки событий в различные инфраструктуры обмена сообщениями, например Redis [5]. Также Debezium может быть использован как библиотека для приложений на Java и Kotlin.

На рис. 2 представлена архитектура приложения, которая использует библиотеку Debezium Engine. В этом случае нет надобности создавать очередь сообщений для поддержки работы платформы. В приложение добавляется Debezium listener сервис, который отслеживает изменения данных в определенных отношениях базы данных. При изменении сущности в базе данных listener почти в реальном времени получает об этом информацию и обновляет соответствующей этой сущности кэш. Таким образом, в задании `evict`-методов или TTL больше нет необходимости, поскольку данные в кэше обновляются практически сразу [6].

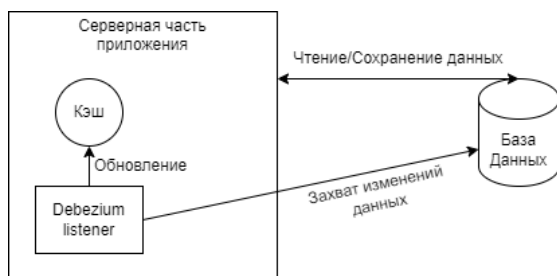


Рис. 2. Архитектура приложения с фреймворком Spring и Debezium

Как выше упоминалось, Debezium можно использовать как коннектор к базе данных с очередью сообщений. На рис. 3 представлена такая архитектура приложения. Данная реализация подходит для тех систем, где уже настроена или легко интегрируется очередь сообщений, например Apache Kafka. Debezium-коннектор захватывает изменения сущностей в базе данных и отправляет сообщения в топик Kafka. Консьюмер приложения вычитывает сообщения из этого топика и обновляет кэш.

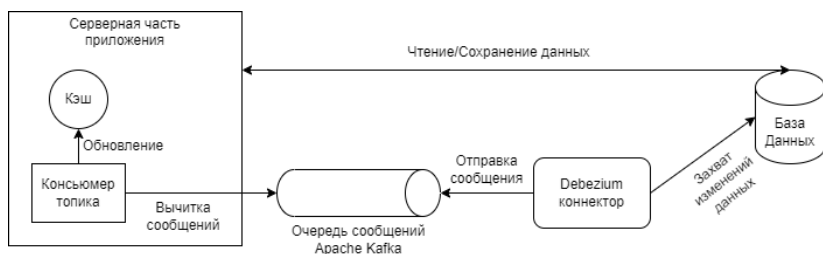


Рис. 3. Архитектура приложения с очередью сообщений Apache Kafka и Debezium

3. Использование Debezium для глобального кэша

Современные веб-приложения зачастую имеют не один, а несколько сервисов бэкенда. Для них используется либо распределенный кэш, либо глобальный [7]. Как было упомянуто выше, Debezium можно использовать как отдельный сервис. На рис. 4 показана архитектура приложения с несколькими сервисами бэкенда. Для них выделен отдельный глобальный кэш, который обновляет Debezium Service при получении изменений сущностей в базе данных. При таком

подходе кэш тоже обновляется в реальном времени, а сервисы имеют одинаковый набор данных.

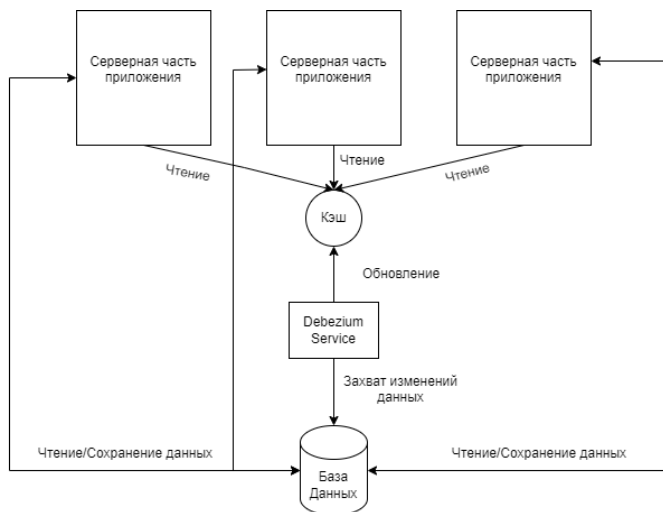


Рис. 4. Архитектура масштабируемого веб-приложения с Debezium Service

Заключение

Данная статья посвящена описанию проблемы инвалидации кэша в приложении и способ ее решения с использованием платформы Debezium. Были рассмотрены разные подходы к использованию Debezium. Вышеописанные решения позволяют инвалидировать кэши в реальном времени без создания `evict`-методов и задания TTL. Также использование Debezium для инвалидации кэша может быть масштабировано без потери согласованности данных.

Список литературы

1. Паттерн кэширование [Электронный ресурс] : статья про кэширование. – Режим доступа: <https://www.geeksforgeeks.org/caching-design-pattern>
2. Стратегии `evict`-методов в кэшах [Электронный ресурс] : статья с описанием различных стратегий по очищению кэша. – Режим доступа: <https://www.geeksforgeeks.org/cache-eviction-policies-system-design>
3. Кэширование в spring фреймворке [Электронный ресурс] : документация spring фреймворк по использованию кэшей в

приложении. – Режим доступа: <https://docs.spring.io/spring-framework/docs/4.0.x/spring-framework-reference/html/cache.html>

4. Сашина, А. Р. Использование Debezium в микросервисной архитектуре для реализации паттерна захвата изменения данных / А. Р. Сашина, А. Л. Гавшин // Труды молодых учёных факультета компьютерных наук ВГУ. – Воронеж, 2023. – Выпуск 3. – С. 195-198.

5. Архитектура Debezium [Электронный ресурс] : документация о различных архитектурах Debezium. – Режим доступа: <https://debezium.io/documentation/reference/2.5/architecture.html>

6. Инвалидация кэша с использованием паттерна захвата изменения данных [Электронный ресурс] : статья про инвалидацию кэша с помощью Debezium. – Режим доступа: <https://debezium.io/blog/2018/12/05/automating-cache-invalidation-with-change-data-capture>

7. Сюй А., System Design. Подготовка к сложному интервью / А. Сюй. – СПб.: Питер, 2022. – 304 с.

Автоматизация процесса аутентификации пользователя в системах прокторинга

А. В. Сиваков

Студент бакалавр

А. В. Швырева

Ассистент

Введение

В настоящее время все большее применение находят дистанционные образовательные технологии, в том числе технологии дистанционного тестирования. Неотъемлемой частью такого тестирования является использование системы прокторинга, позволяющей анализировать множество параметров, такие как: положение рук экзаменуемого, присутствие посторонних лиц в кадре, подозрительные звуки, действия экзаменуемого мышью на компьютере и так далее. При широком многообразии возможностей системы, проверка пользователя при входе в систему прокторинга до сих пор происходит с помощью оператора. Представленное ограничение создает большую нагрузку и снижает пропускную способность при входе системы.

Для решения данной проблемы можно использовать методы машинного обучения, такие как: методы обработки изображений и сверточные нейронные сети. Целью данной работы является исследование возможности применения методов машинного обучения для автоматизации процесса аутентификации пользователя в системах прокторинга. В разрабатываемой системе необходимо

1. Общее устройство системы.

Для работы с системой прокторинга и при регистрации в системе пользователь дает согласие на хранение и обработку его персональных данных. Общий принцип аутентификации в системе прокторинга представлен на рисунке 1.

Аутентификация происходит в три этапа.

1. Пользователь вводит логин/пароль, которые сверяются с данными, указанными при регистрации и хранящимися в базе данных.

2. На изображении, полученном с веб-камеры, выделяется лицо пользователя и вычисляется вектор признаков, который далее сверяется с вектором признаков, хранимом в базе данных.

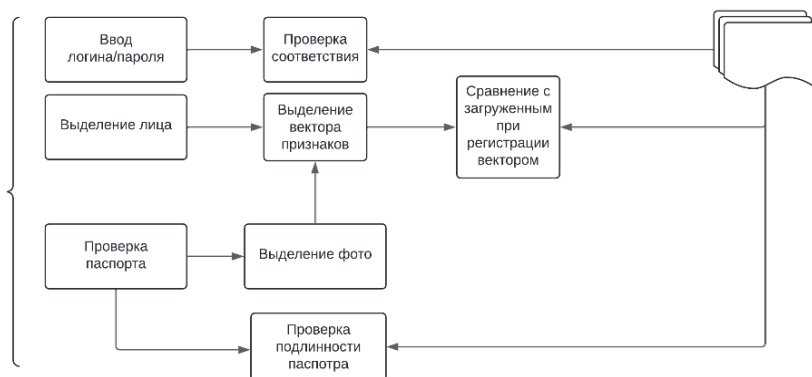


Рис. 1. Порядок аутентификации

3. Система просит пользователя показать в камеру первый разворот оригинала паспорта.

Проверка паспорта проводится в два этапа. На первом этапе выделенное изображение лица на паспорте и сверяется с загруженным в базу данных. На втором этапе изображение паспорта целиком сверяется с изображением в базе данных для проверки подлинности документа. Ответственность за подлинность загруженного при регистрации изображения лежит на платформе, которая проводит экзамен и системой проверена быть не может.

Прохождение всех этапов аутентификации позволяет получить пользователю доступ к системе. В случае возникновения ошибок, предлагается пройти повторную проверку. Если повторная проверка также была отклонена, пользователь направляется к оператору системы для ручной проверки документов.

В качестве языка разработки был выбран Python из-за наличия большого количества библиотек машинного обучения и для работы с изображениями, в том числе OpenCV [1, 2].

2. Захват видеопотока

Для захвата видео с веб-камеры была использована библиотека openCV. Стандартные веб-камеры поддерживают частоту кадров 30-60 кадров в секунду. Предполагается разделять видео на отдельные кадры и анализировать каждый кадр как отдельное изображение. Однако все полученные за секунду кадры использовать нет необходимости: так как

кадры, близкие друг к другу по времени будут очень похожи, и результат их обработки с наибольшей вероятностью будет идентичным, при этом вычислительная нагрузка сильно возрастет. Поэтому было принято решение ограничивать частоту видео искусственно до трёх кадров в секунду.

3. Получение признаков и сравнение изображений

Для получения вектора признаков и сравнения лиц используется сверточная нейронная сеть без выходного классификатора. В качестве такой нейронной сети была использована обученная модель, доступ к которой предоставляется с помощью соответствующего API библиотекой `face_recognition`. Изначально библиотека предполагает сбор всех векторов в единый датасет, по которому выполняется поиск. Этот принцип был изменён: в проектируемой системе было решено оформлять каждый вектор признаков в отдельный файл и выполнять сравнение лишь с одним вектором по составному ключу логин/пароль. Ниже приведены исходный (рис. 2) и доработанный (рис. 3) принцип классификации.

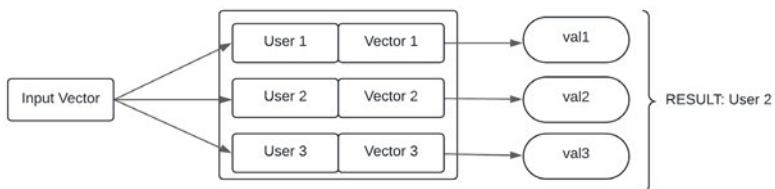


Рис. 2. Исходный принцип классификации

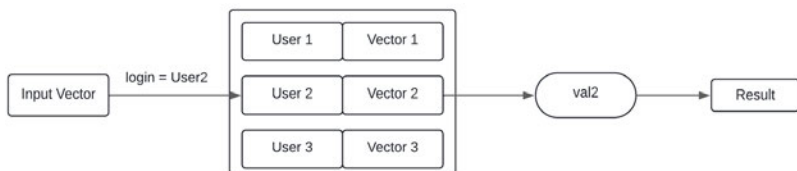


Рис. 3. Доработанный принцип

Была произведена замена задачи идентификации на задачу верификации, что позволяет сократить асимптотическую сложность алгоритма с $O(N * f(N))$ до $O(f(N))$, где N – число пользователей, а $f(N)$ – сложность поиска по ключу, зависящая от индексирования базы данных (предполагая, что сложность сопоставления векторов постоянна).

Для сравнения векторов использована косинусная метрика. Для двух векторов A , B она вычисляется следующим образом [3]:

$$\text{similarity}(A, B) = \frac{A * B}{\|A\| \|B\|},$$

где норма каждого вектора определяется, как: используется модуль (V_i – компонент вектора под номером i , нумерация с 1):

$$\|A\| = \sqrt{\sum_{i=1}^N (a_i)^2},$$

где a_i – i -тый компонент вектора A .

В результате получается значение, лежащее в диапазоне $[0; 1]$, где 0 соответствует минимальному сходству, а 1 – максимальному. Если значение метрики не меньше порогового, считается, что изображения совпадают, иначе – что не совпадают [4,5].

В результате удалось создать систему, которая верно распознавала зарегистрированного и незарегистрированного пользователя по его изображению и изображению из паспорта (рис. 4, 5).

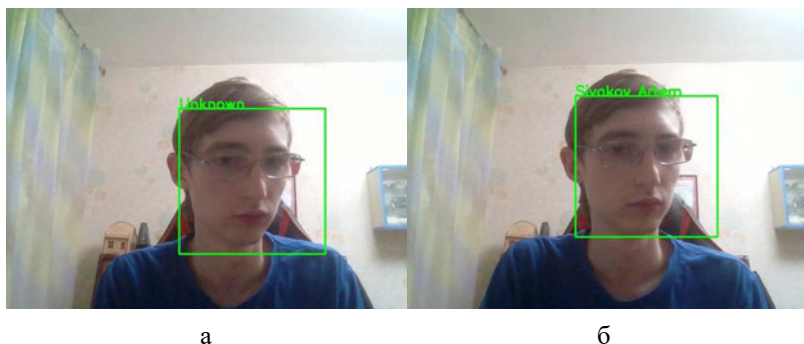


Рис. 4. Распознавание для зарегистрированного и незарегистрированного пользователя



Рис. 5. Распознавание фото из паспорта

4. Сопоставление изображений паспортов при помощи SIFT.

На последнем этапе аутентификации выполняется сопоставление изображений паспорта при помощи алгоритма SIFT. В работе использован готовый алгоритм SIFT из библиотеки openCV. При проверке вычисляется количество совпавших точек. Если оно не ниже порога – изображения считаются совпавшими, иначе – не совпавшими. В ходе тестирования было выявлено, что в ситуациях, когда паспорта действительно совпадают, количество точек всегда 200 (медиана – 290), в то время как для несовпадающих паспортов оно ни в одном тесте не превысило значение 30. В итоге был выбран порог в 150 точек. На рисунках 7 и 8 представлены примеры сравнения паспортов. Даже в ходе визуального анализа очевидна разница в количестве совпадающих точек.



Рис. 6. Сопоставление совпадающих паспортов



Рис. 7. Сопоставление несовпадающих паспортов

Заключение

В результате выполнения работы был разработан алгоритм и реализующий его программный модуль, автоматизирующий аутентификацию пользователя, применимую в системах прокторинга, с целью сокращения нагрузки на оператора системы и повышения пропускной способности, с помощью технологий машинного обучения и биометрических методов. Также было проведено тестирование разработанного программного средства, показавшее его эффективность в контексте решаемой задачи.

Список литературы

1. Свейгард, Э. Автоматизация рутинных задач с помощью Python / Э. Свейгард. – 2 изд. – Диалектика-Вильямс, 2021. – 672 с.
2. Форсайт, Д. Компьютерное зрение. Современный подход / Д. Форсайт, Д. Понс ; Вильямс, 2018. – 960с.
3. Jurafsky, D. Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition / D. Jurafsky; – second edition. – Pearson Education International, 2009. – 1024 pp.
4. Рашка, С. Python и машинное обучения / С. Рашка ; ДМК Пресс, 2017. – 418 с.
5. Домингос, П. Верховный алгоритм / П. Домингос; Манн, Иванов и Фербер (МИФ), 2015. – 480 с.

Разработка приложения для восстановления когнитивных способностей у людей с нарушениями ЦНС и опорно-двигательного аппарата

Д. Ю. Скарга

Студент бакалавр

Е. А. Рогачев

Студент бакалавр

Я. А. Туровский

Заведующий лабораторией

Введение

Значимость реабилитации для восстановления когнитивных функций у пациентов с заболеваниями центральной нервной системы (ЦНС) и опорно-двигательного аппарата является важным аспектом в медицинской практике. Когнитивные нарушения, связанные с этими патологиями, могут оказать значительное влияние на жизнь человека, снижая его способность выполнять повседневные задачи, поддерживать социальные отношения и заниматься деятельностью, требующей участия памяти, внимания и концентрации.

Разработанное приложение предназначено для пациентов, столкнувшихся с травмами головного или спинного мозга, а также для лиц с другими неврологическими заболеваниями, оказывающими влияние на когнитивные функции. Оно также адресовано пользователям с ограниченными возможностями опорно-двигательного аппарата, которым необходима тренировка моторики. Помимо самих пациентов, приложение представляет интерес для их родственников и близких лиц, участвующих в процессе реабилитации, а также для медицинских специалистов – врачей, неврологов, реабилитологов, которые могут использовать его для оценки и восстановления когнитивных функций пациентов.

Основная цель создания данного приложения состоит в автоматизации процесса проведения терапии, направленной на преодоление когнитивного дефицита.

Значимость данного исследования заключается в том, что традиционные методы реабилитации зачастую требуют значительных человеческих ресурсов, что делает этот процесс затратным и длительным, требуя постоянного присутствия медицинского персонала. Автоматизация процесса мониторинга пациентов и предоставление персонализированных программ когнитивной реабилитации предоставляют новый подход, который может существенно улучшить результаты и повысить качество жизни для пациентов с нарушениями ЦНС и опорно-двигательными расстройствами. [1]

1. Теоретическая база

Когнитивные нарушения – это снижение памяти, умственной работоспособности и других когнитивных функций по сравнению с исходным уровнем.

Когнитивными (познавательными) функциями называются наиболее сложные функции головного мозга, с помощью которых осуществляется процесс рационального познания мира и обеспечивается целенаправленное взаимодействие с ним: восприятие информации; обработка и анализ информации; запоминание и хранение; обмен информацией, построение и осуществление программы действий.

К счастью, достижения в области нейронаук привели к более глубокому пониманию нейропластичности – способности мозга реорганизовываться в ответ на травму или опыт. Эта концепция проложила путь к разработке инновационных стратегий реабилитации, направленных на улучшение когнитивных функций. Продолжая оптимально распределять ресурсы, головной мозг передает функцию погибших клеток выжившим. Выстраиваются новые связи и пути взамен утраченных. Чтобы помочь естественным процессам нейропластичности, важно ставить перед нервной системой новые задачи, как можно раньше показывать и помогать искать обходные пути и механизмы компенсации. Нейрогенез – это многоступенчатый процесс образования новых нервных клеток в зрелой центральной нервной системе (ЦНС), являющийся ее адаптивной функцией. Рост новых нейронов в мозге может происходить в ответ на физические упражнения и когнитивные тренировки. [2]

Процесс синаптогенеза описывает создание и укрепление устойчивых нейросинаптических связей в ответ на стимуляцию головного мозга. Ежедневные занятия способствуют улучшению важных когнитивных показателей и нормализации работы центральной нервной системы.

2. Функциональные требования к приложению

При разработке приложения учитывались следующие требования:

- пациенту должны выдаваться задания в звуковой форме;
- должна быть реализована возможность загрузить аудиозаписи и выдавать команды голосами знакомых пациенту людей;
- реализован контроль корректности выполнения заданий;
- обратная связь в звуковой форме о правильности выполнения заданий;
- запись количества правильных и неправильных выполнений заданий и сохранение этой информации для отслеживания прогресса пациента и построения графика на ее основе.

Таким образом, необходимо реализовать приложение, которое будет подавать пациенту команды по переключению предметов между цветными зонами, проверять правильность их выполнения и выдавать обратную связь. Выполняя такие задания, пациент сможет тренировать память, внимание и концентрацию, при этом не будет требоваться постоянное присутствие медицинского персонала, который сможет отслеживать прогресс пациента на основе построенных программой графиков.

3. Реализация

Приложение написано на языке программирования Python с использованием следующих библиотек: `simpleaudio` для воспроизведения звуковых файлов, `matplotlib` для построения графиков отслеживания прогресса пациента на основе выполняемых им заданий, `PyQt` для оконного интерфейса, `nimru` для выполнения необходимых расчетов, `pyttsx3` для преобразования текста в речь, а также библиотека алгоритмов компьютерного зрения и обработки изображений `OpenCV`, которая работала в сочетании с технологией `ArUco` – это система создания и обнаружения маркеров в дополненной реальности, которая используется для добавления виртуальных объектов или информации к реальному окружению с помощью камеры устройства. [3]

В начале реализации исходного кода приложения происходит импорт необходимых библиотек, после этого инициализируется работа с камерой, что позволяет получать видеопоток для дальнейшей обработки изображений. С использованием функций `OpenCV` для работы с `ArUco` происходит обнаружение и распознавание визуальных маркеров на изображении с камеры.

В листинге ниже приводится пример кода для определения положения предмета в пространстве.

Определение положения предмета

```

@staticmethod
def is_in_zone(x, y, left_bottom_x, left_bottom_y,
              right_top_x, right_top_y):
    if (x >= left_bottom_x) and (x <= right_top_x)
    and (y >= left_bottom_y) and (y <= right_top_y):
        return True
    return False

def center_coords(self, aruco_id, corners):
    return self.detector.center_coords(aruco_id, corners)

def in_red_zone(self, x, y):
    left_corner = self.define_params(0)
    right_corner = self.define_params(3)
    if left_corner is None or right_corner is None:
        if self.define_params(1) is not None and
        self.define_params(2) is not None:
            left_corner = self.define_params(1)
            right_corner = self.define_params(2)
            return self.is_in_zone(x, y, min(left_corner[0],
            right_corner[0]), min(left_corner[1],
            right_corner[1]),
            max(left_corner[0], right_corner[0]),
            max(left_corner[1], right_corner[1]))
        return False
    return self.is_in_zone(x, y, min(left_corner[0],
    right_corner[0]), min(left_corner[1], right_corner[1]),
    max(left_corner[0], right_corner[0]),
    max(left_corner[1], right_corner[1]))
def define_params(self, marker_id):
    if marker_id in np.ravel(self.ids):
        index = np.where(self.ids == marker_id)[0][0]
        return self.detector.center_coords(index,
        self.corners)
    return None

```

Информация, полученная из обнаруженных маркеров ArUco, используется для точного определения зоны, в которой располагается предмет. Это позволяет системе точно определять положение предметов на поле, используя его для определения правильности выполнения заданий. [4]

4. Интерфейс пользователя

Приложение имеет оконный интерфейс. Изначально необходимо установить камеру так, чтобы было видно поле для выполнения заданий. Поле состоит из 4 областей разного цвета. На каждое из которых по углам нужно приклеить Aruco маркеры (показаны на рис. 1.),

так же их необходимо приклеить и на предметы (по одному на каждый предмет), которые будут использоваться в задании – 4 предмета – их нужно расположить произвольно на поле внутри цветных областей.

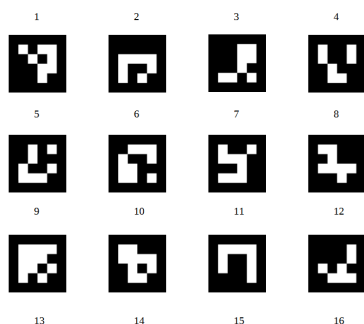


Рис. 1. ArUco Маркеры

После чего, если мы не загружаем аудио записи, то необходимо в соответствующие поля ввести названия предметов и указать время, которое будет даваться пациенту на выполнение каждого задания. Если же вы хотите использовать свои аудио записи для оглашения заданий, то необходимо сначала их записать и загрузить. Далее необходимо указать файл для результатов. После этого можно нажимать кнопку Старт, подождать, пока программа подключится к камере и запустится. Тогда вы услышите первые задания.

Задания бывают двух видов: переложить определенный предмет в определенную область или поменять предметы местами (Например: переложите “название предмета” в зеленую зону). Поле для выполнения заданий представлено на рис. 2.

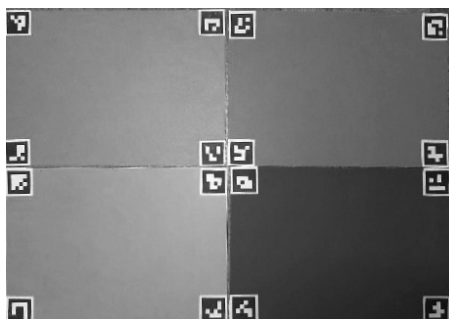


Рис. 2. Поле для выполнения заданий

После запуска программы медицинским работником (интерфейс представлен на рис. 3.) пациент в удобной для себя обстановке выполняет задания, подаваемые программой, правильность выполнения которых анализируется, а результат выдается голосом [5].



Рис. 3. Интерфейс для медицинского персонала

Когда необходимо закончить выполнение заданий, нажмите кнопку «Закончить работу и показать результаты» После чего появится график, отображающий прогресс пациента (в процентах правильного выполнения заданий). Пример такого графика представлен на рис. 4.

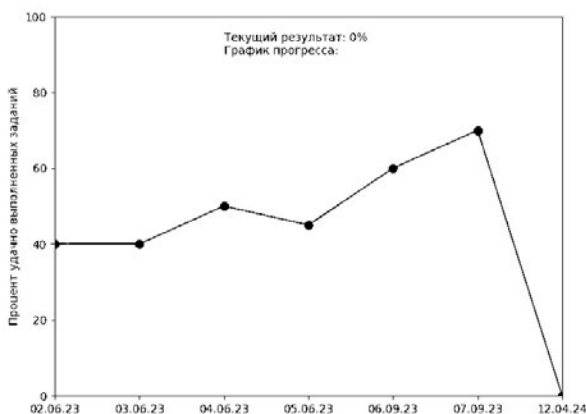


Рис. 4. График прогресса

Заключение

Данная статья посвящена реализации приложения, которое может помочь людям с нарушениями центральной нервной системы и опорно-двигательного аппарата в восстановлении когнитивных способностей.

В качестве дальнейшего направления работы можно рассмотреть добавление новых типов заданий, увеличение количества цветов и предметов на поле, а также интеграция и тестирование приложения в медицинских учреждениях.

Список литературы

1. Интерактивные технологии и когнитивная реабилитация [Электронный ресурс]. – Режим доступа: <https://www.istok-reatech.ru/catalog/interaktivnye-tehnologii-i-kognitivnaya-reabilitatsiya/>
2. Нейропластичность мозга в реабилитации [Электронный ресурс]. – Режим доступа: <https://walkaide.ru/tehnologii/neuroplastichnost-kak-osnova-reabilitatsii/neuroplastichnost-mozga-v-reabilitatsii/>
3. Библиотека компьютерного зрения OpenCV [Электронный ресурс]. – Режим доступа: <https://opencv.org/>
4. Определение положения с ArUco [Электронный ресурс]. – Режим доступа: <https://pyimagesearch.com/2020/12/28/determining-aruco-marker-type-with-opencv-and-python/>
5. Реабилитация на дому [Электронный ресурс]. – Режим доступа: <https://almadeya.ru/reabilitaciya/pochemu-vo-mnogih-sluchayah-reabilitaciya-na-domu-effektivnee/>

Автоматическая генерация пользовательского интерфейса на фреймворке Angular для бэкенд приложений

К.Ю. Скофенко

Студент бакалавр

А.И. Чекмарев

Старший преподаватель

Введение

Создание пользовательского интерфейса – это неотъемлемая часть разработки веб-приложений. При этом его разработка требует значительных затрат времени и усилий со стороны команды разработчиков. Требования к пользовательскому интерфейсу в каждом конкретном случае предъявляются разные и полностью автоматизировать процесс его создания затруднительно.

С другой стороны, возможно создать инструмент, который будет автоматически создавать стандартизированный интерфейс для готовых бэкенд приложений и предоставлять возможность его дальнейшего редактирования. Такой сгенерированный интерфейс с минимальной модификацией может использоваться для администрирования, службы поддержки и прочих внутренних систем. Или он может послужить основой для полноценной разработки сложного, продвинутого интерфейса для взаимодействия с конечными пользователями, тем самым сокращая время его разработки.

Однако существующие решения для автоматической генерации пользовательского интерфейса либо не предоставляют возможности его редактирования как Swagger/OpenAPI, либо требуют ручного задания конфигурации приложения – описания его эндпоинтов, принимаемых и возвращаемых значений, и т.д., как NG Dynamic Forms [1-2].

1. Общая архитектура

Разрабатываемая система включает две основные части: генератор конфигурации бэкенд приложения и средство для построения Angular-приложения по заданной конфигурации. Схематично архитектура представлена на рис. 1.

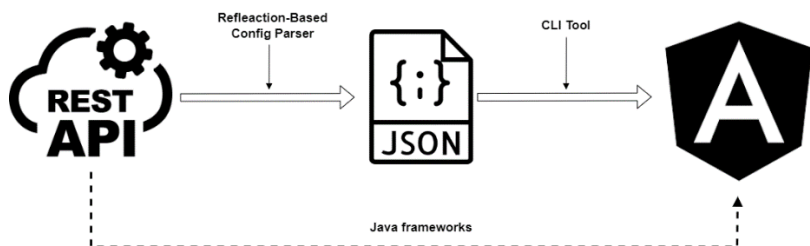


Рис. 1. Архитектура системы на верхнем уровне

Генератор конфигурации не представляет собой единое приложение для всех фреймворков и языков программирования, а набор библиотек, модулей или пакетов для каждого отдельного бэкэнд фреймворка. Каждый генератор должен учитывать особенности платформы, для которой он разрабатывается – используемые в ней аннотации, атрибуты, декораторы, организацию кода и так далее.

Построение конфигурации основано на рефлексии – способности программы анализировать и изменять свою структуру и поведение во время выполнения. При запуске приложения генератор анализирует его структуру и создает соответствующую ей конфигурацию в виде JSON файла, сохраняемого в файловой системе.

При нехватке информации для построения конфигурации без модификации исходной кодовой базы, вводятся синтаксические конструкции, содержащие необходимые метаданные в виде новых декораторов, атрибутов и прочего. Они также могут использоваться для задания дополнительной информации в конфигурации пользовательского интерфейса, например отображаемых имен.

Вторая часть системы – сам генератор пользовательского интерфейса. Он принимает файл конфигурации, созданный на предыдущем шаге и создает по нему Angular приложение, используя FreeMarker шаблоны. Результат его работы – полноценное приложение на Angular, которое может редактировать пользователь по своему усмотрению. Также в генераторе предусмотрена возможность автоматической установки зависимостей и локального запуска созданного пользовательского интерфейса.

Две части системы могут быть объединены в случае создания интерфейса для Java приложений. Тогда библиотека парсера конфигурации будет использовать генератор UI в качестве своей зависимости и напрямую передавать ему созданную конфигурацию, минуя шаг с JSON файлом. Конечный пользователь же будет иметь возможность создать приложение, не прибегая к CLI.

2. Конфигурация приложения

Конфигурация REST-приложения представляет собой совокупность контроллеров, каждый из которых объединяет несколько эндпоинтов. Описание эндпоинта содержит:

- путь по которому будут делаться запросы;
- тип запроса;
- параметры пути, строки или тела запроса;
- список параметров, представляющих тело ответа на запрос.

Также элементы конфигурации содержат отображаемые имена, имена которые будут использовать в коде и в именах файлов. Список поддерживаемых типов параметров приведен в таблице.

Таблица

Поддерживаемые типы параметров

Тип параметра	Описание
TEXT	Строковая переменная
NUMBER	Число – как целое, так и с плавающей точкой
BOOL	Логическая переменная (true или false)
DATE	Дата в ISO
ENUM	Перечисление, дополнительно хранит словарь соответствия имени при отправке к отображаемому имени для каждого своего значения
LIST	Список или массив, хранит тип элемента. Многоуровневые списки не поддерживаются
CLASS	Представление для класса, внутри содержит список параметров – внутренних полей

В листинге ниже приводится фрагмент сгенерированной конфигурации.

Листинг

Фрагмент готовой конфигурации в формате JSON

```
"baseUrl": "http://127.0.0.1:5000",
"components": [
  {
    "name": "bookController",
    "endpoints": [
      {
        "name": "Create Book",
        "mapping": "/api/books/book",
        "requestType": "POST",
```

```

    "requestBody": {
      "entityName": "Book",
      "fields": [
        {
          "displayName": "Book Name",
          "codeName": "name",
          "fieldType": "TEXT",
          "required": true
        },
        ...

```

Генераторы конфигурации, использующие рефлексию для автоматического создания описания приложения, были реализованы для фреймворков: SpringBoot (Java), ASP.NETCore(C#) и Flask(Python).

3. Создание интерфейса по заданной конфигурации

Рассмотрим создание Angular проекта по заданной конфигурации, соответствующей формату, описанному в предыдущем разделе. Диаграмма классов этого модуля представлена на рис. 2.

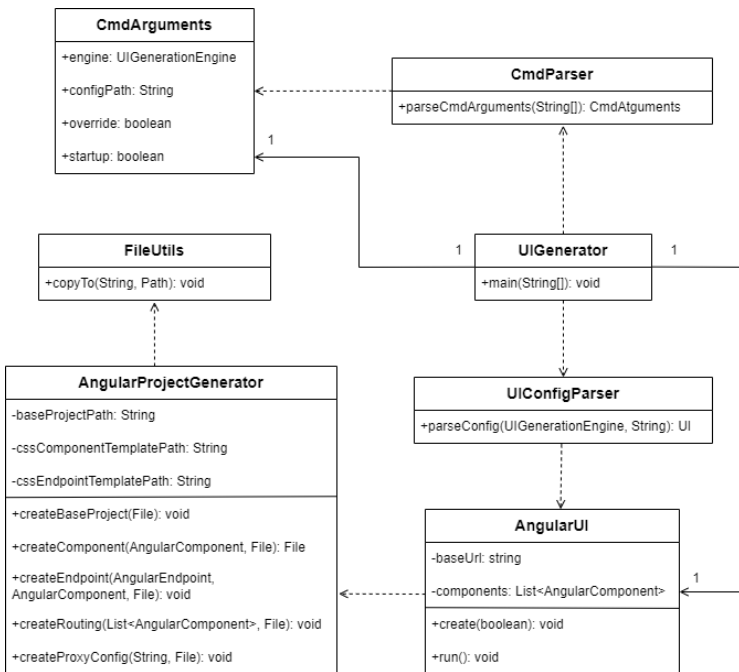


Рис. 2. Диаграмма классов генератора проектов

Поскольку генератор пользовательского интерфейса представляет собой приложения для командной строки, начнем рассмотрение с класса `CmdParser`. Класс использует библиотеку `Apache Commons CLI` для задания и парсинга аргументов командной строки, среди которых:

- `-e, --engine` задает фреймворк, на котором будет построен пользовательский интерфейс. В данный момент поддерживается только `Angular`;

- `-p, --path` задает путь к файлу конфигурации пользовательского интерфейса, является обязательным;

- `-o, --override` – флаг, показывающий нужно ли перезаписывать компоненты в уже сгенерированном проекте при его наличии. Значение по умолчанию – `false`;

- `-s, --startup` – флаг, сигнализирующий о необходимости запуска проекта после его создания с помощью `Angular CLI`, установленного в системе. Значение по умолчанию – `false`.

В результате своей работы парсер возвращает класс `CmdArguments`, содержащие значения параметров, описанных выше. Для описания выбранного фронтенд фреймворка используется перечисление `UIGenerationEngine`.

По этому перечислению выбирается класс пользовательского интерфейса, в нашем случае это `AngularUI`. Для него в классе `UIConfigParser` происходит чтение файла конфигурации по заданному пользователем пути и проводится его десериализация (файл хранит JSON объект). В результате работы создается экземпляр класса `AngularUI`.

Класс `AngularUI` при создании принимает список компонентов, для которых будет создаваться пользовательский интерфейс и URL сервера, который будет использован для отправки к нему запросов.

В классе два метода – `create`, создающий проект (флаг отвечает за то, нужно ли перезаписывать интерфейс, созданный ранее) и `run` – запускающий сгенерированный интерфейс. Для корректной работы последнего метода необходим установленный в системе `Angular CLI`. По завершении работы приложения фронтенд приложение тоже будет закрыто (за это отвечает `shutdown hook`).

`AngularProjectGenerator` ответственен за создание файлов `Angular` проекта. Создаваться они могут двумя способами: копированием независимых от конфигурации файлов из ресурсов (настройки зависимостей, `index.html` и так далее) или рендерингом шаблонов.

Генерация динамических компонентов реализована с помощью обработчика шаблонов `FreeMarker`. Сами шаблоны хранятся в ресурсах библиотеки. По результатам их рендеринга получаются обычные `HTML`

или Typescript файлы, которые будут использоваться Angular проектом [3].

Для корректной работы фреймворка все созданные компоненты необходимо зарегистрировать в главном модуле проекта, поэтому он тоже генерируется на основе шаблона.

Прокси-конфигурация настраивается для корректного взаимодействия с сервером после запуска приложения. Если этого не сделать, механизм CORS будет блокировать любые ответы сервера [4].

4. Пример сгенерированного интерфейса

Все элементы интерфейса выполнены в стиле Material. В верху страницы располагается навигационная панель – каждая кнопка на которой соответствует контроллеру из конфигурации.

Эндпоинт представляет собой сворачиваемую панель. Внутри панели содержатся различные параметры запроса, которые должен ввести пользователь. Под ними располагается поле с ответом сервера, который можно просматривать либо в табличном режиме, либо в режиме JSON. Пример интерфейса эндпоинта представлен на рис. 3.

The screenshot displays a web application interface for a REST client. At the top, there is a blue navigation bar with the text "TestController2" on the left and "Info" on the right. The main content area shows a "POST" request to the endpoint "listEndpoint". Below this, the "Request Body - dto4" section is visible, containing a "description" text field and an "enum_list" section. The "enum_list" section consists of five dropdown menus, each with a red circular icon containing a white square. The first dropdown is labeled "Element 1*" and has the value "Value for shops". The second dropdown is labeled "Element 2*" and has the value "Users". The third dropdown is labeled "Element 3*" and has the value "Book value". The fourth dropdown is labeled "Element 4*" and has the value "Book value". The fifth dropdown is labeled "Element 5*" and has the value "Users". A "Save" button is located at the bottom right of the dropdowns. Below the request body, the "Response" section is visible, showing "No response body" and a "Json" link.

Рис. 3. Пример сгенерированного интерфейса

Заключение

В данной статье было рассмотрено создание системы для автоматической генерации пользовательского интерфейса веб-приложений. В рамках системы был разработан формат конфигурации для описания REST-приложений, созданы библиотеки для генерации конфигурации в 3-х популярных бэкенд фреймворках: SpringBoot, ASP.NET Core и Flask; реализовано построение Angular-проектов по заданной конфигурации.

Список литературы

1. СпецификацияSwagger/OpenAPI [Электронный ресурс]. – Режим доступа: <https://swagger.io/docs/specification>
2. БиблиотекаNG Dynamic Forms[Электронныйресурс]. – Режим доступа: <https://github.com/udos86/ng-dynamic-forms>
3. ДокументацияApacheFreeMarker[Электронныйресурс]. – Режим доступа: <https://freemarker.apache.org/docs/index.html>
4. Фримен, А. Angular для профессионалов / А. Фримен. – Санкт-Петербург: Изд-во Питер, 2018. – 800 с.

Численное решение задачи об изгибе тонкой пластины защемленной по контуру в круговой области

Е. Д. Соболева

Студент бакалавр

М. И. Попов

Доцент

Введение

Тонкие пластины, изготовленные из различных материалов, применяются в строительстве при возведении зданий и промышленных сооружений, например, в качестве железобетонных плит и панелей, металлических настилов, днищ резервуаров. Они применяются также в машиностроении, в качестве конструктивных элементов различных приборов и механизмов. Задачи об изгибе пластин при различных условиях опирания по краям возникают в теории пластин и оболочек [1]. Математическое моделирование данного класса задач приводит к бигармоническому уравнению с разными условиями на границе области. В связи с отсутствием точного аналитического решения таких задач широкое развитие получили численные методы. Одним из наиболее эффективных численных методов является метод конечных разностей. В настоящей работе методом конечных разностей решается задача об изгибе тонкой пластины защемленной по контуру в круговой области.

1. Постановка задачи

Для моделирования изгиба круглой пластины удобно использовать полярные координаты. Начало отсчета координат (полюс) примем в центре срединной плоскости (рис. 1). В общем случае изгиба круглой пластины поперечная нагрузка и все величины, характеризующие напряженное и деформированное состояние пластины, являются функциями двух переменных r и θ [2].

Уравнение изогнутой поверхности в декартовой системе координат, известное также как уравнение Софи Жермена имеет вид:

$$\frac{\partial^4 w}{\partial x^2} + 2 \frac{\partial^4 w}{\partial x^2 \partial y^2} + \frac{\partial^4 w}{\partial y^2} = \frac{q(x, y)}{D}. \quad (1)$$

Здесь $w(x, y)$ – прогиб пластины в точке (x, y) , $q(x, y)$ – поперечная нагрузка, D – цилиндрическая жесткость. Для оператора Лапласа в полярной системе координат справедлива формула:

$$\Delta = \frac{\partial^2}{\partial r^2} + \frac{1}{r} \frac{\partial}{\partial r} + \frac{1}{r^2} \frac{\partial^2}{\partial \theta^2}. \quad (2)$$

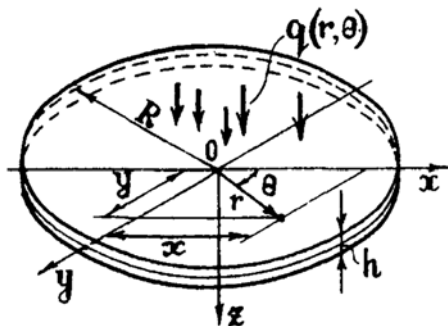


Рис. 1. Круглая пластина

В соответствии с этой формулой дифференциальное уравнение изгиба круглых пластин в полярной системе координат принимает следующий вид:

$$\Delta^2 w(r, \theta) = \frac{q(r, \theta)}{D}, \quad (3)$$

или в развернутом виде

$$\left(\frac{\partial^2}{\partial r^2} + \frac{1}{r} \frac{\partial}{\partial r} + \frac{1}{r^2} \frac{\partial^2}{\partial \theta^2} \right) \left(\frac{\partial^2 w}{\partial r^2} + \frac{1}{r} \frac{\partial w}{\partial r} + \frac{1}{r^2} \frac{\partial^2 w}{\partial \theta^2} \right) = \frac{q(r, \theta)}{D}. \quad (4)$$

На защемленном контуре прогибы и угол наклона касательной к изогнутой срединной плоскости равны нулю:

$$w(R, \theta) = 0, \quad \frac{\partial w}{\partial r}(R, \theta) = 0. \quad (5)$$

2. Понижение порядка уравнения

Уравнение (4) имеет 4 порядок и содержит 8 производных, поэтому конечно-разностная схема для него довольно громоздка. Для упрощения схемы понизим порядок уравнения. Представим уравнение (1) в виде:

$$\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \right) \left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} \right) = \frac{q}{D}, \quad (6)$$

Определим приведенный момент M [3]:

$$M = \frac{M_x + M_y}{1 + \nu} = -D \left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} \right), \quad (7)$$

где M_x, M_y – изгибающие моменты.

Подставив приведенный момент в уравнение (6), получим:

$$\frac{\partial^2 M}{\partial x^2} + \frac{\partial^2 M}{\partial y^2} = -q. \quad (8)$$

Таким образом, уравнение (1) эквивалентно системе 2 уравнений 2 порядка:

$$\begin{cases} \frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} = -\frac{M}{D}, \\ \frac{\partial^2 M}{\partial x^2} + \frac{\partial^2 M}{\partial y^2} = -q. \end{cases} \quad (9)$$

В полярных координатах данная система принимает вид:

$$\begin{cases} \frac{\partial^2 w}{\partial r^2} + \frac{1}{r} \frac{\partial w}{\partial r} + \frac{1}{r^2} \frac{\partial^2 w}{\partial \theta^2} = -\frac{M}{D}, \\ \frac{\partial^2 M}{\partial r^2} + \frac{1}{r} \frac{\partial M}{\partial r} + \frac{1}{r^2} \frac{\partial^2 M}{\partial \theta^2} = -q. \end{cases} \quad (10)$$

3. Аппроксимация системы уравнений и граничных условий

Для построения конечно-разностной схемы, запишем разностные аналоги производных, входящих в систему (10).

Первую производную прогиба по полярному радиусу аппроксимируем со вторым порядком:

$$\left. \frac{\partial w}{\partial r} \right|_{i,j} = \frac{w_{i+1,j} - w_{i-1,j}}{2\Delta r} + O(\Delta r^2). \quad (11)$$

Вторую производную прогиба по полярному радиусу аппроксимируем также со вторым порядком:

$$\left. \frac{\partial^2 w}{\partial r^2} \right|_{i,j} = \frac{w_{i+1,j} - 2w_{i,j} + w_{i-1,j}}{\Delta r^2} + O(\Delta r^2). \quad (12)$$

Вторую производную прогиба по полярному углу аппроксимируем также со вторым порядком:

$$\left. \frac{\partial^2 w}{\partial \theta^2} \right|_{i,j} = \frac{w_{i,j+1} - 2w_{i,j} + w_{i,j-1}}{\Delta \theta^2} + O(\Delta \theta^2). \quad (13)$$

Аналогично поступим с производными приведенного момента. В результате, система дифференциальных уравнений (9) в расчетном узле заменится на систему линейных алгебраических уравнений:

$$\begin{cases} \left. \frac{\partial^2 w}{\partial r^2} \right|_{i,j} + \frac{1}{r_i} \left. \frac{\partial w}{\partial r} \right|_{i,j} + \frac{1}{r_i^2} \left. \frac{\partial^2 w}{\partial \theta^2} \right|_{i,j} = -\frac{M_{i,j}}{D}, \\ \left. \frac{\partial^2 M}{\partial r^2} \right|_{i,j} + \frac{1}{r_i} \left. \frac{\partial M}{\partial r} \right|_{i,j} + \frac{1}{r_i^2} \left. \frac{\partial^2 M}{\partial \theta^2} \right|_{i,j} = -q_{i,j}. \end{cases} \quad (14)$$

Полученная система имеет 2 особенности в полюсе. Во-первых, коэффициенты при первой производной по радиусу и второй производной по углу обращаются в бесконечность. Во-вторых, сама производная по углу в полюсе не определена. Данные особенности возникают за счет перехода к полярным координатам. Чтобы устранить данную проблему, в начале координат будем использовать конечно-разностные уравнения, полученные в декартовых координатах:

$$\begin{cases} \left. \frac{\partial^2 w}{\partial x^2} \right|_0 + \left. \frac{\partial^2 w}{\partial y^2} \right|_0 = -\frac{M_0}{D}, \\ \left. \frac{\partial^2 M}{\partial x^2} \right|_0 + \left. \frac{\partial^2 M}{\partial y^2} \right|_0 = -q_0. \end{cases} \quad (15)$$

Благодаря тому, что мы понизили порядок уравнения, расчетные узлы, входящие в систему, составленную для начала координат, в декартовой и полярной системах совпадают.

Граничное условие равенства нулю прогиба выполняется точно:

$$w_{n,j} = 0. \quad (16)$$

Производную на границе аппроксимируем со 2 порядком:

$$\left. \frac{\partial w}{\partial r} \right|_{n,j} = \frac{3w_{n,j} - 4w_{n-1,j} + w_{n-2,j}}{2\Delta r} = 0. \quad (17)$$

На рис. 2 изображен расчетный шаблон полученной конечно-разностной схемы (14) – (17). Исходя из формул (11)-(13), (17) полученная конечно-разностная схема имеет 2 порядок погрешности.

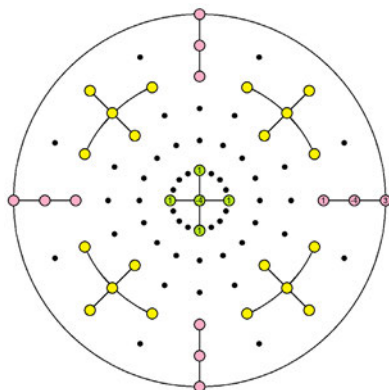


Рис. 2. Расчетный шаблон.

Желтые точки – центрально-разностная аппроксимация оператора Лапласа в полярных координатах для внутренних точек области, зеленые – центрально-разностная аппроксимация оператора Лапласа в декартовых координатах в нуле, розовые – левосторонняя аппроксимация граничных условий.

4. Вычислительный эксперимент

Рассмотрим пластинку радиуса $R = 1$. Подберем тестовую функцию таким образом, чтобы она удовлетворяла следующим условиям:

1. должна выполняться граничные условия,
2. после двукратного применения оператора Лапласа должна получиться непрерывная и ограниченная функция,
3. $w \leq R/20$, данное условие справедливо для тонких пластин.

Всем перечисленным условиям удовлетворяет функция

$$w(r, \theta) = \frac{1}{20} (1 - r^2)^2 r \sin(\theta). \quad (18)$$

$$w(1, \theta) = 0, \quad \frac{\partial w(1, \theta)}{\partial r} = \frac{1}{20} (r^2 - 1)(5r^2 - 1) \sin \theta \Big|_{r=1} = 0 \quad (19)$$

$$\Delta^2 w(r, \theta) = 9.6 r \sin \theta \quad (20)$$

График данной функции изображен на рис. 3. Вычислительные эксперименты показали, что наименьшая абсолютная погрешность достигается при отношении шагов по радиусу к шагам по углу 3:1.

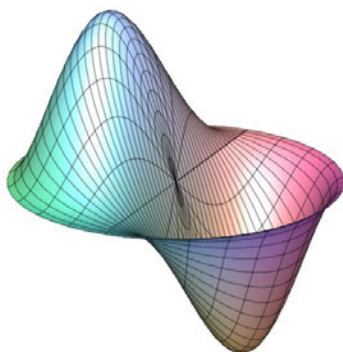


Рис. 3. Тестовая функция.

Наибольшая абсолютная погрешность достигается в точках глобального экстремумов функции. На рис. 4 изображена зависимость максимальной абсолютной погрешности от шага по радиусу.

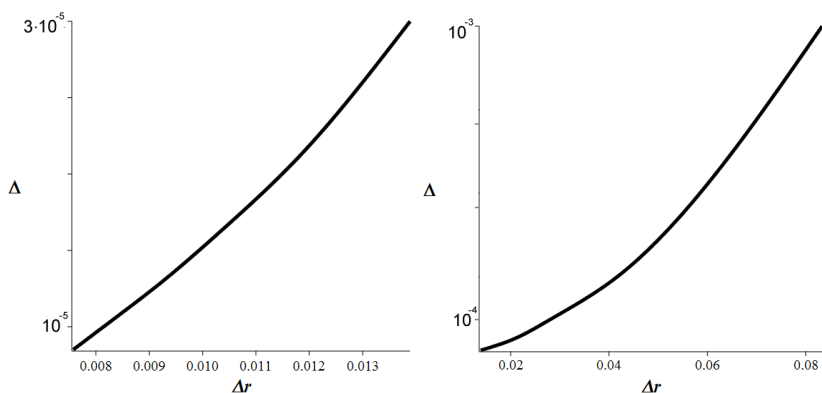


Рис. 4. Зависимость максимальной абсолютной погрешности от шага по радиусу. Слева $\Delta r = 0.0075 - 0.014$, справа $\Delta r = 0.014 - 0.084$

По рисунку мы видим, что погрешность убывает с квадратом шага по радиусу, что полностью согласуется с теоретической оценкой для порядка погрешности.

Заключение

В данной работе получено численное решение задачи об изгибе тонкой пластины, защемленной по контуру в круговой области.

Понижен порядок уравнения, за счет введения функции приведенного момента. Разработана неявная конечно-разностная схема второго порядка точности в полярных координатах. Достоинствами данной схемы является использование минимального количества узловых точек для достижения заданной точности, а также возможность применения в областях, содержащих начало координат. К недостатком следует отнести удвоенное количество уравнений, вследствие необходимости вычисления приведенного момента. Проведены численные эксперименты для тестовой функции. Вычислено оптимальное соотношение между шагами по радиусу и по углу. Полученные расчеты абсолютной погрешности полностью согласуются с теоретическими оценками.

Список литературы

1. Численные методы решения задач об изгибе квадратной пластины / М. Ю. Смирнов, М. И. Попов, А. В. Скрыпников [и др.] // Вестник Поволжского государственного технологического университета. Серия: Материалы. Конструкции. Технологии. – 2021. – № 3. – С. 111-124.
2. Сопротивление материалов с основами теории упругости и пластичности : [Учеб. для строит. спец. вузов] / Г. С. Варданян [и др.] ; Под ред. Г. С. Варданяна. – М. : Изд-во Ассоц. строит. вузов, 1995. – 572 с.
3. Монахов В. А. Теория пластин и оболочек: учеб. пособие по направлению подготовки 08.05.01 «Строительство уникальных зданий и сооружений» / В. А. Монахов. – Пенза : ПГУАС, 2016. – 252 с.

Разработка веб-приложения для проведения экспертизы работ регионального проекта-преакселератора инноваций ФГБОУ ВО «ВГУ» «Лига инноваций»

Я. В. Солодовникова

Студент бакалавр

Е. А. Копытина

Старший преподаватель

Введение

«Лига инноваций» – первый региональный проект-преакселератор ФГБОУ ВО «ВГУ», направленный на формирование базовых компетенций в области инновационной предпринимательской деятельности у обучающихся образовательных организаций Воронежской области [1]. Одним из этапов «Лиги инноваций» является конкурс проектов, которые участники разработали на основе полученных компетенций. Во время конкурса эксперты должны по ряду определенных критериев оценить проект и сделать свое заключение. После этого экспертные оценки переносятся в электронный формат, выполняется их обработка и построение рейтинга участников.

Использование бумажных носителей создает лишние этапы обработки информации, которые замедляют получение результата (в данном случае рейтинга участников), а также повышают вероятность ошибки. Данная проблема крайне актуальна в настоящее время, так как проект «Лига инноваций» активно развивается, и решению подобных проблем посвящены работы [2-6].

Для устранения вышеописанных сложностей разрабатывается веб-приложение, в котором эксперты могут выставлять свои оценки в соответствии с существующими критериями, а организаторы «Лиги инноваций» просматривать автоматически сформированный рейтинг.

1. Постановка задачи

К описываемому в статье веб-приложению были выдвинуты следующие требования:

- Аутентификация экспертов и организаторов;
- Возможность эксперта добавлять, изменять и просматривать экспертные заключения;

- Возможность организатора управлять проектами;
- Возможность организатора управлять критериями и их категориями;
- Возможность организатора просматривать автоматически сформированный рейтинг;
- Возможность организатора управлять аккаунтами экспертов и других организаторов.

2. Реализация логики

Веб-приложение имеет клиент-серверную архитектуру. Клиентская и серверная части взаимодействуют между собой с помощью HTTPs-запросов.

Сервер работает по принципу REST API. Серверная часть веб-приложения написана на языке Java, и для её реализации был использован фреймворк Spring Boot.

Структура сервера соответствует стандартной структуре для Spring Boot проектов:

- Controller классы отвечают за обработку запросов;
- Service классы реализуют функционал приложения;
- Entity классы представляют сущности предметной области;
- DTO классы используются для взаимодействия с клиентской частью;
- Repository классы взаимодействуют с базой данных.

Для серверной части приложения была выбрана архитектура MVC (Model-View-Controller). За компонент Model отвечают Service и Entity классы. За компонент Controller отвечают Controller классы, которые обрабатывают входящие запросы. Компонент View представлен исходящими от сервера JSON ответами, которые потом обрабатываются на клиентской части приложения.

Логика работы сервера разделена между следующими 7 объектами, полученными при анализе предметной области:

- Организатор (Admin);
- Категория критериев (Category);
- Критерий (Criteria);
- Эксперт (Expert);
- Экспертное заключение (Opinion);
- Проект (Project);
- Рейтинг (Rating).

Клиентская часть веб-приложения написана на языке JavaScript с использованием фреймворка React.js.

Структура клиентской части соответствует стандартной структуре для React-приложения:

- В корневом каталоге хранятся файлы index.js и App.js, представляющие собой главный компонент приложения;
- В папке src/components хранятся компоненты интерфейса, например, панель навигации (Header.js);
- В папке src/pages хранятся страницы сайта;
- В папке public хранятся файлы, доступные публично, например, index.html, который является корневым HTML-шаблоном приложения.

Для настройки внешнего вида веб-приложения используются CSS стили, которые представлены в едином для всех страниц файле Page.css.

3. Реализация интерфейса

Работа пользователя в веб-приложении начинается с авторизации. Перед вводом логина и пароля необходимо обязательно выбрать роль эксперта или организатора, для осуществления входа в систему под соответствующей ролью. Вид страницы авторизации представлен на рис. 1.

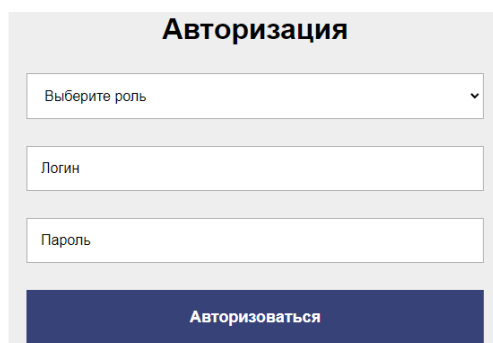


Рис. 1. Страница авторизации пользователя

Если выполнить авторизацию с ролью эксперта, будет осуществлен переход в меню эксперта. В данном меню эксперт может добавить, изменить или просмотреть сделанные им экспертные заключения, а также осуществить выход из аккаунта. Вид страницы с меню эксперта представлен на рис. 2.

При нажатии на кнопку «Добавление экспертного заключения» будет осуществлен переход на страницу добавления экспертного заключения. Аналогично для страниц изменения и просмотра заключений. Вид страницы добавления экспертного заключения представлен на рис. 3.

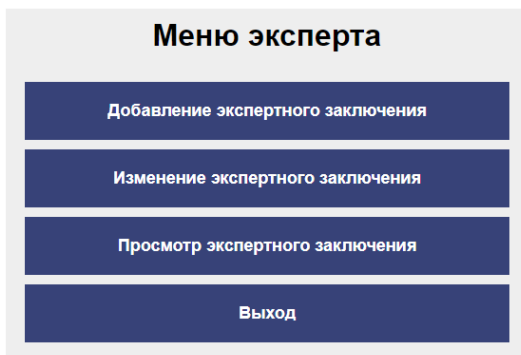


Рис. 2. Страница меню эксперта



Рис. 3. Страница добавления экспертного заключения

Если выполнить авторизацию с ролью организатора, будет осуществлен переход в меню организатора. На данной странице организатор может управлять проектами, критериями, аккаунтами экспертов и организаторов, просматривать рейтинг и экспертные заключения, а также осуществить выход из аккаунта. Вид страницы с меню организатора представлен на рис. 4.

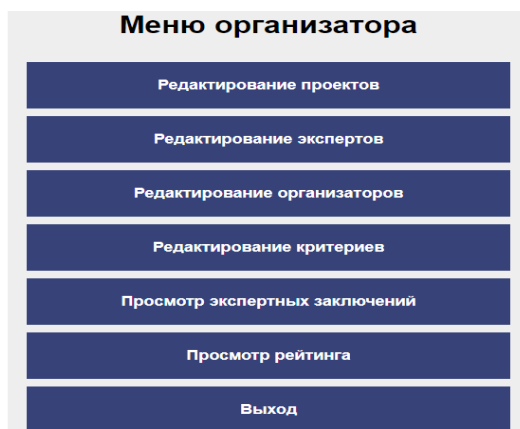


Рис. 4. Страница меню организатора

При нажатии на кнопку «Просмотр рейтинга» будет осуществлен переход на страницу просмотра рейтинга. Аналогично для кнопок управления аккаунтами экспертов и организаторов, проектов, критериев и просмотра экспертных заключений. На странице просмотра рейтинга необходимо выбрать год участия проектов, и таблица с рейтингом заполнится автоматически. Вид страницы для просмотра рейтинга представлен на рис. 5.

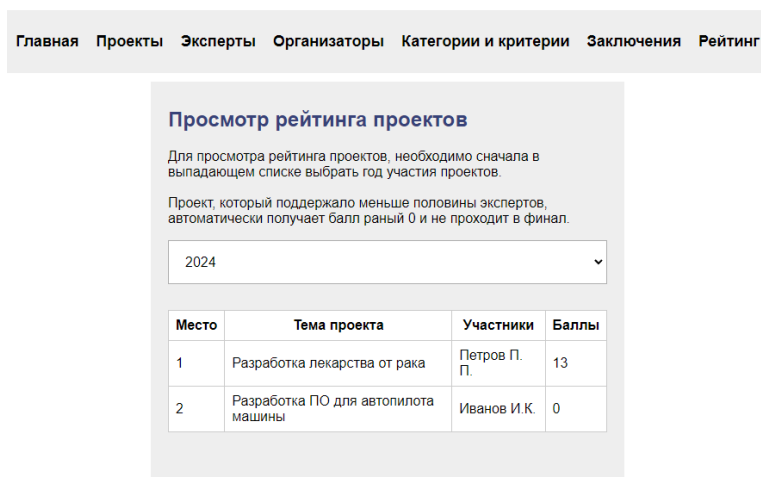


Рис. 5. Страница просмотра рейтинга

Заключение

В данной статье была рассмотрена разработка веб-приложения проведения экспертизы работ регионального проекта-преакселератора инноваций ФГБОУ ВО «ВГУ» «Лига инноваций». Все поставленные в начале разработки приложения функциональные требования были выполнены. Приложение является востребованным и полностью работоспособным.

Список литературы

1. Лига инноваций [Электронный ресурс]: страница конкурса. – Режим доступа : <https://ligavsu.ru/>
2. Кропачев, А. Н. Формирование отчета деятельности наркоконтроля по Воронежской области средствами офисных приложений / А. Н. Кропачев, Е. А. Копытина // Труды молодых учёных факультета компьютерных наук ВГУ : Сборник статей / Под редакцией Д.Н. Борисова. Том Выпуск 1. – Воронеж : Общество с ограниченной ответственностью "Вэлборн", 2021. – С. 80-84. – EDN IUBYNI.
3. Копытина, Е. А. Разработка веб-сервиса для формирования горячего коммерческого предложения ООО «ЧерноземАгроماش» / Е. А. Копытина, Ф. А. Десятириков, А. В. Копытин // Информатика: проблемы, методы, технологии : Материалы XXII Международной научно-практической конференции им. Э.К. Алгазинова, Воронеж, 10–12 февраля 2022 года / Под редакцией Д.Н. Борисова. – Воронеж: Общество с ограниченной ответственностью "Вэлборн", 2022. – С. 990-996. – EDN SXFFFI.
4. Погорелов, Р. И. Адаптация тиражируемого решения от фирмы 1С (1С: Управление нашей фирмой) и разработка приложения на мобильной платформе 1С / Р. И. Погорелов, А. В. Копытин, Е. А. Копытина // Труды молодых учёных факультета компьютерных наук ВГУ : Сборник статей / Под редакцией Д.Н. Борисова. Том Выпуск 1. – Воронеж : Общество с ограниченной ответственностью "Вэлборн", 2021. – С. 121-125. – EDN EOZHFH.
5. Претков, А. И. Автоматизации формирования документов основных профессиональных образовательных программ высшего образования факультета компьютерных наук Воронежского государственного университета / А. И. Претков, Е. А. Копытина // Труды молодых учёных факультета компьютерных наук ВГУ : Сборник статей. – Воронеж : Воронежский государственный университет, 2023. – С. 94-107. – EDN HKXMIS.
6. Шорохова, Т. А. Разработка проекта «List_of_documents» для реализации программ дополнительного образования ФГБОУ ВО «Воронежского государственного университета» / Т. А. Шорохова, Е. А. Копытина // Труды молодых учёных факультета компьютерных наук ВГУ : Сборник статей. – Воронеж : Воронежский государственный университет, 2023. – С. 284-289. – EDN MMGXTK.

Анализ производительности бизнес-процессов платформы Camunda

П. О. Федосова

Студент бакалавр

Н. К. Самойлов

Старший преподаватель

Введение

В современном мире информационных технологий и автоматизации бизнес-процессов существует постоянная потребность в эффективном анализе и оптимизации рабочих потоков. В этом контексте инструменты управления процессами, такие как платформа Camunda, становятся ключевыми элементами для компаний, стремящихся к повышению производительности и качества своей деятельности. Важно иметь возможность не только отслеживать и контролировать ход бизнес-процессов, но и анализировать их эффективность.

В данной работе рассматривается вариант реализации библиотеки для фреймворка Spring Boot с целью анализа процессов бизнес-логики с использованием платформы Camunda, которая предлагает набор инструментов и сервисов для разработки, выполнения, автоматизации и мониторинга бизнес-процессов. Конечным результатом работы данной библиотеки будет построение тепловой карты процесса, позволяющей визуализировать и анализировать данные о процессах бизнес-логики.

Тепловые карты представляют собой графическое представление данных, где различные цвета указывают на уровень интенсивности или значимости определенных факторов. Данный вид отчета помогает в анализе производительности процессов, выявлении узких мест, определении областей, требующих улучшений. Тепловые карты могут быть созданы с использованием исторических данных о выполнении процессов, которые собираются и хранятся в Camunda, отображая необходимую информацию (время выполнения каждого шага процесса, частота его использования, количество ошибок).

1. Camunda

Camunda – платформа автоматизации бизнес-процессов и принятия решений [1]. Данная система поставляется с инструментами для

создания моделей, их эксплуатации и предоставления пользователям возможности выполнять назначенные им задачи.

Основными достоинствами Camunda является снижение затрат на автоматизацию бизнес-процессов, что подразумевает следующие преимущества:

- поддержка BPMN-нотации для моделирования бизнес-процессов, с помощью которой можно описать бизнес-логику любой сложности [2];
- поддержка любого JVM-язык – Java, Kotlin и так далее;
- организованная внутренняя архитектура обеспечивает функционирование системы в соответствии с очевидными и предсказуемыми принципами;
- удобство разработки, тестирования и встраивания в CI/CD за счёт того, что Camunda можно использовать как библиотеку в Java-приложении;
- открытый исходный код и понятная документация.

Для данной работы Camunda интегрировалась как библиотека в проект Spring Boot и настроить базу данных, в данном случае PostgreSQL. Также требовалось создать тестовый процесс с помощью Camunda Modeler [3].

2. Схема базы данных

После запуска проекта Camunda самостоятельно генерирует в базе данных таблицы. Для создания тепловой карты будут необходимы только некоторые из них, а именно (рис. 1):

- «act_ge_bytearray» хранит бинарные данные, такие как файлы и документы, связанные с процессами (например, xml код BPMN диаграммы бизнес-процесса [4]);
- «act_hi_actinst» хранит исторические данные о выполнении действий в рамках процессов, а именно их имя, время начала и окончания выполнения, тип и так далее.
- «act_re_procdef» предназначена для хранения информации об определениях бизнес-процессов. Эти данные включают основные атрибуты процесса, его версии, а также сведения о ресурсах, используемых для определения и визуализации процесса.

Эти таблицы в совокупности позволяют анализировать и визуализировать производительность бизнес-процессов, способствуя их оптимизации и улучшению.

- Service – бизнес-логика, обработка данных и взаимодействие с репозиториями;
- Repository – абстракция для доступа к данным в базе данных.

Визуализации тепловой карты происходит на клиенте, который получает необходимые данные сервера через REST API.

4. Тестовые данные

Для демонстрации работы сервиса использован простой и наглядный тестовый пример (рис. 3), который описывает процесс подтверждения заказа, начинающийся с получения заказа и включающий в себя ряд последующих проверок и действий:

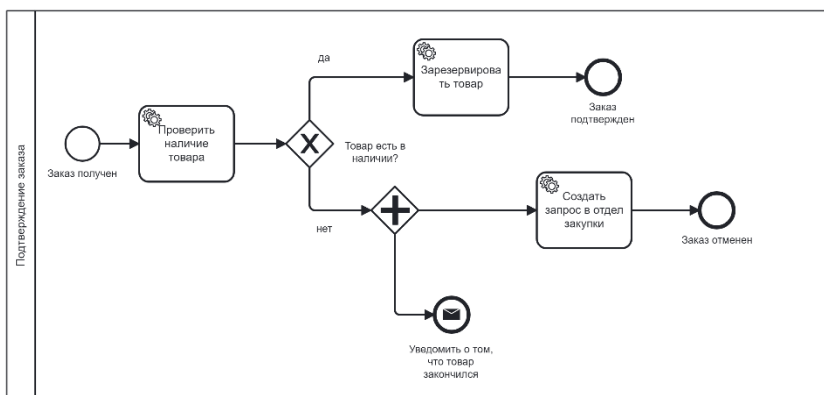


Рис. 3. Схема тестового бизнес-процесса

1. Стартовое событие «Заказ получен» – начало процесса и получение артикула и количества заказываемого товара.
2. Сервисная задача «Проверить наличие товара» проверяет наличие товара.
3. Исключающий шлюз «Товар есть в наличии?» представляет условие: если товар есть в наличии, процесс продолжается сервисной задачей "Зарезервировать товар", если нет – "Создать запрос в отдел закупок".
4. Параллельный шлюз – если товара нет в наличии, процесс расходится на два пути. Для задачи «Создать запрос в отдел закупок» создается запрос для пополнения запасов, а в конечном событии «Уведомить, что товар закончился» заказчик получает уведомление об отсутствии товара.
5. Конечное событие «Заказ отменен» – исход при отсутствии товара – заказ отменяется.

6. Задача «Зарезервировать товар» (если товар есть в наличии) – сервисная задача резервирования товара для выполнения заказа.
7. Конечное событие «Заказ подтвержден» – после успешного резервирования товара заказ считается подтвержденным.

5. Результаты работы

С помощью реализованного в рамках данного проекта сервиса была построена тепловая карта тестового процесса, отражающая данные о среднем времени работы каждого элемента процесса (рис. 4).

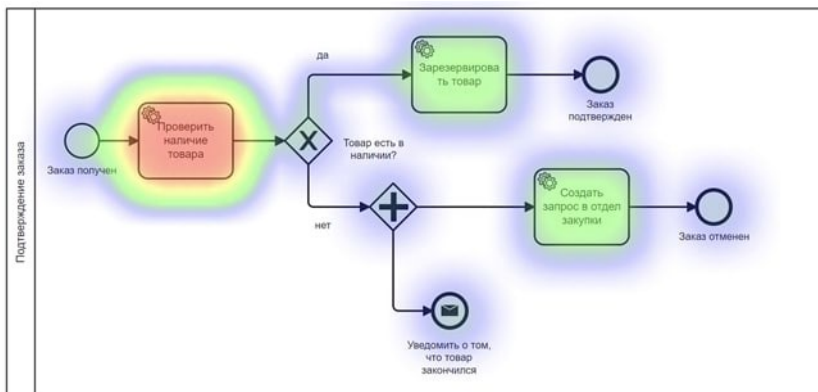


Рис. 4. Тепловая карта тестового процесса

На данном рисунке красным цветом отображаются наиболее медленные процессы, а зеленым – наиболее быстрые. Можно заметить, что самый медленный процесс – сервисная задача проверки наличия товаров.

Каждая сервисная задача в BPMN связана с методом в коде через Java Delegate (интерфейс Spring Boot) с помощью атрибута «delegateExpression» (характеристика xml-объекта) [8], который указывает на компонент, содержащий необходимую бизнес-логику. Код делегирования позволяет выполнять код или оценивать выражения, когда во время выполнения процесса происходят определенные события [9].

Задача «Проверить наличие товара» ссылается на класс CheckProductDelegate, который отвечает за логику проверки наличия товара в базе данных. Для этого он взаимодействует с сервисом ProductService, конкретно с методом checkProduct.

Данный метод сначала извлекает все товары из базы данных, а затем использует потоки java для фильтрации по номеру и количеству.

Такой подход довольно неэффективен, так как требует загрузки всех записей в память перед фильтрацией, что снижает производительность при больших объёмах данных и увеличивать время обработки данных, что и было отражено на тепловой карте.

После улучшения данного метода путем использования возможностей JPA-репозитория для непосредственного запроса к базе данных, производительность возросла, так как операция фильтрации происходит на уровне базы данных, а тепловая карта стала выглядеть следующим образом (рис. 5).

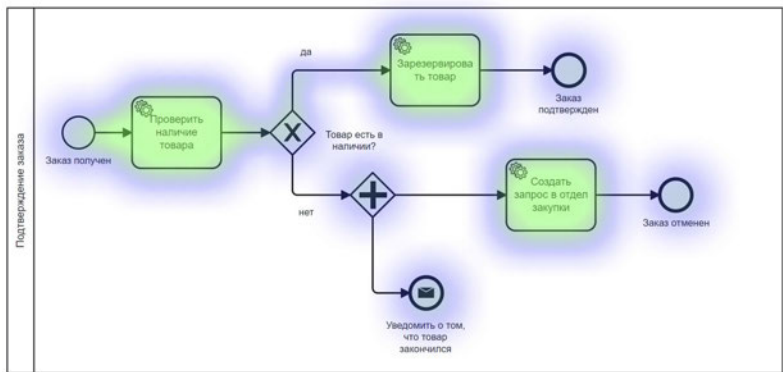


Рис. 5. Тепловая карта тестового процесса после улучшения кода метода проверки наличия товара

Можно заметить, что все элементы стали подсвечиваться зеленым цветом, то есть время работы стало минимальным.

В качестве доказательства корректности работы тепловой карты были проведены измерения среднего времени работы метода проверки наличия товара для разного количества обрабатываемых данных. Результаты представлены в таблице.

Таблица

Результаты измерений времени работы вариантов кода

Количество данных	Время работы улучшенного кода (мс)	Время работы первоначального кода (мс)
1 млн	1.14	3×10^3
5 млн	7.96	24×10^3
10 млн	9.86	40×10^3

Таким образом, улучшенный метод обрабатывает данные за время, не превышающее одну секунду, в то время как первоначальный код требует в среднем 3 секунды для обработки минимального количества данных (1 млн).

Помимо отчетов о среднем времени работы, в сервисе есть возможность построить тепловую карту, которая отражает данные о частоте выполнения каждого элемента процесса (рис. 6).

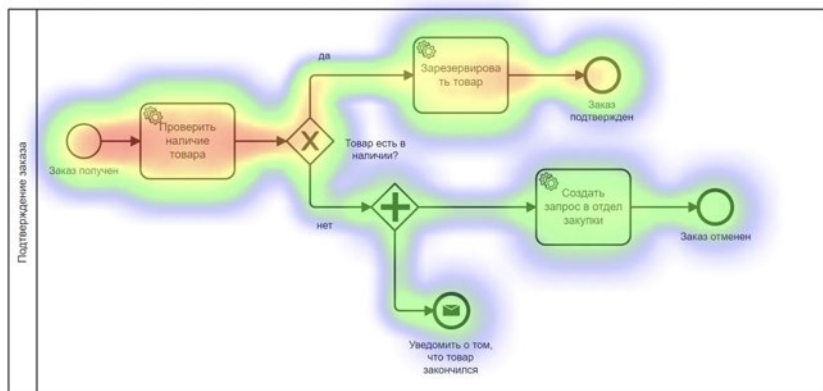


Рис. 6. Тепловая карта тестового процесса по количеству работы элементов

Исходя из результатов построения тепловой карты, можно сказать, что чаще всего выполнялось действие проверки наличия товара, что очевидно, ведь это обязательное действие в любом сценарии системы. Следующим по количеству отработок идет сервисная задача резервирования заказа, из чего можно сделать вывод, что в большем количестве случаев товар имелся в наличии.

Заключение

Результатами данного исследования являются:

- изучение стандарта BPMN и использование его в описании бизнес-процессов;
- исследование платформы Camunda и анализ сгенерированной ею базы данных;
- создание тепловой карты процессов, которая визуализирует различные аспекты выполнения процессов, такие как время выполнения каждого шага и частоту использования;
- создание тестового бизнес-процесса;

- оценка работоспособности реализованной библиотеки на платформе Camunda в контексте фреймворка Spring Boot.

Таким образом, полученные данные помогают выявлять малоэффективные и узкие места бизнес-процессов и определять области для потенциальных улучшений.

Применение такой технологии позволяет повысить производительность процессов за счет точной диагностики и последующей оптимизации. После внедрения улучшений в код, тепловая карта тестового процесса показала значительное уменьшение времени выполнения операций, что подтверждается количественными данными.

Список литературы

1. Rucker, B. Practical Process Automation / B. Rucker. – 1st ed. – South Carolina, USA : O'Reilly Media, 2021. – 292 p.
2. Dijkman, R. M. Semantics and analysis of business process models in BPMN / R. M. Dijkman, M. Dumas, C. Ouyang // Information and Software Technology. – 2008. – Iss. 12. – P. 1281-1294.
3. The Camunda 7 Manual [Электронный ресурс] : сайт. – Режим доступа: <https://docs.camunda.org/manual/7.20>
4. Suarez, G. N. Best Practice Guidelines for BPMN 2.0 / G. N. Suarez, J. Freund, M. Schrepfer // BPMN 2.0 Handbook First Edition: Foreword by Bruce Silver. – 2010. – Iss. 1. – P. 153-167.
5. Spring Boot Reference Documentation [Электронный ресурс] : сайт. – Режим доступа: <https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle>
6. PostgreSQL 14.8 Documentation [Электронный ресурс] : сайт. – Режим доступа: <https://www.postgresql.org/docs/14>
7. Clickhouse Docs [Электронный ресурс] : сайт. – Режим доступа: <https://clickhouse.com/docs>
8. JDK 17 Documentation [Электронный ресурс] : сайт. – Режим доступа: <https://docs.oracle.com/en/java/javase/17>
9. Freund, J. Real-Life BPMN: Using BPMN, CMMN and DMN to Analyze, Improve and Automate Processes in Your Company / J. Freund, B. Rucker – 3rd ed. – South Carolina, USA : CreateSpace Independent Publishing Platform, 2016. – 248 p.

Архитектура Apache Kafka

С. А. Черникова

Студент бакалавр

Н. К. Самойлов

Старший преподаватель

Введение

В 2010 году инженеры компании LinkedIn разработали специальную платформу Kafka, которая позволяла обрабатывать большое количество данных в режиме реального времени. Постепенно она завоевала популярность благодаря своей масштабируемости, высокой пропускной способности и устойчивости к отказам.

В 2012 году ее передали компании Apache Software Foundation. Так платформа превратилась в Apache Kafka. Сейчас она используется во многих отраслях: электронной коммерции, банковском деле, телекоммуникации и других.

1. Введение в Apache Kafka

Представим систему, в которой один или несколько участников генерируют сообщения, а другие – их получают. Данная система представлена на рис. 1. Apache Kafka называет того, кто генерирует и передает сообщения, *producer* (продюсер). А того, кто принимает, – *consumer* (консьюмер) [1].

Чтобы объединить продюсеров и консьюмеров, используется *broker* (брокер), внутри которого находится *topic* (топик). Обычно их несколько, чтобы повысить отказоустойчивость системы. Внутри брокеров – *topic* (топики), использующиеся для регламентирования взаимодействия продюсеров и консьюмеров. Это логическая единица, которая представляет собой очередь сообщений. Именно туда отправляют свои данные продюсеры, а консьюмеры подписываются на топики и периодически читают оттуда эти данные.

Таким образом, Apache Kafka состоит из четырех основных частей: топик, брокер, продюсер и консьюмер.

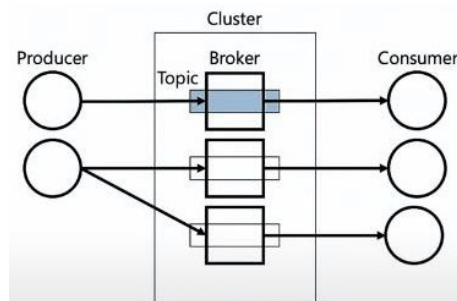


Рис. 1. Схема работы Apache Kafka

2. Topic и partition

Топики состоят из частей – partition (партиции).

Партицию можно рассматривать как единицу параллелизации с точки зрения записи данных, так как данные в партиции пишутся независимо.

События всегда дописываются в конец партиции. У каждого события есть свой порядковый номер или offset. Нумерация начинается с 0.

Физически партиции хранятся на диске в виде сегментов, представленных на рис. 2.

Запись событий всегда происходит в последний сегмент – активный. При этом каждый сегмент начинается с какого-то события, которое называется base offset.

$\text{partition} = \{\text{segment}\}$



Рис. 2. Логическое представление partition

У партиции есть две характеристики: retention.bytes и retention.ms.

Первая отвечает за вместимость партиции в байтах. По умолчанию она имеет значение -1 – это значит, что размеры партиции ограничены только размером диска, на котором она хранится.

Вторая отвечает за время хранения данных и по умолчанию имеет значение 604800000 мс, то есть 7 дней. По истечении этого срока удаляется весь сегмент.

Описание сегмента хранится на диске в виде трех файлов. Они имеют одинаковое название, в котором содержится номер base offset.

Все данные хранятся в файле с расширением .log. Добавление новых данных в сегмент происходит путем записи их в конец данного файла.

Для быстрого и удобного поиска данных используется файл с расширением .index или .timeindex.

Файл .index выглядит следующим образом. Каждая запись в нем занимает 8 байт. Там два int: relative offset, position. Relative offset – это смещение от начала сегмента (число, полученное вычитанием из offset сообщения базового offset). Position – это физическое смещение данного сообщения в лог-файле.

Сообщений в партии могут быть миллиарды, соответственно базовый offset будет большой и возникнет переполнение в int. Но сегмент небольшой, поэтому относительный offset может быть записан с помощью int.

В .timeindex хранится время записи базового офсета, а схема поиска такая же, как в случае с файлом .index.

3. Kafka Broker и replication factor

На рис. 3 представлен Kafka Cluster. Кластер – это множество брокеров. Один из них отвечает за контроллер, он координирует работу кластера.

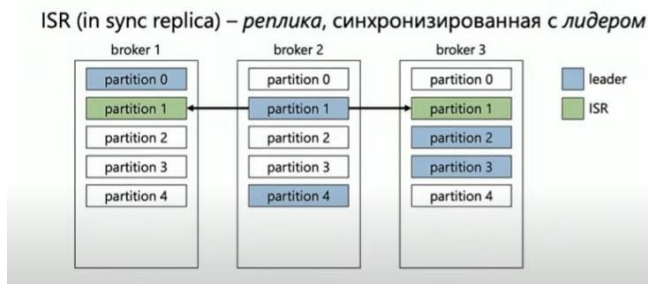


Рис. 3. Логическое представление Kafka Broker

Для увеличения надежности хранения данных используется replication factor – это количество реплик, созданных для каждой партии в топике.

Каждая партиция должна иметь лидера – это тот брокер, который отвечает за запись в конкретную партицию. Сама же реплика, в которую пишут продюсеры, называется лидер-реплика.

За выбор лидера отвечает контроллер. При этом Kafka старается равномерно распределять по кластеру лидеров и балансировать, чтобы не получилось так, что на один брокер слишком большая нагрузка, а остальные простаивают.

Данные, которые пишутся в лидера, должны быть распределены по репликам. Те, кто в себя копируют данные из лидера, называются follower. При этом часто используются in sync replica – это фолловеры, которые синхронизированы с лидером, и в идеале весь кластер должен быть синхронизирован.

Если какой-то из лидеров выходит из строя, то контроллер выбирает нового лидера среди реплик и система не останавливает свою работу, а данные не теряются.

4. Producer и Consumer

Теперь посмотрим, как работает продюсер[2]. В Kafka есть acknowledgement, т. е. подтверждение записи. Продюсер должен убедиться, что данные записались.

Нулевой уровень предоставляет нулевые гарантии, т.е продюсер не ждет никакого подтверждения. При этом данные могли записаться в Kafka, а могли не записаться.

Уровень гарантии 1 означает, что продюсер после отправки данных ждет подтверждения от брокера, что тот точно сохранил данные в лидер-реплику. При этом фолловеры не обязательно могут получить эти данные.

Уровень all означает, что продюсер после отправки данных ждет, когда брокер ему сообщит, что данные записались не только в лидер-реплику, но и во всех ее фолловеров. И тут появляется важная настройка min.insync.replicas [3] – минимальное количество реплик, в которые должны быть сохранены данные. Только после того, как данные будут получены минимальным количеством реплик, брокер сообщит продюсеру, что запись прошла успешно.

Само сообщение можно представить как пару ключ-значение. При этом для Kafka и ключ, и значение – это обычный массив байтов, в который можно писать все, что угодно. Продюсер должен знать, что он туда сериализует и consumer должен знать, что он оттуда десериализует.

Ключ используется для определения номера партии, куда положить данные. Используется MurmurHash в том случае, если ключ есть (ключ и значение могут быть null). Если ключа нет, то используется round robin, когда продюсер перебирает партии по кругу.

Теперь посмотрим, как работает consumer. У нас есть несколько партий, представленных на рис. 4.

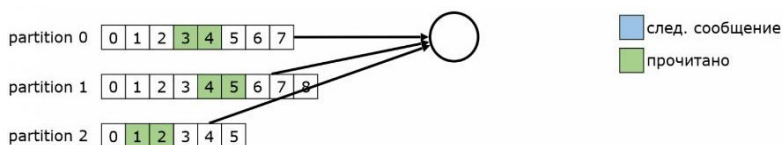


Рис. 4. Чтение сообщений консьюмером

Консьюмер читает порционно сразу из нескольких партиций, запоминая для каждой место, на котором он остановился. Но в его работе может произойти сбой и эта информация потеряется. Для таких случаев в Kafka есть `commit offset`.

Значение, с которого надо начинать чтение, консьюмер коммитит – сообщает контроллеру офсет данного сообщения.

Консьюмеры обладают свойством масштабирования, то есть с ростом нагрузки в системе можно либо увеличивать их количество, либо объединить их в группу с общим идентификатором, что более предпочтительно. Тогда при работе они автоматически распределят все партиции между собой.

Если один из консьюмеров в группе останавливает свою работу, не успевая закомитить прочитанные данные, то любой другой консьюмер возьмет на себя партицию и начнет читать с первого сообщения. После восстановления приостановивший работу консьюмер может снова вернуться в группу и продолжить чтение.

Заключение

Данная статья посвящена описанию архитектуры Apache Kafka. Благодаря тому, что ее серверы объединяются в кластеры, хранение и пересылка сообщений идет параллельно на разных серверах, а это дает большую надежность и отказоустойчивость. Даже при выходе из строя нескольких машин, сообщения все еще будут пересылаться и обрабатываться.

Также сервис легко масштабируется горизонтально. То есть, для наращивания мощности Apache Kafka достаточно вводить в строй дополнительные серверы.

Еще один плюс – консистентность данных. Записи в Apache Kafka хранятся в виде журнала коммитов. Это выглядит как очередь сообщений, в которую можно добавлять записи, а вот удалять или модифицировать – нет. Такой подход дает огромную надежность и простоту изменения любых состояний – всегда понятно, что, как и в какой последовательности менялось.

Список литературы

1. Apache Kafka: что это и как работает [Электронный ресурс] : статья про Apache Kafka. – Режим доступа: <https://cloud.vk.com/blog/apache-kafka-chto-eto-i-kak-rabotaet>
2. Григорий Кошелев – А вы Кафку пробовали? [Электронный ресурс] : лекция от Григория Кошелева про Kafka. – Режим доступа: <https://habr.com/ru/articles/537846/>
3. "Perfect number" to choose as Kafka's ultimate replication factor.[Электронный ресурс] : статья про минимальное количество репликаций в Kafka. – Режим доступа: <https://www.cloudkarafka.com/blog/apache-kafka-replication-factor-perfect-number.html>

Алгоритмы шардирования и репликации для хранилищ данных

Р. С. Шевцов

Студент бакалавр

Н. К. Самойлов

Старший преподаватель

Введение

Распространение технологий за последние два десятилетия приняло по истине колоссальный характер. Большим технологическим фактором является достижения в сетевой области, которые позволяют пользователям получать информацию и сервисы, богатые данными, в любое время. Поскольку сетевые провайдеры продолжают совершенствовать возможности своих беспроводных и широкополосных инфраструктур, эта парадигма продолжает стимулировать изобретение новых и оригинальных сервисов, которые упрощают и обогащают профессиональную и личную жизнь конечных пользователей, приводя при этом к порождению и обмену огромных объемов данных.

Выход из сложившейся ситуации удалось найти в масштабировании. Необходимость проектирования, таким образом, заключается в обеспечении возможности увеличения емкости системы за счет добавления дополнительных аппаратных ресурсов всякий раз, когда это оправдано колебаниями нагрузки. Таким образом, масштабируемость стала как важнейшим требованием, так и фундаментальной проблемой в контексте облачных вычислений. В контексте же баз данных были разработаны методы шардирования и репликации, призванные стабилизировать сложившуюся ситуацию.

1. Шардирование

Первым рассмотренным механизмом масштабирования будет шардирование – шаблон архитектуры базы данных, связанный с практикой разделения строк одной таблицы на несколько разных таблиц, известных как шарды. Каждый из них имеет одинаковую схему и столбцы, но также и совершенно разные строки. Аналогично, данные, хранящиеся в каждом шарде, уникальны и не зависят от данных, хранящихся в других. Шардирование предполагает разбиение данных на

два или более небольших фрагмента, называемые логические разделы. Затем они распределяются по отдельным узлам базы данных, называемые физическими разделами, которые могут содержать несколько логических. Несмотря на это, данные, хранящиеся во всех шардах, представляют собой целый набор логических данных[1].

Шарды базы данных представляют собой архитектуру без общего доступа. Это означает, что они автономны, не используют одни и те же данные или вычислительные ресурсы. Однако в некоторых случаях имеет смысл скопировать определенные таблицы в каждый шард, чтобы они служили справочными таблицами. Например, предположим, что есть база данных для приложения, которая зависит от фиксированных коэффициентов пересчета для измерения веса. Реплицируя таблицу, содержащую необходимые данные в коэффициенте конверсии в каждый шард, гарантирует нахождение данных.

Общая схема шардирования представлена на рис. 1.

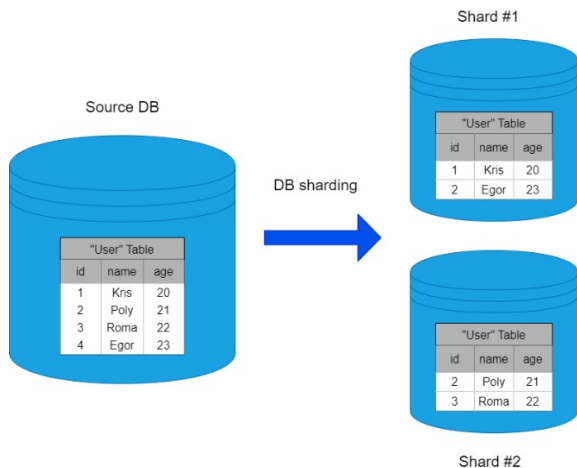


Рис. 1. Пример шардирования базы данных

Часто шардирование реализуется на уровне приложения, что означает, что приложение включает в себя код, который определяет, какой сегмент передавать на чтение и запись. Тем не менее некоторые СУБД имеют встроенные возможности шардинга, что позволяет применять его непосредственно на уровне базы данных.

2. Алгоритмы шардирования

Первым рассматриваемым алгоритмом будет Mod – один из методов горизонтального масштабирования баз данных. Механизм шардирования базируется на применении принципа остатка от деления ключа записи на количество шардов, позволяющий определить узел, на который будет отправлен запрос [2].

Алгоритм получил большое распространение для так называемых NoSQL, или документоориентированных баз, где необходимо равномерно распределить данные между шардами. Такой успех был достигнут благодаря хорошей эффективности при доступе к данным, предсказуемом поведении при работе и относительной простоте реализации. О последнем можно судить, исходя из сложности данного алгоритма. Так, временная сложность будет рассчитываться из вычисления хэша, который имеют временную сложность $O(1)$ и самой операции нахождения остатка от деления за $O(1)$ времени. Итоговая временная сложность алгоритма шардирования получается $O(1)$.

Схема работы алгоритма представлена на рис. 2.

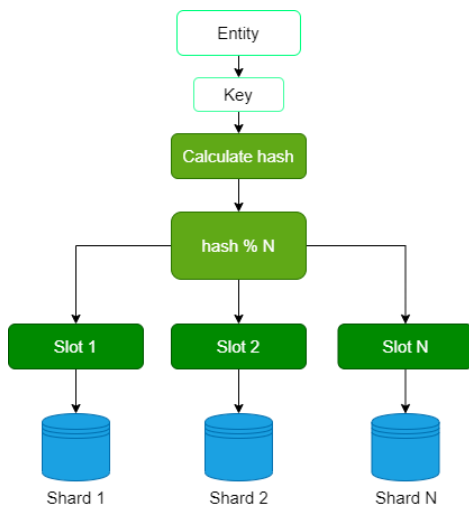


Рис. 2. Схема алгоритма шардирования Mod

Рассматривая алгоритм с технической стороны, его можно представить следующей последовательностью шагов:

- определение количества фрагментов(может быть задано заранее или получено на основе характеристик системы или требований к производительности);

- создание шардов;
- присвоение шардам номера;
- хеширование ключей данных;
- получение остатка деления (применение алгоритма mod) от хэш-значению каждого ключа;
- распределение данных по шардам.

Следующим рассматриваемым методом горизонтального масштабирования баз данных будет Range. Концепция механизма позволяет распределять данные на основе значений определенного поля или диапазона значений этого поля. Выбор которого непосредственно исходит из имеющихся ограничений и концептуальной особенности проектируемой базы данных.

Немаловажным отличием алгоритма Range является его сложность. Первоначально, если данные не отсортированы, может потребоваться предварительная сортировка данных по ключу, что может иметь временную сложность $O(N \log N)$. Далее процесс разделения данных на шарды может иметь временную сложность $O(N)$ если каждый элемент обрабатывается один раз. Также в некоторых случаях может потребоваться балансировка шардов для равномерного распределения данных. Сложность этого шага будет напрямую зависит от используемого алгоритма балансировки. Её можно посчитать равной $O(M)$. Таким образом, общая средняя временная сложность алгоритма шардирования Range может быть представлена суммой $O(N \log N)$ при учете сортировки данных и $O(N + M)$ для процесса разделения и, возможно, балансировки шардов.

Перейдем к реализации. Весь алгоритм можно представить в виде следующей последовательности действий:

- выбор ключа шардирования на основе значения определенного поля;
- создание шардов базы данных, на которые будут распределены данные;
- присвоение каждому из шардов диапазона значений ключа шардирования;
- распределение данных исходной базы данных на шарды.

Схема шардирования приведена на рис. 3.

3. Репликация

Репликация базы данных – это процесс создания и обслуживания нескольких экземпляров одной и той же базы данных, а также процесс обмена данными или изменениями в конструкции между базами в разных местах без необходимости копирования всего содержимого.

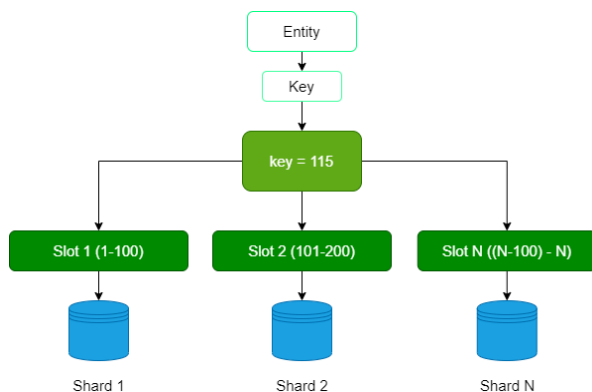


Рис. 3. Схема алгоритма шардирования Range

В большинстве реализаций репликации один сервер поддерживает главную копию базы данных, а дополнительные серверы, в свою очередь, поддерживают подчиненные копии. Две или более реплики одной базы данных остаются синхронизированными, дабы сделать систему отказоустойчивой. Исходная база данных называется проектом Мастер, и каждая копия базы данных называется Репликой. Вместе все они составляют набор реплик. В нем есть только один Мастер, который синхронно управляет другими копиями. Синхронизация – это процесс обеспечения того, чтобы каждая копия базы данных содержала одни и те же объекты и данные. При синхронизации реплик в наборе обновляются только те данные, которые изменились. Записи в базу данных отправляются на главный сервер базы данных и затем реплицируются подчиненными серверами баз данных. Операции чтения базы данных распределяются между всеми серверами баз данных, что приводит к значительному повышению производительности за счет распределения нагрузки [3].

Кроме того, репликация базы данных также может повысить доступность, поскольку подчиненные серверы баз данных могут быть сконфигурированы так, чтобы выполнять роль ведущего, если главный сервер баз данных становится недоступным по техническим причинам.

Общая схема репликации представлена на рис.4.

4. Алгоритмы репликации

Первым рассматриваемым алгоритмом будет транзакционная репликация. Данный метод предоставляет возможность автоматического копирования изменений с одной базы данных на другую, сохраняя при

этом согласованность данных друг с другом, целостность, а также внутренняя непротиворечивость [4].

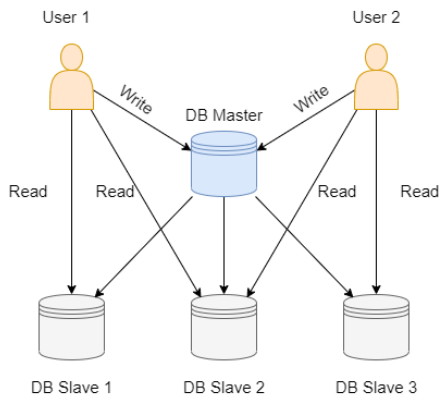


Рис. 4. Схема репликации базы данных

Алгоритм обладает большим набором полезных качеств: высокая доступность, распределенная нагрузка и отказоустойчивость. Благодаря им алгоритм применяется практически во всех системах, где необходима репликация базы данных.

Теперь перейдем к реализации. Весь алгоритм можно представить в виде следующей последовательности действий:

- захват изменений данных, которые происходят в исходной базе данных;
- передача изменений на целевые базы данных;
- применение изменений на целевых базах данных;
- подтверждение выполнения операций на последнем этапе, после успешного применения изменений на целевых базах данных.

Схема транзакционной репликации представлена на рис. 5.

Вторым рассматриваемым алгоритмом будет репликация моментальных снимков. Его идея заключается в создании снимков (точных копий) данных в момент времени, а затем передавать изменения с основной базы данных на копии для их синхронизации. Репликация снимков обычно используется, когда изменение данных должно происходить нечасто[5].

По большей сути данный алгоритм копирует рассмотренный ранее транзакционный. Но это не означает, что он встречается реже. Самое весомое отличие, что в алгоритме не происходит отслеживание изменений исходной базы данных для обновления реплик.

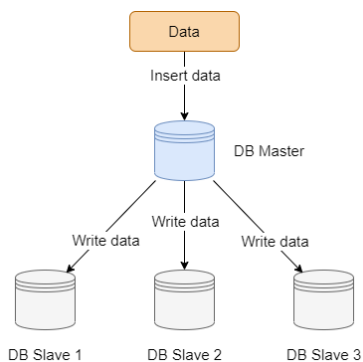


Рис. 5. Схема алгоритма транзакционной репликации

Весь процесс привязан к определенному времени, которое настраивается исходя из задачи, для которой создавалась реплика. Таким образом, происходит существенное уменьшение нагрузки на базы данных, что для некоторых систем играет важную роль.

За свою особенность алгоритм получил широкое применение в системах социальных сетей, финансового трейдинга, системах онлайн игр и других. Все эти направления объединяет то, что должна обеспечиваться быстрая реакция на изменения и мгновенная консистентность данных между узлами в определенный момент времени.

Схема алгоритма моментальной репликации представлена на рис. 6.

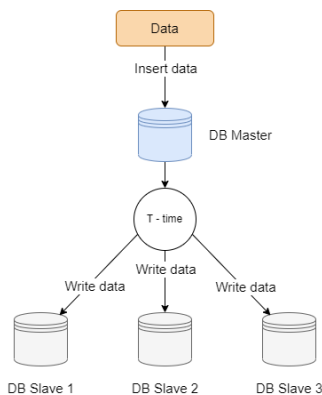


Рис. 6. Схема алгоритма моментальной репликации

Заключение

В данной статье были исследованы широко распространённые механизмы масштабирования на примере различных алгоритмов. На протяжении анализа было выявлено, что эффективная масштабируемость является неотъемлемой частью обеспечения производительности и надёжности базы данных и системы в целом при увеличении нагрузки. Каждый из этих механизмов имеет свои преимущества и ограничения, и их эффективное применение требует глубокого понимания особенностей конкретной задачи и требований к системе.

Список литературы

1. Понимание шардинга базы данных [Электронный ресурс] : сайт. – Режим доступа: <https://www.codeflow.site/ru/article/understanding-database-sharding>
2. Шардирование [Электронный ресурс] : сайт. – Режим доступа: <https://cloud.yandex.ru/ru/docs/glossary/sharding>
3. Replication [Электронный ресурс] : сайт. – Режим доступа: <https://systems.education/ddia-replication>
4. DatabaseReplication [Электронный ресурс] : сайт. – Режим доступа: <https://learn.microsoft.com/en-us/sql/relational-databases/replication/transactional/transactional-replication?view=sql-server-ver16>
5. DataReplicationinDBMS [Электронный ресурс] : сайт. – Режим доступа: <https://www.geeksforgeeks.org/data-replication-in-dbms/>

Разработка чат-бота для унификации доступа студентов ФКН к учебной информации

Шубин Д. А.

Студент бакалавр

Хвостов С. Г.

Студент бакалавр

Коротков В. В.

Старший преподаватель

Введение

В современном образовательном пространстве доступ к информации является одним из ключевых факторов успеха в обучении. Студенты Факультета Компьютерных Наук (ФКН) ежедневно сталкиваются с необходимостью получения актуальной информации о расписании занятий, дате сдачи заданий, контактных данных преподавателей и других важных сведений. Однако, в настоящее время доступ к этой информации часто бывает ограничен и неудобен, что может привести к неэффективному использованию времени и ресурсов студентов.

В последние годы наблюдается стремительный рост популярности чат-ботов как эффективного инструмента для автоматизации и упрощения взаимодействия между пользователями и информационными системами. Чат-боты, интегрированные с популярными мессенджерами, такими как Telegram [1], позволяют пользователям получать необходимую информацию в удобном и привычном формате.

В связи с этим, разработка чат-бота для унификации доступа студентов ФКН к учебной информации является актуальной задачей. Данный чат-бот должен обеспечить студентам быстрый и удобный доступ к информации о расписании занятий, а также другим важным сведениям, связанным с учебным процессом. Целью данной работы является разработка и реализация чат-бота, который будет способен предоставлять студентам актуальную и достоверную информацию в режиме реального времени, тем самым упрощая процесс обучения.

1. Архитектура системы

Бот реализован на основе фреймворка Aiogram [1]. Архитектурно в полученной системе можно выделить следующие компоненты: объект бота, диспетчер, хранилище, базу данных и планировщик задач (рис. 1). Настройка компонентов производится при помощи конфигурационного файла.

Диспетчер (Dispatcher) является центральным компонентом бота, который обрабатывает входящие сообщения и вызывает соответствующие обработчики. Он отвечает за распределение сообщений между различными обработчиками, такими как обработчики inline-кнопок или обработчики текстовых сообщений. Диспетчер также отвечает за сохранение состояний бота и пользователей в хранилище (Storage). В зависимости от настроек может использоваться либо встроенное хранилище в оперативной памяти (MemoryStorage), либо на основе Redis (RedisStorage). База данных (Database) используется для хранения долговременных данных. В зависимости от настроек может использоваться либо асинхронная реализация интерфейса взаимодействия с базой данных (AsyncSQLAlchemy), либо синхронная (SQLAlchemy). Планировщик задач используется для планирования действий, которые должны быть выполнены ботом в определенное время. Использовался планировщик из библиотеки AsyncIOScheduler [2] с хранилищем задач в Redis или в оперативной памяти.

Когда бот получает входящее сообщение, диспетчер обрабатывает его и вызывает соответствующий обработчик. Обработчик может использовать базу данных для хранения и извлечения данных, а также планировщик задач для планирования задач.

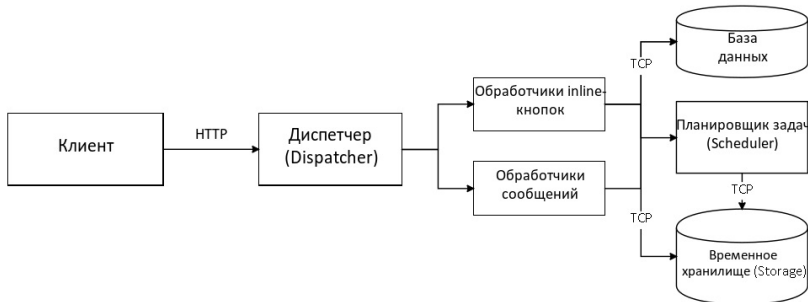


Рис. 1. Архитектура системы чат-бота

2. Функциональность чат-бота

Чат-бот обладает следующими функциональными возможностями, которые обеспечивают эффективное взаимодействие между пользователями и системой:

- Интерактивное меню. Бот реагирует на команду «/menu», отображая интерактивное меню, которое позволяет пользователям легко производить навигацию по функциям бота и получать доступ к необходимым действиям.

- Модуль настроек. Встроенный модуль настроек «/settings» предоставляет пользователям возможность оставить обратную связь, удалить свой аккаунт или вернуться в главное меню.

- Управление учетной записью. Регистрация пользователя подробно описана в разделе 3. Пользователи также могут удалить свой аккаунт через модуль настроек.

- Просмотр расписания. По соответствующей команде бот позволяет посмотреть актуальное расписание на день или неделю для указанной при регистрации учебной группы. Подробнее о получении расписания указано в разделе 4.

- Механизм обратной связи. Бот позволяет пользователям отправлять сообщения для обратной связи, которые затем пересылаются в чат администрации, что позволяет ей получать важные отзывы и предложения.

- Административная панель. Администратор может получить доступ к административной панели, отправив команду «/admin_panel», что позволяет управлять функциональностью бота, включая его перезапуск и другие операции.

- Система уведомлений. Бот реализует отправку уведомлений пользователям о начале пары, используя планировщик задач APScheduler [2], что позволяет пользователям быть в курсе своего расписания и не пропустить важные события.

3. Регистрация пользователя

Принцип работы регистрации пользователей в боте основан на последовательном выполнении нескольких шагов.

Процесс регистрации инициируется, когда пользователь отправляет соответствующую команду боту – «/start». Затем бот предлагает пользователю выбрать курс, направление по данному курсу, профиль, если их несколько, и, наконец, группу, что позволяет боту собрать полную информацию о пользователе. Выбор группы – это окончательный шаг в процессе регистрации.

Процесс регистрации пользователя реализован в соответствии с моделью конечного автомата (finite-state machine) [3]. Множество

состояний, отражающих прогресс пользователя: «выбор курса», «выбор направления», «выбор профиля» и «выбор группы». Каждое состояние имеет свои переходы, которые активируются при выполнении определенных действий пользователем. Например, когда пользователь выбирает курс, конечный автомат переходит в состояние «выбор направления». Это обеспечивает корректную последовательность шагов и предотвращает ошибки в процессе регистрации.

Данные о состоянии процесса регистрации сохраняются во временное хранилище Redis. После завершения всех действий созданный профиль пользователя добавляется в реляционную СУБД PostgreSQL. В качестве объектно-ориентированного интерфейса для работы с базой данных использовалась библиотека SQLAlchemy.

4. Получение расписания

Принцип работы парсера расписания основан на том, что он скачивает таблицу с расписанием занятий из интернета, а затем разбирает ее на отдельные ячейки, преобразуя их в двумерный список. Далее, он создает словарь объектов, где каждый объект представляет собой расписание для конкретной группы студентов.

Внутри каждого объекта парсер находит информацию о днях недели, времени и предметах, и формирует готовое расписание для каждого дня недели. Это позволяет получить полную информацию о занятиях для каждой группы студентов, что может быть использовано для поиска свободных аудиторий, планирования занятий и других целей.

Пример получаемого расписания студентами группы 13.1 курса 2 на Понедельник по знаменательной неделе представлен на рис. 2.

5. Тестирование, сборка и развёртывание

Автоматизированные тесты для чат-бота реализованы на основе фреймворка Pytest [4]. Они разделены на несколько категорий, каждая из которых проверяет определенный аспект функциональности бота. В тестах используются фикстуры (fixtures) для создания объектов, необходимых для тестирования, а также моки (mocks) для имитации поведения определенных объектов или функций. Моки позволяют изолировать тестируемый код и упрощают процесс тестирования.

Сборка и развёртывание системы производится при помощи средств контейнеризации на основе Docker [5]. Для упрощения оркестрации контейнеров, миграции базы данных и управления необходимыми зависимостями, такими как Poetry и pip, сконфигурирован файл инструкций Makefile.

Расписание на Понедельник (Знаменатель):	
Понедельник (Знаменатель)	
8:00 - 9:35 => Методы вычислений в бизнес-приложениях доц. Копытин А.В. 505П	
9:45 - 11:20 => Методы вычислений в бизнес-приложениях доц. Копытин А.В. 291	
11:30 - 13:05 => Практикум СиАКОД ст.преп. Коротков В.В. 290	
13:25 - 15:00 => Практикум СиАКОД ст.преп. Коротков В.В. 295	
15:10 - 16:45 => Структуры и алгоритмы компьютерной обработки данных ст.преп. Коротков В.В. 479	
16:55 - 18:30 => Физическая культура и спорт	16:19

Рис. 2. Пример выдаваемого расписания

В результате сборки создается контейнер `bot-container`, который включает в себя все необходимые зависимости и код бота. Кроме того, создается контейнер `redis-container` для хранения данных в Redis, контейнер `postgre-container` для хранения данных в базе PostgreSQL и контейнер `test-container` для запуска тестов. Вид работы бота в Docker представлен на рис. 3. Утилита Docker Compose используется для запуска контейнеров из образа.

Автоматизация процессов сборки и развертывания при помощи Docker и Makefile позволила значительно сэкономить время и ресурсы в процессе разработки.

Заключение

Данная статья посвящена реализации чат-бота для унификации доступа студентов ФКН к учебной информации. Описана архитектура бота, детали реализации основной функциональности, особенности тестирования, сборки и развертывания. В итоге данной разработки получился полностью рабочий чат-бот. Использование данного решения позволяет обеспечить студентам быстрый и удобный доступ к информации о расписании занятий, а также другим важным сведениям, связанным с учебным процессом. В дальнейшем предполагается использование данного чат-бота в системе электронного образования ФКН для упрощения и оптимизации процесса обучения, а также для улучшения взаимодействия между студентами и преподавателями.

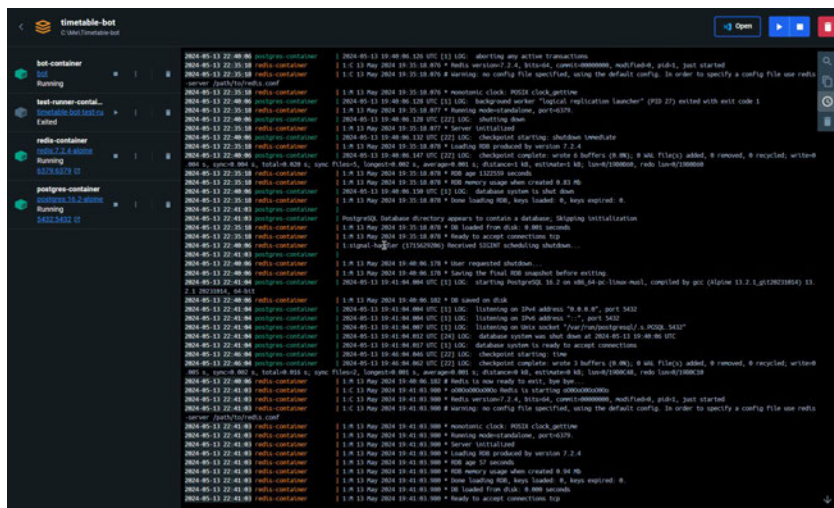


Рис. 3. Контейнеры Docker

Список литературы

1. Кузнецов С.Д. Разработка чат-ботов на основе фреймворка Aiogram. – М.: ДМК Пресс, 2020. – 272 с.
2. APScheduler Documentation. Электронный ресурс. – Режим доступа: <https://apscheduler.readthedocs.io/en/latest/>
3. Воробьев Е.С. Теория автоматов и формальных языков. – М.: Изд-во МГУ, 2018. – 320 с.
4. Okken, B. Python Testing with pytest: Simple, Rapid, Effective, and Scalable / B. Okken. – Pragmatic Bookshelf, 2017. – 600 p.
5. Гош, С. Docker без секретов : пер. с англ. / С. Гош. – Санкт-Петербург : БХВ-Петербург, 2023. – 224 с.

Статьи студентов магистратуры

Многоцелевая система сбора метрик для контейнеризированных веб-приложений

Е. И. Артемьева

Студент магистр

Я. А. Воронцов

Доцент

Введение

К современным веб-приложениям пользователи предъявляют высокие требования по доступности и стабильности. Поэтому перед тем, как продукт достигнет стадии эксплуатации, команде разработчиков нужно обеспечить мониторинг работы системы, чтобы максимально оперативно реагировать на сбои и исправлять возникающие ошибки.

Для обеспечения надежности приложения в IT-сфере есть такая практика, как SRE, проектирование надежности сайта, направленная на автоматизацию задач управления и мониторинга приложения. Таким образом, для внедрения этой практики в разработку программного продукта необходимо реализовать систему сбора и отображения показателей, чтобы обеспечить стабильность работы продукта.

Каждый проект имеет свою специфику, поэтому для каждого проекта будет свой набор показателей. Тем не менее можно выделить общие метрики, пригодные для схожих систем. Таким образом, данная работа посвящена идентификации универсальных метрик, а также реализации их сбора и отображения. Результаты данной работы могут быть использованы как основа для реализации мониторинга при разработке веб-приложений.

1. Идентификация универсальных показателей высоконагруженных систем

Чтобы выделить универсальный набор показателей, на примере одной из компаний, занимающейся разработкой ПО, был произведен анализ проектов по следующим критериям: стек технологий, архитектура и бизнес-процессы. Таким образом, типовым проектом, для которого и разрабатывалась данная система сбора метрик, является проект, на котором разработка ведется на языке программирования Java,

с использованием Spring Boot framework, разворачивается приложение в Kubernetes, а также приложение имеет интеграцию с внешней системой и является высоконагруженным. Исходя из этих критериев были выделены средства реализации, а также выделен набор показателей: показатели кластера, показатели запросов к серверу, показатели клиентских запросов.

2. Реализация сбора метрик

Сбор метрик осуществляется с помощью Spring-boot-actuator [1], зависимости, которая предоставляет конечные точки по состоянию сервиса и ее компонент, а также библиотеки micrometer [2], собирающей различные метрики spring-boot приложения, включающая как метрики виртуальной машины Java, так и метрики запросов, такие как количество запросов, время ответа и так далее. Одним из популярных инструментов для работы с метриками, работающим в связки с Kubernetes, является Prometheus, который обращается на конечные точки предоставляющие метрики и складывает их в Time Series Data Base [3]. Графики показателей отображаются с помощью Grafana, позволяющей разделить группы показателей на dashboard-ы, а также позволяющий манипулировать показателями с помощью языка PromQL для предоставления их в виде формул и отображения в удобном виде на графиках.

3. Показатели кластера Kubernetes

Так как сервисы разворачиваются в Kubernetes, для обеспечения надежности приложения требуется отслеживать работу вычислительных узлов или рабочих машин, на которых запускаются контейнеризированные приложения.

При работе с высоконагруженными системами важно отслеживать ресурсы, которые потребляет развернутое приложение. К ресурсам относится потребление CPU или ядер процессора и потребление RAM, оперативной памяти. При разворачивании приложения в конфигурации установки SRE-инженеру необходимо обозначить, сколько ресурсов каждого типа будет использовать приложение. Соответственно, если указать значение больше требуемого, приложение будет работать корректно, однако это приведет к неэффективному использованию ресурсов и к избыточной трате денег компании. С другой стороны, если выделить недостаточное количество ресурсов, приложение будет работать медленно, что приведет к увеличению отклика системы. Происходит это, потому что запущенное контейнеризированное приложение использует ядро Linux машины, на которой развернут сервис. Каждые 100 миллисекунд Kubernetes выделяет обозначенную в

конфигурации часть времени процессора на вычисления, происходящие в приложении. Если выделенная квота закончилась, а вычисления не завершились, происходит пропуск тактов процессора (throttling), что в результате приводит к медленной работе приложения. Поэтому важно отслеживать показатели потребления ресурсов для выявления оптимального значения. Приведем пример формулы для отображения графика потребления ресурсов, написанной на языке PromQL (Листинг 1).

Листинг 1

Потребление CPU приложением

```
sum(rate(container_cpu_usage_seconds_total{namespace="$namespace", pod="$pod", image!="", cluster="$cluster"}[$__rate_interval])) /  
sum(kube_pod_container_resource_limits{namespace="$namespace", pod="$pod", resource="cpu", job=~"$job", cluster="$cluster"})
```

В данной формуле `sum` – встроенная функция по вычислению суммы, `rate` – вычисляет значению в секунду нарастающего показателя, `container_cpu_usage_seconds_total` – ключ показателя потребления CPU за все время, `kube_pod_container_resource_limits` – ключ показателя выставленного лимита потребления ресурса.

Также для корректной работы высоконагруженной системы, нам нужно отслеживать показатели сети для обеспечения стабильности взаимодействия приложений между собой и пользователей с приложением. Это такие показатели, как пропускная способность, скорость передачи пакетов, ошибки сети. Приведем пример формулы для отображения одного из графиков показателей сети (Листинг 2).

Листинг 2

Скорость передачи пакетов

```
sum(rate(container_network_receive_packets_total{namespace="$namespace", pod="$pod", cluster="$cluster"}[$__rate_interval]))
```

4. Показатели запросов к серверу

Также при сопровождении программного продукта, с которым взаимодействуют конечные пользователи, важно отслеживать показатели того, как приложение отвечает на запросы. Пример отображения метрик приведен на рис. 1.

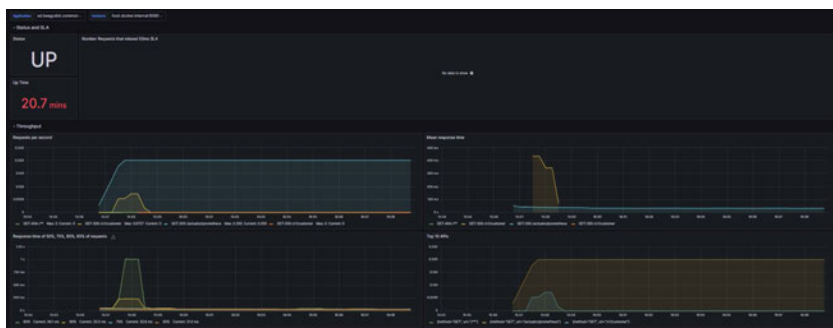


Рис. 1. Пример графиков показателей запросов к серверу

Во-первых, нужно отслеживать количество запросов, отправленных на конечную точку. Рассмотрим формулу для отображения данного графика (Листинг 3). В формуле используется функция `rate`, которая рассчитывает значение за минуту нарастающего показателя количества запросов, а также идет выборка по приложению и экземпляру Prometheus, в котором хранятся временные ряды.

Листинг 3

Количество запросов на конечную точку

```
rate(http_server_requests_seconds_count{application="$application", instance="$instance"}[1m])
```

Во-вторых, важно учитывать среднее время ответов, для того чтобы понимать, насколько быстро отвечает приложение. Вычисляется, как суммарное время ответа конечной точки, деленное на количество запросов на эту конечную точку (Листинг 4).

Листинг 4

Среднее время ответов

```
rate(http_server_requests_seconds_sum{application="$application", instance="$instance"}[1m])
/ rate(http_server_requests_seconds_count{application="$application", instance="$instance"}[1m])
```

В-третьих, среди графиков представлен показатель время ответа, которое не превышает определенный процент всех запросов (Листинг 5). Этот показатель нужен для разделения данного времени ответа на группы, чтобы понимать за сколько в среднем, в большинстве и в подавляющем большинстве отвечает наше приложение. Исходя из этого параметра можно выделить SLA для времени отклика сервиса. Если время ожидаемого отклика будет превышено для определенной

доли запросов, то со стороны команды SRE потребуется предпринимать меры для устранения причины замедления работы.

Листинг 5

Время ответа 50%, 75%, 90%, 95% запросов

```
histogram_quantile(0.95,  
sum(rate(http_server_requests_seconds_bucket{application="$app  
lication", instance="$instance"}[1m])) by (le))
```

Последний важный показатель – это 10 самых востребованных конечных точек, который позволяет отследить, каким конечным точками в коде нужно уделять более приоритетное внимание с точки зрения их производительности и надежности работы (Листинг 6).

Листинг 6

Десять самых востребованных API

```
topk(10, sum by(uri, method)  
(rate(http_server_requests_seconds_count{application="$applica  
tion"}[1m])))
```

5. Показатели клиентских запросов

Задержки в работе приложения могут возникнуть не только на стороне самого приложения. Многие проекты в компании включают в себя запросы к внешним системам. Таким образом, если задержка возникла при взаимодействии с внешним сервисом, ее нужно обрабатывать по-другому, например, обращаться к разработчикам этой системы, для устранения проблемы, или разработать более качественную стратегию повторных запросов к внешней системе. Поэтому важно также отслеживать временные показатели запросов к внешним системам. Пример отображение показателей клиентских запросов приведен на рис. 2.

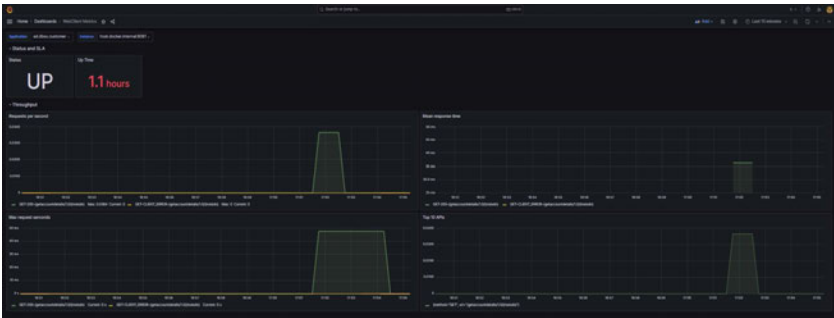


Рис. 2. Пример графиков показателей клиентских запросов

К показателям клиентских запросов относят такие показатели, как количество запросов, среднее и максимальное время ответа, а также десять самых используемых API. Приведем пример формулы для отображения графика среднего времени ответа клиента на запрос (Листинг 7).

Листинг 7

Среднее время ответа внешней системы

```
rate(http_client_requests_seconds_sum{application="$application"  
n",  
instance="$instance"}[1m])/rate(http_client_requests_seconds_c  
ount{application="$application", instance="$instance"}[1m])
```

Эти показатели схожи с показателями запросов к серверу, отличаются они настройкой на стороне кода. В используемом на стороне кода клиенте, будь то RestTemplate или WebClient, нужно указать хранилище, куда библиотека micrometer будет складывать показатели запросов, отправленных через этот клиент.

Заключение

Данная статья посвящена идентификации универсальных метрик, применимых при разработке ПО, а также реализации их сбора и отображения. Описаны способы идентификации универсальных показателей в компании, средства реализации, а также формулы, написанные на языке PromQL, используемые для отображения графиков. В итоге данной работы получился полностью рабочий программный модуль, реализованный в виде библиотеки для непрерывной интеграции и развертывания приложения. Результаты данной работы используются в компании, в результате внедрения библиотеки и использования разработанной системы сбора и отображения метрик были выявлены уязвимые места в нескольких проектах компании, увеличилась стабильность высоконагруженных приложений, а также благодаря появлению универсальной библиотеки уменьшился time-to-market, то есть время разработки продукта.

Список литературы

1. Spring Boot Actuator Web API Documentation [Электронный ресурс]. – Режим доступа: <https://docs.spring.io/spring-boot/docs/current/actuator-api/pdf/spring-boot-actuator-web-api.pdf>
2. Micrometer Documentation [Электронный ресурс]. – Режим доступа: <https://micrometer.io/docs/>
3. Bader A. Survey and Comparison of Open Source Time Series Databases / A. Bader. – Bonn: BTW, 2017. – 20 с.

Построение управления воздушного судна местных воздушных линий с использованием методов машинного обучения

О. Е. Бондаренко

Студент магистр

М. Е. Семёнов

Профессор

Введение

Авиационная индустрия является одной из наиболее инновационных и прогрессивных сфер, которая постоянно стремится к повышению безопасности полетов, снижению затрат и улучшению эффективности использования воздушных судов. Использование искусственного интеллекта (ИИ) в авиационных автопилотах стало одним из ключевых факторов, способствующих достижению этих целей. Использование искусственного интеллекта в авиационных автопилотах позволяет повысить уровень безопасности полетов за счет возможности быстрого и точного анализа большого количества данных, поступающих от различных датчиков и систем самолета. Автопилоты с ИИ могут самостоятельно адаптироваться к изменяющимся условиям полета и принимать решения на основе анализа данных о состоянии воздушного судна, погодных условиях и многих других параметрах [1].

В настоящей работе рассматривается разработка системы управления гражданским воздушным судном, основанной на методах глубокого обучения с использованием архитектуры нейронной сети – трансформер. Данный инновационный подход открывает новые перспективы для создания интеллектуальных систем управления в авиации, способных эффективно адаптироваться к различным условиям и автономно принимать соответствующие решения.

1. Методика эксперимента

В рамках эксперимента планировалось исследование работы функции автоматического управления воздушным судном в условиях, соответствующих нормальной эксплуатации, с целью оценки ее производительности и надежности в стандартных ситуациях движения по аэродрому. Это включало в себя проверку способности программно-

алгоритмического обеспечения обеспечить точное движение воздушного судна по заданному маршруту с минимальными отклонениями и соблюдением требуемых безопасных параметров. Также проводились эксперименты с моделированием нештатных ситуаций, включая возможные отказы систем управления, навигации или других критически важных элементов. Целью таких испытаний была оценка способности программного обеспечения обнаруживать и реагировать на такие ситуации, а также обеспечивать безопасное управление воздушным судном даже в критических условиях. В рамках данного исследования для анализа и тестирования различных аспектов авионики, включая системы навигации, автопилоты и управление двигателем, использовалась интегрированная среда моделирования X-Plane 11. Этот симулятор обеспечивает точное моделирование физики полета и аэродинамики, что гарантирует высокую степень реализма в поведении воздушных судов в разнообразных условиях и на различных этапах полета. Для взаимодействия с авиасимулятором X-Plane 11 использовался плагин XPlaneConnect. Этот инструмент предоставляет программный интерфейс для отправки и получения данных между X-Plane 11 и внешними программами. Плагин позволяет внешним приложениям получать информацию о положении и состоянии воздушного судна, контролировать параметры полета и управлять сценариями. С помощью этого плагина можно было получать, отправлять и обрабатывать данные из симулятора, а также использовать их в PyCharm Community Edition 2023.3.5, где и был написан основной код. В качестве архитектуры сети было решено выбрать нейронную сеть типа трансформер [2-3]. Декодеры данного типа генерируют информацию на основе предыдущего контекста. Механизм внимания в трансформере позволяет анализировать предыдущие шаги и принимать решения, учитывая контекст. Этот подход особенно полезен, когда необходимо с учетом предыдущих шагов сгладить резкие изменения в управлении, что актуально, например, при управлении полетом воздушного судна.

2. Архитектура управления

Общий цикл программы начинается с определения текущего состояния самолета и его положения относительно маршрута. Затем программа инициирует движение самолета в соответствии с заданными правилами управления и моделью управления по взлетно-посадочной полосе. По мере движения по маршруту происходит мониторинг полета и коррекция управления. Когда самолет достигает предела текущего сегмента маршрута, программа переходит к следующему сегменту, повторяя процесс движения и управления.

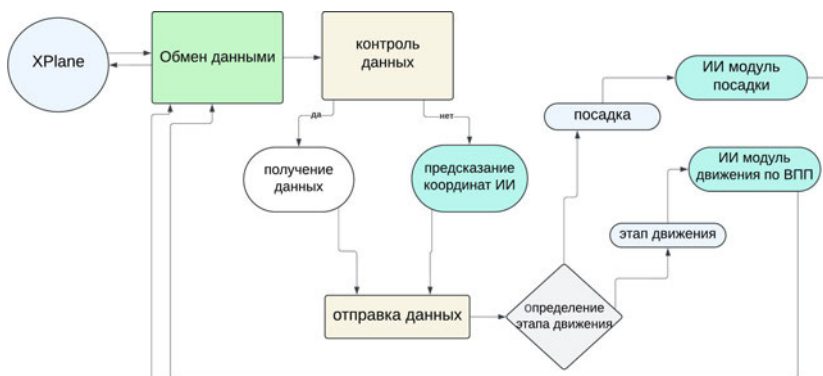


Рис. 1. Архитектура управления

Основные моменты архитектуры исследования:

1. Подключение к X-Plane11 с помощью XPlaneConnect.

Создается экземпляр класса XPlaneConnect, который предоставляет интерфейс для взаимодействия с X-Plane11 через UDP-соединение. Проверяется доступность данных из X-Plane11, используя методы получения данных, предоставленные классом XPlaneConnect.

2. Получение данных из X-Plane11.

Если данные доступны, они получаютсся с помощью методов класса XPlaneConnect, таких как readDATA() для получения данных о положении, readPOSI() для получения данных о положении самолета и т.д.

3. Предсказание координат ИИ (если данные не доступны).

Если данные из X-Plane11 недоступны, происходит предсказание координат с помощью модели, которая была обучена на предыдущих данных о полете или других параметрах.

4. Отправка данных в X-Plane11.

Полученные данные (реальные или предсказанные) отправляются обратно в X-Plane11 с помощью методов класса XPlaneConnect, таких как sendPOSI() для отправки данных о положении, sendCTRL() для отправки управляющих сигналов.

5. Определение этапа движения.

После отправки данных происходит анализ текущего этапа движения самолета. Анализ этапа движения выполняется на основе текущих координат, скорости, высоты и других параметров полета.

6. Запуск ИИ модуля этапа посадки (если требуется).

Если определено, что самолет находится на этапе посадки, запускается специальный ИИ модуль, который управляет процессом

посадки. Он может осуществлять автоматическое управление управляющими поверхностями, двигателями и другими аспектами полета для выполнения безопасной посадки.

3. Метод оптимизации

В контексте обучения с подкреплением агент обычно взаимодействует с окружающей средой (в данном случае с симулятором полета), получая наблюдения и принимая действия в соответствии с текущим состоянием среды и выбранным стратегией. При использовании алгоритма РРО агент обучается на основе опыта, накопленного во время взаимодействия со средой, с целью максимизации некоторой целевой функции, такой как награда:

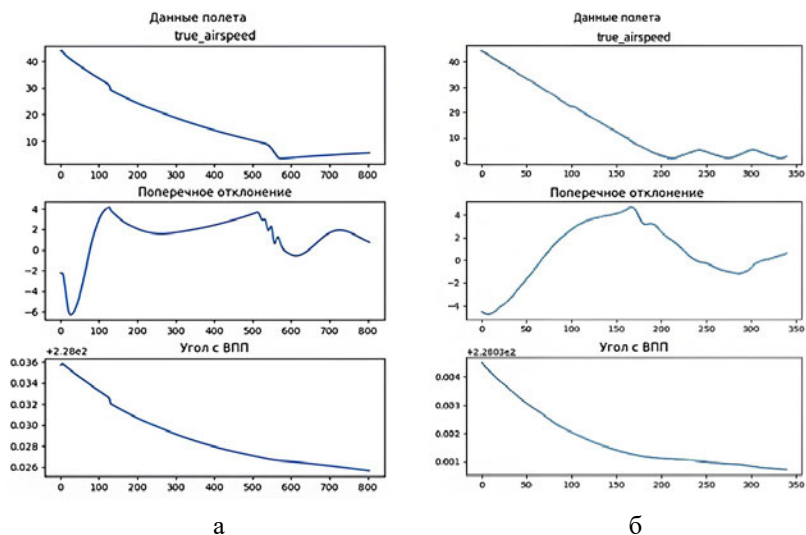
$$r = \lambda_1 y^2 + \lambda_2 \dot{y}^2 + \lambda_3 \dot{d} + f \quad (1)$$

где y^2 - поперечное отклонение от траектории, $\dot{y}^2$ - поперечная скорость от траектории, $\dot{d}$ - скорость изменения управляющими поверхностями, f - штраф за крушение, λ_i - коэффициенты оценок.

Формула (1) вычисляет значение r на основе текущих значений y^2 , $\dot{d}$ и f . Коэффициенты λ_1 , λ_2 и λ_3 позволяют настраивать то, насколько каждая переменная вносит свой вклад в общее значение r . Штраф за аварию включен в расчет r для того, чтобы стимулировать систему избегать аварий. Скорость изменения управляющих поверхностей указывает на скорость изменения положения или угла управляющих поверхностей, таких как руль, элероны или рулевые поверхности, которые управляют ориентацией или движением объекта, например, самолета. Эта переменная отражает, насколько быстро изменяются управляющие поверхности, и может использоваться для оценки динамики или реакции системы на управляющие воздействия. Например, при быстром изменении положения руля самолета его $\dot{d}$ будет высокой, что может привести к резкому изменению курса или ориентации самолета. Нейросеть запоминала значение r и старалась его минимизировать.

4. Результаты эксперимента

В экспериментальном исследовании испытания проводились при штатном и нештатном режимах функционирования систем и оборудования.



*а – штатное функционирование систем,
б – нештатное функционирование систем*

Рис. 2. Результаты работы программно-алгоритмического обеспечения функции самолетовождения для двух режимов

В ходе проведенных испытаний воздушное судно проявило высокую эффективность на этапе движения по аэродрому как в штатных, так и в нештатных режимах работы. Из рис. 2 видно, что поперечное отклонение воздушного судна от заданного курса не превышает 4% от половины ширины взлетно-посадочной полосы (ВПП), а угол с ВПП остается в пределах 5% от половины ширины ВПП. Эти результаты свидетельствуют о том, что воздушное судно удерживалось в пределах ВПП во время движения по ней, не выходя за ее границы.

В таблице левый столбец представляет различные сценарии движения по взлетно-посадочной полосе (ВПП) в различных условиях, включая разные метеоусловия и функционирование систем. Правый столбец отображает результаты экспериментов для каждого сценария. Испытания показали, что среднее отклонение воздушного судна от центральной линии ВПП остается невысоким, что подтверждает его способность удерживаться в пределах ВПП во время движения по ней. Также отмечается отсутствие выезда за границы ВПП и отсутствие

повреждений воздушного судна во всех рассмотренных сценариях движения.

Таблица

Результаты экспериментов

№ п/п	Выполняемое действие	Результат
1	Движение по ВПП на сухой поверхности при простых метеоусловиях.	Среднее отклонение ВС от центральной линии менее 0.2 от ширины ВС, отсутствует выезд за границы ВПП.
2	Движение по ВПП при отказах самолетных систем (системы торможения колес, системы поворота носовой стойки колес).	Отсутствует выезд за границы дорожного покрытия, отсутствие повреждений ВС.
3	Движение по ВПП на влажной поверхности при простых метеоусловиях.	Среднее отклонение ВС от центральной линии менее 0.2 от ширины ВС, отсутствует выезд за границы ВПП, отсутствие повреждений ВС.
4	Движение по ВПП при боковом ветре, не превосходящем 10 узлов.	Отсутствует выезд за границы дорожного покрытия, отсутствие повреждений ВС.

Заключение

Разработанные программно-алгоритмические средства для функции самолётовождения успешно прошли испытания и продемонстрировали высокую эффективность в управлении исполнительными системами ВС МВЛ в процессе движения по аэродрому. В штатном режиме функция обеспечивает устойчивое движение по ВПП с минимальными отклонениями от центральной линии, а в нештатных ситуациях – безопасное управление и сохранность ВС при различных отказах и экстремальных условиях. Эти результаты подтверждают перспективность и надежность разработанных средств в обеспечении безопасности воздушного движения на этапе движения по аэродрому.

Список литературы

1. Kabashkin, Igor, Boriss Misnevs, and Olga Zervina. 2023. "Artificial Intelligence in Aviation: New Professionals for New Technologies." *Applied Sciences* 13, no. 21: 11660. <https://doi.org/10.3390/app132111660>
2. Волосова, А. В. Технологии искусственного интеллекта в ULS-системах: учеб. пособие / А. В. Волосова. – 2-е изд., испр.- СПб: Лань, 2024.- 308 с.
3. Воронина, В.В. Теория и практика машинного обучения : учебное пособие /В. В. Воронина, А. В. Михеев, Н. Г. Ярушкина, К. В. Святков. –Ульяновск : УлГТУ, 2017. – 290 с.

Разработка и реализация онтологии в сфере интеллектуальной собственности

А. А. Борисова

Студент магистрант

В. В. Гаршина

Доцент

Введение

В современном мире правовая область знаний является достаточно сложной предметной областью. Современное общество характеризуется частым изменением законодательства: появлением новых и обновлением существующих нормативных правовых актов. Кроме того, в связи с цифровой трансформацией общества все более значимые аспекты переносятся в цифровую форму. Не является исключением область интеллектуальных прав, которая с бурным развитием информационных систем и технологий претерпевает значительные изменения и становится важнейшей областью урегулирования правовых вопросов в IT-области, поскольку правовая сфера значительно отстает от цифровых технологий.

В настоящее время во многих сферах жизнедеятельности используются системы, основанные на знаниях. Такие новые системы включают в себя возможность получения скрытых знаний, неочевидных при использовании неполной, но непротиворечивой информации [1].

Первым шагом в построении таких систем является разработка онтологии – формального описания терминов предметной области и отношений между ними с использованием дескриптивной логики. В последние годы построение онтологий становится популярной темой для многочисленных исследований, как в России, так и за рубежом. На основании онтологий разработаны базы знаний в различных областях человеческой деятельности. Однако наиболее востребованной является медицина, биология и юриспруденция [2].

1. Формальная модель онтологии предметной области

Онтология права основана на модели, позволяющей описывать понятия и взаимосвязи между ними, включающая в себя такие элементы, как законы, подзаконные акты, правила, нормы и судебные

решения. С помощью правовой онтологии возможно создать базы знаний, которые позволят анализировать и принимать решения на основе имеющихся неполных данных. В этом случае, преимуществами правовой онтологии является:

- возможность более эффективно и точно анализировать и интерпретировать правовую информацию;
- упрощение доступа к правовым знаниям и помощь в решении сложных задач правового характера;
- повышение эффективности работы специалистов в правовой сфере и сокращение затрат на проведение анализа.

Онтологическая модель выражает сущностное (смысловое) содержание с точки зрения лица, принимающего решение (пользователя системы). Разные задачи принятия решений могут требовать различных онтологий даже для одной предметной области, так как онтология – всего лишь проекция предметной области под тем или иным углом [3].

Онтологическая модель – понятие, охватывающее несколько моментов:

- Онтология представляется в виде схемы предметной области,
- Онтология представляется в виде базы утверждений о предметной области, в которой хранятся экземпляры конкретных сущностей.
- Логический вывод реализуется в виде модуля вывода утверждений и рассматривается как совокупность аксиом (правил вывода новых фактов на основе имеющихся).

Часть IV Гражданского кодекса Российской Федерации [4] регламентирует множество видов объектов интеллектуальной собственности: авторского права (глава 70), смежных прав (глава 71), патентного права (глава 72) и средства индивидуализации (глава 76) и нетрадиционных объектов (глава 73). Кроме того, перечень этих объектов не является исчерпывающим. Таким образом, для разрабатываемой системы необходимо предъявить условие, чтобы она могла работать с различными объектами интеллектуальной собственности.

Для разработки онтологии используется система Protégé, основанная на языке Java, которая является свободной распространяемой программой с графическим интерфейсом и большим количеством поддерживаемых форматов.

В ходе анализа были выделены следующие понятия предметной области и связи между ними. Онтология включает в себя шесть классов (концептов) – Защита интеллектуальных прав, Средства индивидуализации, Смежные права, Нетрадиционные объекты,

Патентное право, Авторское право. При этом, построенная онтология предметной области включает в себя подклассы, связанные между собой иерархическими отношениями.

2. Реализация онтологии с использованием Protégé

Класс Авторское право включает в себя 24 подкласса авторского права, класс Защита интеллектуальных прав включает в себя виды договоров в сфере интеллектуальной собственности, источники и методы регулирования, судебную защиту (7 подклассов). Отдельным классом выделены Нетрадиционные объекты, в который входят 3 подкласса, которые регулируются отдельными главами Гражданского кодекса РФ и не относятся к авторскому или патентному праву, но связаны с классом Защита интеллектуальных прав. Класс Патентное право включает в себя объекты и признаки патентных прав и заложены в иерархию классов в виде 10 подклассов. Также в онтологии выделены еще 2 класса: Смежные права и Средства индивидуализации, включающие в себя 10 наследуемых подклассов. Разработанная онтология включает в себя 60 концептов. Пример построения иерархии классов по объектам права интеллектуальной собственности представлен на рис. 1.

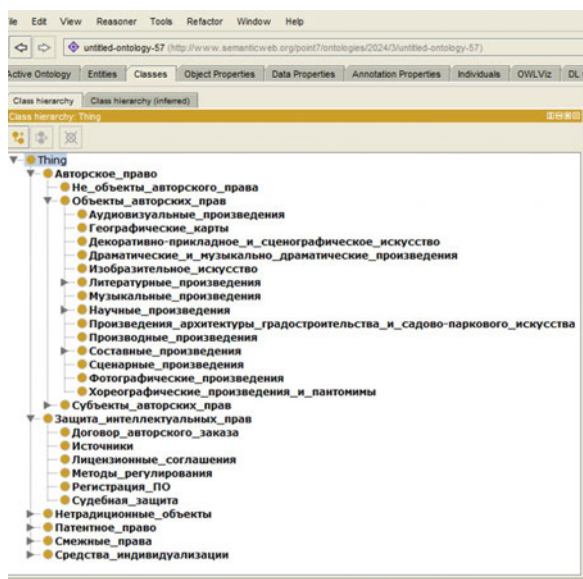


Рис. 1. Иерархия классов в редакторе Protégé

Классы и экземпляры связываются между собой отношениями с помощью вкладки *object properties*. Отношения имеют такие характеристики как функциональность, инверсивность, транзитивность, симметричность и нужны для построения логических правил на основе дескриптивной логики.

Для наполнения онтологии создадим экземпляры классов предметной области и свяжем эти экземпляры с классами и свойствами друг с другом с помощью характеристик отношений. Для создания экземпляров классов используется закладка *Individuals*. Экземпляры наполняют классы, им задаются свойства. Например, индивид «Опера» является экземпляром подкласса «Музыкальные произведения» класса «Объекты авторского права», который, в свою очередь, входит в класс «Авторское право». Пример связей классов и экземпляров показан на онтологическом графе (рис. 2), который наглядно демонстрирует взаимосвязь классов и сущностей.

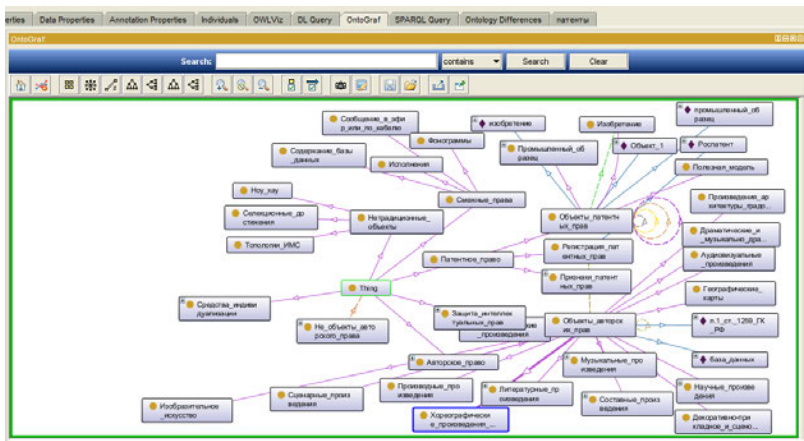


Рис. 2. Пример построения онтологического графа предметной области

3. Реализация онтологической модели права интеллектуальной собственности

После того как были определены классы в иерархическую структуру (подкласс-надкласс), заполнив экземпляры и связав их свойствами – выделим некоторые фундаментальные правила разработки онтологии, построенные на дескриптивной логике. Такие конструкции обладают очень большими возможностями: имея неполную информацию об объектах с помощью построения логических правил,

прописанных в отношениях, можно обработать эту неполную информацию. Например, возьмем экземпляр класса «АО_Дикси_ГРУПП» и зададим информацию, что он имеет коммерческое обозначение Дикси. При запуске обработчика (плагин Reasoner) получаем «скрытые знания» экземпляра Дикси, которые мы не задавали вручную, а программа анализирует заданные правила дескриптивной логики и на ее основе выдает результат, который явно не был прописан: Дикси является коммерческим обозначением «АО_Дикси_ГРУПП», регистрируется в Государственном реестре товарных знаков и регистрируется по ст. 1480 ГК РФ. Пример выявления скрытых знаний с помощью дескриптивной логики представлен на рис. 3. Желтым цветом обозначены «скрытые знания», которые выявляет система в качестве прогноза на основе правил дескриптивной логики.

В характеристиках связей была указана инверсивность связей: «имеет коммерческое обозначение» и «является коммерческим обозначением»; «регистрирует» и «регистрируется в», а также функциональность свойства «имеет коммерческое обозначение», т.е. только один товарный знак может иметь предприятие или организация.

Для успешной работы онтологии некоторым связям была присвоена транзитивность. Транзитивность свойств означает, что если экземпляр А связан с экземпляром Б свойством Р, и экземпляр Б связан с экземпляром В свойством Р, то экземпляр А будет связан с экземпляром В свойством Р. Например, такие свойства как «не охраняется законом», «должна быть зарегистрирована», «может быть зарегистрирована», «охраняется законом», «является» имеют характеристику транзитивности.

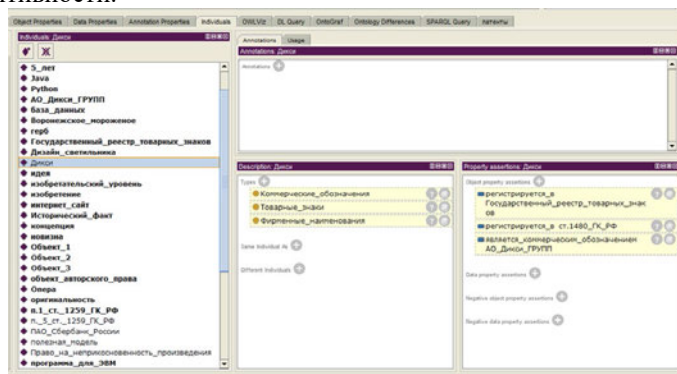


Рис. 3. Выявление «скрытых знаний» экземпляра «Дикси»

Таким образом, разработанная онтология, составленная на основе Гражданского кодекса РФ и построенная на правилах дескриптивной логики, позволяет оперировать скрытыми знаниями об объектах интеллектуальной собственности, связывая при этом экземпляры объектов со статьями Гражданского кодекса РФ. Данные онтологии могут использоваться в различных информационных системах

Заключение

Данная статья посвящена разработке и реализации онтологической модели предметной области – права интеллектуальной собственности – на основе дескриптивной логики. Описаны характеристики связей при построении онтологии предметной области. В итоге показаны логические правила, реализованные в онтологии. Использован подход, позволяющий получить информационную систему на языке Java, с возможной последующей интеграцией онтологии либо в уже готовые решения, например, в справочные юридические системы, либо в системы поддержки принятия решений, или создать новую информационную систему на ее основе.

Наиболее значимым является сам подход разработки систем принятия решений на основе онтологической модели, построенных на нормах закона части IV Гражданского кодекса РФ, позволяющий использовать скрытые (неявные) и непротиворечивые знания, который не применялся ранее для объектов интеллектуальной собственности.

Список литературы

1. Ломов, П. А. Разработка технологии проверки и согласования нормативно-правовой базы на основе онтологий / П. А. Ломов, А. Г. Олейник // Труды ИСА РАН. – 2013. – Т. 63. – С. 62-69.
2. Константинова, Н. С. Онтологии как системы хранения знаний / Н. С. Константинова, О. А. Митрофанова. – Санкт-Петербургский государственный университет, 2012. – 120 с.
3. Лапшин, В. А. Онтологии в информационных системах / В. А. Лапшин. – М.: Научный мир, 2020. – 224 с.
4. Гражданский кодекс Российской Федерации (часть четвертая) от 18.12.2006 N 230-ФЗ // Собрание законодательства РФ, 25.12.2006, N 52 (1 ч.), ст. 5496. (ред. от 30.01.2024).

Использование архитектуры трансформер для решения задачи описания изображения

П. С. Бурдюг

Студент магистр

А. В. Акимов

Старший преподаватель

Введение

Машинное описание изображения – это довольно новая задача в области машинного обучения. Она заключается в генерации моделью предложения, полностью описывающего изображение. Её решение помогает слабовидящим и потерявшим зрение людям лучше понимать изображения, а также позволяет осуществлять поиск изображений в сети Интернет с использованием автоматически сгенерированных описаний.

Модель, решающая задачу, должна уметь извлекать всю информацию из изображения, а затем, используя её, генерировать осмысленные приложения на естественном языке. Таким образом, задача находится на стыке компьютерного зрения и обработки естественного языка.

Ранее была рассмотрена модель энкодер-декодер, решающая эту задачу, а также возможные её улучшения, главным из которых является использование модуля внимания [1]. Было произведено сравнение модели энкодер-декодер с модулем внимания с обычной моделью с помощью специализированных метрик и доказана его эффективность [2]. Тем не менее, данная модель имеет несколько недостатков, первый из которых – рекуррентная сеть в декодере, которая работает последовательно, тем самым замедляя обучение и генерацию описаний, а второй – отсутствие механизмов самовнимания, которые также могут повысить эффективность описания изображений. Эти проблемы могут быть решены использованием архитектуры трансформер, которая в полной мере реализует все механизмы внимания и не требует использования рекуррентных сетей [3]. В данной работе она будет рассмотрена применительно к задаче описания изображений, а также

будет произведено сравнение эффективности такой модели с моделями энкодер-декодер.

1. Архитектура трансформер. Энкодер

На рис. 1 представлена архитектура трансформер для решения задачи описания изображения, рассмотрим её более подробно.

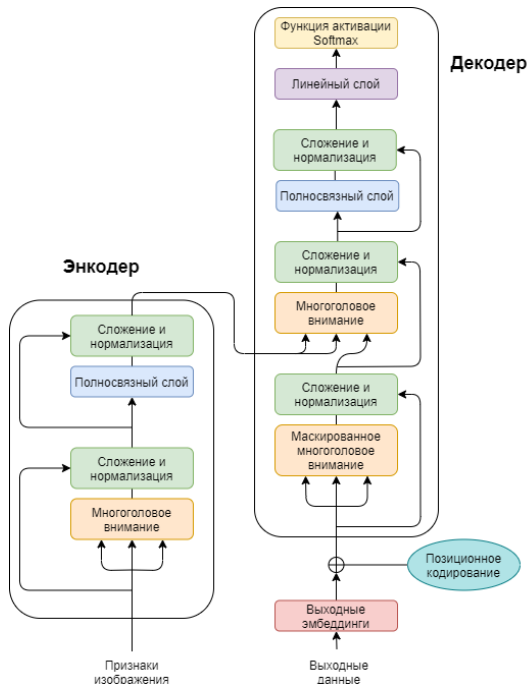


Рис. 1. Схема архитектуры трансформер для решения задачи описания изображения

Модель состоит из двух частей: энкодера, который работает с изображением (левая часть рис. 1) и декодера, который генерирует слова описания (правая часть рис. 1). Энкодер принимает на вход матрицу признаков изображения, полученную из него с помощью любой свёрточной нейросети и приведенную к нужной размерности с помощью дополнительного полносвязного слоя. Она попадает в блок многоголового внимания, который выявляет закономерности между признаками изображения, тем самым реализуя для него механизм самовнимания.

Рассмотрим его устройство. На вход модуль принимает матрицу эмбедингов X : для энкодера строки матрицы – векторы представления изображения, а для декодера – векторные представления слов, причем все эмбединги имеют одинаковую размерность.

Для каждого эмбединга x с помощью обучаемых матриц W^Q , W^K , W^V рассчитываются три новых эмбединга: запрос q , ключ k и значение v . Запрос – вектор обращения к контексту, то есть смысловое значение, на которые мы бы хотели обратить внимание из признака (слова) x , ключ – краткое смысловое значение признака, к которому можно обратиться из других слов, значение – полный смысл признака. В итоге, получаем три соответствующие матрицы эмбедингов Q , K , V :

$$\begin{aligned} Q &= X \times W^Q; \\ K &= X \times W^K; \\ V &= X \times W^V. \end{aligned} \quad (1)$$

Теперь из признака x_1 может быть рассчитан коэффициент внимания a_{12} к признаку x_2 , как скалярное произведение вектора запроса для первого признака q_1 и вектора ключа для второго k_2 :

$$a_{12} = q_1 \cdot k_2. \quad (2)$$

Таким образом, чем более похожи вектора q_1 и k_2 по направлению, тем больше будет коэффициент внимания. Если рассчитать эти коэффициенты для всех пар признаков получим матрицу внимания A :

$$A = \text{Softmax} \left(\frac{Q \times K^T}{\sqrt{d_k}} \right), \quad (3)$$

где d_k – размерности векторов k и q . Деление на корень из размерности необходимо, чтобы сохранить изначальное стандартное отклонение, так как его рост из-за умножения матриц может привести к затуханию или взрыву градиента при обучении [3]. Использование функции softmax необходимо для нормировки коэффициентов внимания, она применяется построчно.

Итоговая матрица эмбедингов Z может быть получена в результате применения взвешанной с использованием коэффициентов внимания суммы эмбедингов значения V , для этого необходимо умножить матрицу внимания A на матрицу эмбедингов значения V :

$$Z = A \times V. \quad (4)$$

В результате, для каждого признака будут учтены смыслы наиболее важных для него признаков.

Так как модуль внимания многоголовый, матриц W^Q , W^K , W^V может быть несколько, каждый набор образует голову, которая возвращает матрицу эмбедингов Z . Каждая голова может обращать внимание на разные взаимосвязи между признаками изображения или словами в предложении. Все их выводы объединяются в одну матрицу, после чего идет линейный слой W^O , который осуществляет проекцию и восстанавливает исходную размерность эмбедингов (рис. 2).

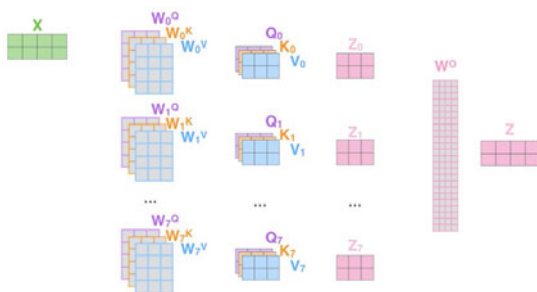


Рис. 2. Преобразования данных, происходящие внутри модуля многоголового внимания

После многоголового внимания в энкодере идет блок сложения и нормализации. Сложение заключается в использовании остаточного соединения, то есть к выводу блока внимания Z мы добавляем матрицу исходных эмбедингов X , это помогает бороться с проблемой затухания градиента. Нормализация преобразует эмбединги, так, чтобы их среднее равнялось нулю, а стандартное отклонение единице. Так вектор представления x будет преобразован в нормализованный вектор x_{norm} :

$$x_{norm} = \frac{x - x_m}{x_{std}}, \quad (5)$$

где x_m — среднее значение, а x_{std} — стандартное отклонение. Нормализация делает обучение более стабильным.

Далее идет полносвязный слой. Он состоит из двух линейных слоев, с нелинейной функцией активации между ними, как правило первый слой увеличивает число размерностей признака в 4 раза, а второй сжимает обратно. Полносвязные слои хранят от 66 до 75 % всех обучаемых весов модели, в них сохраняются основные её знания [4].

После полносвязных слоев также используется сложение и нормализация. Перейдем к устройству декодера.

2. Архитектура трансформер. Декодер

Декодер принимает на вход векторные представления слов описаний, но при этом модель никак не учитывает порядок слов в предложении и результат не изменится от их перестановки, поэтому необходим блок позиционного кодирования. Его логика заключается в добавлении к эмбедингу каждого слова векторного представления его позиции в предложении, сформированного аналогично представлению слов и имеющего такую же размерность.

Следующим является блок многоголового самовнимания, он работает аналогично такому же блоку в энкодере, но в отличии от него имеет маску. Дело в том, что декодер генерирует слова описания последовательно и каждый раз ему доступны только предыдущие сгенерированные слова. Для того, чтобы декодер обращал внимание только на них и игнорировал будущие слова, к матрице внимания применяется маска (рис. 3). Сиреневым цветом на рисунке показаны нулевые значения элементов матрицы маски, а желтым единичные. Таким образом, после применения маски, коэффициенты внимания каждого слова для предыдущих слов останутся прежними, а для следующих обнулятся.

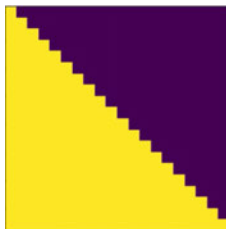


Рис. 3. Маска для модуля самовнимания декодера

После каждого блока, как и в энкодере, используется блок сложения и нормализации. Далее в декодере идет блок многоголового внимания, который позволяет обращать внимание из декодера на информацию, полученную в энкодере. Он работает, аналогично блокам самовнимания, но использует в качестве матрицы запросов матрицу Q из самовнимания декодера, а матрицы ключей K и значений V из модуля внимания энкодера. Таким образом, декодер, используя свой контекст слов в предложении, может генерировать слова на основе смысла различных частей изображения, выделенных в энкодере.

Далее идёт уже знакомый блок полносвязный слой и, наконец, на выходе декодера расположен линейный слой, который преобразует эмбединги к размерности словаря. После этого, чтобы получить вероятности каждого из слов, применяется функция softmax.

3. Обучение модели

Для реализации и обучения модели использовалась облачная среда разработки Kaggle, язык программирования Python и фреймворк Keras.

В качестве сверточной сети для извлечения признаков использовалась EfficientNet-B0.

Модель обучалась отдельно на двух датасетах Flickr8k (8 тысяч изображений) и Flickr30k (30 тысяч изображений), каждое изображение в обоих датасетах имеет 5 вариантов описаний. Датасеты были разделены на обучающую и валидационную выборки в соотношении 80% к 20%.

Для энкодера в блоке многоголового внимания использовалось 2 головы, а для блоков декодера 3.

Важной особенностью обучения модели трансформер является необходимость «разогрева» – тренировки модели с низким темпом обучения в течении первых итераций и постепенное увеличение темпа до нормального [3]. Для реализации этой логики был определён планировщик темпа обучения lr :

$$lr = \begin{cases} lr_n \cdot \frac{s_{curr}}{s_{warm}}, & \text{если } s_{curr} < s_{warm}, \\ lr_n, & \text{если } s_{curr} \geq s_{warm}, \end{cases} \quad (6)$$

где lr_n – обычный темп обучения после «разогрева», s_{curr} – номер текущего шага обучения, s_{warm} – число шагов для «разогрева». В свою очередь s_{warm} рассчитывается как часть k от общего числа итераций, определяемого произведением размера обучающей выборки s_{train} на число эпох обучения e :

$$s_w = k \cdot s_{train} \cdot e. \quad (7)$$

Для «разогрева» использовалась 1/15 часть всех шагов, обучение производилось в течении 30 эпох с темпом обучения после «разогрева» 0.0001 с использованием оптимизатора Adam, размер батча составлял 256. Также была реализована логика ранней остановки обучения: если в течении трёх эпох не удалось уменьшить значение функции потерь на валидационной выборке, то обучение останавливается, а модель

восстанавливает лучшие веса. Так, для датасета Flickr8k обучение было остановлено на 26 эпохе, а для Flickr30k на 22 эпохе.

4. Сравнение эффективности модели трансформер с моделями энкодер-декодер

Сравним эффективность модели трансформер с моделями энкодер-декодер и энкодер-декодер с модулем внимания. Для сравнения будем использовать метрики машинного перевода: BLEU, METEOR, CIDEr [2]. Для реализации расчета метрик использовалась библиотека русосоеvalcar [5]. Метрики рассчитывались на валидационной выборке, одинаковой для всех трёх моделей.

Значения метрик для моделей приведены в таблице 1 и таблице 2, где BLEU-N – метрики BLEU, в каждой из которых с равной степенью важности учитываются только словосочетания с количеством слов от 1 до N.

Таблица 1

Значения метрик для моделей на датасете Flickr8k

Модель/ метрика	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	CIDEr
Энкодер-декодер	51,0	31,2	19,2	10,1	15,3	35,8
Энкодер-декодер с модулем внимания	54,1	33,9	21,1	13,5	18,2	37,1
Трансформер	57,8	38,8	24,6	16,1	18,6	42,3

Таблица 2

Значения метрик для моделей на датасете Flickr30k

Модель/ метрика	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	CIDEr
Энкодер-декодер	53,0	31,3	18,2	10,5	13,0	24,7
Энкодер-декодер с модулем внимания	54,3	33,7	19,2	13,2	17,3	29,7
Трансформер	62,8	45,8	32,8	23,6	19,4	50,9

Можно заметить, что с ростом длины учитываемых словосочетаний, значения метрики BLEU значительно уменьшается, что логично так как модели сильно сложнее сгенерировать похожее словосочетание из 4-х слов, чем более короткое. Все метрики имеют лучшие показатели для модели трансформер.

На рис. 4 представлены диаграммы значения метрик для трех моделей на датасетах Flickr8k и Flickr30k. Здесь для анализа из всех метрик BLEU используется только BLEU-4, так как она учитывает все словосочетания с количеством слов от 1 до 4, а также экспериментально доказано, что она имеет наибольшую корреляцию с человеческими оценками [6].

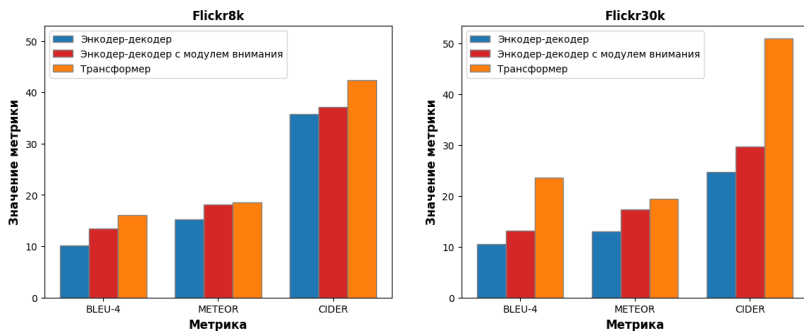
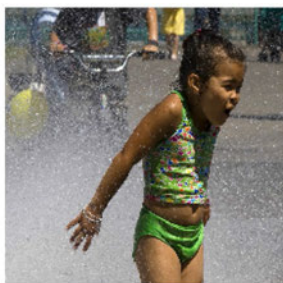


Рис. 4. Диаграммы значений метрик моделей на датасетах Flickr8k и Flickr30k

Анализируя графики на рис. 4 можно сказать, что трансформер имеет качество выше, чем модели энкодер-декодер, при этом на большем датасете Flickr30k эффективность трансформера значительно повышается. Дело в том, что архитектура трансформер очень эффективна с точки зрения потребления данных: качество модели растет с увеличением размера обучающей выборки [3].

Сравним качество генерации описания на конкретном примере. На рис. 5 представлено выбранное для этого изображение из датасета Flickr8k, первое из возможных его описаний в датасете, а также результаты его описания моделью энкодер-декодер: сгенерированное предложение и значения метрик BLEU и METEOR.

Одно из эталонных описаний – “girl in swimsuit stands in the spray of water bicycle in the background” («девочка в купальнике стоит в брызгах воды от велосипеда на заднем плане»). Сгенерированное предложение “boy in green shirt is playing in the water” («мальчик в зеленой рубашке играет в воде») имеет смысл, но плохо описывает изображение. Например, на нём изображена девочка, а модель описывает мальчика. Также в описании не хватает деталей, предложение слишком кратко и просто описывает ситуацию на картинке.



Target: girl in swimsuit stands in the spray of water bicycle in the background
Generated caption: boy in green shirt is playing in the water

BLEU-1: 59.7
BLEU-2: 36.5
BLEU-3: 0.0
BLEU-4: 0.0
METEOR: 15.3

Рис. 5. Пример описания изображения моделью энкодер-декодер

Ещё можно заметить, что метрики BLEU-3 и BLEU-4 имеют нулевые значения, это означает, что ни одного словосочетания с числом слов 3 или 4 из сгенерированного предложения не встречается в описаниях эталонах.

На рис. 6 в аналогичном формате представлен результат для модели трансформер.



Target: a girl in a swimsuit stands in the spray of water a bicycle in the background
Generated caption: a young girl in a yellow bathing suit is splashing in a puddle of water

BLEU-1: 80.0
BLEU-2: 67.6
BLEU-3: 52.0
BLEU-4: 39.1
METEOR: 32.1

Рис. 6. Пример описания изображения моделью трансформер

Сгенерированное описание “a young girl in a yellow bathing suit is splashing in a puddle of water” («маленькая девочка в желтом купальнике плещется в луже воды») уже более развернуто описывает изображение, используя ситуативные слова, которые лучше подходят к картинке и

встречаются в эталонных предложениях. Поэтому и значения всех метрик в данном случае выше.

Таким образом, трансформер, используя механизмы многолового самовнимания и внимания между энкодером и декодером, позволяет качественнее решать задачу описания изображения. Кроме того, отсутствие рекуррентных частей позволяет повысить эффективность его работы и обучения на графических ускорителях.

Заключение

В рамках данной работы была рассмотрена архитектура трансформер и её применение для решения задачи описания изображения. Была реализована и обучена соответствующая модель. Было проведено сравнение её эффективности с моделями энкодер-декодер, рассмотренными ранее, с помощью метрик BLEU, METEOR, CIDEr. Результаты сравнения показали эффективность архитектуры трансформер, особенно при увеличении размера обучающей выборки.

Список литературы

1. Бурдюг, П. С. Решение задачи описания изображений с помощью нейронных сетей / П.С. Бурдюг, А.В. Акимов // Труды молодых учёных факультета компьютерных наук ВГУ. – 2023. – Вып. 3. – С. 308-316.
2. Бурдюг, П. С. Оценка эффективности использования модуля внимания для решения задачи описания изображений с помощью специализированных метрик / П.С. Бурдюг, А.В. Акимов // Информатика: проблемы, методы, технологии. – 2024. – С. 619-628.
3. Vaswani, A. Attention Is All You Need / A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, I. Polosukhin // NIPS. – 2017. – DOI: 10.48550/arXiv.1706.03762.
4. Geva, M. Transformer Feed-Forward Layers Are Key-Value Memories / M. Geva, R. Schuster, J. Berant, O. Levy // CoRR. – 2020. – DOI: 10.48550/arXiv.2012.14913.
5. Chen, X. Microsoft COCO Captions: Data Collection and Evaluation Server / X. Chen, H. Fang, T. Lin, R. Vedantam, S. Gupta, P. Dollár, C. Zitnick // CoRR. – 2015. – DOI: 10.48550/arXiv.1504.00325.
6. Papineni, K. BLEU: a method for automatic evaluation of machine translation / K. Papineni, S. Roukos, T. Ward, W. Zhu. // Proceedings of the 40th Annual Meeting on Association for Computational Linguistics. – 2002. – P. 311-318. – DOI: 10.3115/1073083.1073135.

Современные нейронные сети в задаче стегоанализа

Д. А. Вакулин

Студент магистр

А. Ю. Иванков

Доцент

Введение

Стеганография является важной наукой в области информационной безопасности и конфиденциальности. Она изучает методы скрытия информации в каких-либо данных, например, изображениях, звуковых файлах, видео и т.п. В свою очередь, стегоанализ изучает методы выявления факта передачи скрытой информации.

На данный момент стеганография активно используется в коммерческих, финансовых и государственных организациях. Обычно в форме встраивания цифровых водяных знаков, позволяющих обеспечить защиту авторских прав, легальность транзакций, достоверность и конфиденциальность любых данных.

Однако стеганография также может быть использована злоумышленниками, например, в целях кражи конфиденциальных данных и распространения нелегального контента.

С учетом достижения методами поверхностного обучения, таких как метод опорных векторов и ансамблевые классификаторы на основе бустинга и бэггинга, предельной эффективности выявления факта наличия скрытой информации, актуальными для исследования являются методы стегоанализа на основе глубокого обучения, так как данное подмножество машинного обучения является самым перспективным и быстро развивающимся на данный момент.

В данной работе рассмотрены современные алгоритмы стегоанализа и их особенности.

1. Реализация алгоритма, нечувствительного к разрешению входного изображения

Рассмотрим последние работы, посвященные стегоанализу глубокими нейронными сетями. Авторы работы [1] предложили архитектуру нейронной сети, изображенную на рис. 1.

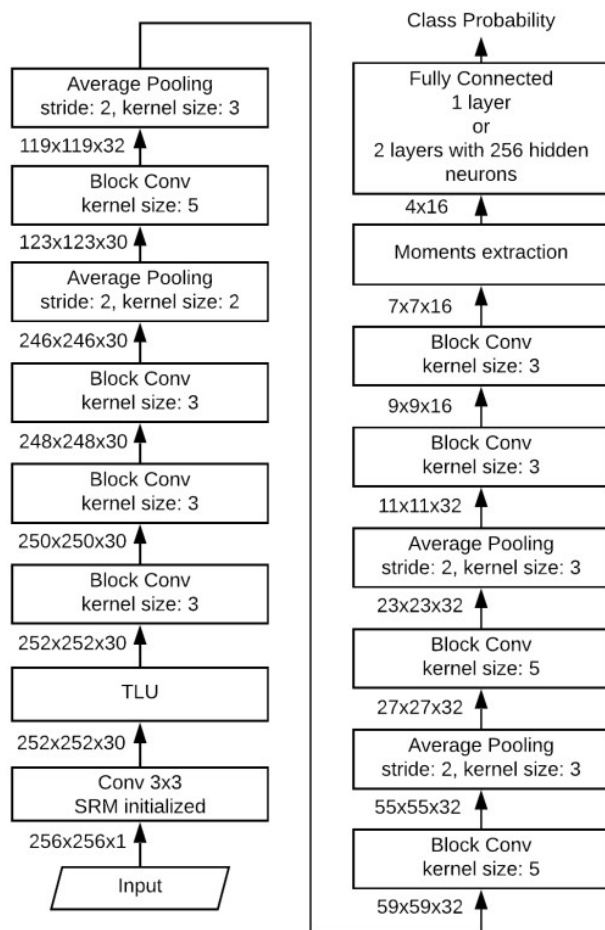


Рис. 1. Архитектура нейронной сети

Особенностью данной архитектуры является использование слоя извлечения моментов карт признаков. Под моментами подразумевается набор из четырех величин: поэлементного минимума, поэлементного максимума, поэлементного среднего, поэлементной дисперсии.

Данные моменты позволяют производить классификацию, точность которой будет слабо коррелировать с разрешением входного изображения. Авторы считают, что из-за самоподобия естественных изображений, передняя часть нейронной сети (набор сверточных слоев) будет являться универсальным экстрактором признаков, поэтому

достаточно переобучать только слой извлечения моментов, чтобы адаптировать сеть к классификации изображений с различными разрешениями.

Результаты данной работы показывают, что данный алгоритм стегоанализа, обученный одновременно на изображениях с разрешением 256x256, 512x512 и 1024x1024 пикселей, отлично справляется с классификацией стегоконтейнеров с разрешением 1024x1024 и меньше. Результаты приведены в табл. 1.

Таблица 1

Точность классификации изображений с произвольным разрешением. Полезная нагрузка 0.12 бит/пиксель для WOW и скорость изменения 0.01 для LSBM

Алгоритм стеганографии	256x256	512x512	1024x1024
LSBM	0.757	0.837	0.915
WOW	0.741	0.803	0.871

2. Сиамские нейронные сети

Авторы работы [2] предложили архитектуру нейронной сети SiaStegNet, представленную на рис. 2.

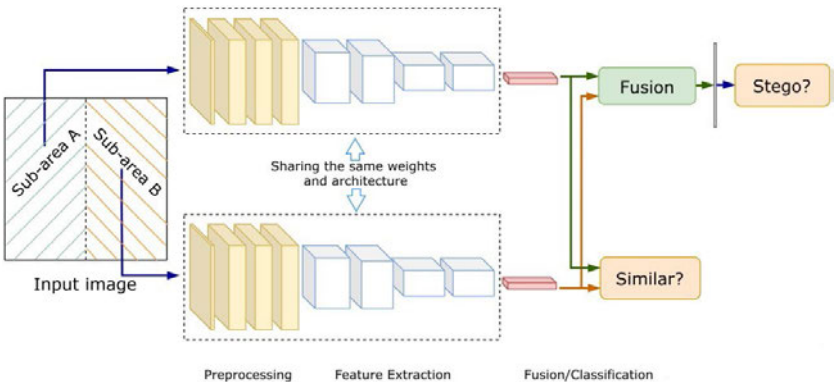


Рис. 2. Архитектура SiaStegNet

Особенностью данной архитектуры является использование сиамской нейронной сети, внутри которой две нейронных сети извлекают признаки из поданных на вход подобластей изображения. На выходе каждой сети получается карта признаков остаточных шумов

изображения. Данные карты признаков классифицируются с использованием двух управляющих сигналов.

Первый сигнал – сигнал классификации, который характеризует отношение подобластей к изображению, не содержащему стегосообщение или изображению, содержащему стегосообщение. Данный слой обучается с использованием перекрестной энтропии.

Второй сигнал – сигнал сходства, характеризует похожесть подобластей. Данный слой обучается с использованием контрастной потери, основанной на евклидовом расстоянии.

Подход с использованием сиамской нейронной сети позволяет добиться удовлетворительной точности классификации при сравнительно небольшом количестве обучаемых параметров. Результаты сравнения данной сети с SRNet [3] приведены в табл. 2.

Таблица 2

Сравнение точностей алгоритмов стегоанализа (в процентах)

Алгоритм стегоанализа	Алгоритм стеганографии	pl=0.1	pl=0.2	pl=0.3	pl=0.4
SRNet	WOW	76.97	85.93	89.89	92.51
	S-UNIWARD	72.27	83.09	88.75	92.22
	HILL	69.02	77.30	82.99	85.57
SiaStegNet	WOW	76.16	85.57	89.91	92.09
	S-UNIWARD	72.99	83.29	88.43	91.89
	HILL	69.17	77.26	82.38	85.97

3. Архитектура Convolutional Vision Transformer

Авторы работы [4] предложили нейронную сеть CVTStego-Net, показанную на рис. 3, основанную на архитектуре Convolutional Vision Transformer (CVT).

Особенностью данной нейронной сети является включение в архитектуру блока CVT, изображенного на рис. 4. Данное решение объясняется наличием у CVT всех достоинств классического трансформера, а также наличием свойств сверточных сетей, которые позволяют моделировать пространственные отношения, фиксируя глобальные зависимости. Также CVT могут моделировать зависимость локальных и глобальных остатков шума изображений со стегосообщением.

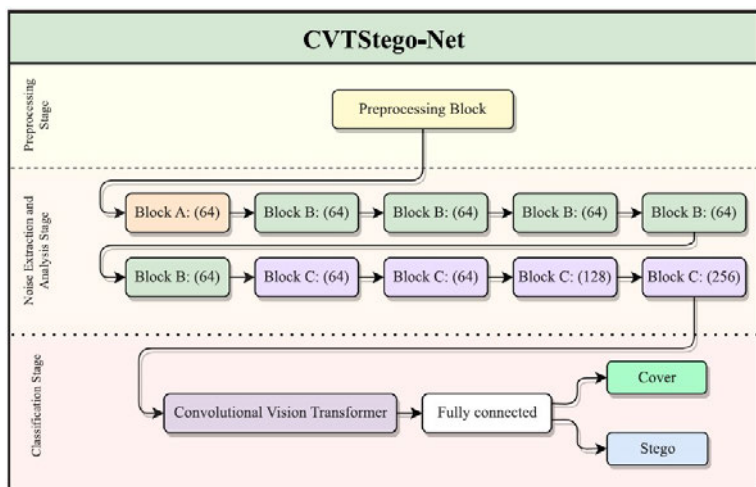


Рис. 3. Архитектура CVTStego-Net

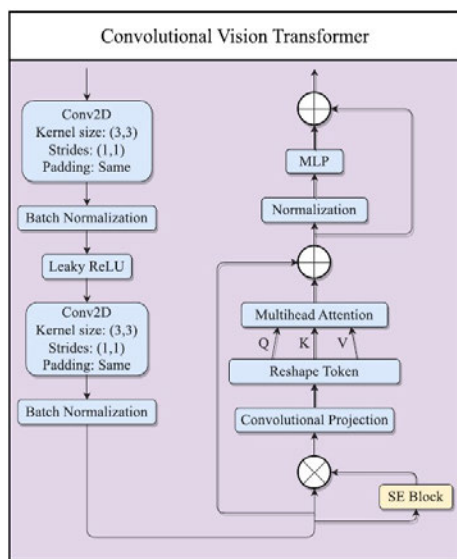


Рис. 4. Архитектура CVT-блока

Также авторы исследовали влияние наличия предобработки и CVT-блока на точность классификации стегоконтейнеров. Результаты приведены в табл. 3.

Таблица 3

*Влияние наличия блоков предобработки и CVT-блока на точность
(в процентах) классификации стегоконтейнеров*

Алгоритм стеганографии	Полезная нагрузка	Точность без предобработки	Точность без CVT-блока
WOW	0.2	53.2	82.0
	0.4	69.1	90.2
S-UNIWARD	0.2	51.1	76.4
	0.4	65.5	87.6
HILL	0.2	50.0	73.9
	0.4	59.1	83.8

Можно заметить, что наличие блока предобработки изображения тридцатью SRM ядрами существенно влияет на точность классификации.

Результаты сравнения данного алгоритма стегоанализа с другими алгоритмами, основанными на использовании сверточных нейронных сетей приведены в табл. 4.

Таблица 4

Сравнение точности алгоритмов стегоанализа (в процентах)

Алгоритм стегоанализа	WOW		S-UNIWARD	
	pl=0.2	pl=0.4	pl=0.2	pl=0.4
Yedroudj-Net	72.3	85.1	63.5	77.4
SR-Net	75.5	86.4	67.7	81.3
Zhu-Net	76.9	88.1	71.4	84.5
CVTStego-Net	86.58	93.8	80.7	90.45

Заключение

Данная статья посвящена исследованию современных глубоких нейронных сетей в задаче стегоанализа. Были рассмотрены три алгоритма стегоанализа: SID, SiaStegNet и CVTStego-Net. Среди их особенностей можно выделить использование CVT-блоков, а также сиамской нейронной сети для одновременного извлечения из двух частей изображения поэлементных минимумов, максимумов, средних и дисперсий, которые можно использовать для построения алгоритма, точность которого слабо зависит от размера входного изображения. Также рассмотрены влияния наличия предобработки входного изображения SRM ядрами и CVT-блоков на точность определения наличия в изображении стегосообщения.

Список литературы

1. Tsang, C. F. Steganalyzing Images of Arbitrary Size with CNNs / C. F. Tsang, J. Fridrich // IS&T International Symposium on Electronic Imaging Science and Technology. – 2018. – С. 121-128.
2. You, W. A Siamese CNN for Image Steganalysis / W. You, H. Zhang, X. Zhao // IEEE Transactions on Information Forensics and Security. – 2020. – № 16. – С. 291-306.
3. Boroumand, M. Deep Residual Network for Steganalysis of Digital Images / M. Boroumand, M. Chen, J. Fridrich // IEEE Transactions on Information Forensics and Security. – 2018. – № 14. – С. 1181-1193.
4. CVTStego-Net: A convolutional vision transformer architecture for spatial image steganalysis / M. A. Bravo-Ortiz [et al.] // Journal of Information Security and Applications. – 2024. – № 81. – С. 1-10.

Анализ рынков NFT и NFT игр

В. И. Валеев

Студент магистр

В. С. Тарасов

Старший преподаватель

Введение

NFT игры являются довольно молодым и вместе с этим до сих пор недостаточно раскрытым направлением в разработке. Однако у них видится большой потенциал ввиду ряда факторов: наличия концепции P2E (play-to-earn), то есть заработка во время игры, возможности однозначного подтверждения на владение активом и принципиальной особенности самих NFT – невзаимозаменяемости (отсюда и их название – non-fungible token). Соответственно, необходим анализ предметной области с целью подтверждения или опровержения актуальности проведения разработки, выделения трендов и формирования базовых концепций игры. В частности, какие игры с наибольшей вероятностью заинтересуют аудиторию, какие хорошие и плохие практики можно выделить в существующих проектах. Отличительной чертой NFT игр являются NFT, поэтому важно проанализировать и оценить именно их текущее состояние. Стоит подчеркнуть, что такого рода анализ очень важен с точки зрения геймдизайна, а хорошо продуманный геймдизайн во многом определяет успех проекта.

1. Анализ рынка NFT

За 2023 год произошло падение капитализации и объема торгов почти в 2 раза, как можно заметить на рис. 1а. Однако, динамика за последние 3 месяца положительная, что подтверждает рис. 1б. Из этого следует, что рынок NFT в целом недостаточно стабилен и сложен с точки зрения оценивания экономических рисков при внедрении NFT в какие-либо программные продукты.

Стоит сказать, что для разработки NFT игры наиболее интересна другая статистика, а именно данные о держателях NFT. Как видно на рис. 2, этот показатель практически стабильно растет, к тому же очень заметен линейный рост.

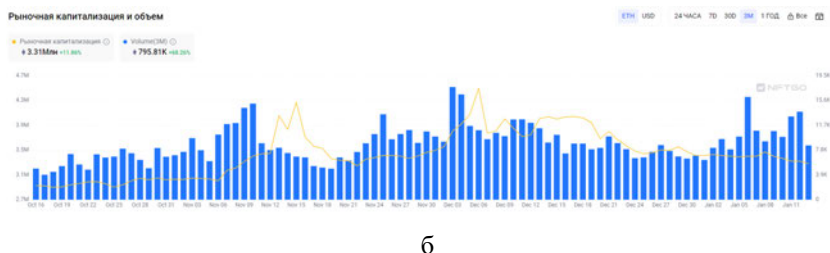
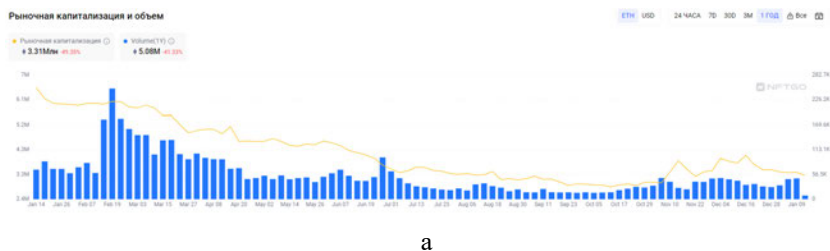


Рис. 1. Динамика рыночной капитализации и объема торгов NFT: а – за 2023 год, б – за последний квартал 2023 года

Также стоит обратить внимание, что держателей всего около 6.5 миллионов человек, что даже меньше промилле населения планеты, и из этого следует, что NFT игра должна быть больше ориентирована на обычных игроков, которых по скромным оценкам на пару порядков больше [1], а NFT должны быть ненавязчивым дополнением.

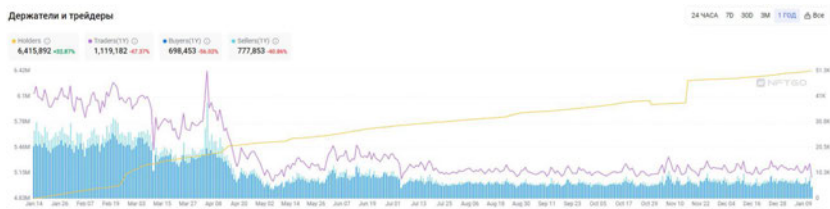


Рис. 2. Динамика держателей и трейдеров NFT за 2023 год

Также стоит отметить, что согласно рис. 3 игровые NFT в сравнении с прочими обладают наилучшей ликвидностью, что обуславливает их актуальность.

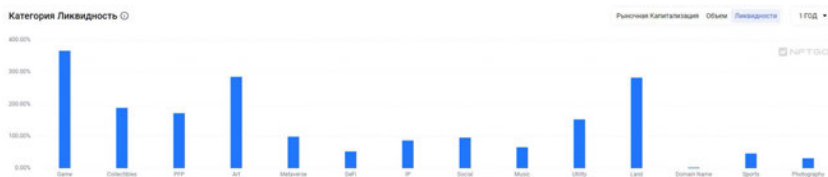


Рис. 3. Ликвидность NFT в различных категориях

Однако по рис. 4 видно, что объем торгов игровых NFT на данный момент довольно мал. Это говорит о том, что рынок таких игр относительно других категорий скуден, и у новой NFT игры есть хорошие возможности продвигать свои внутренние NFT путем вовлечения пользователей в игровой процесс.



Рис. 4. Объем торгов NFT в различных категориях

2. Анализ рынка NFT игр

В этом разделе будет рассмотрен ряд NFT игр с принципом P2E, в которые можно начать играть буквально с нуля, то есть практически без стартовых вложений, что важно в контексте привлечения игровой аудитории. Также стоит отметить, что в 2024 году ожидается выход NFT игр от крупных студий, с большими бюджетами и качественной проработкой [2], что уже говорит о достаточной привлекательности данной сферы индустрии.

Axie Infinity – игра с карточными боями при помощи разнообразных питомцев (акси). Для того, чтобы к ней присоединиться, нужно купить хотя бы трех акси. Однако на текущий момент это немалая инвестиция для новичка. С учетом популярности игры, зверюшки стоят от 20 до 200 долларов на игровом маркетплейсе. В сообществе игры сформировалась тенденция помощи начинающим игрокам, когда один пользователь спонсирует покупку трех первых персонажей, а новичок должен какое-то время отдавать часть заработанных во время игры токенов SLP инвестору, это еще называют стипендиальной программой. Найти пользователей, желающих предоставить стипендию, можно в сообществах Reddit или Discord. Но

надо быть готовым к тому, что сначала зададут несколько вопросов по игре и вообще по блокчейну, чтобы понять уровень знаний. Таким образом, можно сказать, что есть возможность начать играть без стартовых вложений. У игры есть собственные токены: AXS, SLP. Рыночная капитализация AXS составляет порядка 4 миллиардов долларов, что является огромным показателем среди прочих NFT игр. Из недостатков можно выделить вышеупомянутую стипендиальную программу. Новичков может отталкивать необходимость искать спонсора и договариваться с ним о взаимодействии.

Splinterlands – карточная игра с элементами торговли, работающая на блокчейне Hive. Начать играть можно совершенно бесплатно, но потом можно торговать своими игровыми предметами. При желании вложения возможны, чтобы разблокировать получение наград. Игра имеет сюжет, основанный на древней мифологии. Игра была ранее известна как Steem Monsters. Геймплей представляет собой сражения с вражескими монстрами в обмен на награды. Кроме того, игроки могут повышать награды, комбинируя несколько карт. Рыночная капитализация игрового токена SPS составляет порядка 70 миллионов долларов. Из недостатков стоит отметить нестабильную работу игры. Она зависает, моментами не прогружается страница. Помимо этого, обнаружена плохая поддержка разными браузерами. Также нет русской локализации, что только отчасти можно компенсировать встроенным в браузер переводчиком.

Thetan Arena – игра в жанре MOBA, которая в конце 2021 года установила рекорд – собрала всего за две недели после запуска 6 млн активных пользователей. Стилистически она похожа на популярную MOBA League of Legends с ее изометрической перспективой. Thetan Arena работает на Binance Smart Chain. Экономика игры основана на внутриигровой монете Thetan (THC), а также эксклюзивной валюте Thetan Gem (THG), используемой для развития героев. При регистрации дается один стандартный герой, так что можно начать играть без вложений. Однако со временем в процессе развития можно обзавестись более мощными премиальными героями. Рыночная капитализация THG составляет порядка 170 миллионов долларов [3]. Стартовые игровые персонажи не являются NFT, и в ходе исследования было выяснено, что с их помощью невозможно зарабатывать, и возникает необходимость покупать NFT персонажей. Другими словами, новичок сталкивается с обычной MOBA игрой, которая уступает в проработанности знаменитым Dota 2 и League of Legends, и особенность игры проявляется только при реальных вложениях.

Из игр от российских разработчиков можно выделить только Waves Ducks, которая сравнительно не популярная (с рыночной капитализацией токена EGG менее миллиона долларов [4]) и требует стартовых вложений.

Заключение

В ходе работы был проведен анализ рынков NFT и NFT игр, в результате которого можно сказать, что разработка NFT игры на данный момент имеет перспективы. С точки зрения привлечения игровой аудитории наибольший положительный эффект ожидается в случае внедрения NFT как ненавязчивую игровую часть, например, внешний вид игровых предметов или награды за необязательные задания. Стоит отметить, что наиболее популярны NFT игры с казуальным уклоном, и это следует учитывать при разработке. Также практически в каждой рассмотренной игре только условно можно играть без стартовых вложений. В целях наибольшего привлечения аудитории к блокчейну и NFT предлагается сделать NFT финансово максимально доступными, в дальнейшем же в зависимости от успеха проекта и экономической динамики вводить некоторые ограничения, так как будет возрастать риск гриндинга [5] и ботоводства [6], что с большой вероятностью негативно скажется на рынке внутриигровых NFT.

Список литературы

1. DFC: Global game audience reaches 3.7 billion [Электронный ресурс]. – Режим доступа: <https://www.gamesindustry.biz/dfc-global-game-audience-reaches-37-billion>
2. Лучшие NFT игры в 2024 году – Игры с выводом денег без вложений [Электронный ресурс]. – Режим доступа: <https://ru-crypto.com/luchshie-nft-games>
3. 7 лучших NFT-игр без вложений в 2024 году – проекты GameFi, не требующий инвестиций [Электронный ресурс]. – Режим доступа: <https://proinvestment.com/nft-games-without-attachments/>
4. Курс Waves Ducks, график EGG и рыночная капитализация [Электронный ресурс]. – Режим доступа: <https://www.coingecko.com/ru/Криптовалюты/waves-ducks>
5. Definition:Grinding | Game Difficulty Wiki | Fandom [Электронный ресурс]. – Режим доступа: <https://game-difficulty.fandom.com/wiki/Definition:Grinding>
6. Ботоводство в играх заработок [Электронный ресурс]. – Режим доступа: <https://pro-investing.ru/botovodstvo-v-igrah-zarabotok/>

Сравнительный анализ текстов, созданных генеративной сетью и человеком

А. А. Ворсунов

Студент магистр

В. А. Каладзе

Профессор

Введение

Развитие машинного обучения и искусственного интеллекта за последние годы привело к значительному прогрессу в области генеративных сетей, которые могут создавать контент, практически неотличимый от произведенного человеком. Такие системы, как DALL-E и Midjourney, демонстрируют способность создавать высококачественные изображения, конкурирующие с работами профессиональных художников. Этот успех вызывает множество дискуссий о будущем различных профессий и о том, смогут ли нейронные сети заменить человека в тех или иных сферах деятельности.

Особенно остро феномен генеративных сетей стоит в образовательной среде. С появлением ChatGPT многие академические задачи утратили свою актуальность, так как недобросовестные ученики могут использовать эту технологию для мгновенного решения поставленных задач [1].

Проблема различения текстов, созданных человеком и генеративными сетями, возникла одновременно с появлением этих технологий. Тем не менее, существующие модели классификации не всегда справляются с данной задачей. Это связано с тем, что люди создают тексты, изображения и другие творческие работы по-разному, не следуя определенным шаблонам. Например, сочинения, написанные людьми, могут быть как эмоциональными, так и безэмоциональными, что затрудняет их классификацию. Более того, генеративные сети продолжают стремительно развиваться, и модели классификации, построенные для распознавания таких текстов, быстро устаревают.

Основой работы ChatGPT является система, основанная на генеративной нейронной сети GPT. Для проведения исследования в данной работе использовалась версия ChatGPT, работающая на основе

модели GPT-3.5, предоставленной ресурсом OpenAI [2]. GPT представляет собой архитектуру трансформера, ключевой особенностью которой является улучшенный «механизм внимания». Этот механизм имитирует биологическое восприятие человеком последовательных наборов информации, что позволяет нейронной сети более эффективно обрабатывать и генерировать текст.

GPT является представителем архитектуры трансформер, особенность которых заключается в улучшенном «механизме внимания» [3]. Данный механизм имитирует биологическое восприятие человеком последовательных наборов информации.

1. Отличия во введении

Отвечая на вопрос, ChatGPT старается дать максимально формализованный и структурированный ответ, рассмотрев вопрос с разных сторон. Началом сочинения является введение, которое призвано ознакомить читателя с проблемой. Проблему необходимо сформулировать и описать её значимость. ChatGPT может формулировать и описывать значимость проблемы, но сеть делает это не из личного опыта. Данная особенность может позволить легко отличить текст ChatGPT от текста человека.

Примеры введений, сгенерированных ChatGPT, часто характеризуются следующими особенностями:

Формализованность и структурированность: Тексты ChatGPT обычно имеют чёткую структуру и последовательное изложение. Введение часто содержит несколько абзацев, каждый из которых посвящён отдельному аспекту проблемы. Это делает текст логичным и последовательным, но иногда излишне формальным и "безжизненным".

Отсутствие личного опыта и субъективных мнений: ChatGPT не обладает личным опытом или мнениями, что отражается в его текстах. Введение, написанное человеком, может включать личные анекдоты, мнения или эмоциональные вкрапления, которые придают тексту индивидуальность. В текстах ChatGPT такие элементы отсутствуют, что делает их более нейтральными и обобщёнными.

Широкий охват и детализированность: ChatGPT стремится охватить как можно больше аспектов проблемы, предоставляя пользователю максимально полное представление. Это часто приводит к включению большого количества информации и деталей, что может делать текст перегруженным. Введение, написанное человеком, может быть более лаконичным и сфокусированным на ключевых аспектах проблемы.

Тон и стиль: Стиль текста ChatGPT часто академичен и нейтрален. Введение может содержать множество сложных терминов и выражений,

что делает его похожим на научные статьи. Введение, написанное человеком, может быть более разнообразным по стилю и тону, отражая индивидуальные особенности автора и его подход к изложению материала.

Отвечая на вопрос, ChatGPT старается дать максимально формализованный и структурированный ответ, рассмотрев вопрос с разных сторон. Началом сочинения является введение, которое призвано ознакомить читателя с проблемой. Проблему необходимо сформулировать и описать её значимость. ChatGPT может сформулировать и описывать значимость проблемы, но сеть делает это не из личного опыта.

2. Эмоции

Ключевым отличием в тексте может послужить эмоциональный окрас. В связи с тем, что модели машинного обучения не обладают в полной мере сознанием, которым обладает человек, эмоции неподвластны генеративным сетям [4]. Модели могут имитировать их, но не могут в полной мере понимать и осознавать их. Касательно вопросно-ответной системы ChatGPT модель старается дать нейтральный, безэмоциональный текст. Человек же описывает эмоции, используя определённые речевые обороты.

Например, в сочинениях на тему «Интернет – это добро или зло?», человек употребляет такие выражения, как «печальная статистика». В выводах человек подчёркивает необходимость, чтобы высказывания «были реальными, живыми и доброжелательными». ChatGPT же в свою очередь просто подробно описывает плюсы и минусы наличия интернета.

Кроме того, тексты, написанные человеком, часто содержат субъективные мнения, которые делают текст более живым и эмоционально насыщенным. Например, обсуждая влияние интернета, человек может написать: «Я помню, как радовался первым возможностям общаться с друзьями по электронной почте, но со временем осознал, сколько времени теряю на бесполезный серфинг в сети». Такие личные вкрапления придают тексту индивидуальность и эмоциональную глубину.

ChatGPT, хотя и может включать эмоционально окрашенные фразы, делает это на основе шаблонов и вероятностных моделей, а не из личного опыта или искреннего чувства. Это делает эмоциональные элементы в его текстах менее естественными и более предсказуемыми.

3. Словарный запас и ошибки

ChatGPT обучается на огромном количестве текстов, в связи с чем словарный запас данной модели очень обширен. ChatGPT может употреблять слова и обороты, не свойственные, к примеру, ученику средней школы. Однако, данный аргумент недостаточно объективен: необходим анализ лексики конкретного ученика. Кроме того, важно выявление признаков, отличающих текст, написанный человеком от машинно-сгенерированного текста.

Как правило, в рамках обучения ошибки считаются чем-то негативным. Однако, ошибки свойственны любому человеку. Относительно сочинений это может быть пунктуация, орфография и т. д. Данные ошибки не свойственны ChatGPT. поэтому совокупность факторов, включая полное отсутствие ошибок, может помочь корректно классифицировать текст.

4. Позиция и субъективные оценки

Во всех сочинениях ChatGPT приводит аргументы как за одну сторону, так и за другую. Модель старается дать объективную позицию, не занимая какую-либо сторону однозначно. Это объясняется тем, что модель обучена на обширном наборе данных, содержащем множество различных точек зрения, и её цель – предоставить сбалансированный и нейтральный ответ. Таким образом, тексты, сгенерированные ChatGPT, часто включают в себя аргументы как "за", так и "против", стремясь к объективности и всестороннему рассмотрению вопроса.

В то же время человеку свойственно иметь собственное мнение и аргументированно занимать конкретную сторону в жизненных ситуациях и конфликтах. В сочинениях, написанных людьми, можно часто наблюдать выражение личных убеждений и предпочтений. Человек склонен выбирать одну из сторон и строить свою аргументацию, исходя из собственного опыта, ценностей и убеждений. Например, обсуждая тему "Интернет – это добро или зло?", человек может занять чёткую позицию, утверждая, что "Интернет в основном приносит вред, потому что вызывает зависимость и снижает качество личного общения", и далее подкреплять своё мнение конкретными примерами и эмоциями.

5. Примеры текстов

В данном разделе представлены фрагменты текстов на определённую тему, написанными людьми (сверху на белом фоне) и ChatGPT (снизу на чёрном фоне)

но есть и обратная сторона медали. С помощью денежных средств можно спасти человеку жизнь. Например, оплатить дорогостоящее лечение. Можно осуществить свою мечту. Обеспечить нормальный уровень жизни, т.е. хорошее питание, а следовательно - здоровье.

Я думаю, косвенно, деньги могут сделать человека счастливым, например, помочь осуществить мечту, улучшить здоровье или спасти кого-то. Но это только косвенно, потому что человека в этом случае делают счастливым не деньги, а нечто другое, что можно получить с помощью денежных средств. Сами же деньги сделать человека счастливым не могут, т.к. это просто бумажные купюры, которые не способны подарить чувства, ценные моменты, друзей и семью.

Таким образом, можно заключить, что деньги сами по себе не являются гарантией счастья. Они могут способствовать улучшению качества жизни и предоставить определённый уровень комфорта и безопасности, но не могут заменить важные нематериальные аспекты человеческого счастья. Истинное счастье достигается через баланс между материальными и духовными ценностями, через гармоничные отношения с окружающими, чувство самореализации и внутренний покой.

В конечном счёте, деньги – это лишь инструмент, который может помочь в достижении счастья, но не является его источником. Мы должны помнить о важности нематериальных ценностей и стремиться к тому, чтобы наша жизнь была богата не только материальными благами, но и искренними эмоциями, глубокими отношениями и внутренней гармонией.

Рис. 1. Фрагменты сочинений на тему «Деньги – счастье?»

Как видно на рисунке 2, текст человека отличается наличием формулировки “я думаю”.

Понятия добра и зла сложно разделить и определить отдельно друг от друга. Ведь они базируются друг на друге. Аналогично, они неотделимы друг от друга в реальном мире. Они одинаково присутствуют в каждом человеке, и разница заключается только в том, как каждый конкретный человек их в себе воспитывает.

Для подтверждения своих слов приведу историю из собственного опыта. Она произошла в прошлом году. В тот день я сильно поспорил с своим лучшим другом, что сильно повлияло на моё настроение. Негативное влияние оказывал также и тот факт, что я не полностью выздоровел после болезни, а мне нужно было идти на приём в поликлинику.

Понятия добра и зла являются одними из самых фундаментальных и важных в человеческой культуре, философии и религии. На протяжении веков они обсуждались философами, теологами, писателями и мыслителями, и до сих пор продолжают быть актуальными в нашей повседневной жизни. Эти понятия не только определяют наше поведение, но и формируют наши ценности и моральные ориентиры.

Добро обычно ассоциируется с положительными качествами, такими как справедливость, милосердие, честность и любовь. Оно направлено на создание и поддержание гармонии в обществе, на помощь и поддержку другим людям. Делаящие добро стремятся улучшить жизнь других, действуют бескорыстно и руководствуются принципами сострадания и уважения к человеческому достоинству. Например, добровольная помощь нуждающимся, забота о больных и поддержка слабых являются яркими проявлениями добра.

Рис. 2. Фрагменты сочинений на тему «Добро и зло»

Заключение

В рамках данной работы с помощью ChatGPT было сгенерировано более 50 сочинений на разные темы. Найденные отличия и закономерности можно использовать для анализа и распознавания работ, написанных ChatGPT. Однако, опираться только на данные особенности некорректно: каждый человек индивидуален, и педагогу необходимо принимать решения, основываясь не только на данном исследовании, но и на личном опыте и динамике обучения учащегося.

В рамках исследования было выявлено, что наиболее существенное отличие можно заметить во введении и заключении. Это связано с тем, что введение и заключение играют ключевую роль в структуре текста, поскольку они формируют первое и последнее впечатление о работе и влияют на общее восприятие читателем. Введение должно заинтересовать аудиторию, представить проблему и обозначить цели исследования, в то время как заключение должно подвести итоги, подчеркнуть значимость полученных результатов и предложить дальнейшие направления исследования или применения полученных знаний. Поэтому любые отличия в формулировках, стиле или содержании во введении и заключении могут являться ключевыми признаками, позволяющими отличить текст, созданный человеком, от текста, сгенерированного искусственным интеллектом.

Вместе с этим вскрылась основная проблема: академический и научный текст отличить крайне сложно, а в некоторых случаях даже не представляется возможным. В академических текстах принято соблюдать нейтралитет, исключать эмоциональные обороты и заведомо делать текст “сухим”. В текущих реалиях следует пересмотреть данные концепции, т. к. в таком формате тексты неотличимы.

Данное исследование направлено не только на поиск отличия между текстами ChatGPT и текстами человека. Исследование имеет своей целью подчеркнуть проблему образования. По мере развития, генеративные сети потребляют всё больше информации и всё лучше справляются с поставленными задачами. В ближайшее время, вполне вероятно, что генеративные сети смогут создавать тексты полностью неотличимыми от написанных человеком. В связи с этим уже сейчас необходимо пересмотреть формат некоторых подходов в образовании.

Список литературы

1. Нейросеть ChatGPT написала диплом российскому студенту [Электронный ресурс] : Электронный портал новостей – Режим доступа : https://www.cnews.ru/news/top/2023-02-01_nejroset_chatgpt_napisala_diplom

2. ChatGPT [Электронный ресурс] : OpenAI – Режим доступа: <https://chat.openai.com/>

3. Каладзе, В. А. Поддержка специализированного стиля текста лингво-трансформером / В. А. Каладзе, А. А. Ворсунов // Проблемы теории и практики инновационного развития и интеграции современной науки и образования : материалы IV Международной междисциплинарной конференции (Москва, 15 февраля 2023 г.) - Москва, 2024. – С. 244-249.

4. Consciousness, Cognition and Behavior [Электронный ресурс] : Электронный журнал – Режим доступа : https://www.researchgate.net/publication/341597666_Consciousness_Cognition_and_Behavior

Анализ устойчивости оптимального решения задачи нахождения критического пути с нечеткими параметрами

И. В. Горбачева

Студент магистр

Н. А. Алейникова

Доцент

Введение

В сетевом планировании проект представляет собой совокупность выполнения определенных задач, связанных последовательностью активностей, необходимых для достижения конечной цели. Эти активности могут быть разделены на более простые задачи (операции), которые выполняются последовательно или параллельно, в зависимости от требований проекта. При этом могут быть ограничены какие-либо ресурсы, необходимые для выполнения всего проекта.

Поскольку управление проектом происходит на этапе планирования, то сложно заранее указать точные сроки выполнения операций, из которых он состоит, количество затрачиваемых ресурсов. Поэтому для оценивания параметров задачи сетевого планирования используют либо методы статистического оценивания в условиях стохастической неопределенности, либо нечеткие множества в условиях нечеткой неопределенности.

В данной работе рассматривается задача сетевого планирования о нахождении критического пути в условиях нечеткой неопределенности. Параметры задачи, такие как сроки операций, заданы с помощью нечетких треугольных чисел. В работах [1, 2] предложена W -алгебра для решения задач с нечеткими параметрами, заданными с помощью нечетких LR-чисел, к которым можно отнести и треугольные числа. Представление нечетких параметров в виде W -чисел позволяет свести решение нечеткой задачи сетевого планирования к решению двух четких задач, при условии, что группы базисных переменных оптимальных решений этих задач совпадают. В статье [1] доказывается, что этого достаточно для устойчивости решения. Но если решение не устойчиво, необходимо более детально исследовать, где это условие нарушается. В данной статье предлагается исследовать устойчивость

решения с помощью теории двойственности для задач линейного программирования, так как в формализованном виде задачи сетевого планирования часто можно представить в виде задачи линейного программирования.

Таким образом, целью работы является изучение устойчивости решения нечеткой задачи сетевого планирования с помощью теории двойственности.

1. Алгоритм решения задачи нечеткого сетевого планирования с помощью W-чисел

Пусть $G = (n, P)$ сеть (граф), где n – множество событий (вершин), P – множество работ (дуг). Критический путь – это последовательность работ в проекте, которая определяет минимальное время, необходимое для завершения всего проекта. Этот путь определяет, какие задачи являются критическими для соблюдения сроков выполнения проекта. Если хоть одна работа на критическом пути задерживается, весь проект будет задержан. Также на сетевом графе критический путь обозначает самый длительный путь от начального до конечного события проекта, учитывая зависимости между работами и их продолжительность. Формальная постановка задачи сетевого планирования о нахождении критического пути в виде задачи линейного программирования следующая:

$$F = y_n - y_1 \rightarrow \min, \quad (1)$$

$$y_j - y_i \geq t_{ij}, \forall (i, j) \in P, \quad (2)$$

где y_i – моменты начала операции (i, j) , t_{ij} – длительность операции (i, j) .

При нечетких продолжительностях операций $\tilde{t}_{ij}$, заданных в виде треугольных нечетких чисел $\tilde{t}_{ij} = (t_{ij}^L, t_{ij}^m, t_{ij}^R)$, где t_{ij}^m – мода, t_{ij}^L, t_{ij}^R – левая и правая границы носителя нечеткого числа соответственно, задача (1)-(2) примет вид:

$$\tilde{F} = \tilde{y}_n - \tilde{y}_1 \rightarrow \min, \quad (3)$$

$$\tilde{y}_j - \tilde{y}_i \geq \tilde{t}_{ij}, \forall (i, j) \in P. \quad (4)$$

В [2] предлагается решать такого рода задачи с помощью W-чисел, что значительно упрощает решение.

W-число – это вектор $(v^L(\alpha); v^R(\alpha))$, где $v^L(\alpha) \in X$, $v^R(\alpha) \in X$ и $v^L(1) = v^R(1)$ (совпадают значения при $\alpha=1$, α – вещественный

параметр – соответствует понятию α -уровня нечеткого множества [2], $\alpha \in [0; 1]$).

Треугольные нечеткие продолжительности операций могут быть представлены в виде W-чисел:

$$\tilde{t}_{ij} = (t_{ij}^L(\alpha), t_{ij}^R(\alpha)) = (t_{ij}^L + (t_{ij}^m - t_{ij}^L)\alpha, t_{ij}^R + (t_{ij}^m - t_{ij}^R)\alpha), \quad (5)$$

Алгоритм решения задачи с нечеткими параметрами, представленными в виде W-числа следующий:

Шаг 1. Решить задачу отдельно для каждого значения.

Шаг 2. Полученные решения записать в виде W-чисел.

Шаг 3. Провести дефазификацию для получения четкого числа.

2. Исследование устойчивости решения задачи о нахождении критического пути с помощью теорем двойственности

Задачу (1)-(2) назовем прямой, ее нечеткий аналог – задача (3)-(4).

В [3] приведены правила, с помощью которых можно перейти к двойственной задаче.

Двойственная задача к (3)-(4) в нечеткой постановке будет следующей:

$$\tilde{Z} = \sum_{(i,j) \in P} \tilde{t}_{ij} \tilde{x}_{ij} \rightarrow \max, \quad (6)$$

$$\sum_{(i,j) \in P} \tilde{x}_{ij} - \sum_{(j,k) \in P} \tilde{x}_{jk} = 0, \quad \sum_{(1,j) \in P} \tilde{x}_{1j} = 1, \quad \sum_{(i,n) \in P} \tilde{x}_{in} = 1. \quad (7)$$

$i, j \in 0, 1$ – интерпретируется как величина потока, который проходит по дуге (i, j) . Если величина потока равна 1, то по данной дуге проходит критический путь.

Задачи (3)-(4) и (6)-(7) являются несимметричными двойственными задачами линейного программирования, так как прямая задача имеет ограничения типа неравенств, а двойственная задача – ограничения типа равенств.

Связь между оптимальными решениями взаимодвойственных задач можно установить с помощью теорем двойственности [3].

С помощью теории двойственности можно провести анализ чувствительности оптимального решения прямой задачи к изменению ее параметров, например, свободных членов ограничений [3].

В статье [4] приводится понятие устойчивости. Под устойчивостью будем понимать свойство сохранения оптимального решения на всех α -уровнях нечетких чисел от 0 до 1. В задаче нахождения критического пути это означает, что критический путь не будет меняться при всех $\alpha \in [0; 1]$. Для неустойчивой задачи на различных α -уровнях могут

быть разные критические пути. Анализ устойчивости на всех α -уровнях позволяет оценить, как меняется или остается неизменным критический путь в зависимости от степени нечеткости. Это может помочь в принятии решений, управлении рисками и понимании того, как различные факторы влияют на результаты проекта.

Согласно [1] задача имеет устойчивое решение если:

- она имеет решение для всех значений параметров на всех α -уровнях при $\alpha \in [0;1]$;

- каждая компонента решения непрерывная функция на промежутке $\alpha \in [0;1]$.

Для более полного исследования устойчивости будем использовать связь между двойственными задачами и третью теорему двойственности.

Представив свободные члены ограничений (4) в виде W -чисел, получаем задачу параметрического программирования, где параметром является α . Проведем исследование чувствительности решения в зависимости от параметра α отдельно для левой и правой компоненты W -числа по следующему алгоритму:

1. Найдем оптимальное решение прямой задачи (3)-(4) при $\alpha=1$.

2. Определим базисные и свободные переменные оптимального решения двойственной задачи (6)-(7), пользуясь второй теоремой двойственности.

3. Рассмотрим целевую функцию двойственной задачи с коэффициентами, представленными левыми компонентами W -чисел.

4. Выразим целевую функцию двойственной задачи (6) через свободные переменные оптимального решения.

5. Определим из системы неравенств, при каких значениях α оптимальное решение двойственной задачи на максимум не изменится. Для этого достаточно, чтобы все коэффициенты при свободных переменных оптимального решения оставались неположительными.

6. Аналогично проходим шаги 3-5 для случая, когда коэффициенты целевой функции представлены правыми компонентами W -чисел.

7. Если диапазон изменения α , при котором оптимальное решение двойственной задачи на максимум не изменяется, включает отрезок $[0; 1]$, то делаем вывод об устойчивости решения. Если это не выполняется, то определяем значения α , при которых устойчивость нарушается.

3. Пример применения двойственных задач при анализе устойчивости решения

Рассмотрим алгоритм исследования устойчивости задачи о нахождении критического пути на примере проекта постройки некоторого сооружения, исходные данные для которого представлены в таблице.

Таблица

Исходные данные

Операция /Код (i,j)	Описание	Пред- шествен- ник	Нечеткая продолжительность операций (дней), $\tilde{t}_{ij}$			Продолжительность в виде W-числа
			t_{ij}^L	t_{ij}^m	t_{ij}^R	
A (1,3)	Установить взрывные снаряды		3	5	8	$(3 + 2\alpha; 5 - 3\alpha)$
B (1,2)	Эвакуировать окружение		2	4	5	$(2 + 2\alpha; 5 - \alpha)$
C(1,4)	Подготовить колонну грузовиков		2	3	6	$(2 + \alpha; 6 - 3\alpha)$
D(3,4)	Взорвать здание	A, B	1	1	3	$(1 + 0\alpha; 3 - 2\alpha)$
E (4,5)	Разборка развалин и вывоз мусора	C, D	5	7	9	$(5 + 2\alpha; 9 - 2\alpha)$
F(5,6)	Вырыть котлован	E	9	12	14	$(9 + 3\alpha; 14 - 2\alpha)$
G (5,7)	Подвести коммуникации	E	13	15	17	$(13 + 2\alpha; 17 - 2\alpha)$
H (6,11)	Заливка бетона в фундамент	F	8	10	12	$(8 + 2\alpha; 12 - 2\alpha)$
I (7,8)	Возвести металлический каркас	F, G	6	8	10	$(6 + 2\alpha; 10 - 2\alpha)$
J (8,11)	Электромонтажные работы	I	12	15	18	$(12 + 3\alpha; 18 - 3\alpha)$
K (8,9)	Настелить пол и возвести стены	I	16	20	25	$(16 + 4\alpha; 25 - 5\alpha)$
L(8,10)	Установить лифты	I	6	7	9	$(6 + \alpha; 9 - 2\alpha)$
M (11,12)	Отделочные работы	H, J, K, L	12	14	16	$(12 + 2\alpha; 16 - 2\alpha)$

Для проекта запишем прямую задачу (3)-(4) в нечеткой постановке:

$$\tilde{F} = \tilde{y}_{12} - \tilde{y}_1 \rightarrow \min, \quad (8)$$

$$\left\{ \begin{array}{lll} \tilde{y}_2 - \tilde{y}_1 \geq \tilde{t}_{12}, & \tilde{y}_3 - \tilde{y}_1 \geq \tilde{t}_{13}, & \tilde{y}_4 - \tilde{y}_1 \geq \tilde{t}_{14}, \\ \tilde{y}_3 - \tilde{y}_2 \geq \tilde{t}_{23}, & \tilde{y}_4 - \tilde{y}_3 \geq \tilde{t}_{34}, & \tilde{y}_5 - \tilde{y}_4 \geq \tilde{t}_{45}, \\ \tilde{y}_6 - \tilde{y}_5 \geq \tilde{t}_{56}, & \tilde{y}_7 - \tilde{y}_5 \geq \tilde{t}_{57}, & \tilde{y}_7 - \tilde{y}_6 \geq \tilde{t}_{67}, \\ \tilde{y}_{11} - \tilde{y}_6 \geq \tilde{t}_{6,11}, & \tilde{y}_8 - \tilde{y}_7 \geq \tilde{t}_{78}, & \tilde{y}_9 - \tilde{y}_8 \geq \tilde{t}_{89}, \\ \tilde{y}_{10} - \tilde{y}_8 \geq \tilde{t}_{8,10}, & \tilde{y}_{11} - \tilde{y}_8 \geq \tilde{t}_{8,11}, & \tilde{y}_{11} - \tilde{y}_9 \geq \tilde{t}_{9,11}, \\ \tilde{y}_{11} - \tilde{y}_{10} \geq \tilde{t}_{10,11}, & \tilde{y}_{12} - \tilde{y}_{11} \geq \tilde{t}_{11,12}. \end{array} \right. \quad (9)$$

Решаем в соответствии с алгоритмом.

После решения этой задачи при $\alpha=1$, получим, что критический путь данного проекта составляют работы: А, D, Е, G, I, К, М. Длительность критического пути составит 70 дней.

Запишем двойственную задачу (6)-(7) для проекта:

$$\begin{aligned} \tilde{Z} = & \tilde{t}_{12}\tilde{x}_1 + \tilde{t}_{13}\tilde{x}_2 + \tilde{t}_{14}\tilde{x}_3 + \tilde{t}_{23}\tilde{x}_4 + \tilde{t}_{34}\tilde{x}_5 + \tilde{t}_{45}\tilde{x}_6 + \tilde{t}_{56}\tilde{x}_7 + \\ & + \tilde{t}_{57}\tilde{x}_8 + \tilde{t}_{67}\tilde{x}_9 + \tilde{t}_{6,11}\tilde{x}_{10} + \tilde{t}_{78}\tilde{x}_{11} + \tilde{t}_{89}\tilde{x}_{12} + \tilde{t}_{8,10}\tilde{x}_{13} + \\ & + \tilde{t}_{8,11}\tilde{x}_{14} + \tilde{t}_{9,11}\tilde{x}_{15} + \tilde{t}_{10,11}\tilde{x}_{16} + \tilde{t}_{11,12}\tilde{x}_{17} \rightarrow \max, \end{aligned} \quad (10)$$

$$\left\{ \begin{array}{lll} -\tilde{x}_1 - \tilde{x}_2 - \tilde{x}_3 = -1, & \tilde{x}_{10} + \tilde{x}_{14} + \tilde{x}_{15} + \tilde{x}_{16} - \tilde{x}_{17} = 0, \\ \tilde{x}_2 + \tilde{x}_4 - \tilde{x}_5 = 0, & \tilde{x}_1 - \tilde{x}_4 = 0, & \tilde{x}_{13} - \tilde{x}_{16} = 0, \\ \tilde{x}_6 - \tilde{x}_7 - \tilde{x}_8 = 0, & \tilde{x}_3 + \tilde{x}_5 - \tilde{x}_6 = 0, & \tilde{x}_{17} = 1, \\ \tilde{x}_8 + \tilde{x}_9 - \tilde{x}_{11} = 0, & \tilde{x}_7 - \tilde{x}_9 - \tilde{x}_{10} = 0, \\ \tilde{x}_{12} - \tilde{x}_{15} = 0, & \tilde{x}_{11} - \tilde{x}_{12} - \tilde{x}_{13} - \tilde{x}_{14} = 0, \end{array} \right. \quad (11)$$

$$\left\{ \begin{array}{ll} -\tilde{x}_1 - \tilde{x}_2 - \tilde{x}_3 = -1, & \tilde{x}_1 - \tilde{x}_4 = 0, \\ \tilde{x}_2 + \tilde{x}_4 - \tilde{x}_5 = 0, & \tilde{x}_3 + \tilde{x}_5 - \tilde{x}_6 = 0, \\ \tilde{x}_6 - \tilde{x}_7 - \tilde{x}_8 = 0, & \tilde{x}_7 - \tilde{x}_9 - \tilde{x}_{10} = 0, \\ \tilde{x}_8 + \tilde{x}_9 - \tilde{x}_{11} = 0, & \tilde{x}_{11} - \tilde{x}_{12} - \tilde{x}_{13} - \tilde{x}_{14} = 0, \\ \tilde{x}_{12} - \tilde{x}_{15} = 0, & \tilde{x}_{13} - \tilde{x}_{16} = 0, \end{array} \right. \quad (12)$$

По второй теореме двойственности получаем, что основным переменным двойственной задачи будут соответствовать ограничения в прямой задаче, которые на оптимальном решении прямой задачи выполняются как равенства.

Подставив оптимальное решение в ограничения (9), получим, что ограничения под номерами 2, 4, 5, 6, 8, 9, 11, 12, 15, 16, 17 выполняются как равенства. Тогда основные (базисные) переменные задачи (10)-(12): $x_2, x_4, x_5, x_6, x_8, x_9, x_{11}, x_{12}, x_{15}, x_{16}, x_{17}$, и неосновные (свободные): $x_1, x_3, x_7, x_{10}, x_{13}, x_{14}$.

Время выполнения каждой работы представлено в таблице в виде W-числа. Каждая работа может изменяться на величину, которая зависит от α -уровня.

Сначала рассмотрим целевую функцию двойственной задачи (10)-(12) с коэффициентами, представленными левыми компонентами W-чисел. Выразим целевую функцию (10) через свободные переменные оптимального решения:

$$\begin{aligned} \tilde{Z} = & (2 + 2\alpha) \cdot \tilde{x}_1 + (3 + 2\alpha) \cdot (1 - \tilde{x}_1 - \tilde{x}_3) + (2 + \alpha) \cdot \tilde{x}_3 + \\ & + 0 \cdot \tilde{x}_1 + (1 + 0\alpha) \cdot (1 - \tilde{x}_3) + (5 + 2\alpha) \cdot 1 + (9 + 3\alpha) \cdot \tilde{x}_7 + \\ & + (13 + 2\alpha) \cdot (1 - \tilde{x}_7) + 0 \cdot (\tilde{x}_7 - \tilde{x}_{10}) + (8 + 2\alpha) \cdot \tilde{x}_{10} + \\ & + (6 + 2\alpha) \cdot (1 - \tilde{x}_{10}) + (16 + 4\alpha) \cdot (1 - \tilde{x}_{10} - \tilde{x}_{13} - \tilde{x}_{14}) + \\ & + (6 + \alpha) \cdot \tilde{x}_{13} + (12 + 3\alpha) \cdot \tilde{x}_{14} + 0 \cdot (1 - \tilde{x}_{10} - \tilde{x}_{13} - \tilde{x}_{14}) + \\ & + 0 \cdot x_{13} + (12 + 2\alpha) \cdot 1 = \tilde{x}_1 \cdot (-1 + 0\alpha) + \tilde{x}_3 \cdot (-2 - 1\alpha) + \\ & + \tilde{x}_7 \cdot (-4 + \alpha) + \tilde{x}_{10} \cdot (-14 - 4\alpha) + \tilde{x}_{13} \cdot (-10 - 3\alpha) + \\ & + \tilde{x}_{14} \cdot (-4 - 1\alpha) + (56 + 14\alpha). \end{aligned} \quad (13)$$

Двойственная задача (10)-(12) является задачей на максимум, поэтому для того, чтобы решение двойственной задачи не изменилось, нужно чтобы коэффициенты целевой функции двойственной задачи, выраженной через неосновные переменные, оставались неположительными, то есть:

$$\begin{cases} (-1 + 0\alpha) \leq 0, & (-4 + \alpha) \leq 0, & (-10 - 3\alpha) \leq 0, \\ (-2 - 1\alpha) \leq 0, & (-14 - 4\alpha) \leq 0, & (-4 - 1\alpha) \leq 0. \end{cases} \quad (14)$$

Решением системы является отрезок $-2 \leq \alpha \leq 4$. Так как он содержит промежуток от 0 до 1, то критический путь не изменится.

Аналогично рассмотрим случай, когда коэффициенты целевой функции представлены правыми компонентами W-чисел. В результате для правой части нечеткого числа получится:

$$\begin{cases} \alpha \leq 1,5; & -3 \leq 0; & \alpha \leq \frac{16}{3}; \\ \alpha \leq 2,5; & \alpha \leq 4,6; & \alpha \leq 3,5. \end{cases} \quad (15)$$

Критический путь не изменится при $\infty < \alpha \leq 1,5$ в правой части нечеткого числа. Можно сделать вывод об устойчивости решения на всех α -уровнях от 0 до 1. Это значит, что на всех α -уровнях критический путь остаётся неизменным.

Заключение

Таким образом, был предложен алгоритм для исследования устойчивости решения в нечеткой задаче сетевого планирования о нахождении критического пути, продолжительности операций в которой представлены в виде треугольных нечетких W-чисел. Алгоритм основан на свойствах взаимодвойственных задач линейного программирования. Была произведена апробация алгоритма на примере проекта строительства сооружения.

Список литературы

1. Матвеев, М. Г. Анализ и решение задач выбора с параметрической нечеткостью / М. Г. Матвеев // Вестник Южно-Уральского государственного университета. Сер.: Математическое моделирование и программирование. – 2015. – Т. 8, № 4. – С. 14-29.
2. Шевляков, А. О. Нечеткие W-числа и W-алгебра для решения задач с нечеткими параметрами / А. О. Шевляков, М. Г. Матвеев // Актуальные проблемы прикладной математики, информатики и механики : сб. тр. Международной научно-технической конференции (Воронеж, 18-20 декабря 2017 г.). – Воронеж, 2017. – С. 960-967.
3. Кремер, Н. Ш. Исследование операций в экономике : учебник для вузов / Н. Ш. Кремер. – 2-е изд., перераб. и доп. – Москва : Юрайт, 2011. – 430 с.
4. Воронцов, Я. А. Устойчивость решения в задаче о критическом пути с нечёткими параметрами / Я. А. Воронцов, М. Г. Матвеев // Вестник Воронежского государственного технического университета. – 2014. – Т. 10, № 6. – С. 40–43.

Алгоритмы мультиагентного обучения с подкреплением категории on-policy

Д. А. Денисова

Студент магистр

В. В. Гаршина

Доцент

Введение

Обучение с подкреплением находится в процессе внедрения в логистические процессы; крупнейшие компании, такие, как Amazon, используют технологии машинного обучения для анализа больших данных на всех этапах производственного процесса, для повышения эффективности размещения товаров на полках и уровнях складов, оптимизации маршрутов компании, управления ресурсами и погрузчиками на складах в процессе перераспределения потока грузов [1]. Тем не менее, согласно кривой Гартнера [2], мобильные роботы сортировки и роботы-погрузчики приближаются к стадии “избавления от иллюзий”, поскольку текущие алгоритмы планирования маршрутов, доставки товаров и совместной работы требуют постоянного вмешательства сотрудников и перенастройки схем перемещения. В условиях огромных складов с постоянно меняющимися расположениями полок и товаров, а также с высокой динамичностью работы, когда каждая секунда на счету, внедрение, настройка и доработка подобных технологий сопряжены со значительными трудностями, что ограничивает их повсеместное распространение.

Мультиагентные технологии, с другой стороны, активно развиваются; ведутся многочисленные разработки по их объединению с глубоким обучением с подкреплением в различных сферах применения ([3], [4], [5], [6]). Сочетание этих подходов способно изменить принципы организации работы роботов-погрузчиков в складах и обеспечить высокую эффективность в сочетании с автономностью работы.

1. Особенности задачи и среды

Модель склада является кооперативной частично наблюдаемой средой с разряженным вознаграждением (рис. 1):

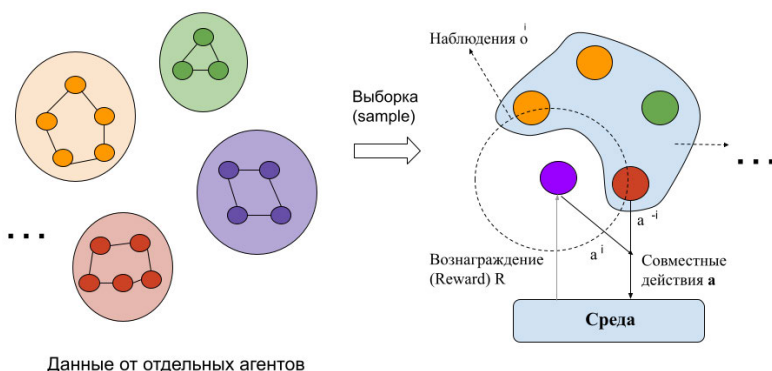


Рис. 1. Мультиагентная модель склада

Q-обучение, блестяще решающие задачи, основанные на переборе траекторий и присваивании каждой определенной значения, что позволяет выбирать наиболее эффективные действия, в подобной среде не дает значимых результатов из-за динамичной природы окружения [6].

Существующие практические подходы включают в себя прокладывание фиксированных маршрутов – с помощью меток на полу или в программе проектирования согласно схеме помещения, маршрутизацию с помощью меток RFID, QR или светоотражающих линий, SLAM-навигацию с помощью лазеров [7]. Традиционные мультиагентные алгоритмы маршрутизации с различными модификациями, которым посвящены многочисленные исследования (например, [8]-[10]), в решении производственных задач показывают низкую эффективность при увеличении числа агентов, особенно при автоматизации больших складских помещений. Сложности в данных подходах связаны в том числе с настройкой и подключением роботов к WMS (системе управления складами); их возможности совместного планирования действий ограничены, при внедрении значительного количества погрузчиков работа замедляется; стратегия работы выбирается из фиксированного набора предварительно заданных сценариев, что не всегда обеспечивает высокую производительность.

Применение обучения с подкреплением – сравнительно новый подход, тем не менее крупные компании, такие, как Amazon, утверждают, что широко используют его в своих разработках и представляют результаты на международных выставках [11]. Многие исследователи разрабатывают децентрализованные мультиагентные алгоритмы маршрутизации на основе обучения с подкрепления в частично обозреваемых средах ([12]-[14]), поскольку стратегии

маршрутизация на основе обучения с подкреплением могут быть применены к любому количеству агентов или любой карте склада без необходимости проводить переобучение [15].

Для сравнительной оценки алгоритмов мультиагентного глубокого обучения будет использована ставшая стандартом для оценки MARL алгоритмов среда RWARE со встроенным алгоритмом избегания коллизий ([6], [16]).

2. Алгоритмы категории on-policy

Изменяющаяся в реальном времени среда делает алгоритмы категории off-policy неэффективными, поскольку лучшая стратегия при предыдущих условиях больше не является оптимальной после изменения обстановки ([6], [15]). Из алгоритмов класса on-policy, в которых агент обучается на основе действий, производных от текущей стратегии, в качестве базового принципа для разработки оптимальной стратегии используется градиентный спуск по стратегиям (policy gradient) на основе методов актора-критика. Наиболее перспективным для децентрализованных марковских процессов принятия решений в среде с частичной обозреваемостью является алгоритм проксимальной оптимизации стратегии (PPO) [17] (листинг 1):

Листинг 1

Алгоритм PPO на основе подхода актор-критик

```

for iteration=1,2,... do
  for actor=1,2,..., N do
    Run policy  $\pi_{\theta_{old}}$  in environment for T timesteps

    Compute advantage estimates  $\hat{A}_1, \dots, \hat{A}_T$ 
  end for
  Optimize surrogate L wrt  $\theta$ , with K epochs and minibatch
  size  $M \leq N$ 
   $\theta_{old} \leftarrow \theta$ 
end for

```

При использовании данного подхода сначала вычисляется коэффициент эффективности стратегий на мини-выборках – так называемое advantage или преимущество, затем оптимизируется функция потерь и обновляются параметры для новой стратегии.

Обозначим a как действие, s – состояние среды, V_ϕ – функцию критика, D – множество получаемых траекторий, τ – конкретную траекторию, R – рассчитанное значение вознаграждения, A – преимущество.

Обучение критика: на каждой итерации улучшается значение value-функции

$$\phi_{k+1} = \arg \min_{\phi} \frac{1}{|D_k|T} \sum_{\tau \in D_k} \sum_{t=0}^T (V_{\phi}(s_t) - \hat{R}_t)^2. \quad (1)$$

Определение стратегии: с использованием оценки преимущества А вычисляется градиент для последующего обновления стратегии в соответствии с функцией потерь

$$L(s, a, \theta_k, \theta) = \min \left(\frac{\pi_{\theta}(a | s)}{\pi_{\theta_k}(a | s)} A^{\pi_{\theta_k}}(s, a), \right. \\ \left. clip\left(\frac{\pi_{\theta}(a | s)}{\pi_{\theta_k}(a | s)}, 1 - \epsilon, 1 + \epsilon\right) A^{\pi_{\theta_k}}(s, a) \right) \quad (2)$$

Здесь ϵ – параметр, который контролирует, насколько новая стратегия может отличаться от предыдущей, применяемый для обеспечения стабильности, π_{θ} – стратегия.

Были реализованы и рассмотрены три вида алгоритмов:

- МAA2C (мультиагентный актор-критик с преимуществом (advantage))
- MAPPO (мультиагентный алгоритм оптимизации проксимальной стратегии)
- IPPO (независимый алгоритм оптимизации проксимальной стратегии)

Первые два относятся к типу “Централизованное обучение – децентрализованное выполнение” (CTDE), у третьего – каждый агент имеет своего локального критика [3].

Рассмотрим каждый из алгоритмов детально:

1. МAA2C

Схема представлена на рис. 2. На этапе сэмплирования агенты обмениваются информацией о своих наблюдениях и предсказанных действиях.

Затем централизованный критик обновляет функцию value

$$\phi_{k+1} = \arg \min_{\phi} \frac{1}{|D_k|T} \sum_{\tau \in D_k} \sum_{t=0}^T (V_{\phi}(o_t, s_t, u_t) - \hat{R}_t)^2, \quad (3)$$

где o – наблюдение, u – объединенные действия агентов.

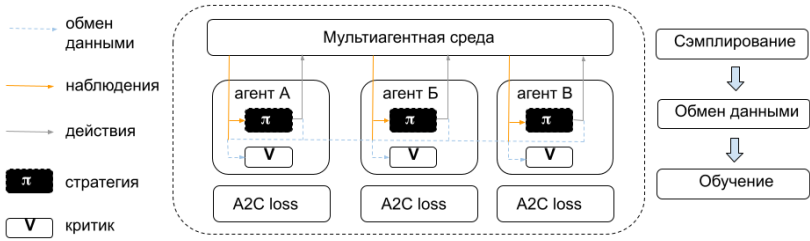


Рис. 2. Мультиагентный актор-критик с преимуществом (advantage) (MAA2C)

Через темпоральные различия вычисляется значение преимущества

$$A_t = r_{t+1} + \lambda V_\phi(o_{t+1}, s_{t+1}, u_{t+1}) - V_\phi(o_t, s_t, u_t), \quad (4)$$

где λ - параметр веса.

После чего обновляется стратегия с помощью policy gradient

$$\nabla_\theta J(\theta) \approx \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(u_t | o_t) A_t. \quad (5)$$

2. MAPPO (схема на рис. 3)

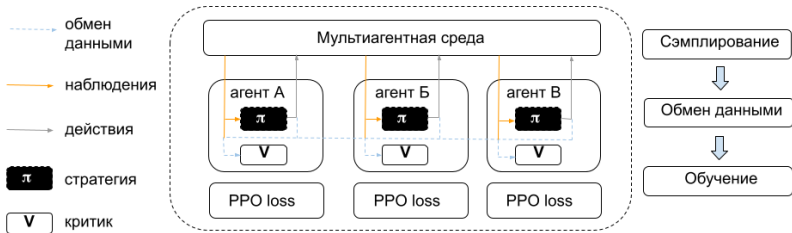


Рис. 3. Мультиагентный алгоритм оптимизации проксимальной стратегии MAPPO

Агенты также обмениваются информацией на стадии сэмплирования, используется централизованный критик. Отличия от предыдущего подхода заключаются в использовании генерализованной оценки преимущества GAE – таким образом в выборке рассматривается не одна траектория, а сразу несколько (6):

$$A_t = \sum_{l=0}^{\infty} (\gamma \lambda)^l \delta_{t+l}^V, \quad (6)$$

где γ - коэффициент дисконтирования.

Также для обновления стратегии используется функция потерь, соответствующая PPO-подходу, когда мы регулируем изменение стратегии с помощью дополнительных параметров (7):

$$L(o, s, u, \bar{u}, \theta_k, \theta) = \min\left(\frac{\pi_\theta(u | o)}{\pi_{\theta_k}(u | o)} A^{\pi_{\theta_k}}(o, s, \bar{u}), \right. \\ \left. clip\left(\frac{\pi_\theta(u | o)}{\pi_{\theta_k}(u | o)}, 1 - \epsilon, 1 + \epsilon\right) A^{\pi_{\theta_k}}(o, s, \bar{u})\right) \quad (7)$$

3. IPPO (схема на рис. 4)

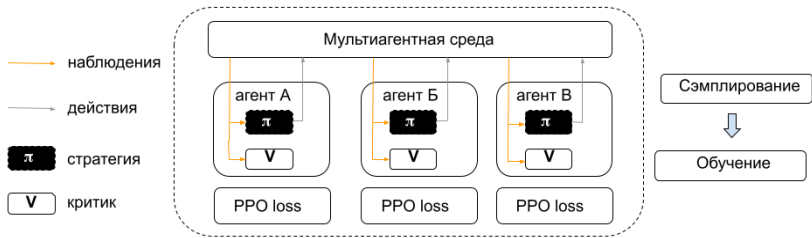


Рис. 4. Независимый алгоритм оптимизации проксимальной стратегии IPPO

В отличие от двух рассмотренных выше подходов, в данном алгоритме не осуществляется обмен информацией между агентами – у каждого агента свой децентрализованный критик. Соответственно, в формуле (3) для вычисления значения функции V будет использованы только локальные наблюдения каждого конкретного агента.

3. Экспериментальные исследования

Для проведения экспериментов были адаптированы фрагменты кода из [3], [4], [18], [19] и создана соответствующая среда; наилучший результат при построении моделей был достигнут при взятии рекомендованных параметров (табл. 1).

Были проведены три серии экспериментов в трех различных вариантах среды RWARE – tiny 2 агента, tiny 4 агента и small 4 агента, с возрастающей сложностью. На результирующих графиках (рис. 5) представлены средние значения после 2000000 эпизодов.

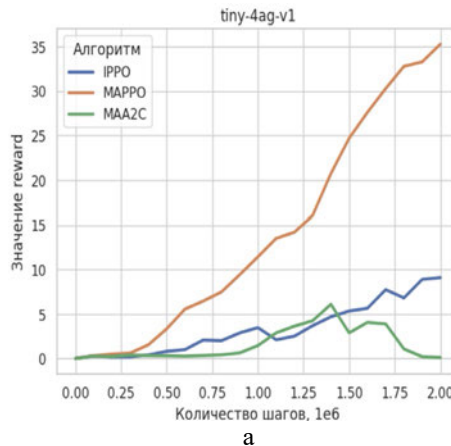
Таблица 1

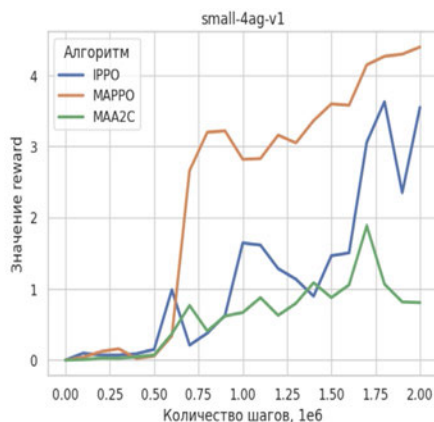
Рекомендованные параметры

Параметры	MAA2C	MAPPO	IPPO
hidden dimension	64	128	128
learning rate	0.0005	0.0005	0.0005
network type	FC	FC	GRU
entropy coefficient	0.001	0.001	0.001
target update	0.01	0.01	0.01
n-step	5	10	10

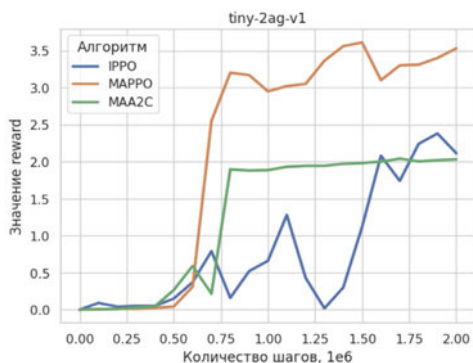
Можно отметить значительное преимущество алгоритма MAPPO во всех трех вариантах среды, хотя при большем количестве испытаний IPPO также может давать значимые результаты ([20]). Для MAA2C были отмечены эпизоды с так называемым «падением в пропасть» [3] – когда после определенного изменения стратегии общее вознаграждение упало до нуля, и модель в дальнейшем становится неэффективной; такие результаты повлияли на среднюю оценку алгоритма.

Временные затраты, представленные на рисунке 6, сопоставимы, тем не менее, для подхода on-policy и дальнейшего повышения эффективности моделей требуются значительные вычислительные ресурсы ([6], [18]).





б



в

a – tiny-4ad, б – small-4ag, в – tiny-2ag

Рис. 5. Значения reward

Закключение

Были рассмотрены алгоритмы мультиагентного глубокого обучения с подкреплением категории on-policy; реализована среда и проведены эксперименты в упрощенной модели склада для оценки их эффективности для задачи маршрутизации роботов-погрузчиков. Наилучшие результаты показал алгоритм MAPPO; для дальнейших исследований и возможного применения в производственных задачах можно предложить его дальнейшие модификации – оптимизация параметров, применение секвенционального обучения, усложнение

среды, предоставление значительных вычислительных ресурсов и т.д. В дальнейшем данная технология может быть использована для современных автоматизированных систем управления складами и схожих задач.

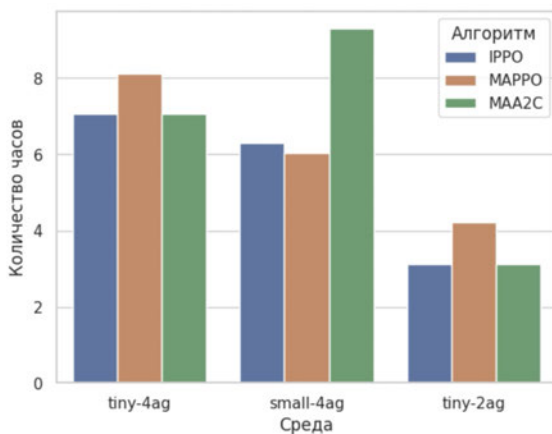


Рис. 6. Временные затраты алгоритмов

Список литературы

1. How Amazon deploys collaborative robots in its operations to benefit employees and customers [Электронный ресурс]. – Режим доступа: <https://www.aboutamazon.com/news/operations/how-amazon-deploys-robots-in-its-operations-facilities>
2. Gartner Hype Cycle Shows Supply Chain Adoption of Mobile Robots Will Far Outpace Drones Over Next Three Years [Электронный ресурс]. – Режим доступа: <https://www.gartner.com/en/newsroom/press-releases/2023-08-17-gartner-hype-cycle-shows-supply-chain-adoption-of-mobile-robots-will-far-outpace-drones-over-next-three-years>
3. MARLlib: A Scalable and Efficient Multi-agent Reinforcement Learning Library [Электронный ресурс]. – Режим доступа: <https://marllib.readthedocs.io/en/latest/>
4. Extended Python MARL framework – EPyMARL [Электронный ресурс]. – Режим доступа: <https://github.com/uoe-agents/epymarl>
5. JaxMARL: Multi-Agent Reinforcement Learning in JAX [Электронный ресурс]. – Режим доступа: <https://github.com/FLAIROx/JaxMARL>
6. Денисова, Д. А. Навигация погрузчиков в логистике с использованием мультиагентного обучения с подкреплением /

Д. А. Денисова, В. В. Гаршина // Информатика: проблемы, методы, технологии: сборник материалов XXIV международной научно-практической конференции / под редакцией Д. Н. Борисова; Воронеж, Воронежский государственный университет, 14-15 февраля 2024 г. – Воронеж, ВГУ, 2024. – С. 656-664.

7. Автономные вилочные погрузчики GEEK+ [Электронный ресурс]. – Режим доступа: <https://nissa-eng.ru/katalog/vendors/geek/vilochnyj-pogruzchik/>

8. Ferner C. Optimal multirobot path planning in low dimensional search spaces / C. Ferner [et al.] // Proceedings of the 2013 IEEE International Conference on Robotics and Automation. – Karlsruhe, Germany, 6-10 May 2013. – P. 3854-3859.

9. Barer M. Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem / M. Barer [et al.] // Proceedings of the seventh International Symposium on Combinatorial Search. – Prague, Czech Republic, 15-17 August 2014. – P. 19-27.

10. Surynek P. Towards optimal cooperative path planning in hard setups through satisfiability solving / Surynek P. // PRICAI 2012 Proceedings of the twelfth Pacific Rim International Conference on Artificial Intelligence. – Kuching, Sarawak, Malaysia, 3-7 September 2012. – P. 564–576.

11. Amazon Dives Deep into Reinforcement Learning [Электронный ресурс]. – Режим доступа: <https://www.forbes.com/sites/cognitiveworld/2019/06/14/amazon-dives-deep-into-reinforcement-learning/?sh=8ce51805fe06>

12. Sartoretti G. PRIMAL: Pathfinding via reinforcement and imitation multi-agent learning / G. Sartoretti [et al.] // IEEE Robotics and Automation Letters. – 2019. – Vol. 4. – Issue 3. – P. 2378-2385.

13. Wang B. Mobile robot path planning in dynamic environments through globally guided reinforcement learning / B. Wang [et al.] // IEEE Robotics and Automation Letters. – 2020. – Vol. 5. – Issue 4. – P. 6932-6939.

14. Ma Z. Distributed heuristic multi-agent path finding with communication / Z. Ma, Y. Luo, H. Ma // Proceedings of the 2021 International Conference on Robotics and Automation. – Xi'an, China, 30 May – 5 June 2021. – P. 8699–8705.

15. RDE: A Hybrid Policy Framework for Multi-Agent Path Finding Problem / J. Gao [et al.] : arXiv, 2023. [Электронный ресурс]. – Режим доступа: <https://arxiv.org/abs/2311.01728>

16. RWARE: A multi-agent reinforcement learning environment [Электронный ресурс]. – Режим доступа: <https://github.com/semitable/robotic-warehouse>

17. Proximal Policy Optimization Algorithms / J. Schulman [et al.]: Computing Research Repository (CoRR), 2017. [Электронный ресурс]. – Режим доступа: <https://arxiv.org/abs/1707.06347>
18. Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks / G. Papoudakis [and all]: arXiv, 2021. [Электронный ресурс]. – Режим доступа: <https://arxiv.org/abs/2006.07869>
19. MAVEN: Multi-Agent Variational Exploration [Электронный ресурс]. – Режим доступа: <https://github.com/AnujMahajanOxf/MAVEN>
20. The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games / C. Yu, A. Velu, E. Vinitzky: arXiv, 2022. [Электронный ресурс]. – Режим доступа: <https://arxiv.org/abs/2103.01955>

Создание защищенных превью цифровых изображений с использованием стеганографических алгоритмов повышенной пропускной способности и обучаемых алгоритмов сжатия

А. С. Железной

Студент магистр

М. А. Дрюченко

Доцент

Введение

Для противодействия возможным злоупотреблениям, связанным с разрушением, кражей, несанкционированным доступом, копированием и тиражированием информации необходимо знать административные и законодательные меры в области информационной безопасности и тщательно выбирать методы и средства защиты информации. На сегодняшний день для защиты информации в компьютерных системах чаще всего применяются программно-аппаратные средства, использующие криптографические методы. Однако для отдельных приложений информационной безопасности, таких как защита авторских прав и контроль незаконного тиражирования цифровых объектов, традиционных криптографических методов может быть недостаточно. Для этих целей целесообразно использовать современные стеганографические методы. Если рассматривать классические схемы создания невидимых цифровых водяных знаков и отпечатков пальцев [1-3], то они позволяют решить указанные выше проблемы частично, оставляя злоумышленникам возможность беспрепятственного использования цифрового контента, естественно, без его размещения и тиражирования в Сети. К примерам подобных вариантов использования цифровых изображений можно отнести просмотр и использование незаконно полученных авторских изображений или изображений конфиденциального характера.

Одним из возможных подходов к защите цифровых изображений от несанкционированного использования является частичное ограничение их функциональности путем фрагментации и создания т.н. «защищенных» превью, представляющих собой небольшие фрагменты

кадров, содержащие в себе сжатые стеганографически скрытые недостающие части исходного их представления. Подобные превью могут храниться на общедоступных ресурсах или передаваться по открытым каналам телекоммуникации, но без знания стеганографических ключей и операций обратного преобразования восстановление исходных представлений соответствующих превью изображений будет невозможно. Получение авторизованными пользователями полнофункциональных изображений на основе превью реализуется путем извлечения, декомпрессии и добавления на результирующие кадры их недостающих стеганографически скрытых фрагментов. Относительные размеры формируемых превью должны быть минимальны по сравнению с размером соответствующих им исходных растровых представлений, что обуславливает необходимость использования стеганографических алгоритмов повышенной пропускной способности, а также алгоритмов, реализующих сжатие скрываемых растровых фрагментов с высокими коэффициентами без существенной потери их качества. К числу последних можно отнести современные обучаемые алгоритмы, основанные на использовании глубоких и генеративно-сопоставительных нейронных сетей [4].

1. Алгоритм создания защищенных превью цифровых изображений

В общем виде задача формирования защищенных превью может быть сформулирована следующим образом. Пусть I – исходное изображение размером $W \times H$ пикселей, $I^{(p)} \subset I$ – прямоугольный фрагмент размером $w \times h$, $w \leq W$, $h \leq H$, выбираемый в качестве превью, $I^{(1)}, I^{(2)}, I^{(3)}, I^{(4)} \subset I$ – непересекающиеся фрагменты, окружающие $I^{(p)}$, $I^{(p)} \cup I^{(1)} \cup I^{(2)} \cup I^{(3)} \cup I^{(4)} = I$ (рисунок 1).

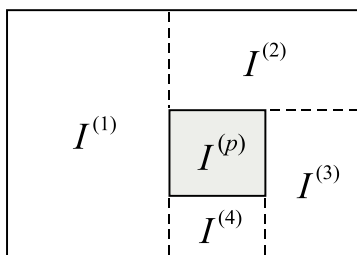


Рис. 1. Пример разметки фрагментов изображения при формировании превью

Далее с использованием скрывающего стеганографического преобразования F реализуется встраивание сжатых представлений $\{I^{(i)}\}$, $i = \overline{1,4}$ внутрь $I^{(p)}$:

$$I^{(p)} = F\left(I^{(p)}, \{J^{(i)}\}, K\right), J^{(i)} = G\left(I^{(i)}\right), i = \overline{1,4}, \quad (1)$$

где G – алгоритм сжатия, K – стеганографический ключ, $I^{(p)}$ – защищенное превью. В терминологии стеганографии $I^{(p)}$ является контейнером, $\{J^{(i)}\}$, $i = \overline{1,4}$ – скрываемым сообщением. Для повышения надежности схемы защиты относительные размеры формируемых превью должны быть минимальны по сравнению с размером соответствующих им исходных растровых представлений, т. е. необходимо минимизировать отношение площади области кадра, соответствующей превью, к общей площади исходного раstra

$$v = \frac{S(I^{(p)})}{S(I)} = \frac{wh}{WH} \rightarrow \min. \quad (2)$$

Защищенное превью $I^{(p)}$ рассматривается в качестве замены исходного изображения при его передаче, хранении и размещении на общедоступных ресурсах. Являясь небольшим фрагментом исходного кадра $I^{(p)}$ тем не менее обеспечивает возможность восстановления перцептивно близкого к исходному представлению I

$$I = I^{(p)} \cup I^{(1)} \cup I^{(2)} \cup I^{(3)} \cup I^{(4)}, I - I^{(p)} \rightarrow \min, \quad (3)$$

$$I^{(i)} = G^{-1}\left(J^{(i)}\right), \{J^{(i)}\} = F^{-1}\left(I^{(p)}, K\right), i = \overline{1,4}.$$

где G^{-1} – алгоритм декомпрессии данных, F^{-1} – восстанавливающее ранее скрытые данные стеганографическое преобразование.

2. Алгоритм стегоскрытия, используемый при создании защищенных превью

В качестве алгоритма стеганографического скрытия при создании защищенных превью рассматривался алгоритм на основе оценки сложности блоков ABCDE (A Block Complexity based Data Embedding) [5]. Данный алгоритм реализует скрытие в пространственную область контейнера в наиболее «шумные» сегменты битовых слоев путем

замены значений соответствующих двоичных разрядов сегментов битами скрываемого сообщения. Особенностью алгоритма является использование нескольких (более старших) битовых слоев контейнера при поиске подходящих для скрытия шумовых сегментов, что позволяет существенно повысить емкость встраивания.

Для определения элементов пространственной области контейнеров, пригодных для скрытия данных, алгоритм ABCDE использует две специальные метрики. Первая из них – метрика неравномерности длин серий. Она вычисляется на основе гистограмм длин серий нулевых и единичных пикселей, формируемых по строкам и столбцам блока. Метрика неравномерности длин серий имеет вид

$$\mu_s = - \sum_{i=1}^l h[i] * \log_2 p_i, \quad (4)$$

где h – гистограмма длин битовых серий, l – максимальная длина серии, $p_i = \frac{h[i]}{\sum_{j=1}^l h[j]}$.

Вторая – метрика зашумленности границ. Она оценивает разницу между соседними последовательностями пикселей в блоке и предназначена для определения границ объектов. Для блока пикселей размером $n \times n$ метрика зашумленности границ имеет вид

$$\mu_n = \min \{E_f(P(r), P(c))\} / (n-1), \quad (5)$$

где $P(r) = g(r_1, r_2) \oplus \dots \oplus g(r_{n-1}, r_n)$, $P(c) = g(c_1, c_2) \dots \oplus g(c_{n-1}, c_n)$, $\oplus$ – операция исключающего или, $g(x)$ – определяет количество единиц в последовательности x , r_i , c_i – строки и столбцы пикселей блока с заданным индексом $i = \overline{1, n}$,

$E_f(X) = \left[1.0 - D(X) / \max_X \{D(X)\} \right] M(X)$, D, M – дисперсия и математическое ожидание, $X = \{x_1, \dots, x_n\}$.

При встраивании сообщения метрики μ_s и μ_n позволяют исключить из рассмотрения блоки пикселей, соответствующие шаблонам с регулярной структурой и содержащих последовательности серий битовых значений одинаковой длины, что позволяет снизить вносимые при ССИ искажения. Примеры таких шаблонов приведены на рисунке 2в, г. На рисунке 2а приведен пример типового «шумового» блока, используемого для встраивания, на рисунке 2б – пример типового непригодного для встраивания «информативного» блока.

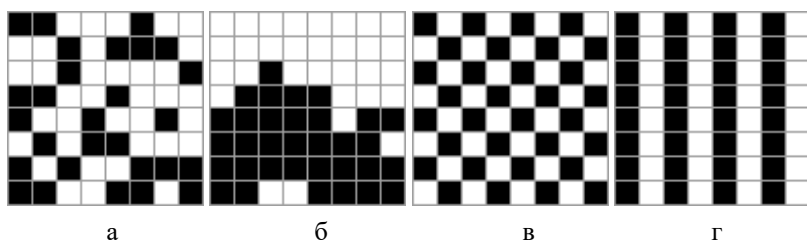


Рис. 2. Бинарные блоки изображения контейнера, имеющие разный уровень сложности

Встраивание сообщения реализуется путем замены элементов найденных бинарных шумоподобных блоков битами сжатого сообщения. Для повышения степени защищенности можно дополнительно использовать стеганографические ключи, которые, например, будут определять порядок (псевдослучайный) обхода контейнера при выборе блоков для встраивания.

3. Алгоритм сжатия изображений, основанный на использовании генеративно-состязательных сетей

В качестве высокопроизводительного алгоритма сжатия G в (1), обеспечивающего выполнение условия (2), и позволяющего получать качественные (реконструированные из превью) представления кадров, рассматривался алгоритм HiFiC (High-Fidelity Generative Image Compression) [4], основанный на использовании генеративно-состязательных нейронных сетей.

В данной работе использовался метод сжатия изображений с помощью условных ГАН-ов и основанное на теории Шеннона «скорость-искажение». Классическая модель состоит из кодировщика E и декодера G . Для кодирования изображения – получаем квантованный латент $y = E(x)$, а для декодирования – используем декодер для получения реконструкции с потерями. Хранение y достигается введением вероятностной модели P для y . Используя P и алгоритм энтропийного кодирования мы можем хранить y без потерь, используя битрейт $r(y) = -\log P(y)$ (на практике накладные расходы на бит, понесенные энтропийным кодером, пренебрежимо малы).

При параметризации E , G и P как CNN, мы можем обучать их совместно, минимизируя компромисс между скоростью и искажениями, где λ управляет компромиссом:

На рисунке 3 представлена схема работы метода сжатия с помощью модели HiFiC.

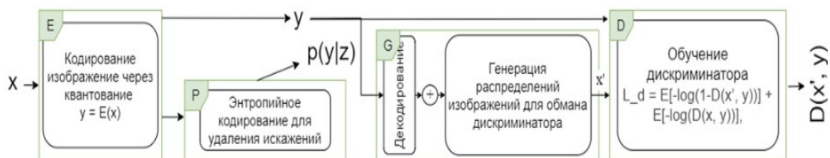


Рис. 3. Схема работы метода сжатия с помощью модели HiFiC

Достоинства:

- экономия битрейта в 2 раза в сравнении с другими алгоритмами;
- наилучшее качество сжатия изображения за счёт использования GANов;
- возможность сжатия изображений с психовизуальной и статистической избыточностью.

Недостатки:

- артефакты затемнения при использовании изображения с разрешением, отличным от размера обучающей выборки;
- необходимость использования формата BPG для наилучших результатов.

4. Тестирование алгоритма создания защищенных превью

Экспериментальный анализ разработанного алгоритма создания защищенных превью проводился с использованием выборки из 50 полноцветных фотографических изображений формата PNG из набора TESTIMAGES [6]. Оценивались следующие показатели:

- коэффициенты сжатия скрываемых в превью фрагментов изображений;
- уровень искажающих изменений реконструируемых изображений;
- отношение размеров областей кадров, соответствующих формируем превью, к размерам целых кадров.

Для оценки искажений использовались объективные критерии, основанные на сравнении исходного и восстановленного из превью изображений – пиковое отношение сигнал-шум (PSNR) и индекс структурного сходства (SSIM).

Компрессия скрываемых растровых фрагментов при формировании превью, а также их декомпрессия при восстановлении исходного представления кадров проводилась с использованием предобученных

моделей HiFiC-low, HiFiC-med, HiFiC-high в [7], обеспечивающих различные показатели степени сжатия и качества восстанавливаемых изображений. При использовании модели HiFiC-low формировались сжатые фрагменты минимального размера, декомпрессия которых давала перцептивно менее качественные результаты. При использовании модели HiFiC-high формировались сжатые фрагменты больших размеров, декомпрессия которых давала наилучший с точки зрения минимизации визуальных отличий от оригинальных изображений результат. Коэффициент сжатия (КС) скрываемых в превью фрагментов оценивался как отношение исходного размера фрагментов к их размеру после HiFiC -сжатия $КС = \frac{1}{4} \sum_{i=1}^4 |I^{(i)}| / |J^{(i)}|$. Значения КС усреднялись для всех скрываемых фрагментов каждого кадра.

Относительный размер превью (ОПР) оценивался как отношение размера заполненного фрагмента $I^{-(p)}$ к размеру исходного изображения. $ОПР = |I^{-(p)}| / |I|$. Результаты экспериментов представлены в табл. 1.

Таблица 1

Усредненные показатели качества работы алгоритма при создании защищенных превью

Модель	КС	PSNR	SSIM	ОПР
HiFiC -low	83.2	26.14	0.9017	0.09
HiFiC -med	72.6	31.25	0.9485	0.11
HiFiC -high	63.0	34.69	0.9731	0.15

Наилучшие результаты по качеству реконструируемых из превью изображений ожидаемо были получены при использовании модели HiFiC-high – усредненные значения PSNR составили порядка 34, SSIM – порядка 0.97. Для выбранного алгоритма стегоскрытия усредненные значения отношений размеров превью к размерам соответствующих им изображений варьировались от 0.09 до 0.15.

В качестве альтернативы использованию HiFiC-сжатия при создании превью могут быть рассмотрены более универсальные традиционные алгоритмы компрессии изображений такие как JPEG или JPEG2000. Однако, как показали результаты предварительных экспериментов, показатели качества работы алгоритма создания защищенных превью, получаемые при использовании данных алгоритмов сжатия, в целом уступают показателям, получаемым для

моделей HiFiC. Проведение более подробных сравнительных исследований с рассмотрением альтернативных алгоритмов сжатия выходило за рамки данной работы.

Заключение

В работе рассмотрен алгоритм защиты цифровых изображений от несанкционированного использования, основанный на частичном ограничении их функциональности путем фрагментации и создания «защищенных» превью, представляющих собой небольшие фрагменты кадров, содержащие в себе сжатые стеганографически скрытые недостающие части исходного их представления. Для минимизации размеров заполненных превью скрываемые фрагменты изображений сжимались по алгоритму HiFiC, характеризующемуся высокими коэффициентами сжатия и возможностью восстановления (с незначительными потерями информации) близких к оригиналу растровых представлений. Среди преимуществ предложенного алгоритма можно отметить следующие:

- отсутствие у неавторизованных лиц возможности полноценного использования контента (изображений в форме превью) без знания стеганографических ключей (фактически соответствующих разрешений со стороны правообладателя);
- возможность предварительного ознакомления потенциальных пользователей с изображениями в форме превью;
- относительная простота реализации.

В качестве ограничений по использованию предложенного алгоритма можно отметить невозможность его применения для защиты данных, не допускающих даже минимальных потерь качества.

Список литературы

1. Liu, K. C. Colour image watermarking for tamper proofing and pattern-based recovery / K.C. Liu // IET Image Processing. – 2012. – Vol. 6(5). – P. 445-454.
2. Coatrieux, G. A watermarking-based medical image integrity control system and an image moment signature for tampering characterization / G. Coatrieux, H. Huang, H. Shu, L. Luo // IEEE Journal of Biomedical and Health Informatics. – 2013. – Vol. 17(6). – P. 1057-1067.
3. Сирота, А.А. Нейросетевые функциональные модели и алгоритмы преобразования информации для создания цифровых водяных знаков / А.А. Сирота, М.А. Дрюченко, Е.Ю. Митрофанова // Известия высших учебных заведений. Радиоэлектроника. – 2015. – Т. 58, № 1. – С. 3-16.

4. High-Fidelity Generative Image Compression [Электронный ресурс]. – Режим доступа: <https://arxiv.org/pdf/2006.09965.pdf>
5. Hirohisa, H. A data embedding method using BPCS principle with new complexity measures / H. Hirohisa // Proc. of pacific rim workshop on digital steganography. – 2002. – P. 30-47.
6. A set of test images «TESTIMAGES» [Электронный ресурс]. – Режим доступа: <http://testimages.tecnick.com>
7. HiFIC_torch_colab_demo [Электронный ресурс]. – Режим доступа : https://colab.research.google.com/github/Justin-Tan/high-fidelity-generative-compression/blob/master/assets/HiFIC_torch_colab_demo.ipynb#scrollTo=ulxLn5j7n_an

Анализ алгоритмов скелетизации бинарных изображений

К. Д. Иванов

Студент магистр

В. В. Фертиков

Доцент

Введение

В задачах компьютерной обработки изображений часто необходимо решать проблему анализа формы объекта. Форму объекта можно описать совокупностью его границ, представленных многоугольными фигурами. Данный подход позволяет сегментировать объекты, имеющие относительно простую форму. Для более глубокого анализа формы, например, с целью определенного геометрического преобразования или деформации объекта, необходимо выделять его скелет.

Скелет – это упрощенное представление формы объекта на изображении, в виде тонких линий или кривых. Планарный граф, которым является скелетное представление, позволяет анализировать особые точки фигуры, используемые, например, для задания ограничений ее геометрического преобразования. Скелет, дополненный характеристикой расстояния каждой его точки до границы объекта (так называемое, медиальное представление), содержит всю необходимую информацию для восстановления контура.

Скелетное описание объекта удовлетворяет следующим утверждениям:

- скелет сохраняет исходную топологию представляемого им объекта
- при масштабировании объекта его скелет пропорционально увеличивается (или уменьшается);
- при повороте или переносе объекта его скелет поворачивается или переносится относительно той же центральной точки, что и сам объект.

Помимо перечисленных свойств, касающихся простых аффинных преобразований фигуры в целом, скелетное представление позволяет реализовать выборочное или различающееся преобразование отдельных фрагментов объекта в соответствии с его топологией.

1. Постановка задачи

В данной статье исследуются популярные алгоритмы скелетизации, которые применяются для построения скелета объектов на бинарных изображениях. Эти алгоритмы различаются принципами построения, вычислительной сложностью и эффективности работы, степенью устойчивости к шуму и другим характеристикам.

После применения каждого алгоритмов скелетизации к серии тестовых рассчитаны показатели точности построения скелета.

Исследование этих показателей имеет важное значение для оценки качества работы алгоритмов скелетизации, их применимости к различным типам изображений и задачам, а также для оптимизации процесса скелетизации в целом.

2. Алгоритм Зонга-Суня

Алгоритм Зонга-Суня основан на операции эрозии. Эрозия – это морфологическая операция последовательного контролируемого утоньшения изображения. Алгоритм заключается в итеративном применения масок к областям пикселей для их удаления. Алгоритм заканчивает свою работу, когда применение маски к каждому пикселю изображения не повлекло изменения изображения. [1] Блок-схема данного алгоритма представлена на рис. 1.

Идея преобразования заключается в перемещении границы фигуры со всех сторон вглубь фигуры, пока два встречных фронта не встретятся.

Результатом преобразования является не графовая структура, а новое изображение, на котором представлен дискретный скелет исходного объекта. Требуется дополнительные преобразования для выделения ребер графа, представляющих отдельные ветви скелета. Также, невозможно восстановить исходное изображение, так как информация о толщине исходных объектов была утеряна в ходе утоньшения.

При зашумленной границе, скелет изображения получается избыточным.

3. Алгоритм распространения волн

Суть алгоритма состоит в прослеживании распространения сферической волны по объекту изображения внутри границ объекта. Ребра скелета формируются перемещением центров масс волны [2].

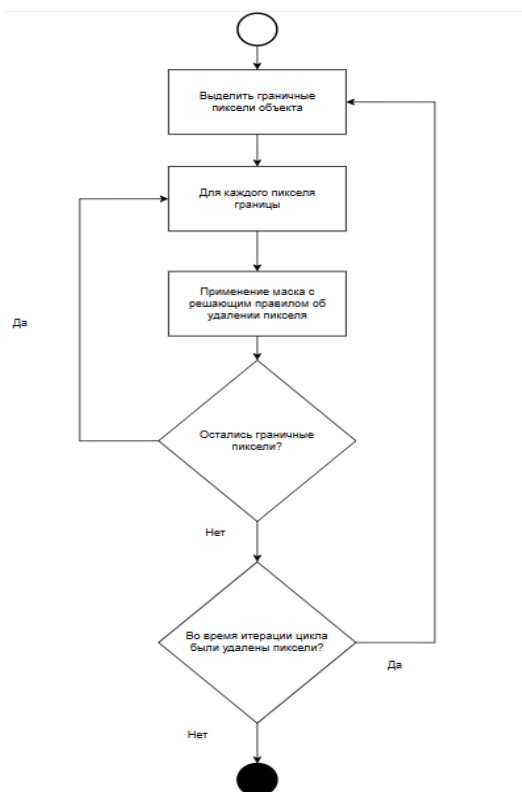


Рис. 1. Блок-схема алгоритма Зонга-Суня

Результатом прослеживания распространения волн является графовая структура, подверженная искажениям и характеризующаяся избыточным количеством ребер. Последующая обработка заключается в объединении ребер, лежащих на одной прямой. Для этого для соседних ребер в графе ищутся углы наклона. Если углы имеют отличия в рамках заданной погрешности, то эти два ребра объединяются в одно с модификацией их концевых точек. Блок-схема алгоритма представлена на рис. 3.

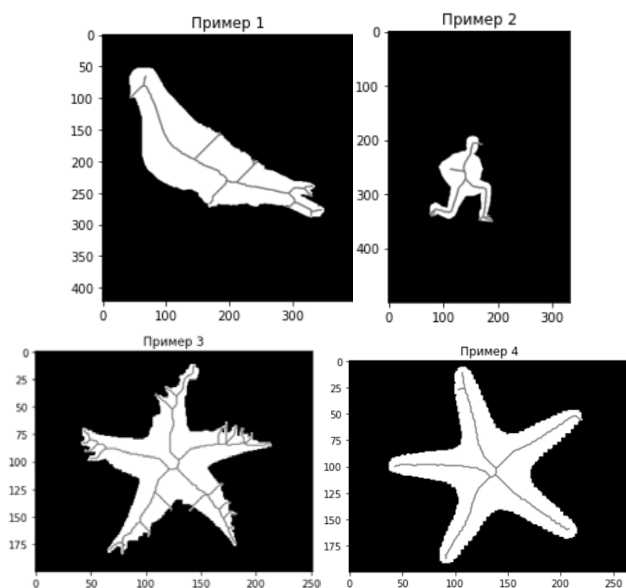


Рис. 2. Иллюстрация работы алгоритма Зонга-Суня

Алгоритм хорошо работает при анализе простых вытянутых фигур.

Алгоритм не гарантирует прохождение скелета через срединные линии фигуры в следствии резких изгибов границы или при наличии зашумленной границы.

4. Алгоритм непрерывной скелетизации

Данный алгоритм в качестве исходных данных использует непрерывное граничное представление объекта, то есть список границ фигуры, представленных ломаными линиями. Идея алгоритма заключается в построении диаграммы Вороного на основании отрезков и вершин границы фигуры. На первом этапе определяются терминальные вершины, которыми являются все выпуклые углы фигуры. На их основе, строятся активные бисекторы – будущие ребра графа скелета.

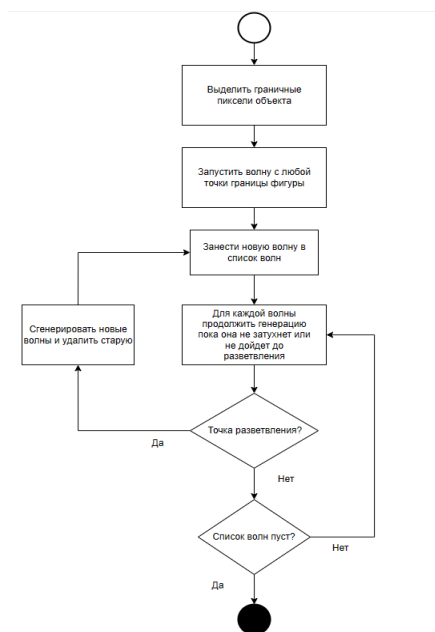


Рис. 3. Блок схема волнового алгоритма

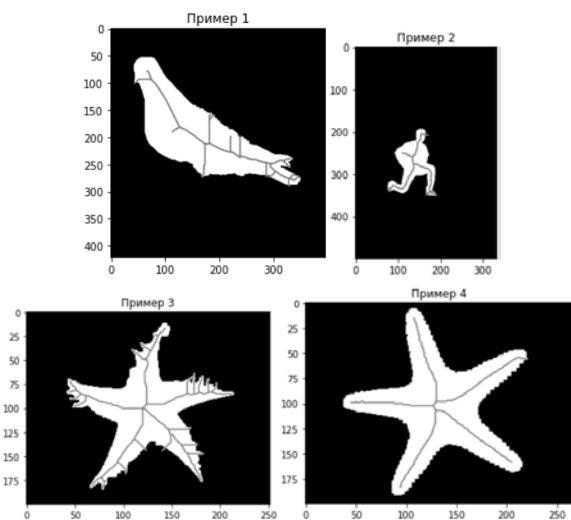


Рис. 4. Иллюстрация работы волнового алгоритма

На следующем этапе для каждого активного бисектора ищется его конечная точка. Поиск конечной точки производится посредством построения вписанной окружности для трех и более сайтов (вершин и отрезков границы фигуры). По окончании построения активного бисектора, порождаются новые на основании сайтов, которых касается вписанная окружность с центром во второй точке бисектора [3]. Блок-схема алгоритма представлена на рис. 5.

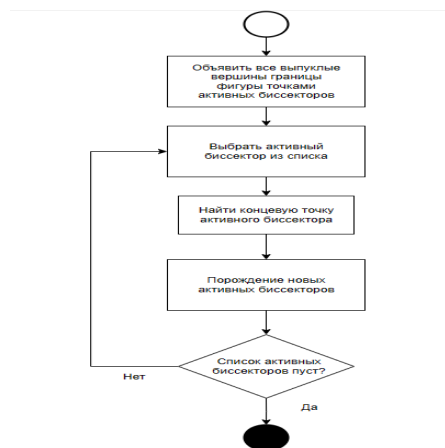


Рис. 5. Блок-схема непрерывного алгоритма

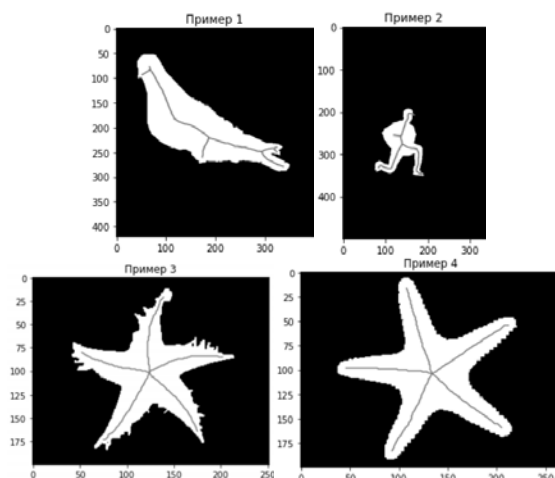


Рис. 6. Иллюстрация работы непрерывного алгоритма

Так как алгоритм работает с непрерывным представлением фигуры, то ширина ребер графа стремится к 1 пикселю. Все вершины графа являются центрами максимальных пустых окружностей. Также, все ребра являются частью диаграммы Вороного, это означает, что они равноудалены от границы изображения. Непрерывный алгоритм генерирует медиальное представление фигуры – скелет в виде графа и максимальных пустых окружностей с центрами в вершинах графа. Медиальное представление позволяет восстановить исходное изображение с минимальными потерями.

5. Оценка эффективности алгоритмов

Для оценки алгоритмов скелетизации применялись метрики толщины скелета ST и метрика избыточности связей скелета SC. [4] Чем ближе значение ST к 1, тем тоньше ребра скелета. Чем ближе значение SC к 0, тем меньше избыточности в скелете.

Средние значения оценок по 50 тестовым изображениям представлены в таблице.

Оценки рассматриваемых алгоритмов

Алгоритм	ST	SC
Зонга-Суня	0,783	0,147
Волновой	0,882	0,054
Непрерывный	0,966	0,010

Каждый из этих алгоритмов обладает своей сферой применения взаимности от требуемого уровня простоты реализации или точности скелета.

Заключение

Данная статья посвящена популярным алгоритмам скелетизации изображений. При анализе алгоритмов были выявлены сферы применимости алгоритмов, их помехоустойчивость и проведен анализ толщины и избыточности полученных скелетов. Анализ показал устойчивость алгоритма непрерывной скелетизации.

Список литературы

1. Клубков И.М. Применение волнового алгоритма для нахождения скелета растрового изображения. Вестник Донского государственного технического университета. 2001. Т. 1, № 1 (7). С. 126–133.
2. Zhang T. Y., Suen C. Y. A fast parallel algorithm for thinning digital patterns. Communications of the ACM, 1984, vol. 27, no. 3, pp. 236–239.

3. Местецкий Л.М. Непрерывная морфология бинарных изображений: фигуры, скелеты, циркуляры. – М.: ФИЗМАТЛИТ, 2009. – 113–121 с.
4. Jang B. K., Chin R. T. One-pass parallel: analysis, properties, and quantitative evaluation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1992, vol. 14, pp. 1129–1140.

Применение градиентного бустинга в задаче предсказания результатов футбольных матчей

К. А. Капкин

Студент магистр

А. В. Акимов

Старший преподаватель

Введение

Задача предсказания исхода противостояния между спортивными клубами может быть решена с помощью различных методов машинного обучения, таких как логистическая регрессия, случайный лес, нейронные сети и другие. Для наиболее точной и полной интерпретации результатов работы моделей необходимо использовать различные метрики оценки их качества, такие как точность, полнота, F1-мера, и т.д. Также в ходе подбора оптимальных параметров для каждой из выбранных моделей предсказания необходимо иметь референтные точки, от которых можно отталкиваться при принятии решения о сдвиге ряда числовых значений параметров в ту или иную сторону. Проведение кросс-валидации в качестве такой точки обеспечивает проверку стабильности модели на разных выборках данных.

Сложив воедино необходимые компоненты, получим алгоритм решения задачи предсказания исхода противостояния между спортивными клубами. В первую очередь нужно подробно разобрать предметную область и подготовить должным образом данные, затем выбрать наиболее подходящие алгоритмы машинного обучения достижения поставленной задачи и произвести моделирование. Также немаловажным этапом является проведение анализа результатов моделей предсказания и их верная интерпретация.

В рамках данной статьи подробно остановимся на задаче выбора моделей предсказания наступления одного из трех результатов противостояния команд для игры в футбол.

1. Методы машинного обучения для предсказания результатов

Любая задача с использованием средств машинного обучения требует большого внимания к подбору данных для обучения и сопоставления полученной информации от модели для дальнейшей

сверки с реальными данными. Подробно этот процесс и его составляющие в контексте задачи предсказания результатов матчей на примере игры в футбол рассмотрен в работе [1]. В рамках данного исследования рассмотрим наиболее эффективное решение в области предсказания результатов для табличных данных на основе градиентного бустинга [2].

Сфера машинного обучения пытается ответить на вопросы о том, каким образом человек может создавать компьютерные системы, которые по мере накопления опыта самостоятельно становятся лучше и каковы фундаментальные законы, которые управляют множеством процессов обучения.

Бустинг – это ансамблевый метод машинного обучения, целью которого является объединение нескольких слабых моделей предсказания для создания одной сильной. Цель бустинга – улучшить точность предсказаний.

Одной из основных особенностей бустинга является динамическое взвешивание обучающих данных. После каждого этапа обучения модели в ансамбле, данные, на которых были допущены ошибки, получают больший вес, следовательно, последующие модели уделяют больше внимания именно этим трудным случаям.

Однако бустинг может быть склонен к переобучению, особенно если в ансамбле присутствует множество сложных моделей.

LightGBM [3] – реализация градиентного бустинга с открытым исходным кодом, в которой вводятся две ключевые идеи: GOSS и EFB, благодаря которым это решение кардинально отличается от других моделей градиентного бустинга.

Градиентная односторонняя выборка (GOSS) является модификацией градиентного бустинга, которая выделяет особое внимание на таких примерах, которые приводят к большему градиенту, благодаря чему, ускоряется обучение и уменьшается вычислительная сложность метода.

Объединение взаимоисключающих признаков (EFB) – это подход объединения практически нулевых взаимоисключающих признаков, таких как категориальные переменные входных данных, закодированные унитарным кодированием. Таким образом, это тип автоматического подбора признаков.

Вместе эти два изменения могут существенно ускорить время обучения алгоритма [4], вплоть до порядка. Таким образом, LightGBM можно рассматривать как деревья решений с градиентным бустингом с добавлением GOSS и EFB.

CatBoost [5] использует решающие деревья малой глубины в качестве базовых моделей, что делает их более устойчивыми к переобучению, где каждый узел дерева делает бинарное разбиение на основе значения одного из признаков.

Основная особенность CatBoost – способность обрабатывать категориальные данные без предварительного кодирования. Другие алгоритмы требуют перевода категориальных данных в числовой формат (например, унитарное кодирование), что может потребовать большого объема памяти и привести к потере информации.

CatBoost использует счетчики для категориальных признаков. Это означает, что для каждой категории создаются числовые признаки, отражающие статистику по этим категориям. Это позволяет избежать необходимости вручную указывать, какие признаки являются категориальными. Например, для задачи классификации, это может быть доля положительных и отрицательных классов внутри каждой категории.

Резюмируя, LightGBM является наиболее эффективным при работе с большими наборами данных благодаря своему гистограммному подходу к построению деревьев и таким оптимизациям как EFB.

CatBoost выделяется своей способностью эффективно обрабатывать категориальные данные без предварительного кодирования. Он обеспечивает высокую производительность и эффективное предотвращение переобучения.

2. Обучение моделей

Общая идея использования рейтинговых моделей в задаче предсказания результатов заключается в следующем. Происходит инициализация рейтинга команд на одной отметке, определяется значения коэффициента изменчивости рейтинга. Далее на основе тестовых данных рейтинг команд актуализируется и обретает свой конечный вид для возможности предсказания будущих матчей. После чего по специальным формулам высчитываются такие параметры как рейтинги игроков или команд и их обновление. Потом через различные корреляционные коэффициенты, которые несут в себе «историю» предыдущих противостояний, осуществляется переход от разницы в рейтинге между командами к вероятности наступления одного из трех результатов противостояния команд.

Для задачи машинного обучения необходимо выделить набор данных, на которых алгоритм будет проводить обучение, и целевую переменную, которую мы желаем предсказать. Также необходимо рассмотреть вопрос о том каким образом необходимо решать поставленную задачу: как задачу регрессии или классификации.

При работе использовался язык программирования Python и реализации LightGBM и CatBoost в виде готовых библиотек. В качестве целевой переменной определены голы, забитые каждой из команд в определенном матче.

В работе используется результат предобработки набора данных из статьи [1]. Полученный датасет разделяется на тренировочную, валидационную и тестовую выборки, в качестве последней выступает сезон 2022/2023 года.

Для обучения моделей были использованы следующие признаки, применяемые к каждому отдельному матчу для каждой из сражающихся команд, рассчитываемые на основе набора данных:

- играет ли команда дома;
- среднее количество голов, которое команда забила за последние 5, 10, 20, 30 матчей;
- среднее количество голов, которое команда соперника пропустила за последние 5, 10, 20, 30 матчей;
- наименование соревнования, в рамках которого происходит встреча команд и его тип;
- среднее количество голов, которое команда забила дома или на выезде за последние 5, 10, 20, 30 матчей;
- среднее количество голов, которое команда соперника пропустила на выезде или дома за последние 5, 10, 20, 30 матчей;

Затем происходит подбор оптимальных гиперпараметров моделей для минимизации ошибок. В моделях LightGBM и CatBoost используются как отличные по наименованию, но одинаковые, по сути, параметры, так и уникальные, присущие только одной из моделей. Ниже представлен список гиперпараметров, свойственные обоим моделям:

- регрессионные или классификационные, определяющие по какому закону распределена целевая переменная;
- функция ошибок;
- метрика, используемая для обнаружения переобучения;
- скорость обучения;
- максимальное количество построенных деревьев;
- глубина каждого дерева;
- минимальное количество обучающих сэмплов в листе;
- количество листьев в дереве;
- остановка обучения при отсутствии улучшения показателей на протяжении некоего числа итераций;
- количество классов для задач мультиклассификации;
- массив категориальных признаков.

3. Анализ результатов

Таким образом, настроенные модели проходят обучение на подготовленных данных. С учетом подобранных ранее параметров для каждой модели получили следующие данные.

Решающее дерево для метода LightGBM имеет 24 листа и глубину 4, дерево сбалансированное. Для метода CatBoost решающее дерево имеет 48 листьев и глубину 6. Не смотря на различия в построении решающих деревьев для моделей, общие принципы одинаковы для обоих методов: наиболее значимые признаки, по которым можно поделить данные, располагаются на верхних уровнях дерева, менее значимые, уточняющие признаки располагаются на нижних уровнях.

Чтобы определить, какие признаки больше влияют на результат предсказания, использовалась оценка важности признаков из библиотек реализаций. Важность признаков является ориентиром при настройке моделей, позволяя графически интерпретировать количество обращений модели к конкретному признаку для разделения данных по всем деревьям в модели. Полученные данные отображают реальный расклад сил: для предсказания забытых мячей результативность на протяжении большого числа игр (средне количество забитых/пропущенных мячей за последние 30 или 20 матчей) будет иметь большее значение, чем за несколько последних игр (средне количество забитых/пропущенных мячей за последние 5 или 10 матчей) [6].

При обучении моделей и интерпретации полученных значений ожидаемых забитых мячей руководствуемся тем, что голы в футболе имеют распределение Пуассона [7]. Сумма независимых пуассоновских случайных величин (т.е. общее количество забитых каждой из команд в рамках одного конкретного матча мячей) также имеет распределение Пуассона. Однако в футболе, как и во многих игровых видах спорта, побеждает тот, кто забивает больше и пропускает меньше. Ответ на вопрос о победителе дает разница забитых и пропущенных мячей, которая имеет распределение Скеллама [8]:

$$p(k, \mu_1, \mu_2) = -\exp(-(\mu_1 + \mu_2)) \left(\frac{\mu_1}{\mu_2} \right)^{\frac{k}{2}} I_k \left(2\sqrt{\mu_1 \mu_2} \right), \quad (1)$$

где I_k – модифицированная функция Бесселя [8], k – разница в забитых мячах «домашней» команды относительно «гостевой». Просуммировав значения вероятности для положительной разницы мячей ($k > 0$), получим вероятность победы домашней команды, для нулевой разницы ($k = 0$) – вероятность ничьей, и для отрицательной ($k < 0$) – вероятность победы выездной команды. Параметры μ_1 и μ_2 – средние

значения двух пуассоновских величин. Применительно к данной задаче, это предсказанные моделями голы из результирующего набора данных, изображенного на рисунке. Для получения вероятности в формулу подставляются значения параметров `pred_home_goals` и `pred_away_goals`, пояснения к обозначениям указаны в табл. 1.

match_id	pred_home_goals	pred_away_goals	home_win_prob	draw_prob	away_win_prob	pred_result	true_result
137944	1,335825086	0,897802495	0,468731105	0,279694922	0,251573973	H	H
137945	1,264254628	0,970818679	0,430814533	0,284520987	0,28466448	H	D
137943	1,210409108	0,912453293	0,42921714	0,293178571	0,277604289	H	D
137942	1,103939252	1,213095279	0,33218104	0,281911484	0,385907476	A	A
137939	1,51314305	0,922702728	0,510328048	0,259603318	0,230068634	H	H
137940	1,351791601	0,846996233	0,48594121	0,279287016	0,234771774	H	D
137941	1,078226531	1,283643112	0,311231219	0,277392429	0,411376352	A	D
137938	1,129791832	0,966906513	0,392531628	0,298496012	0,30897236	H	H
137937	1,194581373	1,148282309	0,371094298	0,280502211	0,348403491	H	H

Рисунок. Конечный набор данных

Таблица 1

Обозначения из конечного набора данных

Название столбца	Описание
match_id	Идентификатор матча, сгенерированный на этапе предобработки датасета
pred_home_goals	Предсказанное моделью количество забитых мячей домашней командой в конкретном матче
pred_away_goals	Предсказанное моделью количество забитых мячей выездной командой в конкретном матче
home_win_prob	Вероятность победы домашней команды в конкретном матче
draw_prob	Вероятность ничьи в конкретном матче
away_win_prob	Вероятность победы выездной команды в конкретном матче
pred_result	Предсказанный моделью результат матча (H – победа домашней команды, D – ничья, A – победа выездной команды)
true_result	Реальный результат матча (H – победа домашней команды, D – ничья, A – победа выездной команды)

Результирующие матрицы ошибок для моделей LightGBM и CatBoost отображены в табл. 2 и табл. 3 соответственно. Обозначения и их смысл схожи с обозначениями свойств `pred_result` и `true_result` из

табл. 1. По горизонтали значения для истинного класса, по вертикали – для предсказанного. Символы H, D, A обозначают принадлежность к классу победа домашней команды, ничья и победа выездной команды соответственно.

По факту в матрице ошибок на главной диагонали содержится значение метрики полноты, отношение верно предсказанных величин класса к их общему количеству. Следует отметить высокую полноту для победы домашней команды при средних показателях для победы выездной команды и крайне низкая возможность предсказания ничейного исхода. Однако хороший показатель полноты для победы домашней команды является побочным проявлением плохой точности – модели присваивают большое количество ничьих и побед выездной команды в пользу домашней (нижний треугольник матрицы).

Таблица 2

Матрицы ошибок для модели LightGBM

	H-pred	D-pred	A-pred
H-true	0,8366	0,0101	0,1527
D-true	0,7493	0,0144	0,2356
A-true	0,5741	0,0156	0,4104

Таблица 3

Матрицы ошибок для модели CatBoost

	H-pred	D-pred	A-pred
H-true	0,8500	0,0099	0,1400
D-true	0,7611	0,0116	0,2272
A-true	0,5895	0,0160	0,3945

Подытоживая, суммарная точность для обеих моделей примерно равна 50,3%. Полученный результат следует интерпретировать как удовлетворительный для задачи определения одного из трех исходов в таком непредсказуемом виде спорта как футбол [9].

Заключение

В ходе работы были рассмотрены различные подходы к решению задачи предсказания результатов футбольных матчей на примере моделей машинного обучения. Также в статье описан подход с использованием моделей машинного обучения на основе деревьев решений, начиная подбора признаков для обучения и заканчивая интерпретацией полученных результатов с помощью специализированных метрик оценки качества. На выходе алгоритм дает

ответ на вопрос о том, какая команда наиболее вероятно окажется сильнее оппонента в конкретном матче.

Следует отметить, что предложенная методика предсказания количества забитых голов, где берется за основу распределение Пуассона и разницы этих величин, используемых в распределении Скеллама, может быть использована не только для игры в футбол, но и для таких игровых видов спорта как футзал и гандбол. При этом изменения, вносимые в алгоритм решения задачи предсказания, будут минимальными и при оценке качества моделей для этих видов спорта ожидается большая точность предсказания из-за того, что футбол является куда более низко результативным и случайные события имеют большее влияние на итоговый результат [9].

Список литературы

1. Капкин, К.А. Подготовка данных для обучения алгоритма градиентного бустинга в задаче предсказания результатов футбольных матчей / К.А. Капкин, А.В. Акимов // Информатика : проблемы, методы, технологии: сб. мат. XXIV межд. н.-пр. конф. (Воронеж, 14-15 февраля 2024 г.). – Воронеж, 2024 – С.710-717.
2. Deep Neural Networks and Tabular Data: A Survey / V. Borisov [et al.] // LG – 2017. – DOI: 10.48550/arXiv.2110.01889.
3. LightGBM: A Highly Efficient Gradient Boosting Decision Tree [Электронный ресурс]. – Режим доступа: https://papers.nips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf
4. So, B. Enhanced Gradient Boosting for Zero-Inflated Insurance Claims and Comparative Analysis of CatBoost, XGBoost, and LightGBM / B. So // LG – 2023. – DOI: 10.48550/arXiv.2307.07771.
5. Catboost [Электронный ресурс]. – URL : <https://catboost.ai/>
6. Does Stability Foster Team Performance [Электронный ресурс]. – Режим доступа: <https://www.professormarkvanvugt.com/images/files/DoesStabilityFosterTeamPerformance.pdf> (дата обращения 09.05.2024).
7. Medium [Электронный ресурс]. – Режим доступа: <https://medium.com/@ricardoandream/poisson-distribution-to-predict-match-results-c4032dcef949> (дата обращения 09.05.2024).
8. Gan, H. L. Approximation of the difference of two Poisson-like counts with Skellam / H. L. Gan, Eric D. Kolaczyk // PR. – 2018.
9. Choi, B.S. Predicting Football Match Outcomes With Machine Learning Approaches. / B.S. Choi, L.K. Foo, S. Chua // Mendel. – Volume 29. – №.2. – 2023. – P. 229-236.
10. Gürkan, O. Analysis of Goals Scored in UEFA Champions League by the Time Periods / O. Gürkan, G. Ertetik, S. Müniroğlu / 2017. – DOI: 10.14486/IntJSCS660.

Анализ и исследование методов прогнозирования базового индекса потребительских цен на основе многомерных временных рядов

Ю. А. Качанова

Студент магистр

Е. А. Сирота

Доцент

Введение

В последние годы происходит стремительное развитие информационных технологий, растет объем хранимых и обрабатываемых данных, в связи с этим потребность в инструментах анализа временных рядов будет только увеличиваться.

Базовый индекс потребительских цен (ССПИ) является важнейшим макроэкономическим показателем, иллюстрирующим изменение цен на основные категории товаров и услуг, приобретаемые конечными потребителями, за определенный период. Прогнозирование ССПИ является ключевым при проведении денежно-кредитной политики государства, а также позволяет определить ключевые факторы, влияющие на этот показатель. Кроме того, ССПИ является основным инструментом, используемым для вычисления и прогнозирования инфляции в США, что впоследствии позволяет оценить перспективы развития экономических процессов [1].

Целью данной работы является разработка и программная реализация метода предварительной обработки данных и последующего прогнозирования ССПИ с помощью статистических методов, позволяющих моделировать многомерные временные ряды.

Для достижения поставленной цели необходимо выполнить ряд задач:

1. Исследовать данные и провести их предварительную обработку;
2. Выбрать статистическую модель, подготовить исходные данные под требования выбранной модели и подобрать оптимальные параметры;
3. Выполнить программную реализацию модели;

4. Построить прогноз временных рядов и провести анализ результатов;

5. Сформировать вывод о полученных результатах.

Программная реализация была выполнена на языке программирования Python с использованием дополнительных библиотек, к которым относятся NumPY, Pandas и Keras.

1. Метод исследования

Временным рядом называют последовательность упорядоченных во времени числовых наблюдений, характеризующих уровень состояния и изменения изучаемого явления. Отдельные наблюдения называются уровнями ряда, которые обозначаются как $y_t (t = 1, 2, \dots, n)$, где n – число уровней [2].

Но часто в социально-экономической науке, финансах, медицине и других областях приходится сталкиваться с данными, содержащими более одной переменной, зависящей от времени. Качество прогнозирования ССРП может быть существенно увеличено с использованием многомерного подхода, при котором учитываются факторы, воздействующие на прогнозируемую переменную.

Для поставленной задачи был использован набор данных, содержащий 11 макроэкономических показателей США за период с 1994 до 2021 год. Данные взяты с открытого электронного ресурса, а именно с платформы Kaggle (URL: <https://www.kaggle.com/datasets/calven22/usa-key-macroeconomic-indicators/data>). Среди показателей уровень безработицы (Unrate), денежная масса (M2), реальный общий эффективный обменный курс (Reer) и другие. Эти показатели представляют собой отдельные временные ряды и будут использоваться как независимые переменные для прогноза зависимой переменной ССРП.

Модель векторной авторегрессии (VAR) является расширением модели авторегрессии, допускающая многомерность временных рядов. В 1980 г. Симс предложил представить каждую из компонент многомерного процесса как линейную комбинацию от предыдущих значений всех переменных.

Фактически VAR – это система эконометрических уравнений, каждая из которых представляет собой модель авторегрессии и распределенного лага (ADL) [3].

ADL модель для i -го временного ряда имеет вид:

$$y_t^i = a_0^i + \sum_{j=1}^k a_{1j}^i y_{t-1}^j + \sum_{j=1}^k a_{2j}^i y_{t-2}^j + \dots + \sum_{j=1}^k a_{pj}^i y_{t-p}^j + \varepsilon_t^i, \quad (1)$$

где a_{pj}^i – коэффициенты авторегрессии, a_o^i – константа, ε_t^i – белый шум.

Если ввести вектор временных рядов $Y_t = (y_t^1, y_t^2, \dots, y_t^k)$, то модель $VAR(p)$ для k переменных и числа лагов p в векторно-матричной записи имеет вид:

$$Y_t = a_0 + A_1 Y_{t-1} + A_2 Y_{t-2} + \dots + A_p Y_{t-p} + \varepsilon_t = a_0 + \sum_{m=1}^p A_m Y_{t-m} + \varepsilon_t, \quad (2)$$

где a_0 – вектор констант, A_m – матрицы коэффициентов, ε_t – вектор ошибок.

Также допустимо дополнение модели экзогенными переменными и их лагами, например, до порядка q . Она имеет вид:

$$Y_t = a_0 + \sum_{m=1}^p A_m Y_{t-m} + \sum_{n=1}^q B_n X_{t-n} + \varepsilon_t \quad (3)$$

Векторные модели авторегрессии строятся по стационарным временным рядам.

Для проверки стационарности временного ряда существуют различные методы [3]:

- Графические, заключающиеся в анализе графика временного ряда и графиков их автокорреляционных функций (ACF);
- Формальные, подразумевающие использование статистических тестов, например, тестов Дики-Фуллера или KPSS-теста.

Для достоверности результатов при проверке гипотезы стационарности временного ряда принято использовать несколько тестов и подкреплять их графическими процедурами.

Для приведения временного ряда к стационарному можно использовать дифференцирование. Это преобразование позволяет перейти от ряда y_t к ряду разностей $y_t - y_{t-1}$.

2. Алгоритм построения модели VAR

Этапы построения модели VAR схожи с подходом, представленным Боксом-Дженкинсом, который часто используется для построения моделей авторегрессии интегрированного скользящего среднего. Алгоритм включает следующие этапы [4]:

1. Проверка стационарности временного ряда и преобразование ряда. Для данных временных рядов было достаточно применение двух последовательных разностей, чтобы добиться стационарности;

2. Выбор модели для экспериментальной проверки. Необходимо подобрать параметр модели – порядок авторегрессии p . Для этого

можно воспользоваться анализом графика частной автокорреляции (PACF);

3. Оценка параметров модели (коэффициенты a_{pj}^i) с помощью метода наименьших квадратов (МНК);

4. Проверка адекватности модели на основе анализа остатков модели. Для проверки соответствия остатков модели нормальному распределению используется статистика Жарка-Бера (JB), вычисляемая по формуле [5]:

$$JB = \frac{n}{6} \left(S^2 + \frac{(K-3)^2}{4} \right), \quad (4)$$

где n – количество наблюдений, S – ассиметрия, K – эксцесс.

Для проверки некоррелированности остатков используется статистика Льюнга-Бокса (LB), которая вычисляется по формуле:

$$Q = n(n+2) \sum_{t=1}^k \frac{r_t^2}{n-t}, \quad (5)$$

где r_t^2 – автокорреляция порядка t , k – количество проверяемых лагов.

5. Построение прогноза осуществляется путем замены в общем уравнении модели VAR (формула 2) t на $t+1$, будущих ошибок на нули, а прошлых – на остатки.

3. Результаты эксперимента

Прежде чем строить модель необходимо подготовить данные. АСФ для каждого показателя из набора данных убывает медленно, поэтому есть основания предполагать нестационарность (рис. 1).

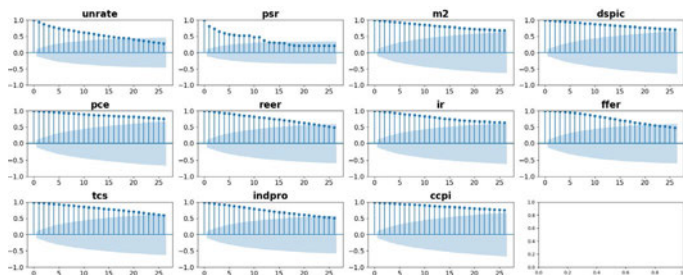


Рис. 1. Графики автокорреляционных функций временных рядов

Для того, чтобы привести данные к стационарной форме, было достаточно провести операцию дифференцирования два раза.

После проведения преобразования коррелограммы каждого показателя из набора данных демонстрируют быстрый спад, что свойственно стационарным рядам (рис. 2).

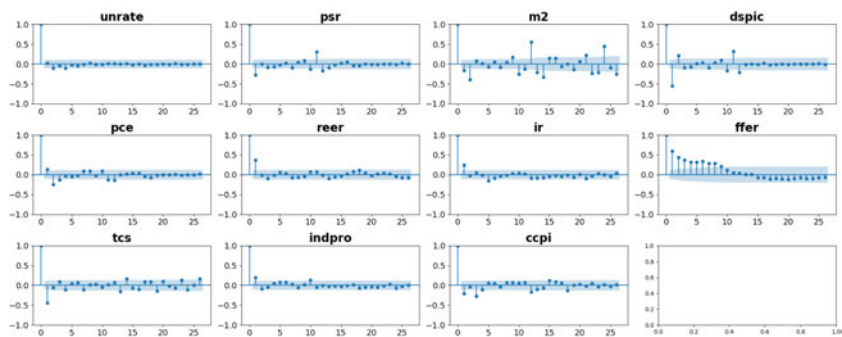


Рис. 2. Графики автокорреляционных функций дифференцированных временных рядов

При использовании векторной авторегрессии необходимо выбрать взаимосвязанные переменные для включения в модель. Статистический тест Гренджера позволяет определить полезен ли тот или иной экономический показатель из набора данных для прогнозирования ССРІ.

В результате теста были получены расчетные значения F-статистики, превышающие уровень значимости $\alpha = 0.05$ для признаков реального эффективного обменного курса, рыночной доходности казначейских ценных бумаг, эффективной ставки по федеральным фондам. Следовательно, считается что эти признаки не являются полезными для прогнозирования ССРІ.

Анализ PACF показал несколько значений порядка авторегрессии, при которых коэффициенты PACF значимы, поэтому было принято решение протестировать несколько моделей VAR с целью выбора лучшей.

В приведенной ниже таблице показаны результаты экспериментов при разных порядках авторегрессии модели VAR. В качестве анализируемого показателя используется средняя квадратичная ошибка (СКО), р-значения статистик JB и LB.

Результаты экспериментов

Экспериментальная модель	СКО	р-значение статистики JB	р-значение статистики LB
VAR(1)	0.609	0.01	0.02
VAR(2)	0.595	0.01	0.12
VAR(3)	0.598	0.02	0.18
VAR(4)	0.578	0.15	0.28

Таким образом, модель VAR с порядком авторегрессии $p = 4$ показала лучшие результаты, а критерии LB и JB подтвердили некоррелированность и нормальность остатков модели.

Также хорошей практикой является изучение графиков выбранной модели (рис. 3):

- Кривой остатков, которая для выбранной модели выглядит как белый шум;
- Нормального квантиля (Q-Q plot), подтверждающий нормальное распределение остатков, так как точки следуют диагональной линии;
- Коррелограммы, показывающий отсутствие коррелированности остатков;
- Гистограммы остатков, напоминающий форму колокола.

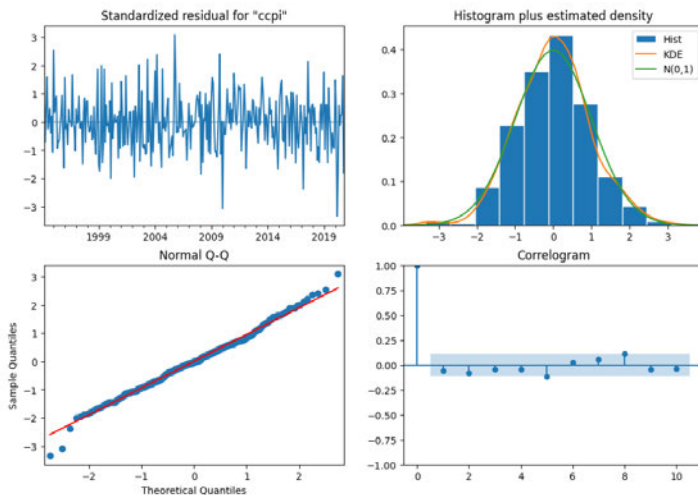


Рис. 3. Графики остатков выбранной модели VAR(4)

Заключение

В рамках данной работы была изучена и выбрана статистическая модель VAR, экспериментально подобран оптимальный параметр порядка авторегрессии. Также были проведены тесты и проанализированы графики остатков, на основании которых была отобрана корректная модель, соответствующая поставленной задаче.

По результатам выполненной работы можно выделить преимущества модели VAR:

- Возможность оценки параметров методом МНК;
- Простота и эффективность использования.

К недостаткам VAR-моделей можно отнести:

- Непростую процедуру определения порядка модели;
- Большое количество коэффициентов, быстро увеличивающихся с ростом числа используемых в модели переменных;
- Применение модели требует тщательной предобработки данных, а также анализа на соответствие данных модели авторегрессии.

Список используемой литературы

1. Ромер, Д. Высшая макроэкономика : учебник / Дэвид Ромер ; пер. с англ. под науч. ред. В. М. Полтеровича ; Нац. исслед. ун-т «Высшая школа экономики». – М. : Изд. Дом Высшей школы экономики, 2014. – 855 с.
2. Новиков, А. И. Экономико-математические методы и модели : учебник / А. И. Новиков. – 6-е изд. – М. : Дашков и К°, 2024. – 532 с.
3. Громов, Е. И. Статистические методы прогнозирования : учебное пособие / Е. И. Громов, О. П. Григорьева, Ю. С. Скрипниченко; Ставропольский государственный аграрный университет. – Ставрополь : АГРУС, 2020. – 168 с.
4. Носко, В. П. Введение в регрессионный анализ временных рядов: учебное пособие / В.П. Носко – 2-е изд. – М.: Издательский Дом МИСиС, 2002. – 274 с.
5. Керимов, А. К. Анализ и прогноз финансовых временных рядов: учебное пособие / А.К. Керимов. – 2-е изд. – М.: Прометей, 2018. – 178 с.

Разработка инструментария для написания тестов в стиле BDD для языка Go

А. С. Колесов

Студент магистр

Д. И. Соломатин

Старший преподаватель

Введение

Behavior-Driven Development (BDD) – это методология разработки программного обеспечения, которая сосредоточена на поведении системы. BDD помогает командам разработчиков, тестировщиков и заказчиков лучше понимать требования к системе и обеспечивает более эффективное взаимодействие между ними [1]. Основной принцип BDD заключается в том, что поведение программного обеспечения должно быть описано на естественном языке, понятном как заказчикам, так и разработчикам.

Одним из популярных инструментов для реализации BDD на Golang является библиотека Godog (она же Cucumber). Godog позволяет писать автотесты на естественном языке с использованием специального синтаксиса под названием Gherkin (листинг 1). Этот синтаксис делает тесты более доступными для всех участников процесса разработки, так как они могут быть написаны на понятном всем языке [2]. Кроме того, написание новых или изменение существующих сценариев может быть выполнено людьми без знания языков программирования, например, ручными тестировщиками или аналитиками.

Листинг 1

Пример спецификации на языке Gherkin

```
Feature: eat godogs
  In order to be happy
  As a hungry gopher
  I need to be able to eat godogs

  Scenario: Eat 5 out of 12
    Given there are 12 godogs
    When I eat 5
    Then there should be 7 remaining
```


Для того, чтобы описанный сценарий можно было использовать в качестве автотеста, необходимо добавить реализацию каждого из используемых шагов, а также написать код, который запустит движок Godog. Пример кода представлен в листинге 2.

Листинг 2

Код, запускающий тесты Godog

```
func thereAreGodogs(ctx context.Context, available int)
(context.Context, error) {
    // Реализация шага
}

// Тестовый метод
func TestFeatures(t *testing.T) {
    suite := godog.TestSuite{
        ScenarioInitializer: InitializeScenario,
        // другие параметры
    }

    if suite.Run() != 0 { \*тест завершился с ошибкой *\ }
}

// Регистрируем шаги
func InitializeScenario(sc *godog.ScenarioContext) {
    sc.Step(`^there are (\d+) godogs$`, thereAreGodogs)
}
```

При использовании Godog, можно заметить несколько его недостатков:

- Определение шага находится отдельно от реализации. Это усложняет чтение и рефакторинг кода.
- Язык определений шагов избыточен и сложен для чтения. Объявление переменных выглядит сложно, и кроме того, для каждого шага мы должны ставим символ начала и завершения строки.
- Нет подсветки синтаксиса/ошибок для определений шагов. Если программист допустит синтаксическую ошибку в определении шага, то она никак не отобразится на интерфейсе, и он узнает о ней только после запуска тестов.

В ходе этой работы была реализована библиотека для тестирования Swede, которая решает перечисленные выше проблемы.

1. Обзор библиотеки Swede

Библиотека Swede – аналог Godog, является средством для написания тестов в стиле BDD на языке Go. Swede имеет отличный от Gherkin язык описания спецификаций (листинг 3). Спецификация

состоит из названия функциональности и набора сценариев, которые описывают корректное поведение. Каждый сценарий состоит из набора шагов (действий).

Листинг 3

Пример спецификации на языке Swede

```
Feature: Math operations

# Just comment

@positive
Scenario: Addition of two positive numbers
- Add 2 and 2
- The result should be 4
```

Для того, чтобы написать автотесты поверх этой спецификации, необходимо

1. Создать тестовый файл Go; добавить в него комментарий go:generate с вызовом кодогенератора Swede. В нем нужно указать файл спецификаций, который будет использоваться в этом тесте;
2. Для каждого шага создать функцию, которая его реализует, и зарегистрировать ее посредством добавления специального комментария;
3. Запустить кодогенератор, используя команду go generate ./...;
4. После этого в тестовом файле появится метод TestSwedeRunner(t *testing.T), который и будет запускать тесты с использованием заданной спецификации.

Пример кода представлен в листинге 4.

При запуске, движок загружает файлы спецификаций, проходит по всем сценариям и последовательно выполняет каждый шаг. Если шаг вернул ошибку или он не был зарегистрирован в движке, то выполнение прекращается и движок завершает работу с ошибкой.

При необходимости, в библиотеки есть возможность добавить код, который будет выполняться до или после запуска каждого сценария/файла спецификации.

В результате работы в консоль выведется информация по статусу выполнения каждого шага.

Листинг 4

Пример кода, запускающего Swede тесты

```
//go:generate swede gen -f ./feature/calculator.swede

// swede:step Add <first:int> and <second:int>
```

```

func addIntAndInt(ctx *runner.Context) error {
    // Реализация шага.
    // Для получения/сохранения данных между шагами
    // используется runner.Context
}

// Другие шаги

// Этот код генерируется при запуске go generate
func TestSwedeRunner(t *testing.T) {
    _runner := runner.NewRunner()
    _runner.LoadFeatureFile("./feature/calculator.swede")
    _runner.RegisterFunc("Add <first:int> and <second:int>",
        addIntAndInt)
    // Регистрация остальных шагов
    _runner.Run()
}

```

2. Реализация механизма аннотаций и кодогенератора

В то время как Java и некоторые другие языки программирования позволяют добавлять и извлекать метаданные функций с помощью аннотаций, в Go такой возможности нет. Чтобы решить проблему размещения определения шага непосредственно рядом с его реализацией, был разработан метод, использующий специальные комментарии. Каждый шаг сопровождается комментарием с отметкой “swede:step”, после которой указывается описание шага.

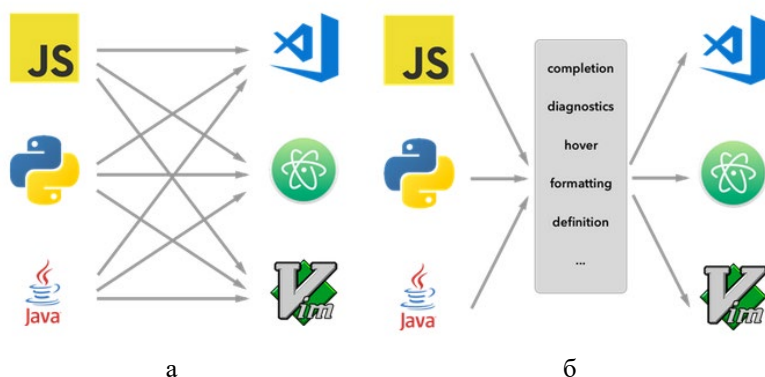
Для того, чтобы создать функцию, запускающую тесты Swede, был реализован кодогенератор. Кодогенератор парсит заданный файл, находит аннотированные функции, по ним и переданным извне параметрам генерирует функцию TestSwedeRunner.

Кодогенератор написан также на Golang, и для получения AST представления исходного файла использует библиотеку dave/dst [3]. Стандартная реализация парсера go/parser не подошла, так как в ней есть проблемы при работе с комментариями.

3. Реализация поддержки Swede для сред разработки

Для добавления поддержки нового языка в среду разработки существуют различные подходы. Один из них включает создание специального плагина, который обеспечит функции, такие как форматирование кода, подсветка синтаксиса и другие необходимые возможности. Однако более предпочтительным и эффективным способом является создание языкового сервера (language server) и его последующий интеграции с выбранной средой разработки. Разница между двумя этими подходами изображена на рис. 1.

Языковой сервер представляет собой отдельную программу, способную анализировать исходный код на конкретном языке программирования и предоставлять редактору кода информацию о структуре кода, типах данных, возможных ошибках и других аспектах. Эта технология позволяет не только улучшить работу с кодом, но и обеспечить высокую степень совместимости между различными редакторами кода и языками программирования. Так, для поддержки нового языка можно написать только один языковой сервер вместо множества плагинов для редакторов кода.



а – без языковых серверов, б – с языковыми серверами

Рис. 1. Количество необходимых связей для поддержки нового языка

Для взаимодействия между средой разработки и языковым сервером используется Language Server Protocol – открытый протокол, разработанный в компании Microsoft. Он реализован поверх протокола JSON-RPC [4].

Стадии работы Language Server Protocol:

1. При открытии файла/проекта клиент (среда разработки) запускает соответствующий Language Server. Обычно для коммуникации между Language Server и клиентом используются стандартные каналы ввода-вывода, хотя также возможно использование протокола WebSocket.
2. Клиент посылает запрос, в котором содержится информация о среде разработки, текущем проекте и наборе доступных возможностей клиента.

3. Language Server присылает ответ с информацией о себе и набором своих доступных возможностей.
4. После этого сервер начинает принимать запросы от клиента, такие как операции над кодом (автозавершение, проверка синтаксиса), информация о коде или его элементах, выполнение команд и прочее. При этом клиент уведомляет сервер о таких событиях, как открытие файла, его изменение, перемещение. В некоторых ситуациях инициатором отправки сообщения может быть сервер, например, для передачи найденных в проекте ошибок.
5. По завершении работы с файлом или проектом сервер и клиент завершают соединение, освобождая ресурсы и закрывая рабочую область.

Языковой сервер Swede был реализован на языке Golang, с использованием библиотеки `tliron/glspl` [5] – она позволяет удобным образом создавать языковые сервера. Языковой сервер Swede поддерживает следующие возможности:

- форматирование текста;
- подсветка синтаксиса;
- подсветка ошибок.

Для реализации всех этих возможностей был написан восходящий парсер, допускающий синтаксические ошибки во входном файле. Архитектура парсера изображена на рис. 2.

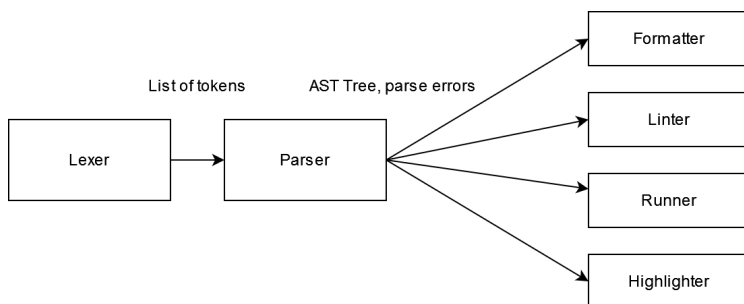


Рис. 2. Архитектура парсера

Для подсветки определений шага и ошибок в нем была взята и расширена реализация языкового сервера для Golang – `Gopls`.

Для подсветки синтаксиса определений шага была добавлена логика в код, обрабатывающий комментарии.

Для подсветки ошибок в определении шага был создан анализатор го файлов. Для реализации анализатора необходимо создать экземпляр

структуры из пакета `analyzer.Analyzer` (листинг 5) и подключить его в файле настроек.

Листинг 5

Структура анализатора кода Go

```
type Analyzer struct {
    // Имя анализатора кода
    Name string
    // Описание анализатора кода
    Doc string
    // Функция для проведения анализа кода.
    // Через объект Pass можно получить результаты
    // работы других зависимых анализаторов
    Run func(*Pass) (interface{}, error)
    // Тип возвращаемого значения.
    ResultType reflect.Type
    // Если параметр равен true, то анализатор
    // будет работать, даже если код содержит ошибки.
    RunDespiteErrors bool
    // Зависимости анализатора.
    // Например, используя возвращаемый результат
    // inspect.Analyzer, мы можем получить доступ к AST
    // дереву Go файла
    Requires []*Analyzer

    // Другие параметры
}
```

Пример работы измененного Gopls сервера показан на рис. 3.

A screenshot of a code editor window showing Go code with syntax highlighting. The code is a function `addIntAndInt` that takes a context and two integers, adds them, and sets the result in the context. The code is as follows:

```
// swede:step Add <first:int> and <second:int>
tabnine: test | explain | document | ask
func addIntAndInt(ctx *runner.Context) error {
    result := ctx.GetIntVariable("first") + ctx.GetIntVariable("second")
    ctx.SetVariable("result", result)

    return nil
}
```

Рис. 3. Подсветка синтаксиса в go файлов в VS Code

Заключение

В результате этой работы разработан инструментарий для написания BDD тестов на языке Go в виде тестовой библиотеки, языкового сервера для файлов спецификаций Swede, и измененного Gopls сервера.

Список литературы

1. Введение в BDD. [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/companies/auriga/articles/808527/>
2. Godog. [Электронный ресурс]. – Режим доступа: <https://github.com/cucumber/godog>
3. Библиотека dst [Электронный ресурс]. – Режим доступа: <https://github.com/dave/dst>
4. LSP specification. [Электронный ресурс]. – Режим доступа: <https://microsoft.github.io/language-server-protocol/specifications/lsp/3.17/specification/>
5. Библиотека glsp [Электронный ресурс]. – Режим доступа: <https://github.com/tliron/glsp>

Выявление признаков жанра и классификация музыкальных фрагментов с помощью инструментов ИИ

Е. Э. Косинов

Студент магистр

В. А. Каладзе

Профессор

Введение

Распознавание звуков часто применяется при решении прикладных задач. Интересным вариантом является его применение в музыке. Иногда неизвестно, что за музыкальная композиция звучит в данный момент, то требуется узнать её название или выяснить в каком жанре играется композиция. Эта работа является продолжением [1].

1. Постановка задачи и выявление музыкальных признаков

Таким образом формируется задача данной работы: классификация музыкальных аудиофайлов по жанрам. Основными методами решения данной задачи являются искусственные нейронные сети.

Для классификации музыкальных композиций по музыкальным жанрам, нужно иметь представление, по каким признакам происходит классификация. В работе [7] приводятся основные признаки, используемые в специальных приложениях для определения музыкальных жанров:

- Временные признаки (temporal features) – вычисляются из кадров звукового сигнала (частота переходов через ноль и т. д.).

- Энергетические признаки (energy features) – признаки, относящиеся к энергии сигнала (среднеквадратичная энергия кадра (фрагмента сигнала), энергия гармонической составляющей спектра, энергия шумовой части спектра и т. д.).

- Признаки формы спектра (spectral shape features) – это признаки, которые описывают форму спектра звукового кадра (центрорид, коэффициенты асимметрии и эксцесса, мел-частотные кедральные коэффициенты (MFCC) и т.д.).

- Перцептивные признаки (perceptual features) – признаки, относящиеся к восприятию, вычисляются с использованием модели

человеческого восприятия звука (относительная громкость, чёткость и т. д.).

Описанные выше признаки, извлекаемые из одного аудиосегмента каждого произведения – это отрезок длиной в 30 секунд, взятый через 30 секунд от начала композиции (чтобы не рассматривать вступления, которые могут отличаться от общего тона произведения) [7].

Алгоритм решения задачи распознавания музыкальных композиций по жанрам имеет вид:

- Определение набора классификационных признаков произведений

- Выделение признаков из файлов аудиозаписей при помощи специализированной библиотеки librosa

- Классификация музыкальных фрагментов при помощи нейронной сети

Музыкальная композиция определяется как звуковой объект, характеризуемый некоторыми признаками, такими как амплитуда колебаний, частотный диапазон, темп, громкость и т.д. [7]. С помощью частотного анализа выделяются числовые значения некоторых признаков звукового сигнала, и после нормализации, в формате вещественных чисел, они в виде массива данных, передаются на вход нейронной сети.

Имеется 8 основных числовых классификационных признаков (характеристик сигнала), описывающих каждую аудиозапись:

- Zero Crossing Rate (Частота переходов аудиосигнала через ноль), – количество перемен знака аудиосигнала на промежутке, то есть количество раз, когда сигнал меняет своё значение с положительного на отрицательное или наоборот [2].

- MFCC (Мел-кепстральные коэффициенты) – это матрица значений, которая захватывает тембральные аспекты. Мел – это единица высоты звука, выраженная в Мелах, вместо Герц. Кепстр – это акустическая волна (параметры голосового тракта, тоновые сигналы, шумы и фильтры) [2].

- Spectral centroid (Спектральный центроид) – это мера, указывающая расположение центра масс спектра. Она отвечает за впечатление яркости звука [8]

- Spectral rolloff (Спектральный спад) – частота, ниже которой находится определённый процент значений спектра.

- Spectral Bandwidth (Ширина спектра) – интервал частот или интервал длин волн, характеризующий спектральные линии [9].

– Volume – это инструмент измерения, который измеряет среднюю громкость аудиодорожки в пределах окна сигнала примерно в 300 миллисекунд.

– Beats-per-minute (ритм) – Показатель, определяющий скорость исполнения или воспроизведения композиции

– Harmonics – временной ряд гармонической части сигнала, которая получается путём разделения основного временного ряда на гармоническую (тональную) и перкуссионную (ударную) части [8].

2. Проектирование нейросети

Реализацией метода классификации музыкальных аудиозаписей по музыкальным жанрам была выбрана искусственная нейронная сеть прямого распространения, т.к. эмпирически было определено, что данный тип сети эффективно решает задачи классификации [5]. На входной слой нейронной сети подаётся восьмимерный вектор признаков музыкальных аудиозаписей, рассмотренных выше (массив нормализованных значений). На выходе сети находится десятимерный вектор по количеству распознаваемых классов (музыкальных жанров): блюз, классика, кантри, диско, хип-хоп, джаз, метал, поп, регги, рок. Активационной функцией на выходном слое была выбрана Softmax [10], чтобы каждое значение выходного вектора представляло из себя вероятность принадлежности объекта к классу. Эмпирическим путём было установлено [5], что три скрытых слоя у нейронной сети будет оптимальным и достаточным количеством для решения данной задачи классификации. Активационной функцией или ReLu (3) (рис. 1), т.к., в отличие от других активационных функций (гиперболический тангенс или сигмоида), выпрямитель требует намного меньше ресурсоёмких вычислений [6]. В качестве оптимизационного метода был выбран Adam из-за его вычислительной эффективности и небольшого требования к памяти [4]. Структура реализованной нейронной сети изображена на рисунке 2. Softmax – это логистическая функция для многомерного случая (1). Это означает, что функция применяется не к отдельному значению, а к вектору. Она используется в том случае, когда стоит задача многоклассовой классификации [10].

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{k=1}^K e^{z_k}} \quad (1)$$

ReLu – это наиболее часто используемая функция активации при глубоком обучении. Она возвращает значение x , если x положительно,

и 0 – в противном случае (2). Графически – это биссектриса первого квадранта.

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \quad (2)$$

Её основным преимуществом является небольшое требование к вычислительным ресурсам, что ускоряет физическое время обучения нейронной сети [6].

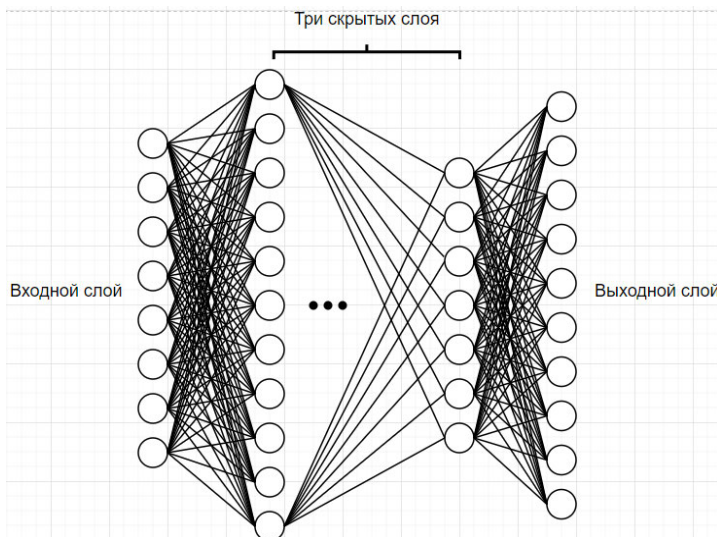


Рис. 1. Структура нейронной сети

Adam (Adaptive Momentum) – это один из самых эффективных методов оптимизации [3] в обучении нейронных сетей, объединяющий в себе идеи таких процедур как Adagard и RMSProp. Математический алгоритм Adam реализуется выражениями (3) – (6):

$$w^{n+1} = w^n - \frac{\alpha}{\sqrt{\bar{v}_n} + \epsilon} \bar{m}_n \quad (3)$$

$$\bar{m}_n = \frac{m_n}{1 - \beta_1^n}; \bar{v}_n = \frac{v_n}{1 - \beta_2^n} \quad (4)$$

$$m_n = \beta_1 m_{n-1} + (1 - \beta_1) g_n \quad (5)$$

$$v_n = \beta_2 v_{n-1} + (1 - \beta_2) g_n^2 \quad (6)$$

В этом алгоритме, в отличие от остальных, подбирают лишь один параметр α – learning rate (скорость обучения). Остальные параметры: $\beta_1=0.9$ (показатель экспоненциальной скорости затухания для первого момента), $\beta_2=0.09$ (показатель экспоненциального спада для оценок второго момента) и $\varepsilon=10^{-8}$ (небольшое число, предотвращающее любое деление на ноль) – считаются стандартными [4].

Подбор параметра α является главным в настройке Adam [4]. Однако в Keras параметр α по умолчанию задан 0.001 [11]. Эмпирически было определено, что это значение является оптимальным для решения задач классификации.

3. Программная реализация нейросети

Для реализации алгоритма классификации были использованы библиотеки с открытым исходным кодом на языке программирования python 3.7.7 в фреймворке PyCharm 2020.1: librosa 0.9.2, tensorflow 2.10.0 и keras 2.10.0, предназначенные для извлечения музыкальных характеристик из аудиозаписей (librosa) и проектирования нейросетей (tensorflow, keras).

Для обучения нейросети использовался общедоступный набор (датасет) музыкальных композиций GTZAN, состоящий из 1000 примеров (10 музыкальных жанров, по 100 композиций на каждый). Используя librosa, все музыкальные композиции были преобразованы в спектрограммы, из которых были извлечены все необходимые характеристики в числовом формате [8]. Полученные характеристики были нормализованы, занесены в датасет и сгруппированы. Датасет был разбит на обучающую, валидационную и контрольную выборки, составляющих 60%, 20% и 20% соответственно от общего числа композиций.

Далее в PyCharm с помощью TensorFlow и Keras была спроектирована искусственная нейронная сеть с тремя скрытыми слоями [11] (рис. 2).

На обучающей выборке модель нейронной сети была обучена. 600 эпох для обучения данной модели является оптимальным вариантом, т.к. при меньшем количестве эпох модель теряет в точности, а при большем количестве она переобучается, и точность модели не улучшается (а при большем количестве эпох точность ухудшается).

После завершения обучения нейросети, она была проверена на валидационной выборке. В результате выводится информация о точности распознавания музыкальных аудиофайлов по жанрам (рис. 3).

Точность составляет ~93%, это говорит о том, что нейросеть обучена и хорошо справляется со своей задачей.

```
model=tf.keras.models.Sequential([
    tf.keras.layers.Dense(512,activation='relu',input_shape=(X_train.shape[1],)),
    tf.keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(256,activation='relu'),
    keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(128,activation='relu'),
    tf.keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(64,activation='relu'),
    tf.keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(10,activation='softmax'),
])

print(model.summary())
model_history=trainModel(model=model,epochs=600,optimizer='adam')
```

Рис. 2. Программный код искусственной нейронной сети

```
26/26 [=====] - 0s 2ms/step - loss: 0.4806
- accuracy: 0.9284
The test loss is 0.48055750131607056
The best accuracy is: 92.84197688102722
```

Рис. 3. Результаты работы нейросети

В результате проверки работы нейросети на контрольной выборке точность распознавания составляет также ~93%.

Заключение

Средняя точность классификации музыкальных композиций по жанрам, равная ~93% является достойным результатом. Таким образом можно сказать, что искусственная нейронная сеть прямого распространения эффективно решает задачу классификации музыкальных композиций по нескольким характеристикам.

Список литературы

1. Каладзе, В. А. Нейросетевое распознавание и классификация музыкальных фрагментов / В. А. Каладзе, Е. Э. Косинов // Актуальные проблемы прикладной информатики : сб. тр. участников IV Международной научно-практической конф. "Проблемы теории и

практики инновационного развития и интеграции современной науки и образования" (Москва, 15 февраля 2023 г.). – Москва, 2023 – С. 144-151.

2. Анализ аудиоданных [Электронный ресурс] : база данных. – Режим доступа: <https://habr.com/ru/post/670676/>

3. Каширина, И. Л. Исследование и сравнительный анализ методов оптимизации, используемых при обучении нейронных сетей / И. Л. Каширина, М. В. Демченко // Вестник ВГУ. – Воронеж, 2018. – № 4. – С. 124-132.

4. Оптимизация в ML [Электронный ресурс] : многопредмет. науч. журн. – Режим доступа: <https://academy.yandex.ru/handbook/ml/article/optimizaciya-v-ml>

5. Типы нейронных сетей. Принцип их работы и сфера применения. [Электронный ресурс] : многопредмет. науч. журн. – Режим доступа: <https://otus.ru/nest/post/1263/>

6. Функции активации в искусственных нейронных сетях [Электронный ресурс] : многопредмет. науч. журн. – Режим доступа: <https://aisimple.ru/8-funkcii-aktivacii-v-iskusstvennyh-nejronnyh-setjah.html>

7. Scaringella, N. Automatic genre classification of music content: a survey / N. Scaringella, G. Zoia, D. Mlynek // Signal Processing Magazine. – 2006. – Vol. 1. – Issue 2. – P. 131-144.

8. Librosa [Электронный ресурс] : база данных. – Режим доступа: <https://librosa.org/doc/latest/index.html>

9. Peeters, G. A large set of audio features for sound description (similarity and classification) in the CUIDADO project / G. Peeters // CUIDADO I.S.T. Project Report. – 2004. – Vol. 1. – Issue 1. – P. 25.

10. Softmax [Электронный ресурс] : база данных. – Режим доступа: <https://congyuzhou.medium.com/softmax-3408fb42d55a>

11. TensorFlow [Электронный ресурс] : база данных. – Режим доступа: https://www.tensorflow.org/api_docs/python

Оптимизация пропускной способности конвейера обработки изображений при помощи генетического алгоритма

Я.В. Лазарев

Студент магистр

Д.И. Соломатин

Старший преподаватель

Введение

В современном мире обработка изображений является важной задачей во многих сферах. Однако, с увеличением объема данных и требований к скорости обработки, эффективное управление пропускной способностью конвейеров обработки изображений становится важным. В этой работе мы обращаемся к использованию генетических алгоритмов для оптимизации пропускной способности конвейеров обработки изображений.

Генетические алгоритмы – это эвристический метод оптимизации, вдохновленный процессами естественного отбора и генетической эволюции. Их применение в области обработки изображений предлагает новый подход к решению проблемы оптимизации производительности конвейеров. Цель работы – исследовать, как генетические алгоритмы могут быть адаптированы и применены для оптимизации пропускной способности конвейера обработки изображений, с целью повышения эффективности и ускорения процесса обработки.

Эта статья предлагает методику применения генетических алгоритмов для улучшения их производительности.

1. Конвейер обработки изображений

Для последовательной обработки изображений используется архитектура канал-фильтр. Она позволяет разделить обработку изображений на независимые этапы и хорошо подходит для оптимизации производительности за счет возможности параллельной обработки. Распараллеливание достигается за счет назначение отдельного потока выполнения инструкций для каждого из фильтров. Каналы служат для передачи данных между различными компонентами системы. Они обеспечивают связь и взаимодействие между

компонентами, позволяя им обмениваться информацией [1]. Простая модель конвейера обработки изображений проиллюстрирована на рис.1:

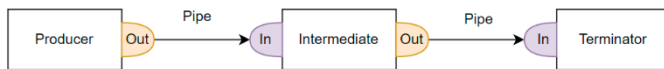


Рис. 1. Абстрактная модель конвейера обработки изображений

Схема отображает основные типы фильтров обработки изображений:

- Producer – это фильтр, порождающий данные. Источником изображений может служить как каталог на диске, так и видеопоток (например, с камеры). Producer не имеет входного канала In.
- Intermediate – фильтр промежуточной обработки изображений. Имеет входной канал In, и выходной канал Out.
- Terminator – завершающий фильтр. Осуществляет последнюю операцию над изображением, например сохранение изображения на устройство долговременного хранения.

Каждый из этапов обработки может иметь произвольное количество потоков, производящих трансформацию изображения. Концептуальная схема конвейера представлена на рис. 2:

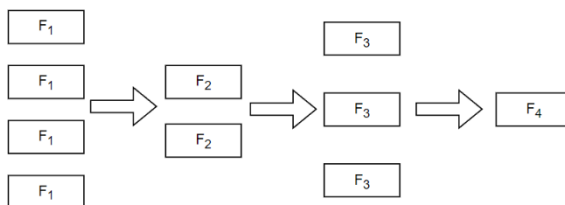


Рис. 2. Схема конвейера с произвольной конфигурацией

Передача данных между этапами обработки (индексы на схеме обозначают номер этапа обработки) осуществляется посредством механизма «Round Robin», то есть записью по круговому движению. Данный алгоритм не участвует в задаче оптимизации.

2. Числовые характеристики конвейера

Каждый фильтр обрабатывает кадр за время t . Поскольку время обработки варьируется в силу разнообразия данных для обработки и

прерывания процессора, будем считать t как среднее время. Тогда пропускной способностью фильтра F называется величина:

$$b = \frac{1}{t}. \quad (1)$$

Слоем обработки будем называть группу фильтров одного класса, находящуюся в конвейере на одном уровне. Если считать среднюю пропускную способность фильтров одного слоя одинаковой, то пропускной способностью слоя будем называть величину B :

$$B = n * b = \frac{n}{t}. \quad (2)$$

Вектором B пропускной способности конвейера будем называть упорядоченный набор значений B , порядок которых соответствует порядку слоев обработки конвейера. Первый слой можно назвать продуцирующим слоем, а последний – слоем-терминатором. Для конвейера, изображенного на рис. 2, вектор будет выглядеть так, как представлено в формуле 3:

$$\overline{B} = \{B_1, B_2, B_3, B_4\}. \quad (3)$$

Далее определим некоторые правила для величин пропускной способности конвейера. Общей пропускной способностью конвейера называется величина B_{pp} , равная минимальной пропускной способности слоя конвейера, то есть минимальное значение вектора B :

$$B_{pp} = \min \overline{B}. \quad (4)$$

На пропускную способность конвейера накладывается ограничение сверху – его мощность не может превышать скорость создания кадров первым (продуцирующим слоем).

Не менее важное свойство конвейера – это поспеваемость обработки. Конвейер считается поспевающим, если каждый слой имеет пропускную способность не меньше, чем предшествующий ему слой. Это гарантирует, что при обработке не будет потеряно никаких кадров, что является важным условием эффективной работы. Данное условие можно выразить следующей формулой:

$$B_i \leq B_{i+1}, i = 1..k - 1. \quad (5)$$

Однако данное условие не является полным. Хотя оно и задает поспевающий конвейер, здесь нет сильных ограничений на пропускную способность. Такое условие подходит и для конвейера, пропускная способность соседних слоев которого различается в несколько раз. Подобное соотношение не является эффективным, поэтому необходимо наложить дополнительное ограничение:

$$e_i = \frac{B_i}{B_{i+1}} \leq 1, u \ 1 - e_i < \varepsilon, i = 1..k-1. \quad (6)$$

Под ε стоит понимать некоторую малую положительную, но не далекую от нуля величину. Конкретное её значение определяется исходя из требований к конвейеру обработки. Чем меньше эта величина, тем более эффективно используются ресурсы для обработки.

3. Оптимизация пропускной способности

Конвейеры с малым количеством этапов обработки могут быть оптимизированы при помощи простого алгоритма перебора. Последовательность действий такова:

1. Выбрать значения количества фильтров для каждого этапа обработки, начиная последовательно с 1 и заканчивая максимально допустимым значением.
2. Произвести расчет пропускной способности фильтра.
3. Записать значение итоговой пропускной способности, полученной по формуле 4.
4. Повторить шаги 1-3 для всех комбинаций количества фильтров.
5. Среди полученных результатов выбрать максимум.

Такой подход можно применить, когда количество комбинаций невелико. Так как количество комбинаций, вообще говоря, является факториалом числа этапов обработки, для больших конвейеров итоговое число комбинаций будет быстро расти. Чтобы сократить число проверяемых конфигураций конвейера в данном случае, используется генетический алгоритм.

Для применения генетического алгоритма необходимо создать модель оптимизируемой системы. Каждому индивидууму соответствует хромосома, представляющая набор генов. Например, хромосому можно представить двоичной строкой, в которой каждый бит соответствует одному гену [2]. Для решаемой задачи хромосомой будет являться упорядоченный набор целых чисел, являющийся количеством фильтров на каждом этапе обработки. Пример такой хромосомы изображен на рис. 3.

4	2	3	1

Рис. 3. Пример хромосомы

Новые хромосомы получаются при помощи оператора скрещивания, суть которого заключается в том, чтобы из родительских наборов хромосом создать новую, содержащую данные обоих

родителей. В данной модели этот оператор представлен простой операцией, где от двух хромосом берется по 50% данных, из которых складывается дочерняя хромосома. Визуализация работы данного оператора представлен на рис.4:

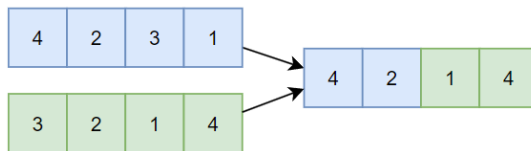


Рис. 4. Оператор кроссовера

Однако одного оператора скрещивания недостаточно для создания набора хромосом (или популяции), который позволит исследовать всю область возможных значений генов, так как простое смешивание родительских хромосом не приводит к появлению абсолютно новых значений для конфигурации конвейера. Для решения этой проблемы в генетических алгоритмах используется оператор мутации.

Мутация – генетический оператор, который с некой вероятностью меняет один или несколько «генов» в случайных позициях «хромосомы». В данном случае мутацией будет случайное изменение количества фильтров для одного из слоев. Изменение количества будет происходить в диапазоне $[1, n_{\max}]$, где $n_{\max}$ – максимальное количество фильтров для одного слоя.[3]

И, наконец, необходимо определить функцию приспособленности для данной модели, которая позволяет оценить, насколько конфигурация конвейера является эффективной:

$$F = \sum \alpha(e_i) - \beta \sum (1 - e_i)^2 + \gamma B_{pp}, \beta, \gamma > 0 \quad (7)$$

Данная функция учитывает все описанные выше характеристики конвейера. Функция $\alpha(e_i)$ является штрафной функцией для оценки успеваемости конвейера обработки. Она задается следующим образом: для значений e_i , не превышающих единицы, её значение есть некая положительная константа A . Для всех остальных возможных значений функция возвращает штраф, пропорциональный входному значению:

$$\alpha(x) = \begin{cases} A, 1 - x \geq 0, A > 0 \\ -Bx^2, 1 - x < 0, B > 0 \end{cases} \quad (8)$$

Чтобы получить значение итоговой пропускной способности конвейера, можно воспользоваться простым расчетом на основе тестового запуска, который даст информацию о пропускной

способности каждого из слоев обработки. Однако, такой подход не гарантирует, что пропускная способность слоев останется той же, что и при первом запуске, так как изменение количества фильтров влечет за собой изменение времени обработки. Поэтому для более точных измерений следует сделать тестовый запуск с полученными параметрами конфигурации фильтров.

Формирование популяции для старта алгоритма можно взять из граничных значений количества фильтров, а также на основе расчета, исходя из пропускной способности фильтров. Допустим, имеется вектор пропускной способности $\{20, 40, 50, 10\}$ для конфигурации $\{1, 1, 1, 1\}$. Тогда можно добавить в стартовую популяцию хромосому $\{2, 1, 1, 5\}$.

Как результат, примененный алгоритм за 2000-3000 итераций находит несколько оптимальных конфигураций фильтров, позволяющих производить обработку изображений с большей пропускной способностью, чем при значениях стартовой популяции.

Заключение

Данная статья посвящена исследованию и разработки модели оптимизации конвейера обработки изображений. В ходе работы описаны модель конвейера и её числовые характеристики, построена модель оптимизации при помощи генетического алгоритма, а также приведены результаты применения такой модели. Результат работы включен как опция библиотеки, предоставляющей инструмент построения конвейера обработки изображений.

Список литературы

1. Каналы и фильтры: эффективный подход к обработке данных в программировании [Электронный ресурс] : справочник. – Режим доступа: <https://nauchniestati.ru/spravka/kanaly-i-filtry>
2. Вирсански, Э. Генетические алгоритмы на Python / Э. Вирсански ; пер. с англ. А. А. Слинкина. – М. : ДМК Пресс, 2020. – 286 с.
3. Искусственный интеллект. Генетический алгоритм и его применения [Электронный ресурс] : электр. журн. – Режим доступа: <https://3dnews.ru/1050009/iskusstvenniy-intellekt-geneticheskiy-algoritm-i-ego-primeneniya>

Автоматизация распределения задач с учетом предпочтений руководителя и членов проектной команды

В. В. Локтев

Студент магистр

В. В. Коротков

Старший преподаватель

Введение

В течение последних десятилетий было разработано множество методологий управления проектами. Особенную популярность в IT и некоторых других сферах получили гибкие методологии Agile, основополагающим принципом которых является то, что люди и их взаимодействие важнее процессов и инструментов. Если в классических подходах исполнители задач, как правило, назначаются менеджментом, то в гибких подходах практикуется самоуправляемость команд и решение проблемы распределения задач путем групповых обсуждений. Однако оставление этого процесса без вмешательства зачастую может приводить к нежелательным последствиям. Именно поэтому существует противоречие, заключающееся в том, что с одной стороны необходимо учитывать интересы самих сотрудников, а с другой – вышестоящего руководителя. Поэтому необходимо разработать такую систему, которая бы позволяла учитывать интересы каждой из сторон.

1. Постановка задачи

Целью работы является разработать многопользовательскую систему автоматического распределения задач с учетом предпочтений руководителя и членов проектной команды на основе метода ветвей и границ и нечеткого TOPSIS.

У нас есть множество задач, которые нужно распределить по некоторому множеству исполнителей. Для каждой задачи руководителем устанавливается оценка, например, в условных единицах сложности, которые принято называть story points. Также у каждого исполнителя тоже есть оценка в тех же единицах измерения, показывающая какой объем задач он может успеть сделать за спринт [1-2]. Далее руководитель проекта отражает свои предпочтения, указав для каждой

работы возможных исполнителей, упорядочив их в порядке возрастания предпочтительности. Это наглядно продемонстрировано на рис. 1.

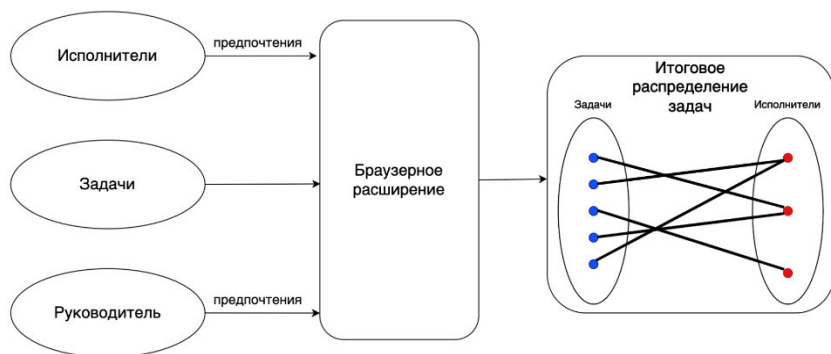


Рис. 1. Схема работы системы

2. Нечеткие оценки

Предпочтения исполнителей носят неопределенный характер и выражаются в виде нечетких чисел. Нечеткое число – выпуклое нормальное нечеткое множество с кусочно-непрерывной функцией принадлежности, заданное на множестве действительных чисел. Нечеткое число – частный случай нечеткого множества.

$$A = \{(x, \mu_A(x)) \mid x \in X\}, \quad (1)$$

где x – действительное число, $\mu_A(x)$ – функция принадлежности числа x .

Для задания этих чисел используем треугольную функцию принадлежности, представленную на рис. 2.

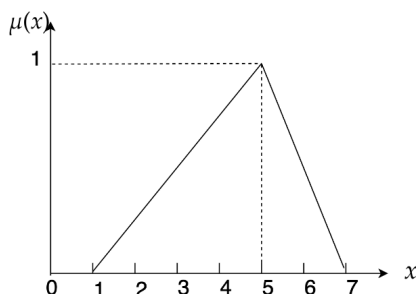


Рис. 2. Треугольная функция принадлежности

Треугольные числа $\langle a, b, c \rangle$ отличаются удобством представления субъективных категорий и простотой вычислений по сравнению с более сложными видами.

Преимущество нечеткого подхода перед вероятностным заключается в повышенной степени обоснованности принимаемых решений [4-5]. Это связано с тем, что в расчёт попадают все возможные сценарии развития событий, что несвойственно вероятностным методам, рассчитанным на конечное множество сценариев.

3. Многокритериальный выбор

Далее оценка предпочтений исполнителей определяется при помощи нечеткого метода TOPSIS [3].

Оценки задач по различным критериям x представляются в виде нечетких треугольных чисел. Также задаются веса важности критериев w . В табл. 1 изображена шкала важности критериев, в табл. 2 – шкала оценки задач, такие параметры, как интерес к предметной области, сложность задачи, посильность задачи и т. д.

Таблица 1

Шкала важности критериев

Важность критериев	Нечеткое треугольное число
Очень незначительная	(1, 1, 3)
Незначительная	(1, 3, 5)
Средняя	(3, 5, 7)
Значительная	(7, 9, 10)
Очень значительная	(9, 10, 10)

Таблица 2

Шкала оценки задач

Оценка задач	Нечеткое треугольное число
Очень низкая	(1, 1, 3)
Низкая	(1, 3, 5)
Средняя	(3, 5, 7)
Высокая	(7, 9, 10)
Очень высокая	(9, 10, 10)

Есть группа из K исполнителей. Каждый исполнитель предоставляет свои оценки задач и весов критериев. Далее вычисляются агрегированные оценки для получения некой общей оценки самих задач и весов.

$$\tilde{x}_{ij} = (a_{ij}, b_{ij}, c_{ij}), \quad (2)$$

где a_{ij}, b_{ij}, c_{ij} – треугольные нечеткие оценки задач от исполнителей.

$$a_{ij} = \min_k \{a_{ij}^k\}, b_{ij} = \frac{1}{K} \sum_{k=1}^K b_{ij}^k, c_{ij} = \max_k \{c_{ij}^k\} \quad (3)$$

$$\tilde{w}_j = (w_{j1}, w_{j2}, w_{j3}), \quad (4)$$

где w_{j1}, w_{j2}, w_{j3} – треугольные нечеткие оценки весов критериев.

$$w_{j1} = \min_k \{w_{j1}^k\}, w_{j2} = \frac{1}{K} \sum_{k=1}^K w_{j2}^k, w_{j3} = \max_k \{w_{j3}^k\} \quad (5)$$

Затем происходит нормализация матрицы оценок.

$$\tilde{r}_{ij} = (\frac{a_{ij}}{c_j^*}, \frac{b_{ij}}{c_j^*}, \frac{c_{ij}}{c_j^*}), \quad c_j^* = \max_i \{c_{ij}\} \quad (6)$$

Умножаем оценки на веса критериев.

$$\tilde{v}_{ij} = \tilde{r}_{ij} \times w_j \quad (7)$$

Находим идеальную и анти-идеальную точки.

$$A^* = (\tilde{v}_1^*, \tilde{v}_2^*, \dots, \tilde{v}_n^*), \tilde{v}_j^* = \max_i \{v_{ij3}\} \quad (8)$$

$$A^- = (\tilde{v}_1^-, \tilde{v}_2^-, \dots, \tilde{v}_n^-), \tilde{v}_j^- = \min_i \{v_{ij1}\} \quad (9)$$

Рассчитываем расстояние до этих точек и находим итоговую метрику на основе этого, с помощью которой и получаем итоговые оценки предпочтительности задач исполнителями [6-7].

$$d(\tilde{x}, \tilde{y}) := \sqrt{\frac{1}{3}[(a_1 - a_2)^2 + (b_1 - b_2)^2 + (c_1 - c_2)^2]} \quad (10)$$

$$d_i^* = \sum_{j=1}^n d(\tilde{v}_{ij}, \tilde{v}_j^*), \quad d_i^- = \sum_{j=1}^n d(\tilde{v}_{ij}, \tilde{v}_j^-), \quad v_{rj}^* = \frac{d_i^-}{d_i^- + d_i^*} \quad (11)$$

4. Комбинаторная оптимизация

Оценка качества соответствия исполнителя и задачи определяется как их среднее геометрическое.

$$v_{rj} = \sqrt{v_{rj}^* v_{rj}^t}, \quad (12)$$

где v_{rj}^t – оценка соответствия задачи и исполнителя по мнению руководителя.

Далее мы ищем оптимальные соответствия как решение оптимизационной задачи с заданным ограничением. Решение ищем методом ветвей и границ.

$$\sum_{r \in R} \sum_{j \in J} x_{rj} v_{rj} \rightarrow \max \quad (13)$$

$$\sum_{j \in J} x_{rj} p_j \leq h_r, \forall r \in R, \quad (14)$$

где p_j – оценка задачи, h_r – объем спринта, который может выполнить исполнитель, R – множество исполнителей, J – множество задач.

Задачи каким-либо образом упорядочены. Тогда на каждом i -ом уровне дерева, представленного на рис. 3, рассматривается i -ая задача. Ветвление происходит назначением i -ой задачи одному из исполнителей. Определение верхней границы происходит следующим образом: для нераспределенных задач выбирается наибольшая оценка среди всех возможных исполнителей. Находим верхнюю границу, решая релаксированную задачу, в которой отброшено ограничение на возможности исполнителя.

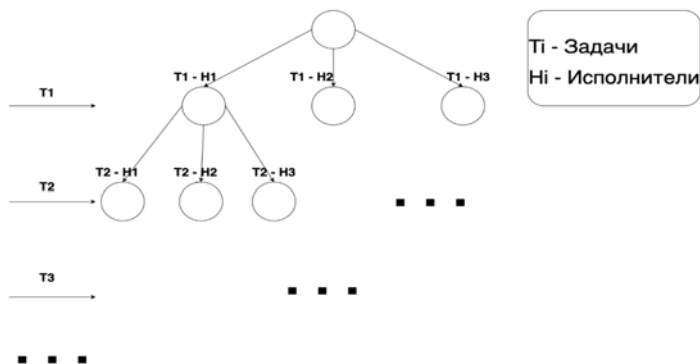


Рис. 3. Дерево решений в методе ветвей и границ

5. Реализация разработанной системы автоматического распределения задач

В качестве технологий для реализации были использованы следующие: TypeScript, React, API браузерных расширений Manifest V3, Jira API, FireBase.

Архитектура системы представлена на рис. 4. С помощью API получаем данные из task-трекера. Обрабатываем в нашем расширении, и сохраняем промежуточные результаты в облачном хранилище.

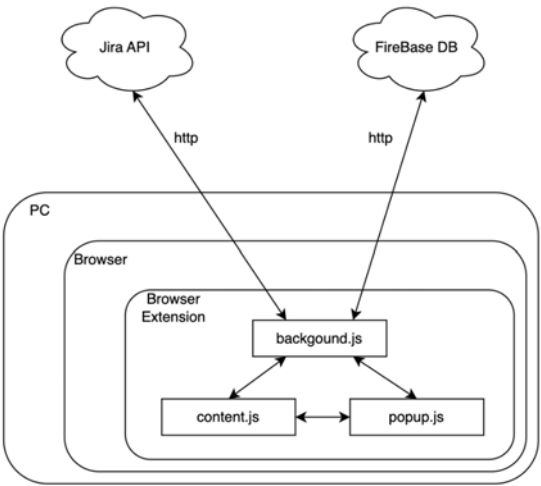


Рис. 4. Архитектура системы

Пример работы расширения представлен на рис. 5-7. Пользователь видит все доступные задачи. Далее он может оценить каждую из них. И после того, как все члены команды выставили свои предпочтения, мы получаем итоговое распределение задач между исполнителями.

Привет, User 1!

Проект: DU

Тип учетной записи: исполнитель

Показать / скрыть задачи

Тип задачи	Ключ задачи	ID задачи	Приоритет	Исполнитель	
Task	DU-1	10001	Средний	-	Оценить
Bug	DU-2	10002	Высокий	-	Оценить
Story	DU-3	10003	Низкий	-	Оценить

Рис. 5. Интерфейс со всеми доступными задачами

DU-2

Тип задачи: Task

ID задачи: 10002

Описание: #описание задачи

Исполнитель: -

Приоритет: Высокий

Статус: В разработке

Дата создания: 24.04.2024

Интерес

Сложность

Посильность

Очень низкая

Низкая

Средняя

Высокая

Очень высокая

Сохранить

Рис. 6. Интерфейс оценивания конкретной задачи

Тип задачи	Ключ задачи	ID задачи	Приоритет	Исполнитель
Task	DU-1	10001	Средний	User 3
Bug	DU-2	10002	Высокий	User 1
Story	DU-3	10003	Низкий	User 2

Рис. 7. Интерфейс итогового распределения задач между исполнителями

Заключение

Данная статья посвящена разработке и реализации многопользовательской системы автоматического распределения задач с учетом предпочтений руководителя и членов проектной команды. Описан метод многокритериального выбора – нечеткий TOPSIS, метод комбинаторной оптимизации – метод ветвей и границ и создание самой разрабатываемой системы. В итоге данной разработки получился полностью рабочий программный модуль, реализованный в виде браузерного расширения. Использование данного решения позволяет автоматически распределять задачи с учетом предпочтений руководителя и членов проектной команды. В дальнейшем предполагается использование данного расширения на любых проектах, основанных на таск-трекере Jira.

Список литературы

1. Hwang, C. L., & Yoon, K. (1981). Multiple attribute decision making: Methods and applications. Springer-Verlag. <http://dx.doi.org/10.1007/978-3-642-48318-9>
2. Roth, A. E. (1985). Common and conflicting interests in two-sided matching markets. *European Economic Review*, 27, 75-96. [https://doi.org/10.1016/0014-2921\(85\)90007-8](https://doi.org/10.1016/0014-2921(85)90007-8)
3. Yu, B., & Yue, Q. (2017). Extend TOPSIS-Based Two-Sided Matching Decision in Incomplete Indifferent Order Relations Setting Considering Matching Aspirations. *Eurasia Journal of Mathematics, Science and Technology Education*, 13(12), 8155-8168. <https://doi.org/10.12973/ejmste/78100>
4. Афанасьева, Т. В., Наместников, А. М., Перфильева, И. Г., Романов, А. А., & Ярушкина, Н. Г. (2014). Прогнозирование временных рядов: Нечёткие модели. Ульяновск: УлГТУ.
5. Ярушкина, Н. Г., Перфильева, И. Г., & Афанасьева, Т. В. (2010b). Интеграция нечетких моделей для анализа временных рядов. *Известия Самарского научного центра Российской академии наук*, 12 (4-2), 506-509.
6. Wang, J., Li, W., Chen, Y., Li, B., & Yang, S.-L. (2018). A novel decision making method for two-way referral based on SMAA-2 and two-sided matching model. *Journal of Discrete Mathematical Sciences and Cryptography*, 21(2), 571-576. <https://doi.org/10.1080/09720529.2018.1453660>
7. Ragone, A., Straccia, U., Di Noia, T., Di Sciascio, E., & Donini, F. M. (2009). Fuzzy matchmaking in e-marketplaces of peer entities using Datalog. *Fuzzy Sets and Systems*, 160, 251-268. <https://doi.org/10.1016/j.fss.2008.07.002>

Аппаратная платформа системы биологической обратной связи в игровой форме

Л. А. Малаков

Студент магистр

А. В. Матвеев

Студент магистр

А. В. Максимов

Старший преподаватель

Введение

В настоящее время состояние окружающей среды и воздуха ухудшается. Многие люди ведут малоподвижный образ жизни и не занимаются спортом. Всё это может привести к нарушению функционирования дыхательной системы. Так же стоит учитывать заболевания, передающиеся по наследству. Для предотвращения, а также терапии заболеваний дыхательной системы, разрабатывается данный аппаратно-программный комплекс.

Цель данной работы состоит в создании аппаратно-программного комплекса, для измерения некоторых спирометрических показателей [1] с дальнейшим применением в составе программного обеспечения для терапии и тренировки дыхания, при заболеваниях дыхательной системы человека.

На текущий момент подобные проекты заключаются лишь в физической реализации игр, для тренировки дыхания, как например продукт компании Breathcise, который реализует игру “Баскетбол”. Проблема данного продукта в его не адаптивности и ограниченности функционала. Разрабатываемый проект, решает данные проблемы, путём перехода от физической реализации игр к их компьютерной адаптации.

1. Функционал разрабатываемого продукта

Тренировочная программа выполняется пользователем с помощью специального контроллера. Данные с датчиков поступают по сети Bluetooth в приложение на стационарный компьютер или мобильное устройство.

Система содержит две симуляции для тренировки дыхания. Один из тренажеров предназначен для развития максимальной скорости выдоха. Пользователь управляет дальностью полета снаряда с помощью дыхания и должен поразить цели, расположенные на различных дистанциях. Динамика появления целей определяется входными данными. Наведение по горизонтали осуществляется путем изменения ориентации контроллера в пространстве.



Рис. 1. Тренажер максимальной скорости выдоха

Другая симуляция представляет собой медитативный дыхательный тренажёр. Выдыхая, пользователь сообщает летящему листу вертикальное ускорение. Горизонтальное положение задается поворотом контроллера. Цель – преодолевать расположенные на пути препятствия, конфигурация и время появления которых задается входными данными. Графическое окружение собирается динамически из модульных ресурсов.

2. Аппаратная реализация контроллера

Аппаратную часть проекта представляет платформа Arduino Pro mini. Она была выбрана, поскольку является очень удобным средством для реализации поставленной задачи, в силу наличия множества подключаемых модулей, а также из-за имеющегося удобного программного обеспечения, предоставляющего функционал по написанию программ для микроконтроллера. Ещё не маловажным фактором стало малый размер платы, при этом контроллер обеспечивает достаточную вычислительную мощность в рамках решения поставленной задачи.



Рис. 2. Медитативный тренажер

Далее описаны компоненты, из которых состоит готовое устройство и их назначение в проекте:

- Определение положения в пространстве достигается за счёт комбинированного датчика MPU 6050, содержащего в себе гироскоп и акселерометр.

- Датчик приближения – это оптический модуль, главным элементом которого является оптопара TCRT5000, состоящая из инфракрасного светодиода и фототранзистора. На лопасти микротурбины нанесён фольгированное покрытие, для отражения инфракрасных лучей.

- Микротурбина была создана при помощи 3д печати и заранее заготовленной 3д модели для её построения, вращение лопастей турбины обуславливается наличием магнитной подушки и железного сердечника в центре крыльчатки. Магниты расположены разными полюсами напротив друг друга, обеспечивая магнитные поля разных полюсов такие что заставляют крыльчатку всегда принимать вертикальное положение, относительно корпуса микротурбины.

- Система может работать без постоянного подключения к источнику питания. Автономность обеспечивается за счёт использования аккумулятора с выходным током в 3.7 В и ёмкостью в 450 мАч.

Поскольку аппаратная часть представляет собой набор модулей, то возникла необходимость в соединении и монтаже всех компонентов.

Первый образец устройства был собран на основе макетной платы, поскольку она является удобным инструментом для отладки и тестирования всех составляющих системы как по отдельности, так и в составе единого устройства.

На данный момент устройство собрано в корпус, распечатанный на 3д принтере и является уже не набором отдельных компонент, а монолитным прибором. Самым удобным способом вывода информации при такой компоновке является соединение через Bluetooth.

3. Программная реализация контроллера

Прошивка представляет собой систему сбора информации с датчиков и правильной их интерпретации. Сбор данных осуществляется путём опроса соответствующих контактов, подключенных к датчикам, так датчик приближения подключен к контакту, отвечающему за систему прерываний, для обработки которых, написана соответствующая функция. Датчик приближения подключен по шине i2c и передаёт своё положение в пространстве в виде линейного ускорения и проекции угловой скорости по каждой из осей, далее эти данные проходят через фильтр Калмана и на выходе получаются углы наклона относительно осей, при этом значения из фильтра Калмана [2] выходят с минимальным количеством “шумов” и погрешностей. Полученные данные с датчиков передаются в порт последовательного вывода информации, который посылает информацию на принимающее устройство через Bluetooth.

4. Реализация программы виртуального тренажера

Основной платформой для разработки программного обеспечения выбрана среда Unity, что позволяет создавать приложения, работающие на широком ряде операционных систем, как настольных, так и мобильных. Графическая составляющая среды поддерживает стандарт DirectX, эксклюзивный для ОС Windows, классический для Linux и Android OpenGL, Vulkan и Metal, характерный для iOS устройств. Шейдерные программы, требуемые для сообщения графическому процессору инструкций по отрисовке геометрии, являются компиляцией вершинного и пиксельного проходов и описывается языком HLSL, что далее транслируется встроенными инструментами в понятные каждому из API форматы [3]. В качестве скриптового языка используется C# на базе open-source аналога системы .NET – Mono.

В качестве визуальных ресурсов использованы как готовые решения, адаптированные под потребности приложения, так и авторские. Ресурсы включают трехмерные модели и текстурные карты. Трехмерная геометрия представлена в виде наборов данных о вершинах:

их положении в пространстве, направлениях векторов нормалей, текстурных координат, отображающих трехмерный вектор в двумерное пространство, и наборов индексов вершин, образующих полигоны. Текстуры карты подразделяются на цветные изображения, карты нормалей и затенения. Широкое применение имеет использование детальных карт – повторяющихся изображений низкого разрешения и масштаба, накладываемых поверх более общих текстур для придания фактуры поверхностям при близком рассмотрении.

Процедурная генерация тренировочных сред использует конфигурационные файлы для воссоздания требуемого окружения. Для построения визуальных образов применяются бесшовно соединяемые плитки и универсальные вариативно конфигурируемые ресурсы.

Приложение принимает с контроллера по протоколу Bluetooth в реальном времени поток данных, включающих информацию о положении устройства в пространстве и скорости вращения ротора, интерпретируемых программой определенным образом, обеспечивая контроль над тренировочной сценой.

В рамках проекта существует понятие “сценарий”, подразумевающее очередь событий, оказывающих влияние на виртуальную среду, что побуждает пользователя выполнять действия, предполагаемые программой тренировки. Сценарии хранятся в текстовых файлах и могут быть заданы вручную или через встроенный редактор.

5. Организация взаимодействия систем

В качестве канала связи между контроллером и приложением на целевом устройстве использована сеть Bluetooth. Основой для поддержки протокола на ОС Windows, с учетом определенного ранее стека технологий, выбран широко известный в разрезе .NET open-source проект 32feet.NET. Android приложение использует библиотеку нативных функций, написанную на языке Java. Контроллер, в свою очередь, оснащен Bluetooth модулем hm-06. Приложение получает от контроллера потоковые данные о скорости вращения ротора и положении устройства в пространстве, применяемые в дальнейшем для выполнения пользователем определенных упражнений.

Заключение

Данная работа описывает принципы функционирования программно-аппаратного комплекса, разрабатываемого для обеспечения дополнительной физической нагрузки при заболеваниях дыхательной системы или используемого в качестве профилактики данных заболеваний.

Список литературы

1. Новик, Г. А. Спирометрия и пикфлоуметрия при бронхиальной астме у детей / Г. А. Новик, А.В. Боричев. – СПб.: ГПМА, 2005. – 69 с.
2. A practical approach to Kalman filter and how to implement it [Электронный ресурс] : TKJ Electronics. – Режим доступа: <http://blog.tkjelectronics.dk/2012/09/a-practical-approach-to-kalman-filter-and-how-to-implement-it>
3. Vries J. D. Learn OpenGL – graphics programming / J. D. Vries. – Kendall & Welling, 2020. – 523 с.

Применение больших языковых моделей для дополнительной разметки данных в задаче оценки времени на исправление дефекта

С. С. Масленникова

Студент магистр

В. В. Коротков

Старший преподаватель

Введение

Оценка времени на исправление дефектов является важным этапом в разработке программного обеспечения, поскольку позволяет эффективно управлять ресурсами и временем команды разработчиков. Однако точное предсказание времени, требуемого для устранения дефектов, часто оказывается сложной задачей из-за разнообразия и сложности самих дефектов. В том числе, оценка может затрудняться, если описание дефекта содержит вставки кода. В данной работе мы исследуем возможности применения больших языковых моделей для автоматической разметки данных в задаче оценки времени на исправление дефектов.

1. Постановка задачи

Исследование представляет собой задачу автоматического выделения кодовых вставок из текстовых описаний тикетов, чтобы оценить их влияние на временные затраты на выполнение задач. Цель состоит в разработке модели машинного обучения, способной определять присутствие кода в тексте тикета и генерировать JSON-структуру, содержащую обнаруженные кодовые фрагменты вместе с соответствующей дополнительной информацией. Входные данные представляют собой текстовые описания дефектов, а выходные данные – JSON-объекты с выделенными кодовыми вставками и сопутствующей метainформацией.

2. Разметка и подготовка данных

Для исследования использовалась выборка из The Public Jira Dataset [1], содержащая информацию о дефектах из различных проектов, включая около 2000 записей для проекта Spring.

Для автоматизации процесса разметки была разработана промпт-инструкция для модели GPT-3.5. Выбор данной модели обусловлен оптимальным соотношением между стоимостью и качеством генерации. Промпт содержал описание задачи для модели, ожидаемый формат выходных данных, а также предусматривал заполнение информации о каждом дефекте. Модель должна была выделять все вхождения кода, определять язык программирования и тип содержимого выделенного фрагмента (логи, код, описание ошибки). Разметка выполнялась с использованием API GPT-3.5. Для этого был создан скрипт, отправляющий промпт и текст тикета, а затем сохраняющий результаты.

После разметки результаты генерации прошли дополнительную валидацию:

1. С использованием скрипта пост-обработки проверялось соответствие результатов формату JSON-структуры. Несмотря на использование температуры $T = 0$, модель GPT-3.5 не всегда точно следовала инструкциям, и некорректные ответы были отфильтрованы на этом этапе.

2. Выделенные фрагменты кода прошли ручную валидацию.

В результате *precision* метрики для определения фрагментов кода в провалидированных результатах составила 0,998.

После дополнительной разметки был сформирован датасет для тонкой настройки. Он содержал информацию в формате: инструкция, входной текст (описание дефекта), выходной текст (валидированный результат генерации).

3. Квантизация, PEFT, LoRA

Развитие больших языковых моделей за последние годы было связано в том числе с увеличением количества параметров. Однако чем больше такая модель, тем больше памяти она требует для хранения и работы. Это является препятствием для запуска на многих устройствах из-за ограничений памяти графических ускорителей и вычислительной мощности процессоров. Для решения этой проблемы была придумана техника квантизации моделей.

Квантизация – это техника сжатия, которая преобразует веса и функции активации модели из пространства с высокой точностью представления в пространство с низкой точностью представления [2]. Например, это переход от 32-битного формата представления числа с плавающей точкой FP32 в 8-битное целое число. Такая трансформация может приводить к небольшим потерям в качестве модели, но при этом значительно расширяет возможности использования и развертывания. В некоторых случаях это довольно просто реализовать: например, при переходе от FP32 к FP16, так как оба этих формата схожи в

представлении данных (оба – это числа с плавающей точкой). Тем не менее, переход от FP32 к INT4 выполнить сложнее, так как INT4 может представлять всего 16 различных целочисленных значений.

Можно выделить две категории квантизованных моделей:

- *Post-Training Quantization (PTQ, квантизация после обучения)*: преобразования производятся после обучения модели. Этот способ проще в реализации и требует меньше данных, однако может приводить к потерям в точности работы модели.

- *Quantization-Aware Training (QAT, обучение с учетом квантизации)*: дообучение базовой модели происходят с учетом квантизации. В отличие от предыдущего метода, все преобразования включены в процесс обучения. Этот способ позволяет получить модель лучшего качества, но требует больших вычислительных затрат.

PEFT (Parameter Efficient Fine-Tuning) представляет собой метод тонкой настройки модели, предназначенный для улучшения производительности предварительно обученных языковых моделей. Основной концепцией PEFT является настройка только небольшого подмножества параметров предварительно обученной большой языковой модели, оставляя большинство параметров замороженными [3].

Преимущества метода PEFT заключаются в следующем:

- Ускорение процесса обучения;
- Экономия вычислительных ресурсов и пространства для хранения;
- Снижение риска переобучения, поскольку большая часть параметров модели остается замороженной;
- Преодоление проблемы катастрофического забывания: ранее полученные знания сохраняются благодаря замороженным параметрам;
- Простота развертывания.

В 2021 году был представлен метод *LoRA (Low-Rank Adaptation)*, как одна из разновидностей PEFT. Предыдущие исследования указывали на то, что большие языковые модели обладают низкой "внутренней размерностью" при адаптации к новым задачам. Это означает, что большая часть параметров модели является избыточной. Следовательно, матрицы параметров моделей могут быть представлены в пространстве меньшей размерности, сохраняя при этом существенную часть информации [4].

Каждая матрица параметров W представлена в виде суммы

$$W + AB \tag{1}$$

во время обучения. Исходная матрица W заморожена, и обучаются только матрицы A и B .

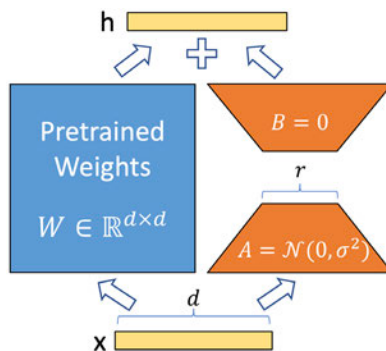


Рис. 1. Иллюстрация работы LoRA

Следовательно, обновленные веса получаются как

$$\Delta W = W + AB \quad (2)$$

Благодаря тому, что матрицы A и B остаются маленькими, процесс обучения становится быстрее и требует меньше ресурсов.

Ранг матрицы r – это гиперпараметр, который может быть использован для указания матриц низкого ранга, используемых для адаптации. Меньший r приводит к более простой матрице низкого ранга и, следовательно, меньшему количеству параметров во время адаптации. Выбор меньшего r предполагает компромисс между сложностью модели, адаптационными возможностями и риском недостаточной или чрезмерной подгонки.

4. Обучение модели

Для тонкой настройки была выбрана модель Llama-3-8b-Instruct [5]. Это компактная модель с открытыми весами, которая является одной из лучших в рейтинге ChatBot Arena [6]. Она представляет собой декодер. Размерность словаря составляет 128 000 токенов, а размер контекстного окна – около 8 000 токенов.

Тонкая настройка модели производилась с помощью фреймворка unsloth. Он предоставляет FP4 квантизованную версию модели, а также интерфейс для использования LoRA. В качестве гиперпараметров были выбраны:

- алгоритм оптимизации AdamW
- максимальное число шагов при обучении – 60
- $r = 16$

На рис. 2 представлен график значений функции потерь в процессе тренировки. Из графика можно сделать вывод, что модель сходится и обучение прошло успешно.

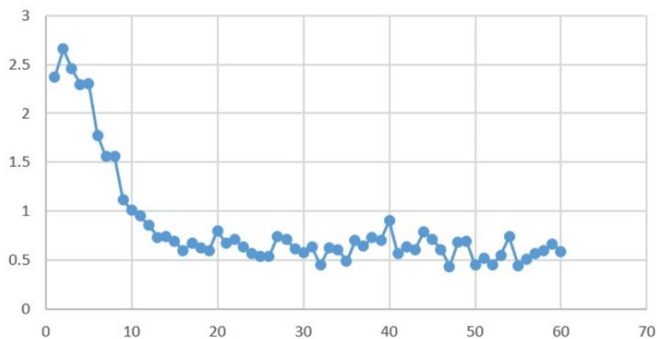


Рис. 2. Изменения функции потерь при тонкой настройке

Заключение

В ходе данной работы была разработана модель для поиска вставок кода в тексте. Для этого был составлен набор данных на основе The Public Jira Dataset, проведена разметка с помощью GPT 3.5 и ручная валидация результатов генерации. Далее на размеченном датасете была произведена тонкая настройка квантизированной LLaMa3 8B Instruct с помощью LoRA. Полученная модель может быть использована как часть системы очистки и предобработки текста.

Список литературы

1. The Public Jira Dataset [Электронный ресурс] : набор данных. – Режим доступа : <https://zenodo.org/records/5901804>
2. Yao, Z. Exploring Post-training Quantization in LLMs from Comprehensive Study to Low Rank Compensation. / Yao, Z. [et al.] // Proceedings of the AAAI Conference on Artificial Intelligence 38(17) : сб. тр. участников. -2024. – С. 19377-19385.
3. Xu L. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment / Xu L. [et al.] // arXiv preprint arXiv:2312.12148. – 2023.
4. Hu E. J. Lora: Low-rank adaptation of large language models / Hu E. J. [et al.] // arXiv preprint arXiv:2106.09685. – 2021.
5. llama3modelcard [Электронный ресурс] : веса модели. – Режим доступа : https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md
6. Chiang W. L. Chatbot arena: An open platform for evaluating llms by human preference / Chiang W. L. [et al.] // arXiv preprint arXiv:2403.04132. – 2024.

Методы повышения энергической эффективности радиоканалов интернета вещей при реализации на микроконтроллере

А. О. Мосолов

Студент магистр

А. Ю. Савинков

Профессор

Введение

В системах интернета вещей и распределённого мониторинга при невозможности использования проводных каналов связи используются радиоканалы, например, использование радиоканалов необходимо:

- на объектах транспортно-логистической инфраструктуры, таких как железнодорожные узлы или складские комплексы;
- в сельскохозяйственной деятельности (мониторинг температуры и влажности почвы, посевов и др.);
- в экологической сфере, например для мониторинга лесов.

В системах мониторинга обычно требуется реализовать однонаправленную передачу данных от множества территориально распределённых датчиков на один принимающий узел, который будем называть концентратором. Удаление датчика от концентратора может составлять до нескольких километров, что с учетом ограниченной мощности передатчиков и ограниченных возможностях передающих антенн (низкое расположение, ограниченный размер, невозможность использования направленных антенн и др.) делает задачу построения радиоканалов в системах мониторинга достаточно сложной.

При этом должна быть обеспечена высокая энергетическая эффективность системы передачи данных, поскольку возможности электропитания автономных устройств ограничены, а частая замена элементов питания или подзарядка аккумуляторов невозможна. Под энергетической эффективностью системы передачи данных будем понимать энергию, затрачиваемую на передачу одного бита информации, усредненную по всем устройствам системы мониторинга.

Для повышения энергетической эффективности системы передачи данных можно использовать ретрансляцию сообщений удаленных

датчиков с использованием радиомодемов других датчиков системы мониторинга, расположенных ближе к концентратору. Ввиду быстрого затухания сигнала в радиоканале, за счет ретрансляции можно будет существенно снизить мощность передатчиков, по сравнению со случаем прямой передачи данных от каждого датчика на концентратор, так что даже с учетом необходимости дополнительных передач, суммарная энергия, затраченная на доставку сообщения от удаленного датчика до концентратора, снизится.

Для реализации ретрансляции сообщений через промежуточные узлы необходима система построения маршрутов передачи сообщений (выбора цепочки ретрансляторов). Построение таких маршрутов вручную трудозатратно и неэффективно, необходима система динамической маршрутизации для автоматического создания маршрутов с учетом особенностей систем распределенного мониторинга.

1. Классификация протоколов динамической маршрутизации

Под маршрутизацией понимается процесс определения пути следования информации в сетях передачи данных с целью выбора оптимального маршрута ее доставки.

Протоколы динамической маршрутизации делятся на проактивные и реактивные.

Проактивные протоколы работают с помощью предварительного формирования таблиц маршрутизации, в которых содержатся все известные пути. В таких случаях передача пакетов начинается немедленно, но требуется время и затраты на поиск маршрутов и формирование таблиц маршрутизации, так как необходима полная информация о топологии сети перед началом передачи. Кроме того, для поддержания актуальности маршрутной информации эти протоколы предполагают дополнительные расходы. Примерами таких протоколов являются DSDV (Destination Sequenced Distance Vector) и OLSR (Optimized Link State Routing). Сразу можно сказать, что для распределенного мониторинга данный подвид протоколов динамической маршрутизации не подходит так, расходы на постоянное поддержание таблиц маршрутизации не соотносятся с принципом энергоэффективности данных систем.

Реактивные протоколы также известны как протоколы по запросу. Узлы не хранят информацию о маршрутах на постоянной основе. Они начинают построение маршрута только тогда, когда поступает запрос на передачу данных.

Поиск маршрута основан на алгоритме наводнения: узел отправляет первый пакет всем своим соседям, а промежуточные узлы пересылают его к своим соседям. Этот процесс обеспечивает доставку

пакета до его адресата. Так же маршрут прохождения сохраняется для последующих передач (например, в виде списка узлов), что позволяет выбирать оптимальное направление передачи данных. Реактивные протоколы имеют меньшие накладные расходы на маршрутизацию по сравнению с проактивными, но большую задержку при инициализации процедуры передачи, так как маршрут устанавливается только при получении запроса на передачу данных. Примерами реактивных протоколов являются

- DSR (Dynamic Source Routing);
- AODV (Ad hoc On-Demand Distance Vector);
- TORA (Temporally Ordered Routing Algorithm);

2. Предлагаемое решение

Как мы выяснили использование проактивных протоколов в системе распределенного мониторинга невыгодно, так как для анализа связей в сети требуется постоянный обмен сообщениями между узлами, что быстро разрядит батарейки узлов, но в то же время недостаток реактивных протоколов это необходимость поиска маршрута перед каждым сеансом передачи данных.

Но в системе распределенного мониторинга узлы сети обычно не перемещаются и связность не меняется. Тогда хорошей идеей кажется использование реактивного протокола для поиска маршрута до концентратора один раз после включения узла и последующее использование найденного маршрута для периодической передачи результатов измерений. Главным критерием выбора маршрута будет энергозатратность, другими словами, количество энергии, затраченное на передачу данных по данному маршруту.

Таким образом, мы можем предложить собственный алгоритм маршрутизации, реализованный на базе аппаратной платформы от фирмы Texas Instrument LAUNCHXL-CC1310.

Блок-схема работы данного алгоритма представлена на рис. 1.

После включения питания узел, допустим это узел А на максимальной мощности передает запрос поиска маршрута до концентратора, в запросе указывается адрес узла, который отправил данный запрос

Каждый узел допустим это узел В и сам концентратор, который услышал данный запрос, в случае если он имеет записи маршрута до концентратора должен ответить на него, пакет ответа на запрос поиска маршрута должен содержать:

- адрес отправителя ответа
- RSSI принятого запроса

– Стоимость самого выгодного маршрута от отправителя ответа до концентратора

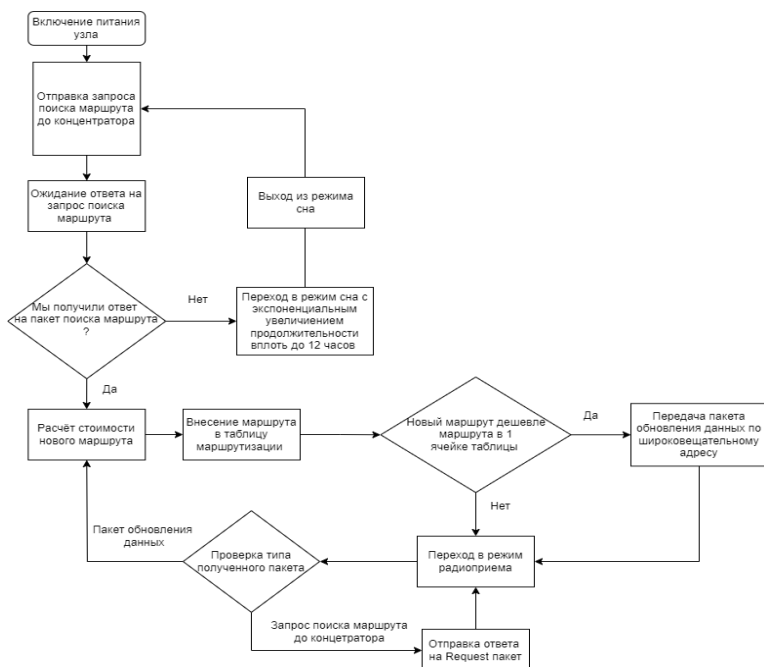


Рисунок. Блок-схема работы алгоритма маршрутизации

Стоимость можно рассчитать, например, следующим образом: пусть узел В знает маршрут до концентратора, он знает, что будет ретранслировать пакет концентратору через соседний узел. Соседний узел в свое время сообщил узлу В стоимость маршрута от себя до концентратора и значения RSSI, с которым он принял запрос поиска маршрута от узла В.

Узел В знает минимально допустимый уровень RSSI для данной аппаратной платформы и определяет требуемую мощность передачи. Эта мощность суммируется со стоимостью маршрута от С до концентратора, и сообщается узлу А как стоимость маршрута до концентратора через узел В. Если отправителем ответа является концентратор, то он ставит нулевую стоимость передачи данных.

Получая ответы, узел прибавляет к полученной стоимости мощность, которую он сам должен использовать при передаче пакета

через соответствующий ретранслятор, и добавляет данную запись в свою таблицу маршрутизации

Если какой-то узел обновил минимальную стоимость или впервые узнал стоимость пути до концентратора, он передает широкоэвещательный пакет обновления данных на максимальной мощности, который содержит:

- адрес отправителя;
- новую стоимость маршрута от себя до концентратора.

Все узлы, которые услышат ответ, могут обновить свои данные, в случае более выгодной стоимости данного маршрута.

Так же реализован механизм сна с экспоненциальным увеличением его продолжительности (максимум до 12 часов), в случае отсутствия ответа на запрос поиска маршрута, это сделано для увеличения энергоэффективности данного алгоритма

Заключение

Предложен алгоритм динамической маршрутизации для систем распределенного мониторинга, учитывающий специфику работы таких систем. В качестве критерия оптимальности для маршрута выбран минимум энергии, требуемый для доставки сообщения от датчика к концентратору. Реализовано автоматическое обновление маршрута при появлении более оптимального, например, при включении нового узла, а также возможность быстрого перехода на резервный маршрут при потере связи, например, при выключении (отказе) узла-ретранслятора.

Предложенный алгоритм динамической маршрутизации был реализован и испытан с использованием оценочных плат LAUNCHXL-CC1310 на основе микроконтроллера TI CC1310 со встроенным радиомодемом.

Список литературы

1. Дибров, М. В. Сети и телекоммуникации. Маршрутизация в IP-сетях: учебник и практикум для вузов. В 2 ч. / М. В. Дибров– М.: Юрайт, 2023. Ч. 1. – 333 с.
2. Джонсон, Д. Б. Динамическая маршрутизация от источника в беспроводных ad hoc сетях и мобильных компьютерных средах / Д. Б. Джонсон, Д. А. Мальц // Springer. –1996. С. 153-181.
3. Мишра, С. Энергоэффективная маршрутизация в мобильных ad hoc сетях: Обзор. / С. Мишра, Б.К. Паттаньяк // ARPN Journal of Engineering and Applied Sciences.8: –2013. –С. 173- 189.
4. LAUNCHXL-CC1310 [Электронный ресурс] : база данных. – Режим доступа: <https://www.ti.com/tool/LAUNCHXL-CC1310>

Разработка Telegram-бота для мониторинга курса валют: удобство и функциональность

Е. А. Нагина

Студент магистратуры

А. А. Вахтин

Доцент

Введение

Разработка Telegram-бота для мониторинга курса валют является актуальной в современном мире, где мессенджеры играют ключевую роль в коммуникации и получении информации. С увеличением числа пользователей мессенджеров и их удобством использования, предоставление финансовой информации через такие платформы как, например, Telegram становится все более востребованным. Люди ценят возможность мониторинга курсов валют в реальном времени прямо в мессенджере, что делает разработку и внедрение подобных ботов востребованными. Кроме того, развитие технологий искусственного интеллекта позволяет создавать более умных и адаптивных ботов, способных предоставлять персонализированные рекомендации и помогать пользователям в принятии финансовых решений.

1. Концепция Telegram-бота

Боты – специальные аккаунты в Telegram, созданные для того, чтобы автоматически обрабатывать и отправлять сообщения. Пользователи могут взаимодействовать с ботами при помощи сообщений, отправляемых через обычные или групповые чаты. Логика бота контролируется при помощи HTTPS запросов к API для ботов [1]. Таким образом, Telegram предоставляет удобный интерфейс для любого проекта на базе запросов.

Рассмотрим механизм взаимодействия компонентов системы Telegram-бота. Приложение клиента Telegram может быть установлено на смартфоне, компьютере или веб-браузере, однако это не имеет значения для понимания логики работы (см. рис. 1).

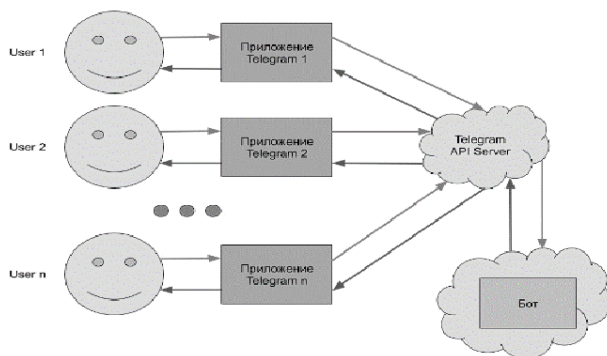


Рис. 1. Схема взаимодействия с Telegram-ботом

Когда пользователь взаимодействует с ботом через приложение Telegram (например, отправляет сообщение, нажимает кнопку или выбирает команду в меню), приложение отправляет запрос на сервер Telegram. Сервер определяет, что это запрос к боту, и в зависимости от настроек бота либо отправляет запрос к боту, либо ожидает запрос от него. Существуют две основные технологии для получения ботом обновлений от сервера – Long polling и Webhook [2].

Long polling заключается в постоянном опросе сервера на предмет наличия обновлений для бота. Webhook – это когда сам сервер уведомляет бота о полученных сообщениях. Long polling обычно используется на этапе разработки и тестирования бота, а Webhook – когда бот уже запущен в реальной работе.

Таким образом, бот получает обновления от сервера, обрабатывает их и отправляет результат обратно на сервер. Сервер определяет, кому из пользователей предназначено сообщение, и передает его в приложение пользователя. Взаимодействие между ботом и сервером осуществляется с использованием http-запросов.

2. Этапы разработки и проектирования Telegram-бота

Разработка и проектирование Telegram-бота для мониторинга курса валют включает несколько этапов, каждый из которых играет важную роль в создании полноценного и эффективного приложения. Основные этапы включают в себя:

1. Анализ требований: на этом этапе определяются основные требования к функциональности бота. Это может включать в себя определение основных функций (например, получение курсов валют, конвертацию валюты, просмотр истории запросов), интерфейса пользователя и выявление основных целей проекта.

2. Проектирование интерфейса: этот этап включает в себя проектирование интерфейса пользователя (UI) и определение пользовательского опыта (UX). Разрабатываются макеты экранов, кнопок и других элементов управления, которые будут использоваться в боте.

3. Выбор технологий: на этом этапе определяются технологии и инструменты, которые будут использоваться при разработке бота. Например, для работы с Telegram API можно выбрать библиотеку `telebot` для Python [3]. Также необходимо выбрать источник данных о курсах валют и способ хранения истории запросов пользователей (например, база данных SQLite).

4. Разработка архитектуры: на этом этапе определяется общая архитектура приложения, включая структуру файлов и каталогов, модули и компоненты, которые будут использоваться для реализации функциональности бота.

5. Разработка функциональности: создание кода для реализации основных функций бота, таких как получение курсов валют, конвертация валюты и управление историей запросов пользователей. В этом процессе также проводится тестирование функциональности для обеспечения корректной работы приложения.

6. Тестирование и отладка: проведение тестирования приложения для выявления ошибок, недочетов и потенциальных проблем. После обнаружения ошибок осуществляется их исправление и отладка кода.

3. Архитектура Telegram-бота для мониторинга курса валют.

Архитектура разработанного Telegram-бота основывается на модульном подходе, который обеспечивает гибкость, масштабируемость и простоту в поддержке и дальнейшем расширении функциональности [4]. Бот включает в себя следующие основные компоненты:

Модуль взаимодействия с Telegram API: отвечает за обработку входящих запросов от клиентов Telegram и отправку ответов. Он использует библиотеку `telebot` для удобного взаимодействия с Telegram Bot API. Модуль реализует обработчики для различных типов входящих сообщений;

Модуль работы с внешним API курсов валют: для получения актуальных данных о курсах валют бот взаимодействует с внешним API (листинг 1), предоставляющим информацию о курсах обмена валют. Для этого используется библиотека `requests` для отправки HTTP-запросов к API и получения ответов в формате JSON [5]. Модуль обрабатывает полученные данные и предоставляет их другим компонентам системы;

Запрос к внешнему API и обработки полученных данных

```
import requests

def get_exchange_rates(base_currency):
    try:
        response = requests.get(f"https://open.er-
api.com/v6/latest/{base_currency}")
        response.raise_for_status()
        data = response.json()
        if data["result"] == "success":
            # Обработка полученных данных
            return rates
        else:
            return None
    except requests.exceptions.RequestException as e:
        print(f"Ошибка при получении курсов валют: {e}")
        return None
```

Модуль работы с базой данных: для хранения истории запросов пользователей бот использует базу данных SQLite (листинг 2). Модуль работы с базой данных реализует функциональность сохранения и извлечения данных из базы, а также управление структурой таблицы и выполнение SQL-запросов;

Создание и выполнение SQL-запроса

```
import sqlite3

conn = sqlite3.connect('database.db')
cursor = conn.cursor()
cursor.execute('''CREATE TABLE IF NOT EXISTS requests_history
(
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    user_id INTEGER NOT NULL,
    base_currency TEXT NOT NULL,
    target_currency TEXT NOT NULL,
    amount REAL NOT NULL,
    timestamp TIMESTAMP DEFAULT
CURRENT_TIMESTAMP
)''')
conn.commit()
conn.close()
```

Модуль бизнес-логики: содержит основную бизнес-логику приложения, включая функции получения курсов валют, конвертации валюты и управления историей запросов пользователей. Он

взаимодействует с другими модулями для получения данных и обработки запросов пользователей.

Модуль управления пользовательским интерфейсом: отвечает за формирование и управление пользовательским интерфейсом бота. Он создает и отправляет текстовые и графические сообщения пользователю (листинг 3), а также управляет клавиатурой и другими элементами интерфейса для удобного взаимодействия с пользователем.

Листинг 3

Отправка сообщения пользователю

```
bot.send_message(message.chat.id, "Привет! Что вы хотите сделать?")
```

На рис. 2 показана связь между модулями Telegram-бота для мониторинга курса валют.

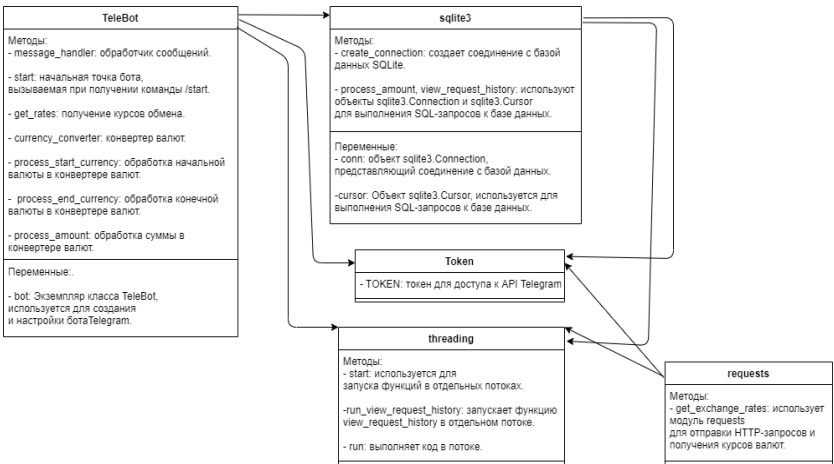


Рис. 2. Схема связи модулей Telegram-бота для мониторинга курса валют

Заключение

Данная статья посвящена разработке и реализации Telegram-бота для мониторинга курса валют. Описаны основные алгоритмы функционирования бота, включая получение актуальных курсов валют, конвертацию валюты и управление историей запросов пользователей.

В результате данной разработки был создан полностью функциональный Telegram-бот, реализованный с использованием

библиотеки telebot для Python. Бот обеспечивает возможность взаимодействия с пользователями через платформу Telegram, предоставляя им информацию о текущих курсах валют и возможность конвертации валюты прямо в чате.

Использование данного решения позволяет пользователям удобно отслеживать изменения курсов валют и проводить операции конвертации прямо в мессенджере Telegram. Кроме того, бот может быть использован в качестве инструмента для обучения и тестирования навыков работы с API и базами данных SQLite.

В дальнейшем планируется развитие и улучшение функциональности бота, а также его интеграция в системы обработки финансовых данных и аналитики.

Список литературы

1. Справочник по Bot API [Электронный ресурс]. – Режим доступа : <https://tlgm.ru/docs/bots/api>
2. Успенская, О. И. О тенденциях развития и состоянии сектора ИКТ в России в современных условиях / О. И. Успенская, Е. И. Солдатов, И. Н. Солдатов // Современные наукоемкие технологии. Региональное приложение. – 2019. – № 3 (59). – С. 59-63.
3. Яворски, М. Python. Лучшие практики и инструменты / М. Яворски, Т. Зиاده. – 4-е изд. – Санкт-Петербург: Питер, 2021. – 560 с.
4. Шумилина, М. А. Разработка чат-бота на языке программирования Python в мессенджере "Telegram" / М. А. Шумилина, А. В. Коробко // Научные известия. – 2022. – № 28. – С. 47–55.
5. Telegram Bot API (documentation) [Электронный ресурс]. – Режим доступа : <https://core.telegram.org/bots/api>

Прототип решения для работы в web-браузере с помощью движения глаз

Е. Д. Нартов

Студент магистр

Д. И. Соломатин

Старший преподаватель

Введение

На данный момент роль компьютерного зрения и технологии eye-tracking в веб-разработке становится всё более значимой. Уникальные возможности, предоставляемые этими инновационными методами анализа взгляда пользователя, позволяют совершенствовать интерфейсы веб-сайтов и делать пользовательский опыт более интуитивным и эффективным.

Первоначально данные технологии создавались для применения в нейропсихологии и для облегчения коммуникации с людьми, у которых есть ограниченные возможности. Однако в дальнейшем она нашла применение и в других областях, таких как тренажеры для обучения пилотов, разработка автомобильных приборных панелей, компьютерные игры, а также в криминалистике, где отслеживание взгляда дополняет использование детекторов лжи [1]. Помимо анализа поведения пользователя на сайте, данную технологию можно использовать как средство ввода информации с помощью взгляда. Данная функциональность позволит создавать новые формы взаимодействия с элементами сайта и управлять компьютером без использования рук или голоса.

Целью данной работы является создание модуля в виде расширения для браузера Chrome, который позволит пользователю управлять элементами сайта с помощью зрения.

1. Технологии отслеживания взгляда

Перед созданием приложения необходимо выбрать способ отслеживания взгляда пользователя для контролирования элементов экрана. На данный момент эту задачу могут помочь решить следующие инструменты:

1. Технология Eye-tracking, которая использует камеры или инфракрасные датчики для отслеживания движения глаз и фиксирования точки фокусировки.
2. Инфракрасные датчики, которые используют инфракрасное излучение для отслеживания движения глаз и точки фокусировки на экране.
3. Методы компьютерного зрения и алгоритмы машинного обучения, которые анализируют изображения и видео для определения точки фокусировки глаз пользователя [2].

Для поставленной цели был выбран 3 вариант, так как данная технология может быть встроена в различные устройства и программное обеспечение без дополнительных затрат на специальное оборудование.

В качестве алгоритма машинного обучения была выбрана свёрточная нейронная сеть (CNN) ввиду её следующих преимуществ:

- Имеет многоуровневую архитектуру, которая предполагает постепенное извлечение все более абстрактных особенностей изображения на каждом уровне свертки и пулинга. Благодаря чему можно эффективно распознавать образы и паттерны на изображениях [3].
- Частичная устойчивость к масштабу, поворотам, смене ракурса и смещениям изображения, что поможет отслеживать глаза на различных разрешениях камеры и позиции глаз пользователя.
- Высокая скорость обучения и работы, что позволяет быстро и эффективно обрабатывать изображения на сайте и отслеживать взгляд пользователя в реальном времени.

2. Разработка модуля для отслеживания взгляда

Для реализации данного модуля был использован язык TypeScript с использованием Angular фреймворка [4]. TensorFlow был выбран в качестве библиотеки для машинного обучения. Clmtrackr [5] был использован для подгонки моделей лица к лицам с видео.

На рис. 1 приведена диаграмма классов данного приложения, которая включает в себя:

- App.module – основной класс подмодуля, включающий в себя все необходимые компоненты.
- Train-app.component – основной класс-компонент приложения, отвечающий за обработку и отрисовку формы взаимодействия пользователя с обучаемой нейронной сетью. Предоставляет пользователю шаблон (html страницу) со следующей функциональностью: создание, импорт и экспорт датасета;

создание, импорт и экспорт модели; отображение/скрытие точки взгляда пользователя.

- `Camera.component` – класс-компонент приложения, отвечающий за обработку и отрисовку видео с камеры пользователя и маской лица. Предоставляет другим классам информацию об изображении глаз пользователя и их координатах на кадре видео.
- `Eye-target.component` – класс-компонент приложения, отвечающий за отрисовку положения взгляда пользователя на экране, а также состояния глаз (открыты, закрыты).
- `CurrentTrainData` – класс-сервис, хранящий информацию о глазах пользователя, полученную от `camera.component`.
- `Dataset` – класс-сервис, хранящий и обрабатывающий информацию о собранных данных пользователя для обучения, такую как изображение глаз пользователя, их положение относительно кадра видео, а также координаты точки взгляда пользователя, считываемые с положения мыши.
- `TrackModel` – класс-сервис для создания, хранения и использования модели свёрточной нейронной сети.
- `EyeTrack` – класс-сервис, использующий обученную модель и данные с камеры пользователя, предоставляющий актуальную информацию о положении взгляда пользователя на экране, такую как координаты взгляда и состояния левого и правого глаза.

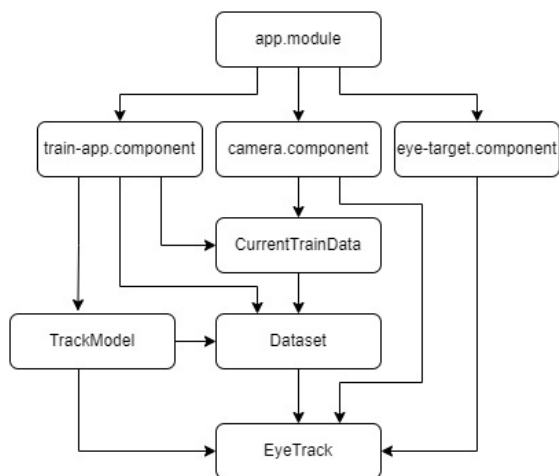


Рис. 1. Диаграмма классов модуля для отслеживания взгляда

Используя главную страницу модуля, пользователю нужно подключить веб-камеру и начать собирать данные, отслеживая мышь на экране. При каждом нажатии мыши в Dataset записываются координаты мыши относительно экрана, а также нормализованное изображение глаз в виде трехмерного массива с координатами x , y и яркостью пикселя.

После сбора необходимого количества данных для обучения создаётся модель свёрточной нейронной сети, в которую входят 2 входных слоя. Первый входной слой содержит изображение глаза с указанными размерами высоты, ширины и тремя каналами (RGB). Второй входной слой содержит метаданные изображения о положении и масштабе относительно кадра видео. К первому входному слою применяется слой свертки Conv2D, а после него слой MaxPooling2D, которые в совокупности выделяют наиболее важные признаки из входного изображения. Далее применяется слой выравнивания данных и слой объединения с входными метаданными. В конце добавляется выходной слой с функцией активации 'tanh' который применяется к выходу от предыдущих. После его применения многомерный входной тензор будет преобразован в одномерный выходной тензор с 2 нейронами, которые будут отвечать за координаты взгляда пользователя по входному изображению. Архитектуру данной сети можно увидеть на рис. 2.

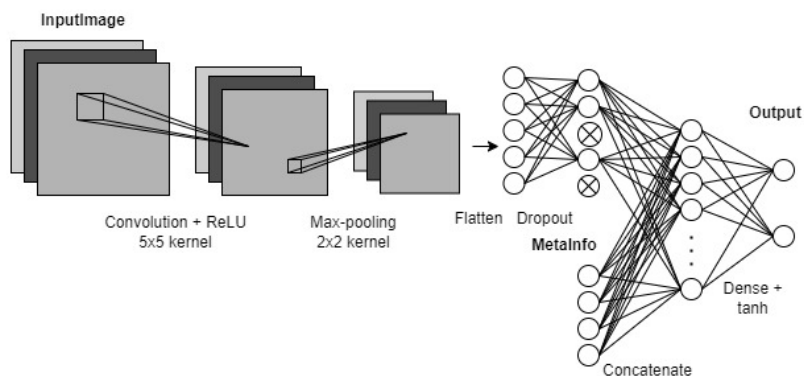


Рис. 2. Архитектура свёрточной нейронной сети

Таким образом, используя созданную модель и передавая в класс EyeTrack актуальную информацию о кадре глаза пользователя, можно определять примерное расположение направления взгляда пользователя при определённом обученном положении.

3. Распознавание моргания

Помимо отслеживания взгляда пользователя необходим, инструмент для совершения действия на сайте. Для этого необходимо в модуль добавить алгоритм распознавания моргания по глазам с видео.

С помощью компонента Camera и библиотеки `clmtrackr` можно получить изображение глаз, а также их маску с точками зрачка и век. Имея такие данные, будем считать глаз закрытым, если кожей покрыта большая часть области, проходящая от верхнего века до нижнего через зрачок. Для более точной идентификации предварительно обрабатываем часть изображения, сконвертировав цветовое пространство в YCbCr и изъяв из него затемнённые диапазоны значений пикселей. На рис. 3 показана область правого глаза по точкам 24-27-26 при открытом и закрытом состоянии. На рисунке показано, что в первом случае большая часть области покрыта зрачком, а во второй лишь небольшая часть задета ресницей. Исходя из этого, будем считать глаз закрытым при заполнении светлым пространством 60% анализируемых пикселей.

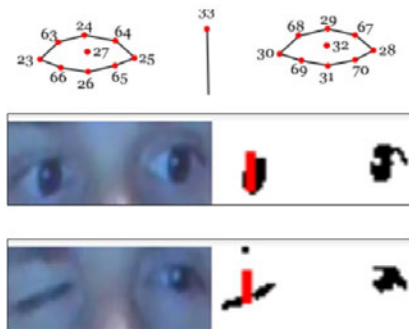


Рис. 3. Маска глаз и области зрачка при открытом и закрытом правом глазе

Добавив 2 асинхронных метода с данным алгоритмом в класс `EyeTrack`, помимо позиции взгляда, можно получать актуальные данные о положениях правого и левого глаза.

4. Обработка действий на странице

Зная координаты точки просмотра пользователя и действия при глазах, необходимо добавить обработку стандартных html элементов на странице сайта. Используя данную информацию от класса `EyeTrack`, при помощи директив фреймворка `Angular` можно добавить обработку наведения точки взгляда и закрытие глаза на любой html тег. На рис. 4 можно увидеть диаграмму классов для такой обработки.

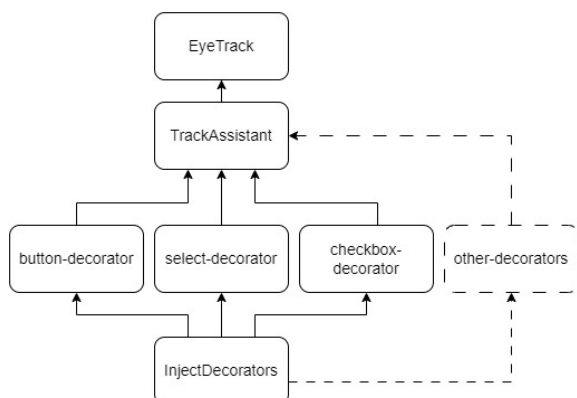


Рис. 4. Диаграмма классов для обработки html элементов

При инициализации страницы класс `InjectDecorators` обрабатывает все элементы на странице и внедряет в них свою директиву по назначению. Каждая директива подписывается на обновление позиции глаза через `TrackAssistant` и добавляет отрисовку дополнительных компонентов, таких как индикатор закрытия глаза. Сам же класс `TrackAssistant` фильтрует предсказанные значения взгляда по `EyeTrack`, а также, используя отсортированный массив расположенных на сайте элементов, доводит точку взгляда до ближайшего элемента по заданному диапазону. На рис. 5 можно увидеть, как к кнопке добавилась обработка наведения и закрытия глаза.

Создав директивы для обработки большинства стандартных элементов html, данную реализацию можно внедрить в любой сайт посредством расширения для браузера. На примере Chrome extension [6] необходимо добавить вывод страницы обучения Train-app по нажатию кнопки расширения, а также внедрить скрипт `InjectDecorators` при открытии любой страницы.

Пример внедрения скрипта для любого URL адреса приведен в листинге ниже.

Листинг

Параметр внедрения скрипта в манифесте расширения для Chrome

```

"content_scripts": [
  {
    "matches": ["<all_urls>"],
    "js": ["/extension/injectDecorators.js"]
  }
]
  
```

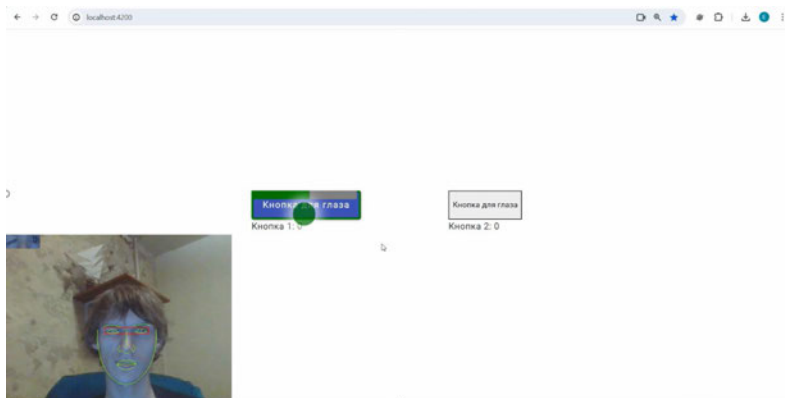



Рис. 5. Наведение точки взгляда на кнопку с директивой

Заключение

Данная статья посвящена разработке и реализации прототипа приложения для анализа взгляда пользователя. Была описана структура модуля, модели обучения и алгоритмы, используемые для решения поставленной задачи. В результате данной разработки был создан прототип в виде расширения Chrome, позволяющий пользователю создать и использовать модель отслеживания взгляда в качестве управления стандартными элементами страницы.

Список литературы

1. Описание Eye-tracking [Электронный ресурс]. – Режим доступа: <https://pixelplus.ru/samostoyatelno/stati/marketing/tekhnologiya-eye-tracking.html>
2. Отслеживание направления взгляда [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/companies/ruvds/articles/426055/>
3. Свёрточные нейронные сети для распознавания образов [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/articles/456186/>
4. Документация Angular [Электронный ресурс]. – Режим доступа: <https://angular.io/api/core/>
5. Библиотека Clmtrackr [Электронный ресурс]. – Режим доступа: <https://github.com/auduno/clmtrackr>
6. Документация Chrome extension [Электронный ресурс]. – Режим доступа: <https://developer.chrome.com/docs/extensions>

Лингвистический анализ названий групп студентов ВГУ с целью выявления различий в текстовых корпусах между разными факультетами

Э.В. Обухова

Студент магистр

А.В. Швырева

Ассистент

Введение

Социальные сети стали неотъемлемой частью жизни современного человека, они хранят множество различной пользовательской информации, которая называется цифровым следом. Разнообразие цифровых следов позволяет составить «цифровой портрет личности», включающий предпочтения и интересы пользователя. Данная информация активно используется рекомендательными системами интернет магазинов, музыкальными и стриминговыми платформами.

В качестве цифрового следа можно рассматривать профиль пользователя в социальной сети Вконтакте, так как он содержит информацию об интересах пользователя, выраженную в подписках на тематические сообщества. Данный цифровой след может помочь в получении объективной оценки предрасположенностей пользователя на основе его интересов, для помощи в выборе факультета для обучения.

В работах [1, 2] рассматривается задача сбора открытых данных пользователей сети Вконтакте, которые в качестве места обучения выбрали Воронежский государственный университет, с целью решения задачи профориентации на основе подписок пользователя.

В данной работе ставится задача лингвистического анализа названий сообществ, на примере студентов факультета компьютерных наук Воронежского государственного университета. Результаты лингвистического анализа необходимы для исследования цифрового портрета типичного представителя факультета для дальнейшего решения задачи профориентации с использованием цифровых портретов личности.

1. Инструменты для анализа текстов

В качестве инструментов анализа текстов был выбран язык Python 3, и нижеперечисленные библиотеки:

- NLTK – пакет программ для символьной и статистической обработки естественного языка, содержит графические представления и примеры данных, сопровождается обширной документацией [3];

- Natasha – библиотека решает основные задачи нейролингвистического программирования для русского языка: токенизация, сегментация предложений, встраивание слов, тегирование морфологии, лемматизация, нормализация фраз, синтаксический анализ, тегирование NER, извлечение фактов [4];

- SpaCy – аналог NLTK, который содержит удобные инструменты для поиска N-грамм [5] ;

- WordCloud – библиотека для генерации облака слов [6] ;

- Pandas – библиотека Python для обработки и анализа структурированных данных;

- Matplotlib, Seaborn – библиотеки для графической визуализации данных, построении зависимостей.

2. Описание анализируемого набора данных

В табл. 1 представлена структура набора данных, используемых для проведения лингвистического анализа названий групп студентов.

Набор состоит из 675465 уникальных записей, каждая из которых содержит уникальный идентификатор группы (ID), уникальный идентификатор факультета (FACULTY_ID), обозначение которых представлено в табл. 2, и наименование группы (GROUP_NAME). Полученный набор названий сообществ в сумме образует корпус из 2093975 слов.

Таблица 1

*Структура набора данных для эксперимента со случайно
выбранными примерами данных*

id	faculty_id	group_name
234	2	Интересные факты и тайны
224	2	Сам себе дизайнер
4	0	СНО истфака ВГУ
9	0	Поиск мемов по описанию
20	0	Книги – скачать
334	3	Донжон – магазин товаров Средневековья и фэнтези

id	faculty_id	group_name
341	3	Типичный программист
26651	7	Нетипичный Воронеж
26632	7	Наука и Факты
26690	7	Дизайн интерьера

Таблица 2

*Таблица соответствия уникального идентификатора
факультета названию факультета ВГУ*

id	Название факультета
0	Исторический факультет
1	Юридический факультет
2	Факультет романо-германской филологии
3	Факультет прикладной математики, информатики и механики
4	Факультет философии и психологии
5	Экономический факультет
6	Математический факультет
7	Факультет компьютерных наук
8	Химический факультет
9	Физический факультет
10	Фармацевтический факультет
11	Филологический факультет
12	Факультет журналистики
13	Медико-биологический факультет
14	Факультет международных отношений
15	Факультет географии, геоэкологии и туризма
16	Геологический факультет

На рис. 1 представлена столбчатая диаграмма, отражающая соответствие между идентификатором факультета и количеством групп, на которые подписаны пользователи, обучающиеся на выбранном факультете.

Рассмотрим обработку корпуса слов для сообществ обучающихся факультета компьютерных наук, аналогичная обработка может применять для любого факультета из представленного набора данных.

3. Общий анализ набора данных

Лингвистический анализ состоит из множества аспектов. Для решения поставленной задачи наиболее интересными являются:

- мера лексического разнообразия;

- средняя длина слов;
- ключевые слова;
- контексты для ключевых слов;
- биграммы;
- построение облака слов.

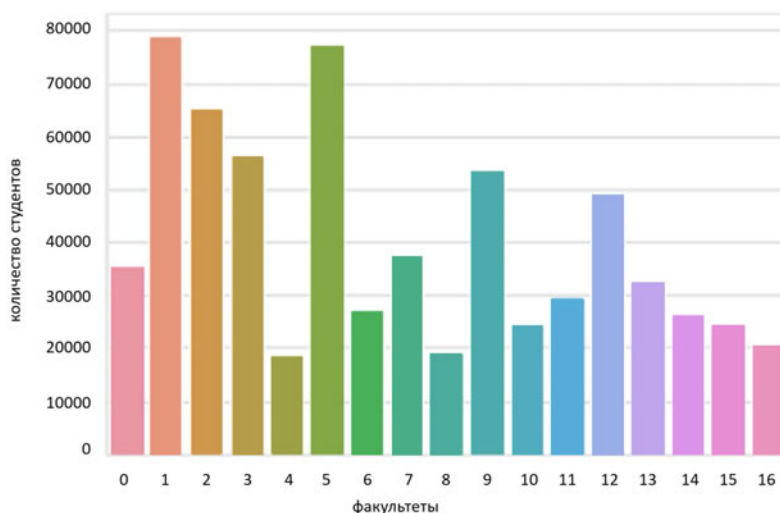


Рис. 1. Диаграмма зависимости количества групп и идентификатора факультета

Мера лексического разнообразия определяется с помощью коэффициента лексического разнообразия, отражающем степень богатства словаря при построении текста заданной длины, и вычисляемом как отношение уникальных лексем к общему количеству лексем в анализируемом тексте (1) [7].

$$L_d = \frac{N_{lex}}{N},$$

где:

L_d – коэффициент лексического разнообразия,

N_{lex} – количество уникальных лексем в анализируемом тексте,

N – количество текстоформ (общее количество словоформ) в анализируемом тексте.

5. Вычисление статистических характеристик полученного набора данных (табл. 3).

Таблица 3

*Статистические характеристики корпуса названий групп для
ФКН*

Название метрики	Значение метрики
Количество групп	37306
Количество слов	109619
Количество слов после удаления стоп-слов	84373
Количество уникальных слов (подсчитывалось после удаления стоп-слов)	24428
Среднее число слов в названии группы	2.74
Коэффициент лексического разнообразия	0,2895
Средняя длина слова	6.40

Анализируя полученные статистические характеристики (табл. 3) в соответствии с полученным значением коэффициента лексического разнообразия можно сказать, что корпус слов характеризуется общностью названий, в результате чего можно сделать вывод, что у обучающихся на одном факультете преобладают одинаковые названия. Если бы данный коэффициент превышал 0,5, то можно было бы судить о различии в профилях студентов одного факультета.

6. Определение ключевых слов полученного набора данных. Под ключевыми словами подразумеваются слова, которые наилучшим образом описывают весь набор данных. В настоящее время не существует общепринятого единственного алгоритма определения ключевых слов, поэтому рассмотрим выбор ключевых слов на основе анализа частотности и по длине слова.

4. Выбор ключевых слов на основе анализа частотности

В результате проведения анализа частотности слов были определены наиболее часто встречаемые слова, а также их контекст. В табл. 4 представлены 10 самых частых слов из набора данных. Анализ контекста вокруг ключевых слов позволяет понять основные темы текста и найти дополнительные скрытые темы.

Наиболее часто встречаемые слова и их контексты в названиях групп обучающихся факультета компьютерных наук

Ключевое слово	Контекст
воронеж	умереть ташир пицца воронеж доставка voronezh п цветочный мастерский воронеж это лето учебный пл журнал даунтаун любви воронеж krahzone btsmusic 방 фач гифки нетипичный воронеж mdk сила орлёнок к трансферы фк факел воронеж пацаны пиво art цита
вгу	тет компьютерный наука вгу black book девочка кос logy научный мем школа вгу идеальный хозяйшка ный программист проект вгу подслушать проект вгу вгу подслушать проект вгу чистосердечный best vi тет компьютерный наука вгу фонд борьба коррупция
наука	журнал чёткие прикол наука факт smart money бизн акультет компьютерный наука вгу find money daesan акультет компьютерный наука вгу english watch под культет компьютерный наука вгу романтика воронеж акультет компьютерный наука вгу вгуит бог Рядом †
мем	games memes пежа мпи мем игра экстремальный сто муары ценитель научный мем почему знать настоящ мореска юмор шрёдингер мем кот киберпанк сиим ф какой информатические мем hd фильм онлайн hdre наука вгу альтернатива мем человек отклонение for
факультет	а спорт психология факультет компьютерный наука video delivery club факультет компьютерный наука мёртвый компьютер факультет компьютерный наука вгу хороший мысль факультет компьютерный наука шутейка техноблог факультет компьютерный наука
Компьютерный	хология факультет компьютерный наука вгу black b а ростов факультет компьютерный наука вгу english groh art факультет компьютерный наука вгу романт пьютер факультет компьютерный наука вгу вгуит мысль факультет компьютерный наука вгу мастерск
программист	тегория б типичный программист anata faggot desu рование библиотека программист вгу факультет ком нетипичный воронеж программист бизнес предприн экран ноутбука iff программист воронеж волонтеры vel itc сообщество программист Грубо простить фак

Ключевое слово	Контекст
новость	комсомольский правда новость воронежа workout бы яндекс плюс filmpro новость кино kotex россия 10 pubg mobile кликклак новость gta 5 online gta 6 а комсомольский правда новость воронежа morgenshte вича pokemon go факт новость дешёвый товар после
музыка	мный чат вконтакте vk музыка голосовой текст бот s fan video world tanks музыка машина оружие нож воронеж москвич tvg музыка игра подслушать мбоу нтакте егэ математика музыка фильм ukula.ru аккор epsi lyrsense хороший музыка dog life пещера гейме

В табл. 5 представлены наиболее популярные биграммы для корпуса слов. Таким образом, можно сделать вывод, что обучающиеся факультета компьютерных наук заинтересованы в новостях факультета, вуза и города, а также в программировании, компьютерных играх, иностранных языках и развлекательном контенте.

Таблица 5

Популярные биграммы в названиях групп обучающихся факультета компьютерных наук

№	Биграмма
1	'факультет', 'компьютерный'
2	'компьютерный', 'наука'
3	'наука', 'вгу'
4	'стена', 'плача'
5	'проект', 'вгу'
6	'cs', 'go'
7	'dota', '2'
8	'вгу', 'подслушать'
9	'новость', 'воронежа'
10	'каждый', 'день'
11	'словарный', 'запас'
12	'английский', 'язык'
13	'dsr', 'corporation'
14	'научный', 'мем'
15	'хороший', 'фильм'

5. Выбор ключевых слов на основе анализа длины

В результате проведения анализа длины слов исследуемого корпуса слов была получена гистограмма количества слов в зависимости от длины слова, представленная на рис. 2. По оси абсцисс отложены длины слов в корпусе, по оси ординат отложено количество слов.

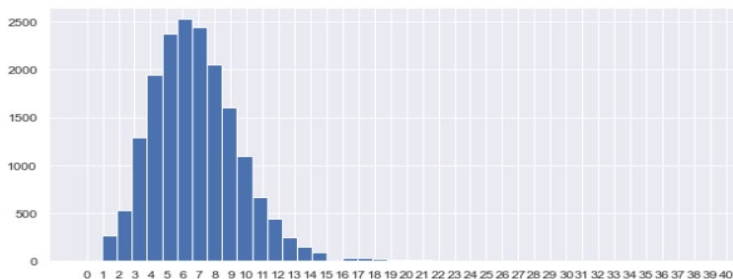


Рис. 2. Гистограмма распределения длины слов в корпусе

Наличие слов длиной более 25 символов указывает на то, что некоторые слова представляют собой комбинацию из нескольких слов, разделенных специальными символами. Для обработки обнаруженных ситуаций и разделения слов пробелами использовалось регулярное выражение. Окончания полученных слов убирались с помощью стемминга.

В результате проведения обработки гистограмма распределения длины слов преобразовалась к виду, представленному на рис. 3.

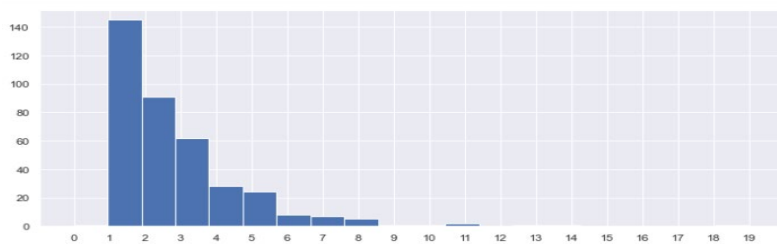


Рис. 3. Гистограмма распределения длины слов в корпусе после стемминга и очистки регулярными выражениями

Предположим, что ключевыми будут являться слова превышающие 95-й квантиль распределения длины слов, то есть длиной 5 и более букв. Фрагмент выделенных слов представлены в листинге 2.

Листинг ключевых слов длиной более 5 букв

'_九代目火影', '°◡', 'dōath', 'Ἱραγ Σ', 'BRONY④BRONY',
 'BUTTER', 'HEZUKI', 'tētīç', 'πτύαλον', 'p□□u(Ŧu⁴e(□u', ع
 'مرحبا', 'بريفا', 'العاهرة', 'الصور', 'ღღღ²⁰¹²', 'DREAM',
 'PORTO', 'WORKET', 'ORIGINAL', '●••●', 'fanfiction',
 '「日本語」', 'から永遠へ', 'ただ素敵な女性', 'アニメと音楽',
 'アニメ怒っスキル', 'カウンターストライク', 'サムライパラダイス',
 'ミラベリカ', 'メイドカフェ', 'ヤクいさん', 'スノーロニン',
 '俄罗斯灰摩卡', '日本ネット', '红龙习近平', '視覚的に美しい',
 '드라마여왕', '디엠지에이', '몬스타엑스', '방탄소년단', '팬사이트의'

Среди обнаруженных ключевых слов встречаются названия популярных аниме, корейских музыкальных групп и игр. Ключевые слова на латинице или кириллице представлены в листинге 3.

*Листинг ключевых слов длиной более 5 букв после обработки
регулярными выражениями и стеммингом*

академическ, нововоронеж, средневеков, streambonus,
 деградирован, сверхинтеллектуа, стратегическ, систематизирова,
 антигосударственник, documentary, управленческ,
 стопкоронавирус, магистратур, underground, субкультурн

В результате сравнения ключевых слов полученных с помощью анализа частотности или длины слов можно сделать вывод, что наиболее частые слова показали общие профессиональные интересы, объединяющие обучающихся, а наиболее длинные слова показали общие субкультурные интересы. То, что между различными множествами ключевых слов есть пересечения в виде общих тем, показывает, что оба метода показывают релевантную картину.

6. Построение облака слов

На рис. 4 представлено облако слов для групп обучающихся факультета компьютерных наук. Облако слов или облако тегов – это визуальное представление текстовых данных, обычно используемое для демонстрации ключевых терминов для визуализации содержания текста. Слова или фразы при этом размещаются в произвольном порядке, а их размер обычно соответствует частоте встречаемости в тексте. Чем чаще определенное слово встречается в тексте, тем больше оно будет в облаке слов.

решения проблемы профориентации / Э. В. Обухова, А. В. Швырева // Информатика : проблемы, методы, технологии: сборник материалов XXIII международной научно-практической конференции (Воронеж, 15-17 февраля 2023 г.). – Воронеж, 2023. – С. 619-624.

2. Обухова, Э. В. Использование цифровых следов в интересах решения задачи профориентации / Э. В. Обухова, А. В. Швырева // Труды молодых ученых факультета компьютерных наук ВГУ. – Воронеж, 2023. – С. 150-157.

3. Библиотека NLTK [Электронный ресурс] : документация библиотеки NLTK. – Режим доступа: <https://www.nltk.org>

4. Natasha [Электронный ресурс] : документация библиотеки natasha. – Режим доступа: <https://github.com/natasha/natasha>

5. Краткий обзор NLP библиотеки SpaCy [Электронный ресурс] : Блог Семёна Погорельцева – Режим доступа: <https://habr.com/ru/articles/504680/>

6. Wordclouds in Python [Электронный ресурс] : Блог Dan Keefe. – Режим доступа: <https://dev.to/perittract/wordclouds-in-python-562e>

7. Захарова, Е.Ю. Лексическое разнообразие текста и его измерения / Е. Ю. Захарова, О. Ю Савина // Вестник Тюменского государственного университета. Гуманитарные исследования. Humanitates. – Тюмень, 2020. – Том 6. №1 (21). – С. 20-34.

8. Биграммы и триграммы. Кейс сбора и анализа информации из аудио с непонятными словами [Электронный ресурс] : блог компании Инфосистемы Джет. – Режим доступа: <https://habr.com/ru/companies/jetinfosystems/articles/707664/>

9. Облака слов из ВК на колоне [Электронный ресурс] : блог участника Хабр Larymar. – Режим доступа: <https://habr.com/ru/articles/458524/>

Моделирование распространения акустических сигналов в подводных каналах связи

Д.А. Пархоменко

Студент магистр

А.Ю. Малыхин

Старший преподаватель

Введение

Влияние неоднородностей подводной среды на распространение звуковых сигналов в подводных каналах связи играет важную роль в обеспечении эффективной связи в морских условиях. Передача акустических сигналов имеет большое значение для обеспечения связи и выполнения функций, таких как контроль и мониторинг подводного оборудования, обмен данными и коммуникации для подводных исследований.

Различные методы и технологии применяются для контроля воздействия неоднородностей подводной среды на передачу звуковых сигналов в подводных каналах связи.

Данная работа посвящена разработке и программной реализации моделирования распространения акустических сигналов в подводной среде.

В программе настраиваются параметры отправленного сигнала и характеристики среды, такие как глубина моря, длина трассы, скорость звука в воде.

1. Расчет звукового поля в океане

Модовый подход к расчёту звукового поля в океане основан на представлении звука как суперпозиции мод, или модов, которые представляют собой неоднородные распределения звукового давления в пространстве. Моды являются решениями волнового уравнения для конкретных частот и их соответствующих пространственных распределений [1-2]. Большая часть энергии звука обычно будет сосредоточена в модах с малыми номерами.

2. Диаграмма вариантов использования

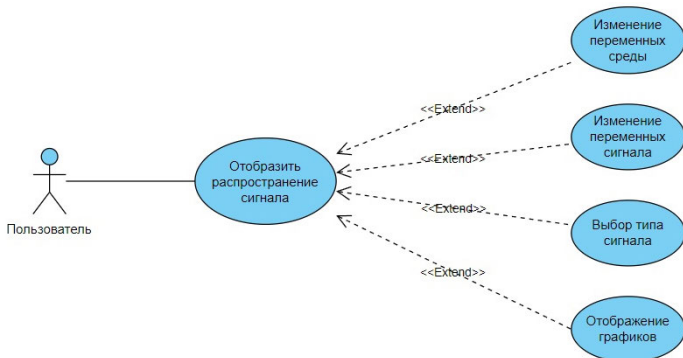


Рис. 1. Use-case диаграмма

Приложение дает пользователю возможность отобразить графики распространения акустического сигнала в среде (рис.1). Пользователь может настраивать различные параметры среды, через которую проходит сигнал и самого сигнала. Пользователь может выбрать тип сигнала, который будет исследовать. Также пользователь имеет возможность просмотреть в приложении графики, полученные в соответствии с заданными им параметрами.

3. Блок-схема программы

На рис. 2 приведена блок-схема программы. В программу вводятся следующие входные данные: H – глубина моря в метрах, l – номер моды, f – частота сигнала в Гц, и вводится некоторая случайная последовательность бит. Далее выполняется модуляция BPSK сигнала.

После этого для массива частот спектра сигнала производится расчет горизонтальных волновых чисел мод:

$$q_l = \sqrt{\frac{(2\pi f)^2}{c^2} - \frac{\pi^2}{H^2} \left(l - \frac{1}{2}\right)^2}, \quad (1)$$

где c – скорость звука в воде в м/с.

После этого выполняется прямое преобразование Фурье от цифрового сигнала. Для каждой спектральной составляющей происходит умножение на коэффициент, отвечающий за изменение фазы этой компоненты при распространении вдоль акустической трассы. Для полученного в результате преобразований массива значений спектра принятого сигнала проводим обратное преобразование Фурье. Далее

демодулируем полученный сигнал и получаем принятую последовательность чисел.

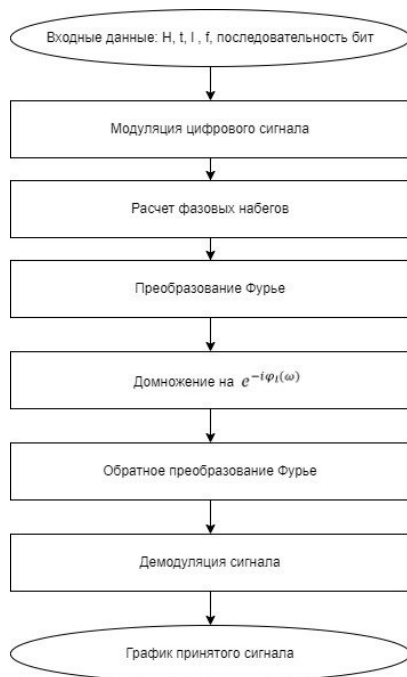


Рис. 2. Блок-схема программы

4. Моделирование распространения BPSK сигнала

Далее рассмотрим моделирование распространения BPSK сигнала. Сначала построим график исходной последовательности (рис. 3).

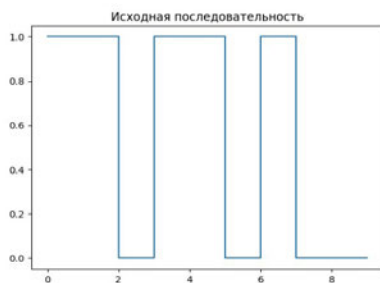


Рис. 3. График исходной последовательности

Далее строим график сигнала после модуляции (рис.4):

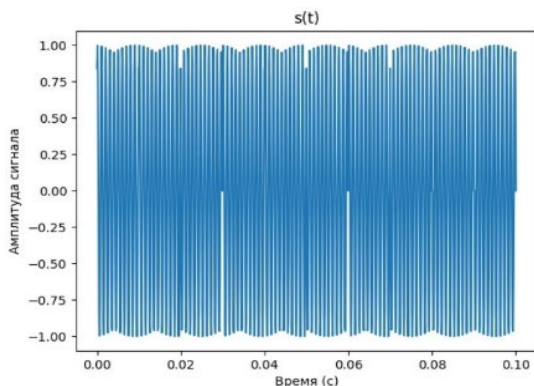


Рис. 4. График исходного BPSK сигнала

Произведя все необходимые вычисления (рис 2), отображаем график полученного сигнала (рис. 5).

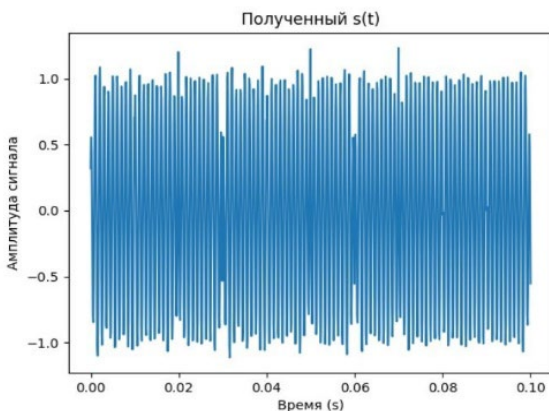


Рис. 5. График полученного сигнала

После демодуляции видим, что график полученной последовательности (рис.6) совпадает с графиком исходной, следовательно, при заданных параметрах точность передачи составляет 100%.

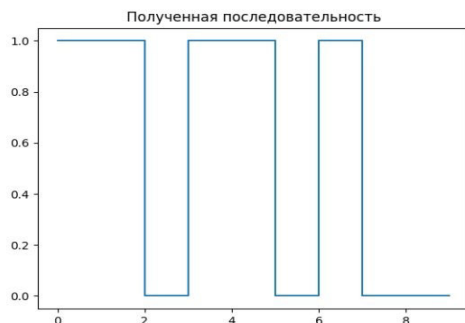


Рис. 6. График полученной последовательности

Заключение

Разработанное на языке Python приложение позволяет графически отображать распространение акустических сигналов в подводной среде. Оно учитывает параметры отправленного сигнала и характеристики среды. Также удалось подтвердить корректность результатов работы программы.

Список литературы

1. Computational ocean acoustics / F.B. Jensen, M.B. Porter, W.A. Kuperman, H. Schmidt. Computational ocean acoustics. New-York, Springer, 2011. – 794 p.
2. Katsnelson, B.G. Fundamentals of Shallow Water Acoustics / B.G. Katsnelson, V.G. Petnikov, J.F. Lynch. New-York.: Springer, 2012. – 556 p.

Исследование развития киберугроз веб-приложений

Д. С. Перепёлкин

Студент магистр

А. А. Вахтин

Доцент

Введение

Современные веб-приложения прочно вошли в нашу жизнь, став неотъемлемой частью цифровой инфраструктуры. Они предлагают широкий спектр функций, от социальных сетей и электронной коммерции до онлайн-банкинга и государственных услуг. Однако, параллельно с расширением возможностей веб-приложений, наблюдается и увеличение угроз, связанных с их эксплуатацией.

Исследование киберугроз, направленных на веб-приложения, является актуальной задачей в контексте постоянно изменяющейся кибербезопасной среды. Угрозы могут проявляться в различных формах, включая атаки на аутентификацию, инъекции кода, кросс-сайтовый скриптинг, уязвимости в сессиях и многие другие.

В этой статье мы рассмотрим статистику кибератак на веб-приложения, проанализируем их и предложим методы защиты, позволяющие обеспечить безопасность веб-ресурсов. Для достижения данной цели будет проведён обзор, включающий в себя отчеты и публикации в области кибербезопасности, а также последствия недавних инцидентов, связанных с нарушениями безопасности веб-приложений.

1. Ключевые цифры и тренды киберугроз

Активное развитие цифровой среды сопровождается увеличением числа кибератак, направленных на веб-ресурсы организаций. В период с 2021 по 2022 год наблюдалось заметное увеличение успешных атак на веб-платформы, что отражается в статистических данных. В 2021 году веб-ресурсы компаний становились объектами атак лишь в 17% случаев, в то время как в 2022 году данная доля увеличилась до 22%. Этот тренд оказал наибольшее воздействие на государственные учреждения, где количество инцидентов увеличилось более чем в два раза, а доля атак увеличилась с 23% до 41%.[1]

Эскалация конфликтов в киберпространстве в 2022 году привела к резкому увеличению числа атак, сфокусированных на сайты организаций.[2] Эти инциденты имели заметные последствия, в основном затрагивая основные бизнес-процессы, особенно если компания предоставляет свои услуги онлайн. По данным статистики, инциденты с веб-ресурсами привели к нарушениям деятельности организаций в 53% случаев. Основные методы атак включали в себя блокировку доступа к сайту или его дефейс. Компрометация веб-ресурсов также открывала двери для массовых атак на пользователей, что привело к всплеску активности в сборе информации о них в III квартале.

На рис. 1 представлена диаграмма процентного соотношения инцидентов в 2021 и 2022 годах. Секторы масс-медиа, транспортной инфраструктуры, государственных учреждений и ритейловые компании, в которых всегда проявлялся повышенный интерес со стороны киберпреступников из-за большого объема обрабатываемых данных, продолжали оставаться целью атак.

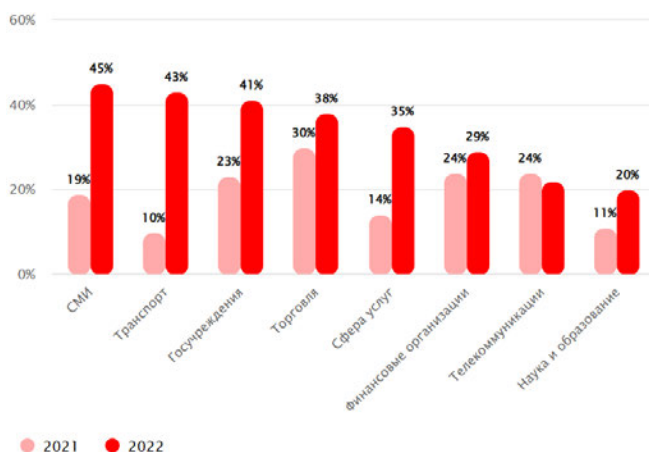


Рис. 1. Доля инцидентов, связанных с атаками на веб-ресурсы в 2021 и 2022 годах

Злоумышленники активно использовали уязвимости веб-приложений, такие как CVE-2021-44228 (Log4Shell в Apache Log4j 2) и CVE-2022-22965 (Spring4Shell в Java Spring Framework), а также CVE-2022-24086 (в Adobe Commerce) и CVE-2021-32648 (в October CMS), для перехвата личных данных и информации о платежных картах.[3-4]

Рост числа атак на веб-ресурсы также стимулировался обнаружением уязвимостей в популярных веб-приложениях, включая такие как WordPress и Magento. Помимо этого, расширение теневого рынка предоставило злоумышленникам новые инструменты для проведения атак, включая DDoS-атаки. Предвидится, что в 2023 году число атак на веб-ресурсы организаций будет продолжать расти, особенно в отношении компаний, предоставляющих онлайн-сервисы и собирающих большие объемы данных о клиентах.

На рис. 2 можно заметить, что в 2021 году наблюдался незначительный прирост числа кибератак всего на 6,5% по сравнению с предыдущим годом. В 2022 году можно отметить более значительный рост общего количества инцидентов – на 20,8%. [4]

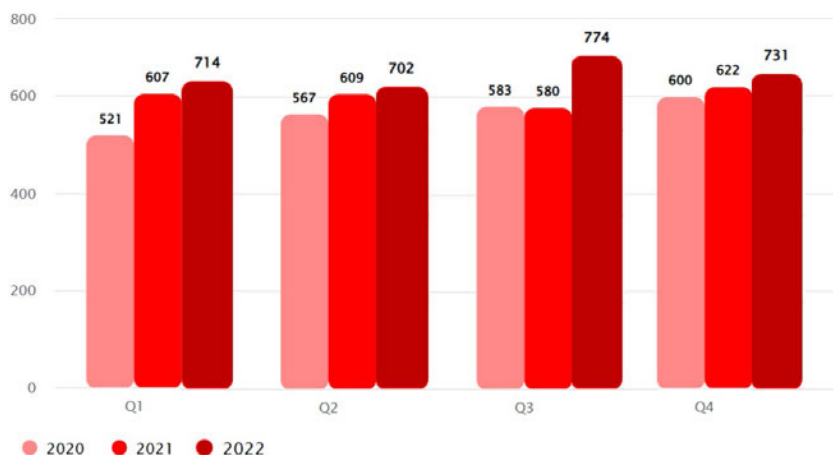


Рис. 2. Сравнение количества инцидентов в 2020-2022 годы(по кварталам)

Особое влияние на увеличение числа атак оказывает утечка персональных данных, предоставляющая киберпреступникам дополнительные возможности для проведения специализированных и высокоэффективных атак. Предполагается, что и в 2023 году можно будет наблюдать увеличение числа кибератак, основываясь на тех же тенденциях и факторах, которые способствовали росту атак в предыдущие годы. Это означает, что организации должны постоянно улучшать свои стратегии безопасности и совершенствовать защиту

своих информационных ресурсов, чтобы минимизировать риски киберугроз и обеспечить надежную защиту своих данных и активов.

2. Результаты массовых утечек

Распространение массовых утечек данных в 2022 году стало явлением, затрагивающим широкий круг организаций и частных лиц как в России, так и за ее пределами. Среди пострадавших выделяются такие известные компании и сервисы, как «Гемотест», «СДЭК», Яндекс.Еда, Delivery Club, DNS. В глобальной перспективе ущерб от утечек продолжает расти: согласно отчету IBM, средняя стоимость утечки данных в 2022 году достигла рекордного уровня в 4,35 млн. долларов.[5]

Злоумышленники успешно компрометируют всё больше конфиденциальной информации. На рис. 3 представлено соотношение типов украденных данных за 2021 и 2022 года. Наблюдается увеличение доли персональных данных среди украденной информации (с 32% до 36%), 41% скомпрометированных данных составили учетные данные, 28% – персональные данные, и 15% – данные платежных карт.

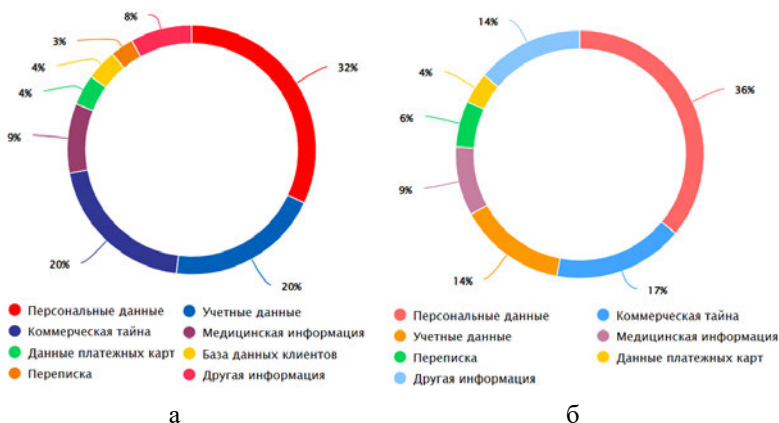


Рис. 3. Соотношение типов украденных данных: а – за 2021 год, б – за 2022 год

Архивы с украденными данными, как правило, становятся предметом продажи на теневых форумах. Этот тренд предполагает возможность формирования цифровых портретов потенциальных жертв и более изощренные атаки, включая применение методов социальной инженерии.

3. Меры безопасности

На основе приведённых данных можно сделать вывод о необходимости усиления мер безопасности как для организаций, так и для частных лиц. Украденные данные становятся объектом торговли на теневых форумах и могут быть использованы для формирования цифровых портретов, что повышает риск различных киберугроз. В связи с этим крайне важно принять эффективные меры для защиты конфиденциальной информации.

Организации и частные лица должны обеспечивать надежную защиту своих данных с помощью передовых технологий шифрования. Шифрование данных позволяет защитить информацию даже в случае её утечки, делая её непригодной для использования злоумышленниками.

Многофакторная аутентификация (МФА) должна стать стандартом для всех пользователей. Она значительно усложняет несанкционированный доступ к учетным записям, требуя подтверждения личности через несколько независимых каналов. Это повышает уровень безопасности, снижая риск компрометации учетных данных.

Важным шагом является регулярное обновление программного обеспечения и применение патчей. Обновления и патчи устраняют уязвимости, которые могут быть использованы злоумышленниками для проведения атак. Организации должны внедрить процессы управления обновлениями, чтобы оперативно закрывать обнаруженные уязвимости.

Кроме того, регулярный мониторинг активности в сети и на веб-ресурсах является важным компонентом информационной безопасности. Этот процесс включает в себя отслеживание подозрительной активности и аномалий, которые могут свидетельствовать о попытках вторжения или компрометации системы. Своевременное обнаружение таких признаков позволяет принять необходимые меры для предотвращения или минимизации последствий атак.[6]

Также необходимо регулярно проводить аудиты безопасности. Аудиты помогают выявить уязвимости и недостатки в существующих системах и процессах защиты данных. На основании результатов аудитов можно разрабатывать и внедрять корректирующие меры, направленные на устранение выявленных проблем и повышение общего уровня информационной безопасности.

Принятие вышеуказанных мер способствует постепенному снижению риска утечек данных и минимизации возможных последствий атак, основанных на украденной информации. Это позволяет защитить

как организационные, так и личные данные от киберугроз, обеспечивая более безопасное использование цифровых технологий.

Заключение

В современном мире киберугрозы, направленные на веб-приложения, постоянно эволюционируют и становятся все более сложными. Злоумышленники применяют всё более утончённые методы атак, что требует от организаций комплексного подхода к обеспечению безопасности веб-ресурсов.

Для обеспечения надлежащего уровня защиты необходимо внедрение современных методов кибербезопасности, регулярное обновление программного обеспечения, своевременное применение патчей безопасности, а также постоянное повышение осведомлённости и компетенций сотрудников в области кибербезопасности.

Только комплексный и систематический подход к защите веб-приложений способен эффективно минимизировать риски кибератак и обеспечить надёжную защиту данных пользователей.

Список литературы

1. Атаки на веб-приложения в 2022 году: анализ действий злоумышленников.[Электронный ресурс]. – Режим доступа: <https://www.anti-malware.ru/>
2. Актуальные киберугрозы: итоги 2021 и 2022 годов [Электронный ресурс]. – Режим доступа: <https://www.ptsecurity.com/>
3. Уязвимости и угрозы веб-приложений в 2021 и 2022 годах.[Электронный ресурс]. – Режим доступа : <https://www.ptsecurity.com/>
4. National Vulnerability Database [Электронный ресурс]. – Режим доступа : <https://nvd.nist.gov/>
5. IBM. IBM Annual Report 2022 [Электронный ресурс]. – Режим доступа : <https://www.ibm.com/annualreport/>
6. Хоффман, Э. Безопасность веб-приложений / Э. Хоффман. – Санкт-Петербург.: Питер, 2021. – 336 с.

Методы вычисления показателей Ляпунова гладких динамических систем

К. О. Петрищев

Студент магистр

С. В. Борзунов

Доцент

Введение

Во многих практически важных задачах в области механики, химии, материаловедения, оптики, биологии, экономики и т. д. математическая модель формулируется в виде некоторой динамической системы. Как известно, состояние динамической системы в любой момент времени однозначно определяется ее начальным состоянием. Формальной основой для задания динамической системы может выступать не только набор дифференциальных уравнений, но и, например, дискретные отображения, интегро-дифференциальные уравнения, уравнения с запаздыванием, марковские цепи, процессы на графах и т. д.

Эволюция таких систем во времени определяется правилами, согласно которым динамические переменные, характеризующие состояние системы, вычисляются в любой последующий момент времени на основе их исходных значений. Другими словами, каждая система такого рода характеризуется своим начальным состоянием и законом, по которому система переходит из начального состояния в последующие. Наиболее важны для современных приложений нелинейные системы, задаваемые нелинейными уравнениями. Результаты многочисленных исследований показывают, что для различных нелинейных динамических систем их эволюция может демонстрировать значительные качественные отличия. В частности, динамические переменные могут зависеть от времени периодически образом, и наблюдается многократное возвращение в исходное состояние через строго определенные промежутки времени. В некоторых случаях возникает квазипериодический режим, соответствующие функциональные зависимости – квазипериодические. Но наиболее сложным для теоретического и численного анализа

является хаотический режим динамики, при котором любое малое возмущение начального состояния системы приводит к значительному изменению траектории по сравнению с невозмущенным случаем.

На достаточно больших интервалах времени исходная и возмущенная траектории в фазовом пространстве расходятся, что приводит к нерегулярным, хаотическим изменениям динамических переменных во времени. Интересно, что одна и та же система в различных диапазонах значений параметров может демонстрировать как периодическое или квазипериодическое поведение, так и хаотический режим [1].

Методы идентификации режимов динамических систем представляют собой важнейшее направление современных исследований в области нелинейной динамики. Известны такие методы, как спектральный анализ, тест 0-1 и другие. Метод, основанный на вычислении спектра показателей Ляпунова, является одним из самых известных и часто используемых [2]. По сравнению, например, с анализом спектра Фурье, показатели Ляпунова являются мерой количественной характеристики хаоса, что позволяет делать выводы о степени выраженности хаоса при тех или иных параметрах задачи.

Основные задачи исследований в этой области включают в себя разработку алгоритмов для вычисления показателей Ляпунова, адаптацию этих алгоритмов для работы, оптимизацию алгоритмов с целью увеличения производительности, а также исследование применимости этих методов для различных типов динамических систем.

1. Алгоритм Бенеттина

Алгоритм Бенеттина – это метод вычисления ляпуновских показателей динамических систем [3]. Этот метод является одним из первых и наиболее широко используемых для оценки спектра ляпуновских экспонент.

Пусть динамическая система описывается дифференциальным уравнением:

$$\frac{dx}{dt} = F(x),$$

где $x \in \mathbb{R}^n$, $F : \mathbb{R}^n \rightarrow \mathbb{R}^n$. Рассмотрим траекторию в конфигурационном пространстве, начинающуюся в точке x_0 :

$$\dot{\phi}_t(x_0) = F(\phi_t(x_0)), \phi_t(x_0) = x_0,$$

Матрица Якоби $J(\mathbf{x}_0)$ характеризует изменение траектории в результате бесконечно малого возмущения начальных условий $\delta\mathbf{x}_0$:

$$\delta\mathbf{x}(t) = \Phi(\mathbf{x}_0 + \delta\mathbf{x}_0) - \Phi(\mathbf{x}_0) = J(\mathbf{x}_0)\delta(\mathbf{x}_0)$$

Обозначим собственные векторы матрицы $J^T(\mathbf{x}_0)J(\mathbf{x}_0)$ через m_i , где $i = 1, \dots, n$. Спектр Ляпунова определяется как упорядоченный набор $(\lambda_1, \lambda_2, \dots, \lambda_n)$ компоненты которого равны

$$\lambda_i = \lim_{t \rightarrow \infty} \frac{1}{t} \ln \sqrt{m_i(t)}.$$

Работа алгоритма Бенеттина начинается с численного решения динамических уравнений на интервале времени, достаточном для установления траектории на аттракторе системы. Для получения статистически достоверной оценки ляпуновских показателей, а также оценки погрешности, можно повторить процедуру несколько раз с разными начальными условиями и провести стандартные процедуры статистической обработки результатов. Алгоритм Бенеттина является мощным инструментом для оценки ляпуновских экспонент в динамических системах и широко используется в анализе сложных динамических процессов.

2. Программная реализация

На языке Python была реализована программа для вычисления показателей Ляпунова с помощью алгоритма Бенеттина. На рис. 1 показана блок-схема алгоритма. На этом рисунке представлен общий поток выполнения кода, начиная с инициализации начальных условий и параметров, далее – итерации вычисления ляпуновских показателей, и отображение результатов.

В листинге 1 представлен программный код, который реализует алгоритм Бенеттина для вычисления показателей Ляпунова системы дифференциальных уравнений.

Используются несколько ключевых функций из библиотек NumPy и SciPy. Рассмотрим их использование и описание процедур.

1. Функция `solve_ivp()` из библиотеки `scipy.integrate` [5]: эта функция применяется для численного решения системы обыкновенных дифференциальных уравнений с начальными условиями. В качестве входных данных рассматривается функция, определяющая систему уравнений, начальные условия, интервал времени для решения и другие параметры. Возвращает объект `scipy.integrate.OdeResult`, содержащий решение на заданном интервале времени.

2. Функция `np.linalg.qr()` из `numpy.linalg` [6]: QR-разложение матрицы. QR-разложение представляет матрицу в виде произведения ортогональной матрицы (Q) и верхнетреугольной матрицы (R). Возвращаемые значения – Q и R.

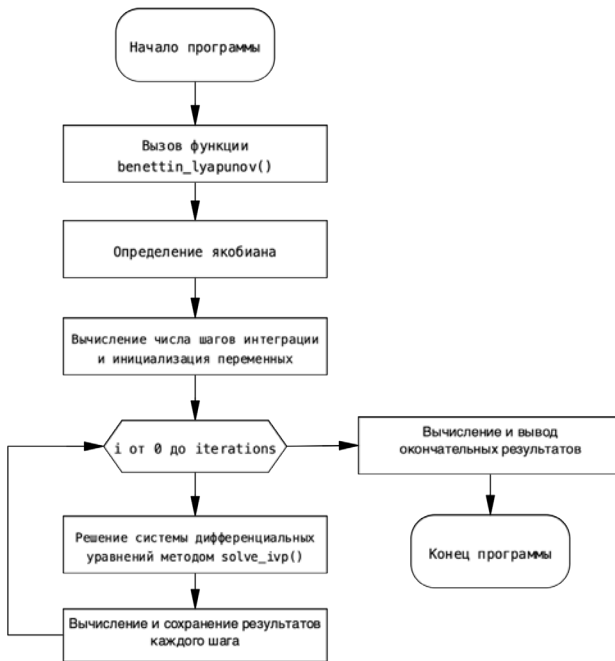


Рис. 1. Блок-схема алгоритма

Листинг

Реализация алгоритма Бенеттина на языке Python

```

def benettin_lyapunov(st_pos, dt, i, tr_stp):
    ic = len(st_pos)
    n_lyap = ic
    le = []
    lc = []

    pos = st_pos
    cur_bs = np.eye(ic)

    for iter_idx in range(i):

```

```

s = solve_ivp( integrate_l, [0, dt],
np.concatenate((pos, cur_bs.flatten())) , t_eval=[dt],
max_step=dt, args=(ic,))
pos, cur_bs = separate_state(s.y[:, -1])
cur_bs, R = np.linalg.qr(cur_bs)
le.append(np.log(abs(R.diagonal())) / dt)
if iter_idx >= tr_stp:
    lc.append(np.mean(le, axis=0))

vle_avg = np.average(le[tr_stp:], axis=0)
return vle_avg, lc

```

В качестве примера вычисления показателей Ляпунова гладкой динамической системы с помощью алгоритма Бенеттина рассмотрим систему Лоренца:

$$\begin{cases} \frac{dx}{dt} = -\sigma x + \sigma y, \\ \frac{dy}{dt} = rx - y - xz, \\ \frac{dz}{dt} = xy - bz, \end{cases} \quad (1)$$

с соответствующими начальными условиями. Здесь σ , r , b – неотрицательные числа. Уравнения (1) впервые были сформулированы для описания воздушных потоков в нижних слоях атмосферы, представленной в виде модели сплошной среды, равномерно нагреваемой снизу и охлаждаемой сверху.

При некоторых наборах параметров система Лоренца демонстрирует хаотическое поведение.

Приведем результаты численных расчетов:

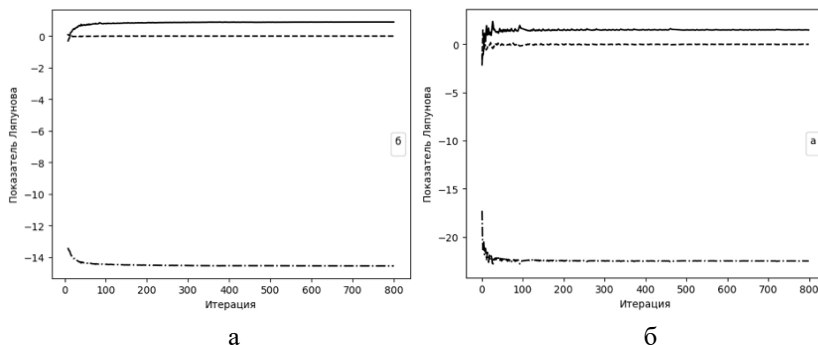
если $\sigma = 10$, $r = 28$, $b = 8/3$, то $(\lambda_1, \lambda_2, \lambda_3) = (0.898, 0.0126, -14.574)$;

если $\sigma = 16$, $r = 45,92$, $b = 4$, то $(\lambda_1, \lambda_2, \lambda_3) = (1.493, 0.0028, -22.488)$;

Наличие положительного старшего показателя в обоих случаях свидетельствует о хаотическом режиме системы Лоренца при указанных значениях параметров [1].

На рис. 2 представлены средние значения показателей Ляпунова в зависимости от количества итераций алгоритма Бенеттина. По оси Ox приведен номер итерации, а по оси Oy – значения ляпуновских показателей. Важное свойство сходимости алгоритма определяет, насколько быстро или медленно средние значения показателей Ляпунова

приближаются к предельным значениям. Из рис. 2 видно, что графики величин $\lambda_1, \lambda_2, \lambda_3$ быстро достигают своих горизонтальных асимптот.



$a - \sigma = 10$, $r = 28$, $b = 8/3$, $b - \sigma = 16$, $r = 45.92$, $b = 4$ (панель б)

Рис. 2. Графики средних значений показателей Ляпунова в зависимости от количества итераций алгоритма Бенеттина.
Значения параметров

Заключение

На языке Python реализован алгоритм Бенеттина вычисления показателей Ляпунова гладких динамических систем. Полученный программный код позволяет вычислять показатели Ляпунова на современных, в том числе, высокопроизводительных вычислительных системах. Программа может быть использована для анализа широкого класса различных динамических систем, представимых в виде систем дифференциальных уравнений.

Список литературы

1. Кузнецов, С. П. Динамический хаос : курс лекций / С. П. Кузнецов. – 2-е изд. – М. : Физматлит, 2006. – 355 с. – (Серия: Современная теория колебаний и волн).
2. Benettin G. Lyapunov characteristic exponents for smooth dynamical systems and for Hamiltonian systems; a method for computing all of them. Part 1: Theory / G. Benettin, L. Galgani, A. Giorgilli, J. M. Strelcyn // *Meccanica*. – 1980. – V. 15. – P. 9-20.
3. Semenov M. E. A new way to compute the Lyapunov characteristic exponents for non-smooth and discontinues dynamical systems / M. E. Semenov, S. V. Borzunov, P. A. Meleshenko // *Nonlinear Dynamics*. – Springer, 2022. – V. 109. – P. 1805-1821. – EDN XFYIDK.

Оптимизация поиска в реестре единого фонда геологической информации

А. В. Постников

Студент магистр

Я. А. Воронцов

Доцент

Введение

Единый фонд геологической информации о недрах (ЕФГИ) – это федеральная государственная информационная система, содержащая реестр первичной и интерпретированной геологической информации о недрах, имеющих в федеральном и его территориальных фондах геологической информации. Основными задачами ЕФГИ являются сбор, обработка, хранение, предоставление и распространение геологической информации.

Одним из главных модулей системы является реестр геологической информации, который предоставляет пользователю возможность поиска всей имеющейся геологической информации, установление местоположения геологической информации, а также возможность запросить подробные данные о реестровом объекте посредством создания заявки.

1. Особенности реестра ЕФГИ

Реестр единого геологического фонда имеет следующие особенности:

- Содержит более 2 млн. реестровых объектов;
- Поддерживает множественную фильтрацию;
- Поддерживает сортировку;
- Является публичным.

Из данных особенностей следует, что нагрузка на реестр достаточно большая. Пиковое число запросов составляет 1 тыс. в секунду, при этом поисковые запросы нетривиальные из-за наличия множественной фильтрации и сортировки. Классические реляционные базы, работая на таком объеме данных при использовании множественной фильтрации и сортировки не могут выдавать приемлемую для пользователя скорость поиска. Эта проблема относится

к реестру ЕФГИ, работающему на реляционной базе PostgreSQL [1]. Для оценки времени отклика системы, в реестре ЕФГИ был произведен замер скорости поиска при различных комбинациях фильтров и сортировки (рис. 1).

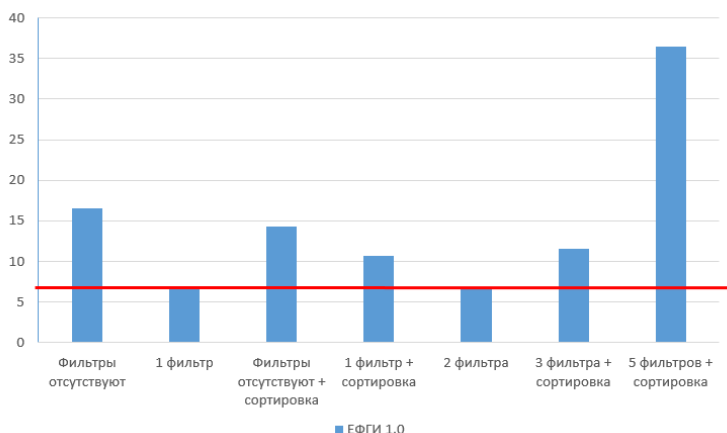


Рис. 1. Гистограмма времени поиска в реестре ЕФГИ 1.0

Из полученных результатов видно, что среднее время поиска в реестре без использования фильтров, превышает 16 секунд. А при установке пяти фильтров время ожидания результатов превышает 30 секунд.

Согласно исследованию Microsoft, проведенное в 2015 году, среднее время концентрации внимания человека составляет 8 секунд [2]. После этого времени человек начинает отвлекаться, переключать внимание на другие вещи, и это может негативно сказываться на репутации медленно работающего веб-ресурса. На гистограмме этот порог выделен красной линией. В большинстве случаев время поиска в реестре его превышает.

2. Постановка задачи

Целью работы являлось создание модуля реестра для новой версии системы ЕФГИ – ЕФГИ 2.0, к которому были выдвинуты следующие требования:

- Поддержка множественной фильтрации;
- Поддержка сортировки;
- Приемлемое для пользователя время поиска (до 8 секунд);

- Синхронизация данных с существующей версией системы ЕФГИ;
- Интеграция с платформой интерактивных карт (ПИК).

3. Архитектура системы

Архитектура системы изображена на рис. 2. В качестве базы данных для поиска был выбран ClickHouse [3]. ClickHouse специализируется на аналитической обработке данных, что позволяет быстрее получать данные в сравнении с базами, которые специализируются на транзакционной обработке данных, как например PostgreSQL, используемая в системе ЕФГИ 1.0 [4].

В базе MongoDB хранится полная информация о реестровом объекте, которая нужна при открытии реестровой карточки у найденного в реестре объекта. При создании нового реестрового объекта информация по нему попадает сразу в 2 базы.

Также система ЕФГИ 2.0 в данный момент работает параллельно с ЕФГИ 1.0 и, согласно ранее заявленному бизнес-требованию, данные в них должны быть синхронизированы. Синхронизация происходит по протоколу AMQP.

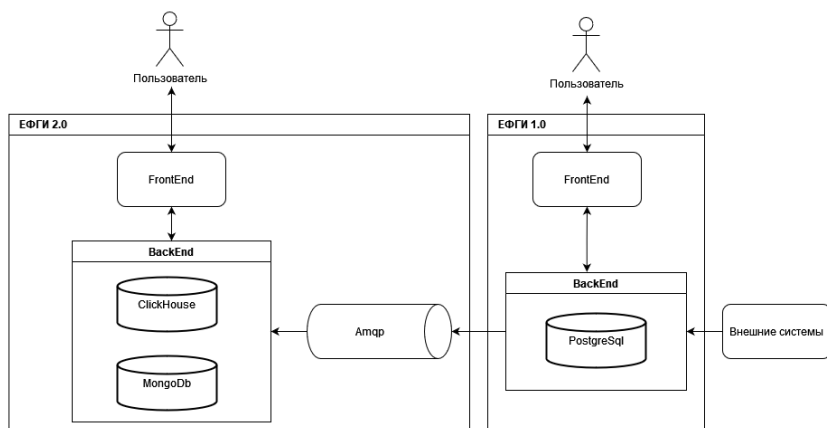
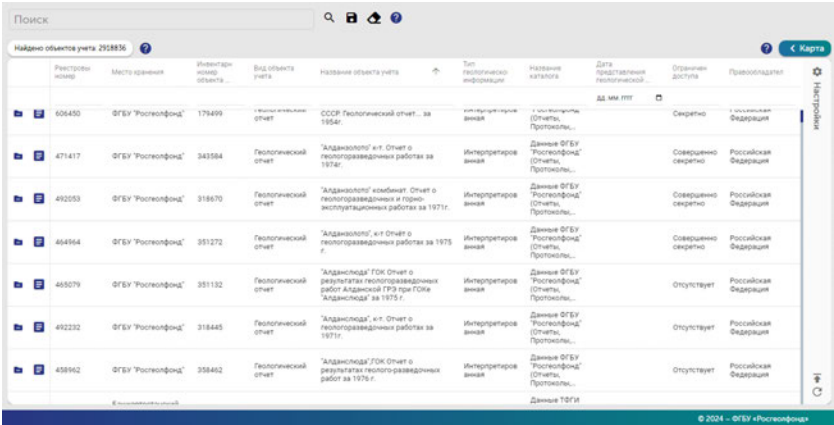


Рис. 2. Архитектура модуля реестра ЕФГИ 2.0

4. Графический интерфейс

Реестр представлен в виде таблицы, в колонках которой пользователь выставляет необходимые ему фильтры (рис. 3). При этом большинство фильтров выбираются из дополнительной панели, так как в таблицу они не помещаются. При выставлении таких фильтров пользователь не видит данные по ним, он видит только поля в таблице.

Поэтому при выборке данных используются только поля, находящиеся в таблице, что ускоряет поиск в сравнении с тем, если бы выбирались все поля.



Поиск

Найдено объектов учета: 250836

Идентификатор	Местонахождение	Идентификатор объекта	Вид объекта	Наименование объекта учета	Тип геологической информации	Наименование категории	Дата предоставления геологической информации	Статус доступа	Правовладитель
606450	ФГБУ "Росгеофонд"	178499	Исходные данные	СССР. Геологический отчет... за 1954г.	интерпретированная	Геологический отчет, Протокол...	ДД. ММ. ГГГГ	Секретно	Российская Федерация
671417	ФГБУ "Росгеофонд"	343584	Геологический отчет	"Алданскотл" и т. Отчет о геологических работах за 1974г.	интерпретированная	Данные ФГБУ "Росгеофонд" (Отчеты, Протоколы...		Современно секретно	Российская Федерация
492053	ФГБУ "Росгеофонд"	318670	Геологический отчет	"Алданскотл" комбинат. Отчет о геологических и горно-эксплуатационных работах за 1971г.	интерпретированная	Данные ФГБУ "Росгеофонд" (Отчеты, Протоколы...		Современно секретно	Российская Федерация
494964	ФГБУ "Росгеофонд"	351272	Геологический отчет	"Алданскотл" комбинат. Отчет о геологических работах за 1975 г.	интерпретированная	Данные ФГБУ "Росгеофонд" (Отчеты, Протоколы...		Современно секретно	Российская Федерация
465079	ФГБУ "Росгеофонд"	351132	Геологический отчет	"Алданскотл" ГОК. Отчет о результатах геологических работ Алданский ГРЗ при ГОКе "Алданскотл" за 1975 г.	интерпретированная	Данные ФГБУ "Росгеофонд" (Отчеты, Протоколы...		Отсутствует	Российская Федерация
492232	ФГБУ "Росгеофонд"	318445	Геологический отчет	"Алданскотл" и т. Отчет о геологических работах за 1971г.	интерпретированная	Данные ФГБУ "Росгеофонд" (Отчеты, Протоколы...		Отсутствует	Российская Федерация
439642	ФГБУ "Росгеофонд"	338462	Геологический отчет	"Алданскотл" ГОК. Отчет о результатах геологических работ за 1976 г.	интерпретированная	Данные ФГБУ "Росгеофонд" (Отчеты, Протоколы...		Отсутствует	Российская Федерация

© 2024 – ФГБУ "Росгеофонд"

Рис. 3. Графическое представление реестра ЕФГИ 2.0

Также была реализована интеграция с платформой интерактивных карт (ПИК). При поиске в реестре на карте отображается список найденных реестровых объектов (рис. 4). Запрос в систему ПИК для отображения объектов на карте происходит асинхронно, что позволяет не замедлять запрос на поиск данных в реестре.

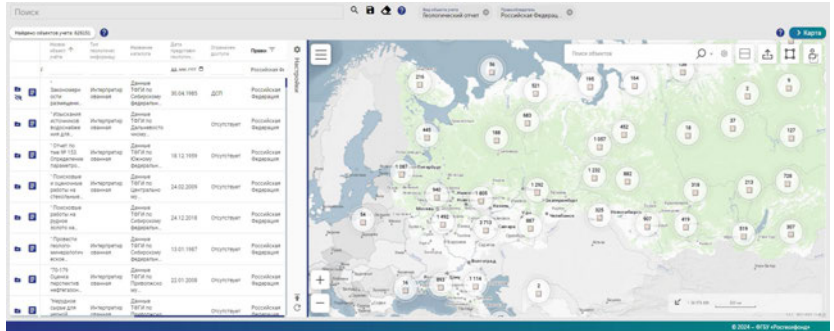


Рис. 4. Интеграция с платформой интерактивных карт

5. Результаты оптимизации

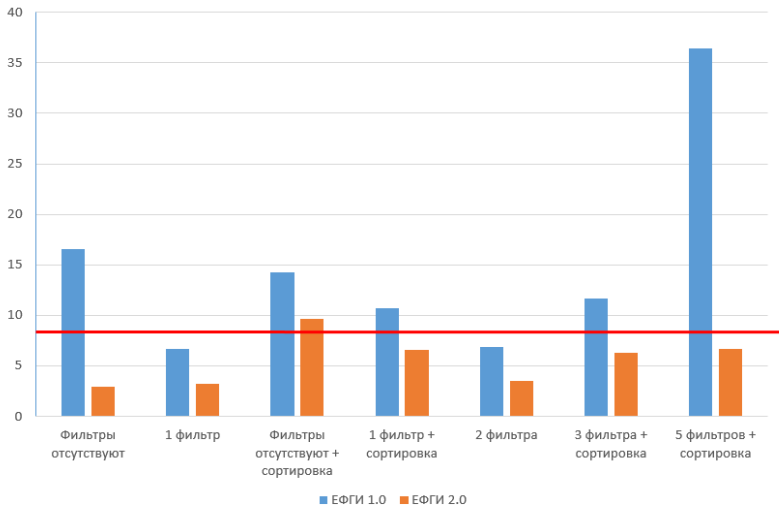


Рис. 5. Гистограмма сравнения времени поиска

Замеры времени поиска в реестре ЕФГИ 2.0 показывают значительный рост производительности (рис. 5). Время поиска в среднем снизилось в 2 раза, при этом запросы любой сложности укладываются в разумное время, как в ситуации с 5-ти фильтрами и сортировкой. Также почти во всех случаях время поиска не превышает среднее время концентрации внимания человека.

В результате опроса было выявлено, что сотрудник фонда в день пользуется поиском в реестре в среднем 20 раз, при этом используя от 1 до 3 фильтров с сортировкой. С помощью оптимизации реестра за месяц один сотрудник экономит около 30 минут только на поиск в реестре, а для Росгеолфонда в целом данная оптимизация экономит 100 человеко-часов в месяц.

Заключение

Результатом выполнения работы является реализованный модуль реестра для системы единого фонда геологической информации. Все поставленные в начале реализации приложения функциональные требования были выполнены. Приложение является востребованным и полностью работоспособным.

Список литературы

1. Документация PostgreSQL [Электронный ресурс]. – Режим доступа: <https://www.postgresql.org/docs>
2. Исследование концентрации внимания человека от Microsoft [Электронный ресурс]. – Режим доступа: <https://sherpapag.com/wp-content/uploads/2017/12/MAS.pdf>
3. Документация ClickHouse [Электронный ресурс]. – Режим доступа: <https://clickhouse.com/docs>
4. Преимущества аналитических баз данных [Электронный ресурс]. – Режим доступа: <https://clickhouse.com/docs/ru/faq/general/olap>

Распознавание типа внутренних элементов планировки зданий с использованием нейронных сетей

А. Ю. Сахаров

Студент магистр

Е. Ю. Митрофанова

Доцент

Введение

В современном мире нейронные сети стали неотъемлемой частью множества областей человеческой деятельности. Они применяются для решения большого количества разнообразных задач, в том числе задач распознавания. Наибольшую популярность и эффективность для решения подобных задач имеет применение сверточной нейронной сети [2].

Данная работа посвящена обучению модели сверточной нейронной сети для решения задачи распознавания типа внутренних элементов планировки зданий используя полутоновые изображения, содержащие границы распознаваемых объектов, а также разработке программного средства для пользовательского взаимодействия с обученной моделью.

1. Сверточная нейронная сеть

Сверточная нейронная сеть (Convolutional Neural Network, CNN) – это класс нейронных сетей, специально разработанных для обработки и анализа многомерных структурированных данных, таких как изображения [1].

Ниже представлена краткая характеристика основных составляющих архитектуры сверточной нейронной сети.

1. Сверточные слои (Convolutional Layers) – слои выполняющие операцию свертки для извлечения признаков из изображения.

2. Слои подвыборки (Pooling Layers) – слои которые уменьшают размерность пространства признаков, уменьшая количество параметров и вычислительную сложность модели. Самый распространенный тип слоев подвыборки – слои максимальной подвыборки, которые выбирают максимальное значение из определенной области.

3. Полносвязные слои (Fully Connected Layers) – слои представляющие собой стандартные нейронные слои, в которых каждый

нейрон связан с каждым нейроном предыдущего слоя. Полносвязные слои обычно используются для выполнения классификации на основе признаков, извлеченных из сверточных слоев.

4. Функции активации (Activation Functions) – функции применяемые к выходам каждого нейрона для добавления нелинейности.

Структура сети, рассматриваемая в рамках данной статьи представлена на рис. 1. Она состоит из 3 слоев свертки, 2 слоев подвыборки и 2 полносвязных слоев. В качестве функций активации использовались Softmax и Relu.

Layer (type)	Output Shape	Param #
sequential_1 (Sequential)	(None, 64, 64, 1)	0
rescaling (Rescaling)	(None, 64, 64, 1)	0
conv2d (Conv2D)	(None, 62, 62, 32)	320
max_pooling2d (MaxPooling2D)	(None, 31, 31, 32)	0
conv2d_1 (Conv2D)	(None, 29, 29, 64)	18496
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 64)	0
conv2d_2 (Conv2D)	(None, 12, 12, 64)	36928
flatten (Flatten)	(None, 9216)	0
dense (Dense)	(None, 64)	589888
dense_1 (Dense)	(None, 3)	195
activation (Activation)	(None, 3)	0

Рис. 1. Структура сверточной нейронной сети

2. Создание обучающего набора изображений

Для обучения модели сверточной нейронной сети была собрана база полутоновых изображений, содержащих элементы планировки зданий (5250 изображений). Полутоновые изображения содержат лишь определенные характеристики исследуемых элементов, такие как края или границы. Использование такой базы данных позволит создать общее представление о типе помещения независимо от конкретного здания.

Процесс создания базы изображений включает запись видеосегментов, представляющих собой движение по различным

участкам здания. Видеофрагменты были разделены на категории в зависимости от типа движения и расположения в помещении. Например, были записаны видеофрагменты движения по лестницам, около дверей и окон. Пример раскадровки одного из фрагментов видеозаписи движения возле двери представлен на рис. 2.

Для увеличения объема изображений, использующихся для обучения модели, применялась аугментация данных, включающая небольшой поворот и масштабирование исходного изображения.

Также в общую базу изображений были включены подходящие изображения из открытых датасетов.

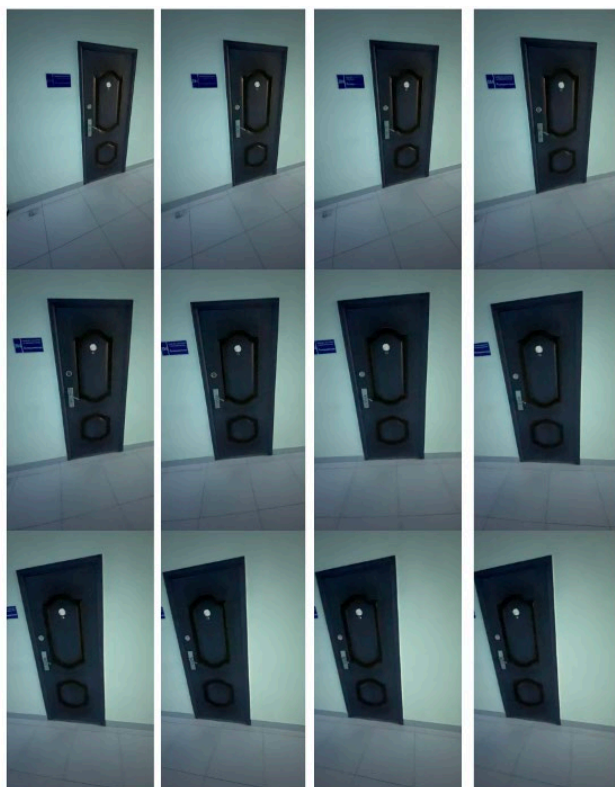


Рис. 2. Раскадровка движения возле двери

3. Предобработка изображений

Как было упомянуто ранее, для обучения модели будут использоваться полутоновые изображения с выделенными чертами распознаваемых объектов. Для этого каждое цветное изображение из исходного набора данных проходило последовательно несколько этапов предобработки [3]:

- Сглаживание. Для выполнения этой операции использовалась реализация фильтра `edgePreservingFilter` из библиотеки `OpenCV`. Применение этого фильтра к изображению позволяет сохранить границы объектов, при этом уменьшить шум и сделать изображение более гладким. Результат применения `edgePreservingFilter` представлен на рис. 3.

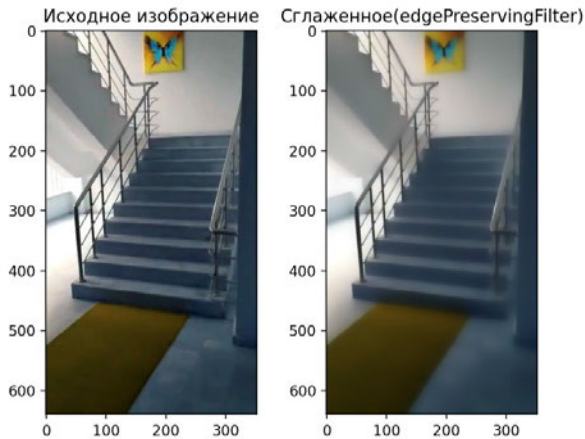


Рис. 3. Результат применения `edgePreservingFilter`

- Перевод изображения из цветного в полутоновое. Результат представлен на рис. 4.

- Выделение границ. Для этих целей использовалась реализация оператора `Laplacian` из библиотеки `OpenCV`. Результат представлен на рис. 5.

- Уменьшение размера изображения. Размер изображения был изменен до 64x64 пикселей, чтобы соответствовать входным требованиям модели нейронной сети.

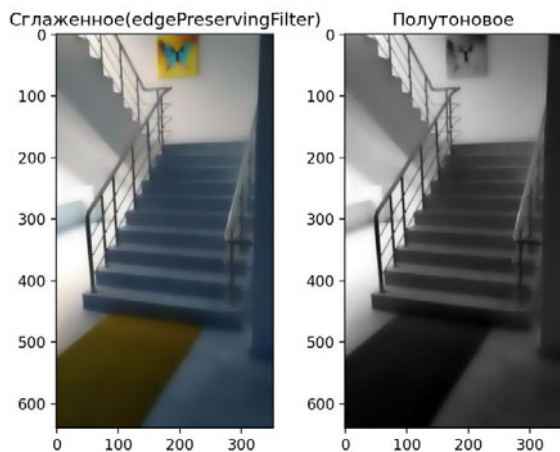


Рис. 4. Результат перевода изображения из цветного в полутонное

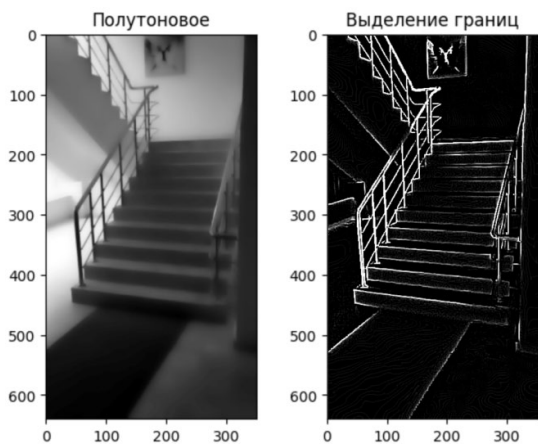


Рис. 5. Результат применения оператора Laplacian

4. Обучение модели и реализация веб-приложения

Обучение модели производилось с использованием ресурсов, предоставляемых google colab, а также открытой библиотеки для машинного обучения TensorFlow [4][5]. Обучение составило 22 эпохи и была достигнута точность 95.72%

Для удобного использования обученной модели было разработано веб-приложение. Для реализации серверной части был выбран язык программирования Python вместе с фреймворком для создания веб-приложений Flask. Для реализации клиентской части использовались HTML, CSS и JavaScript.

Использование приложения начинается с загрузки изображения, на котором должен быть распознан объект. Для этого необходимо переместить графический файл в специальную область для загрузки, представленную на рис. 6.

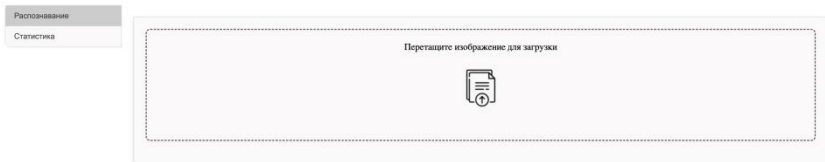


Рис. 6. Внешний вид области загрузки изображения

Затем загруженный файл проходит проверку на соответствие ожидаемому формату, после чего он отправляется на сервер. Сервер сохраняет полученное изображение во временное хранилище для последующей работы с ним. После чего изображение проходит упомянутую ранее предобработку (сглаживание, перевод изображения из цветного в полутоновое, выделение границ). Затем оно подается на вход модели обученной сверточной сети. В качестве ответа сервер возвращает предобработанное изображение, а также класс, к которому принадлежит объект на изображении.

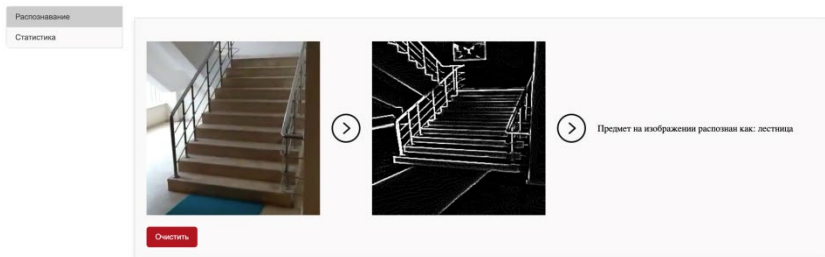


Рис. 7. Отображение ответа от сервера

Заключение

Данная статья посвящена реализации подхода распознавания типа внутренних элементов планировки зданий с применением сверточной нейронной сети, используя для ее обучения полутоновые изображения, содержащие границы распознаваемых объектов. Описана структура используемой модели сверточной нейронной сети, создание обучающего набора данных и его предобработка. В результате получилась обученная модель, имеющая точность распознавания более 95%, а также веб-приложение, позволяющее использовать модель. В дальнейшем предполагается расширить обучающий набор новыми объектами планировки зданий, а также провести сравнение с моделью, обученной на цветных изображениях

Список литературы

1. Дорогой, Я. Ю. Архитектура обобщенных сверточных нейронных сетей / Я. Ю. Дорогой // Информатика, управление и вычислительная техника. – 2011. – №54. – С. 229-234
2. Отус [Электронный ресурс] : информ.-справочный портал – Режим доступа : <https://otus.ru/nest/post/1263/>
3. Смирнов, А. В. Распознавание типа внутренних элементов планировки зданий в задачах первичной навигации мобильных роботов с использованием нейронных сетей / А. В. Смирнов, Д. Н. Степанов // Программные системы: теория и приложения. – 2018. – № 4. – С. 279-291.
4. Гафаров, Ф. М. Искусственные нейронные сети и их приложения: учебное пособие / Ф. М. Гафаров, А. Ф. Галимянов. – Казань: Издательство Казанского университета, 2018. – 121 с.
5. TowardsDataScience [Электронный ресурс] : информ.-справочный портал – Режим доступа : <https://towardsdatascience.com/yolo-object-detection-with-opencv-and-python-21e50ac599e9>

Использование модуля самовнимания и архитектуры трансформер для решения задачи разделения аудиосигнала

Н. А. Селиверстов

Студент магистр

А. Ю. Иванков

Доцент

Введение

Разработка решений, осуществляющих обработку сигналов, является одним из важных направлений исследований в области искусственного интеллекта, так как сфера применения таких решений крайне обширна. Частью данного направления является обработка аудиосигналов при помощи нейронных сетей. Успехи, достигнутые в этой области, позволили эффективно применять глубокое обучение для решения задач классификации аудиосигналов, распознавания и генерации голоса, фильтрации шумов.

Одним из направлений обработки аудиосигналов является работа с музыкальными сигналами. Так технологии искусственного интеллекта применяются для жанровой классификации музыкальных произведений, их генерации, а также для программной эмуляции музыкальных инструментов.

Задача разделения музыкальных источников также входит в область задач, решение которых в настоящее время основано на использовании методов глубокого обучения.

В рамках данной статьи рассмотрена модель глубокого обучения, выполняющая разделение музыкального аудиосигнала с использованием модулей полосового самовнимания.

1. Обзор архитектуры

Поскольку задача разделения аудиосигнала является разновидностью семантической сегментации, многие модели глубокого обучения, используемые для ее решения, основаны на классических архитектурах сегментации изображений, в частности UNet-like архитектурах [2, 7, 8].

Также успехи архитектуры трансформер, изначально предложенной для обработки естественного языка, в решении задач,

связанных с сегментацией изображений, позволили успешно применять ее и для разделения аудиосигнала [2, 9].

Специфика музыкальных сигналов заключается в полосовом характере их частотных представлений: инструменты, которые воспроизводят звуки, принадлежащие музыкальному строю, формируют системы взаимосвязанных горизонтальных полос, в то время как ударные инструменты формируют системы вертикальных частотных полос. Для учета взаимосвязи между частотными полосами в процессе обучения нейронной сети хорошо подходят модули самовнимания и архитектура трансформер.

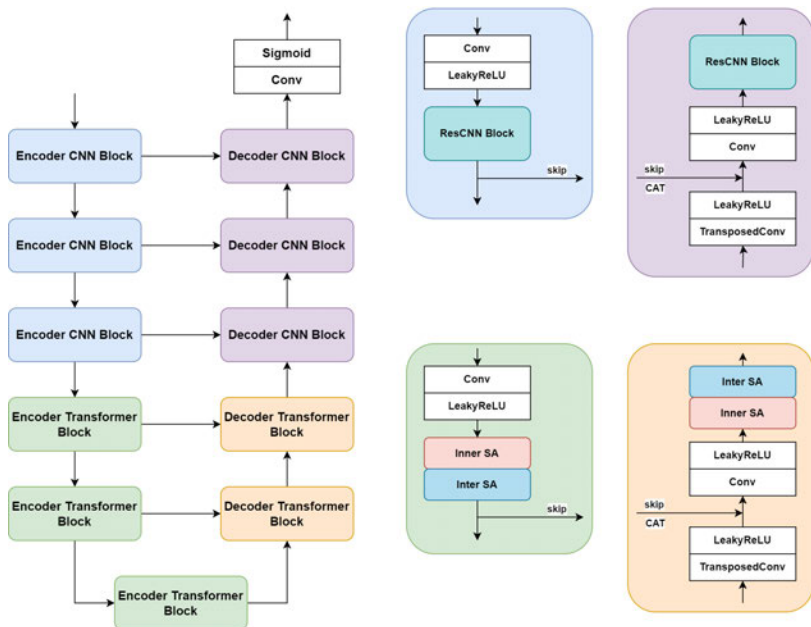


Рис. 1. Схема предлагаемой гибридной нейронной сети

Для реализации такой возможности предлагается архитектура гибридной нейронной сети, изображенная на рисунке 1. Данная нейросеть имеет UNet-like структуру. Первые три блока ее энкодера состоят из сверточного слоя, удваивающего количество каналов и уменьшающего вдвое высоту и ширину тензора, и следующего за ним остаточного сверточного блока (Residual Convolutional Block) [3]. Эти блоки служат для формирования признаков, подаваемых на вход первому из пяти последовательно соединенных трансформер-блоков,

составляющих «бутылочное горлышко» сети и осуществляющих полосовое самовнимание [1, 2]. Тензор, получаемый на выходе секции трансформер-блоков, обрабатывается тремя последовательными блоками, состоящими из слоя обратной свертки, который уменьшает вдвое количество каналов и удваивает высоту и ширину тензора, и следующего за ним остаточного сверточного блока. Выход каждого блока, находящегося в энкодере, конкатенируется со входом соответствующего блока декодера.

2. Остаточный сверточный блок

Остаточные сверточные блоки, входящие в состав энкодера нейронной сети, служат для эффективного формирования признаков, для дальнейшей обработки трансформер-блоками. Эти же блоки, являющиеся частью декодера, обеспечивают извлечение признаков.

Остаточный сверточный блок состоит из двух сверточных слоев, имеющих размер ядра 3×3 . Каждому из них предшествуют функция активации LeakyReLU и слой нормализации BatchNorm. Вход и выход блока соединены через сверточный слой, имеющий размер ядра 1×1 .

Структура остаточного сверточного блока приведена на рисунке 2.

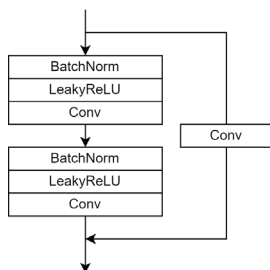


Рис. 2. Схема остаточного сверточного блока [3]

3. Трансформер-блок

Трансформер-блок, выполняющий полосовое самовнимание, служит для формирования признаков, учитывающих взаимоотношения частотных составляющих как внутри одной частотной полосы, так и между соседними частотными полосами.

Таким образом, в сети представлены два вида трансформер-блоков: первый выполняет внутриполосовое самовнимание, а второй – межполосовое. Каждый из двух видов самовнимания применяется для обработки как горизонтальных, так и вертикальных частотных полос. Логика самовнимания для обоих видов частотных полос идентична за

исключением используемого измерения, по которому формируются полосы.

Рассмотрим механизм работы модуля внутриполосового самовнимания. На вход модуля подается тензор $X \in x^{H \times W \times C}$, где H – высота тензора, W – ширина тензора, C – количество каналов в нем. Полученный тензор последовательно обрабатывается при помощи нормализации LayerNorm и сверточного слоя с размером ядра 1×1 , чтобы получить тензоры признаков, подаваемые на вход блоков горизонтального и вертикального самовнимания:

$$(X^h, X^v) = \text{Conv}(\text{LayerNorm}(X)), \quad (1)$$

где $X^h, X^v \in x^{H \times W \times D}$ – тензоры входных признаков для блоков горизонтального и вертикального самовнимания, $D = C / 2$ – количество каналов для каждого из тензоров.

Далее рассмотрим работу блока горизонтального самовнимания. Входной тензор X^h представляется как тензор токенов $X_i^h \in x^{W \times D}$, $i = \{1, 2, \dots, H\}$. Матрицы Query, Key и Value $Q_{ij}^h, K_{ij}^h, V_{ij}^h \in x^{W \times D/m}$, $j = \{1, 2, \dots, m\}$, где m – количество голов в блоке самовнимания, формируются следующим образом:

$$(Q_{ij}^h, K_{ij}^h, V_{ij}^h) = (X_i^h P_j^Q, X_i^h P_j^K, X_i^h P_j^V), \quad (2)$$

где $P_j^Q, P_j^K, P_j^V \in x^{D \times D/m}$ – матрицы линейной проекции.

Признаки $O_{ij}^h \in x^{W \times D/m}$ горизонтальной полосы i для головы j блока внимания вычисляются следующим образом:

$$O_{ij}^h = \text{Softmax}\left(\frac{Q_{ij}^h (K_{ij}^h)^T}{\sqrt{D/m}}\right) V_{ij}^h. \quad (3)$$

Полученные тензоры объединяются по каналам, формируя общий тензор для полосы i , после чего тензоры всех полос объединяются в тензор $O^h \in x^{H \times W \times D}$. Аналогичным образом формируется выходной тензор для блока вертикального самовнимания $O^v \in x^{H \times W \times D}$.

Выходные тензоры блоков горизонтального и вертикального самовнимания объединяются и проходят через сверточный слой с размером ядра 1×1 , после чего складываются с входным тензором X для получения тензора $O_{attn} \in x^{H \times W \times C}$:

$$O_{attn} = \text{Conv}(\text{Conc}(O^h, O^v)) + X, \quad (4)$$

Для получения выходного тензора $O_{inner} \in x^{H \times W \times C}$ признаков модуля внутриполосового самовнимания тензор O_{attn} последовательно проходит через нормализацию LayerNorm, многослойный персептрон (MLP), складывается со своим исходным значением и подается на вход блока позиционного кодирования (Conditional Positional Encoding):

$$O_{inner} = CPE(MLP(Norm(O_{attn})) + O_{attn}). \quad (5)$$

Рассмотрим механизм работы модуля межполосового самовнимания. Тензоры входных признаков $X^h, X^v \in x^{H \times W \times D}$ формируются по формуле (1). Затем из полученных тензоров вычисляются трехмерные тензоры $Q_i^h, K_i^h, V_i^h \in x^{H \times W \times D/m}$. Данные тензоры приводятся к двумерному представлению с размерностью $H \times D^h / m$, где $D^h = W \times D$. Таким образом, полученные матрицы представляют собой набор токенов, каждый из которых является последовательностью всех частотных признаков одной полосы.

Тензор $O_j^h \in x^{H \times D^h / m}$ признаков для головы j блока горизонтального самовнимания вычисляется следующим образом:

$$O_j^h = \text{Softmax}\left(\frac{Q_j^h (K_j^h)^T}{\sqrt{D^h / m}}\right) V_j^h. \quad (6)$$

Аналогичным образом вычисляются тензоры $O_j^v \in x^{W \times D^v / m}$ признаков для каждой из голов блока вертикального самовнимания, где $D^v = H \times D$. Полученные тензоры объединяются по каналам, формируя тензоры $O^h \in x^{H \times D^h}$ и $O^v \in x^{W \times D^v}$, которые затем приводятся к трехмерному представлению с размерностью $H \times W \times D$.

Наконец, выходной тензор $O_{inter} \in x^{H \times W \times C}$ признаков модуля межполосового самовнимания вычисляется путем применения операций (4) и (5) к полученным тензорам O^h и O^v .

Схемы модулей самовнимания приведены на рис. 3.

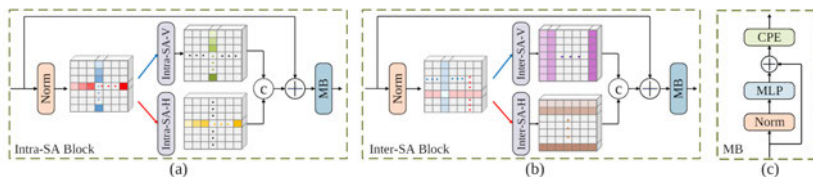


Рис. 3. Схемы внутрисполосового (а) и межполосового (б) модулей самовнимания [1]

4. Подготовка обучающих данных

Для обучения модели использован датасет MUSDB18, содержащий 150 композиций, 86 из которых формируют обучающую выборку, 14 – валидационную выборку, 50 – тестовую выборку [6]. Все композиции имеют частоту дискретизации 44100 Гц и звуковые частоты до 16000 Гц.

Для каждой композиции помимо смешанного сигнала (самой композиции) заданы также четыре целевых сигнала выделяемых источников: ударные, бас, вокал, остальной аккомпанемент. Пример магнитуд смешанного сигнала и целевого сигнала вокала на рисунке 4.

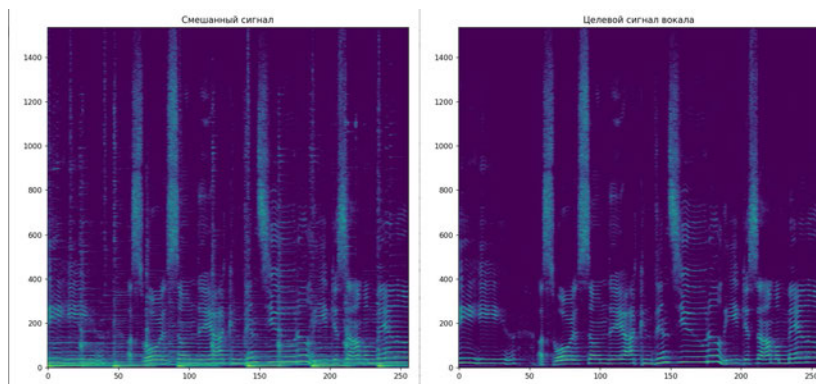


Рис. 4. Магнитуды смешанного сигнала и целевого сигнала вокала

Так как архитектура трансформер чувствительна к объему выборки и требует большого количества обучающих данных, использованы два метода аугментации – смена каналов аудиосигнала и линейное суммирование отрезков из различных композиций [2].

Для получения частотного представления сигналы обрабатываются скользящим оконным преобразованием Фурье, имеющим ширину окна 4096 и расстояние между окнами 1024. Для обучения сети случайным

образом выбираются фрагменты полученных спектрограмм длиной 256. Таким образом, один образец обучающих данных представляет собой тензор $X \in \mathbb{C}^{2 \times 2049 \times 256}$, однако ограничение частоты позволяет обрезать высоту тензора и получить тензор $X' \in \mathbb{C}^{2 \times 1536 \times 256}$. Непосредственно на вход сети поступает магнитуа выбранного образца $M = |X'| \in \mathbb{R}^{2 \times 1536 \times 256}$.

Выходом сети является маска магнитуды $\hat{M} \in \mathbb{R}^{2 \times 1536 \times 256}$. Таким образом, комплексная спектрограмма восстановленного сигнала источника вычисляется следующим образом [3]:

$$\hat{S} = (\hat{M} \times |M|) e^{j\angle X'}. \quad (7)$$

5. Процесс обучения и полученный результат

В процессе обучения размер батча установлен 16, использован оптимизатор Adam с темпом обучения 0.0001. В качестве функции потерь использована функция L1, принимающая целевую и восстановленную магнитуды источника.

Количество каналов для остаточных сверточных блоков составляет 32, 48 и 64. Для трансформер-блоков количество каналов – 128, 256, 512 и голов – 4, 8, 16.

Пример магнитуд целевого и выделенного сигналов вокала приведен на рис. 5.

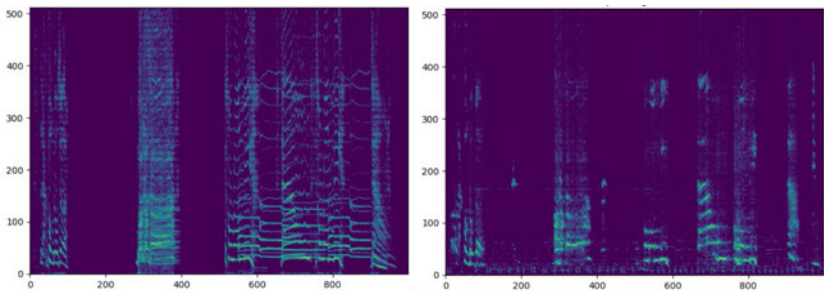


Рис. 5. Магнитуды целевого сигнала и выделенного сигнала вокала

Среднее значение метрики SDR на тестовой выборке составило 1.9 Дб [5]. При этом среднее значение метрики, полученное при обучении модели, в которой трансформер-блоки были заменены остаточными сверточными блоками составило 0.9 Дб. Таким образом, гибридная модель позволила добиться более высокого качества разделения, чем модель, составленная полностью из сверточных блоков. Низкое

абсолютное значение метрики на тестовой выборке может объясняться тем, что использованная функция потерь не учитывает информацию о фазе сигнала, а также вычисляется исключительно для частотного представления сигнала.

6. Направления развития

Можно выделить несколько направлений развития предложенной модели:

- искусственное расширение набора обучающих данных путем применения дополнительных видов аугментаций;
- получение комплексной Ideal Ratio Mask на выходе нейросети вместо оценки маски магнитуды. Это позволит сети также предсказывать фазу восстанавливаемого сигнала [3];
- реализация функции потерь, учитывающей информацию о фазе сигнала, а также работающей и с временным, и с частотным представлениями сигнала [3, 4].

Также потенциальное улучшение архитектуры нейронной сети может быть достигнуто благодаря учету особенностей музыкального строя, заключающихся в том, что плотность информации неравномерна во всей частотной области. Рассмотрим наиболее распространенный в современной музыке равномерно темперированный строй, в котором частота ноты может быть получена из следующего соотношения:

$$f(i) = f_0 \cdot 2^{i/12}, \quad (8)$$

где f_0 – частота камертона, i – количество полутонов между целевой нотой и эталонной нотой f_0 . Очевидно, что частоты нот формируют геометрическую прогрессию, следовательно, в некоторой области высоких частот находится меньше нот, чем в такой же области более низких частот. Для учета этих особенностей может применяться моделирование частотных полос, которое изменяет их размерности, с целью усреднения плотности информации [4].

Заключение

В рамках данной работы предложена архитектура гибридной нейронной сети, выполняющей разделение музыкальных источников с использованием модуля полосового самовнимания. Используя предложенную архитектуру, была разработана, реализована и обучена модель. В ходе работы проведено сравнение полученной модели с моделью, состоящей только из остаточных сверточных слоев. На основании анализа полученных результатов с учетом условий предметной области предложен ряд потенциальных улучшений архитектуры.

Список использованной литературы

1. Stripformer: Strip Transformer for Fast Image Deblurring / F. Tsai [et al.] // European Conference on Computer Vision. – 2022. – DOI: 10.48550/arXiv.2204.04627.
2. Stripe-Transformer: deep stripe feature learning for music source separation / J. Qian [et al.] // EURASIP Journal on Audio, Speech, and Music Processing. – 2023. – DOI: 10.1186/s13636-022-00268-1.
3. Decoupling Magnitude and Phase Estimation with Deep ResUNet for Music Source Separation / Q. Kong [et al.] // International Society for Music Information Retrieval Conference. – 2021. – DOI: 10.5281/zenodo.5624475
4. Yi L. Music Source Separation with Band-split RNN / L. Yi, Y. Jianwei // IEEE/ACM Transactions on Audio Speech and Language Processing. – 2022. – DOI: 10.48550/arXiv.2209.15174.
5. Vincent, E. Performance measurement in blind audio source separation / E. Vincent, R. Gribonval, C. Févotte // IEEE Transactions on Audio Speech and Language Processing. – 2006. – DOI: 10.48550/arXiv.2209.15174.
6. MUSDB18 – a corpus for music separation / Z. Rafili [et al.] // Zenodo e-prints. – 2017. – DOI: 10.5281/zenodo.1117372.
7. Stoller, D. Wave-U-Net: A Multi-Scale Neural Network for End-to-End Audio Source Separation / D. Stoller, S. Ewert, S. Dixon // International Society for Music Information Retrieval Conference (ISMIR). – 2018. – DOI: 10.48550/arXiv.1806.03185.
8. Defossez A. Music Hybrid Spectrogram and Waveform Source Separation / A. Defossez // arXiv e-prints. – 2022. – DOI: 10.48550/arXiv.2111.03600.
9. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows / Z. Liu [et al.] // IEEE International Conference on Computer Vision. – 2021. – DOI: 10.1109/ICCV48922.2021.00986.

Анализ нейросетей для обнаружения киберугроз

И. В. Сияков

Студент магистр

В. С. Тарасов

Старший преподаватель

Введение

Защита информационных систем от киберугроз является основной задачей в области информационной безопасности. В наши дни технологии защиты информации стремительно развиваются, но и уровень киберпреступности растет в связи с разработкой злоумышленниками новых способов обхода защиты инфраструктур. Число кибератак постоянно увеличиваются, и это требует больших инноваций в области защиты от киберугроз.

Ограничения на применение традиционных методов обнаружения атак, таких как сигнатурный анализ и обнаружение аномалий, имеют место быть. При обнаружении новых и неизвестных атак, они могут оказаться не эффективными. Данная информация открывает возможности для использования нейросетей, которые могут обучаться на больших объемах данных и обнаруживать сложные образцы атак.

Интерес в использовании машинного обучения растет во многих областях науки и техники, особенно в области кибербезопасности. В связи с тем, что появилось большое количество программного обеспечения и технологии, позволяющие использовать нейросети, становятся все более распространенными, методы машинного обучения могут быть использованы для анализа большого количества данных и определения киберугроз.

Существует множество алгоритмов и способов, которые используются в машинном обучении. Они делятся на контролируемые и неконтролируемые. С помощью таких методов, как контролируемое обучение, можно использовать в контексте классификации, когда входные и выходные данные сопоставляются друг с другом или происходит регрессия.

В основе неконтролируемого обучения лежит кластеризация, которая используется для проведения исследовательских работ и уменьшения размерности[1].

Оба этих подхода могут быть использованы в области кибербезопасности для выявления вредоносного ПО с помощью современных технологий, которые позволяют проводить анализ в реальном времени. Это позволяет устранить недостатки традиционных методов обнаружения угроз.

1. Классификация сетевых атак

Сетевая атака – это действия, которые направлены на реализацию угроз несанкционированного доступа к информации, воздействия на нее или на ресурсы системы, с использованием программных и (или) технических средств.

Обнаружение сетевых атак – это процесс распознавания и реагирования на подозрительную деятельность, направленную на сетевые или вычислительные ресурсы организации. В зависимости от используемых методов анализа, которые применяются для обнаружения сетевых атак, сильно влияет эффективность их обнаружения.

Анализ трафика сети является одним из распространенных способов применения нейросетей. Нейросеть может быть обучена на большом количестве данных трафика, чтобы распознавать нормальное поведение сети и выявлять любые аномалии, которые могут указывать на наличие уязвимости[1].

Также данный метод используется для анализа кода. Благодаря использованию большого количества кода, нейросеть может научиться распознавать структуры и шаблоны, которые могут указывать на наличие уязвимости. При необходимости сканирования больших объемов кодовой базы, это может быть очень полезно.

Для анализа данных о безопасности, таких как журналы событий, нейросети могут использоваться с целью выявления неожиданных действий или атак. Анализ трафика помогает организациям быстро реагировать на угрозы, а также разрабатывать меры по их нейтрализации[2].

Не смотря на все свои преимущества, использование нейросетей не является идеальным способом для поиска уязвимостей. Существует ряд особенностей, которые могут затруднить обнаружение и распознавание их даже для нейросетей, в особенности, если злоумышленник использует новые или неизвестные методы атаки. Кроме того, для проведения обучения нейросети необходимо большое количество информации, что может быть затруднительно для небольших организаций или проектов. В целом, алгоритмы обучения представляют собой мощный инструмент для поиска уязвимостей при хакерских атаках. Они могут использоваться для анализа трафика сети, кода и

данных о безопасности, чтобы выявлять аномалии и потенциальные угрозы[3].

2. Постановка задачи

Основополагающей задачей является подбор алгоритма, который будет наиболее эффективным в использовании нейронной сети. Чтобы осуществить данный анализ, необходимо провести сравнение существующих способов обучения нейросети, сопоставить различные алгоритмы между собой при одном наборе данных, выявить метрики для анализа, по которым будут сравниваться алгоритмы, и затем на базе этих метрик реализовать подходящую под задачу реализацию[3].

В качестве одного из возможных путей решения проблемы, можно рассмотреть обнаружение вредоносного ПО или ботнет-трафика, которые могут быть получены из набора данных NetFlow с помощью различных методов машинного обучения.

Для того, чтобы обнаружить вредоносное ПО или ботнет-трафик из данных Netflow, предлагается следующий подход. В системе должно быть возможно принять любой набор данных Netflow любого размера, чистый или с вредоносным ПО. Затем система должна классифицировать его как обычный или атакующий трафик. С помощью рассматриваемого подхода анализируются данные, предоставленные NetFlow. Содержимое этих записей является достаточно информативным для идентификации трафика и не содержит персональной информации. В настоящий момент, сеть NetFlow и его версия IPFIX, которая соответствует общепринятому стандарту, используются для мониторинга и управления сетями.

3. Алгоритмы обучения нейронной сети

Extreme Learning Machine (ELM) представляет собой алгоритм, который использует нейронные сети с прямолинейным движением. Возможно, что скрытый слой состоит из нескольких слоев. Настраиваемые случайным образом скрытые слои имеют свои весовые коэффициенты, которые определяются алгоритмом на основе анализа. Создатель алгоритма обучения считает, что данный алгоритм имеет возможность обеспечить хорошие результаты обобщения и обучаться в тысячу раз быстрее, чем обычные алгоритмы обучения для нейронных сетей с прямолинейным движением, алгоритм изображен на рис.1.

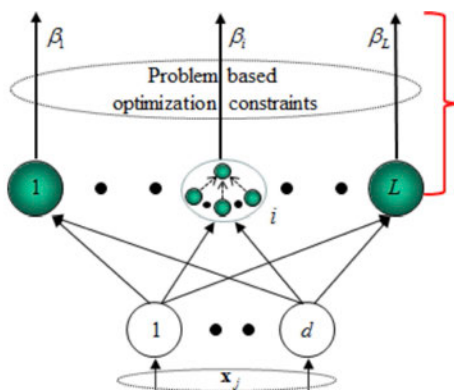


Рис. 1. Структура алгоритма ELM

Для решения задач классификации и регрессии, алгоритм Random Forest (RF) использует несколько деревьев решений. В алгоритме Random Forest заложена концепция мажоритарного голосования нескольких деревьев, что позволяет считать его ансамблевым алгоритмом машинного обучения. Алгоритм, представленный в виде предсказания класса, определяет итог всех классов, которые были предсказаны отдельными деревьями. Недавно проводились исследования, которые показали эффективность Random Forest при атаках на системы безопасности. В их число включались такие атаки как инъекционные и фильтрация спама, а также обнаружение вредоносных программ, алгоритм изображен на рис.2.

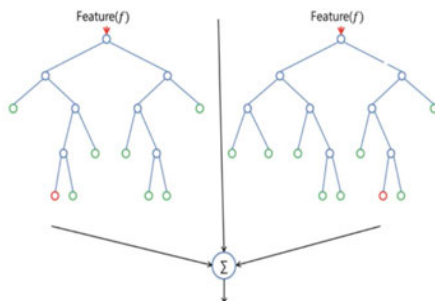


Рис. 2. Структура алгоритма Random Forest

Градиентный бустинг – метод машинного обучения для задач регрессии и классификации, который создает модель предсказания в виде ансамбля слабых моделей предсказания, обычно деревьев

решений. Она сочетает в себе элементы функции потерь, и аддитивной модели. Модель, указанная на рис.3, добавляет обучающие элементы, чтобы минимизировать функцию потерь. Основная идея градиентного наращивания заключается в том, чтобы повторно использовать паттерны в остатках и усилить модель со слабыми предсказаниями, сделав ее лучше, математически это означает минимизацию функции потерь [4,5].

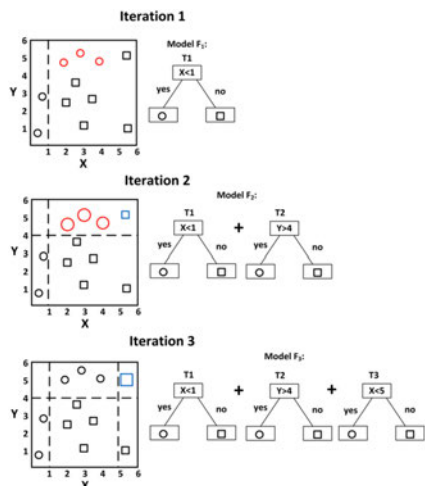


Рис. 3. Структура алгоритма градиентного бустинга

Согласно предложенному способу исследования, задача будет решена следующим образом:

Подготовка данных. Это необходимо для того, чтобы получить информацию о транспортном потоке. Это возможно реализовать, используя получение реального трафика. В случае отсутствия трафика, можно воспользоваться набором данных, которые доступны в открытом доступе. STU-13 является наиболее известным и часто используемым набором для проведения подобных исследований. Также, KDDCUP993 и CIC-IDS-20174 являются наиболее распространенными. Затем следует исследование набора информации с целью выявления общих признаков и частоты. Установлено, что в качестве признаков можно использовать необработанные атрибуты данных Netflow: starttime и duration; proto и srcAddr; sport и state; totPkts; totBytes; srcBytes. С помощью этой группы признаков можно определить потоки.

Извлечение признаков. Чтобы определить и получить признаки, нужно выбрать набор данных. Этот шаг – важная часть методики. Признаки, которые содержатся в данных NetFlow, имеют

категориальный характер, и их необходимо кодировать в числовые или булевы значения, что может привести к слишком большому размеру матрицы и нарушению работы памяти. Для уменьшения объема данных, которые будут обработаны, применяется схема для выбора трафика NetFlow с использованием временного окна. Затем в процессе обработки набора данных, предоставленных NetFlow, из него извлекаются категориальные и числовые характеристики, которые описывают набор данных, который определен в заданном временном промежутке.

Выбор признаков. Методы отбора признаков, используемые на данном этапе, направлены на уменьшение размеров входной обучающей матрицы. Например, фильтрация признаков с помощью корреляции Пирсона или же методы обертывания, которые основаны на Backward Feature Elimination, являются вспомогательными методами в классификаторе Random Forest. Также, в него встроены методы для анализа главных компонент (PCA) и t-распределенное внедрение Stistic Neighbour Embedding (t- distributed).

Сравнение алгоритмов. Предлагается сравнить пять выбранных алгоритмов. Различные модели машинного обучения, которые должны быть обучены на этом этапе, например: Логистическая регрессия, Машина опорных векторов (SVM), Random Forest Classifier, Gradient Boosting и Dense Neural Network.

Обнаружение ботнетов. Последний этап рассматриваемой методологии включает в себя тестирование модели на предмет обнаружения ботнет-трафика из набора данных STU-13. Общая эффективность обнаружения ботнетов определяется по показателю $f1$ вышеупомянутых моделей.

На рис. 4 показано представление первых двух компонент PCA с набором предварительно отобранных признаков (3, 4, 5, 9, 11). Ботнеты помечены как 6, а некоторые точки, не относящиеся к ботнетам, не отображаются. Первая компонента (вертикальная ось) объясняет 58 % дисперсии набора данных, а вторая компонента (горизонтальная ось) – 35 % дисперсии набора данных. Видно, что точки ботнета попадают в группу в самом низу длинной прямой линии фоновых точек.

Это представление можно использовать для обучения классификатора K-Nearest Neighbours[4,5].

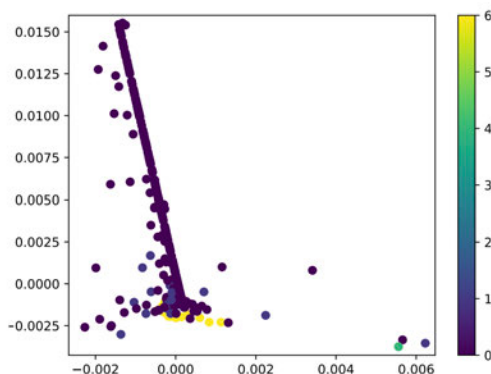


Рис. 4. Анализ главных компонент PCA

4. Результаты

Для того чтобы сравнить результаты работы различных алгоритмов, необходимо выбрать некоторые метрики для оценки эффективности методов. Метрики указаны в таблице. Обычный способ – посмотреть на количество ложноположительных (т.е. количество фоновых сообщений, помеченных как ботнеты) и ложноотрицательных (т.е. количество ботнетов, помеченных как фоновые сообщения).

Таблица

Анализ алгоритмов нейросетей

Алгоритмы	Точность	Отклик	F1, показатель
Логистическая регрессия	0.74	0.97	0.83
Метод опорных векторов	0.94	0.80	0.86
Случайный лес	1.0	1.0	1.0
Градиентный бустинг	1.0	0.99	1.0
Сверточная нейронная сеть	0.90	0.98	0.94

Из таблицы видно, что Random Forest, Gradient Boosting и Dense Neural Network превосходят остальные алгоритмы в обнаружении ботнетов среди сетевого трафика с показателем f1, равным 0,97. Однако методы логистической регрессии и машины опорных векторов показали

f1 на уровне 0,85, причем один из них с низкой точностью, а другой – с низким показателем recall [5].

Заключение

Данная работа является теоретическим исследованием алгоритмов обучения нейронной сети для определения киберугроз. В ходе проделанной работы, были изучены методы обучения нейронной сети, из полученных результатов исследования можно сделать вывод о том, что алгоритм Random Forest является самым эффективным среди других и наиболее быстрым. Самым сложным алгоритмом оказался алгоритм рекурсивных нейронных сетей. Информация, которая легла в основу написания данной статьи потребуется для дальнейшего исследования и выполнения практических работ. Рассмотренные подходы дают представления о том, какие способы обучения стоит рассматривать для практической реализации эффективного алгоритма решения задачи. В дальнейшей работе предстоит рассмотреть эффективность алгоритмов при более сложных сценариях, а также тестирование обучения без контроля, чтобы обнаружить поведение ботнетов без использования меток данных.

Список литературы

1. Анализ зашифрованного трафика без его расшифровки [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/articles/497658/>
2. Cyber Attack Detection thanks to Machine Learning Algorithms [Электронный ресурс]. – Режим доступа: <https://paperswithcode.com/paper/cyber-attack-detection-thanks-to-machine>
3. Технологии обнаружения и предотвращения атак. [Электронный ресурс]. – Режим доступа: <http://www.cnews.ru/reviews/free/security/part8/>
4. Сетевые атаки и технологии их обнаружения. [Электронный ресурс]. – Режим доступа: <http://www.inf74.ru/safety/ofitsialno/setevyie-ataki-i-tehnologii-ihobnaruzheniya/>
5. Хайкин, С. Нейронные сети. Полный курс / С. Хайкин. – М.: Вильямс, 2016. – 1104 с

Сравнение эффективности алгоритмов шифрования и кодирования в контексте мобильной системы сбора данных

В. А. Соколов

Студент магистр

С. А. Зуев

Доцент

Введение

Мобильные системы сбора и обработки данных в современном мире стремительно развиваются, внедряясь в различные области человеческой деятельности, от проката транспорта и систем «умный дом» до оборонной промышленности и контроля режима и безопасности на предприятиях. Рост использования автономных устройств на основе микроконтроллеров ведёт за собой рост требований к связи между ними. Этим требованиям удовлетворяют алгоритмы шифрования и кодирования.

Алгоритмы шифрования и кодирования, как следует из названия, решают две большие категории задач. Первая категория – обеспечение безопасности сети, которое зачастую приводит к увеличению объёма передаваемой информации. Вторая категория – сжатие данных, цель которого заключается в минимизации объёма передаваемой информации для уменьшения расхода трафика. Данная статья посвящена обзору различных алгоритмов шифрования и кодирования, наиболее подходящих для использования в современной мобильной системе сбора данных, с целью дальнейшей реализации алгоритма шифрования для реализованной системы.

1. Ограничения платформы реализации алгоритмов

Мобильные системы сбора данных, основанные на микроконтроллерах, имеют ряд характерных свойств, затрудняющих или накладывающих ограничения на работу некоторых алгоритмов. К этим ограничениям относятся:

- Объём памяти, как оперативной, так и постоянной, зачастую достаточно ограниченный. Это ограничивает использование алгоритмов, требующих большого количества вспомогательной памяти;

- Быстродействие, в зависимости от модели, может сильно отличаться. При большом количестве потоков многопоточные блочные алгоритмы позволят ускорить обработку данных, при малом требуется использование оптимизированных поточных алгоритмов;

- Энергоэффективность, к которой в мобильных системах предъявляются высокие требования, обеспечивается как можно более редким пробуждением контроллера и как можно более быстрой работой алгоритмов в этом периоде.

С учётом этих ограничений необходимо выбрать или разработать собственный набор алгоритмов для каждого конкретного применения. При реализации необходимо отслеживать преимущественно следующие параметры:

- Итоговый объём сообщения;
- Вычислительная сложность;
- Сложность по памяти;
- Криптостойкость.

Далее будут перечислены группы алгоритмов шифрования и кодирования с анализом их применимости в мобильных системах.

2. Симметричные криптографические алгоритмы

Характерной особенностью симметричных криптоалгоритмов, также называемых криптоалгоритмами с закрытым ключом, является использование одного и того же ключа в процессах шифрования и дешифрования. При сравнении их с асимметричными алгоритмами, которые будут рассмотрены далее, симметричные алгоритмы работают быстрее на одном объёме данных, их проще реализовать и модифицировать, и также для достижения аналогичной криптостойкости канала связи требуется меньший размер ключа [1]. К недостаткам относятся сложность управления ключами при росте сети и высокая опасность реализации злоумышленниками уязвимости «атака посередине» при настройке сети [2].

К наиболее применяемым алгоритмам такого типа относятся:

- AES – алгоритм на основе SP-сети. Схема одного SP-раунда представлена на рис. 1.

S-блоки обеспечивают лавинный эффект – изменение одного бита на входе S-блока изменяет все биты на его выходе. Р-блоки обеспечивают перемешивание бит между разными S-блоками. Добавление ключа обеспечивает однозначность шифрования и расшифрования только при использовании такого же ключа. Обеспечивает высокую криптоустойчивость.

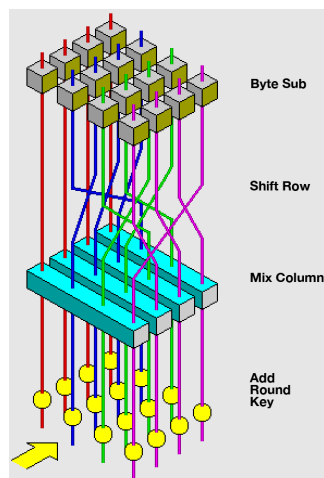


Рис. 1. Схема одного раунда алгоритма AES

- DES, 3DES – алгоритмы на основе сети Фейстеля. Схема одного раунда представлена на рис. 2.

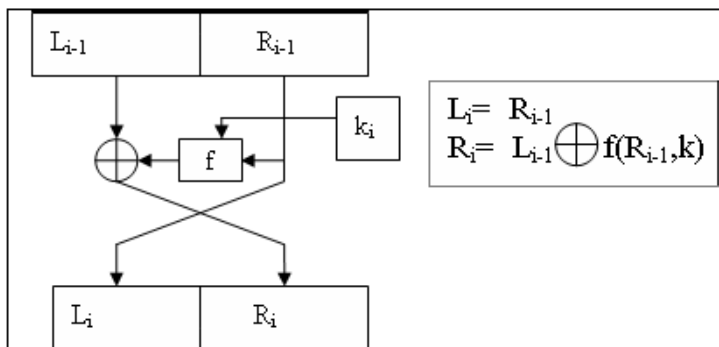


Рис. 2. Блок-схема одного раунда алгоритма DES

Имеют пять режимов работы: ECB – легко распараллеливается, одинаковые блоки шифрует одинаково; CBC, CFB, OFB, - используют промежуточный или конечный результат шифрования предыдущего блока для шифрования следующего; режим счётчика использует дополнительную переменную, известным образом изменяющуюся после шифрования каждого блока.

- Магма – отечественный алгоритм шифрования, устроен по принципу сети Фейстеля. Обеспечивает высокое быстродействие среди алгоритмов сети Фейстеля, обеспечивает защиту от навязывания ложных данных.

- Кузнечик – отечественный алгоритм, использующий SP-сеть для генерации раундовых ключей и операции над полем Галуа для самого шифрования. Устойчив ко всем известным способам атак на блочные шифры, требователен к вычислительной мощности устройства.

3. Асимметричные криптографические алгоритмы

Асимметричные криптоалгоритмы широко используются в современных сетевых протоколах. Их особенностью является использование двух ключей – открытого, используемого в процессе шифрования и не дающего никакой информации при перехвате, и закрытого, при помощи которого осуществляется расшифрование. Такие алгоритмы основаны на использовании односторонних функций с лазейкой: при наличии открытого ключа и открытого текста без закрытого ключа вычисление закрытого ключа или закрытого текста займет достаточно много времени, чтобы информация к моменту расшифровки стала неактуальной [3].

Такие алгоритмы являются гораздо более устойчивыми к атакам, требуют меньшей работы по распределению ключей. При этом асимметричные алгоритмы требуют гораздо больший размер ключа для аналогичного по криптоустойчивости шифрования, чем симметричные. Их вычислительная сложность и сложность по памяти гораздо выше, чем у других типов алгоритмов, и чаще всего для такого шифрования используется узкоспециализированный аппаратный модуль или высокопроизводительный процессор, которые чаще всего отсутствуют в широкодоступных микроконтроллерах.

Наиболее применяемые алгоритмы асимметричного шифрования:

- Протокол Диффи-Хельмана позволяет получить секретный ключ, используя незащищенный от прослушивания канал связи. Основан на вычислении степени чисел и остатка от деления. Без изменений уязвим к атакам посередине, используется вместе с другими алгоритмами в протоколе TLS.

- Алгоритм RSA использует вычисления на основе двух случайных простых чисел, в результате которых пользователи получают открытый и закрытый ключи. Активно используется в схемах цифровой подписи и для защиты программного обеспечения.

4. Алгоритмы кодирования

Алгоритмы сжатия информации делятся на две большие группы:

– Сжатие без потерь – сокращает количество бит на символ, убирая избыточность в тексте. Чаще всего требует использования какого-либо свойства данных или предположений о статистике встречаемости символов в файле.

– Сжатие с потерями – сокращает размер структуры данных, при этом лишаясь части данных, которые не влияют на восприятие всего файла.

Сжатие с потерями является узкоспециализированной областью, и применяется чаще всего для сжатия фото-, видео- и аудиоинформации. Для сжатия общих типов данных, большого количества измерений температуры, влажности, веса, давления, освещённости и других параметров больше приспособлены универсальные алгоритмы сжатия без потерь [4]. Из них можно отметить:

– Алгоритмы Элиаса (гамма-код, омега-код, дельта-код) кодируют положительные целые числа, однако перед таким кодированием возможно установить соответствие любого набора данных кодам Элиаса. При малой дисперсии данных позволяет с высокой эффективностью сжимать числовые или строковые данные.

– Алгоритмы Лемпеля-Зива (LZ77, LZ78, LZW) кодируют последовательности символов с использованием словаря. Чем больше схожих данных передаётся таким методом, тем более эффективно они работают. При нерегулярных данных данные алгоритмы будут очень требовательны к памяти устройства.

– Алгоритм Хаффмана – широко распространённый алгоритм оптимального префиксного кодирования. При известной частотной характеристике данных такой алгоритм будет всегда оптимален, однако для каждой последовательности данных частоту символов потребуется пересчитывать. Также этот алгоритм справляется с длинными повторяющимися последовательностями хуже, чем алгоритмы Лемпеля-Зива.

Заключение

Данная статья посвящена обзору различных алгоритмов шифрования и кодирования, наиболее подходящих для использования в современной мобильной системе сбора данных, с целью дальнейшей реализации алгоритма шифрования для реализованной системы. Описаны основные ограничения мобильных систем и значимые для оценки алгоритмов метрики.

В итоге рассмотрены различные алгоритмы шифрования и кодирования, оценена их применимость в контексте мобильной системы сбора данных. В дальнейшем планируется формирование мобильной

сети, реализация собственного алгоритма на основе изученной информации и анализ его работы на сформированной сети.

Список литературы

1. Рубин Ф. Криптография с секретным ключом / Ф. Рубин. – М. : ДМК Пресс, 2023. – 386 с.
2. Защита от атак типа человек посередине [Электронный ресурс] : научная электронная библиотека. – Режим доступа : <https://cyberleninka.ru/article/n/zaschita-ot-atak-tipa-chelovek-poseredine>
3. Шнайер Б. Прикладная криптография / Б Шнайер. – М. : Триумф, 2012. – 815 с.
4. Пантелеев Е. Р. Алгоритмы сжатия без потерь / Е. Р. Пантелеев, А. Л. Алыкова. – М. : Издательство «Лань», 2021. – 172 с.

Применение сопрограмм для решения прикладных задач

Соловьев А. С.

Студент магистр

Лысачев П. С.

Старший преподаватель

Введение

В современном мире программирования, эффективное управление параллелизмом и асинхронными операциями становится все более важным и актуальным для разработчиков. В этом контексте Kotlin сопрограммы представляют собой мощный инструмент, который обеспечивает удобный и эффективный способ работы с асинхронным кодом.

Kotlin Coroutine – это новый механизм для работы с асинхронным программированием, который позволяет писать сопрограммы (coroutines) в стиле императивного кода, при этом обеспечивая высокую производительность и удобство использования. Они предлагают альтернативный подход к работе с асинхронными операциями по сравнению с традиционными многопоточными моделями.

В данной статье будут рассмотрены основные понятия Kotlin Coroutines: сопрограммы, сопрограммные строители, приостановка и возобновление выполнения, а также проанализированы основные концепции и прикладные задачи, решаемые с помощью Kotlin сопрограмм.

Кроме того, будет проведен анализ, как сопрограммы позволяют управлять выполнением асинхронного кода. Также будут выявлены преимущества сопрограмм по сравнению с традиционными подходами и открыты новые возможности для разработчиков при решении задач конкурентного программирования. На практике будет рассмотрен пример нагрузочного теста, иллюстрирующий сравнение многопоточности, реактивного подхода и Kotlin сопрограмм в прикладной задаче, чтобы продемонстрировать их практическое применение и преимущества.

Таким образом, необходимо углубиться в изучение Kotlin Coroutines и исследовать их потенциал для создания эффективного асинхронного кода на языке программирования Kotlin.

1. Основные понятия

В современном программировании асинхронные операции играют важную роль в разработке систем, особенно это актуально в высоконагруженных приложениях, где требуется выполнение параллельных операций.

Основная функциональность, которую предоставляют Kotlin Coroutines [1] – это возможность приостановить сопрограмму и возобновить ее в будущем. Именно поэтому, имеется возможность запустить код в основном потоке и приостановить его, когда происходит выполнение задачи, например, запрос данных из сервиса по API. После приостановки сопрограмма практически не потребляет ресурсы и освобождает основной поток для использования другими сопрограммами. Это сильно отличается от обычных Thread, где поток можно только заблокировать. Как только данные будут готовы, сопрограмма возобновит выполнение, ожидая освобождения основного потока. После получения потока, она продолжит работу с того момента, на котором была приостановлена.

Таким образом, можно запустить порядка 10 000 сопрограмм на ноутбуке, имеющем 8 Гб оперативной памяти, поставить их на N-секундную паузу, и через N секунд они отработают почти одновременно.

Достоинства классической функциональности Kotlin Coroutines:

- Легковесность: Сопрограммы не создают новые потоки для каждой асинхронной операции, а работают на основе потока, что делает их более эффективными и экономичными по сравнению с традиционными потоками.

- Удобство использования: Код с использованием Kotlin Coroutines выглядит последовательно и понятно, что упрощает его понимание и отладку. В отличие, например, от реактивного подхода, где часто сложно выявлять и устранять ошибки.

- Отсутствие блокировки потоков: Сопрограммы позволяют приостанавливать и возобновлять выполнение кода без блокировки потоков, что способствует более эффективному управлению ресурсами. Во время приостановки поток освобождается, и на нем может работать другая сопрограмма.

- Поддержка отмены операций: Kotlin Coroutines предоставляют механизмы для отмены асинхронных операций, что улучшает управление потоком выполнения.

- Интеграция с другими асинхронными библиотеками: Kotlin Coroutines могут использоваться вместе с другими асинхронными

библиотеками, такими как Retrofit и Room, что делает их универсальным инструментом для работы с асинхронным кодом.

2. Создание и запуск сопрограмм

Для создания и запуска сопрограмм используются сопрограммные строители (coroutine builders), такие как launch, async, runBlocking[2].

launch: Используется для запуска новой сопрограммы, которая работает асинхронно. Он возвращает объект типа Job, который представляет собой задачу сопрограммы. launch не возвращает результат выполнения сопрограммы, поэтому он подходит для выполнения задач без необходимости получения результата.

async: Также запускает новую сопрограмму, но возвращает объект типа Deferred, который представляет собой отложенный результат выполнения сопрограммы. С помощью async можно запустить асинхронную задачу и затем получить результат выполнения этой задачи с помощью функции await.

runBlocking: Используется для запуска корневой сопрограммы, которая блокирует поток выполнения до завершения всех дочерних сопрограмм. Этот строитель часто используется в тестировании или в функциях main, где нужно выполнить асинхронный код синхронно.

3. Прикладная задача

Имеется приложение, в котором одна из функций – это поиск и агрегация данных из нескольких источников. К примеру, происходит поиск билетов на самолет и опрос всех поставщиков таких услуг. Пусть таких источников 5. Соответственно, для каждого пользователя, система должна сделать 5 запросов.

Напишем подобное приложения используя следующие базовые технологии.

- Kotlin. Используется как основной язык программирования.
- Spring framework. Универсальный фреймворк с открытым исходным кодом для Java-платформы. Он используется для быстрого создания веб приложений.
- Gradle. Система автоматической сборки, построенная на принципах Apache Ant и Apache Maven.
- RestTemplate. Библиотека для создания запросов к серверам.

4. Реализация серверной части

В первую очередь стоит реализовать функцию опроса 5 источников на обычных Thread.

Следует отметить, что 1 thread stack составляет примерно 1 mb, и на каждый запрос будет создаваться новый Thread (это условное утверждение)[3].

Также стоит отметить что, одна сопрограмма весит порядка 100 байт.

Появляется понимание: все отлично работает, когда в системе 10, 100, 200 пользователей одновременно. Однако, что произойдет при 1000 пользователей? В этом случае возникает проблема. Для ее решения следует обратиться к многопоточности, используя класс Thread. Это подразумевает создание и управление несколькими потоками выполнения, а также решение проблем синхронизации. Реализация на Thread представлена на листинге 1[4].

Листинг 1

Реализация на Thread-ax

```
fun getTestData(): String {
    val results = mutableListOf<String>()
    val threads = urls.map { url ->
        Thread {
            val content = URL(url).readText()
            synchronized(results) {
                results.add(content)
            }
        }
    }
    threads.forEach { it.start() }
    threads.forEach { it.join() }

    return results.joinToString()
}
```

Следующий шаг реализации будет основан на принципах реактивного программирования.

Реактивное программирование (Reactive Programming)[5] – это парадигма программирования, ориентированная на обработку потоков данных и событий. Основная идея реактивного программирования заключается в том, что программы реагируют на изменения данных или событий в реальном времени, обрабатывая их асинхронно и реагируя на них немедленно. Реализация представлена на листинге 2.

Листинг 2

Реализация на реактивном программировании

```
suspend fun getByFlux(): Flux<String> {
    val flux = Flux.fromIterable(urls)
        .publishOn(Schedulers.parallel())
}
```

Окончание листинга 2

```
.publishOn(Schedulers.boundedElastic())
.map {
    val connection = URL(it).openConnection()
    val input = connection.getInputStream()
    val wrap =
ByteBuffer.wrap(input.readAllBytes())
    val content = wrap.toString()
    content
}
return flux
}
```

Последний шаг – реализация на сопрограммах. В коде на листинге 3 вызывается `asunc` строитель для асинхронного запроса данных с сервера. Дополнительно все обернуто в `runBlocking` строитель сопрограмм, который гарантирует, что все сопрограммы завершат свою работу.

Листинг 3

Реализация на сопрограммах

```
suspend fun getCoroutineTestData(): String {
    return runBlocking {
        return@runBlocking urls.map { url ->
            async {
                URL(url).readText()
            }.await()
        }.joinToString()
    }
}
```

5. Результаты нагрузочного тестирования

Для чистоты эксперимента, после каждого нагрузочного теста системы были перезапущены. Стоит обозначить, существует система для взаимодействия с клиентом, которая делает 5 запросов к другой системе, которая отдает данные через 2 секунды после запроса. Таким образом происходит имитация работы сервера.

Будем использовать следующие параметры: система будет поддерживать 2000 пользователей одновременно, каждый пользователь будет отправлять два запроса в систему каждые 20 секунд. Для каждого пользователя система будет генерировать 5 запросов к другой системе.

Рассмотрим полученные результаты реализаций, описанных в листингах 1-3.

Анализ результатов показал, что реализация на `Thread-ax`, рис. 1, имеет наихудшие показатели из-за фиксированного количества

возможных потоков на процессоре. Таким образом, это самое дорогое решение с точки зрения ресурсов.

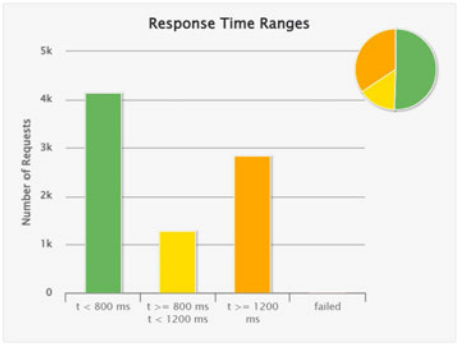


Рис. 1. Результат на Thread 2000 пользователей

Реактивный подход, представленный на рис. 2, позволяет ускориться, однако он не является прикладным решением. Применение данного подхода ограничено, поскольку чаще всего, клиент подписывается на стрим и получает данные по мере их получения от сервера.

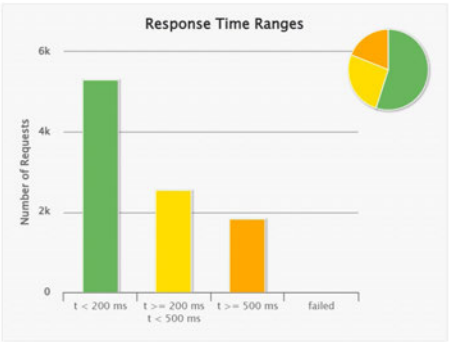


Рис. 2. Результат на реактивном программировании 2000 пользователей

Сопрограммы, на рис. 3, демонстрируют высокую производительность благодаря возможности одновременного запуска большого количества сопрограмм и их почти одновременной обработке. Это решение является наиболее экономичным с точки зрения использования ресурсов.

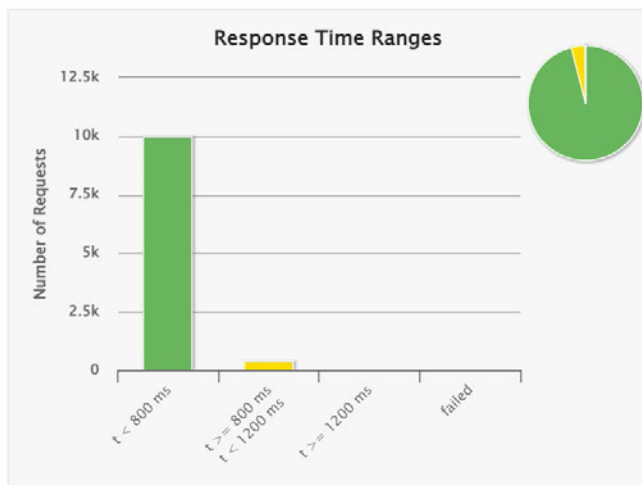


Рис. 3. Результат на сопрограммах 2000 пользователей

Заключение

Использование сопрограмм в разработке программного обеспечения представляет собой эффективный и мощный подход к управлению асинхронными операциями. В данной статье были рассмотрены основные преимущества сопрограмм, такие как улучшенная читаемость кода, упрощенная обработка асинхронных задач, улучшенная отзывчивость приложений и уменьшение нагрузки на систему.

Сопрограммы в Kotlin предоставляют разработчикам удобные инструменты для организации асинхронных операций, а также позволяют элегантно решать проблемы, связанные с многопоточностью и конкурентностью. Благодаря легкости использования и интеграции с существующим кодом, сопрограммы становятся привлекательным выбором для разработчиков, стремящихся к улучшению производительности и отзывчивости своих приложений.

Использование сопрограмм в разработке ПО является ключевым элементом для создания более эффективных и масштабируемых приложений. Для повышения качества и производительность программных продуктов необходимо применять новый механизм Kotlin Coroutine для работы с асинхронным программированием.

Список литературы

1. Электронный каталог Kotlin[Электронный ресурс] : база данных. – Режим доступа : <https://kotlinlang.org/docs/home.html>
2. Marcin, M. Kotlin Corutine deep dive / M. Marcin. – Victory: Leanpub, 2023. – 469 p.
3. Kotlin Coroutines. Часть 1: Первое погружение [Электронный ресурс]: Электронный каталог Habr. – Режим доступа: <https://habr.com/ru/companies/reksoft/articles/802353/>
4. Жемеров, Д. Б. Kotlin в действии / Д. Б. Жемеров, С. С. Исакова. – М. : ДМК-Пресс, 2018. – 402 с.
5. Babic, F Kotlin Coroutines by Tutorials / F. Babic, N. Srivastava – 3rd ed. – NY: Razeware LLC, 2022. – 301 p.

Исследование возможностей и оценка качества распознавания звучащей речи различными нейросетевыми моделями

А. А. Ушакова

Студент магистр

В. В. Гаршина

Доцент

Введение

В современном мире, где цифровые технологии проникают во все сферы нашей жизни, распознавание речи становится ключевой технологической задачей. С возрастанием числа голосовых интерфейсов, аудиоконтента и систем автоматизации, способных взаимодействовать с пользователем на естественном языке, уровень точности и качества распознавания звучащей речи приобретает критическое значение.

Нейросетевые модели играют важную роль в этой области, предоставляя средства для создания точных и эффективных систем распознавания. Однако, существует необходимость в постоянном исследовании и оценке эффективности этих моделей, учитывая разнообразие акустических условий, диалектов, акцентов и других факторов, влияющих на качество распознавания.

В данной работе проводится исследование возможностей различных нейросетевых моделей в этой области, а также предлагаются методы оценки и сравнения их качества.

1. Подходы к автоматическому распознаванию речи

Задача распознавания речи (ASR) решается уже не первый десяток лет, но качественный уровень был достигнут только в последние годы. Можно выделить два основных подхода к распознаванию речи – гибридный (классический) и end-to-end. Особенностью гибридного подхода является отдельное использование акустической и лингвистической модели. Подход end-to-end появился несколько лет назад и основывается на нейросетевых архитектурах, позволяющих переводить непосредственно одну последовательность в другую. А именно, последовательность звуков в последовательность букв.

Общая схема построения системы распознавания речи представлена на рис. 1. Речевой сигнал проходит предобработку и преобразовывается в вектор акустических характеристик. Они подаются на вход декодера, программе которая сопоставляет данные, полученные в ходе распознавания от акустической и языковой моделей, и с помощью их объединения, вычисляет наиболее вероятную последовательность слов, являющуюся окончательным результатом распознавания предложенной речи.



Рис. 1. Архитектура ASR системы

2. Vosk

Vosk наиболее популярный представитель классического подхода. Это встроенный (оффлайн/локальный) механизм преобразования речи в текст с открытым исходным кодом, который может работать с очень низкими задержками. Vosk основан на общей архитектуре DNN-HMM. Глубокая нейронная сеть используется для оценки звука (акустическая оценка), платформы HMM и WFST используются для моделей времени (языковых моделей). Он основан на Kaldi, но предлагает умный, упрощенный и эффективный интерфейс.

В Vosk есть два типа моделей – большие и маленькие. Маленькие модели идеально подходят для некоторых ограниченных задач в мобильных приложениях. Они могут работать на смартфонах Raspberry Pi. Их также рекомендуют для desktop приложений. Маленькая модель обычно имеет размер около 50 МБ и требует около 300 МБ памяти во время выполнения. Большие модели предназначены для высокоточной транскрипции на сервере. Большим моделям требуется до 16 ГБ памяти, поскольку в них применяются передовые алгоритмы искусственного интеллекта.

3. NVIDIA FastConformer-Hybrid Large

NeMo (Neural Modules) – это фреймворк глубокого обучения от NVIDIA, предназначенный для разработки и тренировки различных типов нейронных сетей. Основной целью NeMo является обеспечение простоты использования и масштабируемости при разработке и

экспериментах с моделями. Семейство моделей NeMo включает в себя множество предварительно обученных моделей для различных задач обработки естественного языка, распознавания речи, обработки звука и многих других.

Для тестирования была выбрана модель FastConformer-Hybrid Large. Для русского языка доступна только «большая» версия модели (около 115 млн параметров). Это гибридная модель, обученная на двух лосах: Transducer (по умолчанию) и CTC что позволяет пользователю использовать единую модель, которая работает как модель CTC, так и RNNT.

FastConformer – это оптимизированная версия модели Conformer. На рис. 2 представлена архитектура кодера Conformer.

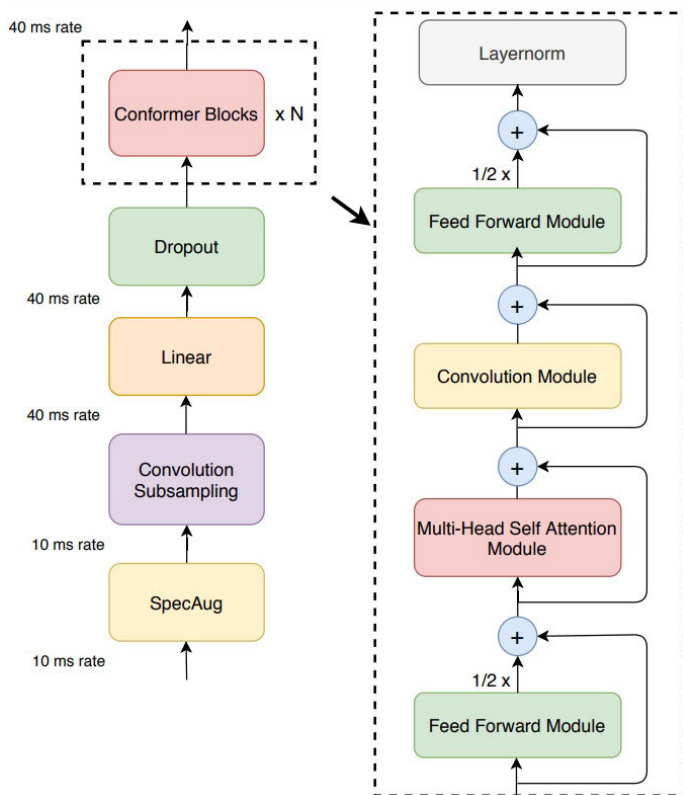


Рис. 2. Архитектура кодера Conformer

NeMo toolkit использовался для обучения модели на протяжении более чем нескольких сотен эпох. Модель обучалась на составном наборе данных (NeMo PnC ASRSET), состоящем из 1840 часов русской речи:

- Golos (1200 часов)
- Sova (310 часов)
- Dusha (200 часов)
- RULS (92,5 часа)
- MCV12 (36,7 часов).

4. Whisper

Whisper – это предобученная модель для автоматического распознавания и перевода речи. Модели Whisper, обученные на 680 тысячах часов размеченных данных, демонстрируют высокую способность обобщать множество наборов данных и доменов без необходимости тонкой настройки. Это кодер-декодер модель на основе Transformer, также называемая sequence-to-sequence модель рис. 3.

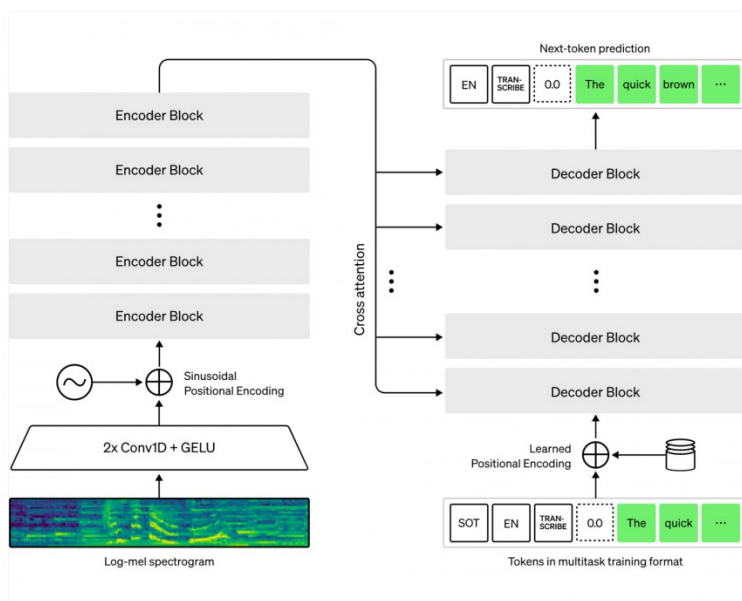


Рис. 3. Архитектура Whisper

Мультиязычные модели были обучены как распознаванию речи, так и переводу речи. Для распознавания речи модель прогнозирует

транскрипцию на том же языке, что и звук. Для перевода речи модель прогнозирует транскрипцию на язык, отличный от аудио.

Для тестирования выбрана мультязычная модель Whisper large-v3. Whisper large-v3 обучена на 1 миллионе часов аудио со слабой разметкой и 4 миллионах часов аудио с псевдомаркировкой, собранных с помощью Whisper large -v2. Модель была обучена в течение 2.0 эпох на этом наборе смешанных данных. Модель large-v3 демонстрирует улучшенную производительность на широком спектре языков, демонстрируя снижение количества ошибок на 10–20 % по сравнению с Whisper large -v2.

5. GigaAM-CTC

Giga Acoustic Model это предобученная на русском языке модель с дообученными состояниями под распознавание речи (GigaAM-CTC) и определение эмоций (GigaAM-Emo) рис. 4. GigaAM-CTC использует архитектуру Conformer в качестве кодера. Такая архитектура обеспечивает баланс между выразительностью модели и ее простотой дообучения в условиях ограниченных вычислительных ресурсов. Модель использует CTC-подход с независимыми выходами для демонстрации возможностей encoder-only модели. Для предобучения применялся wav2vec2.0 подход. Модель дообучалась на смеси данных: около 2000 часов размеченных данных с весом 0.85 и около 5700 часов полу размеченных данных с весом 0.15.

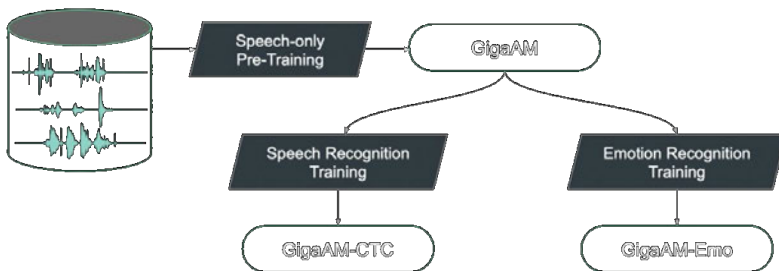


Рис. 4. Giga Acoustic Model

6. Тестирование

Производительность системы ASR измеряется путем сравнения гипотетических транскрипций и эталонных транскрипций. Коэффициент ошибок в словах (WER) является наиболее широко используемым показателем. Две последовательности слов сначала выравниваются с использованием алгоритма выравнивания строк,

основанного на динамическом программировании. После выравнивания определяют количество удалений (D), замен (S) и вставок (I). Удаление, замена и вставка считаются ошибками, а WER рассчитывается по отношению количества ошибок к количеству слов (N) [1].

$$WER = \frac{I + D + S}{N} * 100\% \quad (1)$$

Вычисление метрик будет выполнено с помощью библиотеки JiWER. JiWER – это простой и быстрый пакет Python для оценки системы автоматического распознавания речи.

Перед вычислением WER обычно применяются некоторые преобразования к прогнозу модели и/или базовой истине, чтобы попытаться минимизировать неблагоприятный эффект различий в форматировании между обучающим корпусом модели и тестовыми данными. Этот процесс известен как «нормализация текста».

Whisper имеет собственный нормализатор текста, который применяет стандартные преобразования, такие как строчные буквы и удаление знаков препинания. Для тестирования будет использоваться `whisper normalizer`. Для оценки качества популярных open-source моделей используются следующие валидационные датасеты: Ru Librispeech, Golos Crowd, Golos Farfield, OpenSTT Youtube, OpenSTT Phone Calls, OpenSTT Audiobooks.

Таблица 1

Результаты тестирования WER %

Датасет	Vosk Big	Nvidia RNNT Fastconformer	Whisper Large V3	GigaAM- CTC
Ru Librispeech	11.5	11.2	9.5	5.8
Golos Crowd	3.6	2.7	14.7	3.2
Golos Farfield	6.6	7.1	17.6	5.9
OpenSTT Youtube	16.0	19.4	16.4	13.4
OpenSTT Phone Calls	24.0	28.4	26.8	20.6
OpenSTT Audiobooks	10.1	13.2	13.9	10.6

На 4 из 6 датасетов GigaAM-СТС показывает лучшие результаты среди рассматриваемых моделей. На корпусе Golos Crowd меньший процент ошибок у модели Nvidia RNNT Fastconformer. Лучше всего на OpenSTT Audiobooks показала себя модель Vosk Big. Модели, тестируемые на том же домене, на котором они были обучены, показывают высокое качество, что подчеркивает важность настройки под конкретный домен. Результаты тестирования подтверждают, что предобучение дает значительный прирост в метриках.

Заключение

Данная работа посвящена детальному обзору различных подходов к автоматическому распознаванию речи. Описаны как классические методы, так и современные end-to-end модели, что предоставляет разработчикам широкий выбор инструментов для создания систем распознавания.

Также в работе была проанализирована одна из наиболее распространенных метрик для оценки точности ASR систем. Это помогло объективно сравнить результаты различных моделей. Такая оценка качества играет важную роль при выборе наиболее подходящей модели для конкретного применения.

В ходе исследования было проведено тестирование моделей на различных датасетах. Следует отметить, что в последние годы нейросетевые модели активно развиваются и демонстрируют значительное улучшение качества распознавания русской речи.

Список литературы

1. Speech Recognition [Электронный ресурс]. – Режим доступа: https://speechprocessingbook.aalto.fi/Recognition/Speech_Recognition.html
2. Vosk [Электронный ресурс]. – Режим доступа: <https://alphacephei.com/vosk/>
3. Conformer: Convolution-augmented Transformer for Speech Recognition [Электронный ресурс]. – Режим доступа: <https://arxiv.org/pdf/2005.08100.pdf>
4. Robust Speech Recognition via Large-Scale Weak Supervision [Электронный ресурс]. – Режим доступа: <https://arxiv.org/pdf/2212.04356.pdf>
5. GigaAM: класс открытых моделей для обработки звучащей речи [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/companies/sberdevices/articles/805569/>

Анализ и сравнение алгоритмов машинного зрения для выявления дефектов при производстве печатных плат

А. В. Фуфаева

Студент магистр

И. Ю. Стародубцев

Доцент

Введение

Одной из ключевых задач систем машинного зрения является задача поиска и обнаружения дефектов на разного рода поверхностях или устройствах.

Системы машинного зрения позволяют выполнять визуальный контроль изделий и деталей любой сложности непосредственно на конвейере или производственной линии. Благодаря этому возможно одновременно отслеживать целый ряд характеристик для оценки качества продукции, что способствует значительному снижению выходного брака. При этом время анализа изделия на наличие дефектов сокращается в 10-15 раз, а увеличение точности обнаружения дефектов достигает 30-50%.

1. Применение машинного зрения в производстве печатных плат

Традиционный ручной метод обнаружения дефектов при производстве печатных плат напрямую зависит от человеческого фактора, в следствие этого имеет низкую скорость обнаружения дефектов, низкую эффективность и не может удовлетворить потребности быстрого производства. Поэтому становится важным использовать машины для замены человеческих глаз при измерении и вынесении решений – технология машинного контроля зрения имеет огромную значимость как замена традиционному ручному визуальному контролю.

Машинное зрение может использоваться для инспекции и контроля качества печатных плат. При этом широкое применение для диагностики печатных плат получили бесконтактные оптические методы контроля с использованием современных программно-технических комплексов обработки изображения на основе технологий машинного зрения.

Машинное зрение при контроле печатных плат позволяет обнаружить такие проблемы, как дефекты сварки, отклонение положения компонентов, отсутствие или неправильное расположение компонентов, короткие замыкания и обрывы, хроматическую aberrацию и несоответствие цветов путем анализа и сравнения изображений печатных плат, что позволяет повысить качество и надежность печатной платы.

2. Принцип работы систем обнаружения дефектов

В основе системы – высокоскоростные камеры, которые позволяют обрабатывать входящий видеопоток со средней скоростью равной 5000 кадров в секунду. Они устанавливаются над конвейерной лентой на участках, где необходимо детектировать брак. Камеры подбираются в зависимости от скорости работы конвейера и минимального размера дефектов, которые необходимо детектировать.

На сервер устанавливается соответствующая системе программа, которая анализирует поступающий с камеры видеопоток и с помощью алгоритмов машинного зрения определяет брак и передает данные в информационную систему предприятия или в приложение оператора. Программа оператора отображает событие, регистрирующее брак.

Существующие решения можно разделить на два основных типа по характеру обнаружения дефекта. Первый тип систем способен обнаружить дефекты на поверхностях материалов, таких как древесина, металлы, камень и другие. Второй тип систем предназначен для обнаружения дефектов на технических устройствах, в том числе на печатных платах.

3. Методы обнаружения дефектов

Алгоритмы, используемые в автоматических системах контроля топологии, можно условно разделить на три типа – алгоритмы, использующие эталон, алгоритмы, основанные на контроле проектных норм, и гибридные алгоритмы.

Алгоритмы, основанные на использовании эталона, могут сравнивать непосредственно эталонное и тестовое изображение печатных плат или использовать в качестве эталона набор моделей с заранее определенными информативными признаками.

При сравнении с эталоном возможно, как непосредственное попиксельное сравнение тестового изображения с изображением эталонного образца (вычитание изображений), так и выделение, и последующее сравнение информационных признаков элементов.

На рис. 1 представлен пример с эталонным изображением материала, а также изображения, содержащие дефекты материала.

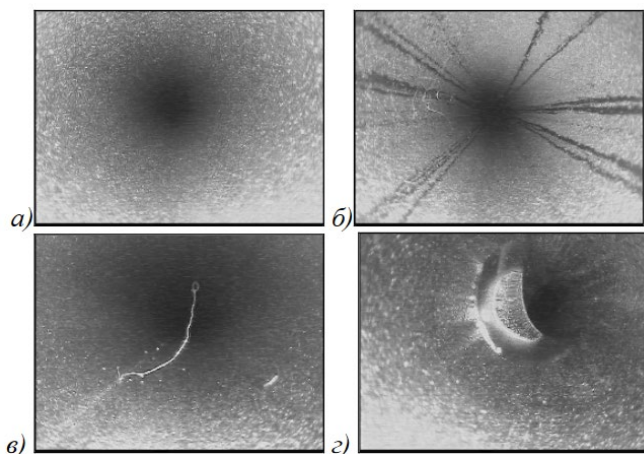


Рис. 1. Изображение эталонной поверхности металлической трубы (а), изображения с дефектами качества (б-г)

Алгоритмы, основанные на использовании модели, такие как синтаксический анализ, алгоритм сравнения графов, алгоритм графов описаний производят сравнение элементов топологии объекта по набору моделей, описывающих эталон. В таких алгоритмах топология определяет взаимное пространственное расположение элементов объекта. Автоматические системы контроля, не использующие эталон, проверяют элементы топологии на соответствие стандартам изделия.



Рис. 2. Эталонное сварное соединение и соединение не соответствующее стандартам изделия

Также могут применяться алгоритмы, основанные на анализе границы элементов. После получения границы производится проверка элемента путем движения вдоль границы с контролем специальных параметров. Изображение, полученное путем применения метода выделения границ, представлено на рис. 3.

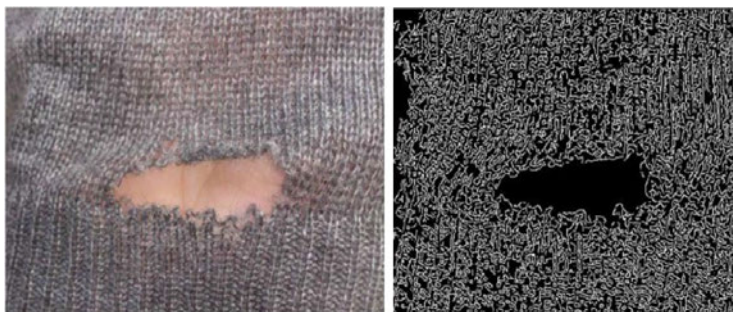


Рис. 3. Применение метода выделения границ

Гибридные алгоритмы контроля основаны и на сравнении с эталоном, и на методах контроля правил топологии элементов.

Главный недостаток данного множества алгоритмов – это зависимость точности локализации дефектов от аккуратности совмещения эталонного и тестового изображений, в том числе от результата операций масштабирования и поворота тестового изображения, его предварительной обработки и бинаризации.

4. Существующие решения

Для сравнения был выбран ряд существующих систем обнаружения дефектов. Сравнение проводилось по следующим критериям:

- Точность распознавания;
- Основное направление (материалы/устройства);
- Типы продукции (количество различных материалов/устройств, на которых система способна обнаружить дефект);
- Стоимость;
- Отзыв пользователей.

По данным критериям было рассмотрено более 30 систем. В качестве примера ниже представлены системы с наилучшими показателями.

1. Система ML Sense

- Заявленная точность распознавания – не менее 98%, при дефектах от 0.1мм
- Основное направление – материалы
- Типы продукции – 3 основных (металлы, волокна, сыпучие материалы)
- Стоимость – 59000 рублей/шт
- Отзыв пользователей – 4.5 из 5

2. Система Easy Scan Smart WEINIG
 - Заявленная точность распознавания – до 99%
 - Основное направление – материалы
 - Типы продукции – только древесина
 - Примерная стоимость – от 190000 руб
 - Отзыв пользователей – 4.7 из 5
 3. Система Statany Technologies
 - Заявленная точность распознавания – максимальная точность по тестовым данным достигается при анализе поверхности древесины и составляет 83%
 - Основное направление – материалы
 - Типы продукции – 3 основных (дерево, сталь, мрамор) и ряд других поверхностей
 - Примерная стоимость – 300000 рублей за стартовую лицензию и доплата за доработки
 - Отзыв пользователей – 4.4 из 5
 4. Автоматическая оптическая инспекция
 - Заявленная точность распознавания – может достигать 99%
 - Основное направление – устройства
 - Типы продукции – печатные платы
 - Примерная стоимость – в среднем 15000000 рублей, в зависимости от количества обнаруживаемых дефектов и мощности
 - Отзыв пользователей – отзывы отсутствуют
- Полученные данные были обработаны и сведены в табл. 1.

Таблица 1

Общие характеристики систем выявления дефектов

Характеристика	Дефекты материалов	Дефекты технических устройств
Количество систем	>90%	<10%
Средняя точность распознавания	80-98% в зависимости от сложности системы	90-95%
Стоимость	от 50.000 рублей	от 500.000 рублей
Возможности	предлагают широкий ряд материалов, но ограниченное количество распознаваемых дефектов	специализируются на конкретном устройстве, однако имеют большое количество распознаваемых дефектов

Характеристика	Дефекты материалов	Дефекты технических устройств
Дополнительные возможности и улучшение качества	возможно улучшение и развитие систем путем добавления новых материалов или дополнения распознаваемых дефектов	возможно улучшение характеристик системы путем увеличения точности распознавания, дефекты устройств в большинстве случаев уникальны – добавление новых устройств проблематично
Применение	большое количество пользователей на разного рода предприятиях	узкий круг пользователей, системы могут разрабатываться специализированно для предприятия
Используемые методы	сравнение по набору моделей, описывающих эталон	сравнение с эталоном, гибридные методы
Общая оценка	большое количество схожих друг с другом систем, стоимость варьируется в широком диапазоне, системы обладают отличной точностью и стремительно развиваются	системы уникальны, обладают хорошей точностью, высокая стоимость обусловлена специализацией систем

Заключение

Применение машинного зрения в производстве позволяет повысить эффективность и точность производства, автоматизировать процессы контроля качества производимой продукции и снизить процент брака.

Алгоритмы машинного зрения позволяют с большой точностью выявлять дефекты в различных ситуациях, однако большая часть существующих на текущий момент на рынке решений направлена на выявление дефектов на поверхностях материалов. Решения,

относящиеся к дефектам технических устройств, способны обнаруживать дефекты узких направлений, ориентированные на запросы заказчика. Реализация таких систем обнаружения дефектов устройств сложнее, чем систем обнаружения дефектов материалов, что значительно повышает стоимость данных систем.

Выполненный в рамках исследования анализ существующих решений показал, что применение гибридных алгоритмов машинного зрения позволяет получить наилучшую точность распознавания дефектов при производстве, сопоставимую с точностью метода сравнения с эталоном, который также находит свое широкое применение в современных системах машинного зрения.

Список литературы

1. Визильтер, Ю. В. Обработка и анализ изображений в задачах машинного зрения / Ю. В. Визильтер, С. Ю. Желтов, А. В. Бондаренко. – М. : Физматкнига, 2010. – 689 с.
2. Бергер Е. Контроль качества печатных плат / Е. Бергер // Технологии в электронной промышленности : ООО «Медиа КиТ», 2012. – № 5. – С. 18–20.
3. Горшенин Г. С. Техническое зрение в машиностроении // Теория и практика современной науки. – Саратов, 2016. – № 2. – С. 139–142.
4. Изосимова, Т. А. Разработка автоматизированной системы управления диагностикой печатных плат на основе машинного зрения / Т. А. Изосимова, М. В. Максимова, О. В. Михайлова // Вестник НГИЭИ. – 2018. – №1. – С. 7-17.

Извлечение данных о технологии проведения химических реакций из неструктурированных PDF-файлов

П. А. Хоменко

Студент магистр

В. В. Гаршина

Доцент

Введение

В современном мире, где информация является одним из самых ценных ресурсов, специалистам из различных областей необходимо иметь доступ к качественным и актуальным данным. В области химии, например, научная литература, такая как журнальные статьи и патенты, является важным источником информации для поиска способов проведения химических реакций и синтетических процедур. Однако заполнение существующих баз данных реакций вручную требует значительных временных и трудовых затрат, особенно учитывая быстрый рост специализированной литературы.

В связи с этим возникает необходимость в разработке автоматизированных методов извлечения информации о реакциях из текстовых данных. Одним из таких методов является интеллектуальный анализ текста (Text Mining), который представляет собой разновидность методов Data Mining и подразумевает извлечение знаний и качественной информации из текстовых массивов.

Целью данного исследования является разработка и реализация автоматизированного метода извлечения информации о химических реакциях из научных статей и патентов с использованием интеллектуального анализа текста. В качестве целевого источника информации рассматриваются журнальные публикации в формате PDF, которые отражают последние достижения в области химии.

1. Схема реакции

Химическая реакция – это процесс, в котором один набор химических веществ превращается в другой. Поэтому схема химической реакции должна быть информативной и отражать это превращение. В ней должны быть указаны исходные вещества, результаты и условия

реакции. Кроме того, схема должна быть лаконичной и удобной для моделей управления данными.

На основе этого принципа была разработана схема, объединяющая все реакции в одном семантическом фрейме. В центре фрейма находится продукт, а вокруг него – восемь связанных с ним ролей реакции. Эти роли, или слоты, необходимо заполнить. Они включают реактивы, тип реакции, катализатор/реагенты, рабочие реагенты, растворитель, температуру, время и выход.

В табл. 1 приведены подробные пояснения к каждой из предопределенных ролей в схеме.

Таблица 1

*Схема реакции с подробным объяснением
каждой конкретной роли*

Роль	Описание
Продукт	Химическое вещество, являющееся конечным результатом (основным продуктом) реакции
Реактант	Химические вещества, которые вносят тяжелые атомы в продукт
Катализатор / Реагенты	Химические вещества, которые участвуют в реакции, но не дают тяжелых атомов (например, кислоты, основания, комплексы металлов)
Реагенты для отработки	Химические вещества, которые используются после для завершения реакции или получения продукта (например, реактивы для тушения, экстракционный растворитель, нейтрализующие кислоты/основания)
Растворитель	Химические вещества, используемые для растворения/смешивания других химических веществ, обычно количественно измеряемые по объему и используемые в сверхстехиометрических количествах (например, вода, толуол и т.д.)
Температура	Температура, при которой происходит реакция
Время	Продолжительность проводимой реакции
Тип реакции	Описание типа химической реакции
Выход	Выход продукта

На рис. 1 ниже приведены примеры реакций, извлеченные из текста с использованием схемы.

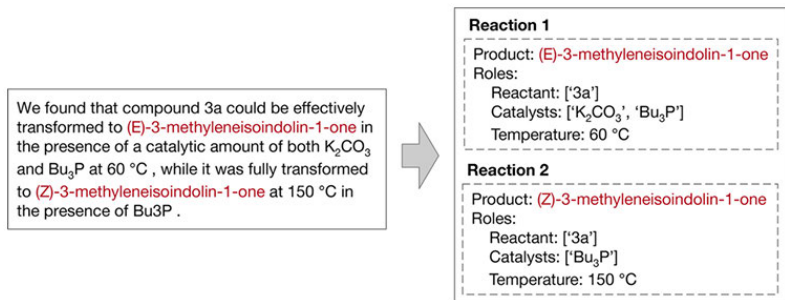


Рис. 1. Пример выделенных реакций с использованием предложенной схемы

2. Данные и аннотация

В данной работе рассматриваются журнальные статьи в качестве основного источника текста, в отличие от большинства существующих инструментов извлечения химической информации, которые используют патентную информацию. Для исследования была взята коллекция из примерно 200 000 статей, опубликованных в различных химических журналах в период с 1906 по 2016 год.

Подробные сведения представлены в табл. 2, где указано количество статей и их источники.

Таблица 2

Количество статей с целевым источником в корпусе

Название журнала	Количество статей
Journal of the American Chemical Society	95 668
The Journal of Organic Chemistry	72 482
Organic Letters	23 631
Journal of Organic Chemistry	295
Organic Process Research & Development	2440

Полученный набор данных, содержит 329 отрывков текста. Каждый из этих отрывков был размечен одной или несколькими реакциями.

Обучающие данные должны содержать тексты (последовательности токенов) и известные целевые метки. Была

использована стандартная схема маркировки для информации из химической сферы, где B-{type} указывает начало определенного типа объекта (например, продукт, реагенты, растворитель) и I-{type} означает внутреннюю часть объекта.

Извлечение ролей реакции подразумевает наличие нескольких столбцов меток, каждый из которых соответствует одному продукту.

#	passage=10.1021/ja00020a078-5			segment=1
Reaction	0	0	0	
of	0	0	0	
diphenylacetylene		B-Reactants	B-Reactants	B-Reactants
with	0	0	0	
complex	0	0	0	
19A	B-Reactants	B-Reactants	0	
led	0	0	0	

Рис. 2. Разметка в стиле CoNLL и маркировка ролей в реакции.

Чтобы разделить данные на обучающий, валидационный и тестовый наборы, было выбрано соотношение 8:1:1. Более подробно о сформированных наборах можно узнать из табл. 3.

Таблица 3

Статистика данных аннотированного корпуса реакций, включая количество текстовых отрывков, реакций, отрывков с несколькими реакциями и пар отношений «продукт-роль в реакции»

	Текстовые отрывки	Реакции	Несколько реакций	Пары отношений
Train	251	599	159	2457
Development	41	96	22	482
Test	37	111	22	469

Важно подчеркнуть, что недостаток обучающих данных может серьёзно усложнить процесс обучения высокопроизводительной модели.

3. Программная реализация

Извлечение реакций из текста – это задача, которая заключается в предсказании структуры. На вход поступает последовательность лексем, а на выходе – структуры реакций, состоящие из пар отношений «продукт-роль».

Для решения этой задачи необходимо использовать принцип двухэтапного конвейера. Он объединяет модуль извлечения продукта и модуль маркировки роли реакции.

На первом этапе модуль извлечения продукта находит все возможные сущности продукта в тексте. Затем модуль маркировки роли реакции извлекает связанные элементы для каждой сущности продукта. Эти элементы соответствуют различным ролям реакции, представленным в контексте. В результате получается окончательная структура реакции. Каждый модуль обучается независимо.

Для аннотирования наборов данных о химических реакциях требуются значительные усилия и специальные знания. Поэтому необходимо снизить зависимость от большого количества размеченных данных, которые обычно требуются для контролируемого обучения глубоких нейронных моделей.

Для этого была выбрана стратегия дополнительного обучения ChemBERT v3.0 – предварительно обученный текстовый кодер на основе Transformer (Bidirectional Encoder Representations from Transformers). Он был обучен на молекулярных данных, чтобы обнаруживать объединённые молекулярные представления. Исходный ChemBERT был соединён с декодерами для конкретных задач, таких как детектирование и аннотирование химических сущностей. Кодировщик был настроен с использованием ограниченного количества информации для каждой конечной задачи.

На рис. 3 показано, как формулируется задача маркировки ролей.

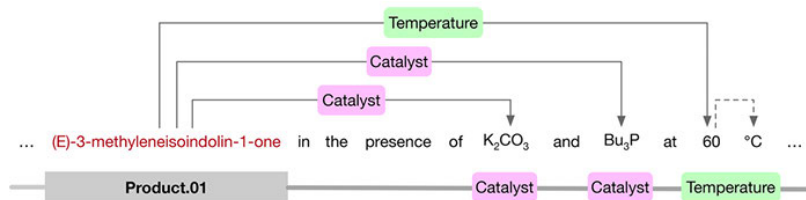


Рис. 3. Выделение роли реакции как задача извлечения отношений

4. Оценка результатов модели

Для тестирования полученных результатов был использован небольшой кусок текста. Это позволило проверить, насколько успешно модель справляется с задачей извлечения реакций из текстовых данных.

We were excited to find that , with 2.0 equiv of copper acetate and DMSO as the solvent , 2-(3-(methylthio)naphthalen-2-yl)pyridine was obtained as a single product in 89 % yield at 125 °C (Table 1) .

Extract

We were excited to find that , with 2.0 equiv of copper acetate and DMSO as the solvent ,

Catalyst

Solvent

2-(3-(methylthio)naphthalen-2-yl)pyridine was obtained as a single product in 89 % yield at 125 °C (Table 1) .

Product.01

Yield

Temperature

Рис. 4. Результат работы дообученной модели на тексте, используемого для экспертной оценки.

Оценка производительности моделей по маркированию ролей и продуктов реакции проводилась на отдельных тестовых наборах с использованием стандартных метрик: Precision, Recall и f_1 – score (F1).

$$precision = \frac{true\_positives}{(true\_positives + false\_positives)} . \quad (1)$$

$$recall = \frac{true\_positives}{(true\_positives + false\_negatives)} . \quad (2)$$

$$f_1 = 2 \cdot \frac{precision \cdot recall}{(precision + recall)} . \quad (3)$$

Были получены следующие результаты:

Таблица 4

Результаты оценки моделей по производительность извлечения продуктов с использованием стандартных меток

	precision(%)	recall(%)	f1(%)
BERT	54.3	49.1	51.6
BioBERT	78.8	56.8	66.0
ChemBERT	76.4	61.3	68.0
NEW_ChemBERT	84.6	69.4	76.2

Таблица 5

Результаты оценки моделей по производительность извлечения ролей с использованием стандартных меток

	precision(%)	recall(%)	f1(%)
BERT	69.2	69.2	69.2
BioBERT	73.3	75.5	74.3
ChemBERT	77.0	76.4	76.7
NEW_ChemBERT	79.3	78.1	78.7

На уровне предложений ChemBERT достиг значительного превосходства над BERT, а адаптивный к задачам NEW_ChemBERT дает дополнительные 2% улучшения в метрике F1- score (F1).

5. Формирование онтологии

Онтология – это формальная спецификация концептуализации предметной области. В контексте химии онтология может использоваться для представления химических реакций, реагентов и условий проведения.

Необходимо разработать онтологию, которая будет описывать основные аспекты химических реакций, выявленные на предыдущем этапе. Для этого использовалась библиотека Owlready2, которая позволяет легко создавать и манипулировать онтологиями на основе OWL (Web Ontology Language).

Были определены следующие классы онтологии:

- Chemical_Reaction – класс, представляющий химическую реакцию.
- Reactants – класс, представляющий реагенты, участвующие в реакции.
- Product – класс, представляющий продукт реакции.
- Catalyst_Reagents – класс, представляющий катализаторы, участвующие в реакции.
- Workup_Reagents – класс, представляющий рабочие реагенты, используемые для обработки продукта реакции.
- Solvent – класс, представляющий растворитель, используемый в реакции.
- Yield – класс, представляющий выход реакции.
- Time – класс, представляющий время проведения реакции.
- Temperature – класс, представляющий температуру, при которой проводится реакция.

Следующим шагом было необходимо определить свойства для каждого класса. Например, класс `Chemical_Reaction` будет иметь свойство «`ChemicalReaction_Reactants`», которое будет указывать на список реагентов, участвующих в реакции. Класс `Reactants` будет иметь свойство «`is_involved_in`», которое будет указывать на список реакций, в которых участвует данный реагент.

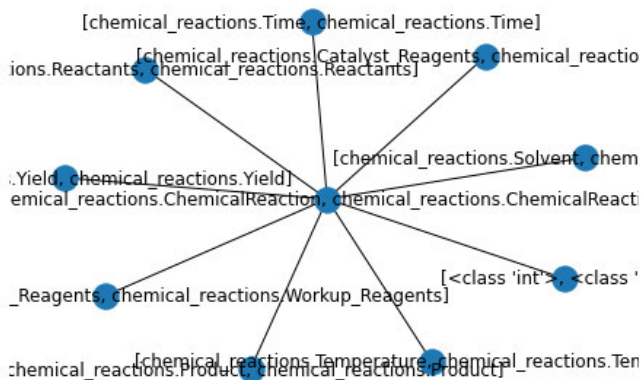


Рис. 5. Графовое представление разработанной онтологии

Protege – это инструмент для создания, редактирования и управления онтологиями. Protege позволяет создавать и редактировать классы, свойства, отношения и экземпляры онтологий, а также детально ознакомиться с разработанной онтологией.

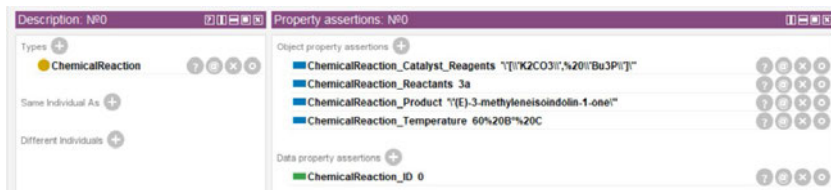


Рис. 6. Экземпляр класса `Chemical_Reaction` онтологии в Protégé

Заключение

В данной статье проводится разработка и реализация автоматизированного метода извлечения информации о химических реакциях из научных статей и патентов с использованием интеллектуального анализа текста. Для решения задачи обнаружения молекулярных наименований в формате номенклатуры ИЮПАК и

маркировки ролей сущностей в рамках рассматриваемой реакции была применена предварительно обученная модель ChemBERT.

Данная модель была выбрана благодаря её универсального формата разметки текстового корпуса в стиле CoNLL. В процессе экспертной оценки результатов разбиения на токены и маркировки ролей лексических единиц было обнаружено разделение на составные части названий молекул. Однако эти элементы всё ещё идентифицируются как часть реакций.

Для создания онтологии, описывающей основные аспекты химических реакций, была использована библиотека Owlready2, которая позволяет создавать и манипулировать онтологиями на основе OWL.

Список литературы

1. The CHEMDNER corpus of chemicals and drugs and its annotation principles / M. Krallinger [et al.] // Journal of Cheminformatics. – 2015. – № 1. – С. 1-17.
2. Learning continuous and data-driven molecular descriptors by translating equivalent chemical representations. / R. Winter [et al.] // Chemical Science. – 2019. – № 10. – С. 1692-1701.
3. ChemDataExtractor: A Toolkit for Automated Extraction of Chemical Information from the Scientific Literature [Электронный ресурс]: набор инструментов. – Режим доступа : <http://chemdataextractor.org/docs/intro>
4. ChemicalTagger: A tool for semantic text-mining in chemistry. / L. Hawizy [et al.] // J. Cheminf. – 2021. – № 3. – С. 17-31.
5. Prediction of organic reaction outcomes using machine learning. / C. W. Coley [et al.] // ACS Cent. Sci. – 2017. – № 3. – С. 434–443.
6. Using machine learning to predict suitable conditions for organic reactions. / H. Gao [et al.] // ACS Cent. Sci. – 2018. – № 4. – С. 1465-1476.
7. Automated extraction of chemical synthesis actions from experimental procedures. / A. C. Vaucher [et al.] // Nat. Commun. – 2020. – № 11. – С. 1-11.

Разработка клиент-серверной системы анализа аудиоданных

С. А. Шипилов

Студент магистр

В. В. Твердохлебов

Студент магистр

А. В. Максимов

Старший преподаватель

Введение

В настоящее время анализ кашля может быть очень утомительным и длительным процессом для врачей, особенно при хроническом кашле. Это связано с тем, что кашель может быть вызван широким спектром основных состояний, от простой аллергии до более серьезных респираторных заболеваний.

У каждого человека присутствуют некие индивидуальные черты голоса. Выделяют две основные классификации кашля – сухой и влажный. В зависимости от стадии и типа заболевания можно определить, к какому классу относится кашель. Кашли для каждого индивидуальны. И даже кашли одного и того же человека сильно отличаются друг от друга в зависимости от ситуации.

Цель работы – разработка клиент – серверной системы анализа респираторной активности.

1. Функционал системы

Для хранения аудиофайлов обычно используют хранилища и взаимодействуют через FTP – протокол для их передачи. Разрабатываемая система должна выполнять следующие задачи.

- Хранить данные о пациентах;
- Хранить у каждого пациента соответствующие обследования;
- Возможность добавлять файлы на сервер;
- Реализация аутентификации и авторизации;
- Возможность создать отчет на основе данных.

Система должна иметь возможность проверять роли вошедших пользователей. Для проверки подлинности каждому пользователю, в данном случае пользователем является врач, будет выдаваться токен,

который определяет, что именно этот пользователь с определенным идентификатором может сделать какие – то действия в системе. Основным стеком данной системы является язык программирования Python и веб – фреймворк Django.

2. Основные сущности базы данных

Основными сущностями базы данных являются таблицы (рис. 1):

- Доктора (Doctor);
- Пациента (Patient);
- Обследования (Examination);
- Аудиофайлов обследования (FileOn);
- Общего отчета по обследованиям (Report).

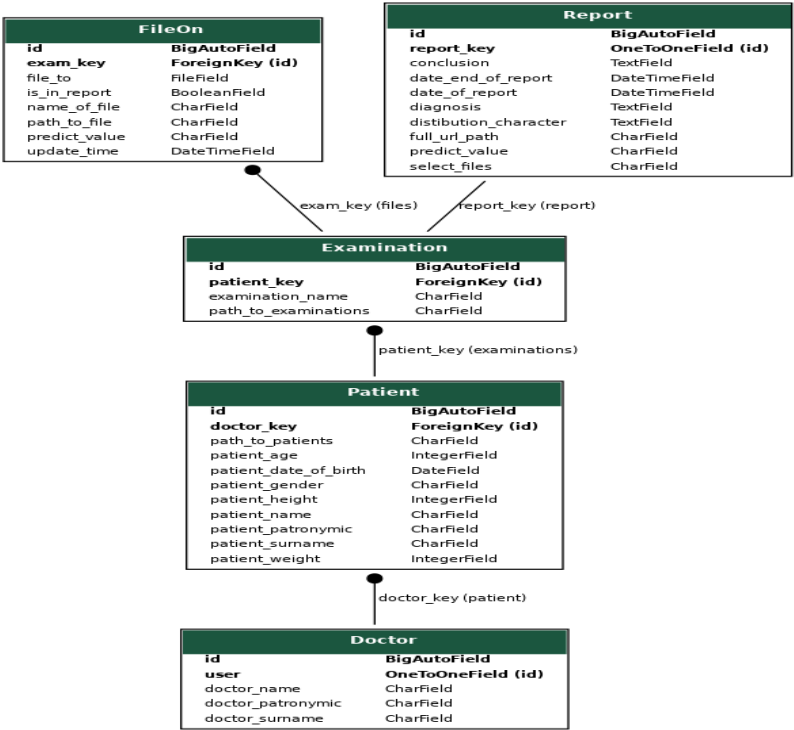


Рис. 1. Схема основных сущностей базы данных

Зашедший в систему врач должен иметь возможность просмотреть своих пациентов и их обследования. После того, как в обследовании будут находиться файлы, у врача должна быть возможность просмотреть

аудиофайл и его частотные характеристики для дальнейшего анализа. На основе полученных данных врач может написать отчет о том, какие препараты или лечение необходимо выписать пациенту на основе обследования. Вывод аудиоданных пациента представлен на рис. 2.

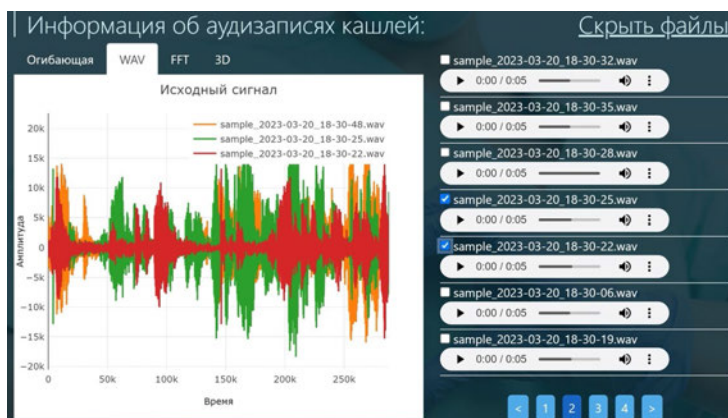


Рис. 2. Функционал для анализа аудиоданных

3. Моделирование API для внешних приложений

Одной из основных задач является отправка файлов на сервер. Пациенту дается прибор, записывающий его кашли. Нужно отправить этот файл на сервер тому пациенту, за которым закреплено обследование. Для этого необходимо создать для сервиса API, которое будет принимать файл в качестве параметра файл и дополнительные параметры для указания того, на какое обследование прикреплен этот файл. И после этого врач сможет проанализировать данные и сделать вывод. Основными API – запросами являются такие действия, как:

- Методы авторизации и аутентификации;
- Загрузка аудиофайла с мобильного приложения;
- Просмотр пациентов зашедшего врача.

4. Фоновые задачи

Фоновые задачи полезны тогда, когда необходимо асинхронно выполнять какие-то однотипные действия. В качестве пакета для выполнения задач был выбран Python – пакет Celery [1].

Данный пакет предназначен для асинхронной и параллельной обработки большого количества задач, что обеспечивает высокую пропускную способность и повышенную масштабируемость. Очередь задач Celery состоит из нескольких компонентов:

- Брокер сообщений;
- Задачи;
- Рабочие узлы;
- Результаты;
- Мониторинг.

Чтобы использовать Celery, нужно определить задачи, которые необходимо выполнить, и зарегистрировать их в приложении. Затем отправить задачу в очередь и она будет выполнена асинхронно одним из рабочих узлов.

Пусть дана задача предсказать с помощью свёрточной нейронной сети, какой кашель может быть в загруженном аудиофайле. После того, как файл загружен, нужно добавить в стек определенную задачу, которая принимает выходные параметры: ID экземпляра объекта файла, путь к загруженному файлу и его названия для инициализации

Пример функции для создания фоновой задачи приведен в листинге.

Листинг

Фоновая задача на предсказание кашля у аудиофайла

```
@shared_task
def predict_cough(fileon_id, path_to_predict, filename):
    wavelet_solo(path_to_predict)
    result = predict(filename)
    fileon = FileOn.objects.get(id=fileon_id)
    return fileon
```

Данная задача создает по аудиофайлу вейвлет – скалограмму и передает ее нейронной сети для прогнозирования значений. После успешного выполнения задачи результаты записываются в базу данных в соответствующий экземпляр, принадлежащий аудиофайлу.

Результат фоновой задачи представлен на рис. 3.

```
1/1 [=====] - ETA: 0ser-194]
1/1 [=====] - 41s 41s/step4]
[2023-05-16 18:53:53,304: WARNING/ForkPoolWorker-194] [[0.613839 0.386161]]
[2023-05-16 18:53:53,310: WARNING/ForkPoolWorker-194] Cough
[2023-05-16 18:53:53,311: WARNING/ForkPoolWorker-194] That's a cough
1/1 [=====] - ETA: 0ser-193]
1/1 [=====] - 6s 6s/step193]
[2023-05-16 18:53:54,186: WARNING/ForkPoolWorker-193] [[0.75247717 0.24752288]]
[2023-05-16 18:53:54,190: WARNING/ForkPoolWorker-193] Cough
[2023-05-16 18:53:54,191: WARNING/ForkPoolWorker-193] That's a cough
1/1 [=====] - ETA: 0ser-193]
1/1 [=====] - 15 618ms/step]
[2023-05-16 18:53:55,053: WARNING/ForkPoolWorker-193] [[9.9999917e-01 8.5791532e-07 1.9628511e-14 2.0190147e-14]]
[2023-05-16 18:53:55,054: WARNING/ForkPoolWorker-193] Кашля
1/1 [=====] - ETA: 0ser-194]
1/1 [=====] - 3s 3s/step194]
[2023-05-16 18:53:56,059: WARNING/ForkPoolWorker-194] [[9.99999094e-01 9.0302146e-06 1.8853929e-12 1.9110406e-12]]
[2023-05-16 18:53:56,059: WARNING/ForkPoolWorker-194] Кашля
[2023-05-16 18:53:57,061: INFO/ForkPoolWorker-193] Task app.tasks.predict_cough[7528023c-0831-4654-8701-ba7629894fb4] succeeded in 150.47092407181628s: 2804
[2023-05-16 18:53:57,366: INFO/ForkPoolWorker-194] Task app.tasks.predict_cough[9d6efdf8-718f-4f85-9283-35196f07e18f] succeeded in 150.64317092392594s: 2803
```

Рис. 3. Пример результатов фоновой задачи распознавания кашля

Заключение

Данная статья посвящена разработке клиент-серверной системы для анализа респираторной активности, хранения и обработки аудиофайлов кашлей. В рамках данной работы были получены следующие результаты:

- Возможность проанализировать спектр файла;
- Реализована система авторизации и аутентификации;
- Создан интерфейс API для отправки файла на сервер и последующего анализа;
- Возможность вывода спектра графиков на странице;
- Встроена нейронная сеть для предсказания кашлей.

Список литературы

1. Документация по инструменту Celery [Электронный ресурс] : Celery – Distributed Task Queue. – Режим доступа : <https://docs.celeryq.dev/en/stable/index.html>
2. Multi-Resolution Analysis [Электронный ресурс] : Wavelets – Режим доступа : <https://www5.informatik.uni-erlangen.de/fileadmin/lectures/2012w/Lecture.2012w.IntroPR/wavelets.pdf>
3. Меле А. Django 2 в примерах / А. Меле. – М. : ДМК Пресс, 2019. – 408с.

Прогнозирование сахарного диабета у пациентов с использованием аппарата цепей Маркова

А. В. Шипилова

Студент магистр

М. Г. Матвеев

Профессор

Введение

Прогнозирование – это важная задача, которая встречается во множестве предметных областей. Примерами таких задач могут быть прогноз изменений курса валют в экономике, прогнозирование чрезвычайных ситуаций, прогноз течения некоторого заболевания в медицине и т.д.

В области медицины для врача-эксперта существенно важно достоверно прогнозировать развитие того или иного заболевания. Данная работа посвящена проблеме поиска адекватной модели, которая позволит получить прогноз течения заболевания сахарный диабет. Для оценки состояния пациента предлагается анализировать динамику уровня глюкозы в крови с течением времени, вместо традиционного подхода на основе исследования гликированного гемоглобина.

В качестве инструмента для прогнозирования сахарного диабета у пациентов предложена модель на основе цепей Маркова с нечеткими состояниями. Такая модель эффективна в условиях неустойчивости временного ряда к внешним воздействиям (случайным выбросам и помехам в данных), а также позволяет учесть индивидуальный характер течения диабета для разных пациентов.

1. Актуальность проблемы исследования

Сахарный диабет (СД) – это совокупность заболеваний обмена веществ, при которых характерно постоянное повышение уровня сахара в крови, вызванное нарушением выработки инсулина, его действия на клетки или комбинацией этих причин. На фоне диабета может возникать поражение, нарушению функций и недостаточность разных органов, в частности глаз, почек, нервной системы, сердца и сосудов [1].

СД – достаточно распространенное заболевание на территории РФ: согласно [2], по состоянию на 01.01.2023, общая численность пациентов

с диагнозом СД, состоящих на диспансерном учете, составила 4 962 762 (3,31% населения РФ). При этом, авторами исследования [3] отмечается динамика роста заболеваемости на протяжении последних 12 лет. Такая статистика и динамика заболеваемости подтверждает актуальность поставленной в работе задачи прогнозирования СД у пациентов.

2. Используемый метод оценки состояния больного сахарным диабетом

Нынешний подход диагностики и прогнозирования сахарного диабета у пациентов заключается в анализе такого показателя как гликированный гемоглобин (гликогемоглобин) [3]. Гликогемоглобин – это биохимический показатель, отражающий усредненное содержание глюкозы в крови за предшествующие 3 месяца. На основе анализа этого показателя врачами-экспертами оценивается состояние пациента и эффективность программы подобранного лечения.

С точки зрения качества прогнозирования дальнейшего течения болезни, мы не можем считать методику анализа показателя гликированный гемоглобин точной и эффективной по следующим причинам:

- усредненное значение, которое представляет собой гликогемоглобин, не может давать оценку тренда – ухудшилось или улучшилось состояние пациента;
- для объективного анализа вместе с гликированным гемоглобином необходимо проведение дополнительных анализов – сахар в крови натощак и глюкозотолерантный тест;
- выполнение исследования гликогемоглобина может занимать до пяти суток [3], что существенно влияет на адекватность результатов оценки заболевания.

В связи с неэффективностью вышеописанного подхода, предлагается исследовать изменения уровня глюкозы в крови пациента с течением времени. Выполнение этого анализа не требует больших временных затрат, т.е. измерения могут проводиться ежедневно. Наблюдения уровня глюкозы за определенный период позволяют получить адекватный прогноз и оценить тенденцию развития заболевания.

3. Обоснование выбора модели прогнозирования сахарного диабета у пациентов

Наблюдения уровня глюкозы у пациента во времени образуют временной ряд. В ходе работы был выполнен обзор известных моделей временных рядов:

- модель авторегрессии типа ARIMA;

- рекуррентные нейронные сети (РНС);
- цепи Маркова.

Модели типа ARIMA могут использоваться в условиях нестационарности, но они имеют существенный недостаток – стремление получить наиболее точное числовое значение прогноза, что значительно повышает сложность задачи.

В РНС для моделирования временных рядов соединения узлов образуют направленный граф вдоль временного ряда. Такая РНС дает возможность исследовать динамическое поведение ряда во времени [4]. Для точного прогноза с использованием РНС требуется много данных и времени на ее обучение. Кроме того, РНС могут работать только с числовыми прогнозами, что не дает возможность оценивать качественные состояния пациента – ухудшение или улучшение здоровья.

Эффективным инструментом в области исследования временных рядов являются цепи Маркова. Цепь Маркова (ЦМ) – последовательность случайных событий с конечным или счётным числом исходов, характеризующаяся тем, что при фиксированном настоящем будущее независимо от прошлого [5].

Классическая ЦМ способна работать только с наблюдаемыми четкими состояниями. При прогнозировании сахарного диабета на основе исследования уровня глюкозы в крови мы имеем дело с качественными нечеткими состояниями – увеличился или уменьшился этот показатель. Для возможности качественной оценки предлагается использовать модель ЦМ с нечеткими состояниями. Это позволит врачу-эксперту адекватно оценивать полученные данные наблюдений уровня глюкозы в крови – ухудшилось или улучшилось состояние пациента за определенный период времени.

4. Задача и методика эксперимента

В рамках эксперимента предлагается ежедневно проводить измерения уровня глюкозы в крови пациента. Для пороговых значений содержания уровня глюкозы построен график функции принадлежности, рис. 1.

На приведенном графике (рис. 1) 3 ммоль/л – минимально возможный уровень содержания глюкозы в крови человека, 15 ммоль/л – максимально возможный уровень. В зависимости от содержания уровня глюкозы будем давать качественную оценку степени тяжести диабета, используя три нечетких состояния:

- З – пациент здоров (зеленая линия на графике);
- С – средняя степень тяжести (серая линия на графике);
- Т – тяжелая степень заболевания (красная линия на графике).

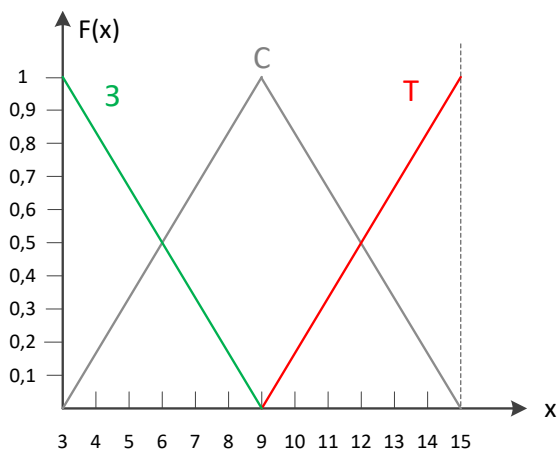


Рис. 1. График функции принадлежности для пороговых значений содержания уровня глюкозы в крови

Ежедневные измерения уровня глюкозы в крови пациента предлагается выполнять на протяжении двух недель (14-ти дней). В первую неделю будет получен вектор наблюдений из семи значений. Начиная с 8-го дня, на каждый день наблюдений будем получать новый вектор из семи значений по методу скользящего среднего – со сдвигом на одно значение вправо.

По результатам численной апробации модели на основе ЦМ с нечеткими состояниями получим следующие значения стационарных вероятностей системы:

- $p_1^i(З)$ – вероятность нахождения пациента в состоянии З (здоров);
- $p_2^i(С)$ – вероятность нахождения пациента в состоянии С (средняя степень тяжести заболевания);
- $p_3^i(Т)$ – вероятность нахождения пациента в состоянии Т (тяжелое течение болезни).

5. Численная апробация модели и анализ результатов

Вектор наблюдений уровня глюкозы за первые 7 дней:

$$x_7 = (4.6; 7.3; 6.4; 7; 10.5; 12; 8). \quad (1)$$

В любой Марковской цепи, в т. ч. нечеткой, каждый переход из одного состояния в другое может быть описан стохастической матрицей вероятностей переходов размерностью $s \times s$ вида:

$$M^t = \|p_{ij}\|, \quad (2)$$

где s – число состояний; p_{ij} – вероятности перехода системы из состояния i в j .

Для вектора (1) получена следующая стохастическая матрица:

$$M = \begin{pmatrix} 0,002 & 0,005 & 0,025 \\ 0,027 & 0,1 & 0,02 \\ 0,025 & 0,09 & 0,01 \end{pmatrix}. \quad (3)$$

Для полученной матрицы (3) требуется выполнить нормировку для удовлетворения вероятностей (элементов матрицы) обязательному условию:

$$\forall i = \left(\sum_j p_{ij} = 1 \right). \quad (4)$$

После нормировки получена новая матрица (4):

$$M_{\text{норм}} = \begin{pmatrix} 0,063 & 0,156 & 0,781 \\ 0,184 & 0,68 & 0,136 \\ 0,2 & 0,72 & 0,08 \end{pmatrix}. \quad (5)$$

Полученные значения теперь могут быть использованы в качестве компонентов матрицы вероятностей переходов, что позволяет рассматривать их как вероятности для случайных нечетких событий – переход из нечеткого состояния $\tilde{x}_i$ в нечеткое состояние $\tilde{x}_j$.

Чтобы найти вероятности стационарных состояний системы, необходимо решить следующую систему однородных линейных уравнений:

$$\begin{cases} (0,063-1)p_1^t + 0,184p_2^t + 0,2p_3^t = 0 \\ 0,156p_1^t + (0,68-1)p_2^t + 0,72p_3^t = 0. \\ p_1^t + p_2^t + p_3^t = 0 \end{cases} \quad (6)$$

После решения системы (6) найдены вероятности стационарных состояний системы:

$$\begin{cases} p_1^t = 0,167 \\ p_2^t = 0,602. \\ p_3^t = 0,231 \end{cases} \quad (7)$$

В соответствии с лингвистической шкалой, описывающие возможные нечеткие состояния, полученные результаты мы можем

интерпретировать следующим образом: пациент, с вероятностью 0,167 находится в состоянии З (здоров), с вероятностью 0,602 – в состоянии С (средней степени тяжести) и с вероятностью 0,231 – в состоянии Т (тяжелое течение заболевания). Т.е. с большей степенью вероятности мы можем утверждать, что наш пациент со значениями уровня глюкозы, описанными в (1), имеет заболевание диабетом средней тяжести, что соответствует состоянию С. При этом, учитывая, что вероятность для состояния Т больше, чем для состояния З, можем говорить, что тренд идет в сторону ухудшения состояния с дальнейшим переходом в тяжелое течение болезни.

За вторую неделю эксперимента получены еще 7 векторов наблюдений по методу скользящего среднего – каждый новый вектор получаем со сдвигом на одно значение вправо. Например, результаты на 8-й день наблюдений:

$$x_8 = (7.3; 6.4; 7; 10.5; 12; 8; 7.5). \quad (8)$$

Результаты на 14-й день наблюдений:

$$x_{14} = (7.5; 7; 8; 8.2; 8.5; 11.5; 12). \quad (9)$$

Для каждого вектора наблюдений аналогично были найдены векторы вероятностей стационарных состояний. Результаты этих вычислений описаны в табл. 1.

Картина результатов прогнозирования в табл. 1 говорит о том, что на каждый день наблюдений уровня глюкозы в крови, пациент находится в средней степени тяжести заболевания, поскольку $p'_2(C)$ имеет наибольшее значение (более 0,6 на каждый день). В то же время, мы можем говорить о том, что состояние пациента ухудшается с каждым днем наблюдений, так как вероятность того, что он здоров уменьшается ($p'_1(C) \rightarrow 0$), а вероятность тяжелого течения болезни увеличивается ($p'_3(T) \rightarrow \infty$).

В прикладной медицине, основываясь на этих результатах прогноза, мы можем судить о том, что применяемая терапия неэффективна для пациента и стоит ее в дальнейшем скорректировать, например, подобрать другие препараты.

Заключение

Данная статья посвящена подбору модели прогнозирования сахарного диабета у пациентов. Предложен метод оценки сахарного диабета на основе анализа уровня глюкозы в крови пациента.

Таблица 1

Результаты расчета вероятностей стационарных состояний

День наблюдений	Вектор наблюдений	P'		
		$P'_1(3)$	$P'_2(C)$	$P'_3(T)$
7	x_7	0,167	0,602	0,231
8	x_8	0,167	0,633	0,2
9	x_9	0,165	0,633	0,202
10	x_{10}	0,149	0,635	0,216
11	x_{11}	0,132	0,641	0,227
12	x_{12}	0,13	0,643	0,227
13	x_{13}	0,128	0,644	0,228
14	x_{14}	0,127	0,644	0,229

В качестве модели для анализа временных рядов со значениями уровня глюкозы использована модель на основе аппарата цепей Маркова с нечеткими состояниями. Численная апробация модели в практической части работы показала свою эффективность и адекватность с точки зрения качества прогноза. Использование нечетких качественных состояний дает возможность оценивать не только текущее состояние пациента в нечетких терминах (здоров, средняя тяжесть заболевания, тяжелое течение болезни), но и прогнозировать динамику течения заболевания – пациент выздоравливает, либо же его состояние ухудшается. Это дает возможность врачу-эксперту адекватно оценить характер течения заболевания и принять решение по необходимой терапии.

Очевидно, что предложенная модель оценки и прогнозирования сахарного диабета имеет наибольшую эффективность по сравнению с традиционной методикой исследования уровня гликированного гемоглобина. Ведь анализ усредненных значений не может дать представления о тенденции развития диабета с течением времени.

Список литературы

1. Фадеев П. А. Сахарный диабет / П. А. Фадеев. – М.: ООО «Издательство Оникс»: ООО «Издательство «Мир и Образование», 2009 – 208 с.
2. Королев В. А. Гликированный гемоглобин: методические аспекты применения / В. А. Королев // Военная медицина. – 2011. – №3. – С.92-98.

3. Единая платформа ЭНЦ [Электронный ресурс]: Эпидемиология в РФ. – Режим доступа: https://diabet.endocrincentr.ru/o_platforme/epidemiologiya_saharnogo_diabeta_v_rf

4. Гафаров Ф.М. Искусственные нейронные сети и приложения: учеб. пособие / Ф.М. Гафаров, А.Ф. Галимянов. – Казань: Изд-во Казан. ун-та, 2018. – 121 с.

5. Каштанов, В. А. Случайные процессы: учебник и практикум для прикладного бакалавриата / В. А. Каштанов, Н. Ю. Энатская. – М. : Издательство Юрайт, 2017. – 156 с.

Разработка динамически обновляемой структуры данных для прогнозирования движения судов

Ю. В. Шишко

Студент магистр

Д. И. Соломатин

Старший преподаватель

Введение

Геоинформационные системы (ГИС) представляют собой мощный инструмент для обработки, анализа и визуализации географической информации. В контексте морской навигации и управления судоходством ГИС играют ключевую роль, обеспечивая морякам и судовладельцам необходимую информацию для принятия обоснованных решений. Системы ГИС позволяют интегрировать данные о морских картах, погодных условиях, гидрометеорологических параметрах, портах, маршрутах и других факторах, влияющих на безопасность и эффективность морской навигации.

Одним из важных аспектов применения ГИС в морской навигации является планирование оптимальных маршрутов для судов. С помощью ГИС можно проводить анализ различных вариантов маршрутов, учитывая такие параметры, как расстояние, время в пути, тип судна, наличие препятствий, актуальные погодные условия и другие факторы. Это позволяет выбирать наиболее безопасные и экономически выгодные маршруты для судов.

ГИС также используются для отслеживания движения судов и контроля за исполнением выбранных маршрутов. Системы ГИС позволяют в реальном времени отображать положение судов на карте, а также мониторить их скорость, курс, изменения маршрута и другие параметры. Это обеспечивает оперативное реагирование на изменяющиеся условия и помогает предотвращать возможные аварийные ситуации[1].

Существуют различные подходы к прогнозированию пути, в данной работе будет рассмотрена модель на основе графа судоходства, представленном на рис. 1.



Рис. 1. Базовая графовая модель

Базовая идея модели представлена Европейским институтом в проекте *searoutes* (<https://github.com/eurostat/searoute>)[2]. Именно от этой идеи мы отталкиваемся в данной работе, однако она не удовлетворяет всем нашим потребностям.

В ходе работы необходимо будет рассмотреть принцип работы прогнозирования движения судов на основе данной модели, выделить основные недостатки и способы решения. В том числе, будет описана реализация дополнения графа на основе данных, поступающих в систему. В заключении будет приведен результат доработки и продемонстрирован интерфейс прогнозирования пути.

1. Базовая графовая модель

Как было указано ранее, базовая идея модели была описана в проекте *searoutes*, который в рамках работы был интегрирован и адаптирован под нужды проекта.

Базовая модель для прогнозирования маршрута представляет из себя граф. В его состав входят:

- Вершины с уникальным идентификатором, координатами, представленными в формате *geometry* (модуль *postgis*).
- Ребра с уникальным идентификатором, ссылками на две соединяемые вершины, а также путем в формате *geometry* и длиной этого пути.

Для ускорения вычислений данный граф помещается в PostgreSQL, так как конкретно модуль *postgis* имеет подходящий для данной работы набор методов. Граф хранится в базе в формате, указанном выше.

Основные этапы по интеграции решения в систему таковы:

- Уменьшение детализации до определенного (5 км) разрешения.
- Перенос вершин графа в PostgreSQL.
- Подсчет весов – длин маршрутов ребер.

В такой конфигурации граф состоит из 23067 вершин и 88291 ребер.

У данной модели, тем не менее, есть ряд недостатков:

- Отсутствие связи по 180 и -180 меридиану.
- Устаревание маршрутов.
- Отсутствие маршрутов в областях.
- Отсутствие обновления доступности маршрутов.
- Не учитывается тип судна.
- Не учитываются риски: пиратство, погодные условия и прочее.

2. Первичная доработка графа

Первичная доработка, которую необходимо сделать для данной модели – сделать склейку карты по 180 меридиану. Такая склейка позволит адекватно строить маршруты через Тихий океан, и позволит избежать неэффективных маршрутов через весь земной шар.

Алгоритм:

- Выбор всех точек на 180 меридиане.
- Выбор всех точек на -180 меридиане.
- Попарная сшивка с применением postgis функции distance.
- Обновление графа с дополненными ребрами.

Как правило, такая доработка производится единоразово, однако при добавлении новых вершин, особенно в области рядом со 180 меридианом, важно проводить повторную генерацию таких путей.

3. Доработка графа на основе поступающих данных

Дальнейшая доработка графа нацелена на обновление графа в соответствии с актуальными маршрутами. Базово для такой доработки необходимо уметь определять в потоке данных маршруты из порта в порт, в свою очередь для этого необходимо уметь определять события входа и выхода судна в область порта.

Ивент судозахода (Arrival):

- Судно вошло в зону порта.
- Скорость судна менее 0.3 узлов/час.
- Предыдущая позиция не была отмечена, как судозаход.
- Предыдущий судозаход не был отмечен в текущем порту.

Ивент отправления из порта (Departure):

- Судно вышло из зоны порта.
- Судно ранее имело статус нахождения в этом порту.

В дальнейшем именно на основе этих данных происходит дальнейшая доработка:

1. Обработка потока данных с целью определения захода в порт и выхода из него.
2. Формирование выборки путей на основе одинаковых последовательностей судозаходов (множество выходов из порта А и последующего захода в порт В).
3. Агрегация похожих путей в общую “ветку” в графе с помощью heatmap.
4. Обновление графа с дополненным набором вершин.

Отдельно здесь стоит отметить, что агрегация похожих путей в единый путь происходит при помощи heatmap. Heatmap (Карта плотностей) необходим для построения и доработки графа[3]. Для каждого типа судов выполняются шаги:

- Агрегация данных по сетке с учетом типов судов (Continues Aggregate).
- Отрисовка heatmap с учетом проекции.
- Нормализация и отсечение по порогу.
- Преобразование в вершины и связи. (как показано на рис.2)

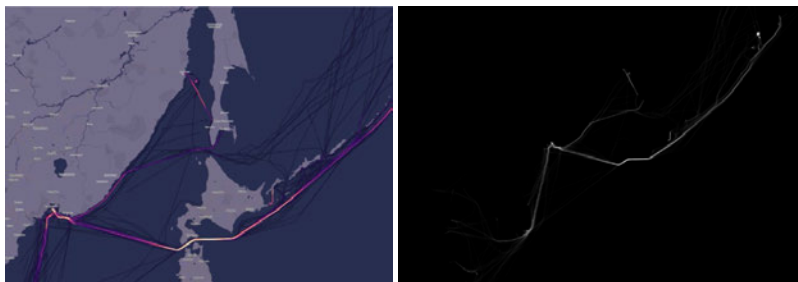


Рис. 2. heatmap и итоговая ветвь графа

После доработки совокупное количество вершин в 7 графах (по типам судов) составило 254 148, а общее количество ребер составило 1 018 899. В среднем на один граф это 36350 вершин и 145557 ребер.

Таким образом, данные доработки позволяют не только периодически актуализировать маршруты в любой области, где стабильно есть данные о перемещениях судов, но также позволяют делать различные графы для разных типов судов, чтобы, например, для танкеров не было возможности идти по рекам к Азовскому морю.

4. Поиск оптимального маршрута алгоритмом Дейкстры

Базовый алгоритм поиска маршрута происходит таким образом:

1. Поиск вершины в графе, находящейся ближе всего к точке начала маршрута. Точкой маршрута может быть либо порт отправления, если это прокладка маршрута из порта в порт, либо текущее местонахождение судна, если это прогнозирование маршрута для конкретного судна.
2. Поиск вершины в графе, находящейся ближе всего к точке завершения маршрута (порт прибытия).
3. Поиск кратчайшего пути с помощью алгоритма Дейкстры.

Алгоритм Дейкстры подходит для поиска ввиду относительно небольшого изначального количества вершин, а также отсутствия ребер отрицательного веса[4]. Тем не менее, при значительном разрастании графа понадобится доработка для A^* , что будет описано в следующем разделе.

На исходном наборе данных поиск происходит за 0.287 секунд в среднем. Ресурсы базы: 72 CPU, 256 RAM, 1TB SSD.

5. Поиск оптимального маршрута алгоритмом A^*

Дополнение графа новыми вершинами делает его более детализованным и замедляет работу алгоритма Дейкстры. Помимо этого, в ранее указанной реализации все еще отсутствует возможность учитывать риски, такие как области пиратства, штормы или области блуждания айсбергов.

Таким образом, помимо доработки графа, также существует необходимость доработать алгоритм поиска пути. Было решено использовать алгоритм A^* , который позволяет как ускорить работу алгоритма Дейкстры[5], так и учесть при планировании маршрута прочие параметры, которые мы обозначили как потенциальные риски.

Доработка с учетом нового алгоритма предполагает такие шаги:

1. Подбираем подходящую эвристику $h(u)$.
2. Изменяем вес ребер с опорой на потенциалы вершин.
3. Ищем наиболее короткий маршрут с помощью алгоритма Дейкстры.

Таким образом, необходимо пересчитать длины ребер с учетом потенциала вершин:

$$h(u) = \text{distance}(u, y)$$

$$\text{new length}(u, v) = \text{old length}(u, v) \cdot (1 + \text{danger}(v)) + h(v) - h(u) \quad (2)$$

Такая доработка позволила учитывать коэффициент опасности в области и учитывать это при планировании маршрута, а также ускорить работу алгоритма Дейкстры на 28%.

Построение пути, указанного на рис. 3, занимает 0,26 секунд.

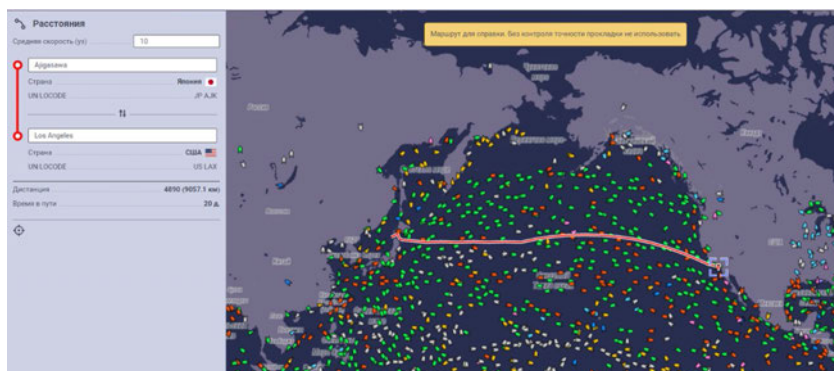


Рис. 3. Построение пути из Японии

Заключение

В данной работе рассмотрен алгоритм для прокладки маршрутов на основе графовой модели, а также варианты доработки базовой модели. Улучшенная модель позволяет учитывать тип судна, принимает во внимание множество параметров риска, имеет более высокую производительность и точность, чем исходная модель.

Использование данной модели в области судоходства может благоприятно повлиять на планирование маршрутов, а также избежать набора рисков. Тем не менее, без контроля точности такой маршрут не стоит использовать, так как он носит скорее справочный характер.

Список литературы

1. Миркин, Б. Г. Введение в анализ данных / Б.Г. Миркин. – М.: Юрайт, 2020. – 175 с.
2. Чашкин, Ю.Р. Математическая статистика. Анализ и обработка данных: Учебное пособие / Ю.Р. Чашкин; Под ред. С.Н. Смоленский. – Рн/Д: Феникс, 2017. – 236 с.
3. Маккинли, У. Python и анализ данных / У. Маккинли. – М.: ДМК, 2015. – 482 с.
4. Тюрин, Ю.Н. Анализ данных на компьютере: Учебное пособие / Ю.Н. Тюрин, А.А. Макаров; Науч. ред. В.Э. Фигурнов. – М.: ИД ФОРУМ, 2017. – 368 с.
5. Ниворожкина, Л.И. Статистические методы анализа данных: Учебник / Л.И. Ниворожкина, С.В. Арженовский, А.А. Рудяга. – М.: Риор, 2018. – 320 с.

Содержание

Статьи студентов бакалавриата

Бабкина Е. А., Копытина Е. А. Разработка веб-приложения для генерации документов, сопутствующих государственной регистрации объектов интеллектуальной собственности	4
Бадалян Д. С., Борисов Д. Н. Моделирование распространения сигнала в городских условиях	11
Бордюжа Е. О., Копытина Е. А., Богданова Н. С. Разработка информационной системы, автоматизирующей генерацию шаблонов документов, сопутствующих прохождению учебных практик ФКН ФГБОУ ВО «ВГУ»	18
Будников А. С., Романов А. В. GPU-реализация машинно-обучаемых потенциалов для молекулярной динамики	26
Бутовцевская С. И., Копытина Е. А. Разработка информационной системы факультета компьютерных наук ФГБОУ ВО «Воронежский государственный университет»	34
Быков М. Н., Копытина Е. А. Разработка системы аутсорсинга краткосрочных работ на предприятиях	42
Воробцов И. А., Ермаков М. В. Сравнительный анализ эффективности SAST/DAST решений в CI пайплайнах	51
Гурьева А. М., Зуев С. А. Автоматический контроллер температуры и влажности для домашних питомцев на базе Arduino	58
Долгих С. В. Организация канала связи клиент-сервер на базе модулей NRF24L01 и двух микроконтроллеров Arduino	66
Загреба Д. А., Стромов А. В. Моделирование итеративной кодовой модуляции с побитовым перемежением с последовательным составным свёрточным кодеком	74
Змаев Д. А., Лысачев П. С. Разработка алгоритма сбора информации о товарах с маркетплейсов	80
Иванников С. П., Самойлов Н. К. Анализ топологии сервиса с помощью технологии Annotation processing	84
Карташов Д. А., Акимов А. В. Обучение искусственному интеллекту для пошаговой стратегии	91
Кобзарь А. И., Татьяна Д. А., Киселев Е. А. Двухкомпонентные системы равномерных сдвигов, порожденные функцией Гаусса и контуром Лоренца	97

Лазуткина А. А., Гаршина В. В. Семантический разбор деловой переписки.....	104
Ламзин С. С., Залыгаева М. Е. Закон Ципфа для решения лингвистических и социально-экономических задач	112
Линьков А. Г., Залыгаева М. Е. Статистический и нейросетевой подходы к прогнозированию временных рядов	120
Луговской К. П., Проскуряков Е. Д., Соломатин Д. И. Виртуальные потоки в Java'21	128
Мамонов Д. В., Самойлов Н. К. Анализ модели графа свойств на примере языка Cypher	134
Мерзляков Н. В., Акимов А. В. Разделение вокала и музыкального сопровождения в песнях с помощью нейронных сетей.....	143
Михалевская Е. Н., Алейникова Н. А. Разработка системы поддержки принятия решения при выборе логистического партнера	153
Нартова К. Р., Швырева А. В. Разработка программного средства защиты отдельных видов информации конфиденциального характера от несанкционированного доступа путем внесения мешающих искажений во фрагменты графических файлов	165
Наумова И. Д., Копытина Е. А. Разработка информационной системы для автоматизации формирования отчетных документов аналитика Центрального Федерального Округа компании ООО «Партнер	174
Небогина Д. Р., Рыкунов Р. К., Киселев Е. А. О константах Рисса для систем целочисленных сдвигов, порожденных окнами Ханна и Хэмминга	182
Новиков А. С., Акимов А. В. Детектирование метеоритных кратеров на спутниковых фотографиях	187
Покушалова Т. С., Гаршина В. В. Выделение именованных сущностей и отношений из предложений на естественном языке на основе контекста	193
Полев А. А., Вахтин А. А. Разработка сайта для купли-продажи авто	200
Положенцев А. А., Самойлов Н. К. Детекция границ объектов на изображении.....	207
Потапов А. Г., Романов А. В. Моделирование межатомного потенциала по результатам квантовой-химического расчёта с использованием симметричных функций Бехлера-Парринелло	215

Пустовалов А. А., Копытина Е. А. Разработка информационной системы управления контентом малого бизнеса на базе телеграм-бота	223
Рогова Е. М., Попов М. И. Численное интегрирование бигармонического уравнения в квадратной области	230
Саратов И. Р., Михайлюк Е. А. Разработка портала центра развития карьеры и бизнес-партнерства ВГУ	236
Сашина А. Р., Гавшин А. Л. Инвалидация кэшей с использованием Debezium	243
Сиваков А. В., Швырева А. В. Автоматизация процесса аутентификации пользователя в системах прокторинга	248
Скарга Д. Ю., Рогачев Е. А., Туровский Я. А. Разработка приложения для восстановления когнитивных способностей у людей с нарушениями ЦНС и опорно-двигательного аппарата	254
Скофенко К. Ю., Чекмарев А. И. Автоматическая генерация пользовательского интерфейса на фреймворке Angular для бэкенд приложений	261
Соболева Е. Д., Попов М. И. Численное решение задачи об изгибе тонкой пластины защемленной по контуру в круговой области	268
Солодовникова Я. В., Копытина Е. А. Разработка веб-приложения для проведения экспертизы работ регионального проекта-преакселератора инноваций ФГБОУ ВО «ВГУ» «Лига инноваций»	275
Федосова П. О., Самойлов Н. К. Анализ производительности бизнес-процессов платформы Camunda	281
Черникова С. А., Самойлов Н. К. Архитектура Apache Kafka	289
Шевцов Р. С., Самойлов Н. К. Алгоритмы шардирования и репликации для хранилищ данных	295
Шубин Д. А., Хвостов С. Г., Коротков В. В. Разработка чат-бота для унификации доступа студентов ФКН к учебной информации	303

Статьи студентов магистратуры

Артемяева Е. И., Воронцов Я. А. Многоцелевая система сбора метрик для контейнеризированных веб-приложений	310
Бондаренко О. Е., Семёнов М. Е. Построение управления воздушного судна местных воздушных линий с использованием методов машинного обучения	316
Борисова А. А., Гаршина В. В. Разработка и реализация онтологии в сфере интеллектуальной собственности	323

Бурдюг П. С., Акимов А. В. Использование архитектуры трансформера для решения задачи описания изображения.....	329
Вакулин Д. А., Иванков А. Ю. Современные нейронные сети в задаче стегоанализа.....	339
Валеев В. И., Тарасов В. С. Анализ рынков NFT и NFT игр	346
Ворсунов А. А., Каладзе В. А. Сравнительный анализ текстов, созданных генеративной сетью и человеком	351
Горбачева И. В., Алейникова Н. А. Анализ устойчивости оптимального решения задачи нахождения критического пути с нечеткими параметрами	358
Денисова Д. А., Гаршина В. В. Алгоритмы мультиагентного обучения с подкреплением категории on-policy.....	366
Железной А. С., Дрюченко М. А. Создание защищенных превью цифровых изображений с использованием стеганографических алгоритмов повышенной пропускной способности и обучаемых алгоритмов сжатия.....	377
Иванов К. Д., Фертиков В. В. Анализ алгоритмов скелетизации бинарных изображений	386
Капкин К. А., Акимов А. В. Применение градиентного бустинга в задаче предсказания результатов футбольных матчей.....	394
Качанова Ю. А., Сирота Е. А. Анализ и исследование методов прогнозирования базового индекса потребительских цен на основе многомерных временных рядов	402
Колесов А. С., Соломатин Д. И. Разработка инструментария для написания тестов в стиле BDD для языка Go.....	409
Косинов Е. Э., Каладзе В. А. Выявление признаков жанра и классификация музыкальных фрагментов с помощью инструментов ИИ.....	417
Лазарев Я. В., Соломатин Д. И. Оптимизация пропускной способности конвейера обработки изображений при помощи генетического алгоритма	424
Локтев В. В., Коротков В. В. Автоматизация распределения задач с учетом предпочтений руководителя и членов проектной команды	430
Малаков Л. А., Матвеев А. В., Максимов А. В. Аппаратная платформа системы биологической обратной связи в игровой форме	438
Масленникова С. С., Коротков В. В. Применение больших языковых моделей для дополнительной разметки данных в задаче оценки времени на исправление дефекта	444

Мосолов А. О., Савинков А. Ю. Методы повышения энергетической эффективности радиоканалов интернета вещей при реализации на микроконтроллере	449
Нагина Е. А., Вахтин А. А. Разработка Telegram-бота для мониторинга курса валют: удобство и функциональность	454
Нартов Е. Д., Соломатин Д. И. Прототип решения для работы в web-браузере с помощью движения глаз	460
Обухова Э. В., Швырева А. В. Лингвистический анализ названий групп студентов ВГУ с целью выявления различий в текстовых корпусах между разными факультетами	467
Пархоменко Д. А., Малыхин А. Ю. Моделирование распространения акустических сигналов в подводных каналах связи	479
Перепёлкин Д. С., Вахтин А. А. Исследование развития киберугроз веб-приложений	484
Петрищев К. О., Борзунов С. В. Методы вычисления показателей Ляпунова гладких динамических систем	490
Постников А. В., Воронцов Я. А. Оптимизация поиска в реестре единого фонда геологической информации	496
Сахаров А. Ю., Митрофанова Е. Ю. Распознавание типа внутренних элементов планировки зданий с использованием нейронных сетей	502
Селиверстов Н. А., Иванков А. Ю. Использование модуля самовнимания и архитектуры трансформер для решения задачи разделения аудиосигнала.....	509
Синяков И. В., Тарасов В. С. Анализ нейросетей для обнаружения киберугроз	518
Соколов В. А., Зуев С. А. Сравнение эффективности алгоритмов шифрования и кодирования в контексте мобильной системы сбора данных.....	526
Соловьев А. С., Лысачев П. С. Применение сопрограмм для решения прикладных задач	532
Ушакова А. А., Гаршина В. В. Исследование возможностей и оценка качества распознавания звучащей речи различными нейросетевыми моделями	540
Фуфаева А. В., Стародубцев И. Ю. Анализ и сравнение алгоритмов машинного зрения для выявления дефектов при производстве печатных плат	547
Хоменко П. А., Гаршина В. В. Извлечение данных о технологии проведения химических реакций из неструктурированных PDF-файлов.....	554

Шипилов С. А., Твердохлебов В. В., Максимов А. В. Разработка клиент-серверной системы анализа аудиоданных.....	563
Шипилова А. В., Матвеев М. Г. Прогнозирование сахарного диабета у пациентов с использованием аппарата цепей Маркова.....	568
Шишко Ю. В., Соломатин Д. И. Разработка динамически обновляемой структуры данных для прогнозирования движения судов.....	576

Научное издание

**ТРУДЫ МОЛОДЫХ УЧЁНЫХ
ФАКУЛЬТЕТА КОМПЬЮТЕРНЫХ НАУК ВГУ**

В ы п у с к 4

Под редакцией *Д. Н. Борисова*

Издано в авторской редакции