



ТРУДЫ МОЛОДЫХ УЧЕНЫХ
ФАКУЛЬТЕТА КОМПЬЮТЕРНЫХ
НАУК ВГУ

Выпуск 3

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 3

Под редакцией кандидата технических наук Д. Н. Борисова

Воронеж
2023

УДК 004.65 + 004.438.5
ББК Ч481(2)22
Т78

*Рекомендовано к опубликованию
Ученым советом факультета компьютерных наук ВГУ*

Труды молодых учёных факультета компьютерных наук ВГУ. Выпуск 3 / под ред. Д. Н. Борисова ; Воронежский государственный университет. – Воронеж, 2023. – 559 с.

В сборник включены научные работы студентов бакалавриата и магистратуры факультета компьютерных наук ВГУ, выполненные в 2022–2023 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, 2023

Статъи студентов бакалавриата

Разработка сервиса, автоматизирующего учет посещаемости онлайн-занятий в портале LMS Moodle

А. Ю. Брюханова

Студент бакалавриата

А. П. Толстобров

Доцент

Введение

Использование средств электронного обучения в образовательном процессе становится все более обычным делом. В Воронежском госуниверситете для этих целей используется образовательный портал «Электронный университет ВГУ», созданный на популярной LMS-платформы Moodle. Поддержка образовательного процесса в портале Moodle не сводится к функционалу, касающемуся только задач обучения. Это могут быть сервисы, реализующие потребности сопровождения учебного процесса, выполнение различного вида учетно-документальных функций. Модульный характер и открытый код системы Moodle позволяет расширять ее функционал путем самостоятельного создания подключаемых к системе программных модулей-плагинов.

Данная работа посвящена совершенствованию сервисов портала, позволяющих автоматизировать достаточно актуальную практическую задачу учета и фиксации в базе данных системы посещаемости студентами учебных занятий. Эта задача оказывается особенно актуальной для занятий, проводимых в онлайн-формате вне учебных аудиторий.

В настоящее время в портале Moodle существует модуль, который реализует довольно широкий набор сервисов по фиксации информации о посещаемости студентами занятий. Он позволяет преподавателю отмечать вручную приход студентов на занятие, делегировать самим студентам отмечать свой приход на занятие в соответствующее расписанию время, настраивать режим автоматической отметки прихода студентов на занятие по факту их захода в электронный курс.

Однако, недостатком функционала этого модуля является то, что все его режимы позволяют фиксировать только факт появления студента

на занятии или его отсутствия. То есть этот модуль не позволяют контролировать электронным способом у студента, отметившегося свой приход в начале занятия, его присутствие на этом занятии в течении всего времени его проведения. Этот фактор не является существенным при проведении занятия в аудитории. Но оказывается актуальным при проведении занятий в онлайн-режиме, когда у преподавателя отсутствует возможность визуального контроля присутствия студента на протяжении всего занятия.

Данная работа посвящена устранению этого недостатка.

1. Постановка задачи

Задачей данной работы является разработка программного модуля-плагина, подключаемого к системе Moodle, и включаемого в состав электронных курсов элемента, обеспечивающего контроль присутствия студента за компьютером на протяжении всего онлайн-занятия.

Возможность такого решения позволяет модульной архитектурой системы Moodle и открытостью ее исходного кода системы Moodle.

Для решения поставленной задачи предлагается в разрабатываемом модуле реализовать не только средства для подтверждения и фиксации факта обращения студента к электронному курсу в начале занятия, но и средства подтверждения нахождения студента за компьютером в данном курсе в случайно выбираемые моменты времени прохождения занятия.

Для того, чтобы студент имел возможность подтвердить свое присутствие на онлайн-занятии, разрабатываемое приложение должно в течении занятия, в случайные, настраиваемые преподавателем моменты времени, выставлять на короткое время на экране компьютера модальное окно с радиокнопкой, которую должен нажать студент для подтверждения своего присутствия. Отметки подтверждения или не подтверждения студентами своего присутствия на занятии должны фиксироваться и суммироваться в электронном журнале посещаемости.

2. Реализация модуля регистрации присутствия студентов на онлайн-занятии

Для отработки функционала разрабатываемого программного модуля на персональном компьютере была развернута среда системы Moodle, включающая в себя:

- Веб-интерфейс для управления курсами и пользовательскими аккаунтами.
- Хранилище данных для хранения всех материалов и ресурсов курса.
- Механизмы обмена сообщениями и комментариев между преподавателями и студентами.

- Инструменты для создания и организации онлайн-тестов и опросов.
- Функции совместной работы, такие как форумы и чаты.
- Интеграция с различными средствами авторизации и идентификации пользователей.
- Аналитические инструменты для отслеживания прогресса и оценки успеваемости студентов.
- Инструменты для управления курсами и создания настраиваемых отчетов;
- Различные плагины и сценарии для расширения возможностей системы.

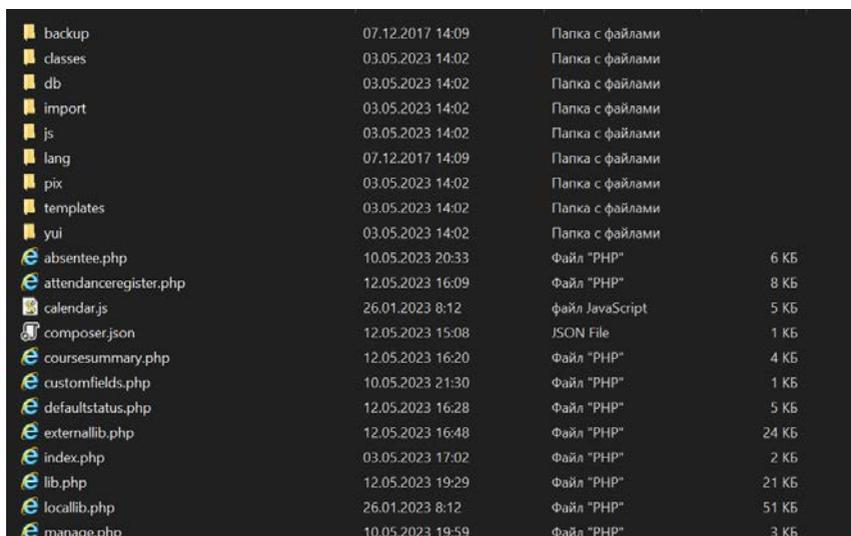
Разработка приложения проводилась на языке PHP, на котором реализована вся система Moodle.

Разработанный плагин имеет следующую структуру:

- backup: содержит основные функции резервного копирования курса: резервное копирование, восстановление и импорт курсов;
- pix: содержит изображения, которые будет использовать плагин. Например, файл с иконкой плагина icon.png;
- db: в эту папку добавлены файлы, отвечающие за работу с базой данных;
- classes: используется для хранения создаваемых разработчиком вспомогательных классов. Данная директория не является необходимой для работы плагина, но позволяет легко сориентироваться в структуре проекта;
- lang: эта папка используется для размещения языковых пакетов и библиотек обработчиков юникода;
- attendanceregister.php - основной файл плагина, который включает всю логику работы с посещаемостью;
- version.php - файл, содержащий информацию о версии плагина.
- lib.php - файл, содержащий вспомогательные функции для работы с посещаемостью;
- renderer.php - файл, содержащий класс `Renderer`, который отвечает за вывод информации о посещаемости;
- style.css - файл со стилями для отображения информации о посещаемости на странице курса;
- README.md - файл с описанием плагина и инструкцией по его установке и настройке;

Таким образом, структура разработанного плагина изображена на рис. 1 и включает в себя основные файлы и компоненты, необходимые для работы с посещаемостью в курсах в системе. Все файлы и папки

располагаются внутри директории плагина, которая находится в директории /mod/ в структуре Moodle.



backup	07.12.2017 14:09	Папка с файлами	
classes	03.05.2023 14:02	Папка с файлами	
db	03.05.2023 14:02	Папка с файлами	
import	03.05.2023 14:02	Папка с файлами	
js	03.05.2023 14:02	Папка с файлами	
lang	07.12.2017 14:09	Папка с файлами	
pix	03.05.2023 14:02	Папка с файлами	
templates	03.05.2023 14:02	Папка с файлами	
yui	03.05.2023 14:02	Папка с файлами	
absentee.php	10.05.2023 20:33	Файл "PHP"	6 КБ
attendanceregister.php	12.05.2023 16:09	Файл "PHP"	8 КБ
calendar.js	26.01.2023 8:12	файл JavaScript	5 КБ
composer.json	12.05.2023 15:08	JSON File	1 КБ
coursesummary.php	12.05.2023 16:20	Файл "PHP"	4 КБ
customfields.php	10.05.2023 21:30	Файл "PHP"	1 КБ
defaultstatus.php	12.05.2023 16:28	Файл "PHP"	5 КБ
externallib.php	12.05.2023 16:48	Файл "PHP"	24 КБ
index.php	03.05.2023 17:02	Файл "PHP"	2 КБ
lib.php	12.05.2023 19:29	Файл "PHP"	21 КБ
locallib.php	26.01.2023 8:12	Файл "PHP"	51 КБ
manage.php	10.05.2023 19:59	Файл "PHP"	3 КБ

Рис. 1. Структура плагина

Для обеспечения его встраивания в систему в его составе должно быть:

- Файл плагина с расширением .zip.
- Файл манифеста плагина с информацией о его версии, названии, описании, авторах и других параметрах.
- Файлы языковых переводов для всех поддерживаемых языков.
- Файлы кода и шаблонов для реализации функций плагина.
- Файлы изображений, CSS-стилей и других ресурсов для оформления интерфейса плагина.
- Инструкция для пользователей по установке и использованию плагина.

На рис. 2 приведен скриншот интерфейса настройки занятия, добавленного в соответствии с расписанием курса.

Добавить занятие

Тип занятия Общее Групповое

Количество появлений

Дата занятия мая 2023

Время с до

Описание

Rich text editor toolbar with icons for bold, italic, underline, strikethrough, link, unlink, list, list-group, link, unlink, help, undo, redo, link, unlink, help.

Создавать для занятия событие в календаре

Добавить несколько занятий

Рис. 2. Скриншот интерфейса настройки занятия

Журнал занятий формируется во вкладке «Занятия». Таблица данной страницы включает такие данные, как: дата и время занятия, тип и описание занятия. Также преподаватель имеет возможность отмечать посещаемость студентов вручную, изменять и удалять занятия, выбирая соответствующие обозначения в столбце «Действия». Скриншот данной страницы представлен на рис. 3.

#	Дата	Время	Тип	Описание	Действия
1	14.05.22 (Сб)	11:30 - 13:05	Общее	Обычное занятие	
2	21.05.22 (Сб)	11:30 - 13:05	Общее	Обычное занятие	
3	28.05.22 (Сб)	11:30 - 13:05	Общее	Обычное занятие	
4	4.06.22 (Сб)	11:30 - 13:05	Общее	Обычное занятие	
5	11.06.22 (Сб)	11:30 - 13:05	Общее	Обычное занятие	

Рис. 3. Скриншот интерфейса журнала

Автоматизация отметки посещаемости заключается в том, что на протяжении занятия появляется всплывающее окно, в котором студент должен отметить свою посещаемость. Число появлений данного окна устанавливается преподавателем в настройках занятия. Время появления устанавливается системой автоматически в произвольные промежутки времени. В соответствии с этой отметкой студенту

присваивается статус присутствовал или отсутствовал. На рис. 4 изображено реализованное всплывающее окно.

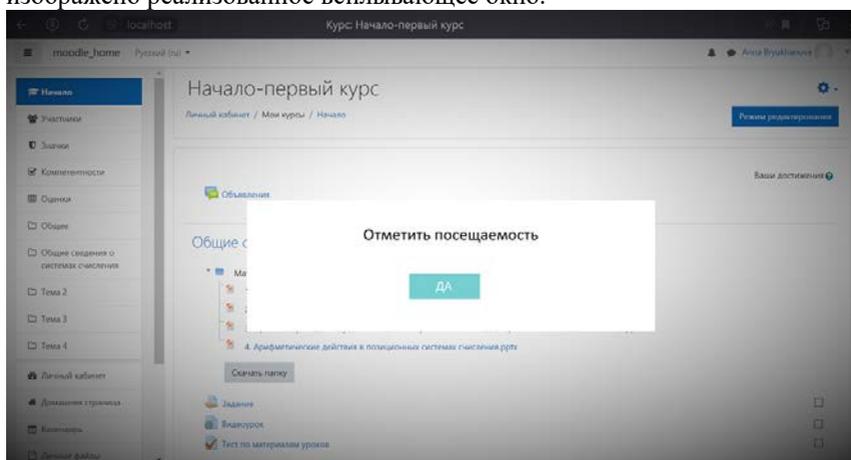


Рис. 4. Всплывающее окно

Скриншот вывода информации о посещаемости изображен на рис. 5.

Пользователи		Занятия		
Фамилия / Имя	Адрес	14.05	21.05	28.05
Логин	электронной почты	11:30	11:30	11:30
Головко Олеся Ивановна	16190132 olesya.qwq@mail.ru	Общее	Общее	Общее
Описание		Присутствовал: 0 Не был: 0	Присутствовал: 0 Не был: 0	Присутствовал: 0 Не был: 1

Рис. 5. Вывод информации о посещаемости

Закключение

Данная статья посвящена разработке и реализации программного модуля для автоматизации учета посещаемости студентами онлайн-занятий в портале LMS Moodle. В ходе работы был разработан плагин с удобным и интуитивно понятным интерфейсом для преподавателей и студентов, который обеспечивает удобство использования данного модуля. Стоит отметить, что сервис позволяет фиксировать не только факт появления студента на занятии или его отсутствия, но и

контролировать у студента, отметившегося свой приход в начале занятия, его присутствие на этом занятии в течении всего времени его проведения. Также сервис значительно облегчает работу преподавателей и сокращает время на учет посещаемости.

Список литературы

1. Гильмутдинов А. Х. Электронное образование на платформе Moodle / А. Х. Гильмутдинов, Р. А. Ибрагимов, И. В. Цивильский – Казань.: Изд-во КГУ, 2008. – 169 с.
2. Анисимов А. М. Работа в системе дистанционного обучения Moodle / А. М. Анисимов – Харьков.: ХНАГХ, 2009. – 292 с.
3. Белозубов А. В. Система дистанционного обучения Moodle: учебно-методическое пособие / А. В. Белозубов, Д. Г. Николаев– СПб.: Изд-во СПбГУ ИТМО, 2007. – 108 с.
4. Раецкий А.Д. Разработка плагина «Портфолио СибГИУ» для системы управления обучением «Moodle» / Раецкий А.Д., Шлянин С.А., Ермакова Л.А. // Кибернетика и программирование: электрон. журн. – 2016. – № 2. – С. 52 - 61.

Реализация рекомендательной системы пользовательских предпочтений на основе сингулярного разложения матриц на языке Python

Д. С. Бурнашев

Студент бакалавриата

М. Е. Залыгаева

Старший преподаватель

Введение

В современном мире, где количество информации растет в геометрической прогрессии, необходимость в эффективных алгоритмах, позволяющих обрабатывать и анализировать большие объемы данных, становится все более актуальной. Рекомендательные системы являются одним из таких алгоритмов, которые помогают пользователям находить нужную информацию и товары в огромном количестве предложений. В данной статье мы рассмотрим реализацию рекомендательной системы на основе сингулярного разложения матриц на языке Python. Этот метод является одним из наиболее эффективных и широко используемых подходов для построения рекомендательных систем.

1. Алгоритмы полного и усеченного сингулярного матричного разложения в контексте рекомендательной системы

Сингулярное матричное разложение, которое позволяет представить исходную матрицу в виде произведения трех матриц - матрицу левых сингулярных векторов, матрицу правых сингулярных векторов и диагональную матрицу сингулярных значений. SVD разложение позволяет выделить наиболее значимые факторы, которые влияют на оценки пользователей.

В теореме о сингулярном разложении утверждается, что у любой матрицы A размера $m \times n$ существует разложение в произведение трех матриц U, Σ, V^T [1]:

$$A = U \times \Sigma \times V^T .$$

$n \times m \quad n \times n \quad n \times m \quad m \times m$

Матрицы U и V^T ортогональные, а Σ – диагональная (хотя и не квадратная).

$$UU^T = I_n, VV^T = I_m,$$

$$\Sigma = \text{diag}(\lambda_1, \dots, \lambda_{\min(n,m)}), \lambda_1 \geq \dots \geq \lambda_{\min(n,m)} \geq 0$$

Причем лямбды в матрице Σ будут упорядочены по невозрастанию.

Перемножив матрицы Σ и U , обозначим результат как новую матрицу P с такой же размерностью $n \times d$. Правую сингулярную матрицу V^T из усеченного разложения обозначим как Q . Тогда получаем:

$$R \approx \begin{matrix} P & \times & Q \\ n \times m & n \times d & d \times m \end{matrix}$$

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u,$$

$$p_u \in P, u \in [1, n], q_i \in Q, i \in [1, m],$$

где p_u – вектор признаков пользователя u , q_i – вектор признаков фильмов.

Помимо обычного разложения, бывает еще усеченное, когда из λ остаются только первые d чисел, а остальные мы полагаем равными нулю.

$$\lambda_{d+1}, \dots, \lambda_{\min(n,m)} := 0$$

Это равносильно тому, что у матриц U и V^T d мы оставляем только первые d столбцов, а матрицу Σ обрезаем до квадратной $d \times d$ (см. рис. 1):

$$A' = \begin{matrix} U' & \times & \Sigma' & \times & V'^T \\ n \times m & n \times d & d \times d & d \times m \end{matrix}$$

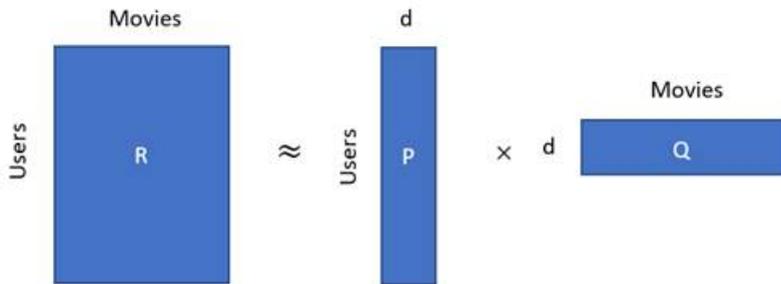


Рис. 1. Матричное разложение

Полученная матрица A' хорошо приближает исходную матрицу A и, более того, является наилучшим низкоранговым приближением с точки зрения среднеквадратичного отклонения.

Усеченное сингулярное матричное разложение позволяет сократить размерность матрицы, оставляя только наиболее важные компоненты, что упрощает ее обработку и анализ. Разреженная матрица, такая как рейтинг фильмов, содержит множество нулевых элементов, что делает ее размерность значительно больше, чем количество ненулевых элементов. Усеченное разложение позволяет эффективно обрабатывать разреженные матрицы, такие как рейтинг фильмов, и извлекать из них полезную информацию [2–4].

2. Использование градиентного спуска для предсказания рейтинга пользователя

Математический аппарат SGD основан на минимизации функции ошибки, которая измеряет разницу между прогнозируемыми и фактическими оценками пользователей.

В каждой итерации SGD выбирает случайный пользователь и элемент, вычисляет ошибку прогнозирования и обновляет соответствующие веса модели (с помощью градиента функции потерь).

Стохастический градиентный спуск для сингулярного разложения:

$$\begin{aligned}
 b_u &\leftarrow b_u + \gamma(e_{ui} - \lambda b_u) \\
 b_i &\leftarrow b_i + \gamma(e_{ui} - \lambda b_i) \\
 p_u &\leftarrow p_u + \gamma(e_{ui} \cdot q_i - \lambda p_u) \\
 q_i &\leftarrow q_i + \gamma(e_{ui} \cdot p_u - \lambda q_i)
 \end{aligned}$$

где $e_{ui} = r_{ui} - \hat{r}_{ui}$; b_u – смещение пользователя u ; b_i – смещение фильма i ; p_u – вектор признаков пользователя u ; q_i – вектор признаков фильма i ; γ – скорость обучения; e_{ui} – ошибка предсказания рейтинга пользователя u для фильма i ; λ – коэффициент регуляризации.

Метод численной оценки качества работы рекомендательной системы.

RMSE (Root Mean Square Error) – это метрика, которая измеряет среднеквадратичное отклонение между фактическими значениями и предсказанными значениями модели. Она показывает, насколько точно модель предсказывает результаты. Чем меньше значение RMSE, тем лучше модель предсказывает рейтинги.

Метрика RMSE:

$$RMSE = \sqrt{\frac{1}{n} \cdot \sum (y_{pred} - y_{true})^2},$$

где y_{pred} – предсказанные значения, y_{true} – фактические значения, n – количество наблюдений.

3. Архитектура рекомендательной системы

Была проработана логика входа в рекомендательную систему, как для новых пользователей, так и для зарегистрированных в системе. Также была реализована концепция единого основного меню, включающего в себя весь основной функционал рекомендательной системы. А именно:

Оценка нового фильма. Выбор в поисковике фильма и выставление оценки в диапазоне [1...5].

Поиск фильмов наиболее схожих с фильмами, которые высоко оценивал пользователь. Система из облачного хранилища загружает актуальное SVD разложение и матрицу пользователь/оценка, выбирает самую высокую оценку, которую ставил пользователь фильмам, находит все фильмы с такими оценками и выводит на экран пользователя для каждого такого фильма заданное количество наиболее схожих фильмов.

Рекомендация новых фильмов на основе предсказанных рейтингов для непросмотренных фильмов. Система из облачного хранилища загружает актуальную матрицу предсказанных оценок непросмотренных фильмов, выбирает заданное количество фильмов с наибольшим рейтингом и выводит на экран пользователю.

Графическая визуализация активностей рекомендательной системы показана на рис. 2.

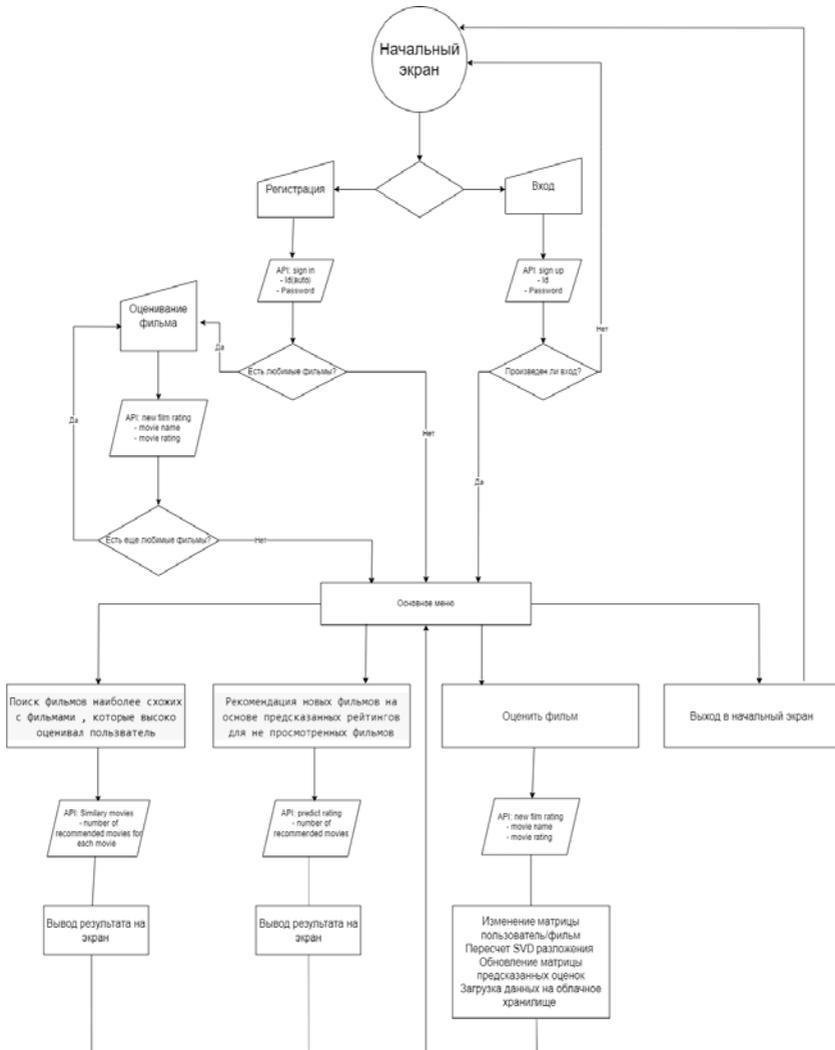


Рис. 2. Графическая визуализация архитектуры активностей рекомендательной системы

4. Решение проблемы «холодного старта»

При добавлении нового пользователя в систему возникает проблема того, что у пользователя нет оцененных фильмов и, следовательно, нет и рекомендаций. Данная проблема называется «холодным стартом».

Для решения проблемы была реализована архитектура, позволяющая пользователю добавить данных о себе в систему, путем выбора в поисковике любимых фильмов с последующим добавлением в систему. Если же у пользователя нет любимых фильмов, выбирается наиболее оцениваемый фильм пользователями, находится среднее значение оценки и добавляется для нового пользователя в базу данных. На основе этих минимальных данных пересчитываются все матрицы, после первого самостоятельного оценивания фильма новым пользователем, первая «искусственная» оценка убирается из базы данных.

При выполнении действий по добавлению новых пользователей, а также выставление пользователями новых оценок для фильмов пересчитывается и сингулярное матричное разложение, и матрица предсказанных рейтингов пользователей

Также стоит отметить, что скорость получения сингулярного разложения, а также матриц P и Q , а, следовательно, и матрицы предсказанных оценок при помощи градиентного спуска отличается своей быстротой выполнения.

На рис. 3–8 показаны примеры визуализации некоторых элементов графического интерфейса и вывод на экран результатов работы рекомендательной системы.

Рекомендательная система на основе SVD разложения

Войдите в систему под своим id:

User ID:	<input type="text" value="132"/>
Password:	<input type="password" value="*****"/>

Рис. 3. Окно входа в рекомендательную систему

Данного ID нет в базе данных

Я новый пользователь

Ввести ID заново

This screenshot shows a light blue header with the text "Я новый пользователь" and a white button below it with the text "Ввести ID заново".

Рис. 4. Окно, в случае если пользователя с введенным ID нет в базе данных.

Произведен вход под вашим id

Оценить фильм

Поиск фильмов наиболее схожих с фильмами , которые высоко оценивал пользователь

Рекомендация новых фильмов на основе предсказанных рейтингов для не просмотренных фильмо

Выход

This screenshot shows a menu with four items: "Оценить фильм" (green), "Поиск фильмов наиболее схожих с фильмами , которые высоко оценивал пользователь" (blue), "Рекомендация новых фильмов на основе предсказанных рейтингов для не просмотренных фильмо" (blue), and "Выход" (pink).

Рис. 5. Основное меню для пользователя

Выберите фильм

arg

Fargo (1996) ▼

This screenshot shows a text input field containing "arg" and a dropdown menu with "Fargo (1996)" selected and a downward arrow.

Выберите оценку для фильма

4 ▼

Оценить фильм

This screenshot shows a dropdown menu with "4" selected and a downward arrow, and a green button labeled "Оценить фильм" below it.

Рис. 6. Окно оценивания фильма для пользователя находящегося в системе

Введите количество рекомендуемых фильмов

Кол-во 3

Подтвердить

This screenshot shows a text input field with "3" and a button labeled "Подтвердить" below it.

Рис. 7. Вывод схожих оцененных фильмов для пользователя

Наибольшая оценка, которую вы давали фильмам = 5.0

Для каждого фильма с наибольшей оценкой будет найден наиболее похожий фильм

Набор из 3 фильмов схожих с фильмом: Fargo (1996)

Trainspotting (1996)

Leaving Las Vegas (1995)

Twelve Monkeys (1995)

Набор из 3 фильмов схожих с фильмом: Godfather, The (1972)

GoodFellas (1990)

Boat, Das (1981)

Godfather: Part II, The (1974)

Набор из 3 фильмов схожих с фильмом: Good Will Hunting (1997)

Amistad (1997)

As Good As It Gets (1997)

Apt Pupil (1998)

Рис. 8. Вывод схожих оцененных фильмов для пользователя

Заключение

В данной статье рассмотрены полное и усеченное сингулярное матричное разложение. Для сингулярного матричного разложения рассмотрен алгоритм SGD с целью предсказания рейтингов фильмов для пользователей. Подготовлен набор данных для рекомендательной системы. Построена архитектура рекомендательной системы на основе сингулярного матричного разложения. Решена проблема «Холодного старта» для новых пользователей. Программно реализована рекомендательная система на языке Python и оценено качество рекомендаций.

Список литературы

1. Степанов, А. Ю. Матричные разложения в задачах рекомендательных систем / А. Ю. Степанов, А. А. Гуцин // Вестник МГУ. Сер. 15, Вычислительная математика и кибернетика. – 2018. – № 3. – С. 4–22.
2. Корень, Я. Методы факторизации матриц для рекомендательных систем / Я. Корень, Р. Белл, К. Волинский // Компьютер. – 2009. – № 42(8). – С. 30–37.
3. Корень, Я. Методы факторизации матриц для рекомендательных систем / Я. Корень, Р. Белл, К. Волинский // Обзор SIAM. – 2009. – № 51(3). – С. 455–500.
4. Корень, Я. Факторизация встречает окрестность: многоаспектная модель коллаборативной фильтрации / Я. Корень // Материалы 14-й международной конференции ACM SIGKDD по открытию знаний и добыче данных. – 2010. – С. 426–434.

Глубокое обучение в задаче стегоанализа цифровых изображений

Д. А. Вакулин

Студент бакалавриата

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 нелегальной направленности.

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 был предложен алгоритм стегоанализа на основе глубоких сверточных нейронных сетей с использованием алгоритма вторичной обработки результатов классификации блоков изображения.

1. Алгоритмы стеганографии в пространственной области

Современные алгоритмы стеганографии (встраивания) WOW [1], S-UNIWARD [2] и HILL [3] во многом схожи, так как основаны на применении синдромного сверточного кода под названием Syndrome-

Trellis Code (STC) [4], позволяющего максимально снизить вносимые в исходное изображение искажения, а также обеспечить возможность восстановления стегосообщения после повреждения стегоконтейнера из-за сжатия, воздействия шумов или других причин.

Рассматриваемые алгоритмы встраивания различаются в основном только функциями расчета цены встраивания и универсальностью. Общий алгоритм встраивания изображен на рис. 1.

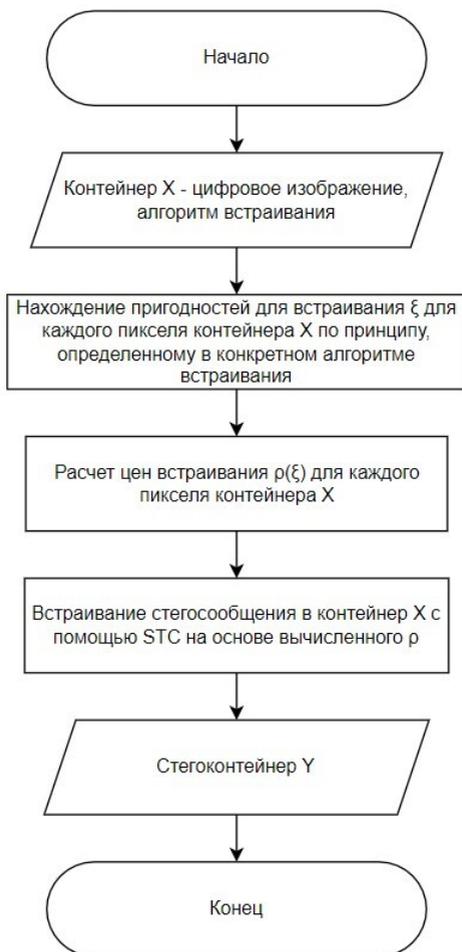


Рис. 1. Обобщенный алгоритм встраивания в пространственную область цифрового изображения

STC универсален, с его помощью возможно закодировать бинарное стегосообщение под любой тип встраивания: бинарный, тернарный и т.д., а также, соответственно, декодировать. Это возможно благодаря алгоритмам сведения небинарных случаев встраивания к бинарным.

2. Используемая сверточная нейронная сеть

Архитектура используемой СНС представлена на рис. 2. Она состоит из трех модулей: модуля предобработки, сверточного модуля и модуля классификации. Модуль предобработки представляет собой сверточный слой с пространственным высокочастотным фильтром для повышения значимости зашумленных областей изображения, в которых содержится стегосообщение. Сверточный модуль и модуль классификации состоят из трех сверточных слоев и трех полносвязных слоев соответственно.

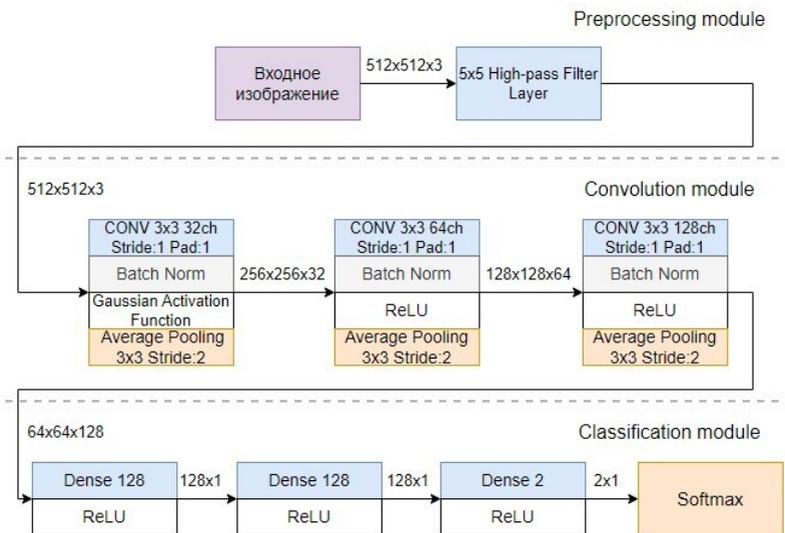


Рис. 2. Архитектура используемой СНС

Данная СНС обучалась на наборе данных PPG-LIRMM-COLOR [5], содержащем 10000 цветных изображений с разрешением 512×512 . Для исследования всего были подготовлены 20000 изображений: 10000 исходных и по 10000 изображений со стегосообщением, встроенным различными алгоритмами стеганографии. Обучающая выборка состояла из 16000 изображений, а валидационная и тестовая из 1000 и 3000 изображений соответственно.

Для обучения сетей с различными размерами входного блока изображения вырезались фрагменты изображения с разрешением $n_x \times m_y$ и составлялись выборки указанных ранее объемов. Из каждого изображения случайным образом вырезался один из блоков с высоким значением энтропии Шеннона, так как энтропия больше в зашумленных областях и, соответственно, в этих блоках находится оптимальное количество измененных пикселей со встроенным стегосообщением.

Обучение сетей проводилось с различными гиперпараметрами для различных разрешений входных изображений. Это обусловлено ограничениями облачной платформы Kaggle, на которой проводилось исследование. Ограничения затрагивают время использования графических процессоров и доступный объем видеопамяти. Сети обучались в течении 10-20 эпох.

Результаты оценки точности моделей рассматриваемой СНС без алгоритма вторичной обработки результатов классификации блоков представлены в табл. 1.

Таблица 1

Результаты оценки точности (в процентах) моделей без алгоритма вторичной обработки результатов классификации блоков изображения при различных значениях полезной нагрузки стегоконтейнера (pl)

Алгоритм стеганографии	WOW		S-UNIWARD		HILL	
	pl=0.2	pl=0.4	pl=0.2	pl=0.4	pl=0.2	pl=0.4
32x32 P=256	74.37	84.77	70.30	84.30	68.40	78.70
64x64 P=64	76.17	88.23	71.07	84.50	70.43	85.17
128x128 P=16	76.80	89.07	73.90	88.63	72.57	85.93
512x512 P=1	58.87	89.90	56.23	89.23	55.83	86.10

3. Алгоритм вторичной обработки результатов классификации

При анализе цветных изображений с разрешением $N_x \times M_y$ в ходе вторичной обработки результатов классификации блоков с разрешением $n_x \times m_y$, получим $P = (N_x \times M_y) / (n_x \times m_y)$ бинарных решений.

Получаемые по каждому блоку решения можно считать статистически независимыми. Обозначим эти решения как $x_k, x_k \in \{0,1\}$, где значение $x_k = 0$, обозначает решение в пользу отсутствия стегосообщения в рассматриваемом блоке, а значение $x_k = 1$ обозначает решение в пользу наличия стегосообщения в блоке.

На вход алгоритма поступают не все блоки изображения, а только те, которые потенциально могут содержать определенное количество модифицированных пикселей, которое выше заданного порога R . Данный порог определяется экспериментальным путем. Выбор блоков происходит путем встраивания стегосообщения в исходное изображение и подсчета числа измененных пикселей в каждом блоке с помощью матрицы разности изображений.

Пусть L_x – количество единиц, полученных в ходе классификации блоков изображения. Таким образом получим следующий алгоритм вторичной обработки:

$$L_x = \sum_{k=1}^P x_k \begin{matrix} \omega_2 > \\ < \\ \omega_1 \end{matrix} L_0, \quad (1)$$

$$R = \rho m, \quad (2)$$

$$m = \frac{1}{P} \sum_{k=1}^P R_k. \quad (3)$$

где L_0 – порог для принятия решения, x_k – решение по k -му блоку, R – порог количества потенциально измененных пикселей блока для его подачи на вход алгоритма вторичной обработки, ρ – коэффициент, регулирующий порог, m – среднее арифметическое количества потенциально измененных пикселей в изображении, R_k – количество потенциально измененных пикселей k -го блока.

Главной задачей является определение оптимального порога L_0 . Значение коэффициента ρ определяется экспериментальным путем, в данном исследовании $\rho = 0.5$.

4. Результаты тестирования

При тестировании эффективности алгоритма вторичной обработки классификации блоков изображения, блоки с наибольшим количеством потенциально модифицированных пикселей выбирались не по значению энтропии Шеннона, а с помощью матрицы разности исходного изображения и изображения со встроенным стегосообщением.

В ходе исследования экспериментально были определены оптимальные значения порога L_0 . При классификации стегоконтейнеров с полезной нагрузкой 0.4 бит/пиксель максимальная точность стегоанализа достигается при $L_0 = 0.65P$, а при полезной нагрузке 0.2 бит/пиксель – $L_0 = 0.55P$, где P – количество рассматриваемых блоков.

В табл. 2 представлены результаты оценки точности моделей с внедренным алгоритмом вторичной обработки результатов классификации блоков изображения.

Таблица 2

Результаты оценки точности (в процентах) моделей с алгоритмом вторичной обработки результатов классификации блоков изображения при различных значениях полезной нагрузки стегоконтейнера (pl)

Алгоритм стеганографии	WOW		S-UNIWARD		HILL	
	pl=0.2	pl=0.4	pl=0.2	pl=0.4	pl=0.2	pl=0.4
32x32 P=256	86.50	94.40	83.67	94.33	81.20	93.60
64x64 P=64	86.90	94.93	83.97	94.43	83.43	94.07
128x128 P=16	87.17	95.23	84.60	94.97	83.30	94.20

Алгоритм вторичной обработки результатов классификации блоков позволяет получить прирост точности модели до 8% при полезной нагрузке стегоконтейнера 0.4 бит/пиксель и до 28% при полезной нагрузке стегоконтейнера 0.2 бит/пиксель.

Заключение

В ходе работы был разработан алгоритм вторичной обработки результатов классификации блоков изображения, реализованный по

схеме наивного байесовского классификатора. Результаты исследования показывают, что применение данного алгоритма существенно повышает точность стегоанализа при малых значениях полезной нагрузки стегоконтейнера.

Данный алгоритм является универсальным, он также может быть использован для управления соотношением ошибок первого и второго рода без значительной потери в эффективности стегоанализа.

Список литературы

1. Holub, V. Designing steganographic distortion using directional filters / V. Holub, J. Fridrich, // 2012 IEEE International Workshop on Information Forensics and Security (WIFS). – Costa Adeje, 2012. – С. 234-239.

2. Holub, V. Universal distortion function for steganography in arbitrary domain / V. Holub, J. Fridrich, T. Denmark // EURASIP Journal on Information Security. – 2014. – № 1. – С. 1-13.

3. Li, B. A new cost function for spatial image steganography / B. Li, M. Wang, J. Huang, X. Li // 2014 IEEE International Conference on Image Processing (ICIP). – Paris, 2014. – С. 4206-4210.

4. Filler, T. Minimizing Additive Distortion in Steganography Using Syndrome-Trellis Codes / T. Filler, J. Judas, J. Fridrich // IEEE Transactions on Information Forensics and Security. – 2011. – № 3. – С. 920-935.

5. PPG-LIRMM-COLOR base [Электронный ресурс]: база данных. – Режим доступа : <https://www.lirmm.fr/~chaumont/PPG-LIRMM-COLOR.html>

Анализ уязвимостей смарт-контрактов платформы Ethereum и их устранение на примере языка программирования Solidity

В. И. Валеев

Студент бакалавриата

В. С. Тарасов

Старший преподаватель

Введение

В настоящее время развиваются технологии блокчейна и NFT. Компании разворачивают корпоративные блокчейны, разрабатываются и набирают популярность игры с NFT.

Однако, речь идет о довольно молодой технологии, в которой не так много опытных специалистов. Здесь важно отметить, что нередко в блокчейне происходят транзакции по перечислению криптовалюты с одного счета на другой. Также крайне важно подчеркнуть, что все развернутые в блокчейнах программы — смарт-контракты — являются общедоступными для просмотра. Безусловно, доступные для чтения программы по перечислению криптовалюты являются точками притяжения для злоумышленников. Ввиду этого на разработчика смарт-контракта ложится достаточно серьезная ответственность. Помимо реализации основной бизнес-логики необходимо дополнительно рассматривать аспект безопасности. В данной статье будут рассмотрены 3 уязвимости, характерных для смарт-контрактов на платформе Ethereum, за устранение которых должен отвечать разработчик: рекурсивный вызов, управление доступом и арифметическое переполнение. Существуют и другие уязвимости, но именно перечисленные можно назвать основными, за недопущением которых должен следить программист. Также будут приведены практические примеры на языке программирования Solidity, так как Solidity является самым популярным языком разработки смарт-контрактов для блокчейнов на платформе Ethereum.

1. Рекурсивный вызов

Рекурсивный вызов является одной из самых известных уязвимостей Ethereum. Рекурсивный вызов происходит, когда вызовам

внешнего смарт-контракта разрешено делать вызовы вызывающего смарт-контракта до завершения первоначального выполнения [1]. Можно сказать, что внутри одной транзакции формируется новая, внутри новой еще одна и так далее. Так, уязвимый смарт-контракт совершает вызов к другому контракту, при этом внешний смарт-контракт может сделать ответный вызов функций уязвимого смарт-контракта внутри начального вызова. Схема уязвимости представлена на рис. 1.

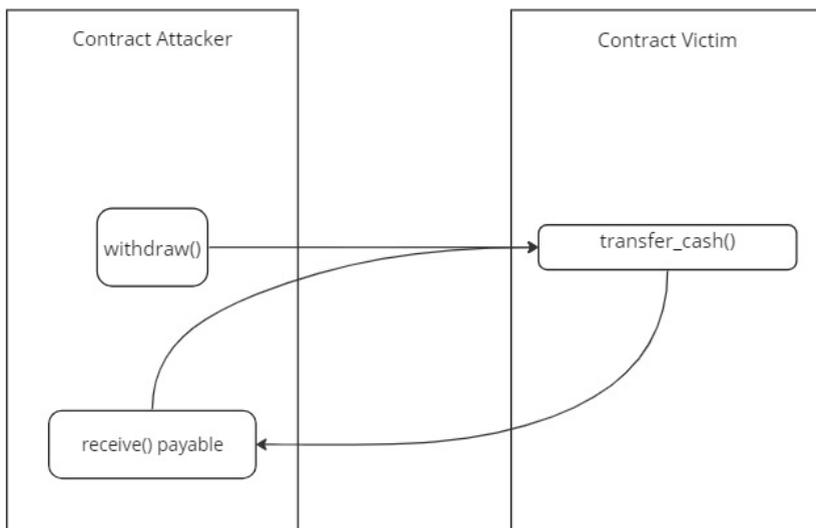


Рис. 1. Общая схема уязвимости “Рекурсивный вызов”

Проиллюстрирован пример для случая, когда смарт-контракт Attacker обращается к контракту Victim с целью получить от него криптовалюту. Для перечисления, конечно, у Attacker, если он реализован на языке программирования Solidity, должна быть функция receive() с модификатором payable, которая может быть реализована в принципе как угодно, в том числе внутри себя вызывать ту же функцию transfer_cash() для получения криптовалюты. Смарт-контракт жертвы не знает об этом и неявно при перечислении криптовалюты обращается к функции receive() атакующего смарт-контракта, что запускает рекурсию, в соответствии с оговоренной ранее реализацией функции receive().

Допущение такой уязвимости однажды привело к сильному ветвлению Ethereum, а нанесенный ущерб оценивается свыше 50 миллионов долларов США [2].

Безусловно, имеются контрмеры. Стандартным путем является фиксация статуса транзакции на момент ее начала, а этому должна предшествовать проверка этого же статуса. Таким образом, при попытке рекурсивно второй раз вызывать ту же функцию не будет пройдена проверка, так как транзакция уже началась. Соответственно, рекурсия сразу же прервется. Схема решения проблемы представлена на рис. 2.

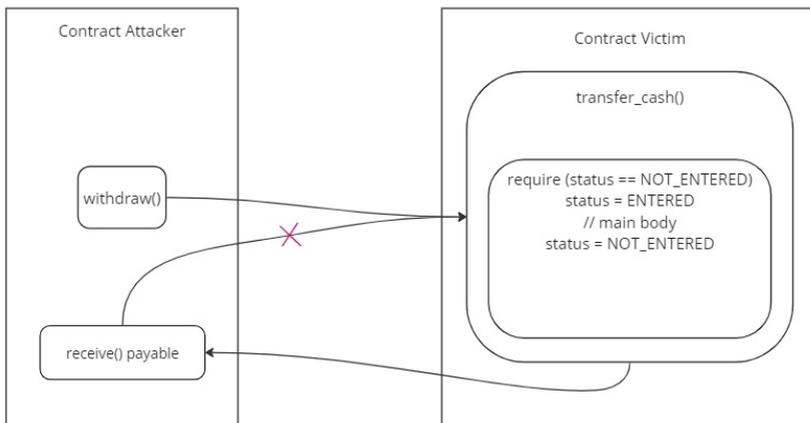


Рис. 2. Схема устранения уязвимости “Рекурсивный вызов”

В Solidity `require` — проверка условия в скобках, в случае его истинности транзакция продолжается, иначе откатывается.

На практике программисту необязательно именно таким образом устранять уязвимость. В Solidity есть лаконичный способ через определение собственных модификаторов, которыми можно помечать нужные функции. Однако, рекомендуется использовать готовую реализацию из стандартной библиотеки OpenZeppelin, представленной смарт-контрактом `ReentrancyGuard`, который необходимо импортировать и унаследовать от него разрабатываемый смарт-контракт и использовать модификатор `nonReentrant` на функциях, для которых должна быть устранена рассматриваемая уязвимость [3]. Общий вид кода можно представить следующим образом (листинг 1):

Листинг 1

Практическое устранение уязвимости “Рекурсивный вызов”

```
contract Victim is ReentrancyGuard {
    function transferCash() external nonReentrant {
        // основная бизнес-логика}}

```

2. Управление доступом

Неправильное управление доступом, вообще говоря, не является специфичной для смарт-контрактов проблемой. Однако, ввиду финансового характера транзакций, на нее стоит обратить пристальное внимание, так как довольно легко допустить эту уязвимость.

Как можно понять по названию, уязвимость заключается в том, что к определенной функциональности смарт-контракта есть доступ извне, который изначально разработчиком не предусматривался. Также важно упомянуть, что всё, что находится в блокчейне, вообще говоря, общедоступно. Другими словами, исходный код любого развернутого смарт-контракта можно просмотреть, что неопытный или наивный разработчик может не учитывать, реализовывая публичную функцию, которая предназначена для него или узкого круга лиц. Более того, практически любые развернутые смарт-контракты можно не только просматривать, но и использовать при написании своих, что легко может использовать злоумышленник, найдя уязвимый смарт-контракт [4]. Общая схема уязвимости приведена на рис. 3.

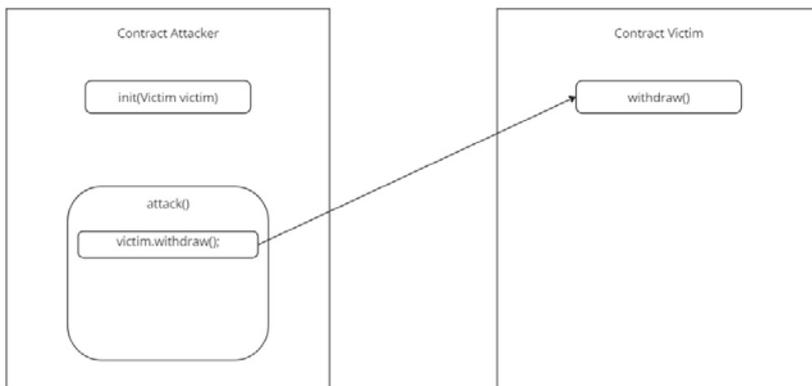


Рис. 3. Общая схема уязвимости “Управление доступом”

Смарт-контракт Attacker через конструктор получает экземпляр смарт-контракта Victim, и таким образом, получает возможность вызывать любую публичную или внешнюю функцию смарт-контракта жертвы. Например, перечислить криптовалюту со счета смарт-контракта жертвы на свой смарт-контракт, что в дальнейшем уже совершенно без проблем позволит распоряжаться полученной криптовалютой.

Устранить эту проблему не так сложно. Стандартным решением является проверка учетной записи, которая инициировала вызов

функции, на соответствие с адресом владельца смарт-контракта — учетной записи, с которой было осуществлено развертывание смарт-контракта в блокчейне. Здесь важно отметить, что смарт-контракт является разновидностью учетных записей, то есть не только физические владельцы криптокошельков имеют учетные записи. Дело в том, что согласно схеме на рисунке 3, вызывать функцию `withdraw()` будет смарт-контракт `Attacker`, и именно его адрес будет считаться адресом вызывающей функцию `withdraw()` учетной записи. Схема устранения уязвимости приведена на рис. 4.

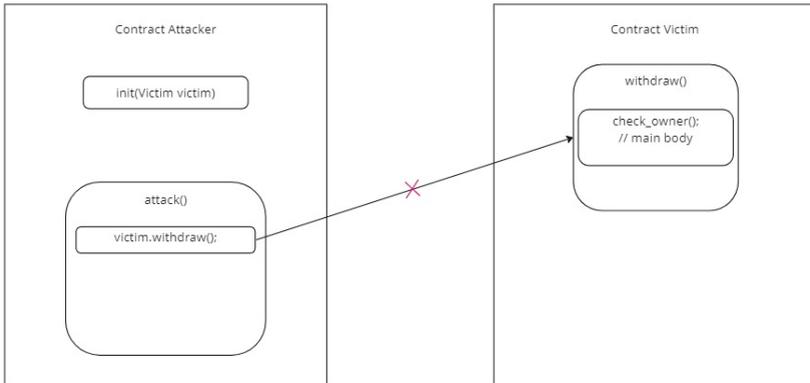


Рис. 4. Схема устранения уязвимости “Управление доступом”

Практически уязвимость устраняется совершенно аналогично рекурсивному вызову. Необходимо импортировать смарт-контракт `Ownable` из той же стандартной библиотеки `OpenZeppelin`, унаследовать от него разрабатываемый смарт-контракт и для нужных функций использовать модификатор `onlyOwner` [5]. Общий вид кода представлен в листинге 2

Листинг 2

Практическое устранение уязвимости “Управление доступом”

```
contract Victim is Ownable {
    function withdraw() external onlyOwner {
        // основная бизнес-логика}}

```

3. Арифметическое переполнение

Это известная практически любому программисту проблема, связанная с ограничением, накладываемым на память, выделяемую для хранения чисел.

Особенно важно задумываться об этой уязвимости при написании смарт-контрактов, ведь множество транзакций сопровождаются математическими операциями, а допущение арифметического переполнения может повлечь за собой катастрофические последствия, если это происходит при проведении финансовой операции перечисления криптовалюты.

Помимо этого, необходимо всегда помнить о публичности всех развернутых в блокчейне смарт-контрактов, что дает возможность злоумышленникам найти уязвимый смарт-контракт и использовать его в целях заработка путем провоцирования арифметического переполнения.

До некоторого времени проблема стояла довольно остро. До версии 0.8.0 язык программирования Solidity не мог безопасно проводить арифметические операции без подключения дополнительной библиотеки SafeMath, что позволяло неопытным, не знающим об особенностях языка и о нужной библиотеке разработчикам писать уязвимый код. Начиная с версии 0.8.0, Solidity уже неявно использует SafeMath, что помогает писать более надежные смарт-контракты [6].

Однако, существует обстоятельство, из-за которого об уязвимости не стоит забывать. Важно подчеркнуть, что проверка средствами языка Solidity на арифметическое переполнение — это такая же операция, как и любая другая, и за ее проведение тоже должна уплачиваться комиссия, а значит, чем больше математических операций, тем больше проверок на арифметическое переполнение, тем больше итоговая комиссия за всю транзакцию, что, безусловно, заставляет задуматься программисту о том, чтобы сэкономить на проверках и писать так называемые `unchecked` блоки, то есть блоки с кодом, которые не проверяются средствами языка в том числе на арифметическое переполнение по итогу математических операций. Злоупотребление этим и необдуманное использование `unchecked` блоков может привести к печальным последствиям, так как все смарт-контракты общедоступны, и когда-нибудь недобросовестный программист найдет ненадежно запрограммированный смарт-контракт и использует созданную уязвимость ради своей выгоды. Пример простейшего `unchecked` блока в Solidity приведен в листинге 3.

Листинг 3

Пример unchecked блока кода

```
uint8 v = 255;  
unchecked { v++; }
```

В приведенном примере после выполнения обеих строк кода переменная `v` будет хранить значение 0, и это никак не скажется на факте проведения транзакции. Без использования `unchecked` будет

произведен откат транзакции сразу же, как только обнаружится факт арифметического переполнения, то есть после инкремента переменной v .

Заключение

Данная статья посвящена анализу уязвимостей смарт-контрактов платформы Ethereum. Были приведены их общие схемы и стандартные методы устранения с конкретными примерами на языке программирования Solidity. Применение этих знаний позволяет разрабатывать относительно безопасные и надежные смарт-контракты для разного рода бизнес-проектов, в том числе NFT игр.

Список литературы

1. Loi, L. Making Smart Contracts Smarter / L. Loi, C. Duc-Hiep, O. Hrishu // CCS '16: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (Vienna, 24-28 October 2016). – Vienna, 2016. – P. 254-269.
2. DAO теряет миллионы долларов в час из-за ошибки в своём коде и тянет Ethereum за собой [Электронный ресурс]. – Режим доступа: <https://xakep.ru/2016/06/17/splitdao-attack/>
3. OpenZeppelin-contracts/ReentrancyGuard.sol [Электронный ресурс]. – Режим доступа: <https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/security/ReentrancyGuard.sol>
4. Уязвимости смарт-контрактов: как маленькая ошибка приводит к потере денег [Электронный ресурс]. – Режим доступа: <https://crypto-fox.ru/faq/uyazvimosti-smart-kontraktov-i-kak-s-nimi-borotsya/>
5. OpenZeppelin-contracts/Ownable.sol [Электронный ресурс]. – Режим доступа: <https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/access/Ownable.sol>
6. Solidity Documentation [Электронный ресурс]. – Режим доступа: https://docs.soliditylang.org/_/downloads/en/latest/pdf/

Разработка системы мониторинга и встраивания ЦВЗ в измененные текстовые файлы

А. А. Володина

Студент бакалавриата

А. В. Швырева

Ассистент

Введение

Изучение инцидентов в области информационной безопасности, связанных с утечкой конфиденциальной информации, показало, что за 2022 год в России зарегистрировано 710 случаев утечки данных, что более чем в 2.1 раза превышает количество утечек, зарегистрированных в прошлом году. При этом за 2021-2022 год количество сведений, позволяющих идентифицировать источник утечки, значительно уменьшилось. В 80% случаев определить внешний или внутренний вектор воздействия так и не удалось [1].

Таким образом, проблема предотвращения утечек и выявления нарушителей остается актуальной. В качестве полезной информации, позволяющей установить обстоятельства утечки, могут оказаться сведения об устройстве, на котором был открыт документ, учетной записи пользователя устройства, дате и времени создания. Для хранения подобной информации в самом текстовом файле можно использовать технологию внедрения цифровых водяных знаков (ЦВЗ). Встроенный ЦВЗ дает гарантию того, что файл был создан в определенной системе документооборота, а при необходимости существует возможность определения подлинности и целостности текстового документа.

Данная работа посвящена разработке программного средства, позволяющего производить мониторинг действий с файловой системой, а именно отслеживание действий с текстовыми файлами, а также встраивание ЦВЗ в измененный файл.

1. Анализ известных программ для мониторинга

Существует множество программ, позволяющих производить мониторинг файловой системы, выполнять некоторые действия с файлами, а также определенным образом реагировать на изменение

файлов. Для проведения сравнительного анализа были выбраны наиболее известные и распространенные из них:

- Process Monitor,
- Daemon FS,
- File Watcher Simple,
- Watch 4 Folder,
- The Folder Spy,
- Simple Watcher.

В качестве критериев для сравнения были выбраны кросс-платформенность, возможность наблюдения за несколькими файлами или целым диском, фильтрации типов отслеживаемых файлов, выполнения определенных действий при изменениях в файловой системе и сохранения логов. Результаты проведенного сравнительного анализа представлены в таблице 1.

Таблица 1

Сравнительный анализ программ для мониторинга

		Программы					
		Filemon (Process Monitor)	Daem on FS	File Watche rSimpl e	Watch 4 Folder	The Folde r Spy	Sim ple Wat cher
Критерии	Кросс-платформенность	-	+	+	-	-	-
	Наблюдение нескольких папок	+	+	+	+	+	-
	Наблюдение всего диска	+	+	+	+	+	+
	Фильтрация типов файлов	+	-	+	-	+	+
	Выполнение действий при изменениях	+	-	+	+	+	+
	Сохранение логов	+	-	+	+	+	+

2. Обзор стеганографических методов

Стеганографические методы скрытия информации разнообразны и могут разделяться по типу контейнера и по способу хранения информации.

Методы текстовой (или лингвистической) стеганографии основываются на изменении формата текста, а также естественную избыточность языка. Выделяют следующие виды методов лингвистической стеганографии [2]:

- методы изменения оформления и форматирования текстового документа,
- синтаксические методы,
- семантические методы.

Синтаксические методы лингвистической стеганографии изменяют знаки пунктуации, стиль и структуру текста. Скрытие данных в семантических методах реализуется с использованием таблицы синонимов. Среди методов изменения оформления и форматирования текстового документа распространено совершение манипуляций с пробелами.

При работе с текстовыми файлами кроме методов текстовой стеганографии могут применяться методы, основанные на использовании специальных свойств файловых форматов, особенности и возможности которых определяются особенностями форматов данных.

Рассмотрим самый распространенный формат хранения текстовых документов - Portable Document Format (PDF). Для формата PDF можно выделить следующие виды методов стеганографического скрытия:

- методы скрытия в содержании документа,
- методы скрытия в структуре документа.

Методы скрытия в содержании документа заключаются в различных изменениях размера и вида текста, обладают широким многообразием вариантов реализации, но могут быть легко обнаружены.

Методы скрытия в структуре документа PDF могут использовать особенности строения и синтаксиса документа, например, использование возможности оставлять комментарии с помощью символа процента (%), встраивать небольшие количества информации перед указанием формата документа. Использование таблицы перекрестных ссылок, дает возможность задействовать «удаленные» объекты, добавлять и изменять ссылки на объекты и многое другое [3].

3. Алгоритмы встраивания ЦВЗ

Рассмотрим несколько алгоритмов внедрения ЦВЗ разного типа.

Реализуемый метод стеганографического скрытия в текстовом контейнере основывается на методе двойных пробелов, алгоритм

которого представлен на рисунке 1. Преимуществами выбранного метода являются вариативность встраиваемого символа, возможность считывания из печатной и электронной версии, устойчивость к орфографическим проверкам, устойчивость к сжатию, отсутствие необходимости в словарях. Извлечение ЦВЗ проводится в той же последовательности: находится пробел, проверяется наличие дополнительного символа, после чего битовая последовательность преобразуется в строку.

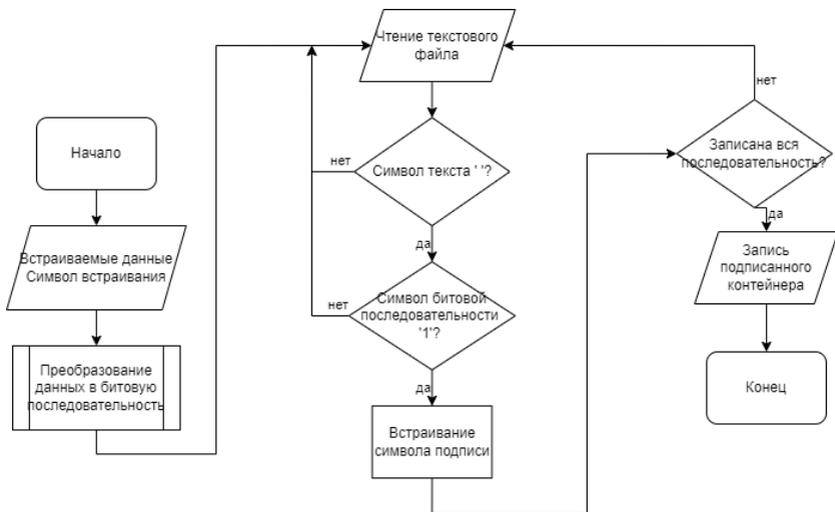


Рис. 1. Алгоритм встраивания ЦВЗ методом дополнительного символа к пробелу

Реализуемый метод стеганографического скрытия в формате PDF основывается на методе скрытия в структуре документа путем удаления страниц, алгоритм которого представлен на рисунке 2. Преимуществами выбранного метода являются устойчивость к сжатию, устойчивость к визуальным проверкам, независимость временной сложности скрытия от встраиваемой информации, независимость объема встраиваемой информации от объема текстового файла. Извлечение ЦВЗ проводится путем нахождения последней «удаленной» страницы и расшифровкой ее содержимого.

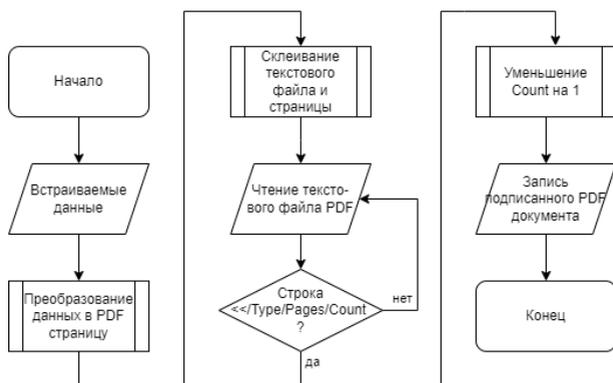


Рис. 2. Алгоритм встраивания ЦВЗ методом удаления страниц

4. Проектирование и разработка программного модуля

В основе системы мониторинга и встраивания ЦВЗ в измененные текстовые файлы лежит алгоритм, представленный на рисунке 3.

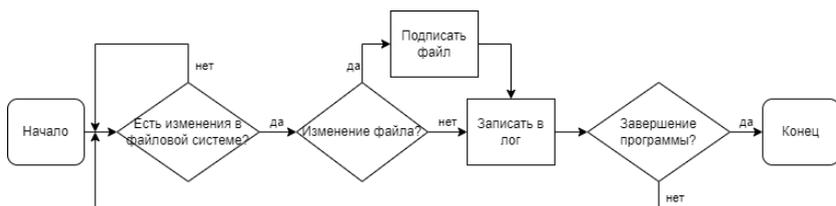


Рис. 3. Алгоритм реализуемого программного средства

В качестве средств реализации программного модуля использовался язык C#. При реализации мониторинга файловой системы использовался класс FileSystemWatcher, который отслеживает каталоги, подкаталоги и файлы на предмет изменений, а также при необходимости имеет возможность инициировать события, при выполнении действий с файлом. Интерфейс программы был реализован средствами платформы Windows Form и представлен на рисунке 4.

Интерфейс состоит из нескольких страниц. На первой странице пользователю предоставлена возможность запустить или остановить мониторинг части или всей файловой системы с встраиванием ЦВЗ в измененные текстовые файлы. Также на этой странице динамически отображается лог с данными о пользователе, дате, времени, виде его действий и путь к файлу, с которым выполнялось действие. Эти же

данные записываются в специальный лог-файл. Вторая и третья страницы предоставляют возможность извлечения ЦВЗ.

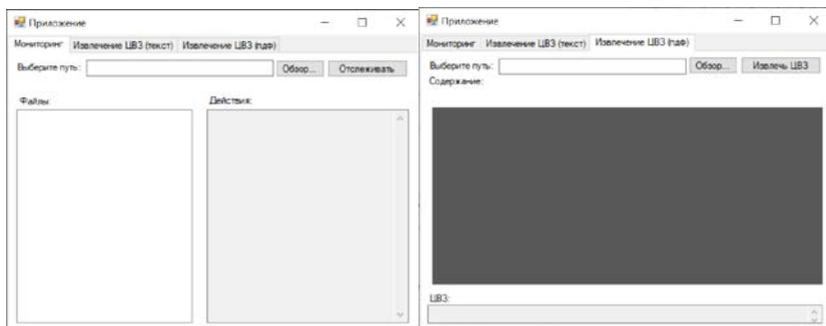


Рис. 4. Интерфейс мониторинга и извлечения ЦВЗ из PDF файла

5. Тестирование разработанного программного модуля

Возможности реализованных методов стеганографического встраивания информации в текстовый файл представлены в табл. 2.

Таблица 2

Результаты тестирования

Методы встраивания		Добавление символа к пробелу	Скрытие PDF страницы
		Критерии	Зависимость объема встраиваемой информации от содержания файла
Устойчивость к сжатию	+		+
Устойчивость к визуальным проверкам	+		+
Устойчивость к отображению скрытых элементов	-		+
Сохранение ЦВЗ при печати/фотографии	+		-

На рисунках 5 и 6 представлен пример тестирования разработанного программного модуля.

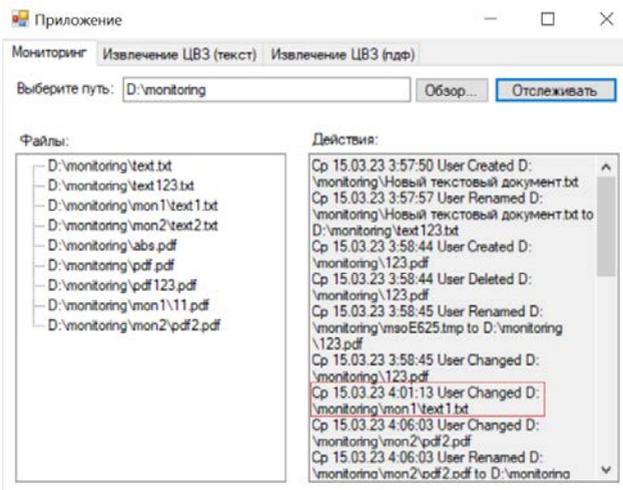


Рис. 5. Выполнение мониторинга и фоновое встраивание ЦВЗ при изменениях в файловой системе

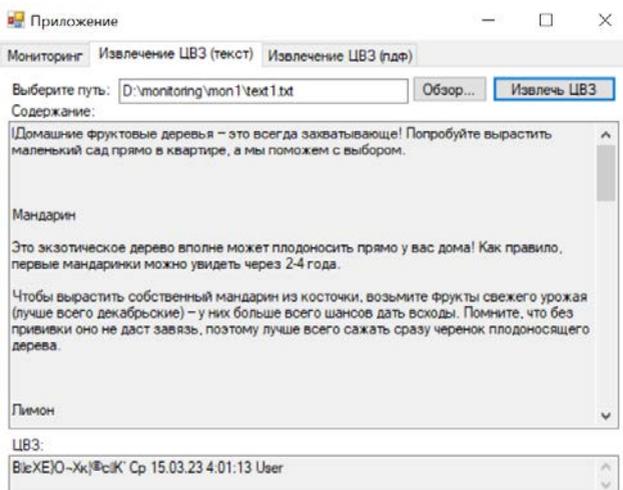


Рис. 6. Извлечение встроенного ЦВЗ

Заключение

Данная работа посвящена разработке системы мониторинга и встраивания ЦВЗ в измененные текстовые файлы. В статье рассмотрены существующие программы для мониторинга файловой системы, а также

проанализированы методы стеганографического скрывания в текстовых файлах, выявлены их преимущества и недостатки. В результате был разработан программный модуль способный осуществлять мониторинг, логирование, встраивание и извлечение ЦВЗ в измененные текстовые файлы различными стеганографическими методами. Анализ разработанного средства выявил ряд преимуществ и недостатков используемых алгоритмов, которые стоит учитывать при использовании модуля.

Список литературы

1. Россия: утечки информации ограниченного доступа в 2022 г [Электронный ресурс] : аналитический отчет / Экспертно-аналитический центр InfoWatch. – Режим доступа: <https://www.infowatch.ru/sites/default/files/analytics/files/utechki-informatsii-ogranichennogo-dostupa-v-rossii-za-2022-god.pdf>
2. Семантический и лексико-синтаксический подходы в стеганографии [Электронный ресурс] : А. В. Джунковский. – Режим доступа: <https://openbooks.itmo.ru/ru/file/8407/8407.pdf>
3. Document management – Portable document format – Part 1: PDF 1.7 [Электронный ресурс] : Adobe Systems Incorporated. – Режим доступа: https://opensource.adobe.com/dc-acrobat-sdk-docs/pdfstandards/PDF32000_2008.pdf

Повышение результативности работы отделов продаж заказчиков

М. И. Воронцова

Студент бакалавриата

И. В. Абрамов

Доцент

Введение

В настоящее время компаниям, чья деятельность связана с реализацией различной продукции и предоставлением услуг, необходимо постоянно улучшать работу отделов продаж для роста экономической составляющей организации. По мере своего развития, предприятие может сталкиваться с различными новыми задачами и проблемами, для решения которых в заведении может отсутствовать необходимая экспертиза. Тогда организации необходимо расширить уже имеющиеся информационные системы. Качественная поддержка данных сервисов может помочь в улучшении имиджа компании и продуктов предприятия, за счет отзывов, рекомендаций и обратной связи от пользователей.

Проблемы, возникающие по мере эксплуатации, необходимо вовремя и правильно устранять, чтобы не терять конкурентное преимущество. В достижении данной цели полезными инструментами могут служить различные математические методы, связанные с системным подходом к сложным проблемам принятия решений.

В данной работе рассматривается задача применения метода анализа иерархии в сфере продаж. Метод успешно применяется в задачах ранжирования – многокритериального упорядочивания заданного множества альтернатив. Эта проблема рассматривается в настоящей статье.

1. Формирование иерархии целей

Постановка задачи: предприятие воспользовалось услугами разработки сервисов для улучшения качества работы отделов продаж. Компании необходимо сформировать наиболее эффективную последовательность решения задач, возникающих в ходе эксплуатации различных сервисов системы.

При формировании иерархии целей получена модель, представленная на рисунке: на верхнем уровне находится глобальная цель, а именно – выбор наиболее приоритетной задачи, на втором уровне представлен переход к основным критериям (их описание представлено в табл. 1), на третьем уровне показаны альтернативы, в качестве которых в данном задании выступают различные задачи.



Рисунок. Иерархия целей

Таблица 1

Критерии для цели
«Определение последовательности решения задач»

№	Название критерия	Описание критерия
1	Приоритет заявителя	Приоритетность задачи для заявителя
2	Критичность	Важность сервиса (ИТ-услуги) для корректной работы бизнес-приложений. Влияние на аудиторию приложения
3	Трудоемкость	Количество времени, необходимое для решения задачи
4	Источник задачи	Уровень заявителя в бизнес-процессе
5	Категория	Группа, к которой относится поставленная задача (ошибка\запрос на обслуживание\консультация)
6	Массовость	Количество однотипных задач

Выявление критериев является одной из основных задач при использовании метода анализа иерархии. Критерии, представленные в

табл. 1, дают возможность оценить каждую заявку и сформировать эффективную последовательность их решения.

2. Определение приоритетов

Следующий шаг – определение приоритетов представленных критериев. Веса критериев устанавливаются на основе экспертной оценки содержания рабочих задач сотрудников поддержки систем и сервисов отдела продаж. Для определения приоритетов критериев используется шкала Саати, представленная в табл. 2.

Таблица 2

Шкала Т.Л.Саати

Баллы	Определение	Объяснение
1	Равная важность	Важность объектов (факторов) A_i и A_j одинакова
3	Умеренное превосходство одного над другим	Опыт и суждения дают легкое превосходство одному объекту (фактору) над другим
5	Существенное или сильное превосходство	Имеющиеся данные свидетельствуют о заметном превосходстве A_i над A_j
7	Очень сильное превосходство	Превосходство объекта (фактора) A_i над A_j очевидно
9	Абсолютное превосходство	Очевидность превосходства A_i над A_j подтверждается всеми имеющимися признаками
2,4,6,8	Промежуточные решения между двумя соседними суждениями	Применяются в компромиссных случаях

3. Расчет локальных векторов приоритетов

После установки экспертной оценки, ранее поставленные критерии попарно сравниваются. Результаты оценки и нормализованный вектор приоритетов, который определяет вес критерия или же степень значимости, представлены в табл. 3.

$$w_i = \frac{a_i}{\sum a_j} \quad (1)$$

Где a_i – среднее геометрическое i – строки матрицы, $\sum a_i$ – сумма средних геометрических.

Сравнение критериев и степень значимости

	Приоритет	Критичность	Источник задачи	Категория	Массовость	Трудоемкость	Нормализованный вектор приоритетов
Приоритет заявителя	1	1/5	1/3	3	1/3	4	0,088
Критичность	5	1	7	7	3	9	0,474
Источник задачи	3	1/7	1	3	1/5	5	0,114
Категория	1/3	1/7	1/3	1	1/5	3	0,050
Массовость	3	1/3	5	5	1	5	0,245
Трудоемкость	1/4	1/9	1/5	1/3	1/5	1	0,029

4. Проверка ограниченности оценки приоритетов

Для проверки логичности суждений необходимо рассчитать три характеристики:

- собственное значение матрицы;
- индекс согласованности;
- отношение согласованности.

Формулы для расчета индекса согласованности и отношения согласованности:

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

$$ОС = \frac{ИС}{СС} \quad (3)$$

При расчете отношения согласованности используется случайная согласованность, которая для таблицы размерностью 6 равна 1,24.

Результаты расчетов:

СЗМ= 6,572, ИС=0,114, ОС=0,092

Так как ОС составляет менее 10% (9,2%) – оценки в матрице считаются согласованными, нарушение логичности суждений не выявлено.

5. Синтез приоритетов. Подбор наиболее эффективной последовательности решения задач

В данном задании альтернативами являются четыре задачи, содержащие следующие проблемы:

1. Задача 1

Клиент авторизовался на портале интернет-магазина. При выборе желаемого товара на главной странице сайта отображается цена с персональной скидкой для авторизованного пользователя. При добавлении желаемого продукта в свою пользовательскую корзину онлайн-магазина с целью последующего приобретения, покупатель наблюдает изменение цены на материал и отмену персональной скидки. Клиент создает обращение в поддержку интернет-магазина с целью выяснения проблемы изменения цены. При проверке сотрудниками первой линии поддержки обращений клиентов совместно с сотрудниками поддержки сервисов отображения товаров на главной странице сайта выясняется, что наблюдается проблема в сервисе расчета корзины клиента, а именно – выбранная акция не применяется к конкретному товару. На онлайн-витрине магазина ошибки нет. Требуется проанализировать и выявить причины ошибки в сервисе расчета заказа.

2. Задача 2

ВИП клиент авторизовался под своей учетной записью на портале интернет-магазина. При добавлении желаемых товаров в свою пользовательскую корзину онлайн-магазина ВИП покупатель встречается с проблемой увеличения цены на товар. Стоимость материалов в корзине отличается от ценника, представленного на главной странице сайта. Клиент оформляет заявку в поддержку интернет-магазина с целью анализа отличия цены на товар в каталоге приложения и в пользовательской корзине. При изучении инцидента сотрудниками поддержки обращений ВИП клиентов, а также специалистами поддержки отображений товаров на сайте приложения удалось локализовать проблему в сервисе расчета корзины покупателя. Ошибка связана с отменой персональной скидки ВИП пользователя. Цена на главной странице приложения отображается правильная. Необходимо выяснить причины ошибки в сервисе расчета заказа, из-за которой наблюдается отображение некорректной цены товара в корзине ВИП клиента.

3. Задача 3

В интернет-магазине запускается распродажа определенной категории товаров на всю торговую сеть. Скидка доступна для каждого покупателя во всех регионах. Акционная цена отображается на витрине

онлайн-магазина. При этом в поддержку пользователей приложения поступают жалобы от покупателей, связанные с пересчетом цены в корзине интернет-магазина. На основании анализа со стороны сотрудников отдела отображения данных на сайте удалось установить, что проблема связана с отменой скидки на акционный товар в корзине покупателей. Требуется провести анализ на стороне сервиса расчета заказа и установить причины, по которым на товары не применяется акция.

4. Задача 4

Поступило обращение от сотрудника технической поддержки смежной системы, а именно – сайта интернет-магазина. Авторизованные под своей учетной записью клиенты добавили товары в пользовательскую корзину онлайн-магазина. При добавлении продуктовых позиций в корзину отображалась цена по персональной скидке. Спустя несколько дней покупатели решили приобрести добавленные ранее товары и столкнулись с увеличением стоимости корзины. При анализе на стороне сотрудников поддержки отображения данных в web-приложении удалось установить причину изменения стоимости продуктов, которая связана с отменой персональной скидки для данных пользователей. Необходима консультация, связанная с вопросом корректности отмены скидки на выбранные товары.

После попарной оценки каждой задачи по ранее определенным критериям, благодаря методу анализа иерархии была сформирована табл. 3, в которой отражается глобальный приоритет каждой задачи. Локальные приоритеты каждой альтернативы перемножаются на приоритеты соответствующего критерия и суммируются по каждому элементу.

Таблица 4

Итоговая оценка альтернатив

	Критерии						Глобальный приоритет
	Приоритет	Критичность	Источник задачи	Категория	Массовость	Трудоемкость	
	0,088	0,474	0,114	0,05	0,245	0,029	
Задача 1	0,094	0,084	0,051	0,221	0,057	0,05	0,080
Задача 2	0,219	0,155	0,525	0,057	0,057	0,114	0,173
Задача 3	0,645	0,712	0,212	0,664	0,664	0,581	0,631
Задача 4	0,041	0,049	0,212	0,057	0,221	0,255	0,115

На основании представленных выше расчетов можно составить эффективную последовательность решения задач:

1. Задача 3
2. Задача 2
3. Задача 4
4. Задача 1

Данная последовательность, составленная на базе анализа содержания проблем, возникающих в практике работы сотрудников, позволяет определить задачи, которые имеют большее влияние на систему и которые необходимо решать в первую очередь.

Заключение

В данной статье рассматривается задача применения метода анализа иерархии в сфере продаж, а именно: выявление категорий критериев и их ценности на базе анализа содержания задач, возникающих в практике работы, и, соответственно, применение метода анализа иерархии для нахождения оптимальной последовательности решения входящих заявок.

Благодаря структурированию потока входящих заявок, определения приоритетов и рассмотрению основных факторов, влияющих на эффективность решения обращений клиентов, можно повысить производительность работы поддержки сервисов, связанных с отделами продаж. Улучшение качества работы и обслуживания систем, задействованных в процессе торговли, может помочь компании сохранять и улучшать свою конкурентоспособность на рынке сбыта.

В дальнейшем можно рассмотреть возможность применения данного метода для других отделов компании, а также возможность его автоматизации.

Список литературы

1. Лукич Р. М. Управление продажами / Р. М. Лукич - М. : Альпина Паблшер, 2013. – 256 с.
2. Саати, Т. Л. Принятие решений при зависимостях и обратных связях. Аналитические сети / Т. Л. Саати. – М. : Ленанд, 2021. – 360 с.
3. Саати, Т. Л. Принятие решений. Метод анализа иерархий / Т. Л. Саати. – М. : Радио и связь, 1993. – 167 с.

Разработка моделей оптимизации управления проектами

И. В. Горбачева

Студент бакалавриата

Н. А. Алейникова

Доцент

Введение

Под проектом понимается комплекс операций, в результате выполнения которых к заданному сроку должна быть достигнута некоторая цель. При этом могут быть ограничены какие-либо ресурсы, например, время, деньги, человеческие ресурсы или другие. Особенность проектов состоит в том, что это разовые, неповторяющиеся мероприятия.

Одной из важнейших характеристик проекта является критический путь и его продолжительность. Критический путь – это последовательность операций от начала до конца проекта, имеющая из всех путей наибольшую продолжительность по времени.

В сетевом планировании и управлении проектами возникают различные оптимизационные задачи, связанные с улучшением организации выполнения комплекса операций. В частности, необходимость в таких задачах возникает для достижения оптимального соотношения между продолжительностью критического пути проекта и ресурсами, требуемыми для его выполнения.

Оптимизационные задачи сетевого планирования можно решать, например, алгоритмически, с помощью сетевого графика проекта. Но этот способ является слишком громоздким в случае большого количества операций, также отсутствует аналитическая связь между результатом и входными данными, что усложняет исследования модели. Еще одним способом решения оптимизационных задач сетевого планирования является их формализация в виде моделей математического программирования.

На стадии планирования проекта часто нельзя точно указать сроки выполнения операций. Существует два основных способа определения продолжительностей операций в условиях неопределенности – на основе статистических методов и с помощью экспертов. Для применения статистических методов требуется статистика по срокам

выполнения операций, которой часто нет, так как проекты уникальны. Поэтому для определения продолжительностей операций можно использовать опыт и знания экспертов. Но экспертам сложно задать сроки тех или иных операций в виде четких чисел. Естественнее представлять их в виде числовых интервалов, отдельные значения из которых могут быть указаны экспертом с разной степенью уверенности. Наиболее подходящим способом для задания такого рода оценок экспертами является использование нечеткой или лингвистической переменных [1].

В данной работе будет проведена формализация задачи оптимизации комплекса операций по времени за счет привлечения дополнительных средств в условиях неопределенности. Параметры задачи, такие как продолжительности операций, будут заданы в виде треугольных нечетких чисел. Для решения этой задачи с нечеткими параметрами предлагается использовать W -алгебру и W -числа. Это позволит упростить решение, устранить искажение естественных свойств и отношений классических моделей.

1. Формализация задачи сетевого планирования в условиях неопределенности в виде задачи линейного программирования и разработка алгоритма решения с помощью W -чисел

Задачу нахождения критического пути можно формализовать в виде задачи линейного программирования

$$Z = y_n - y_1 \rightarrow \min, \quad (1)$$

$$y_j - y_i \geq t_{ij}, \forall (i, j) \in P, \quad (2)$$

где y_i - моменты начала операции (i,j) , t_{ij} - длительность операции (i,j) , P - множество операций (работ).

Ограничения (2) позволяют учитывать порядок выполнения работ с помощью отношения предшествования.

Так как на стадии планирования проекта нельзя точно указать сроки выполнения операции, то удобно для этого использовать экспертную оценку продолжительности в виде нечеткого треугольного числа $\tilde{t}_{ij} = (t_{ij}^L, t_{ij}^m, t_{ij}^R)$. Функция принадлежности для таких чисел запишется следующим образом:

$$\mu_{\tilde{t}_{ij}}(t_{ij}) = \begin{cases} \mu_{\tilde{t}_{ij}}^L = \frac{t_{ij} - t_{ij}^L}{t_{ij}^m - t_{ij}^L}; & t_{ij} \in [t_{ij}^L; t_{ij}^m], \\ \mu_{\tilde{t}_{ij}}^R = \frac{t_{ij} - t_{ij}^R}{t_{ij}^m - t_{ij}^R}; & t_{ij} \in [t_{ij}^m; t_{ij}^R], \\ 0, & \text{иначе,} \end{cases} \quad (3)$$

где t_{ij}^m – мода (наиболее возможная продолжительность операции); t_{ij}^L, t_{ij}^R – наименьшая и наибольшая наименее возможная продолжительность операции; $\mu_{\tilde{t}_{ij}}(t_{ij})$ - степени принадлежности [1].

Тогда в нечеткой постановке задача определения критического пути (1)-(2) запишется следующим образом [2]:

$$\tilde{Z} = \tilde{y}_n - \tilde{y}_1 \rightarrow \min, \quad (4)$$

$$\tilde{y}_j - \tilde{y}_i \geq \tilde{t}_{ij}, \forall (i, j) \in P, \quad (5)$$

Для решения задачи (4)–(5) будем использовать W-алгебру и W-числа.

W-число – это вектор $(v^L(\alpha); v^R(\alpha))$, где $v^L(\alpha) \in X$, $v^R(\alpha) \in X$ и $v^L(1) = v^R(1)$, α - вещественный параметр, $\alpha \in [0; 1]$.

Нечеткие продолжительности работ $\tilde{t}_{ij} = (t_{ij}^L, t_{ij}^m, t_{ij}^R)$ в виде W-числа запишем следующим образом:

$$\tilde{t}_{ij} = (t_{ij}^L(\alpha), t_{ij}^R(\alpha)) = (t_{ij}^L + (t_{ij}^m - t_{ij}^L)\alpha, t_{ij}^R + (t_{ij}^m - t_{ij}^R)\alpha), \quad (6)$$

Для представления результатов моделирования в виде четкого числа применим дефаззификацию нечетких чисел [3]. Формула для дефаззификации нечеткой продолжительности работ:

$$t_{ij} = \frac{\int_{t_{ij}^L}^{t_{ij}^m} x \left(\frac{x - t_{ij}^L}{t_{ij}^m - t_{ij}^L} \right) dx + \int_{t_{ij}^m}^{t_{ij}^R} x \left(\frac{t_{ij}^R - x}{t_{ij}^R - t_{ij}^m} \right) dx}{\int_{t_{ij}^L}^{t_{ij}^m} \left(\frac{x - t_{ij}^L}{t_{ij}^m - t_{ij}^L} \right) dx + \int_{t_{ij}^m}^{t_{ij}^R} \left(\frac{t_{ij}^R - x}{t_{ij}^R - t_{ij}^m} \right) dx} \quad (7)$$

Алгоритм решения задачи нечеткого сетевого планирования с W-числами будет состоять из следующих шагов:

Шаг 1. Решить задачу (4)-(5) отдельно для каждого значения $\tilde{t}_{ij} : t_{ij}^L; t_{ij}^m; t_{ij}^R$.

Шаг 2. Полученные решения записать в виде W-чисел (6).

Шаг 3. Провести дефаззификацию для получения четкого числа по формуле (7).

2. Формализация задачи оптимизации комплекса операций по времени за счет привлечения дополнительных средств.

Целью решения данной задачи является сведение продолжительности критического пути к минимуму с помощью привлечения дополнительных ресурсов.

Математическая модель этой оптимизационной задачи запишется следующим образом:

$$T = \sum_{i=1}^n t_{pi} \rightarrow \min \quad (8)$$

$$\left\{ \begin{array}{l} t_{pj} - t_{pi} \geq t_{ij}^{\min}, \forall (i, j) \in P, \\ t_{pj} - t_{pi} \geq t_{ij}(1 - k_{ij}x_{ij}), \forall (i, j) \in P, \\ \sum_{(i,j) \in P} x_{ij} \leq B, \\ x_{ij} \geq 0, t_{pi} \geq 0, \forall (i, j) \in P, \end{array} \right. \quad (9)$$

где t_{ij}^{\min} – минимальные возможные продолжительности выполнения работ; x_{ij} – дополнительно вложенные средства в каждую работу; t_{pj}, t_{pi} – предшествующие и последующие ранние сроки событий; B – сумма всех затрат; P – совокупность всех операций; k_{ij} – удельные затраты, которые находятся по формуле

$$k_{ij} = \frac{t_{ij} - t_{ij}^{\min}}{C_{ij}^{\max}}, \quad (10)$$

и показывают, как уменьшатся продолжительности работ с увеличением дополнительных затрат на 1 единицу; C_{ij}^{\max} – величина

дополнительных затрат при минимальных вложениях дополнительных средств.

В рассматриваемой оптимизационной задаче требуется найти t_{pj} – ранние сроки событий и x_{ij} – дополнительно вложенные средства в каждую операцию при сведении продолжительности критического пути к минимуму за счет привлечения дополнительных средств, сумма которых не должна превышать величину B . При этом каждая операция должна выполняться за время, не меньшее, чем t_{ij}^{\min} .

Формула для нахождения новых продолжительностей работ:

$$t_{ij}^* = t_{ij} * (1 - k_{ij} x_{ij}). \quad (11)$$

С нечеткими продолжительностями операций задача (8)-(9) будет выглядеть следующим образом:

$$\begin{cases} \tilde{T} = \sum_{i=1}^n \tilde{t}_{pi} \rightarrow \min & (12) \\ \left\{ \begin{array}{l} \tilde{t}_{pj} - \tilde{t}_{pi} \geq t_{ij}^{\min}, \forall (i, j) \in P, \\ \tilde{t}_{pj} - \tilde{t}_{pi} \geq \tilde{t}_{ij} (1 - k_{ij} x_{ij}), \forall (i, j) \in P, \\ \sum_{(i,j) \in P} x_{ij} \leq B, \\ x_{ij} \geq 0, \tilde{t}_{pi} \geq 0, \forall (i, j) \in P, \end{array} \right. & (13) \end{cases}$$

Новые продолжительности работ будут находиться по следующей формуле:

$$\tilde{t}_{ij}^* = \tilde{t}_{ij} * (1 - k_{ij} x_{ij}). \quad (14)$$

3. Пример решения задачи оптимизации комплекса операций по времени за счет привлечения дополнительных средств в условиях неопределенности

Рассмотрим решение оптимизационной задачи сетевого планирования на примере проекта строительства некоторого сооружения, исходные данные для которого представлены в таблице.

Исходные данные

Операция/Код (i,j)	Описание	Предшественник	Нечеткая продолжительность операций (дней), \tilde{t}_{ij}			Стоимость операции, при ее нормальной продолжительности	Удельные затраты, k_{ij}
			t_{ij}^L	t_{ij}^m	t_{ij}^R		
A (1,3)	Установить взрывные снаряды		3	5	8	6	0,12
B (1,2)	Эвакуировать окружение		2	4	5	20	0,1
C(1,4)	Подготовить колонну грузовиков		2	3	6	1	0,1
D(3,4)	Взорвать здание	A,B	1	1	3	4	0,08
E (4,5)	Разборка развалин и вывоз мусора	C,D	5	7	9	30	0,2
F(5,6)	Вырыть котлован	E	9	12	14	42	0,3
G (5,7)	Подвести коммуникации	E	13	15	17	60	0,2
H (6,11)	Заливка бетона в фундамент	F	8	10	12	8	0,05
I (7,8)	Возвести металлический каркас	F,G	6	8	10	15	0,2
J (8,9)	Электромонтажные работы	I	12	15	18	35	0,08
K (8,10)	Настелить пол и возвести стены	I	16	20	25	45	0,13
L(8,11)	Установить лифты	I	6	7	9	20	0,12
M (11,12)	Отделочные работы	H,J,K, L	12	14	16	40	0,2

Сумма всех затрат по дополнительным средствам В равна 10.

Решим задачу оптимизации комплекса операций по времени за счет привлечения дополнительных средств (12)-(13). Математическая модель запишется следующим образом:

$$\begin{aligned} \tilde{T} = & \tilde{t}_{p1} + \tilde{t}_{p2} + \tilde{t}_{p3} + \tilde{t}_{p4} + \tilde{t}_{p5} + \tilde{t}_{p6} + \tilde{t}_{p7} + \tilde{t}_{p8} + \\ & + \tilde{t}_{p9} + \tilde{t}_{p10} + \tilde{t}_{p11} + \tilde{t}_{p12} \rightarrow \min \end{aligned} \quad (15)$$

$$\left\{ \begin{aligned} \tilde{t}_{p3} - \tilde{t}_{p1} &\geq 3, & \tilde{t}_{p3} - \tilde{t}_{p1} &\geq (3 + 2\alpha, 8 - 3\alpha) * (1 - x_{13} * 0.12), \\ \tilde{t}_{p2} - \tilde{t}_{p1} &\geq 3, & \tilde{t}_{p2} - \tilde{t}_{p1} &\geq (2 + 2\alpha, 5 - 1\alpha) * (1 - x_{12} * 0.1), \\ \tilde{t}_{p4} - \tilde{t}_{p1} &\geq 1, & \tilde{t}_{p4} - \tilde{t}_{p1} &\geq (2 + 1\alpha, 6 - 3\alpha) * (1 - x_{14} * 0.1), \\ \tilde{t}_{p3} - \tilde{t}_{p2} &\geq 0, & \tilde{t}_{p3} - \tilde{t}_{p2} &\geq 0 * (1 - x_{23} * 0), \\ \tilde{t}_{p4} - \tilde{t}_{p3} &\geq 1, & \tilde{t}_{p4} - \tilde{t}_{p3} &\geq (1 + 0\alpha, 3 - 2\alpha) * (1 - x_{34} * 0.08), \\ \tilde{t}_{p5} - \tilde{t}_{p4} &\geq 5, & \tilde{t}_{p5} - \tilde{t}_{p4} &\geq (5 + 2\alpha, 9 - 2\alpha) * (1 - x_{45} * 0.2), \\ \tilde{t}_{p6} - \tilde{t}_{p5} &\geq 10, & \tilde{t}_{p6} - \tilde{t}_{p5} &\geq (9 + 3\alpha, 14 - 2\alpha) * (1 - x_{56} * 0.3), \\ \tilde{t}_{p7} - \tilde{t}_{p5} &\geq 11, & \tilde{t}_{p7} - \tilde{t}_{p5} &\geq (13 + 2\alpha, 17 - 2\alpha) * (1 - x_{57} * 0.2), \\ \tilde{t}_{p7} - \tilde{t}_{p6} &\geq 0, & \tilde{t}_{p7} - \tilde{t}_{p6} &\geq 0 * (1 - x_{67} * 0), \\ \tilde{t}_{p11} - \tilde{t}_{p6} &\geq 7, & \tilde{t}_{p11} - \tilde{t}_{p6} &\geq (8 + 2\alpha, 12 - 2\alpha) * (1 - x_{6,11} * 0.05), \\ \tilde{t}_{p8} - \tilde{t}_{p7} &\geq 6, & \tilde{t}_{p8} - \tilde{t}_{p7} &\geq (6 + 2\alpha, 10 - 2\alpha) * (1 - x_{78} * 0.2), \\ \tilde{t}_{p9} - \tilde{t}_{p8} &\geq 10, & \tilde{t}_{p9} - \tilde{t}_{p8} &\geq (12 + 3\alpha, 18 - 3\alpha) * (1 - x_{89} * 0.08), \\ \tilde{t}_{p10} - \tilde{t}_{p8} &\geq 13, & \tilde{t}_{p10} - \tilde{t}_{p8} &\geq (16 + 4\alpha, 25 - 5\alpha) * (1 - x_{8,10} * 0.13), \\ \tilde{t}_{p11} - \tilde{t}_{p8} &\geq 4, & \tilde{t}_{p11} - \tilde{t}_{p8} &\geq (6 + 1\alpha, 9 - 2\alpha) * (1 - x_{8,11} * 0.12), \\ \tilde{t}_{p11} - \tilde{t}_{p9} &\geq 0, & \tilde{t}_{p11} - \tilde{t}_{p9} &\geq 0 * (1 - x_{9,11} * 0), \\ \tilde{t}_{p11} - \tilde{t}_{p10} &\geq 0, & \tilde{t}_{p11} - \tilde{t}_{p10} &\geq 0 * (1 - x_{10,11} * 0), \\ \tilde{t}_{p12} - \tilde{t}_{p11} &\geq 10, & \tilde{t}_{p12} - \tilde{t}_{p11} &\geq (12 + 2\alpha, 16 - 2\alpha) * (1 - x_{11,12} * 0.2), \\ & & \sum_{(i,j) \in P} x_{ij} &\leq B, \quad x_{ij} \geq 0, \tilde{t}_{pi} \geq 0, \end{aligned} \right. \quad (16)$$

Результаты задачи (15)-(16) для примера строительства некоторого сооружения показаны на рисунке и также их можно выразить в естественной интерпретации:

- до привлечения дополнительных средств продолжительность выполнения всего проекта была равна (56, 70, 88), в виде W-числа (56+14 α , 88-18 α). После дефазификации получится 71,333. После привлечения дополнительных средств продолжительность сократилась до (49, 54,5, 65,9), в виде W-

числа ($49+5,5 \alpha$, $65,9-11,4 \alpha$). В дефаззифицированом виде 56,4667.

- общая сумма дополнительных средств составила (5,13, 10, 10), в виде W-числа ($5,13+4,88 \alpha$, $10-0 \alpha$), После дефаззификации получится 8,3766.

На рисунке показано как изменились продолжительности выполнения каждой операции при минимальном, наиболее вероятном и максимальном временах.



Рисунок. Результаты оптимизации задачи (15)-(16)

Заключение

Таким образом, была произведена формализация и решение задач оптимизации сетевого планирования в условиях неопределенности с использованием нечетких треугольных чисел и W-чисел. Проведена оптимизация по времени, стоимости и ресурсам, которые нужны для выполнения проекта. Использование нечетких чисел позволяет экспертам формализовать представления о продолжительности работ в условиях неопределенности и отсутствии статистических данных.

Список литературы

1. Павлов А. Н., Соколов Б. В. Принятие решений в условиях нечеткой информации: учеб. пособие / А. Н. Павлов, Б. В. Соколов; ГУАП – СПб., 2006 – 72 с.
2. Таха, Хемди А. Введение в исследование операций, 7-е издание.: Пер. с англ. – М.: Издательский дом «Вильямс», 2005. – 912 с.
3. Пегат А. Нечеткое моделирование и управление / А. Пегат; пер. с англ. – 4-е изд., электрон. – М.: Лаборатория знаний, 2020. – 801 с.

Разработка сервиса принятия решений при прогнозировании временных рядов цен акций

А. В. Долгова

Студент бакалавриата

Н. А. Алейникова

Доцент

Введение

Прогнозирование роста и падения акций является актуальной проблемой [1]. Успешное предсказание будущей цены акций позволяет получить максимальную прибыль и снизить риски. В настоящее время существует множество площадок, благодаря которым человек может выбрать куда инвестировать свой капитал. Они предоставляют различные методы прогнозирования роста и падения цен акций, на основе чего инвестор может получить рекомендации по рентабельности определённых инвестиций. Однако уже существующие решения часто имеют перегруженный интерфейс, также прогнозы на многих сайтах недостаточно информативные, нет возможности выбора окна прогнозирования, как правило, отсутствует интервальный прогноз, нельзя оценить качество прогноза. Поэтому в данной работе ставится задача разработать сервис, устраняющий все вышеперечисленные недостатки. В качестве математического обеспечения сервиса будут использованы эконометрические модели прогнозирования временных рядов класса ARIMA, потому что они описывают большой спектр характеристик ВР и являются достаточно гибкими. Таким образом, целью данной работы является разработка веб-сервиса, реализующего ARIMA, который в дальнейшем можно будет использовать для поддержки принятия решений по наиболее рентабельной игре на бирже.

1. Математическое обеспечение веб-сервиса для прогнозирования временных рядов цен акций

Под временным рядом понимается последовательность наблюдений значений некоторой переменной, произведенных через равные промежутки времени [2]. Обозначим через y_t — значение переменной y в момент времени t , где $t = 1, \dots, T$.

Временной ряд называется стационарным если его основные вероятностные характеристики постоянны во времени и нестационарным, если хотя бы одна из вероятностных характеристик непостоянна. Именно благодаря тому, что вероятностные характеристики временного ряда постоянны можно строить прогнозы стационарных временных рядов на основе их предыстории [3].

ARMA-процесс определяется как стационарный процесс, характеризующийся порядками P и Q его AR и MA составляющих соответственно, удовлетворяющий уравнению:

$$y_t = \delta + \theta_1 y_{t-1} + \dots + \theta_p y_{t-p} + \varepsilon_t + \alpha_1 \varepsilon_{t-1} + \dots + \alpha_q \varepsilon_{t-q} \quad (1)$$

Модель $ARMA(p, q)$ при $p = 0$ будет представлять собой процесс скользящего среднего порядка P , а при $q = 0$ – процесс авторегрессии порядка q .

Для проверки стационарности временного ряда существуют различные методы [4]:

- Графический, который заключается в построении графика временного ряда и дальнейшего его анализа.
- Формальные, которые подразумевают использование статистических тестов, например, тестов Дики-Фуллера или KPSS-теста.

В случае если выборка данных временного ряда небольшая, на основании теста Дики-Фуллера можно сделать ошибочный вывод о нестационарности этого ряда, поэтому для более точного определения стационарности необходимо использовать KPSS-тест и после принимать решение.

Классические методы анализа и прогнозирования временных рядов направлены на то, чтобы сделать данные нестационарных временных рядов стационарными, например, последовательным взятием разностей, порядок разностей исходного временного ряда отображает параметр d , присутствующий в модели авторегрессии-скользящего среднего (ARIMA).

ARIMA использует три основных параметра (p, d, q) , которые выражаются целыми числами. Потому модель также записывается как $ARIMA(p, d, q)$.

Характерная запись модели $ARIMA(p, d, q)$ имеет следующий вид:

$$\Delta^d \cdot y_t = \sum_{i=1}^p \alpha_i \cdot \Delta^d \cdot y_{t-i} + \sum_{j=1}^q \beta_j \cdot \varepsilon_t + \varepsilon_t \quad (2)$$

где Δ^d – это оператор разности d -го порядка, например, $\Delta y_t = y_t - y_{t-1}$ – разность первого порядка.

2. Алгоритм построения модели ARIMA (p,d,q)

Впервые систематический подход к построению модели ARIMA был изложен Боксом и Дженкинсом в 1976 году. Методология Бокса-Дженкинса подбора ARIMA-модели для исследуемого временного ряда включает в себя следующие этапы [3]:

1. Проверка стационарности временного ряда с помощью теста Дики-Фуллера и KPSS-теста. Подбор параметра d модели ARIMA.
2. Идентификация модели. Подбор параметров p и q модели ARIMA с помощью анализа поведения автокорреляционной функции (ACF) и частной автокорреляционной функции (PACF) [4].
3. Оценивание параметров α и β с помощью метода максимального правдоподобия.
4. Проверка адекватности модели на основе следующих критериев:
 - оценки коэффициентов модели должны быть статистически значимы;
 - анализ остатков модели;
 - информационные критерии, которые характеризуют качество уравнения, например, информационный критерий Шварца (BIC)

$$BIC = \ln T \frac{p+q}{T} + \ln\left(\frac{\sum e_t^2}{T}\right) \quad (3)$$

где T - объем выборки, p и q — параметры ARIMA модели, e_t - остатки временного ряда.

Чем лучше модель, тем меньше критерий Шварца, так как это значит, что оба слагаемых не очень большие, то есть модель достаточно простая и достаточно хорошо описывает данные.

5. Прогнозирование на основе полученной модели.

Точечный прогноз заключается в получении прогнозного значения y_p , которое определяется путем подстановки в уравнение регрессии соответствующего (прогнозного) значения.

Интервальный прогноз заключается в построении доверительного интервала прогноза, т. е. нижней и верхней границ $y_{p \min}$, $y_{p \max}$

интервала, содержащего точную величину для прогнозного значения y_p с заданной вероятностью.

При построении доверительного интервала прогноза используется стандартная ошибка прогноза:

$$m_{y_p} = \sigma_{ocm} \cdot \sqrt{1 + \frac{1}{n} + \frac{(x_p - \bar{x})^2}{\sum (x - \bar{x})^2}} \quad (4)$$

где

$$\sigma_{ocm} = \sqrt{\frac{\sum (y - \hat{y})^2}{n - m - 1}} \quad (5)$$

Строится доверительный интервал прогноза:

$$\hat{y}_{y_p} = \hat{y}_p \pm t_{табл} \cdot m_{y_p} \quad (6)$$

3. Разработка алгоритма прогнозирования цен акций на основе методики Бокса-Дженкина

Основная задача веб-сервиса — определить оптимальную модель ARIMA для построения прогноза, для этого используется следующий алгоритм, представленный на рис.1.

Сначала проверяется стационарность исходного временного ряда цен акций с помощью тестов Дики-Фуллера и KPSS. Если ряд нестационарный, то находят разности, снова проверяется на стационарность и этот процесс происходит до тех пор, пока оба теста не покажут, что модифицированный временной ряд является стационарным. Порядок разностного оператора берется в качестве значения параметра d модели ARIMA. Далее рассчитываются АКФ и ЧАКФ по формуле:

$$k = \min(10 \times \ln n, n - 1) \quad (7)$$

где k – количество лагов, n – размер датасета.

Таким образом, как правило, будет использоваться первый аргумент в функции минимума, если будет 10 или меньше наборов данных, будут учитываться все допустимые лаги. Путем анализа АКФ и ЧАКФ определяются возможные значения порядка авторегрессии P и порядка скользящего среднего q . Алгоритм для каждого из них одинаков и заключается в том, что всегда будут рассматриваться значения 0 и 1 как самые вероятные и дополнительно находиться третий. Значение одного из параметров P или q фиксируется и

приравнивается к 0 или 1, тогда значение второго параметра перебирается до тех, пока не начнется прямое или осциллирующее убывание АКФ или ЧАКФ. Если оно находится, то запоминается данная комбинация значений p и q . Таким образом формируются всевозможные комбинации (p, d, q) . В следующей фазе алгоритма производится построение модели ARIMA для каждого полученного (p, d, q) и отбирается та из них, у которой минимальное значение критерия Шварца. Полученная модель используется в дальнейшем для осуществления прогноза.

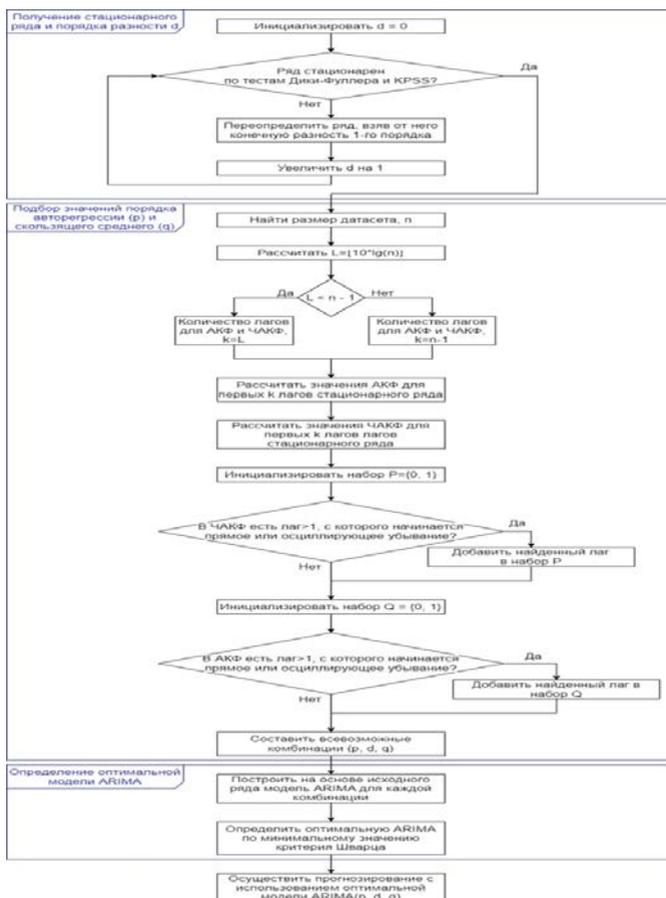


Рис. 1. Алгоритм прогнозирования

4. Реализация веб-сервиса

В соответствии с поставленными задачами был реализован веб-сервис принятия решений при прогнозировании временных рядов цен акций, с помощью которого пользователи могут для интересующих их ценных бумаг видеть как непосредственно прогноз, так и информацию о модели прогнозирования и детали прогноза: его точность, коррелограммы, значения критериев и т. д.

Приложение представляет собой клиент-сервер, архитектура которого представлена на рис. 2.

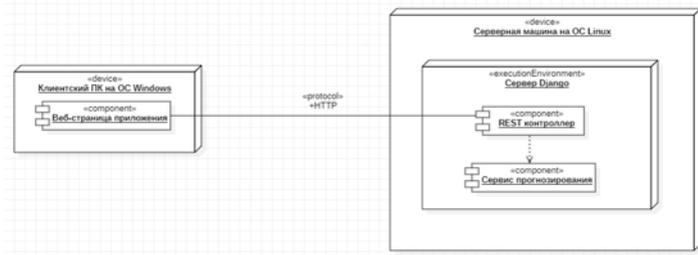


Рис. 2. Архитектура веб приложения

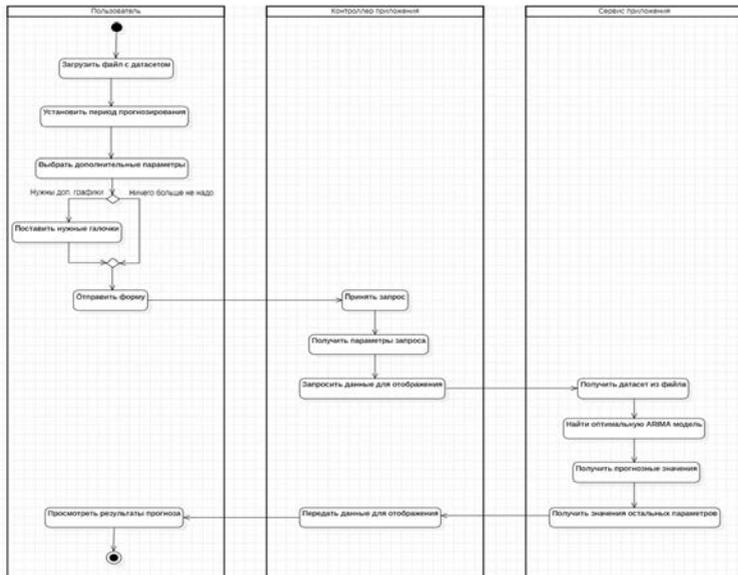


Рис. 3. Логика взаимодействия с приложением

При работе с приложением пользователю следует заполнить форму, представленную на рис. 4, загрузить файл с датасетом (в формате CSV или Excel), задать нужный период прогнозирования. Также есть на форме есть опции для возможности отображения АКФ и ЧАКФ. После ее заполнения, клиент ее обрабатывает, контроллер сервера принимает запрос, получает параметры формы из запроса, обращается к сервису, передавая параметры с целью получить данные для отображения клиенту, сервис сначала считывает данные из файла, используя их находит оптимальную ARIMA модель, рассчитывает прогнозные значения и ряд других значений и параметров, и, наконец, возвращает всё для последующего отображения на клиентском устройстве. Сервис открыт для всех пользователей, регистрация в приложении не требуется.

Датасет (CSV или Excel файл): Файл не выбран

Сделать прогноз на: дней

Строить график автокорреляционной функции:

Строить график частной автокорреляционной функции:

Рис. 4. Форма приложения

Результат работы сервиса изображен на рис. 5.



Рис. 5. График прогноза цены акций

На первом графике отображаются исходный и прогнозный временные ряды и 95%-ный доверительный интервал актуального прогноза. Далее указывается прогнозное значение на последний день прогноза и его 95%-ный доверительный интервал. Ниже выводится совет пользователю касательно действий по отношению к акциям и таблица с информацией о модели прогнозирования (рис. 6). В таблице указывается используемая модель для построения прогноза, значение точности прогноза, среднеквадратическое отклонение и значения критериев Акаике и Шварца. Имеется возможность вывода прогнозных значений цен акций в виде таблицы.

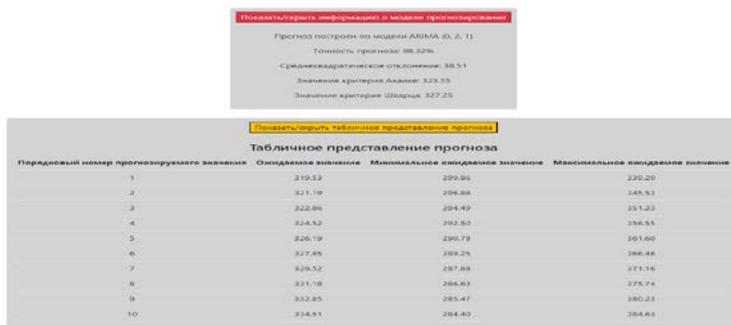


Рис. 6. Информация о модели прогнозирования и табличное представление прогноза

Заключение

Данная статья посвящена разработке алгоритма прогнозирования цен акций на основе модели ARIMA и методики Бокса-Дженкинса и реализации соответствующего веб-сервиса для поддержки принятия решений инвесторов при принятии решений по наиболее рентабельной игре на бирже. Особенности сервиса являются простота в использовании и отсутствие ограничений, пользователю не нужно авторизовываться и можно самому выбирать период прогноза, бесплатное использование, что позволяет пользователям с достаточным комфортом получать интересные прогнозы.

Список литературы

1. Почему инвестиционные приложения – это тренд будущего 2022 года и как подготовиться уже сейчас? [Электронный ресурс] Режим доступа: https://partnerkin.com/tribuna/blog_workle_700/pochemu_investicii_onnye_prilozh

2. Носко, В.П. Введение в регрессионный анализ временных рядов: учебное пособие / В.П. Носко – 2-е изд. – М.: Издательский Дом МИСиС, 2002. – 274 с.

3. Керимов, А.К. Анализ и прогноз финансовых временных рядов: учебное пособие / А.К. Керимов. – 2-е изд. – М.: Прометей, 2018. – 178 с.

4. Вербик М. Путеводитель по современной эконометрике. Пер. с англ. В.А. Банникова. Научн. ред. и предисл. С.А. Айвазяна. – М.: Научная книга, 2008 – 616 с.

Разработка приложения для организации Mesh-сети посредством Nearby Connections API

А. А. Донцов

Студент бакалавриата

Н. К. Самойлов

Старший преподаватель

Введение

В наши дни использование мобильных устройств стало неотъемлемой частью нашей повседневной жизни. Мы используем их для общения, работы, развлечений, и многого другого. Однако, часто бывает так, что мобильная связь оказывается недоступной - например, в местах с плохой сетевой инфраструктурой. Не более редко встречаются ситуации, когда встаёт необходимость общения через защищённый канал связи, без выхода трафика за пределы определённой группы. В таких ситуациях будет полезно приложение, которое позволяет организовать mesh-сеть для общения и обмена данными. Очевидный плюс такого подхода, по сравнению с другими средствами связи, используемыми в аналогичных условиях – это минимальная входная стоимость (не нужно покупать какое-либо дополнительное оборудование).

Цель работы – изучить способы организации mesh-сетей при помощи мобильных приложений, спроектировать и разработать приложение, которое позволит организовать mesh-сеть. Данное приложение должно обеспечивать соединение устройств пользователей для обмена сообщениями в безопасном и быстром режиме, при отсутствии доступа к интернету.

1. Обзор предметной области

Mesh-сети – это беспроводные сети с ячеистой топологией, которые позволяют передавать данные между устройствами через непосредственное соединение без необходимости использования центральной точки доступа и могут обеспечивать высокую скорость передачи данных на большие расстояния. В таких сетях каждый узел может принимать на себя роль коммутатора для остальных участников. [1] Пример сети с ячеистой топологией приведён на рис. 1.

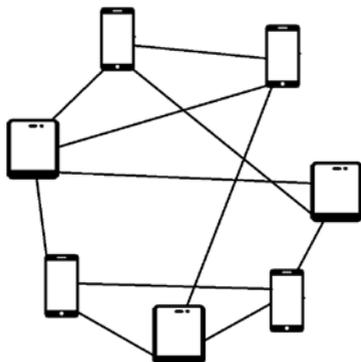


Рис. 1. Пример сети с ячеистой топологией

Основные особенности mesh-сетей:

- «Интеллектуальность» сети: каждый узел сети автоматически получает информацию об остальных узлах и определяет свою роль.
- Самоадаптация и самовосстановление сети: при выходе из сети одного из узлов, сеть способна автоматически переопределить маршруты.
- Быстрое и недорогое развёртывание: развёртывание ячеистой сети не требует дорогостоящей инфраструктуры и прокладки кабелей, а также экономна в эксплуатации. [2-3]

Структура mesh-сетей представляет собой множество узлов, которые взаимодействуют между собой и могут связываться не только друг с другом, но и с сетью интернет в целом. При этом каждый узел является равным по значимости, и данные могут передаваться через разные маршруты, что увеличивает надёжность сети и уменьшает вероятность отказа. [4]

Для построения mesh-сети на мобильных устройствах в первую очередь необходимо рассмотреть возможные способы их связи без использования сети GSM и выбрать наиболее оптимальный вариант. В качестве возможных технологий для рассмотрения были выбраны: WiFi Direct, Bluetooth, Wifi Aware и Nearby Connections API.

В результате анализа доступных технологий был сделан выбор в пользу Nearby Connections API, так как в ней сочетаются все преимущества WiFi Direct, Bluetooth и Bluetooth LE, а также присутствует поддержка mesh-топологии соединения устройств.

2. Особенности работы с Nearby Connections API

Nearby Connections API доступен только на Android (на iOS не поддерживается). [5-6]

API не ограничивает размер пакетов, передаваемых по Bluetooth и одноранговому Wi-Fi, а также не требует установленного интернет-соединения, что обеспечивает его работу в автономном режиме.

Одной из особенностей данной технологии является обязательное наличие на устройстве установленного Google Play Services, иначе API работать не будет.

В Nearby Connections API существует 3 варианта стратегии соединения устройств, представленные на рис. 2.

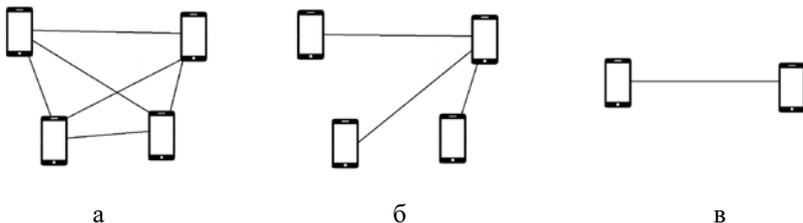


Рис. 2. Стратегии соединения в Nearby Connections API: а – P2P_CLUSTER, б – P2P_STAR, в – P2P_POINT_TO_POINT

Для обеспечения успешного соединения двух устройств необходимо, чтобы на каждом из них была задана одна и та же стратегия соединения. [6]

3. Реализация приложения

Для приложения были определены следующие требования:

- динамическое масштабирование mesh-сети;
- возможность подключения до 10 узлов к одному устройству;
- возможность передачи текстовых сообщений;
- возможность передачи голосовых сообщений;
- удобный пользовательский интерфейс.

В качестве средств реализации были выбраны:

- Android SDK (минимум Android 4.1);
- Nearby Connections API;
- Material design library.

На рис. 3 представлена Use Case диаграмма приложения.

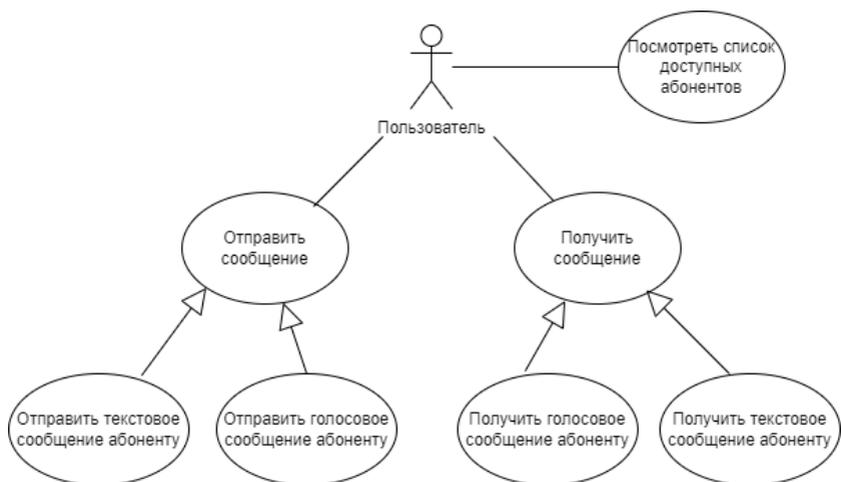


Рис. 3. Use Case диаграмма

На рис. 4 представлена диаграмма классов приложения.

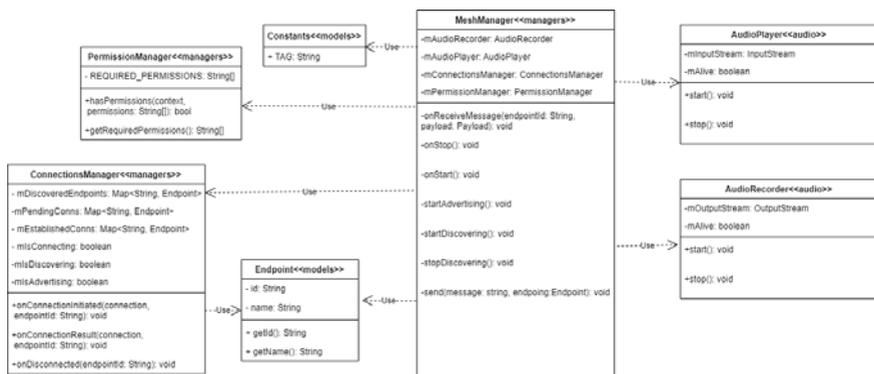


Рис. 4. Диаграмма классов

Основной алгоритм работы программы:

1. После запуска приложения, автоматически начинается процесс поиска устройств и соединения с ними;
2. Если найдено устройство с таким же запущенным приложением, то ему отправляется запрос на соединение;
3. Устройство, которому отправлен запрос, обрабатывает его и принимает либо отклоняет соединение;

4. В случае, когда соединение установлено, в списке доступных абонентов у обоих устройств появляется новое соединение, отправляется информация о новом соединении другим устройствам;
5. После этого можно отправлять новому абоненту текстовые и голосовые сообщения;
6. Когда соединение разрывается – в списке абонентов пропадает соответствующее устройство.

На рис. 5 показана диаграмма последовательностей приложения.

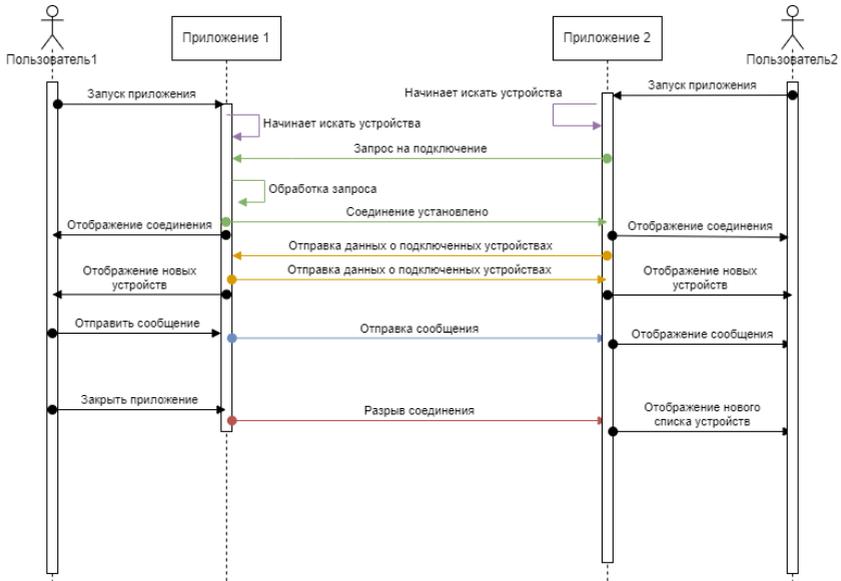


Рис. 5. Диаграмма последовательностей

4. Пользовательский интерфейс

На рис. 6 показано окно с доступными абонентами (на двух соединённых устройствах).

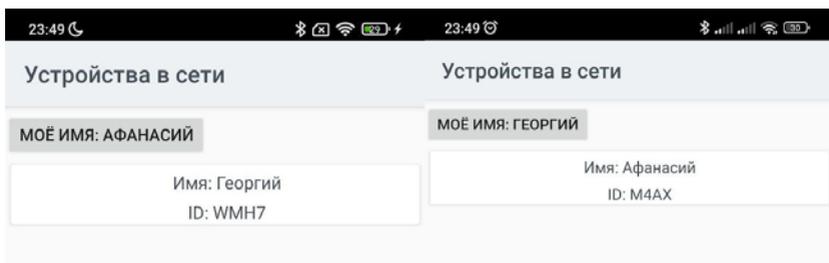


Рис. 6. Окно с доступными абонентами

Если нажать на кнопку «Моё имя ...», то откроется окно редактирования имени пользователя (рис. 7).

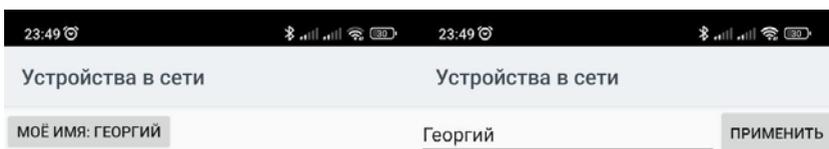


Рис. 7. Окно редактирования имени пользователя

При нажатии на карточку доступного абонента открывается окно диалога с данным абонентом (представлено на рис. 8). Здесь пользователь может отправлять сообщения абоненту (фиолетовым цветом выделены собственные сообщения, а серым сообщения собеседника). Также если зажать клавишу регулировки громкости в данном окне, то начнётся передача голосового сообщения.

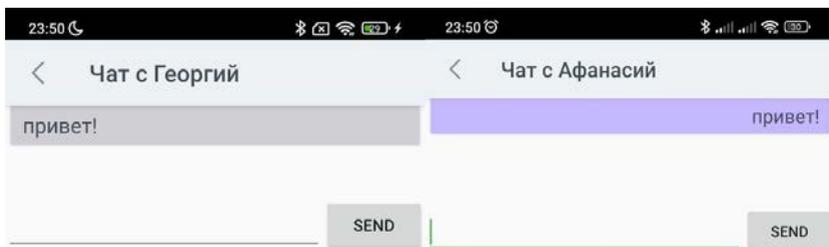


Рис. 8. Окно диалога с абонентом

Заключение

В ходе работы были рассмотрены теоретические аспекты сетей с ячеистой топологией и изучены основные особенности работы с Nearby Connections API.

В результате работы было разработано приложение, позволяющее быстро организовать mesh-сеть и передавать сообщения внутри этой сети без доступа к интернету.

Список использованной литературы

1. Голубничая, Е.Ю. Упрощённый алгоритм маршрутизации в Wi-Fi Mesh-сетях мониторинга / Е. Ю. Голубничая, Б. Я. Лихтциндер // Инфокоммуникационные технологии. – 2014. – Т. 12. – №2. – С. 53-57.

2. IEEE Standard for Information Technology [Электронный ресурс] : сайт. – Режим доступа : <https://cengproject.cankaya.edu.tr/wp-content/uploads/sites/10/2017/12/SDD-ieee-1016-2009.pdf>

3. Uludag, S. A taxonomy and evaluation for developing 802.11-based wirelessmesh network testbeds / S. Uludag, T. Imboden, K. Akkaya // International Journal of Communication Systems. – 2012. – Vol. 25. – Iss. 8. – P. 965-990.

4. Колыбельников, А. И. Обзор технологий беспроводных сетей / А. И. Колыбельников // Труды Московского физико-технического института. – 2012. – Т. 4. – №2-14. – С. 4-28.

5. Android documentation [Электронный ресурс] : сайт. – Режим доступа : <https://developer.android.com/docs>

6. Nearby Connections documentation [Электронный ресурс] : сайт. – Режим доступа : <https://developers.google.com/nearby/connections>

Методы и алгоритмы машинного обучения для решения задачи распознавания языка жестов

А. А. Евсеева

Студент бакалавриата

Е. Ю. Митрофанова

Доцент

Введение

В настоящее время во всем мире насчитываются миллионы людей с нарушениями речи и слуха. Эта значительная группа людей оказывается ограниченной в привычной для большинства речевой коммуникации. В качестве основного средства общения глухих и слабослышащих людей служит жестовой язык. Такая знаковая система в основном использует ручную коммуникацию для передачи сообщений, включающую в себя одновременное сочетание формы кисти, ориентации и локализации ладони, направление движения, немануальные компоненты [1].

Ежедневно носители жестового языка сталкиваются с коммуникационными барьерами, обусловленными тем, что большинство слышащего сообщества не владеет жестовым языком. Прогресс в области машинного обучения, а также нарастание вычислительных мощностей позволяют расширить возможности для разработки систем автоматического распознавания жестового языка, способствующих адаптации слабослышащих людей в обществе.

Современные исследования в области распознавания языка жестов делятся на две категории: основанные на технологии компьютерного зрения (визуальный подход), основанные на использовании датчиков [2]. Подход, использующий методы машинного зрения, является наиболее популярным решением, поскольку взаимодействие человека с системой делается простым и естественным, не требующим использования дополнительных устройств (к тому же зачастую дорогостоящих).

Данная статья посвящена разработке системы автоматического распознавание языка жестов, используя методы машинного обучения и компьютерного зрения. Основными задачами исследования являются: анализ известных подходов решения задачи распознавания жестов; исследование общего алгоритма распознавания; выделение методов,

наиболее подходящих для достижения цели исследования; реализация алгоритма распознавания языка жестов.

1. Общий вид алгоритма распознавания языка жестов

На сегодняшний день в области автоматического распознавания жестов и компьютерного зрения используется многообразие различных методов. В общем случае алгоритм распознавания можно представить в виде этапов [3], представленных на рис. 1.



Рис. 1. Общий алгоритм распознавания языка жестов

В рамках визуального подхода в качестве входных данных используются изображения или видеозаписи, полученные одной или несколькими камерами. Исходное изображение зачастую подвергается различным методам предварительной обработки с целью устранения нежелательных шумов и повышения качества изображения. Затем следует этап сегментации, целью которой является выделение областей интереса (фрагменты исходного изображения, содержащие жестовые артикуляторы). После этого становится возможным извлечение информативных признаков, применяемых далее для формирования и обучения модели классификатора.

2. Подход, основанный на геометрических характеристиках руки

При реализации алгоритма распознавания языка жестов в рамках рассмотренного подхода был избран метод, оперирующий внешними признаками скелета руки [4]. На рис. 2 изображены ключевые точки руки, задающие ее геометрические характеристики (расположение суставов, точки основания ладони и пальцев).

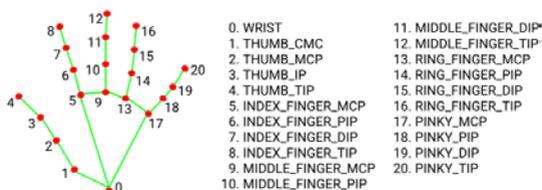


Рис. 2. Ключевые точки руки

Таким образом, алгоритм распознавания жестов будет состоять из этапов, отраженных на рис. 3.



Рис. 3. Алгоритм распознавания жестов по ключевым точкам руки

Популярным и высокоточным решением задачи трекинга человеческого тела (в т. ч. рук и пальцев) являются модели машинного обучения, предоставляемые фреймворком Google Mediapipe, обеспечивающее производительность в реальном времени. В его основе лежит конвейер, состоящий из нескольких моделей машинного обучения, работающих вместе: модель обнаружения ладони (Palm Detection Model) и модель ключевых точек рук (Hand Landmark Model) [5].

Модель обнаружения ладони представляет собой сверточную нейронную сеть, построенную в виде пирамиды (Feature Pyramid Net, FPN). Модель ключевых точек руки является регрессией, выполняющей предсказание локализации 21 ключевой точки руки на сегментированном изображении. Обучение моделей осуществлялось на большом размеченном вручную наборе изображений Google.

3. Архитектура системы

Разработка алгоритма распознавания жестов была реализована за счет конвейера моделей машинного обучения, состоящего из детектора

ладони, модели ключевых точек рук и классификатора жестов. Архитектура системы изображена на рис. 4. Первые две модели являются рассмотренными решениями модуля Mediapipe Hands. Формирование и обучение модели классификатора жестов будет описано в следующих разделах.

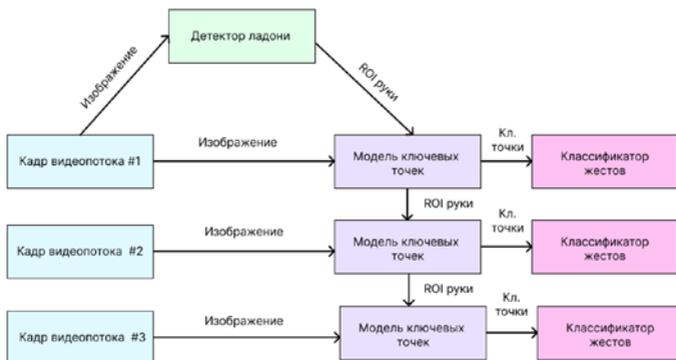


Рис. 4. Конвейер моделей машинного обучения

Первым шагом модель обнаружения ладони выполняет точную сегментацию ладони на входном изображении. Затем модель ключевых точек рук предсказывает координаты 21 ключевой точки руки по сегментированной области ладони. Последний шаг состоит в распознавании жеста моделью классификатора, которая принимает на вход образ жеста, представленного в виде вектора признаков (координат ключевых точек) и возвращает метку класса для данного образа.

4. Формирование выборки

При реализации классификатора жестов в качестве жестового корпуса были взяты дактилемы русского жестового языка. Каждая из используемых дактилем представляет собой статический жест руки, отображающий букву русского алфавита. На рис. 5 представлена русская дактильная азбука.

Формирование выборки для каждого класса жеста выполнялось путем их демонстрации в реальном времени и выполнения в цикле:

- захвата кадра веб-камеры;
- извлечения признаков путем обработки кадра (разрешением 800×600) моделями Mediapipe Hands;
- запись ключевых точек в CSV-файл.

Таким образом, для 28 классов (букв) было собрано по 500 образов. Общий объем обучающей выборки составил 14000 образов.



Рис. 5. Русский тактильный алфавит

5. Предобработка и нормализация

Использование описания ключевых точек рук как их пиксельных координат на исходном изображении приводит к тому, что при демонстрации одного и того же жеста смещение руки по оси x или y (а также при приближении или отдалении от камеры) означает сильное различие векторов-образов в признаковом пространстве. Следствием этого являются сложности в обучении модели, необходимость сбора большого числа обучающих данных для разных вариаций положения жеста на изображении. Чтобы избежать этого было решено выполнить следующие преобразования:

1. смещение центра системы координат в нулевую ключевую точку (точку основания ладони);
2. нормализация координат с целью «выравнивания» их значений, т.е. приведения к интервалу $[0,1]$.

6. Модель классификатора

В качестве модели классификатора выбрана полносвязная нейронная сеть прямого распространения, реализованная средствами фреймворка Keras. Архитектура сети имеет следующий вид:

Обучение модели методом обратного распространения ошибки выполнялось с использованием:

- алгоритма оптимизации Адама;
- кросс-энтропии в качестве функции потерь;
- метрики точности (accuracy).

Обучение модели проходило в 412 эпох. Значение точности на последней эпохе для валидационной выборки достигло 0.97, а функции потерь - 0.11.

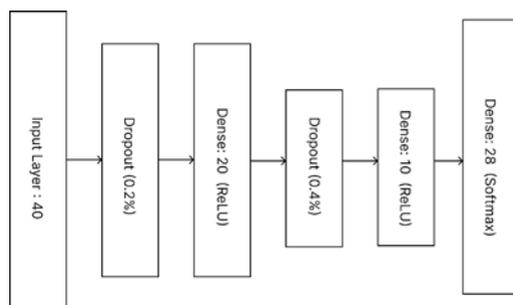


Рис. 6. Архитектура классификатора жестов

7. Результаты распознавания

Оценка качества классификатора на тестовом наборе данных производилась с помощью следующих показателей:

- точность (precision): демонстрирует способность алгоритма обнаруживать данный класс;
- полнота (recall): характеризует способность отличать этот класс от других классов;
- f1-score: средне-гармоническое значение метрик precision и recall.

Из отчета, представленного на рис. 7 можно сделать вывод о том, что модель имеет приемлемую точность при классификации новых данных.

	precision	recall	f1-score
A	0.95	0.91	0.93
Б	0.91	0.92	0.92
В	0.94	0.98	0.96
Г	0.94	0.90	0.92
Д	0.97	0.95	0.96
Е	0.90	0.94	0.92
Ж	0.95	0.96	0.95
И	0.97	0.95	0.96
К	0.89	0.90	0.89
Л	0.95	0.94	0.95
М	0.97	0.95	0.96
Н	0.98	0.93	0.95
О	0.98	0.93	0.95
П	0.99	0.99	0.99
Р	1.00	0.98	0.99
С	0.95	0.99	0.97
Т	0.97	0.98	0.98
У	0.96	0.98	0.97
Ф	0.99	0.98	0.98
Х	0.97	0.96	0.96
Ц	0.97	0.99	0.98
Ч	0.95	0.98	0.96
Ш	0.91	0.89	0.90
Ы	0.98	0.92	0.95
Ь	0.92	1.00	0.96
Э	0.97	1.00	0.99
Ю	0.90	0.93	0.91
Я	0.97	0.97	0.97

Рис. 7. Оценка качества распознавания метриками

Пример распознавания букв русского жестового языка в реальном времени, полученный с помощью разработанной программной системы, представлен на рис. 8.

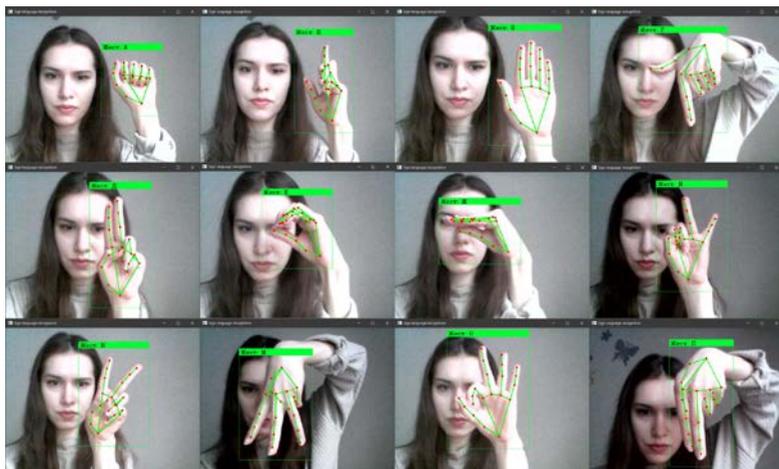


Рис. 8. Пример распознавания жестов реализованным алгоритмом

Заключение

Рассмотрена задача распознавания языка жестов методами машинного обучения. Выделен общий алгоритм распознавания жестов в рамках подхода, основанном на компьютерном зрении.

В соответствии с поставленной задачей разработан алгоритм распознавания статических жестов русского дактильного алфавита на основе геометрических характеристик рук. В качестве решения была предложена архитектура, состоящая из конвейера моделей машинного обучения, включающая модель обнаружения ладони, модель ключевых точек руки, классификатор жестов. В качестве реализации первых двух моделей выбрано решение фреймворка Mediapipe Hands. Для реализации модели классификатора была построена и обучена модель полносвязной нейронной сети средствами фреймворка Keras. Оценка качества распознавания реализованного алгоритма показала приемлемую точность для тестовой выборки.

Список литературы

1. Гейльман, И. Ф. Специфические средства общения глухих. Дактилология и мимика: учебное пособие / И. Ф. Гейльман. – ч.1– Л. : Ленинград, 1975. – 26 с.
2. Рюмин, Д. А. Подходы к автоматическому распознаванию жестовой информации: аппаратное обеспечение и методы / Д. А. Рюмин, И. А. Кагиров // Пилотируемые полеты в космос. – 2021.– № 3(40) .– С. 83-88.
3. Machine learning methods for sign language recognition: A critical review and analysis / I. A. Adeyanju [et. al.] // Intelligent Systems with Applications. – 2021. – № 12. – С. 154-161.
4. Bencheri, M. A. Sign Language Recognition System Using 2D Hands and Body Skeleton Data / M. A. Bencheri // IEEE Access, 2021. – vol. 9. – С. 59612-59627.
5. Mediapipe hands solution [Электронный ресурс].– Режим доступа : <https://chuoling.github.io/mediapipe/solutions/hands.html>

Сравнение оптимизаторов для обучения нейронных сетей

В. И. Запорожец

Студент бакалавриата

А. В. Акимов

Старший преподаватель

Введение

В настоящее время для обучения нейронных сетей используют различные методы оптимизаций, основанные на градиентных методах. В данной статье предлагается рассмотреть теорию одних из самых популярных методов обучения нейронных сетей, а также их практическое сравнение на задаче регрессии положения вершин 3D-модели человека по его фотографии.

1. Обучение нейронной сети. Общие сведения

Нейронные сети обучаются с использованием градиентных методов оптимизации, в которых для нахождения минимума функции ошибки используются значения градиентов для каждого параметра сети.

Градиентом функции многих переменных называется вектор, значения которого равны частной производной функции в точке: пусть есть функция от нескольких переменных $f(x_1, x_2, \dots, x_n)$, тогда градиентом данной функции является вектор:

$$\text{grad } f = \left(\frac{df}{dx_1}, \frac{df}{dx_2}, \dots, \frac{df}{dx_n} \right).$$
 Суть градиента заключается в том, что он

указывает на направление наискорейшего роста функции. Отрицательное же значение указывает на направление наискорейшего спуска, что и используется для минимизации значения функции ошибки нейронной сети.

Существует несколько вариантов получения значения производной: ручной, численный, автоматический.

При ручном программист задаёт не только саму функцию, но ещё и функцию, возвращающую её градиент. Этот способ обычно используется для несложных функций, так как вывод формулы градиента функции обычно трудоёмкий процесс.

Численное дифференцирование – метод приближённого вычисления значения градиента функции согласно формуле:

$$\text{grad } f(w_i) = \frac{f(w_i + h) - f(w_i - h)}{2h}, \text{ где } i \text{ – размерность вектора}$$

параметров w , h – подбираемая константа. Данный подход позволяет задать только прямую функцию, но при этом он вычисляет лишь примерное значение производной, что ухудшает сходимость.

При автоматическом дифференцировании в память компьютера сохраняется граф вычислений, в который записывается порядок применения и результат каждого оператора. Пример графа вычислений для функции $g(x, y, z) = z * (x + y)$ изображён на рис. 1. Во время дифференцирования градиент распространяется от конца графа к его началу, через цепное правило, которое можно сформулировать так:

пусть $y = y(x(t))$, тогда $\frac{dy}{dt} = \frac{dy}{dx} \frac{dx}{dt}$. Главными преимуществами

данного метода является простота использования программистом, а также точность ручного дифференцирования.

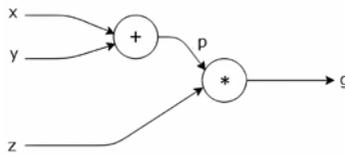


Рис. 1. Пример графа вычислений

Суть алгоритма градиентного спуска заключается в том, чтобы на каждой итерации от значений параметров вычитать значение градиента, умноженное на некоторую константу, называемую скоростью обучения.

Пусть w – вектор параметров нейронной сети, $l(w)$ – функция ошибки, $\nabla l(w)$ – значение градиента функции в точке w , μ – скорость обучения. Тогда, итерационно, $w^{new} = w^{old} - \mu * \nabla l(w^{old})$ [1]. Данный алгоритм также называется SGD.

При обучении нейронной сети чаще всего функция ошибки представлена как функция от двух векторов: вектора-результата нейронной сети и целевого вектора: $l(prediction, target)$, где

$prediction = net(w, inputs)$, net – функция нейронной сети, w – её параметры, $inputs$ – входной тензор, $target$ – ожидаемый результат.

2. Модификация градиентного спуска

У градиентного спуска есть несколько известных проблем. Многие из них связаны с тем, что значение градиента вычисляется не на всём обучающем наборе данных, а на небольших его частях, называемых мини-пакетами. Вследствие чего, на каждой итерации градиентный спуск может двигаться не в сторону локального минимума из-за шума значений градиентов. Чтобы от него избавиться, на каждой итерации значения весов изменяются не на значение градиентов, а на скользящее среднее предыдущих значений градиента, что позволяет уменьшить влияние шума и получить более точное направление к глобальному минимуму [2]. Формула, по которой высчитывается значение скользящего среднего: $M_t = \beta * M_{t-1} + (1 - \beta) * N_t$, где M_t – значение скользящего среднего, β – гиперпараметр, N_t – зашумлённые данные. Пример работы алгоритма «скользящее среднее» изображён на рис. 2.

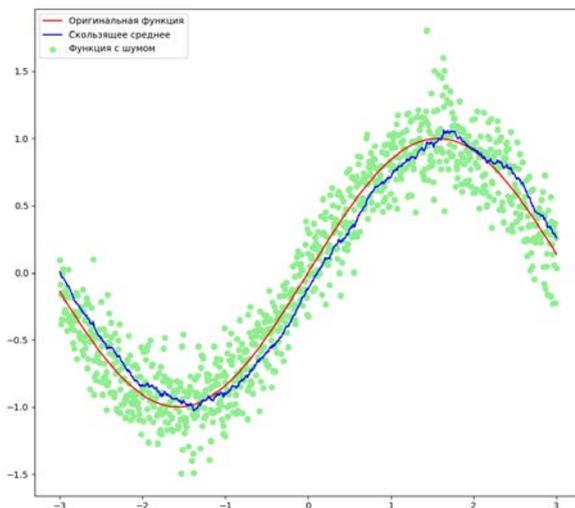


Рис. 2. Иллюстрация работы алгоритма «скользящее среднее»

За гладкость результирующей последовательности отвечает гиперпараметр β – чем он меньше, тем ближе значение алгоритма к значениям исходной последовательности, чем он больше, тем результат

алгоритма более гладкий, но при этом появляется задержка между входным значением и реакцией алгоритма. Влияние этого параметра на последовательность изображено на рис. 3. В большинстве случаев значение гиперпараметра β принимается равным 0,9–0,95.

При этом чем больше значение этого параметра, тем легче алгоритму будет избегать неглубоких локальных минимумов и седловых точек, где значения градиента функции близки к нулю, но при этом могут возникать осцилляции вокруг оптимальных точек, либо же алгоритм и вовсе будет расходиться.

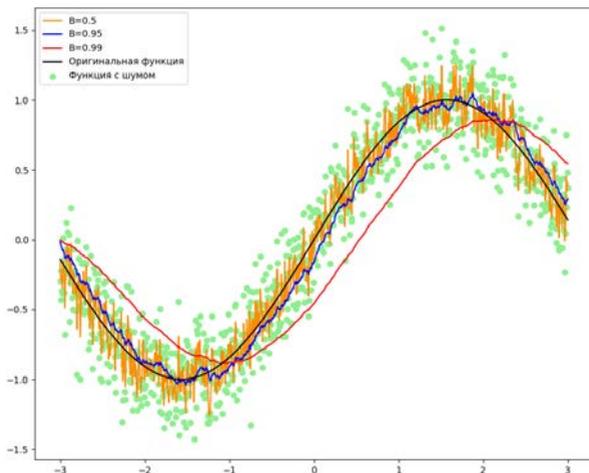


Рис. 3. Иллюстрация поведения скользящего среднего с разными значениями гиперпараметра β

В итоге, формула для стохастического градиентного спуска с импульсом принимает следующий вид [2]:

$$V_t = \beta * V_{t-1} + (1 - \beta) * \nabla l(\text{prediction}, \text{target}),$$

$$W_t = W_{t-1} - \mu * V_t,$$

где V_t – скользящее среднее градиентов, W_t – веса нейронной сети на итерации t . Данный алгоритм называется SGD с импульсом.

Также, существуют случаи, когда признаки в исходных данных, либо же признаки, полученные внутренними слоями нейронной сети, являются редкими, но при этом достаточно информативными и важными для правильного обучения нейронной сети. Поэтому важно обновлять параметры нейронной сети с учётом того, насколько

типичный признак они фиксируют. По-другому можно сказать, что для каждого параметра необходимо хранить и со временем обновлять свою скорость обучения. Достигается это путём хранения суммы квадратов градиентов для каждого параметра нейронной сети. При этом чем чаще и сильнее обновляется параметр, тем больше соответствующая ему накопленная сумма. В результате, формулы для стохастического градиентного спуска с накоплением квадратов градиентов (этот алгоритм называется AdaGrad) выглядят так:

$$G_t = G_{t-1} + \nabla l(\text{prediction}, \text{target})^2,$$

$$W_t = W_{t-1} - \frac{\nabla l(\text{prediction}, \text{target}) * \mu}{\sqrt{G_t} + \epsilon},$$

где G_t – накопленные квадраты градиентов. Параметр ϵ – небольшая положительная константа, позволяющая избежать деления на ноль.

Из формул выше можно сделать вывод, что чем реже обновляется параметр, тем меньше соответствующая ему накопленная сумма, а чем чаще, тем больше. А из-за того, что величина обновления веса обратно пропорциональна квадрату накопленного градиента, получается, что скорость обучения тем выше, чем реже обновляется параметр, и тем меньше, чем он обновляется чаще.

К особенностям данного алгоритма можно отнести постепенно затухающую скорость обучения, что позволяет к концу обучения делать маленькие, но при этом аккуратные шаги. С другой стороны, это же может являться и недостатком, так как скорость обучения может снизиться слишком быстро, что не позволит достичь оптимальной точки, в результате нейронная сеть останется недоученной [3].

Недостаток, описанный выше, решается путём хранения не суммы квадратов градиентов, а их скользящего среднего. Данный алгоритм называется RMSProp [4]. Его формула для изменения весов имеет следующий вид:

$$E_t = \gamma * E_{t-1} + (1 - \gamma) * \nabla l(\text{prediction}, \text{target})^2,$$

$$W_t = W_{t-1} - \frac{\nabla l(\text{prediction}, \text{target}) * \mu}{\sqrt{E_t} + \epsilon},$$

где γ – коэффициент использующийся для вычисления скользящего среднего градиентов, E_t – скользящее среднее квадратов градиентов.

Очевидно, что теперь значение в знаменателе может не только возрастать, но и убывать, что предотвращает проблему со слишком маленькой скоростью обучения при большом числе итераций алгоритма.

Дальнейшим улучшением алгоритма оптимизации является объединение идеи импульса с идеей сильного обновления для более редких, но информативных признаков. Этот алгоритм называется Adam – Adaptive Moment Estimation (метод адаптивной оценки моментов) [5]. Формулы, описывающие обновление параметров, выглядят следующим образом:

$$V_t = \beta_1 + (1 - \beta_1) * \nabla l(\text{prediction}, \text{target}),$$

$$\hat{V}_t = \frac{V_t}{1 - \beta_1^t}, \hat{M}_t = \frac{M_t}{1 - \beta_2^t},$$

$$W_t = W_{t-1} - \mu * \frac{\hat{V}_t}{\sqrt{\hat{M}_t + \epsilon}},$$

где β_1 – коэффициент, использующийся для вычисления скользящего среднего градиентов, β_2 – коэффициент, использующийся для вычисления скользящее среднее квадратов градиентов, t – номер текущей итерации, V_t – скользящее среднее градиентов, M_t – скользящее среднее квадратов градиентов, \hat{V}_t – нормализованное скользящее среднее градиентов, \hat{M}_t – нормализованное скользящее среднее квадратов градиентов, α – скорость обучения.

3. Сравнение различных оптимизаторов

Также было проведено исследование с целью определить, какой из оптимизаторов лучше всех справляется с задачей регрессии положения вершин 3D-модели человека по его фотографии.

Для каждого оптимизатора был использован один и тот же набор данных, одна и та же скорость обучения. При этом у градиентного спуска с импульсом использовалось значение параметра $\beta_1 = 0.9$, у RMSProp $\gamma = 0.99$, у Adam $\beta_1 = 0.9; \beta_2 = 0.999$.

При этом хуже всех себя показал оптимизатор SGD. SGD с импульсом показал себя лучше, чем простой SGD, но к концу обучения их ошибки оказались сравнимыми. Adagrad оказался значительно лучше, чем SGD и SGD с импульсом, но хуже, чем Adam и RMSProp. RMSProp и Adam показали сравнимые ошибки, но при этом Adam всё равно немного лучше. Графики зависимости значения ошибки на

валидационном наборе данных от итерации обучения показаны на рис. 4.

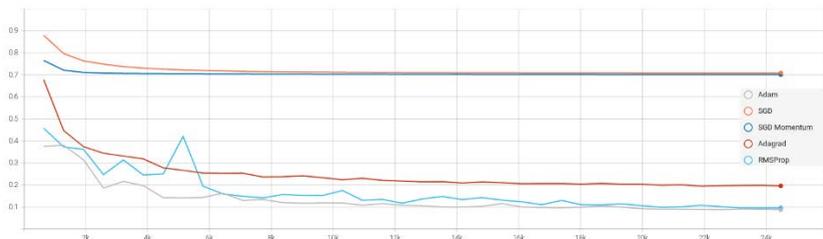


Рис. 4. Графики ошибки на валидационном наборе данных

4. Заключение

В данной работе была описана теория обучения нейронных сетей с использованием градиентных методов оптимизаций, а также были описаны алгоритмы SGD, SGD с импульсом, Adagrad, RMSProp, Adam. Было проведено их сравнение на задаче регрессии положения вершин 3D-модели человека по его фотографии, при этом лучшую валидационную ошибку показал оптимизатор Adam, что говорит о том, что он лучше справился с задачей. Поэтому в качестве начальной точки экспериментов рекомендуется использовать именно его.

Список литературы

1. Gradient Descent Algorithm — a deep dive. [Электронный ресурс]. – Режим доступа: <https://towardsdatascience.com/gradient-descent-algorithm-a-deep-dive-cf04e8115f21>.
2. Stochastic Gradient Descent with momentum [Электронный ресурс]. – Режим доступа: <https://towardsdatascience.com/stochastic-gradient-descent-with-momentum-a84097641a5d>.
3. Duchi J., Adaptive subgradient methods for online learning and stochastic optimization / J. Duchi, E. Hazan, Y. Singer // Journal of machine learning research. – 2011. – Т. 12. – №. 7. – P. 2122-2159.
4. RMSProp [Электронный ресурс]. – Режим доступа: <https://paperswithcode.com/method/rmsprop>.
5. Kingma, D.P. Adam: A method for stochastic optimization / D.P. Kingma, J. Ba // arXiv preprint arXiv:1412.6980. – 2014. – P. 1-15.

Применение обучения с подкреплением для поиска оптимальных стратегий в играх

В. В. Кимбрис

Студент бакалавриата

А. Ф. Клинских

Профессор

Введение

Обучение с подкреплением – раздел машинного обучения, в котором испытуемая сущность (агент) взаимодействует с некоторой системой для достижения определенных целей. Методы обучения с подкреплением активно используются для создания роботов, способных заменить физический труд людей, в чат-ботах, для построения систем оптимального управления. В настоящее время ведутся различные исследования в этой области [1–3].

1. Основные понятия обучения с подкреплением

1. Модель агент-окружение.

Важнейшими понятиями в обучении с подкреплением являются агент и окружение. Агент – это сущность, способная наблюдать текущее состояние системы и на основании этого состояния принимать решения (действия) для максимизации общей награды, полученной от окружения. Окружение – это некоторая система, чьи состояния меняются по определенному правилу в зависимости от действий агента и предыдущего состояния. Одна итерация взаимодействия агента и окружения называется эпизодом, в рамках которого агент непрерывно взаимодействует с окружением, что отображено на рис. 1.

В рамках этой статьи действия и состояния являются элементами дискретных множеств. В общем случае они могут быть вещественными, однако рассмотрение этого вопроса выходит за рамки работы.

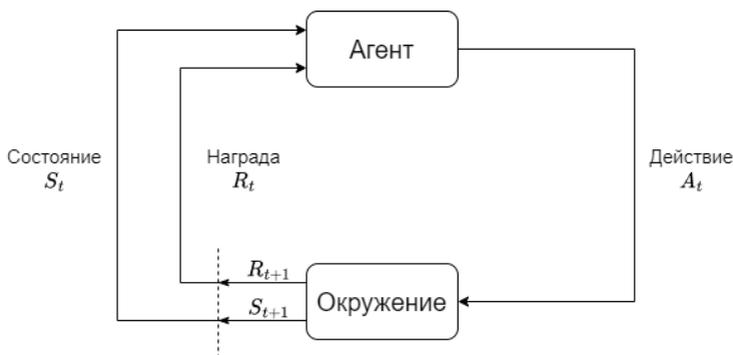


Рис. 1. Модель взаимодействия агента и окружения

2. Динамика перехода.

Процесс изменения состояния окружения называется динамикой перехода. В обучении с подкреплением предполагается, что этот процесс является марковским, то есть очередное состояние системы зависит только от предыдущего:

$$s', r \sim p(s', r | s, a).$$

Динамика перехода определяется совместным распределением состояния и награды: агент после каждого взаимодействия получает вознаграждение.

Важно отметить, что существует специальное состояние — терминальное, при переходе в которое эпизод заканчивается. Однако его может и не быть — в таком случае задача является бесконечной.

3. Стратегия агента.

Агент принимает решения на основании наблюдаемого состояния системы. Иначе говоря, агент действует согласно некоторой стратегии, являющейся распределением вероятности действий при условии текущего состояния системы:

$$a \sim \pi(a | s).$$

Стратегии могут быть и детерминированными, то есть каждому состоянию соответствует определенное действие.

4. Целевая функция.

Целью агента является максимизация общей дисконтированной награды. Так как задачи могут быть не только конечные, но и бесконечные, целевая функция в общем случае представляет собой ряд следующего вида:

$$G_t = R_{t+1} + R_{t+2} + \dots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}.$$

Коэффициент перед наградой называется дисконтирующим, в конечных задачах он полагается равным 1, в бесконечных должен быть меньше 1 для сходимости ряда.

Таким образом, цель агента – максимизировать дисконтированную награду, начиная с какого-то момента времени.

5. Уравнения Беллмана.

Для поиска оптимальных стратегий необходимо ввести функции, оценивающие ожидаемую дисконтированную награду агента в конце эпизода:

$$V(s) = E[G_t | S_t = s] = \sum_a \pi(a | s) \sum_{s', r} p(s', r | s, a)(r + \gamma V(s')),$$

$$Q(s, a) = E[G_t, S_t = s, A_t = a] =$$

$$= \sum_{s', r} p(s', r | s, a)(r + \gamma \sum_{a'} \pi(a' | s') Q(s', a')).$$

Эти функции задаются рекурсивно. Для того, чтобы определить их значения для некоторой стратегии, требуется решить систему линейных уравнений.

Однако для нахождения оптимальной стратегии необходимо записать оптимальные уравнения Беллмана:

$$V^*(s) = \max_a \sum_{s', r} p(s', r | s, a)(r + \gamma V^*(s')),$$

$$Q^*(s, a) = \sum_{s', r} p(s', r | s, a)(r + \max_{a'} Q^*(s', a')).$$

Особый интерес представляет вторая функция, так как с помощью нее можно записать оптимальную стратегию в следующем виде:

$$\pi^*(a | s) = \arg \max_a Q^*(s, a).$$

Оптимальным действием в некотором состоянии является действие, максимизирующее оптимальную функцию значений Q .

Для того, чтобы решить систему оптимальных уравнений Беллмана для функции Q , необходимо записать уравнения для всех пар состояний

и действий. Число этих пар может быть экстремально большим, поэтому этот подход вычислительно не эффективен.

6. Эпсилон-жадная стратегия.

В обучении с подкреплением играет большую роль баланс между исследованием среды и использованием полученных знаний для построения оптимальной стратегии. Существует множество способов решения этой проблемы, но самым простым является введение эпсилон-жадной стратегии:

$$\pi(a | s) = \begin{cases} 1 - \varepsilon + \frac{\varepsilon}{|A(s)|}, a = a^*, \\ \frac{\varepsilon}{|A(s)|}, \forall a \neq a^*. \end{cases}$$

С определенной вероятностью вместо оптимального действия, выбирается случайное неоптимальное. Без какого-либо исследования среды агент сможет найти лишь субоптимальную стратегию.

2. Q-обучение

В большинстве реальных задач динамика перехода среды неизвестна, поэтому одним из самых популярных методов обучения с подкреплением является Q-обучение, которое не использует вероятности перехода из одного состояния в другое. Его идея заключается в том, что оценка дисконтированной награды в следующем состоянии точнее, чем в текущем, так как оно находится ближе к концу эпизода. Этот алгоритм является итерационным, значение функции обновляется по следующему правилу:

$$Q(s, a) \leftarrow Q(s, a) + \alpha(r + \gamma \max_{a'} Q(s', a') - Q(s, a)),$$

где α – скорость обучения.

По достижении сходимости алгоритма, будет получена таблица значений функции для всех пар состояний и действий, которую можно использовать для построения оптимальной стратегии.

3. Применение Q-обучения к игре с детерминированными правилами

Для примера будет рассмотрена простая игра с фиксированными правилами.

7. Правила игры.

Имеется шестигранный честный кубик, то есть вероятности выпадения каждой грани одинаковы и равны $1/6$. Значения на всех гранях равны 1, 2, 3, 4, 5, 6.

Игроку предлагается бросить кубик, после чего у него есть два варианта:

- 1) забрать выпавшее число в качестве награды;
- 2) перебросить кубик и забрать новое выпавшее число в качестве награды.

Вопрос: при каком значении стоит перебрасывать кубик, а при каком забирать первоначальную награду?

Решение задачи очевидное: если выпало 1, 2, 3 – перебросить, если 4, 5, 6 – не перебрасывать. Однако можно убедиться в этом с помощью обучения с подкреплением.

8. Постановка задачи в виде модели агент-окружение.

Многие задачи обучения с подкреплением можно представить в виде графа переходов, представленного на рис. 2. На нем окружностями изображены все возможные состояния системы, прямоугольником – терминальное состояние, точками – два возможных действия: «бросить кубик» и «не бросить кубик». Из начального состояния доступно только первое действие – «бросить кубик», после чего агент может перейти в одно из шести состояний, соответствующих выпавшему при первом броске числу. В каждом из этих состояний агенту разрешается сделать оба действия. После чего он перейдет в терминальное состояние, и эпизод завершится.

Агент получает награду только после перехода в терминальное состояние.

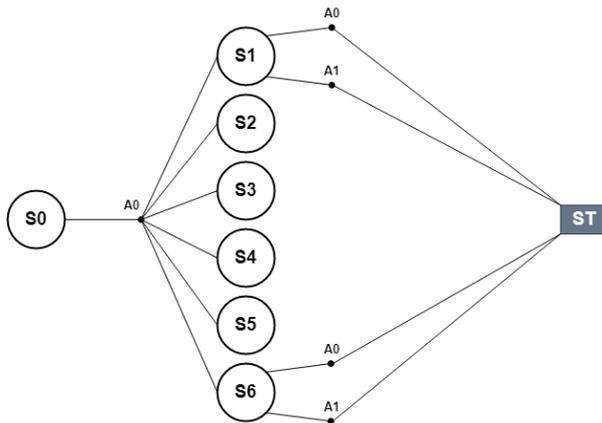


Рис. 2. Представление игры в виде графа переходов

9. Решение с помощью Q-обучения.

Модель агент-окружение и алгоритм Q-обучения были реализованы на языке Python с использованием пакета Numpy. Была создана библиотека, позволяющая реализовать любую игру с фиксированными правилами.

Для обучения использовалась эпсилон-жадная стратегия с экспоненциально затухающей вероятностью совершения субоптимального действия. Скорость обучения выбрана равной 0.01. Число итераций обучения – 1 000 000. Коэффициент дисконтирования равен 1, так как задача конечная. Результаты обучения приведены в таблице.

Таблица

Результаты обучения

	Бросить	Не бросать
S1	3.75	1.23
S2	3.76	2.23
S3	3.69	3.23
S4	3.77	4.23
S5	3.75	5.23
S6	3.75	6.23

Значения второго столбца сравниваются со значениями первого: если после первого броска выпало 1, 2, 3, то стоит перебросить кубик. Если же выпало 4, 5, 6, то перебрасывать кубик не нужно.

Следуя полученной оптимальной стратегии, средняя награда равняется 4.27. Если же выбирать действия случайным образом, средняя награда будет равна 3.5, что является математическим ожиданием единичного броска кубика. Процесс сходимости изображен на рис. 3.

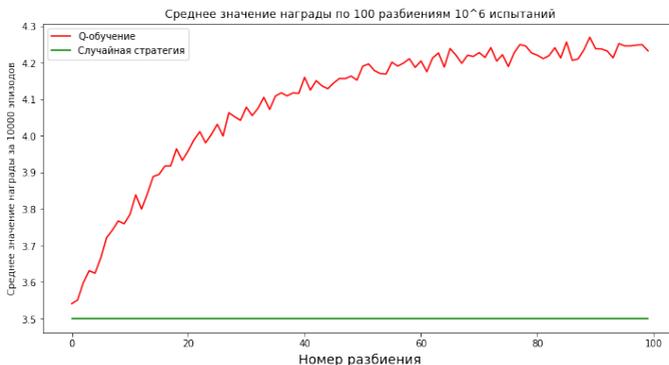


Рис. 3. Процесс сходимости Q-обучения

Заключение

В данной работе сформулированы основные понятия обучения с подкреплением и алгоритм Q-обучения, который был применен к игре с фиксированными правилами. Предположения об оптимальной стратегии были подтверждены результатами работы алгоритма Q-обучения. Также была реализована библиотека на языке Python, позволяющая реализовывать другие игры и находить в них оптимальные стратегии.

Список литературы

1. Sutton, R. S. Reinforcement learning: An introduction / R. S. Sutton, A. G. Barto. – MIT press, 2018.
2. Regehr, M. T. An Elementary Proof that Q-learning Converges Almost Surely [Электронный ресурс] / M. T. Regehr, A. Ayoub. – Режим доступа : <https://arxiv.org/abs/2108.02827>
3. Беллман, Р. Введение в теорию матриц / Р. Беллман. – М. : Наука, 1969.

Автоматизации формирования документов основных профессиональных образовательных программ высшего образования факультета компьютерных наук Воронежского государственного университета

А. И. Претков

Студент бакалавриата

Е. А. Копытина

Старший преподаватель

Введение

Большое количество электронных документов необходимо периодически актуализировать, однако, несмотря на большое количество функций для поиска и замены текста в документе эта задача является сложной и рутинной, особенно когда редактируемый документ связан с другим с документом и необходимо найти отличия между ними и только после этого внести необходимые изменения.

Данные действия требуют внимательности, большого количества времени и чреватые ошибками.

Данная проблема крайне актуальна в настоящее время и её решению посвящены работы [1-6]. Особенно вышеуказанная проблема характерна для высших учебных заведений: периодически появляются новые учебные стандарты и, соответственно, обновляется учебный план. В дальнейшем на основе измененного учебного плана нужно вносить изменения в документ с информацией об образовательной программе. Данный документ сокращенно называется «ОПОП».

Для устранения вышеописанных сложностей был разработан программный комплекс, формирующий Word-документ с информацией об образовательной программе на основе Excel-файла с учебным планом, предыдущей версии Word-документа образовательной программы, профессиональных стандартов, описывающих деятельность выпускников и информации об универсальных компетенциях, нормативных документах и материально-техническом обеспечении.

1. Постановка задачи

К описываемому в статье программному комплексу для автоматизации формирования документов основных профессиональных

образовательных программ высшего образования были выдвинуты следующие функциональные требования:

- Чтение информации об учебном плане из Excel-таблиц;
- Генерация Word-документа ОПОП на основе данных из учебного плана;
- Поддержка ввода дополнительной информации, отсутствующей в учебном плане.

2. Средства реализации программного комплекса

Проект реализуется посредством чтения Excel-таблиц и другой информации, переданной пользователем и формирования на их основе ОПОП документа, подстановкой редко изменяемой информации из предыдущей версии ранее созданного документа ОПОП, и копированием его форматирования в генерируемый.

Проект реализован на нескольких языках, поскольку их библиотеки для работы с офисными документами в некоторых моментах серьезно различаются, особенно в доступных настройках для форматирования документа.

В процессе работы были использованы следующие технологии:

- Java 8. Данный объектно-ориентированный язык программирования обладает готовыми библиотеками для решения многих прикладных задач и является строго типизированным, что облегчает написание кода и поиск ошибок. Использовался: для парсинга Excel-таблиц; для создания нового документа на основе данных полученных из Excel таблиц и введенной пользователем информации; для создания графического интерфейса; для автоматической загрузки информации о профессиональных стандартах;

- Python 3.10. Мощный объектно-ориентированный язык программирования, на котором написано множество библиотек. На этом языке были написаны утилиты для копирования редко изменяемой информации вместе с форматированием из уже созданного документа ОПОП в новый документ и конвертации информации из учебного плана в изображения для вставки в ОПОП: учебный план для каждого курса обучения и календарный учебный график;

- Idea Professional. Среда разработки для эффективного программирования Java-приложений с мощным отладчиком и подсказками при написании кода;

- PyCharm Professional. Среда разработки языка Python, функционально практически аналогична Idea за исключением ориентации на язык Python.

3. Реализация логики программного комплекса

Часть программного комплекса написана на языке Java. Для её реализации была использована библиотека Apache POI.

Часть программного комплекса считывает Excel-файл и формирует на его основе ОПОП документ.

Помимо этого, используется дополнительная информация, переданная пользователем – универсальные компетенции, нормативные документы, материально-техническое обеспечение и профессиональные стандарты.

Присутствует возможность автоматической загрузки информации о профессиональных стандартах из реестра профессиональных стандартов.

Программный комплекс состоит из классов.

Для классов по формированию содержания документа выделен отдельный интерфейс SectionWriter, содержащий метод writeSection, принимающий на вход формируемый Word-документ, Excel-документ с информацией об учебном плане и данные о направлении подготовки, для которого формируется ОПОП. Все реализации интерфейса SectionWriter вынесены в отдельный пакет docWriters.

Опишем реализации этого интерфейса подробнее:

– FirstSectionWriter отвечает за формирование первого раздела документа, состоящего из таблицы с содержанием документа, общих положений, информации о нормативных документах, используемых при формировании ОПОП для данного направления подготовки и перечня сокращений;

– SecondSectionWriter отвечает за формирования раздела документа с характеристикой профессиональной деятельности выпускников;

– ThirdSectionWriter отвечает за формирование общей характеристики основной профессиональной образовательной программы. На основе данных из учебного плана записывается профиль образовательной программы, квалификация, присваиваемая выпускникам, объем программы, срок получения образования и минимальный объем контактной работы по образовательной программе;

– FourthSectionWriter формирует планируемые результаты освоения ОПОП. На основе файла со списком категорий универсальных компетенций и учебного плана добавляет таблицу с информацией о каждой компетенции и индикаторах ее достижения, также формируется таблица общепрофессиональных компетенций: из учебного плана берутся коды компетенций, их формулировки и индикаторы достижения

компетенции и записываются в таблицу, аналогично формируется таблица с профессиональными компетенциями;

- FifthSectionWriter добавляет таблицу со структурой программы и объемом в зачетных единицах каждого пункта учебной программы и добавляет подробное описание каждого блока учебной программы;

- SixthSectionWriter добавляет матрицу соответствия компетенций, индикаторы их достижения и элемент ОПОП, приложения с учебными планами за каждый курс обучения,

- приложение с календарным учебным графиком из учебного плана, профессиональные стандарты и трудовые функции, относящиеся к направлению подготовки;

- SixthApplicationWriter записывает информацию о материально-техническом обеспечении;

- EighthDocApplicationWriter записывает аннотации рабочих программ дисциплин и практик на основе данных из учебного плана: общая трудоемкость дисциплины, формируемые компетенции и форма промежуточной аттестации.

Классы, описывающие графический интерфейс вынесены в пакет gui:

- CompetitionForm содержит графический интерфейс для добавления универсальных компетенций;

- DocumentsForm содержит графический интерфейс для добавления и удаления нормативных документов и документов с данными о материально-техническом обеспечении;

- MainAppForm содержит графический интерфейс для создания ОПОП и кнопки для перехода на другие формы;

- ProfStandartForm содержит графический интерфейс для добавления информации о профессиональных стандартах;

- StudyProgramsForm содержит графический интерфейс для добавления и удаления направлений подготовки;

- DocumentCreatingWorker предназначен для формирования ОПОП в отдельном потоке для предотвращения зависания графического интерфейса во время формирования документа;

- StudyProgramsComboBoxModel содержит реализацию модели данных для поля со списком, хранящую направления подготовки;

- TableModel содержит реализацию модели данных для таблицы с поддержкой сортировки по её полям;

- ProfStandartWorker предназначен для автоматической загрузки информации о профессиональном стандарте в отдельном потоке для предотвращения зависания графического интерфейса;

- LongCellRenderer предназначен для корректного вывода полной информации о направлениях подготовки, если длина превышает ширину экрана;

- MultiLineTableCellRenderer предназначен для корректного отображения ячеек таблицы, содержащих текст, не уместающийся по ширине в ячейку таблицы.

Классы, отражающие используемую в приложении модель данных вынесены в пакет Models:

- Document инкапсулирует информацию о документе – учебную программу к которой он относится, тип документа и его свойства;

- DocumentType содержит перечисление типов документов, которые могут быть созданы;

- EducationProgram содержит информацию о направлении подготовки – код специальности, название специальности, название профиля обучения и квалификация, присуждаемая выпускникам;

- StudyDegree содержит перечисление квалификаций, которые могут быть присвоены выпускникам;

- OporCreatingMode содержит перечисление с режимами формирования ОПОП.

Классы для вспомогательной логики вынесены в пакет Utils:

- ParagraphUtil содержит методы для выполнения часто повторяющихся в ходе работе программного комплекса действий, например, создание маркированного или нумерованного списка, добавление подпункта в списке, объединение ячеек таблицы, создание таблицы с заданными заголовками, строками и размером;

- ExcelReader содержит методы для работы с Excel-таблицей учебного плана и инкапсулирует в себе логику по получению данных в нужном формате из таблиц;

- FontConstants содержит константы с настройками шрифта, например, названия шрифта, стандартный размер шрифта в тексте;

- FontReplacer содержит метод для установки заданного шрифта в параграфах;

- DocumentMarginsSetter содержит методы для настройки отступов и размера страницы в документе;

- GuiUtils содержит методы для создания элементов графического интерфейса с поддержкой прокрутки и расположенных в центре экрана по горизонтали или вертикали;

- DatabaseConnector предназначен для взаимодействия приложения и встраиваемой базы данных SQLite;

– ProfStandartLoader предназначен для автоматической загрузки информации о профессиональном стандарте из реестра профессиональных стандартов;

– Logger предназначен для записи подробной информации об ошибках в работе приложения в отдельный файл.

Остальная работа по формированию документа связанная с функциональными возможностями, отсутствующим в общедоступных библиотеках Java, производится Python-скриптами, запускаемыми Java-приложением.

Данные скрипты при вызове могут получать входные параметры. Например, название Excel-листа с учебным планом.

Первый скрипт конвертирует Excel-таблицы с учебным планом в картинки для последующей вставки в Word-документ.

Второй скрипт считывает исходный Word-документ для переноса в формируемый ОПОП документ редко изменяемых данных вместе с их форматированием, например, условия осуществления образовательной деятельности, цель и задачи программы. Второй скрипт в силу более сложной логики содержит вспомогательные методы: `copy_paragraph` – копирует параграф вместе с его форматированием; `copy_font` – копирует настройки шрифта; `copy_style` – копирует настройки форматирования; функция `iter_block_items` позволяет в цикле обойти все элементы документа в правильном порядке; `move_table_after` возвращает таблицу, добавленную после заданного параграфа; `table_insert_paragraph_after` и `insert_paragraph_before` возвращают новый параграф, вставленный перед заданным параграфом; `insert_table_before` возвращает таблицу вставленную перед заданным элементом; `copy_paragraphs` копирует содержимое параграфов из одного документа в другой.

4. Реализация интерфейса

Взаимодействие с приложением идет через графический интерфейс, реализованный с помощью Java библиотеки Swing.

Работа с приложением начинается с главного окна, представленного на рис. 1.

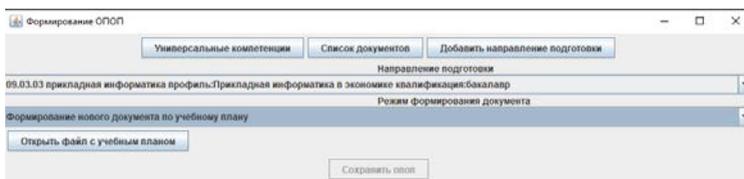


Рис. 1. Главное окно приложения

На главном окне пользователь может перейти к просмотру и редактированию универсальных компетенций, документов и направлений подготовки.

Просмотр и редактирование универсальных компетенций представлено на рис. 2.

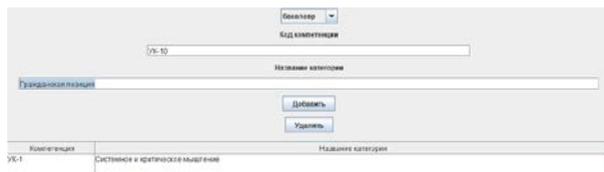


Рис. 2. Окно для просмотра и редактирования универсальных компетенций

Добавление и просмотр информации о направлениях подготовки представлено на рис. 3.



Рис. 3. Окно для просмотра и добавления направлений подготовки

Окно для просмотра и редактирования документов представлено на рис. 4. Пользователь должен выбрать направление подготовки, тип документа и добавить необходимые данные.

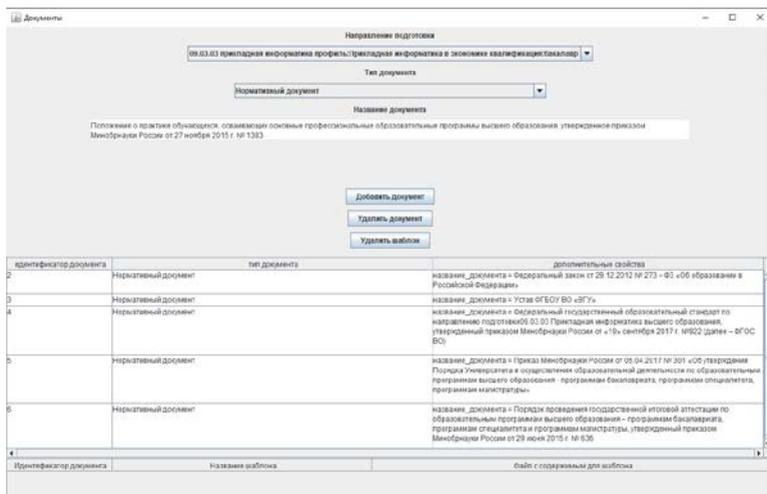


Рис. 4. Окно для просмотра и редактирования документов

Пользователь может добавить следующие типы документов, представленные на рис. 5.

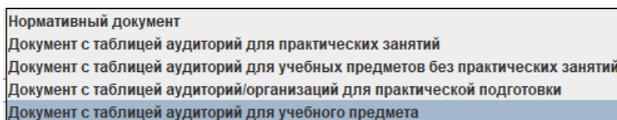


Рис. 5. Доступные типы документов

Для нормативных документов, по которым составляется ОПОП нужно ввести название документа. Данный функционал представлен на рис. 6.

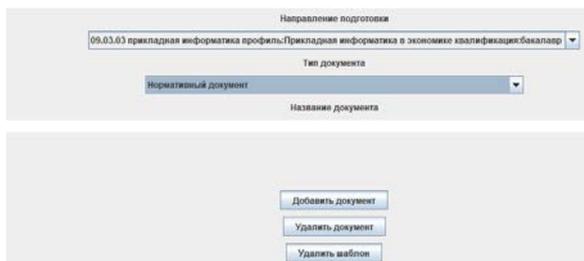


Рис. 6. Добавление нормативного документа

При добавлении документов с данными по аудиториям необходимо добавить файл с документом. Добавление информации на примере аудиторий для практических занятий представлено на рис. 7.

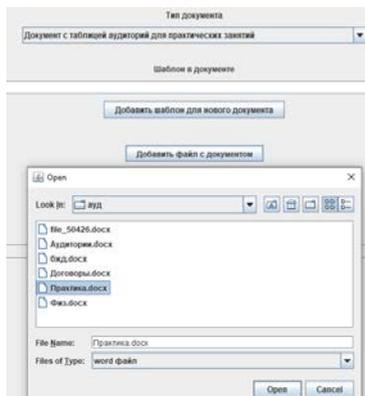


Рис. 7. Добавление информации об аудиториях для практических занятий

Помимо этого, есть возможность добавить шаблон и текстовый файл, содержимое которого будет подставляться вместо указанного шаблона. Добавление шаблона представлено на рис. 8.

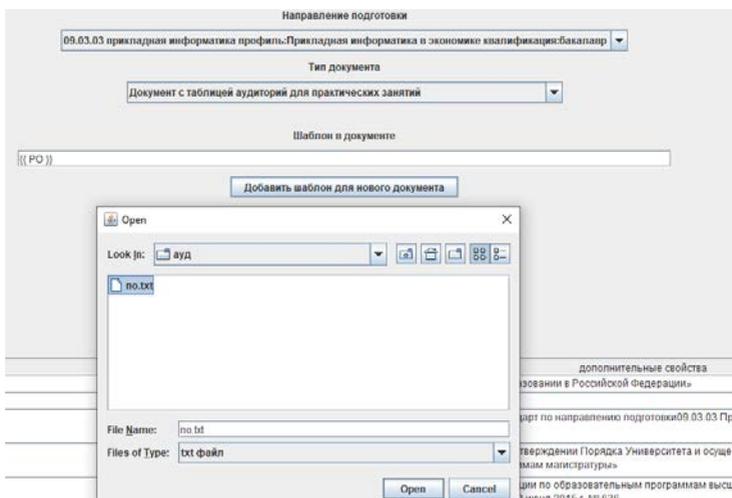


Рис. 8. Добавление шаблона

Также есть возможность добавить шаблон для уже добавленного документа, выбранного пользователем. Это продемонстрировано на рис. 9.



Рис. 9. Добавление шаблона для уже добавленного документа

При добавлении данных об аудиториях для определенного учебного предмета необходимо ввести код дисциплины. Это представлено на рис. 10.

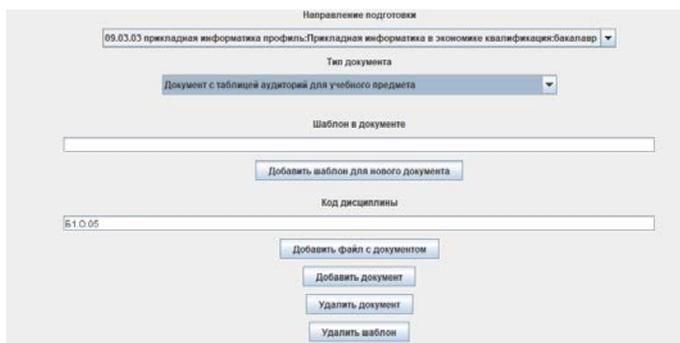


Рис. 10. Добавление документов с аудиториями для учебного предмета

После ввода всех вспомогательных данных при необходимости можно сменить режим формирования документа по умолчанию. Доступные режимы формирования документа представлены на рис. 11.

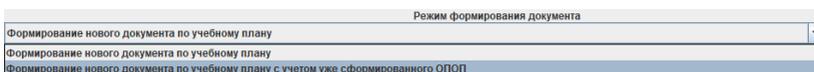


Рис. 11. Доступные режимы формирования документа

При формировании документа только по учебному плану нужно открыть файл с учебным планом, далее начнется загрузка данных о профессиональном стандарте, указанном в учебном плане. Работа в данном режиме представлена на рис. 12.

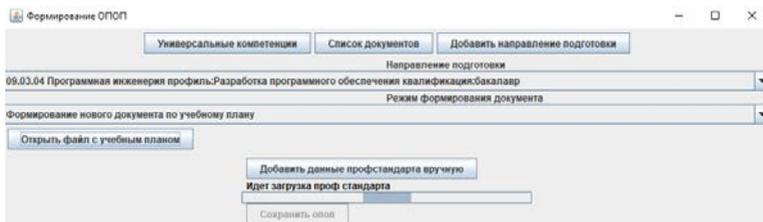


Рис. 12. Формирование ОПОП по учебному плану

При выборе режима формирования с учетом заранее сформированного ОПОП нужно помимо учебного плана указать файл с ОПОП. Этот режим демонстрируется на рис. 13.

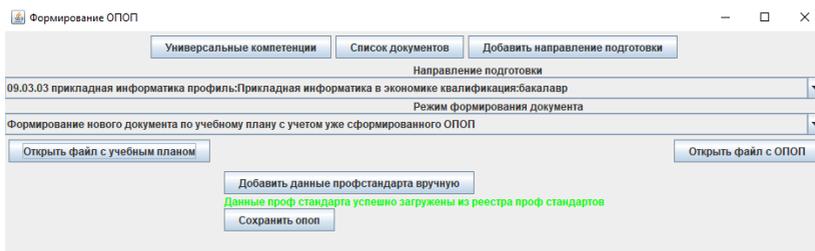


Рис. 13. Формирование ОПОП по учебному плану с учетом заранее сформированного ОПОП

Для случаев, когда невозможно автоматически загрузить данные о профессиональном стандарте есть возможность ручного ввода информации. Окно для добавления информации о профессиональных стандартах представлено на рис. 14 и 15.

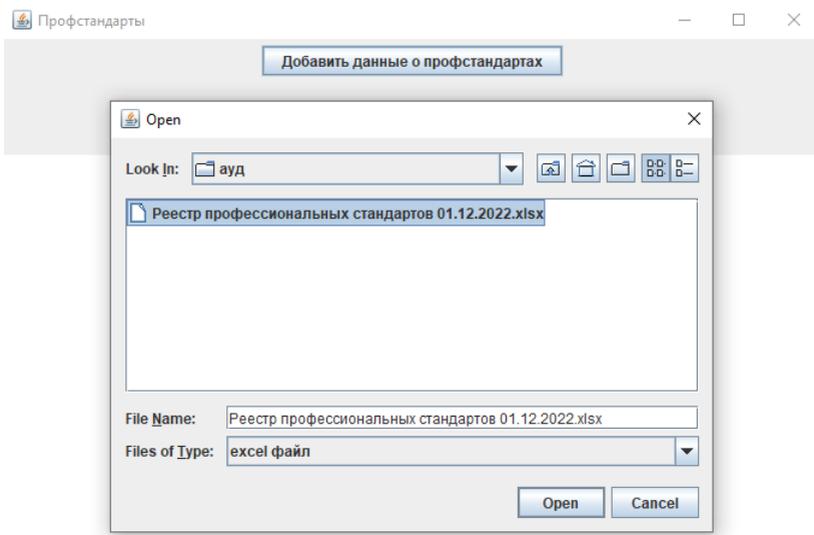


Рис. 14. Добавление общей информации о профстандартах

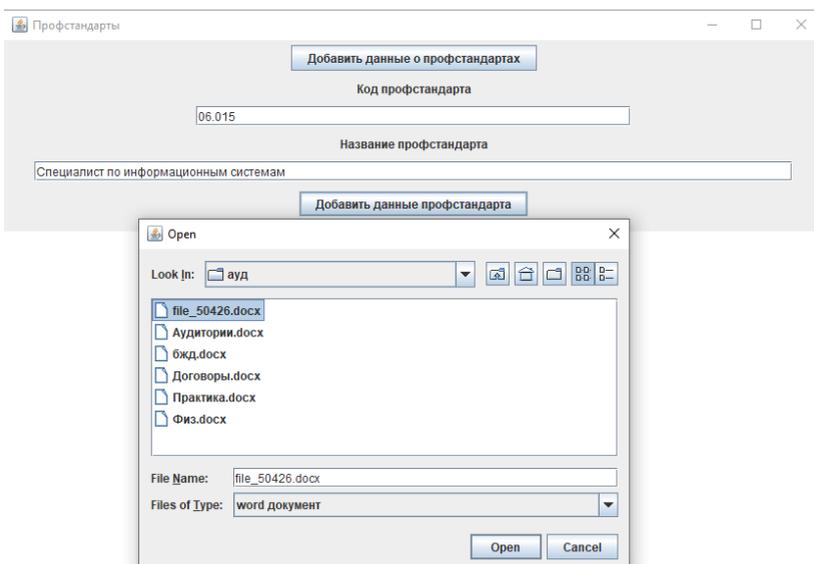


Рис. 15. Добавление информации о профстандарте

После ввода всех данных пользователь нажимает кнопку «Сохранить опоп», выбирает место для сохранения файла и видит индикацию процесса создания документа, это демонстрируется на рис. 16 и 17.

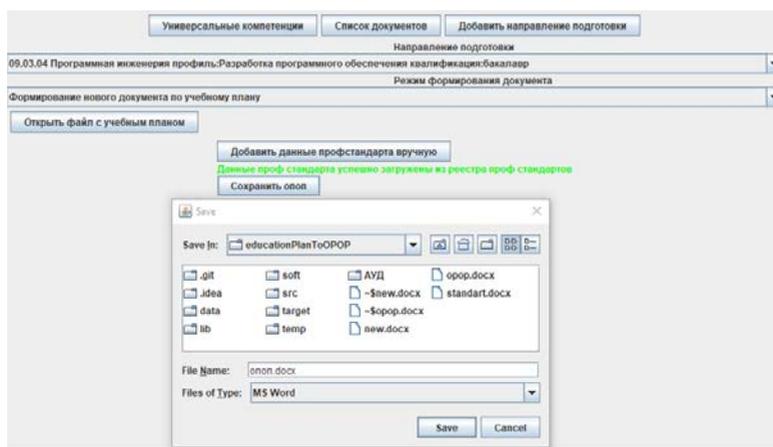


Рис. 16. Выбор места для сохранения сформированного ОПОП

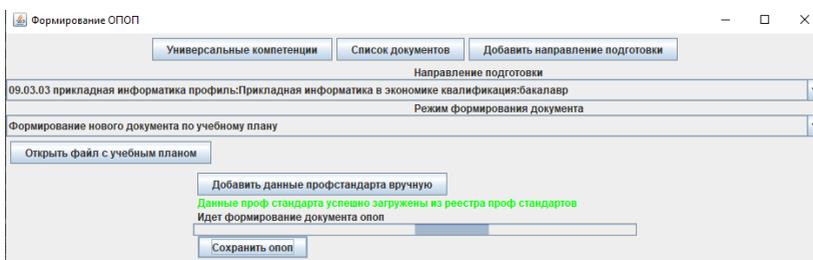


Рис. 17. Индикация процесса формирования ОПОП

При успешном формировании документа пользователь увидит сообщение о том, что документ сформирован и после закрытия сообщения сформированный документ будет автоматически открыт в программе, ассоциированной для работы с docx файлами.

Заключение

В данной статье была рассмотрена реализация, графический интерфейс и основная информация по работе с программным комплексом для автоматизации формирования документа основной профессиональной образовательной программы высшего образования.

Список литературы

1. Реализация проекта по созданию реляционной базы данных для Контрольно-счётной палаты Воронежской области / А. Г. Потапов, А. В. Порядин, Т. Д. Братышев [и др.] // Труды молодых ученых факультета компьютерных наук ВГУ. Том Выпуск 2. – Воронеж : ООО «ВЭЛБОРН», 2022. – С. 132-137. – EDN FEIMSS.

2. Разработка парсера для автоматизации деятельности Контрольно-счётной палаты Воронежской области / А. В. Порядин, А. Г. Потапов, Т. Д. Братышев [и др.] // Труды молодых ученых факультета компьютерных наук ВГУ. Том Выпуск 2. – Воронеж : ООО «ВЭЛБОРН», 2022. – С. 127-131. – EDN TMFGUT.

3. Кропачев, А. Н. Проектирование системы классификации правонарушений наркоконтроля по Воронежской области / А. Н. Кропачев, Е. А. Копытина // Труды молодых ученых факультета компьютерных наук ВГУ. Том Выпуск 2. – Воронеж : ООО «ВЭЛБОРН», 2022. – С. 96-101. – EDN EZCGIV.

4. 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.

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. Алгазинова, Воронеж, 10–12 февраля 2022 года / Под редакцией Д.Н. Борисова. – Воронеж: Общество с ограниченной ответственностью "Вэлборн", 2022. – С. 1342-1355. – EDN ONMOJZ.

6. 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.

Разработка сервиса определения местоположения устройства на основе GSM и Wi-Fi сетей

Д. Д. Кравцов

Студент бакалавриата

А. А. Вахтин

Доцент

Введение

Отслеживание местоположения применяется во многих сферах – от транспорта и логистики до промышленности, офисов, выставок и музеев.

Технология способствует оптимизации процессов транспортировки и позволяет пассажирам быстро найти путь к нужному месту внутри зданий аэропортов, железнодорожных вокзалов. С ее помощью можно строить интерактивные карты с точками интереса, отслеживать потоки людей, обеспечивать перемещение персонала транспортных узлов. Постоянный мониторинг пассажирских перевозок позволяет быстро реагировать на меняющиеся условия и улучшать качество обслуживания пассажиров.

Компании, которые занимаются логистикой и складированием, могут использовать систему для отслеживания комплектации заказов, перемещения транспорта, грузов. Система предоставляет детальную информацию о местоположении активов и помогает автоматизировать многие складские операции. За счет постоянной навигации сотрудников склада можно корректировать повседневные задачи в реальном времени, повышать производительность труда, без промедлений реагировать на внештатные ситуации.

Позиционирование может успешно внедряться в системы определения местоположения внутри помещений при эксплуатации крупных административных зданий. Система обеспечивает построение пошаговых маршрутов к нужному кабинету, отправляет уведомления и подсказки, которые ориентированы на месторасположение человека. При использовании технологии, руководство предприятия получает возможность контролировать передвижение офисных работников и при необходимости делегировать задачи или обязанности.

Благодаря непрерывному сбору информации о перемещениях в офисе можно выполнять точную аналитику и оптимизировать работу офисных систем.

Посредством позиционирования можно создавать гиды и мобильные приложения, которые улучшат работу выставок и музеев. Беспроводная система позиционирования в помещении помогает составлять интерактивные карты залов, разрабатывать короткие маршруты к нужному экспонату. Организаторы выставок могут собирать геоданные по посетителям и тщательно изучать их поведение, чтобы более рационально планировать экспозиции.

1. Определение местоположения устройства с помощью точек доступа Wi-Fi

Для получения координат с точностью до нескольких метров, будем использовать Wi-Fi-сети. Для получения точек доступа Wi-Fi воспользуемся сервисом Wigle. Wigle – сайт, который объединяет местоположение и другие данные о беспроводных сетях по всему миру. Эти данные собираются добровольцами, которые загружают приложение на свои телефоны, и приложение регистрирует все точки доступа, с которыми они сталкиваются, и их GPS-координаты. Затем все эти данные поступают в базу данных Wigle. Затем эти данные представляются пользователям на простом в использовании веб-сайте и в приложении (рис. 1).

Для получения координат точек доступа необходимо передать идентификатор точек доступа, в зоне которых находится устройство. После получения данных, определение местоположения устройства выполняется с помощью тех же самых способов, что и для GSM вышек[1].

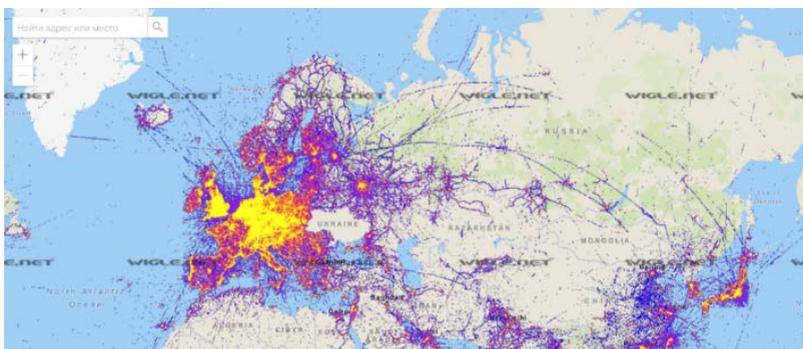


Рис. 1. Изображение доступных точек доступа Wi-Fi в Европе

2. Определение местоположения устройств на основе базовых станций

Самая масштабная Open-source база данных с координатами базовых станций – OpenCellId. Позволяет получить координаты базовых станций по странам. Было принято решение из списка стран выбрать Россию.

Все координаты базовых станций были добавлены в базу данных их количество составило 2 000 072 361.

В таблице, схема которой представлена на рис. 2, обозначили следующие поля:

- mcc – код страны;
- mnc – идентификатор оператора;
- cid– идентификатор базовой станции (уникален в рамках сотового оператора);
- lac – код зоны расположения;
- lon – долгота базовой станции;
- lat–широта базовой станции.

coordinates	
PK	mcc
PK	mnc
PK	cid
PK	lac
	lon
	lat

Рис. 2. Структура Entity моделей

Будет использоваться база данных, состоящая из одной таблицы coordinates. Она будет содержать в себе данные о координатах базовых станций и составной первичный ключ.

В таблицу входят такие данные как код страны, идентификатор оператора, идентификатор базовой станции, долгота базовой станции, широта базовой станции.

3. Алгоритм триангуляции

Способ определения местоположения с помощью триангуляции заключается в нахождении расстояния до объекта для всех доступных передатчиков сигнала, например, базовых станций [2](рис. 3).

Уровень сигнала от передатчика (dbm) можно перевести в расстояние. Полученное расстояние будем считать зоной действия

передатчика. Стоит учитывать на уровень сигнала может влиять множество факторов, что означает, что прямой корреляции быть не может и можно только примерно определить их зависимость друг от друга [3] (рис. 4).



Рис. 3. Объект в зоне видимости базовых станций

Формула затухания сигнала будет иметь вид:

$$\Delta B = 32.44 + 20 \cdot (\log F) + 20 \cdot (\log D), \quad (1)$$

где D – расстояние (м), F – частота (Гц).

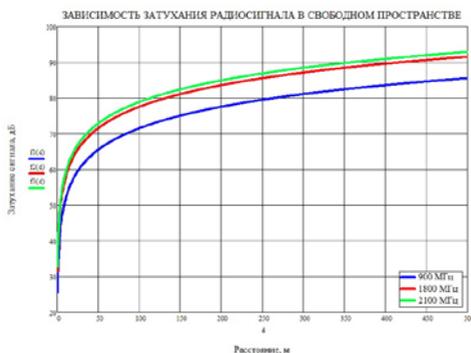


Рис. 4. График зависимости затухания сигнала от расстояния в свободном пространстве для разных частот

При идеальных условиях, объект будет находиться точно в центре пересечения обозначенных нами зон действий передатчиков.

Таким образом, для начала определение местоположения стоит построить окружности, центрами которых будут координаты передатчика, а в качестве радиуса используем децибелы (dbm), переведённые в расстояние.

Формула перевода уровня сигнала в метры:

$$m = 10^{\frac{dbm + k_1 + k_2 \cdot (\ln(F))}{-k_2 \cdot 1000}}, \quad (2)$$

где dbm –затухание сигнала (дБ), F – частота (Гц), D – расстояние (м), k1 – коэффициент для расчёта расстояния = 32.44, k2 – коэффициент для расчёта расстояния = 23.

Искомая координата будет находиться в центре пересечения окружностей.

Так как условие не идеальны, не все построенные окружности будут между собой пересекаться. Дальнейший действия будут производиться в зависимости от количества пересечений построенных окружностей и количества базовых станций, в зоне видимости которых находится устройство. Так как отсутствие точек пересечения окружностей следствие погрешностей в вычислении расстояние от устройства до передатчика, можно с точностью сказать, что в действительности устройство находится под покрытием сигнала всех обнаруженных передатчиков, таким образом, можно сказать, что точки пересечения окружностей будут решениями системы уравнений:

$$\begin{cases} (x_1 - a_1)^2 + (y_1 - b_1)^2 = R_1^2 \\ (x_2 - a_2)^2 + (y_2 - b_2)^2 = R_2^2 \\ \dots \\ (x_n - a_n)^2 + (y_n - b_n)^2 = R_n^2 \end{cases} \quad (3)$$

Для написания алгоритма рассмотрим несколько случаев.

2. Объект находится в зоне видимости 3 и более передатчиков все они пересекаются между собой (рис. 5).

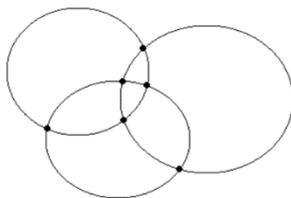


Рис. 5. Схема определения местоположения, когда устройство находится в зоне видимости трёх передатчиков

В данном случае точкой местоположения устройства будет считаться центр тяжести точек пересечения окружностей.

3. Объект находится в зоне видимости нескольких передатчиков, имеется две точки пересечения построенных окружностей (рис. 6).

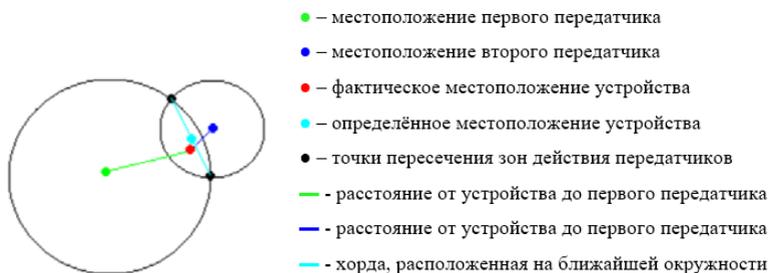


Рис. 6. Схема определения местоположения, когда устройство находится в зоне видимости нескольких передатчиков

4. Объект находится в зоне видимости 1 передатчика.

В этом случае лучшим решением будет считать, что координаты объекта совпадают с координатами передатчика, иначе в общем случае погрешность только увеличится.

4. Внедрение в производство

После разработки микросервиса, удалось договориться с компанией ООО «ЭКТОСТРОЙ» о внедрении данного функционала в админ-панель личного кабинета. Данные о местоположении вышались системой ectoControl.

«ectoControl» – система дистанционного мониторинга и управления с помощью телефона и ПК, разработанная компанией Эктострой (рис. 7). Использует GSM и Wi-Fi протоколы связи. Идеально подходит для контроля загородных домов и серверных комнат, т.к. позволяет мониторить температуру, влажность и другие параметры, а также хранить показания в личном кабинете.



Рис. 7. Система ectoControl V3.3

Процесс внедрения в серверную часть продукта состоит из следующих пунктов:

1. Определение целей, возможностей и ограничений проекта.
2. Определение взаимодействия между компонентами системы для внедрения.
3. Интегрирование микросервиса текущей системой.
4. Запуск микросервиса в производство.
5. Анализ результатов и оценка эффективности работы микросервиса в производстве.

5. Определение целей, возможностей и ограничений проекта

В ходе анализа требований выяснилось, что компания хочет знать местонахождение своих систем с точностью от 5 до 100 метров. Для решения этой задачи вполне подходят технологии, используемые разработанным сервисом.

Данные о GPS и Wi-Fi сетях системы шлют в виде бинарных пакетов раз в 6 часов или через пол часа после перезагрузки. Это не позволит отслеживать их местоположение в реальном времени, однако в данном случае такой задачи не стоит, что означает отсутствие ограничений как таковых

6. Определение взаимодействия между компонентами системы для внедрения

Взаимодействие компонентов между собой можно представить в виде схемы (рис. 8).

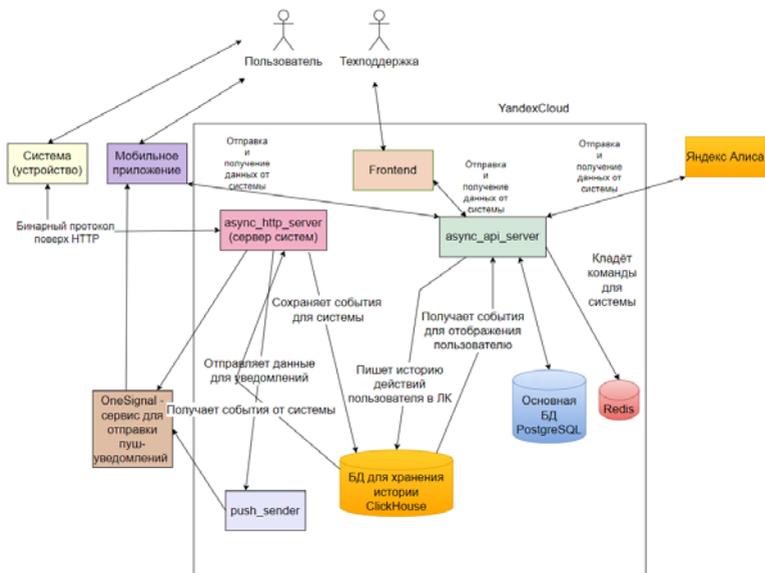


Рис. 8. Диаграмма взаимодействия компонентов системы, в которую будет встраиваться сервис

7. Анализ результатов и оценка эффективности работы сервиса в производстве

Спустя время, системы, находящиеся на связи, выслали свои данные на сервер, что позволило определить их местоположение.

На первый взгляд можем сказать, что определение локации систем удалось и мы можем наблюдать местонахождение нескольких тысяч систем, находящихся по всей России (рис. 9).

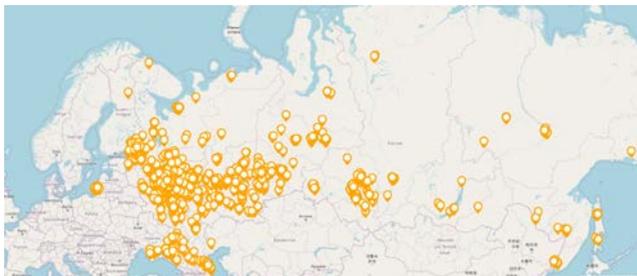


Рис. 9. Карта местоположения систем, определённое с помощью триангуляции

Однако, проведём более детальный анализ. Для этого используем устройства, локация которых заранее известна. Можем наблюдать, что все системы, находящиеся в одном здании, отображаются с приемлемой погрешностью (рис. 10).

Определим зависимость погрешности вычисления местоположения от количества базовых станций на примере системы из офиса.

После получения и анализа пакета с данными о локации системы, видим координаты 4 базовых станций, в зоне действия которых находится устройство. После вычисления местоположения посредством триангуляции, определим расстояние между этими координатами, переведа их в радианы и используя формулу вычисления расстояния.

Формула для перевода градусов в радианы:

$$r = \frac{n\pi}{180}, \quad (4)$$

где n – значение в градусах, r – значение в радианах.

$$dist = ER \cdot (2 \cdot \arcsin) \cdot$$

$$\cdot \sqrt{\sin(\Delta \frac{lat}{2})^2 + \cos(lat1) \cdot \cos(lat2) \cdot \sin(\Delta \frac{lon}{2})^2}, \quad (5)$$

где $dist$ – расстояние между двумя точками, ER – радиус земли, Δlat – разница между широтами двух точек, Δlon – разница между долготой двух точек, $lat1$ – широта первой точки, $lat2$ – широта второй точки.

Получим погрешность в 140м. Проигнорируем данные о всех вышках, кроме двух ближайших и повторим расчёты.

Получим погрешность в 310м. При выборе вышек с более низким уровнем сигнала, точность позиционирования, снизится ещё сильнее. Произведя расчёты, учитывая только ближайшую базовую станцию, получим погрешность в 210м. Такая точность следствие относительно близкого расположения GSM вышки. В общем случае, точность будет значительно хуже, чем в остальных рассмотренных случаях.

Проанализировав полученные значения, можно сказать, что в общем случае, определение местоположения с большим количеством базовых станций, даст более точную информацию о местоположении устройства. Это хорошо заметно на примере систем, расположенных в Москве и Санкт-Петербурге, чьё расположение определяется с большой точностью, и системы в Норильске, точность определение местоположения которой ниже, чем систем, находящихся в очень крупных городах.



а – в городе Воронеж, б – в городе Москва, в – в городе Санкт-Петербург, г – в городе Норильск

Рис. 10. Системы, находящиеся в офисе

Заключение

Данная статья была посвящена разработке сервиса, который определяет местоположение устройства на основе GSM и Wi-Fi сетей. Он может быть полезным в различных областях, включая производственную сферу, логистику, транспорт и т.д. Сервис позволяет определить местоположение устройства с высокой точностью, что помогает улучшить качество обслуживания и повысить эффективность бизнес-процессов.

В процессе работы был проведен анализ существующих методов определения местоположения устройств, и были выбраны GSM и Wi-Fi сети, как наиболее подходящие для решения задачи определения местоположения. Была проведена разработка и реализация сервиса, который позволяет определить местоположение устройства, используя сигналы от базовых станций GSM и Wi-Fi точек доступа.

Для тестирования сервиса были проведены эксперименты, результаты которых показали, что сервис обладает высокой точностью определения местоположения устройства. Были обнаружены некоторые

ограничения в работе сервиса, такие как наличие помех от других радиочастотных источников, однако они были сведены к минимуму.

В целом, результаты данной работы показывают, что разработанный сервис определения местоположения устройства на основе GSM и Wi-Fi сетей может быть полезным инструментом в различных областях. Его точность и удобство использования помогут улучшить качество обслуживания и повысить эффективность бизнес-процессов.

Список литературы

1. Кокорева, Е. В. Результаты натурных испытаний системы определения местоположения Wi-Fi / Е. В. Кокорева, А. Е. Костюкевич // Экономика и качество систем связи. – 2021. – № 1. – С. 65-67.
2. Быбка, А. И. Особенности позиционирования абонентов в сети GSM / А. И. Быбка, А. В. Дерев, А. Н. Зеленин // Восточно-Европейский журнал передовых технологий. – 2009. – № 1. – С. 61-62.
3. Рудневский, А. В. Определение местоположения по базовым станциям GSM / А. В. Рудневский // Беспроводные технологии. – 2001. – № 3. – С. 16-18.

Приближенные формулы для построения узловых функций на основе систем целочисленных сдвигов функции Гаусса и базисных сплайнов высокого порядка

А. Е. Матюхин

Студент бакалавриата

А. А. Махно

Студент бакалавриата

Е. А. Киселев

Доцент

Введение

В настоящее время интерполяционные алгоритмы широко применяются в цифровой обработке сигналов и компьютерной графике. Поскольку шаг дискретизации на практике чаще всего постоянен, особого внимания заслуживают семейства равномерных сдвигов. Наиболее популярны системы сдвигов, порожденные функцией Гаусса и базисными сплайнами [1, 2]. Именно они и будут подробно рассмотрены в рамках данной работы.

Задачу об интерполяции удобно решать с помощью вспомогательной конструкции под названием узловая функция. К сожалению, явные формулы для ее построения либо отсутствуют, либо достаточно сложны. Из-за этого алгоритмы интерполяции обладают высокой вычислительной сложностью, а этого хотелось бы избежать на практике. В рамках данной работы мы предлагаем несколько простых приближенных формул в случае функции Гаусса и базисных сплайнов, которые позволяют строить узловые функции с точностью до 10%, считая по равномерной норме.

1. Интерполяция и узловые функции

Пусть $f(x)$ – исследуемый аналоговый сигнал. Проведем его дискретизацию с шагом h , в результате чего получим набор значений $f_k = f(kh)$, $k \in Z$. В дальнейшем для простоты, не ограничивая общности, примем шаг сетки h равным единице. Соответственно, вместо равномерных сдвигов будем использовать целочисленные.

Требуется построить интерполирующую функцию $\tilde{f}(x)$, которая совпадает с $f(x)$ в целочисленных точках $x = k$, $k \in \mathbb{Z}$. Рассмотрим ситуацию, когда $\tilde{f}(x)$ строится на основе системы целочисленных сдвигов $\varphi(x-k)$, $k \in \mathbb{Z}$:

$$\tilde{f}(x) = \sum_{k=-\infty}^{\infty} c_k \varphi(x-k), \quad (1)$$

где c_k – неопределенные коэффициенты. Удобной вспомогательной конструкцией для нахождения $\tilde{f}(x)$ является узловая функция [1, 2].

Определение 1. Функция $\tilde{\varphi}(x)$, являющаяся линейной комбинацией целочисленных сдвигов $\varphi(x)$

$$\tilde{\varphi}(x) = \sum_{k=-\infty}^{\infty} d_k \varphi(x-k), \quad (2)$$

называется узловой, если она удовлетворяет условию

$$\tilde{\varphi}(m) = \delta_{0,m}, m \in \mathbb{Z}. \quad (3)$$

Если $\tilde{\varphi}(x)$ известна, то интерполирующую функцию $\tilde{f}(x)$ тогда получить довольно легко по формуле

$$\tilde{f}(x) = \sum_{k=-\infty}^{\infty} f_k \tilde{\varphi}(x-k). \quad (4)$$

2. Система целочисленных сдвигов функции Гаусса

Рассмотрим случай, когда

$$\varphi(x) = \exp\left(-\frac{x^2}{2\sigma^2}\right), \sigma > 0. \quad (5)$$

Известен следующий результат [1].

Утверждение 1. Пусть $\varphi(x)$ определяется равенством (5). Тогда формула для коэффициентов узловой функции из формулы (2) имеет вид

$$d_k = \frac{\exp(k^2/(2\sigma^2))}{K(\sigma)} \sum_{p=|k|}^{\infty} (-1)^p \exp\left(-\frac{(2p+1)^2}{8\sigma^2}\right), \quad (6)$$

$$K(\sigma) = \sum_{p=-\infty}^{\infty} (4p+1) \exp\left(-\frac{(4p+1)^2}{8\sigma^2}\right). \quad (7)$$

Соотношение (6) хоть и содержит быстро сходящийся ряд, но все равно требует привлечения значительных вычислительных ресурсов. Дело в том, что для надежного восстановления узловой функции требуется иногда довольно много коэффициентов d_k . Например, при $\sigma = 1$ их необходимо учесть порядка 50 [3, 4]. Поэтому в рамках данной работы мы поставили себе целью получить более простую чем (6) приближенную формулу.

Для начала докажем вспомогательную предельную теорему.

Теорема 1. Пусть $\varphi(x)$ определяется равенством (5). Тогда имеет место следующее предельное соотношение:

$$\lim_{k \rightarrow \infty} \left| d_k \exp\left(\frac{k}{2\sigma^2}\right) \right| = \frac{\exp(-1/(8\sigma^2))}{K(\sigma)}. \quad (8)$$

Доказательство. Прежде всего, заметим, что ряд в формуле (6) сходится равномерно, так как функция Гаусса относится к быстро убывающим. Сделаем замену индекса $q = p - |k|$ и внесем экспоненциальный множитель внутрь суммы:

$$d_k = \frac{1}{K(\sigma)} \sum_{q=0}^{\infty} (-1)^{q+|k|} \exp\left(\frac{4k^2 - (2q + 2|k| + 1)^2}{8\sigma^2}\right). \quad (9)$$

Теперь умножим d_k на $\exp(k/(2\sigma^2))$ и возьмем модуль. После некоторых упрощений приходим к следующему равенству:

$$\left| d_k \exp\left(\frac{k}{2\sigma^2}\right) \right| = \frac{\exp(-1/(8\sigma^2))}{K(\sigma)} \left| \sum_{q=0}^{\infty} (-1)^q \exp\left(-\frac{q^2 + q(2|k| + 1)}{2\sigma^2}\right) \right|. \quad (10)$$

Все слагаемые суммы, за исключением первого, стремятся к нулю при $k \rightarrow \infty$. В силу равномерной сходимости ряда к пределу можно переходить почленно. Отсюда получаем непосредственно соотношение (8). Теорема доказана.

Отметим еще одно обстоятельство. Как было доказано в статье [4], коэффициенты d_k знакопереваются. Опираясь на этот факт и теорему 1, мы эмпирически получили следующую приближенную формулу:

$$d_k \approx (-1)^k d_0 \exp\left(-\frac{|k|}{2\sigma^2} + \lambda \left(1 - \exp\left(-\frac{|k|}{\sigma^2}\right)\right)\right), \quad (11)$$

$$\lambda = -\frac{1}{8\sigma^2} - \ln(d_0 K(\sigma)). \quad (12)$$

Формула (12) обеспечивает выполнение доказанной нами теоремы 1.

Перейдем к вычислительным экспериментам. На рис. 1 представлены графики модулей коэффициентов узловой функции, рассчитанных по точным формулам (6) и (7), а также по предложенным нами приближенным формулам (11) и (12). Как видим, (11) и (12) обеспечивают достаточно хорошее качество аппроксимации d_k . На рис. 2 представлены графики соответствующих узловых функций. Отличие по равномерной норме составляет не более 5%. Если $\sigma > 1.2$, приближенная формула работает заметно хуже, но практически важным является диапазон значений параметра σ примерно от 0.5 до 1.5 [3].

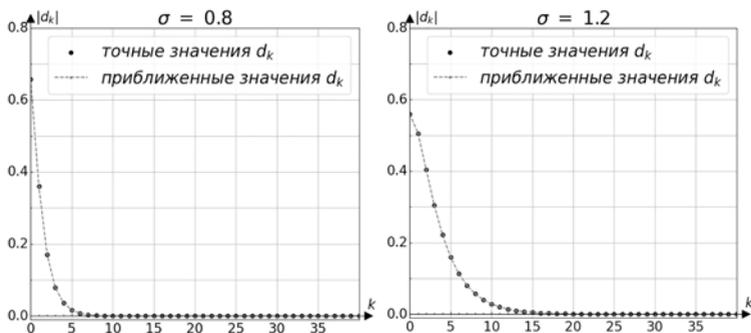


Рис. 1. Значения коэффициентов узловой функции на основе системы сдвигов функции Гаусса при $\sigma = 0.8$ и $\sigma = 1.2$

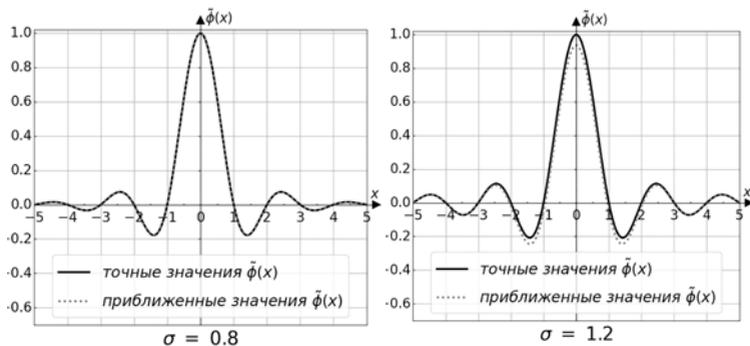


Рис. 2. График узловой функции на основе системы сдвигов функции Гаусса при $\sigma = 0.8$ и $\sigma = 1.2$

3. Базисные сплайны

Пусть $\chi_{[a,b]}(x)$ – характеристическая функция отрезка $[a,b]$, т. е.

$$\chi_{[a,b]}(x) = \begin{cases} 1, x \in [a,b], \\ 0, x \notin [a,b]. \end{cases} \quad (13)$$

Тогда центрированные базисные сплайны $B_n(x)$ определяются рекуррентно с помощью соотношений [1]

$$B_1(x) = \chi_{[-0.5,0.5]}(x), \quad (14)$$

$$B_{n+1}(x) = \int_{-\infty}^{\infty} B_n(x-t)B_1(t)dt, n > 1. \quad (15)$$

Узловая функция $\tilde{\varphi}(x)$, построенная на основе целочисленных сдвигов базисных сплайнов, носит специальное название – фундаментальный сплайн. Для сплайнов порядка $n \leq 10$ известны явные формулы для расчета коэффициентов d_k . Например,

$$\begin{aligned} d_k &= (-1)^k (3 - 2\sqrt{2})^{|k|}, n = 3; \\ d_k &= (-1)^k \sqrt{3}(2 - \sqrt{3})^{|k|}, n = 4. \end{aligned} \quad (16)$$

С ростом n эти формулы заметно усложняются, приводя при $n > 10$ к очень громоздким выражениям.

Далее нам понадобится следующий факт [5].

Утверждение 2. Справедливо предельное соотношение

$$\lim_{n \rightarrow \infty} \left(B_n(x) - \frac{1}{\sigma_n \sqrt{2\pi}} \exp\left(-\frac{x^2}{2\sigma_n^2}\right) \right) = 0, \sigma_n = \sqrt{\frac{n}{12}}. \quad (17)$$

Оказывается, уже при $n = 10$ базисный сплайн и соответствующая функция Гаусса слабо отличаются. Эта ситуация представлена на рис. 3.

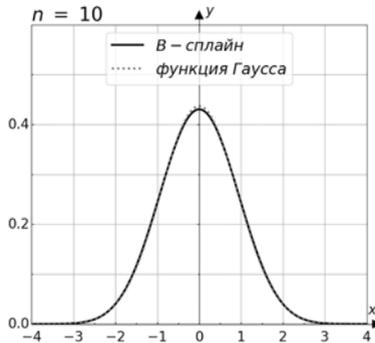


Рис. 3. Функция Гаусса и $B_{10}(x)$

Мы предлагаем следующее: заменить при $n > 10$ точные громоздкие формулы для коэффициентов d_k приближенным соотношением (11), положив в нем $\sigma = \sigma_n = \sqrt{n/12}$.

Перейдем к вычислительным экспериментам. На рис. 4 представлены графики коэффициентов фундаментального сплайна. Как видим, качество аппроксимации заметно хуже, чем в случае функции Гаусса. На рис. 5 представлены графики соответствующих фундаментальных сплайнов. Отличие по равномерной норме составляет не более 10%.

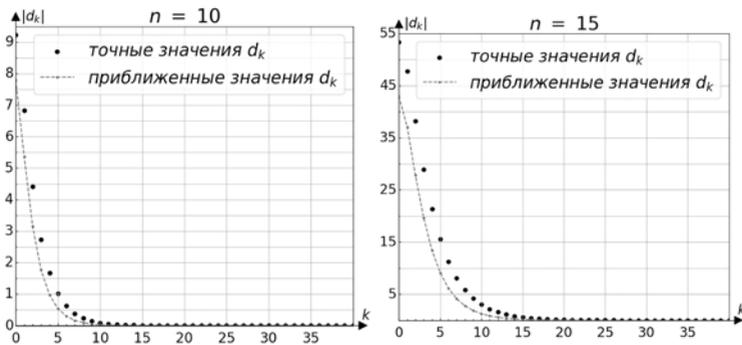


Рис. 4. Значения коэффициентов узловой функции на основе системы сдвигов базисных сплайнов при $n = 10$ и $n = 15$

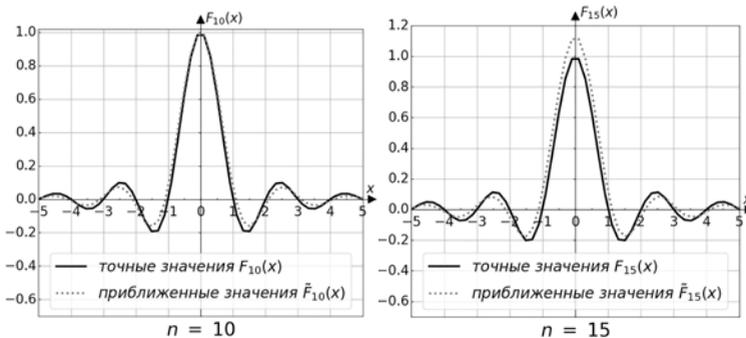


Рис. 5. Фундаментальные сплайны при $n = 10$ и $n = 15$

Заключение

В данной работе получены несколько новых приближенных соотношений для коэффициентов узловых функций, порожденных

сдвигами функции Гаусса и базисными сплайнами. В случае функции Гаусса формула оказывается применимой для значений параметра ширины $\sigma = 0.5 \div 1.5$, что наиболее востребовано на практике. Погрешность, с которой рассчитывается узловая функция по этим коэффициентам, составляет в среднем менее 10%, считая по равномерной норме в узловых точках. Аналогичная формула получена и для базисных сплайнов. Порядок сплайнов при этом должен лежать примерно в диапазоне $n = 10 \div 30$. В дальнейшем планируется провести серию вычислительных экспериментов по интерполяции реальных сигналов.

Список литературы

1. Чуи Ч. Введение в вейвлеты : учебное пособие для студентов вузов / Ч. Чуи ; пер. с англ. Я. М. Жилейкина. – Москва : Мир, 2001. – 412 с.
2. Maz'ya V. Approximate approximations / V. Maz'ya, G. Schmidt. – AMS Mathematical Surveys and Monographs, 2007. – V. 141. – 350 p.
3. Журавлев М. В. О вычислительных особенностях интерполяции с помощью целочисленных сдвигов гауссовых функций / М. В. Журавлев, Л. А. Минин, С. М. Ситник // Научные ведомости Белгородского государственного университета. Серия: Физика. Математика. – 2009. – № 13(68). – вып. 17/2. – С. 89–99.
4. Минин Л. А. Поведение коэффициентов узловых функций, построенных из равномерных сдвигов функций Лоренца и функций Гаусса / Л. А. Минин, С. М. Ситник, С. Н. Ушаков // Научные ведомости БелГУ. Серия: Физика. Математика. – 2014. – № 12(183), вып. 35. – С. 214–217.
5. Unser M. On the Asymptotic Convergence of B-Spline Wavelets to Gabor Functions / M. Unser, A. Aldroubi, M. Eden // IEEE Transactions on Information Theory. – 1992. – V. 38, № 2. – P. 864–872.

Глубокое обучение в задаче распознавания онкологии кожи

В. А. Мелихова

Студент бакалавриата

А. Ю. Иванков

Доцент

Введение

Рак кожи – это обобщенное название большого количества разновидностей злокачественных опухолей. Каждая опухоль имеет свои специфические биологические особенности: клиническое проявление, тканевую структуру, метастазирование и т.д. Существует три основных типа рака кожи: базальноклеточный рак, плоскоклеточный рак и злокачественная меланома. Среди всех раков кожи злокачественная меланома кожи является наиболее опасным видом рака, так как в подавляющем большинстве случаев приводит к гибели пациентов, особенно при обнаружении патологии на поздней стадии. По данным Всемирной организации здравоохранения (ВОЗ) частота возникновения этого заболевания неуклонно растет из года в год. Высокий уровень смертности при заболевании меланомой, как правило, объясняется поздней диагностикой заболевания. В то же время, при обнаружении на ранних стадиях, хирургическое лечение дает хорошие прогностические результаты и может обеспечить практически 100 % -ю выживаемость. В борьбе со смертностью от меланомы на первый план выходит вторичная профилактика или превентивная медицина. Своевременные осмотры, терапия и контроль распространения рецидивов. Однако, принимая во внимание низкую (около 50%) диагностическую точность пигментных опухолей на ранней стадии врачами общей практики, требуется создание новых методов контроля опухолей. Трудности диагностики меланом врачами общей практики связаны со сложностью в интерпретации клинических признаков опухоли и невозможностью отличить меланому от доброкачественных пигментных образований (таких как невус) на ранней стадии развития. Также при подозрении на наличие меланомы медицинский персонал лишён возможности использования инвазивных методов исследования, таких как биопсия с гистологическим или цитологическим исследованием, в связи с повышенным риском прогрессирования поражений. Стремительное

развитие в последние годы методов искусственного интеллекта (ИИ) и, в частности, глубокое обучение, привели к тому, что появилось целое направление, связанное с разработкой и использованием интеллектуальных систем в медицине, которая, как оказалось, является одной из наиболее востребованных и перспективных областей применения. Целью исследования является применение глубокого обучения для распознавания онкологии кожи.

На начальном этапе происходит предварительная обработка изображений кожных образований. Выполняется обрезка областей затенений для снимков, полученных с микроскопа.

Вторая задача заключается в проведении сегментации с использованием обученной свёрточной нейронной сети (СНС) для формирования бинарной маски.

Далее происходит классификация также с использованием обученной свёрточной нейронной сети.

При решении указанных задач использовалась среда Matlab с использованием фреймворка Deep Learning Toolbox.

1. Предварительная обработка

На многих снимках датасета присутствуют затенения от рамки микроскопа, а также от других факторов. Затенения будут вносить искажения в работу СНС, так как они будут лучше реагировать на перепад яркости между тенью и светом, а не на само образование на коже. Поэтому первым шагом в предварительной обработке являлось определение наличия области затенения и, при наличии таковой, её выделение.

Для изображения I размера $w \times h$, приведенному интервалу яркости от 0 до 1, проверка наличия области затенения выполнялась путём сравнения значений яркости угловых пикселей с фиксированным порогом t_{gray} , для которого, в результате проведенных испытаний, было выбрано значение $t_{\text{gray}} = 0.25$. Краевые пиксели, для которых выполняется данное условие, формируют бинарную маску области затенения:

$$M_{ij} = \begin{cases} 1, & I_{ij} \leq t_{\text{gray}}, i = 1, w; j = 1, h, \\ 0, & I_{ij} > t_{\text{gray}}, i = 1, w; j = 1, h \text{ или } i \neq 1, w; j \neq 1, h. \end{cases} \quad (1)$$

Наличие хотя бы одного ненулевого пикселя в маске M сигнализирует о наличии области затенения.

Для правильного выделения области затенения важно избегать использования алгоритмов бинаризации с глобальным или локальным

порогом. Это связано с тем, что существует множество новообразований, которые имеют темное покрытие, и если алгоритм бинаризации будет основываться только на яркости пикселя, то темные объекты будут неправильно отнесены к области затенения и исключены из дальнейшей обработки.

Вместо этого необходимо использовать алгоритм, который учитывает положение и взаимосвязь соседних пикселей, чтобы "привязать" выделенную область к граничным пикселям, которые попали в маску. Это позволит более точно определить границы затененной области и избежать ошибочного исключения темных объектов.

Поэтому был выбран алгоритм сегментации GraphCat (и его модификация для цветных изображений GrubCat), в котором сегментация изображения основана на поиске минимального разреза графа. Алгоритм представляет изображение в виде ненаправленного графа, где каждый пиксель соответствует вершине. Дополнительно есть две терминальные вершины - исток (S), представляющий объект, и сток (T), представляющий фон или область затенения. Ребра графа разделяются на два типа: p -ребра, соединяющие соседние пиксели, и t -ребра, соединяющие каждый пиксель с терминальными вершинами. Веса p -ребер отражают близость соседних пикселей в терминах вероятности принадлежности объекту или фону. Веса t -ребер определяются вероятностями принадлежности пикселей объекту или области затенения. Для запуска алгоритма необходимо задать начальные области объекта и фона. Маска используется для определения начальной области затенения, а небольшой прямоугольной областью в центре изображения задается начальное расположение объекта. Алгоритм минимизирует функционал (математическое выражение, которое описывает, какой результат мы стремимся достичь при выполнении) для сегментации. Это позволяет учесть взаимосвязь соседних пикселей.

Далее следует второй этап предварительной обработки. Для сохранения максимального количества информации об объекте из обнаруженной области затенения требовалось извлечь прямоугольную область наибольшей площади. Для этого был разработан простой переборный алгоритм, который вычисляет параметры вписанного прямоугольника максимальной площади.

Для уменьшения вычислительной сложности изображение и маска выделенной области были приведены к размеру $L \times L$. На уменьшенной маске вычисляются координаты центра масс (x_0, y_0) пикселей объекта. Затем выполняется перебор длин сторон прямоугольника,

центрированного в точке (x_0, y_0) , в пределах от l_{\min} до l_{\max} , где $l_{\max} = \min(l_0, L - l_0)$, $l_{\min} = l_{\max} / d_l$ l_0 – соответствующая координата центра масс. В процессе перебора фиксируются комбинации длин сторон, которые не пересекаются с маской области затенения. По завершении перебора формируется бинарная маска прямоугольника, стороны которого определяются комбинацией, имеющей наибольшее произведение. Затем данная маска и координаты ее угловых точек возвращаются к исходному масштабу.

2. Семантическая сегментация

Для обучения была сформирована обучающая выборка, состоявшая из пар изображений и соответствующих бинарных масок разметки. Все изображения были обработаны алгоритмом выделения рабочей области, описанным в 1 этапе и приведены к размеру 256×256 . Таким образом был сформирован датасет, включающий 10015 пар цветных и бинарных изображений, 90% которых использовались для обучения и 10% для валидации.

Для реализации сегментации использовалась сеть основана сеть с архитектурой DeepLab v3 с использованием свёрточных слоёв от обученных СНС ResNet. Одной из проблем, которую решает ResNet, является сохранение точности при увеличении глубины сети, в то время как другие СНС могут показывать увеличение точности с увеличением глубины сети, а затем быстрое ухудшение. ResNet используют «остаточную» структуру обучения, где осуществляется сложение входного сигнала с выходным сигналом свёрточных слоёв. Таким образом, ResNet обеспечивает легкую оптимизацию и позволяет повысить точность путем увеличения глубины сети, что сложнее с другими архитектурами.

Для обучения был использован оптимизатор sgdм на 5 эпохах с порогом ограничения градиента 0.01, минибатчем размера 4 и параметром L2-регуляризации 0.0001. Начальная скорость обучения составляла 0.1 и снижалась в 10 раз в начале каждой последующей эпохи. Для оценки результатов обучения использовалось значение меры сходства Жаккара, усредненная по 1000 изображений, случайным образом отобранных из обучающей выборки. Данный показатель равен отношению чисел ненулевых пикселей бинарных изображений, полученных в результате пересечения и объединения эталонной бинарной маски из обучающей выборки и маски, сформированной СНС. Для дообученной нейронной сети ResNet18 был получен усредненный показатель Жаккара 0.8872, для ResNet50 0.9153, соответственно.

3. Классификация

На последнем этапе выполнялся перенос обучения с использованием предобученной сверточной нейронной сети. Лучшие результаты показала сеть DarkNet19. Перенос обучения происходил путём замены последнего слоя свертки, отвечающего за формирование вектора признаков, полносвязным слоем с 4 нейронами (по числу классов).

Первым шагом к началу классификации являлось выделение рабочей области, затем использовалась СНС для сегментации, чтобы выделить фрагмент, содержащий область новообразования. Но была обнаружена проблема в недостаточном количестве примеров в классах «плоскоклеточный рак» и «базиломы». Недостаток изображений приведёт к переобучению алгоритма и не позволит достичь удовлетворительной точности (данное предположение было подтверждено в ходе испытаний). Поэтому была применена аугментация классов. Для каждого из выше перечисленных классов в начале алгоритма было определено число дополнительных примеров:

$$N_{aug_i} = \text{round}((N - N_i) / N_i), i = \overline{1,4},$$

которые требуется сгенерировать с каждого изображения, чтобы результирующий объем был порядка N (желаемое число образцов). Таким образом, для тестовой выборки общее число изображений составляет 674, для обучающей выборки 4235.

При подготовке тестирующей выборке датасета было замечено, что изображения отсняты в различных условиях, о чем свидетельствует большой разброс в качестве материалов, размерах и разрешении изображений, наличие на некоторых образцах засветки и существенный разброс цветового баланса. Поэтому, по результатам первых испытаний, было принято решение включить в процедуру подготовки датасетов коррекцию цветового баланса, которая выполнялась с использованием алгоритма «серый мир». В результате такой коррекции средняя яркость в цветовых каналах приводится к единому значению, а изображение становится более «серым». Это с одной стороны ослабило возможность классификации с опорой на цветовые характеристики, но с другой стороны позволило сбалансировать вес цветовых признаков с другими используемыми признаками.

В результате классификации была получена матрица ошибок, показывающая точность выполнения классификации. По главной диагонали видно, что точность определения базилиомы=81,0%, меланомы=88,9%, невусов=88,3%, плоскоклеточный рак=78,6%. Таким образом, итоговая точность классификации составила 84.1%.



Рис. 1. Примеры тестирующих изображений из всех 4 классов (по строкам)

Точность=0.84125

True Class	базилома	81.0%	7.6%	4.8%	6.7%
	меланома	1.2%	88.9%	8.0%	1.9%
	невусы		11.1%	88.3%	0.6%
	плоск. рак	7.1%	11.4%	2.9%	78.6%
		базилома	меланома	невусы	плоск. рак
		Predicted Class			

Рис. 2. Матрица путаницы

Закключение

В ходе исследования было показано, что глубокие нейронные сети способны достичь высоких показателей точности при обнаружении различных типов кожных раков, таких как базалиома, плоскоклеточный рак и меланома. Эта точность в данной работе составляет примерно 84%. Это открывает перспективы для применения глубокого обучения в медицине.

Однако, несмотря на неплохие результаты, представленные в данной статье, глубокое обучение всё ещё сталкивается с некоторыми

проблемами. Во-первых, это риск переобучения. Проблема возникает, когда модель слишком хорошо запоминает тренировочные данные и не обобщает свои знания на новые, ранее не виденные данные. Также недостаток данных приводит к снижению у свёрточных нейронных сетей обобщающей способности на разных популяциях и этнических группах.

Список литературы

1. Диагностика меланомы с использованием нейронной сети Inceptionv3 [Электронный ресурс] – Режим доступа: <https://www.graphicon.ru/html/2018/papers/266-269.pdf>
2. Гриневич, В. Н. Классификация опухолей кожи ВОЗ-2018 Кирсанов [Электронный ресурс] – Режим доступа: <https://www.oncopathology.ru>
3. Глубокое обучение в Matlab [Электронный ресурс] – Режим доступа: <https://docs.exponenta.ru>

Моделирование процессов арбитража при торговле цифровыми активами

А. В. Москаленко

Студент бакалавриата

А. А. Громковский

Доцент

Введение

Торговля цифровыми активами является одним из основных способов заработка в сети Интернет. В 2021 году по версии сайта *tadviser* объём рынка цифровых активов приблизился к значению в 3 триллиона долларов США. В этом же году данный рынок привлёк инвестиций на сумму более чем 30 миллиардов долларов [1].

Цифровые активы получили своё развитие в цифровой экономике в 2011 году, после публикации статьи «Crypto Currency» Энди Гринберга в журнале *Forbes* [2]. В 2017 году начала набирать популярность добыча цифровых активов (так называемый майнинг). Данный процесс вызвал подорожание видеокарт, ввиду активного их использования в майнинг-фермах. На конец 2022 года многие компании и предприниматели инвестируют в цифровые валюты, или даже создают свои активы, как например, Павел Дуров, основатель социальной сети «ВКонтакте» и мессенджера Telegram, благодаря которому была создана сеть TON (Telegram Open Network) для реализации операций обмена цифровыми активами посредством одноимённой цифровой валюты.

Государство признаёт экономическую значимость цифровых активов: регулирование использования цифровых валют и налогообложение прибыли, полученной в результате торговли цифровыми активами в Российской Федерации осуществляется на основе 259-ФЗ «О цифровых финансовых активах, цифровой валюте и о внесении изменений в отдельные законодательные акты Российской Федерации», принятого в 2020 году [3].

Номинальная стоимость цифровых активов регулируется людьми, являющихся держателями этих активов. Это означает, что у данного рынка отсутствует какой-либо централизованный регулятор. По этой причине, в зависимости от доверия пользователей к тому или иному

активу, его цена может достаточно сильно варьироваться, относительно цен на существующие мировые валюты. На платформах различных операторов цены на одни и те же цифровые активы могут отличаться. Совокупность данных факторов обуславливает возможность реализации процесса арбитража цифровых валют.

Моделирование процесса арбитража при торговле цифровыми активами предполагает реализацию следующих этапов:

- описание сущности работы арбитража цифровых активов;
- построение модели процесса арбитража цифровых активов;
- реализация архитектуры информационной системы для автоматизации процесса арбитража торговли цифровыми активами;
- обобщение результатов моделирования с целью выявления перспектив автоматизации процесса арбитража цифровых активов.

Моделирование процесса арбитража цифровых валют

При построении модели процесса арбитража цифровых валют были использованы основные понятия, приведённые в табл. 1:

Таблица 1

Основные понятия

Термин	Определение
Арбитраж цифровых валют	Это механизм заработка на основе разницы курсов цифровых валют у различных операторов
Stable Coin	Цифровая валюта, стоимость которой привязана к реальной валюте какой-либо страны. При решении задачи моделирования была использована валюта USDT, стоимость которой эквивалентна 1 доллару США
Пара	Числовое отношение стоимости двух цифровых валют, определяющее курс валют друг относительно друга
Круг	Цикл приобретения цифровой валюты за stable coin, перевод валюты на другую платформу, совершение операций обмена на иные цифровые валюты и конечный обмен на изначальный stable coin
Вилка	Набор пар, которые будут использоваться в круге
Длина вилки	Количество пар, которые составляют набор вилки

Для выявления сущности процесса арбитража, рассмотрим упрощённый пример выполнения круга. Апостериори известно, что данный круг генерирует прибыль. Пары для цифровых валют на двух биржах показаны в табл. 2 и табл. 3. Приведённые данные максимально приближены к реальной ситуации по состоянию на август 2022 года.

Таблица 2

Котировки цифровых валют на бирже 1

Биржа 1	ETH	BTC	LTC	BNB	NEO	USDT
ETH	-	0,0718 41	-	-	-	1665,73
BTC	13,919 62807	-	377,21 61449	72,458 5175	2016,1 29032	23187,4 2
LTC	-	0,0026 51	-	-	-	-
BNB	-	0,0138 01	-	-	-	-
NEO	-	0,0004 96	-	-	-	-
USDT	0,0006 003373 896	0,0000 431268 3343	-	-	-	-

Таблица 3

Котировки цифровых валют на бирже 2

Биржа 2	ETH	BTC	LTC	BNB	NEO	USDT
ETH	-	0,0718 430000				1665,91 00000
BTC	13,919 24057	-	376,93 17753	72,453 26764	2008,0 32129	23187,4 5
LTC		0,0026 53	-			
BNB		0,0138 02		-		
NEO		0,0004 98			-	
USDT	0,0006 002725 237	0,0000 431267 7763				-

Из табл. 2 и табл. 3 видно, что курсы цифровых валют на двух биржах отличаются, причём отличия достаточно небольшие. Это является нормой для рынка цифровых активов и определяет особенность решения поставленной задачи.

Отношения цифровых валют в виде пар можно представить в виде ориентированного графа. Данный граф представлен на рис. 1. Рёбра красного цвета показывают отношение активов в паре для первой биржи. Рёбра синего цвета показывают отношение активов в паре для второй биржи. Числа рядом с рёбрами показывают, сколько единиц актива, указанного в начальной вершине, можно приобрести за одну единицу актива, указанного в конечной вершине.

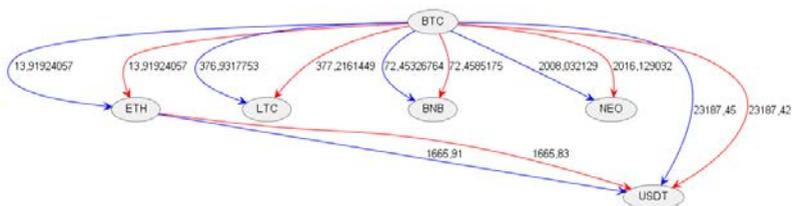


Рис. 1. Граф котировок цифровых валют

Граф, приведённый на рис. 1 является упрощённым. Для решения реальных задач арбитража цифровых активов предполагается использовать полный граф, отражающий все взаимосвязи для котировок цифровых валют. Проведение анализа пар на основе имеющихся данных позволяет определить вилки, которые должны генерировать прибыль. Для рассматриваемого примера генерировать прибыль могут вилки, изображённые на рис. 2:

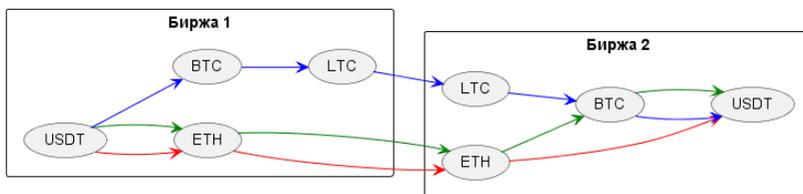


Рис. 2. Вилки, генерирующие прибыль

Реализация операции последовательного приобретения активов в вилках позволяет выполнить круг и извлечь прибыль.

Проведённый анализ демонстрирует действительную возможность получения прибыли при реализации процесса арбитража цифровых активов.

С учётом полученного вывода необходимо разработать модель, показывающую, какие именно процессы должны быть реализованы для одного круга с целью извлечения прибыли.

Модель процесса арбитража торговли цифровыми активами была построена в нотации BPMN для случая неавтоматизированной обработки информации, то есть в формате «As is» и изображена на рис. 3.

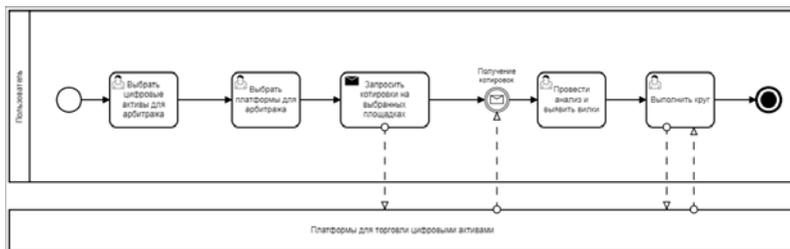


Рис. 3. Модель процесса арбитража цифровых активов в нотации BPMN формата «As Is»

Моделируемый процесс при неавтоматизированном выполнении требует затрат значительного количества времени. Так, на примере котировок из табл. 2 и табл. 3, неподготовленному респонденту понадобилось 3 минуты, чтобы найти все вилки для 6 видов активов на 2 платформах в рамках упрощённой модели. При увеличении количества цифровых валют и количества платформ затраченное время не будет возрастать линейно. Это связано с тем, что вычисление вилок для набора пар – сложный процесс. Он представляет собой задачу поиска самого длинного пути в ориентированном графе. Такие задачи решаются с помощью динамического программирования.

Для реализации процесса арбитража время является ключевым фактором. Это обусловлено быстрым варьированием цен на цифровые активы. Для уменьшения временных затрат необходимо спроектировать модель информационной системы, способной выполнять операции, связанные с анализом котировок и выполнением кругов в автоматическом режиме. Модель бизнес-процесса арбитража при торговле цифровыми активами в форме процесса «To Be», приведена на рис. 4.

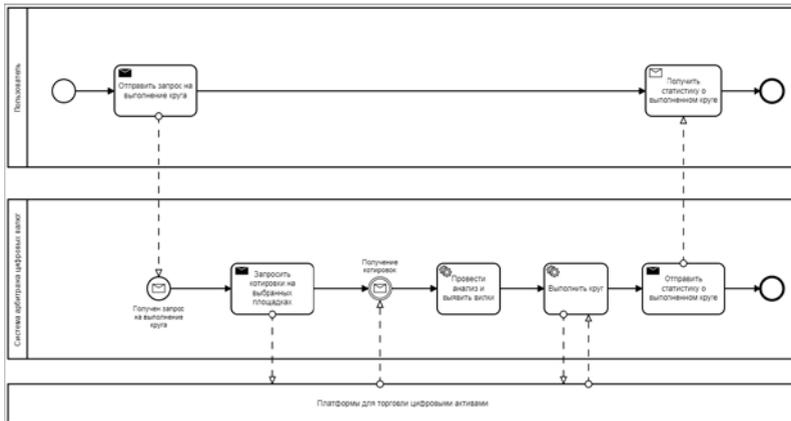


Рис. 4. Модель процесса арбитража цифровых активов в нотации BPMN формата «To Ve»

На рис. 5 представлена архитектура информационной системы, для автоматизированного сбора и обработки информации по котировкам цифровых активов, а также их приобретения.

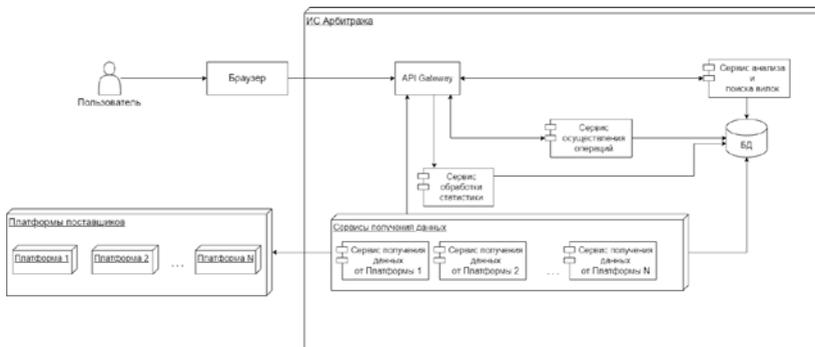


Рис. 5. Архитектура Информационной Системы для осуществления арбитража цифровых активов

В табл. 4 представлено функциональное описание каждого компонента системы.

*Описание компонентов информационной системы арбитража
цифровых валют*

Компонент	Описание
API Gateway	Программный интерфейс, реализующий взаимодействие отдельных компонентов информационной системы арбитража
Сервис операций	Сервис, предназначенный для автоматического проведения всех необходимых операций с целью выполнения круга и извлечения прибыли
Сервис анализа и поиска вилок	Сервис, предназначенный для анализа собранных данных из разных платформ, поиска вилок для полученного набора информации и оповещения сервиса операций о найденных вилках
Сервис обработки статистики	Сервис, предназначенный для предоставления пользователю статистической информации о найденных вилках и фиксации полученной прибыли
Сервисы получения данных	Совокупность сервисов, каждый из которых предназначен для фиксации котировок одной конкретной платформы и сохранения этой информации в базу данных

Для выявления вилок, на основе которых реализуется круг, была построена модель данных. Она выполнена на основе информации, которую можно получить с помощью выполнения запроса к API практически любой платформы торговли цифровыми активами. Модель данных представлена в виде ER-диаграммы на рис.6.

Котировки валют можно представить в виде ориентированного взвешенного графа. Особенностью данного графа является наличие между двумя вершинами графа нескольких связей. Количество связей определяется числом платформ торговли цифровыми активами. На рис. 7 изображены графы для описания моделей арбитража с 6 цифровыми активами на 2 и 3 платформах.

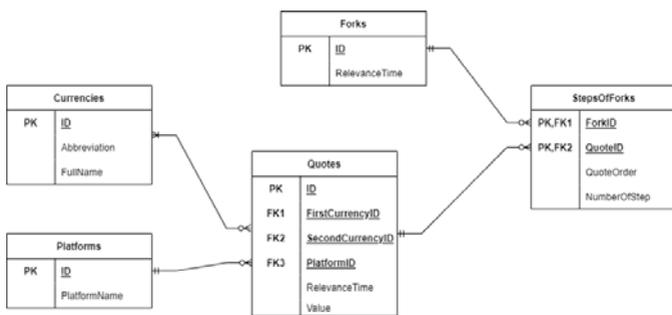
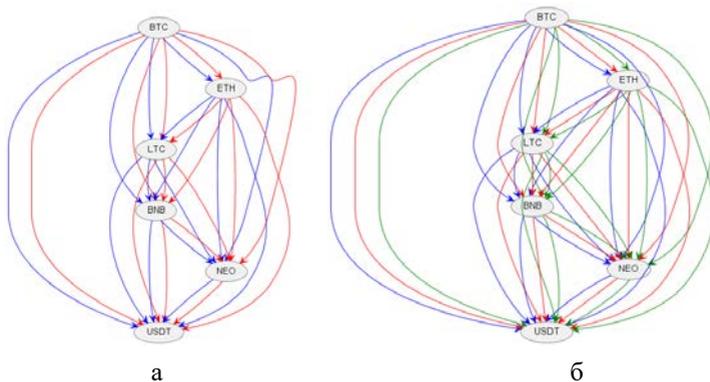


Рис. 6. ER-диаграмма, описывающая модель данных ИС арбитража цифровых валют



а – граф, описывающий модель арбитража с 6 цифровыми активами на 2 платформах, б – граф, описывающий модель арбитража с 6 цифровыми активами на 3 платформах

Рис. 7. Графы, описывающие модели арбитража

На основании анализа графов на рис. 7 видно, что подключение большего количества платформ будет увеличивать число потенциальных вилок, которые, по сути своей, будут являться вариантами обхода графа. При этом количество необходимых вычислительных операций будет возрастать.

На основе операционных данных был построен алгоритм, с помощью которого можно находить вилки, генерирующие прибыль. При реализации последовательности обработки пар можно определять количество цифровых валют, используемых в одной вилке, не включая stable coin. Этот параметр задаёт глубину обхода графа. После

нахождения потенциальных вилок, необходимо произвести расчёт значения эквивалента относительно stable coin по потенциальным вилкам. Если в случае выполнения круга в рамках расчёта начальный баланс увеличивается, вилка признаётся релевантной и сохраняется. Такую вилку можно использовать для генерации прибыли. Графическая форма представления алгоритма определения потенциальных вилок для данных котировок приведена на рис. 8.

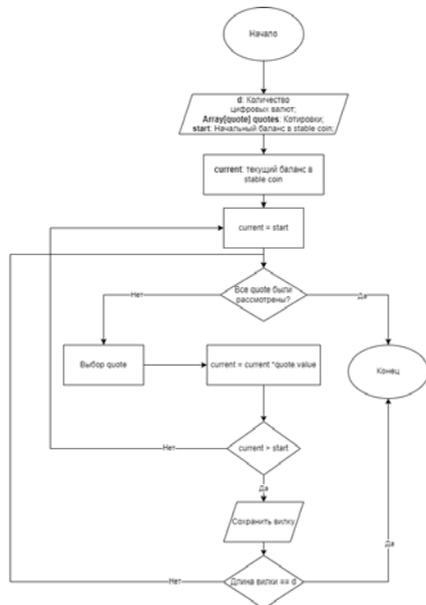


Рис. 8. Алгоритм выявления и сохранения вилок

Жизненный цикл одной вилки является крайне непродолжительным. Следует учитывать, что платформа взимает комиссию за каждую операцию. Поэтому, для уменьшения временных затрат и снижения комиссии необходимо стремиться к уменьшению длины вилки.

На основе опыта проведения операций с цифровыми активами можно прийти к выводу, что эмпирически приемлемая длина вилки должна составлять от 2 до 3 пар. Это обусловлено возрастанием количества необходимых вычислений и увеличением размера комиссии платформ торговли цифровыми активами.

Заключение

Исследование процесса арбитража цифровых активов представляет собой актуальную научно-практическую задачу. В работе предложено решение данной задачи на основе разработки формальных моделей для ручного и автоматического вариантов выполнения. Бизнес-процесс арбитража цифровых активов был формализован посредством моделей в нотации BPMN. Непосредственное описание процесса реализовано бизнес-моделью в форме «As Is». На основе формы «As Is» была построена модель в форме «To Be». Построенная модель послужила основой для проектирования архитектуры информационной системы, модели данных, и построения алгоритма нахождения прибыльных вилок.

Дальнейшее развитие задачи исследования процесса арбитража цифровых активов может быть направлено на построение оптимальных алгоритмов поиска вилок, нахождение коммерческих каналов реализации построенных алгоритмов, их применение на практике.

Список литературы

1. Криптовалюты [Электронный ресурс] : Tadviser – Интернет-портал – Режим доступа: <https://www.tadviser.ru/a/611202>
2. Гринберг, Э. Crypto Currency [Электронный ресурс] : Forbes, 2011. – Режим доступа : <https://www.forbes.com/forbes/2011/0509/technology-psylocybin-bitcoins-gavin-andresen-crypto-currency>
3. Федеральный закон от 31.07.2020 г. № 259-ФЗ «О цифровых финансовых активах, цифровой валюте и о внесении изменений в отдельные законодательные акты Российской Федерации» // СПС «КонсультантПлюс».
4. Вигерс, К. Разработка требований к программному обеспечению / К. Вигерс, Д. Битти. – СПб. : БХВ-Петербург, 2021. – 736 с.
5. Репин В. Бизнес процессы. Моделирование, внедрение, управление / В. Репин. – М. : Манн, Иванов и Фербер, 2012. – 470 с.
6. Ковалёв, С. Настольная книга аналитика. Практическое руководство по проектированию бизнеспроцессов и организационной структуры / С. Ковалев, В. Ковалев. – М. : 1СПаблишинг, 2021. – 360 с.

Кодогенерация многопоточных классов с помощью Annotation processor

А. А. Мясоедов

Студент бакалавриата

Н. К. Самойлов

Старший преподаватель

Введение

В современном мире информационные технологии приобретают всё более широкий охват и, как следствие, растёт нагрузка на информационные сервисы. Чтобы справляться с большой нагрузкой и максимально повысить пропускную способность, подавляющее большинство современных сервисов работают в многопоточном режиме или используют многопоточные алгоритмы. Многопоточное программирование является более сложным и имеет высокий порог вхождения. Разработчик должен обладать большим багажом знаний и навыков, а также иметь релевантный опыт, чтобы его многопоточный код работал корректно, быстро и приносил пользу, а не ухудшал показатели производительности или отказоустойчивости сервисов. Для приобретения таких знаний и навыков необходимо хорошо изучать теорию параллельных программ и алгоритмов и также решать много практических задач, но даже после этого, работая с многопоточным кодом, вероятность совершить ошибку выше, чем при работе с кодом, в котором не требуется явное участие программиста в управлении многопоточностью. Чтобы облегчить написание программ, использующих параллельные алгоритмы, специально создаются библиотеки и фреймворки, предоставляющие нужную функциональность разработчикам и берущие реализацию часто встречаемых шаблонов и алгоритмов на себя. Такие фреймворки дают целый ряд плюсов программистам, использующим их: они уменьшают вероятность совершения ошибки разработчиком, ускоряют написание кода.

1. Annotation processor

Annotation processing[1] – технология позволяющая обрабатывать аннотации[2] в исходном коде Java для последующей кодогенерации или

анализа на этапе компиляции. Впервые данная технология появилась в Java 5. С помощью annotation processor'ов можно генерировать не только исполняемый код, но и другие типы файлов, которые могут быть связаны с аннотациями в исходном коде, например файлы документации, метаописания, файлы ресурсов приложения. Другими словами, annotation processing даёт возможность запустить какой-либо java код на этапе компиляции основного приложения. Внутрь классов процессоров аннотаций можно положить любой код, написанный на Java. Этот код может создавать новые директории, валидировать и проверять какие-либо внешние ресурсы перед запуском основного приложения. Важной особенностью технологии annotation processing является то, что она не позволяет модифицировать уже написанные файлы с расширением .java. С помощью процессора аннотаций возможно только создавать новые файлы.

При необходимости модифицировать уже созданные java файлы, существует библиотека Spoon. Эта библиотека также написана на java и её процессоры обрабатывают во время компиляции. Она позволяет изменять существующие java файлы с помощью модификации синтаксического дерева, даёт возможность работать как с токенами или узлами дерева, так и с синтаксическими конструкциями в виде строк текста.

Ещё одной особенностью процессора аннотаций является то, что один и тот же процессор аннотаций может вызываться и обрабатывать несколько раз пока все исходные файлы не будут обработаны им. Данная схема представлена на рис. 1.

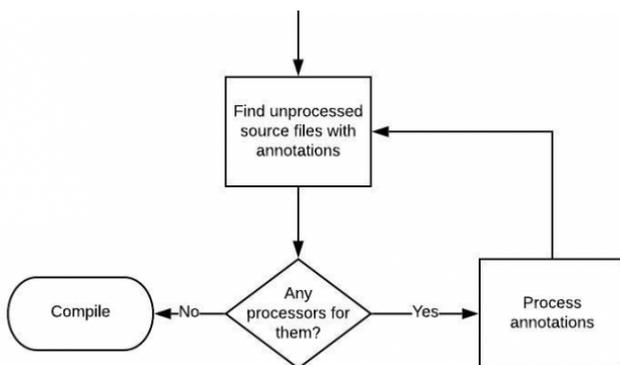


Рис. 1. Схема работы процессора аннотаций

Это расширяет возможности для генерации кода и позволяет с помощью annotation processor'a генерировать классы, которые будут

обработаны этим же процессором на следующем круге на этапе компиляции, или новый сгенерированный файл может обработать другой процессор, подключенный к проекту. Также заниматься препроцессингом в одном проекте могут несколько процессоров аннотаций, это позволяет изолировать каждый процессор для отдельной аннотации, строить сложные цепочки обработки, когда один процессор создаёт java файл, который будет обработан следующим процессором. В этой цепочке может быть сколько угодно элементов.

Для создания процессора аннотаций в проекте необходимо создать аннотации, которые должны будут обрабатываться препроцессорами, так как `annotation processor` обрабатывает 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`. Данный пример представлен на рис. 2.

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

Рис. 2. Переопределение метода `process`

Внутри метода `process` происходит вся логика обработки аннотаций. Необходимо пометить и сам класс процессора с помощью аннотации `@SupportedAnnotationTypes`, параметром которой является массив строк, где каждая строка полное имя аннотации, обрабатываемой этим процессором. Данный пример представлен на рис. 3.

```
@SupportedAnnotationTypes({"ru.cs.processor.CustomA
nnotation"})
public class CustomProcessor extends
AbstractProcessor {...}
```

Рис. 3. Использование аннотации `@SupportedAnnotationTypes`

Также необходимо зарегистрировать созданный `annotation processor`. Для этого нужно создать файл `javax.annotation.processing.Processor` в директории `resources/META-INF/services`. В файле необходимо указать полные имена классов созданных препроцессоров, каждое имя на новой строке. Данный пример представлен на рис. 4.

```
ru.cs.processor.CustomProcessor1
ru.cs.processor.CustomProcessor2
ru.cs.processor.CustomProcessor3
```

Рис. 4. Пример регистрации процессора аннотаций

Важной особенностью является то, что на этапе, когда работают процессоры аннотаций, классы проекта, для которого они применены, ещё не скомпилированы. Это означает, что внутри метода `process` нельзя получить объект `Class<OurProcessedClass>`, где `OurProcessedClass` – класс, в котором мы обрабатываем аннотацию. То есть нельзя будет использовать технологию Reflection API для извлечения метаданных или изменения внутренностей класса. Данное ограничение можно обойти, загрузив класс, с которым необходимо поработать рефлексией, через загрузчик класса, передав ему имя класса, который хотим обработать рефлексией. Данный пример представлен на рис. 5.

```
public class CustomProcessor extends
AbstractProcessor {

    @Override
    public boolean process(Set<? extends
TypeElement> annotations, RoundEnvironment
roundEnv) {
        var classloader =
this.getClass().getClassLoader();
        try {
            Class<?> clazz =
classloader.loadClass("ru.cs.vsu.CustomClass");
        } catch (ClassNotFoundException e) {
            throw new RuntimeException(e);
        }
        return false;
    }
}
```

Рис. 5. Пример работы с классом через механизм рефлексии

Также можно загрузить класс, имея его содержимое, с помощью `URLClassLoader`'а и наследования `SimpleJavaFileObject`.

Но даже после этого использовать рефлексию можно только для работы с метаданными, нет смысла генерировать что-либо для обрабатываемого класса или модифицировать его с помощью рефлексии, так как инстанс `Jvm`, в которой работают процессоры на

этапе компиляции и инстанс `jvm`, в которой будет работать приложение, не имеют связи, так как классы, загруженные во время процессинга аннотаций, не доживут до старта приложения.

2. Запуск задач по расписанию.

Запуск задач по расписанию является распространенным шаблоном в программировании. Программист, использующий процессор аннотаций и библиотеку, может с помощью аннотации `@Scheduled` пометить метод, который он хочет, чтобы тот выполнялся по расписанию. В параметрах аннотации указываются время первого запуска метода, интервал между запусками метода и стратегия поведения, если предыдущее выполнение ещё не завершено (начать новое исполнение немедленно или отложить задачу в очередь). Процессор аннотаций во время компиляции сгенерирует класс, который сможет зарегистрировать задачу в общем пуле задач, после при запуске программы, разработчик сможет инициализировать общий пул задач и все задачи из «регистрирующих» классов автоматически попадут в этот пул и начнут исполняться по расписанию. Но есть некоторые ограничения на методы, которые возможно запускать по расписанию с помощью данной библиотеки:

- метод должен иметь модификаторы `public` (так как из класса, регистрирующего задачу, метод должен быть доступен для вызова) и `static` (т.к. данная библиотека не имеет контейнера для хранения экземпляров классов и для вызова нестатического метода нужен инстанс класса, при необходимости использования нестатического метода разработчик может вручную зарегистрировать задачу);

- метод не должен принимать аргументы.

3. Синхронизация на данных

Синхронизация помогает сохранить консистентность данных при работе многопоточной программы. С другой стороны, подход с синхронизациями в коде может серьезно снизить пропускную способность. Чтобы снизить сторонние эффекты от синхронизаций, можно делать синхронизацию не на целые куски кода или методы при каждом вызове, а ставить синхронизацию только в тех случаях, когда данные, попавшие на вход алгоритма, уже обрабатываются программой. С помощью аннотации `@DataLock` разработчик может пометить метод, на котором он хочет делать синхронизацию по `uid` (уникальный идентификатор для входящих данных). В качестве параметра в аннотации указывается строка, представляющая собой фрагмент кода, возвращающий `uid` в виде строки из аргументов метода. Процессор аннотаций сгенерирует класс наследник, с переопределённым методом,

помеченным аннотацией `@DataLock` и выполняющим синхронизацию по `uid`. Синхронизация происходит с помощью `ReentrantLock`, объекты блокировок хранятся в парах ключ значение, где ключ – `uid` для входящих данных, значение – объект `ReentrantLock`. Перед вызовом метода сначала ищется блокировка для `uid`, если не находится, то она создаётся и дальше, если объект `ReentrantLock` свободен он блокируется. Если же на нём уже стоит блокировка, то происходит ожидание, пока блокировка не снимется.

4. Межсервисные синхронизаторы

В Java реализовано несколько механизмов синхронизации[3], которые упрощают написание многопоточного кода.[4] Но у этих механизмов есть недостаток – они работают в пределах одного экземпляра запущенной Java программы. В современной разработке сервисы зачастую запускаются не в одном экземпляре и поэтому возможности синхронизаторов иногда ограничены, для решения данной проблемы можно “вынести” какой-нибудь алгоритм синхронизации (например, семафор), тогда появится возможность реализовать какой-либо алгоритм, завязанный на синхронизацию данных между несколькими сервисами. Чтобы хранить данные о такой синхронизации, можно воспользоваться любой базой данных или, даже, просто внешним файлом. Но хорошим решением будет Redis, так как он предназначен для быстрого доступа к данным, у которых не очень большой срок жизни. Для использования межсервисного семафора разработчику необходимо указать данные для подключения к Redis[5] и создать `InterServiceSemaphore` с помощью `InterServiceSemaphoreFactory`. Для создания `InterServiceSemaphoreFactory` понадобится `DataProvider` – это интерфейс предоставляющий контракт для операций: увеличение/уменьшение счётчика в источнике данных (например Redis). Чтобы понимать к какому именно семафору обращаться, у каждого объекта `InterServiceSemaphore` есть уникальный идентификатор, который можно задать при создании. Метод `acquire()` запрашивает доступ к ресурсам и блокирует выполнение, если свободных ресурсов нет (если счётчик в источнике данных = 0) или захватывает ресурс (уменьшает счётчик на 1). Метод `release()` освобождает ресурсы (увеличивает счётчик в источнике данных на 1)

Заключение

В результате работы была разработана библиотека, которая позволяет использовать межсервисные синхронизации, выполнение задач по расписанию и синхронизацию на данных с помощью метаописания в коде.

Сформированы перспективные направления дальнейшего развития проекта:

- Добавление новых видов синхронизации таких как, циклический барьер, счётчик, обменник;
- Добавление визуализации разбиения задачи на подзадачи с помощью графа;
- Добавление новых внешних баз данных для хранения данных синхронизаций.

Список литературы

1. Java Annotation processing [Электронный ресурс] : сайт. – Режим доступа : <https://docs.oracle.com/javase/7/docs/api/javax/annotation/processing/package-summary.html>
2. Java Annotations [Электронный ресурс] : сайт. – Режим доступа : <https://docs.oracle.com/javase/8/docs/api/java/lang/annotation/Annotation.html>
3. Java Synchronizers [Электронный ресурс] : сайт. – Режим доступа : <https://habr.com/ru/post/277669/>
4. Java Concurrency [Электронный ресурс] : сайт. – Режим доступа : <https://docs.oracle.com/javase/8/docs/api/index.html?java/util/concurrent/package-summary.html>
5. Redis Documentation [Электронный ресурс] : сайт. – Режим доступа : <https://redis.io/docs>

Использование цифровых следов в интересах решения задачи профориентации

Э. В. Обухова

Студент бакалавриата

А. В. Швырева

Ассистент

Введение

Современное цифровое пространство в открытом доступе содержит множество личной информации о людях и их интересах. Такая информация является цифровым следом и может помочь составить виртуальный портрет личности. Использование цифровых следов в социальных сетях для обучения моделей машинного обучения является актуальным предметом исследования для разных задач, одной из которых является выявление профессиональных склонностей человека.

Существующие тесты профориентации используют стереотипные вопросы, из-за чего обладают низким уровнем доверия, а их результаты легко могут быть скомпрометированы. Использование цифровых следов в сети Интернет для анализа профессиональных склонностей личности позволит обеспечить большую надежность и объективность полученных данных. Поэтому целью данной работы является подготовка набора данных, содержащих цифровые следы из открытых источников студентов и выпускников Воронежского государственного университета (ВГУ), а также создание и обучение модели машинного обучения для классификации профилей пользователя к «техническим» или «гуманитарным» факультетам.

1. Подготовка набора данных

Важным этапом создания и обучения моделей машинного обучения является подготовка набора данных. В качестве площадки была выбрана социальная сеть ВКонтакте, так как она удовлетворяет следующим критериям:

1. Открытый API для отправки запросов.
2. Широкая аудитория.
3. Возможность указать информацию о полученном образовании в профиле пользователя.

4. Возможность для пользователя оставить цифровые следы.

Использование информации из профилей ВКонтакте не нарушает законов о передаче и использовании персональной информации, поскольку открытый API, предоставляемый ВКонтакте для разработчиков, не позволяет получить доступ к информации из закрытых профилей, а информация, публикуемая в открытых профилях, не является приватной. Личный идентификатор пользователя не используется, что обеспечивает анонимность при сборе данных.

ВКонтакте предоставляет пользователю возможность очень подробно описать свою личность, публиковать посты, фотографии, слушать музыку, вступать в тематические сообщества. Все перечисленные действия могут рассматриваться в качестве цифровых следов, однако для решения поставленной задачи были выбраны списки тематических сообществ (групп по интересам).

С помощью открытого API из профиля пользователя можно получить следующую структуру:

- числовой идентификатор вуза, в котором учился респондент;
- название вуза;
- числовой идентификатор факультета;
- название факультета;
- список из пар: числовой идентификатор группы + название.

В результате сбора [1-3] и первичной обработки данных [4-5], включающей удаление дубликатов, некорректных и неполных записей, было получено 8536 профилей пользователей, гистограмма распределения которых по факультетам представлена на рисунке 1.

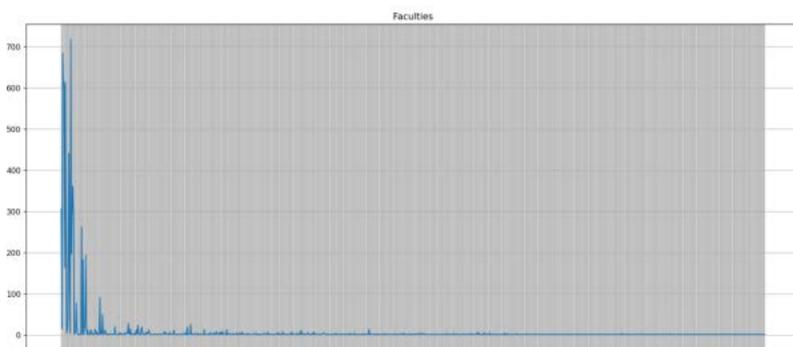


Рис. 1. Распределение респондентов по факультетам

Полученное распределение неравномерно и содержит шумовые одиночные значения, которые также являются выбросами и не подходят

для исследования. Согласно информации о количестве и названии факультетов ВГУ, было получено 6067 результирующих профилей, которые были распределены на два класса: гуманитарные и технические направления в соответствии с таблицей 1.

Таблица 1

Распределение факультетов по двум классам с количеством собранных профилей

Гуманитарное направление		Техническое направление	
Название	Кол-во профилей	Название	Кол-во профилей
журналистика	361	биолого-почвенный	3
исторический	306	географический	182
международных отношений	261	геологический	194
романо-германской филологии	618	компьютерных наук	441
филологический	301	математический	208
философии и психологии	163	прикладной математики	601
экономический	614	фармацевтический	197
юридический	685	физический	719
		химический	204

Фрагмент полученного набора данных представлен на рисунке 2, где в каждой строке содержится уникальный идентификатор факультета, название факультета, а также уникальные идентификаторы групп, участником которых является пользователь.

```

19,37802,'Факультет компьютерных наук',26788786,68433088,66931622,43696984,164967358,86973788,107019848,1133162622,405647146,3044779
20,137185,'Химический факультет',29534144,13642660,82300345,38834885,4325149,98849036,45938293,162260077,90983046,52722250,596484
21,153865,'Математический факультет',181906446,35034571,29573241,29544671,194570815,31976785,43215063,198079074,199521818,2451858
22,8559,'Исторический факультет',144624886,145510087,23408047,1985910,94158280,173165484,159146575,50543881,25752703,29774481,237
23,137185,'Химический факультет',177889617,51909303,202274758,164337287,52969542,75296700,23769931,145465582,148948438,44425853,4
24,2187,'Юридический факультет',137451337,10889156,198760720,160436446,31836774,165168122,183577260,173364997,28905875,130897668,
25,139366,'Экономический факультет',12382740,17711662,26776509,27726642,29197363,41734504,58201910,61605131,
26,27626,'Факультет прикладной математики',29559271,58802848,34173859,46603834,151696853,22798006,23769931,146265615,45184984,786
27,5787,'Физический факультет',18889156,98018205,29559271,207014980,108370837,186208863,116021695,155790705,42296923,48940689,111
28,18314,'Фармацевтический факультет',136962432,23213239,19977823,111526081,29833734,39566948,52598206,109349119,79138567,7695362
29,4150,'Филологический факультет',92042578,31092329,136149656,30464713,61901308,109427560,54169630,10493531,31584605,80630534,1
30,8559,'Исторический факультет',24834832,10889156,53424088,42800749,136962432,152106690,29534144,116739007,109132603,69275730,22
31,25029,'Факультет романо-германской филологии',201025248,88397640,98018205,159286269,10889156,35824409,123851409,185070994,1278
32,18314,'Фармацевтический факультет',3001581,10889156,23339074,23769931,30178525,32432151,35824409,55284725,57684637,58201910,58
33,27626,'Факультет прикладной математики',38691559,202038713,31480508,159146575,167161311,67580761,183136788,199351430,30350935,
34,34391,'Факультет журналистики',80020826,209684121,197703869,46617996,201025971,206520598,100892059,127146702,183293188,1653788
35,8559,'Исторический факультет',147845620,26603427,57721072,142367600,64094774,120939804,43696984,141871181,36648726,25189523,71
36,37802,'Факультет компьютерных наук',25397178,57846937,30598666,32801218,450612348,22751485,35824409,161919913,72495085,30022666
37,5859,'Филологический факультет',148881515,15213818,74599974,130255447,77567648,91466870,35346891,153873438,138871416,20400808
38,212188,'Медико-биологический факультет',460389,12818778,14208296,23626127,23661808,24089576,24253689,25189523,25273562,2531336
39,2187,'Юридический факультет',23339074,28556858,51675727,63133420,81431431,104344721,117731827,125035994,134084423,.....

```

Рис. 2. Фрагмент полученного набора данных

При обучении сети были получены результаты, представленные в таблице 2.

Таблица 2

Точность обученной модели

	precision	recall	F1-score	Support
0	0.79	0.84	0.81	790
1	0.84	0.79	0.81	821
Accuracy			0.81	1611
Macro avg	0.81	0.81	0.81	1611
Weighted avg	0.81	0.81	0.81	1611

Точность модели составляет 81%, что считается хорошим результатом для неоднозначной задачи отнесения личности человека к некоторому классу.

3. Создание программного комплекса

Для использования и тестирования обученной модели был разработан веб-сайт, предоставляющий пользователю возможность протестировать классификатор и получить анализ своего профиля. В качестве средств реализации были выбраны: язык Python, фреймворк Flask, библиотеки pandas, keras, sklearn, matplotlib, numpy, requests.

Реализована клиент-серверная архитектура приложения, с клиентом в виде веб-страниц, отображаемых в браузере, и сервером, развертываемом на компьютере локально (на этапе тестирования) или на любой доступной площадке для развертывания. Для передачи GET- и POST-запросов используется протокол HTTP.

На рисунке 5 представлена диаграмма последовательности обращения пользователя к сервису, а также взаимодействие сервиса с открытым VK.API.

Интерфейс программы представлен на рисунках 7 и 8. На рисунке 7 представлена стартовая страница, с возможностью размещения URL-ссылки на профиль ВКонтакте для проведения тестирования. На рисунке 8 представлен вывод результата классификации, полученный по максимальному значению вероятностей, а также значения вероятностей для каждого класса.

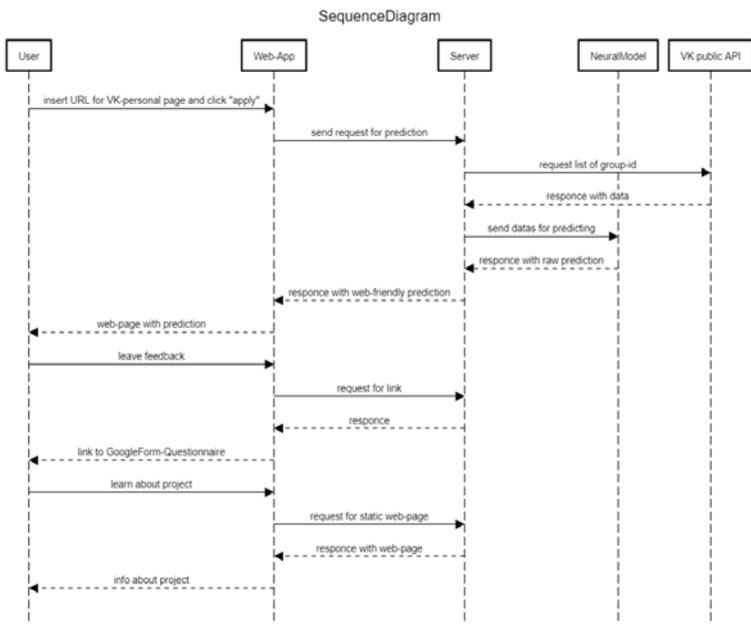


Рис. 5. Диаграмма последовательностей обращения к сервису

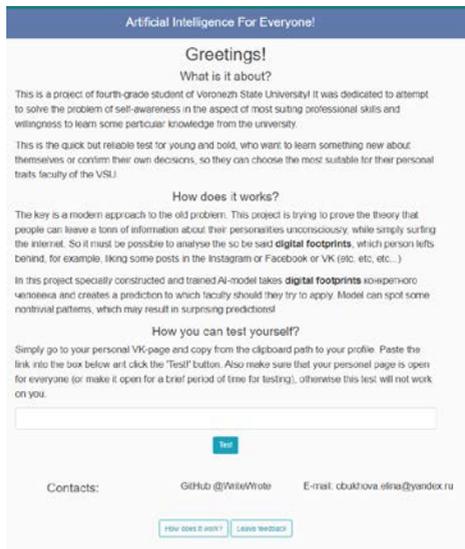


Рис. 6. Интерфейс программного средства - стартовая страница

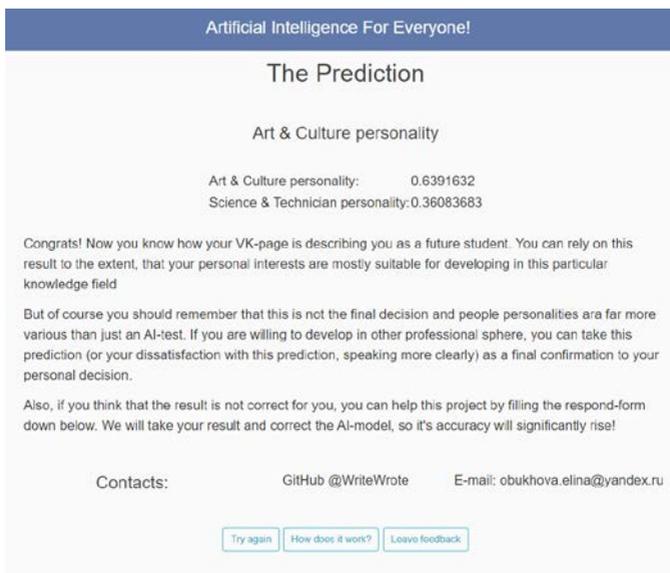


Рис. 7. Интерфейс программного средства - вывод результата

Заключение

В результате работы было разработано программное средство, позволяющее собрать и обработать публично доступную информацию о студентах Воронежского государственного университета в социальной сети ВКонтакте. На основе полученных данных был создан и обучен классификатор, обеспечивающий разделение профилей пользователей с точностью 81%, а также создан ресурс, позволяющий тестировать классификатор.

Для более точной оценки необходимо учитывать семантические особенности цифровых следов (названия групп), и добавить учет других цифровых следов (просмотры и реакции под тематическими постами).

Список литературы

1. Библиотека Requests: HTTP for Humans [Электронный ресурс] : блог сообщества MoscowPython. – Режим доступа: <https://python.ru/post/97/>
2. VK library [Электронный ресурс] : документация сервиса ВКонтакте. – Режим доступа: <https://vk.readthedocs.io/en/latest/index.html>
3. Python Requests Tutorial [Электронный ресурс] : сайт GeeksForGeeks. – Режим доступа: <https://www.geeksforgeeks.org/python-requests-tutorial/>

4. Как подготовить датасет к Machine Learning с PySpark и построить систему потоковой аналитики больших данных на Apache Kafka и ELK: пример прогнозирования CTR [Электронный ресурс] : блог компании «BD School». – Режим доступа: <https://www.bigdataschool.ru/blog/ctr-prediction-with-kafka-spark-elk-case.html>

5. Как собрать датасет за неделю: опыт студентов магистратуры «Наука о данных» [Электронный ресурс] : блог компании «SkillFactory». – Режим доступа: <https://habr.com/ru/company/skillfactory/blog/534682/>

6. О важности датасета и о том, как сделать его лучше [Электронный ресурс] : блог компании «Технологика». – Режим доступа: <https://www.technologika.ru/blog/how-to-create-great-dataset>

Исследование быстродействия ORM библиотек

М. К. Пилеич

Студент бакалавриата

А. И. Чекмарёв

Старший преподаватель

Введение

На сегодняшний день большинство выпускаемых программных продуктов для хранения данных используют реляционные системы управления базами данных. Использование реляционной базы данных для хранения объектно-ориентированных данных приводит к нарушению семантики, в результате чего приходится разрабатывать программное обеспечение, которое должно как поддерживать работу с программными объектами, так и уметь сохранять данные этих объектов в реляционной форме.

Необходимость постоянно преобразовывать данные из одной формы в другую серьёзно снижает скорость работы решения, а также оказывает дополнительную нагрузку на разработчиков и предъявляет повышенные требования к знанию языка структурированных запросов SQL. Простым способом устранения данных проблем является использование ORM библиотек.

Использование ORM библиотек упрощает задачу для разработчиков, так как это освобождает их от написания запросов к базе данных на «чистом» SQL. Это также позволяет сократить время разработки, уменьшить количество строк кода, уменьшить количество ошибок.

Но главный недостаток работы с ORM – снижение скорости работы с базой данных. Слой транзакций, отвечающий за преобразования форм данных, может быть недостаточно эффективным, и при работе с большим количеством данных приложение может работать медленнее, чем при использовании «чистого» SQL, поэтому важно выбрать оптимальную библиотеку. Данная работа посвящена изучению различных ORM средств и определению наилучших вариантов использования для каждого из них.

1. Объектно-реляционное отображение

Распространённым является подход, в котором логика хранения данных отделена от логики использования этих данных. Такое разделение позволяет разработчикам сосредоточиться на своих задачах, а также упрощает и ускоряет процесс разработки и поддержки масштабных систем. С другой стороны, усложняется взаимодействие этих компонент.

В большинстве случаев сегодня для хранения данных, используются реляционные базы данных. ORM упрощает процесс сохранения и извлечение объектов из реляционной базы данных, при этом ORM сама заботится о преобразовании данных между двумя состояниями.

Механизм ORM позволяет оперировать данными через специальный интерфейс, где вместо написания SQL-запросов разработчик вызывает простые методы, а всю остальную работу (преобразование данных в нужный формат) берёт на себя ORM. Механизм работы ORM представлен на рис. 1.

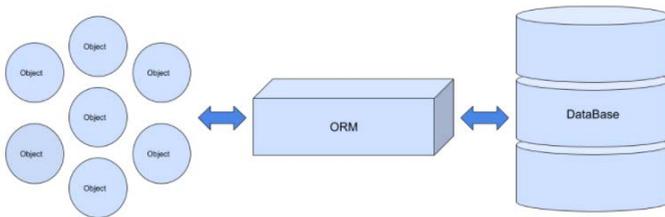


Рис. 1. Основной принцип работы ORM

2. ORM библиотеки

1. Django – это один из самых популярных веб-фреймворков Python со встроенным ORM модулем. Django ORM больше направлен на прямое манипулирование объектами, где он предоставляет прямое сопоставление между таблицами базы данных и классами Python. Django ORM встроен в веб-фреймворк, поэтому его не рекомендуется использовать отдельно от Django Python приложения [1].

2. SQLAlchemy SQL Toolkit и ORM модуль – это набор инструментов для работы с базами данных и Python. SQLAlchemy популярен благодаря своему ORM модулю, используя который, классы приложения могут быть сопоставлены с таблицами базы данных, что

позволяет разрабатывать объектно-ориентированную модель независимо от схемы базы данных [2].

3. Peewee это небольшая простая ORM библиотека, которая поддерживает SQLite, MySQL, PostgreSQL и Cockroachdb. Peewee реализовывает шаблон Active Record. Большой плюс данной библиотеки заключается в том, что она занимает мало места и её легко освоить [3].

4. Hibernate – ORM фреймворк для языка программирования Java, который позволяет связывать Java-классы с таблицами реляционной БД и сопоставлять типы данных Java с типами данных SQL, а также предоставляет средства для автоматического построения запросов и извлечения данных. В Java есть спецификация под названием Java Persistence API (JPA), которая описывает управление объектами в реляционной базе данных. Hibernate является реализацией JPA [4].

5. Фреймворк MyBatis является альтернативой спецификации JPA. Основное отличие заключается в том, как производится маппинг объектов: MyBatis сопоставляет Java методы SQL запросам. Это даёт свободу в написании SQL кода и последующем его выполнении. Таким образом, за формирование запросов полностью ответственен разработчик, и только от него будет зависеть быстрота работы приложения [5].

3. Способы измерения быстродействия кода

Тестирование кода на эффективность и скорость является ключевым аспектом в разработке программного обеспечения. Если код выполняется слишком долго или потребляет слишком много ресурсов памяти или процессора, разработчик может столкнуться с рядом проблем. Машины, на которых выполняется код, могут стать нестабильными, программа может вызывать ошибки, а иногда и потерю данных, поэтому следует проверять код на наличие проблем с производительностью.

Рассмотрим средства, с помощью которых можно определить время выполнения кода на языках программирования Python и Java.

1. Модуль `timeit` позволяет легко измерить время выполнения небольших кусочков кода на языке Python. Помимо простоты в использовании плюсом данного модуля является то, что во время измерения производительности кода он не учитывает фоновые процессы, которые запускаются, когда выполняется программа.

2. Библиотека `line_profiler` (Python) позволяет получить время выполнения каждой строки в файле. Данная функция библиотеки может быть полезна, когда требуется устранить проблемы медленного выполнения функций или вызовов из сторонних файлов. Увидев время, которое уходит на выполнение каждой строки, можно быстро выявить

проблемы вместо того, чтобы вручную замерять время выполнения каждой строки.

3. Модуль `time` в Python содержит различные функции для работы со временем. Самый простой вариант измерения быстродействия кода – посчитать разницу во времени до и после его выполнения. Функция `time.perf_counter()` вернёт время в секундах, прошедшее с 1.01.1970 00:00:00 (UTC), в виде числа с плавающей запятой. Плюс данной функции состоит в том, что она не зависит от времени, установленного на компьютере [6].

4. В Java в классе `System` есть методы `currentTimeMillis()` и `nanoTime()`, с помощью которых можно измерить время, прошедшее между двумя временными отметками. В отличие от `currentTimeMillis()`, метод `nanoTime()` не зависит от системного времени.

Рассмотренные выше библиотеки и функции находятся в открытом доступе.

4. Ленивая и жадная загрузка

Одна из основных проблем разработчиков, когда они создают приложение с ORM — это N+1 запрос в их приложениях. Проблема N+1 запроса — это не эффективный способ обращения к базе данных, когда приложение генерирует запрос на каждый вызов объекта. Эта проблема обычно возникает, когда мы получаем список данных из базы данных без предварительной загрузки (ленивой или жадной).

При использовании жадной загрузки вместе с родительской сущностью будут загружены и все её дочерние сущности, и кроме того, ORM постарается сделать это одним SQL-запросом, чтобы сразу получить все данные.

При ленивой загрузке родительской сущности дочерняя сущность не будет загружена сразу. Вместо неё будет создан прокси-объект, с помощью которого библиотека будет отслеживать обращение к дочерней сущности и при первом обращении загрузит её в память.

5. Проектная часть

Для проведения исследования было принято решение реализовать приложение, на примере которого можно будет оценить скорость выполнения запросов к базе данных с помощью различных ORM библиотек.

В приложении используется система управления реляционными базами данных PostgreSQL. PostgreSQL — это универсальная СУБД, использующая реляционную модель и обладающая широкими возможностями.

Для примера было решено создать базу данных для электронной библиотеки и приложение для работы с ней. Схема базы данных представлена на рис. 2.

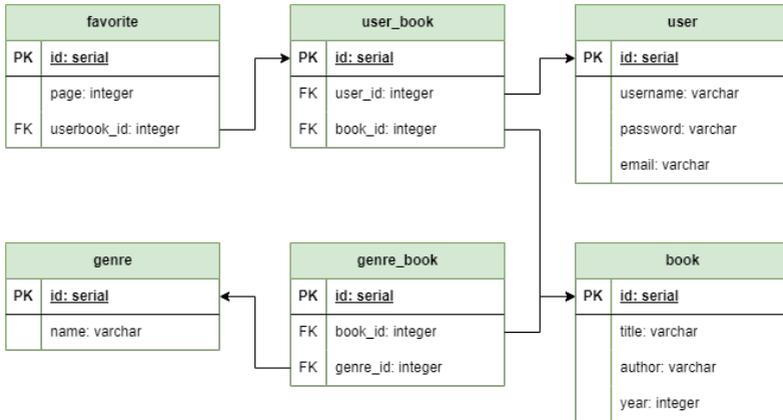


Рис. 2. Схема базы данных

После создания базы данных были созданы пять приложений для каждой из рассматриваемых ORM библиотек: Django ORM, SQLAlchemy, peewee, Hibernate и MyBatis. В каждом приложении было установлено соединение с базой данных, были созданы таблицы и тестовые данные для них с помощью Python библиотеки Faker. Было создано следующее количество записей в таблицах: genre_book и user_book – по 800 000 записей, book – 600 000 записей, favorite – 600 000 записей, user – 300 000, в таблице genre – 50.

Для измерения быстродействия были определены следующие запросы к базе данных:

- запрос на получение записи по id из таблицы (ленивая и жадная загрузка)
- запрос на получение всех записей из таблицы
- запрос на получение записи по сложному условию
- запрос на обновление записи
- запрос на добавление одной записи в таблицу

Для всех запросов для каждой ORM библиотеки были написаны методы для измерения времени их выполнения.

6. Анализ результатов

Результаты, полученные после выполнения запросов, отображены в таблицах, расположенных ниже. В табл.1 указано время выполнения

запросов для рассмотренных Python библиотек, в табл.2 – для Java библиотек. Время в таблицах указано в миллисекундах.

Таблица 1

Результаты выполнения запросов для Python библиотек

	Django	SQLAlchemy	peewee
select(id)	371.8 (lazy) 408.72(eager)	320.9 (lazy) 260.38 (eager)	577.35 (lazy) 426.32 (eager)
select all	7622.45	14526.95	9314.9
select (сложное условие)	3785.01	7000.25	9313.5
update	151.12	28.22	198.42
insert	174.12	20.35	181.56

Таблица 2

Результаты выполнения запросов для Java библиотек

	Hibernate ORM	MyBatis
select(id)	508 (lazy) 384 (eager)	1197
select all	8767	4388
select (сложное условие)	6805	3149
update	684	906
insert	663	33

По результатам, приведённым в таблицах, можно сделать вывод, что запросы get, update, insert быстрее всего выполняются с помощью библиотеки SQLAlchemy, а запросы на выборку всей таблицы и выборки по сложному условию на Python быстрее с помощью Django. На Java для большинства запросов лучшее время показала Hibernate ORM.

Заключение

Данная статья посвящена изучению различных ORM средств и определению наилучших вариантов использования для каждого из них. Был проведён анализ предметной области, определены программные средства для проведения исследования. Разработана и реализована структура для выполнения исследования быстрейшего ORM

библиотек. Произведено сравнение быстродействия ORM библиотек с помощью данных, полученных в ходе исследования.

Список литературы

1. Документация по Django [Электронный ресурс]. – Режим доступа: <https://docs.djangoproject.com/en/4.2/>
2. Документация по SQLAlchemy [Электронный ресурс]. – Режим доступа: <https://docs.sqlalchemy.org/en/20/>
3. Документация по peewee [электронный ресурс]. – Режим доступа: <https://docs.peewee-orm.com/en/latest/>
4. Документация по Hibernate [Электронный ресурс]. – Режим доступа: <https://hibernate.org/orm/documentation/6.2/>
5. Документация по MyBatis [Электронный ресурс]. – Режим доступа: <https://mybatis.org/mybatis-3/index.html>
6. Функции модуля Timer в Python [Электронный ресурс]. – Режим доступа: <https://realpython.com/python-timer/>

Формирование публичного расписания с помощью направленной синхронизации календарей

Е. А. Плаксина

Студент бакалавриата

Я. А. Воронцов

Доцент

Введение

Календарь – важный инструмент в управлении рабочим и личным временем. Часто людям нужно делиться своим календарем, чтобы остальные участники смогли быстро и удобно запланировать встречу без лишних согласований. Классический пример подобного взаимодействия – работа менеджера по продажам через веб-сайт компании: клиенты, желающие назначить онлайн-встречу, должны иметь возможность выбрать удобный для них временной слот.

Для этих целей многие используют облачные календари для планирования (например, Google Calendar, Яндекс Календарь и др.). Однако у всех бесплатных решений в поставке по умолчанию есть ключевой недостаток [1]: для предоставления доступа к календарю неограниченному кругу лиц необходимо его опубликовать, что может нарушить настройки приватности остальных событий и привести к утечке конфиденциальных данных. Помимо этого, при использовании двух или более календарей возникают проблемы синхронизации событий между ними и поддержания всех календарей в актуальном состоянии.

В данной работе рассматривается разработка веб-приложения для формирования публичного расписания, обладающего возможностью синхронизации с несколькими личными календарями с учетом настроек конфиденциальности для событий. Веб-приложение также имеет следующую функциональность:

- Создание событий, которые будут отображаться сразу и в публичном календаре, и в Google Calendar;
- Выбор статуса добавляемого события в Google Calendar;
- Выбор Google Calendar для синхронизации.

1. Анализ аналогов

В ходе работы над данным веб-приложением были проанализированы следующие аналоги:

1. «Youcanbook.me» – это программное обеспечение для бронирования, которое интегрируется с Google или iCloud Calendar [2].

В приложении доступно множество настроек для публичного календаря, удобный и доступный интерфейс, есть возможность для монетизации встречи.

В результате анализа «Youcanbook.me» были выделены следующие недостатки:

- Фиксированная длительность создаваемого события;
- Нет возможности выбора нескольких календарей для синхронизации событий;
- Большинство функций для настройки календаря и событий являются платными.

2. «Calendly» – это сервис для бронирования встреч, который позволяет упростить и автоматизировать процесс назначения встреч с клиентами и коллегами.

Данный аналог в основном рассчитан для команд. В нем поддерживаются следующие функции: расписание, чек-лист, настройки расписания, планирование следующей встречи сразу же после окончания онлайн-конференции. Сервис можно интегрировать с различными другими сервисами, такими как Google Calendar, Office 365, Salesforce, Zoom, Slack и другими.

Среди недостатков «Calendly» можно выделить следующие:

- Сложная настройка календаря;
- Необходимо скачать расширение для календаря;
- Направленная синхронизация один-к-одному;
- Отсутствует выбор календаря для синхронизации.

3. «Singularity App» — это программное обеспечение для управления проектами, разработанное компанией Singularity Technologies. Его основные функции включают планирование проектов, отслеживание временных рамок, анализ статистических данных и коммуникацию с участниками проекта.

«Singularity App» использует искусственный интеллект для улучшения управления проектами, позволяя улучшить эффективность командной работы и сократить время выполнения проектов. Он также может интегрироваться с другими веб-приложениями, такими как Slack и Trello, чтобы обеспечивать более интегрированный опыт пользователей.

Главный недостаток данного аналога – календарь доступен только в платной версии.

2. Формирование требований к веб-приложению

В веб-приложении должно быть разделение пользователей на две роли: владельца календаря и остальных пользователей.

Для владельца календаря приложение должно позволять

- аутентификацию с помощью учетной записи Google;
- выбор одного или несколько календарей для синхронизации;
- просмотр публичного календаря;
- создание событий, которые будут отображаться и в публичном календаре, и в личном календаре;

– выбор приватности добавляемого события («заглушка/placeholder» или «полное»).

– Для остальных пользователей приложение должно обеспечивать возможность

- создания событий, которые будут отображаться как в публичном календаре, так и в связанных с ним личных;
- просмотра публичного календаря с синхронизированными событиями.

3. Средства реализации

Для реализации веб-приложения были выбраны следующие технологии:

- Язык программирования JavaScript;
- Google API
- JavaScript-библиотека google-auth-library;
- JavaScript-библиотека googleapis;
- JavaScript-библиотека axios.
- JavaScript-библиотека React.Js;
- Sass;
- Typescript;
- JavaScript-библиотека Fullcalendar;
- Библиотека react-oauth/google.

4. Реализация веб-приложения

Реализация серверной части потребовала решения следующих технических задач:

1. Аутентификация и авторизация пользователя. Для этого нужно получить сессионный токен для дальнейшей работы с данными Google пользователя. Данная задача решается с использованием метода OAuth2Client из библиотеки google-auth-library. Для инициализации

OAuth2Client необходимы GoogleId и SecretKey, полученные в кабинете разработчика Google API.

2. Получение информации о календарях пользователя. Метод API серверной части приложения решает данную задачу с помощью метода Google API «calendarList.list» [3].

3. Получение событий. Данная функция реализована с использованием метода Google API «events.list» [3] и параметра уникального идентификатора Google Calendar для синхронизации с выбранным пользователем календарем.

4. Создание события в Google Calendar. Для создания события реализован метод API, принимающий в параметрах название, описание, локацию, начальное время и конечное время события, уникальный идентификатор Google Calendar и булевая переменная для конфиденциальности события. В методе API серверной части веб-приложения используется метод Google API «events.insert»[3].

При реализации клиентской части решались следующие задачи:

1. Выбор Google Calendar для синхронизации по умолчанию. В веб-приложении пользователю предоставляется возможность выбора одного или несколько Google календарей для синхронизации. В аналогах такой возможности нет. Для это нужно нажать на кнопку с названием нужного календаря.



Рис. 1. Выбор одного или несколько календарей для синхронизации

2. Настройка конфиденциальности создаваемых событий календаря. Пользователю предоставляется возможность выбрать состояние добавляемого события в Google Calendar, которое может быть приватным (без сведений) или публичным (со всеми сведениями). Реализация данной возможности представлена на Рис. 2.

Настройка конфиденциальности календарей Выберите, какой календарь Вы бы хотели сделать приватным

Рабочий календарь
plaksina.ekaterina20@gmail.com

Рис. 2. Выбор статуса добавляемого события в Google Calendar

3. После настройки конфиденциальности пользователь видит созданный календарь. События в публичном календаре приватны, т. е. для каждого из них видно только начальное и конечное время события, но не полная информация о событии.



Рис. 3. Созданный публичный календарь

4. Для создания события нужно нажать на кнопку «Create Event», открывающую всплывающее окно с названием, описанием, локацией, временем начала и окончания события (Рис 4). Созданные события в публичном расписании и в Google Calendars с настройкой приватности представлены на Рис. 5, Рис. 6, Рис. 7.

The image shows a form for creating an event, overlaid on a grid background. The form contains the following fields and elements:

- Summary:** A text input field containing the word "Конференция".
- Description:** A larger text input field.
- Location:** A text input field.
- Start Date Time:** A date and time picker showing "12.05.2023, 16:41".
- End Date Time:** A date and time picker showing "12.05.2023, 17:41".
- create event:** A button at the bottom of the form.

In the bottom right corner of the grid, there is a small dark grey box containing the text "6:30 - 7:30".

Рис. 4. Форма создания события

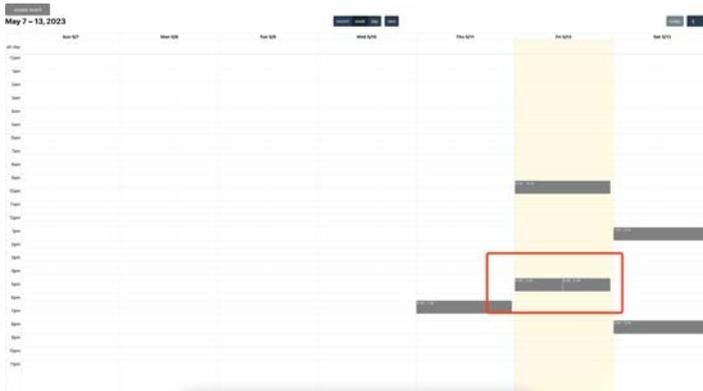


Рис. 5. Созданные события в публичном расписании выделены красным цветом

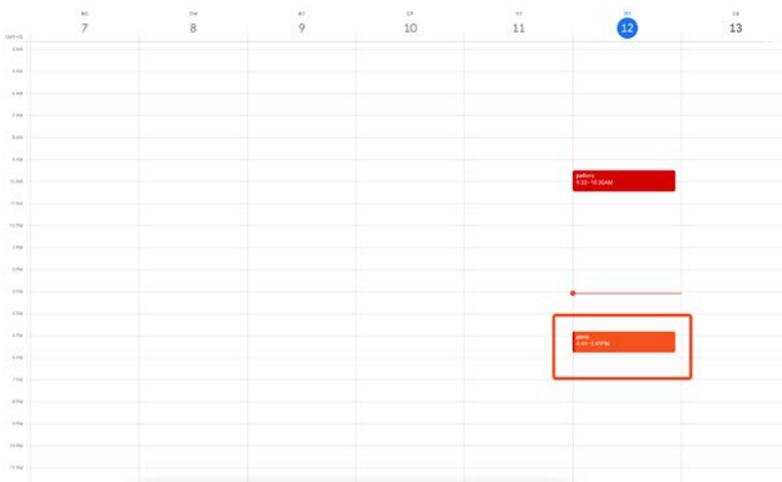


Рис. 6. Созданное приватное событие в рабочем Google Calendar, выделенное красным цветом

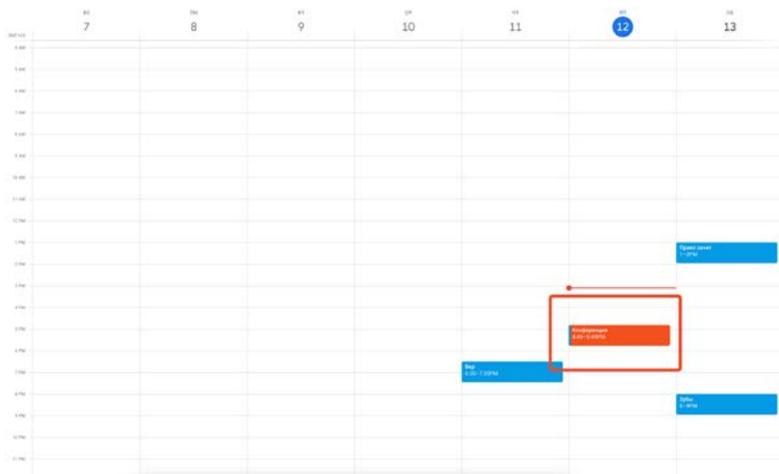


Рис. 7. Созданное публичное событие в другом Google Calendar, выделенное красным цветом

Заключение

В результате проделанной работы разработано веб-приложение для создания публичного расписания с возможностью выбора одного или несколько календарей для направленной синхронизации и видимости

добавляемого события в Google Calendars, сохранением конфиденциальности событий в публичном календаре.

Список литературы

1. Время для альтернативы календарю Google: Установить календарь Google на всеобщее обозрение легко - и опасно. [Электронный ресурс] – Режим доступа: <https://tutanota.com/ru/blog/posts/google-calendar-alternative>
2. LostApp [Электронный ресурс] – Режим доступа: <http://lostapp.ru/soft/youcanbook-me>
3. Google Calendar Api [Электронный ресурс] – Режим доступа: <https://developers.google.com/calendar/api/v3/reference>

Практическое применение рекомендаций OWASP для тестирования REST API

И В. А. Рощупкин

Студент бакалавриата

Я. А. Воронцова

Доцент

Введение

Контроль безопасности приложений в наше время очень важен, поскольку при ненадлежащих технических мерах безопасности возможно нарушение конфиденциальности, целостности и доступности данных, с которыми приложения работают. Частный случай тестирования на безопасность - проверка REST API, чему и посвящена данная статья.

Для метрик оценки и тестирования используются рекомендации, предложенные открытым проектом по безопасности веб-приложений (OWASP). OWASP – это мировой некоммерческий проект, объединяющий тысячи профессионалов в области кибербезопасности со всего мира. В рамках данного проекта был создан список из десяти наиболее распространенных уязвимостей веб-приложений [1].

В данной статье рассматриваются три наиболее распространенные типа уязвимостей, которые могут быть обнаружены при тестировании REST API: Broken Access Control, SQL-injection и XSS, а также показывается практическое применение рекомендаций OWASP для обнаружения этих типов уязвимостей [2]. Далее в статье рассматривается тестирование безопасности на примере некоторых методов REST API финансового приложения, которое предоставляет пользователям удобные и эффективные инструменты для управления и контроля своими финансами и совершения финансовых операций. Предлагается качественным результатам тестирования (набор тестов, пройден тест или нет) сопоставить количественные метрики и построить оценочную функцию, результат которой будет использоваться менеджерами предприятия для принятия решения о готовности сборки к выпуску.

1. Broken Access Control

Broken Access Control (Нарушение контроля доступа) — это уязвимость, которая позволяет злоумышленникам получить доступ к конфиденциальной информации, который они не должны иметь. Это может произойти, если приложение неправильно ограничивает доступ к конкретным ресурсам. Для проверки корректности контроля доступа к ресурсам необходимо удостовериться, что различные пользователи имеют доступ только к тем ресурсам, которые им предоставлены. Проще всего такой контроль осуществлять с помощью одной из моделей доступа, например, RBAC (Role-based access control) - ролевой модели доступа. Также следует проверить, что авторизация происходит только после подтверждения подлинности пользователя.

Следуя описанной логике, достаточно проверить, что администратор может иметь доступ ко всем ресурсам, а обычный пользователь – только к данным своего профиля, а также к данным, специфичным для его роли в модели RBAC [3].

В ходе тестирования Broken Access Control были протестированы следующие API:

- POST /createNewAccount
- POST /createNewUser
- POST /checkAccount
- GET /checkUser
- GET /checkBalance

Исследуемое приложение предоставляет такие роли, как:

- Администратор: доступ ко всем API
- Пользователь: доступ только к /checkBalance API

На рис. 1 демонстрируется успешное выполнение метода GET /checkUser для роли администратора.

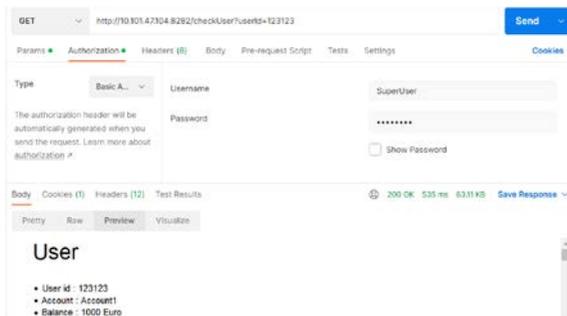


Рис. 1. Выполнение API запроса от имени администратора

В ходе дальнейшего тестирования этот же метод был вызван от лица пользователя (Рис. 2.)

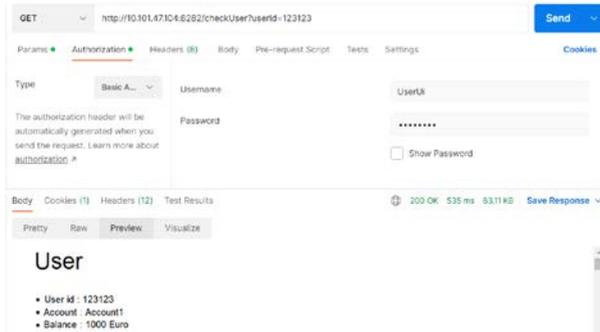


Рис. 2. Выполнение API запроса от имени пользователя

Согласно логике работы приложения, пользователь не должен иметь доступа к /checkUser API, однако результат выполнения запроса свидетельствует об обратном, и пользователь получил доступ к чужому профилю. Этот факт говорит о нарушении контроля доступа.

В ходе исправления ошибки была внедрена ролевая модель доступа: доступ к ресурсу /checkUser был предоставлен только пользователям с административными правами. Повторный вызов метода /checkUser для учетной записи с правами обычного пользователя завершился с кодом HTTP 403 (Forbidden), который указывает на то, что доступ к этой API для данной роли ограничен. (рис. 3.)

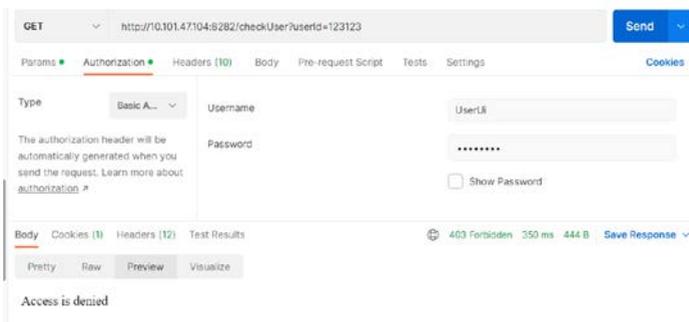


Рис. 3. Ожидаемое поведение системы при выполнении API запроса для роли пользователя

Приложение корректно проверило роль текущего пользователя, обращающегося к API, и заблокировало доступ к ресурсу, поскольку доступ к выполнению /checkUser для пользователя недоступен.

2. SQL-injection

SQL-injection (SQL-инъекция) — это вид атаки на приложения, использующие реляционные СУБД для хранения данных, который осуществляется путем внедрения вредоносного кода через параметры, используемые для ввода данных. Например, злоумышленник может вставить SQL-запрос в форму, предназначенную для поиска, обновления, удаления данных или вставки, что может привести к их утечке или нарушению конфиденциальности, такой как пароли, имена пользователей и другие данные.

Для предотвращения SQL-injection рекомендуется использовать параметризованные запросы, что позволяет избежать возможности выхода за контекст данных. Кроме того, необходимо проверять входные данные на наличие символов, которые могут быть использованы для SQL-injection, таких как одинарные кавычки или точки с запятой [3].

В ходе тестирования SQL-injection были проверены следующие методы API:

- POST /register
- POST /login
- POST /logout

Предположим, что приложение не содержит параметризованные запросы, а выполняется сразу выборка из базы данных. Исходя из этого предположения был сконструирован следующий вид входных данных (рис. 4.)



The image shows a web form with two input fields: 'Username' and 'Password'. The 'Username' field contains the text 'hacker' or 1 = 1; --. The 'Password' field contains three asterisks. Below the fields is a green 'Login' button. The form is set against a light gray background.

Рис. 4. Внедрение SQL-инъекции

На рис. 5 показан результат выполнения входа. Инъекция сработала успешно, и злоумышленник смог получить административный доступ.

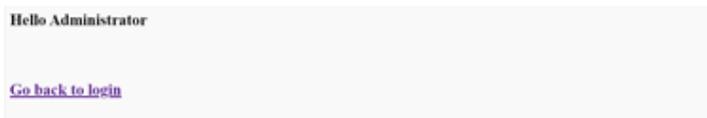


Рис. 5. Удачный вход для несуществующего пользователя

На рис.6 изображен результат тестирования после исправления уязвимости согласно рекомендациям OWASP. SQL-инъекция не сработала, поскольку были применены параметризованные запросы



Рис. 6. Неуспешная авторизация пользователя

3. XSS-уязвимость

XSS-уязвимость (Межсайтовый скриптинг) — это уязвимость, которая позволяет злоумышленникам внедрять JavaScript код на веб-странице, который будет выполняться в браузере пользователя. Это может привести к краже активных пользовательских сессий, учетных записей, перенаправлению пользователя на фальшивые сайты, распространению вредоносных программ и другим негативным последствиям.

Для обнаружения и предотвращения XSS-атак в REST API необходимо проверять все входные данные, которые могут быть использованы для отображения на веб-странице. Рекомендуется использовать HTML-кодирование для всех данных, которые передаются от пользователя. Это помогает предотвратить внедрение злонамеренного кода в HTML-страницы и защитить приложение от XSS-атак [5].

В ходе тестирования XSS-уязвимости были протестированы следующие API:

- POST /checkAccount
- POST /checkUser
- POST /checkBalance

Для демонстрации XSS-уязвимости был использован /checkUser API. Прежде чем проверять API на уязвимость, следует убедиться, что он корректно работает на веб-странице (Рис. 7.):



Рис. 7. Успешное выполнение /checkUser API

XSS-уязвимость требует внедрения злоумышленного кода на страницу. Исходя из этого для /checkUser API в параметр для поиска был добавлен JavaScript код, который позволяет видеть куки пользователя. (Рис. 8.)

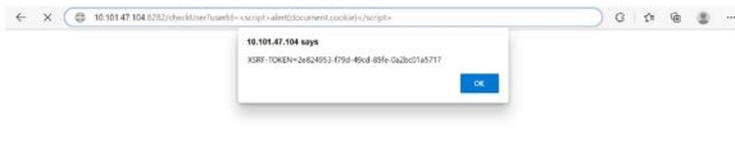


Рис. 8. Внедрение XSS

XSRF-токен используется для защиты от межсайтовой подделки запросов (Cross-Site Request Forgery). Раскрытие этого токена с помощью XSS, говорит о том, что такой токен может быть украден и использован уже для другой атаки [1].

Для фильтрации входных данных и предотвращения возможной XSS атаки использовалась техника "экранирования" (escaping). Это означает, что определенные символы, которые могут быть использованы для XSS атаки должны быть заменены на их HTML-сущности. Это позволяет сохранить текст в его исходном виде, не давая браузеру интерпретировать его как HTML-код.

Повторное выполнение /checkUser API с вставленным в параметр для поиска JavaScript кодом отобразило сведения о пользователя на веб-странице (Рис. 11.):



попадает в красную зону, что означает, что приложение не готово к выпуску

Уязвимость	Баллы	Готовность
Broken Access Control:	200	
SQL-injection:	50	
XSS:	50	
Сумма:	300	

Рис. 11. Результаты тестирования уязвимостей

Частичное исправление уязвимостей приводит к следующему результату (Рис. 12.)

Уязвимость	Баллы	Готовность
Broken Access Control:	0	
SQL-injection:	0	
XSS:	50	
Сумма:	50	

Рис. 12. Частичная готовность приложения к выпуску

Наконец, полностью исправленные уязвимости дают оценку 0 баллов, что говорит о полной готовности приложения к выпуску.

Уязвимость	Баллы	Готовность
Broken Access Control:	0	
SQL-injection:	0	
XSS:	0	
Сумма:	0	

Рис. 13. Отсутствие заявленных уязвимостей

Заключение

Тестирование уязвимостей согласно рекомендациям OWASP, позволило выявить и исправить потенциальные слабые места, которые могли быть использованы злоумышленниками для атак и компрометации системы, а также помогло повысить общий уровень безопасности, предотвратить финансовые потери, связанные с нарушением данных и соответствовать требованиям регулятивных организаций и стандартов безопасности.

На примере трёх наиболее популярных уязвимостей было рассмотрено, как можно качественную оценку перевести в количественную при помощи оценочной функции. В дальнейшем предполагается расширение перечня уязвимостей и их поддержки в оценочной функции.

Список литературы

1. OWASP TOP 10 [Электронный ресурс]. – Режим доступа : <https://owasp.org/www-project-top-ten/>
2. OWASP Testing Guide [Электронный ресурс]. – Режим доступа : <https://owasp.org/www-project-web-security-testing-guide/>
3. REST Security Cheat Sheet [Электронный ресурс]. – Режим доступа: https://cheatsheetseries.owasp.org/cheatsheets/REST_Security_Cheat_Sheet.html
4. Чек-лист устранения SQL-инъекций [Электронный ресурс]. – Режим доступа : <https://habr.com/ru/companies/pentestit/articles/546232/>
5. Безопасность REST API от А до ПИ [Электронный ресурс]. – Режим доступа : <https://sushkov.ru/api-security/>

Блокчейн-хранилище сертификатов моделей устройств интернета вещей

Д. В. Рыбас

Студент бакалавриата

Д. И. Соломатин

Старший преподаватель

Введение

Современный мир тяжело представить без интернета вещей. Его устройства делают жизнь каждого человека проще и эффективнее. Для обеспечения их надежности и безопасности предусмотрен механизм сертификации. Это процесс, который позволяет проверить, соответствуют ли устройства определенным стандартам безопасности и качества. Однако традиционные методы сертификации могут быть очень затратными и медленными. Кроме того, они могут быть подвержены мошенничеству и не обладать достаточной прозрачностью. В этом контексте технология блокчейн представляется наиболее перспективной для создания распределенного, отказоустойчивого и прозрачного хранилища сертификатов моделей устройств интернета вещей. Данное хранилище позволит компаниям-участникам осуществлять процесс сертификации в децентрализованной среде, где каждый участник сети будет иметь доступ к проверенной и подтвержденной информации о сертификации устройств.

В данной работе будет рассмотрен процесс разработки такого хранилища.

1. Процесс сертификации

Процесс сертификации можно разбить на несколько этапов:

- создание модели устройства, подлежащего сертификации;
- сертификация устройства по стандарту Matter;
- добавление информации о модели устройства в хранилище;
- добавление информации о версии модели устройства в хранилище;
- добавление информации о сертификации версии модели устройства в хранилище.

2. Требования к блокчейн-хранилищу

При разработке рассматриваемой системы учитывались следующие требования [1]:

- все транзакции должны подписываться ключами участников сети;
- неизменяемый блокчейн, являющийся авторитетным, прозрачным и публично доступным для любого члена сети;
- легкость добавления новых узлов;
- протокол консенсуса, требующий одобрения транзакций от большинства участников сети;
- наличие WEB и CLI интерфейсов.

3. Реализация

В качестве средств реализации описанной системы были выбраны язык Go, Cosmos SDK и Tendermint.

Cosmos SDK является основой для множества успешно запущенных проектов. Он построен на основе Tendermint. Cosmos отвечает за логику блокчейна, а Tendermint за консенсус.

Ниже приводятся модели данных, которые используются в разработанной системе для сертификации устройств интернета вещей.

В листинге 1 представлена Protobuf-модель устройства [2].

Листинг 1

Protobuf-модель устройства

```
message Model {
  int32 vid = 1;
  int32 pid = 2;
  int32 deviceId = 3;
  string productName = 4;
  string productLabel = 5;
  string partNumber = 6;
  string creator = 7 [(cosmos_proto.scalar) =
                    cosmos.AddressString];
}
```

Данная модель содержит следующие поля:

- vid – идентификатор вендора;
- pid – идентификатор продукта модели;
- deviceId – идентификатор типа устройства;
- productName – имя модели;
- productLabel – описание модели;
- partNumber – идентификатор модели для отслеживания на складе;
- creator – создатель транзакции.

В листинге 2 представлена Protobuf-модель версии устройства [2].

Листинг 2

Protobuf-модель версии устройства

```
message ModelVersion {
  int32 vid = 1;
  int32 pid = 2;
  uint32 softwareVersion = 3;
  string softwareVersionString = 4;
  int32 cdVersionNumber = 5;
  uint32 minApplicableSoftwareVersion = 6;
  uint32 maxApplicableSoftwareVersion = 7;
  string creator = 8 [(cosmos_proto.scalar) =
    "cosmos.AddressString"];
}
```

Данная модель содержит следующие поля:

- vid – идентификатор вендора;
- pid – идентификатор продукта модели;
- softwareVersion – версия ПО модели;
- softwareVersionString – дополнительная информации о версии ПО модели (дата выпуска, номер сборки и т.д.);
- cdVersionNumber – номер текущей версии устройства, который указывается в его сертификации;
- minApplicableSoftwareVersion – минимальная совместимая версия ПО;
- maxApplicableSoftwareVersion – максимальная совместимая версия ПО.

В листинге 3 представлена Protobuf-модель информации о сертификации [2].

Листинг 3

Protobuf-модель информации о сертификации

```
message ComplianceInfo {
  int32 vid = 1;
  int32 pid = 2;
  uint32 softwareVersion = 3;
  string certificationType = 4;
  string softwareVersionString = 5;
  uint32 softwareVersionCertificationStatus = 6;
  string date = 7;
  string reason = 8;
  string owner = 9 [(cosmos_proto.scalar) =
    "cosmos.AddressString"];
  string cdCertificateId = 10;
}
```

Данная модель содержит следующие поля:

- vid – идентификатор вендора;
- pid – идентификатор продукта модели;
- softwareVersion – версия ПО модели;
- certificationType – тип сертификации;
- softwareVersionString – дополнительная информации о версии ПО модели (дата выпуска, номер сборки и т.д.);
- softwareVersionCertificationStatus – статус сертификации версии ПО модели;
- date – дата сертификации;
- reason – причина сертификации.

4. Интерфейс пользователя

Хранилище имеет WEB и CLI интерфейсы.

На рис. 1 представлена домашняя страница Web-интерфейса. На ней можно увидеть различные данные о хранилище. Например, количество активных аккаунтов, устройств и т.д.

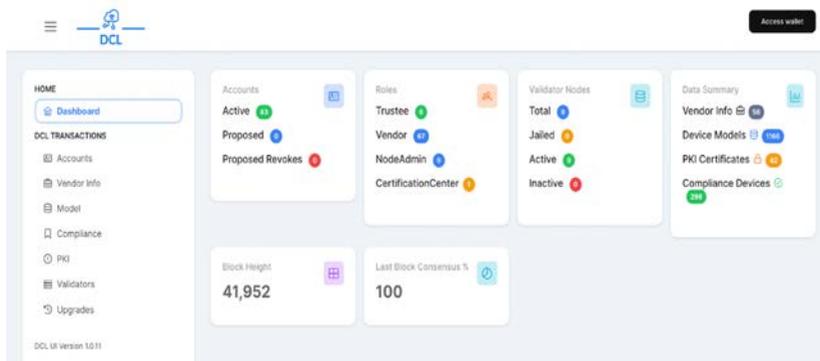


Рис. 1. Домашняя страница Web-интерфейса

На рис. 2 представлены данные об устройствах, модель которых описана в листинге 1.

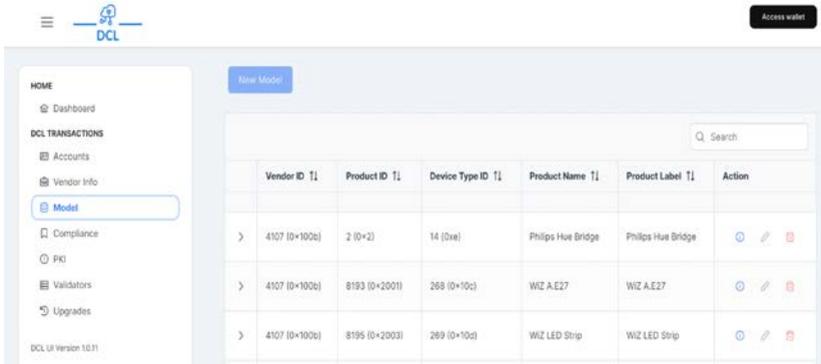


Рис. 2. Список устройств в Web-интерфейсе

На рис.3 представлены данные о сертификации устройств, модель которой описана в листинге 3.

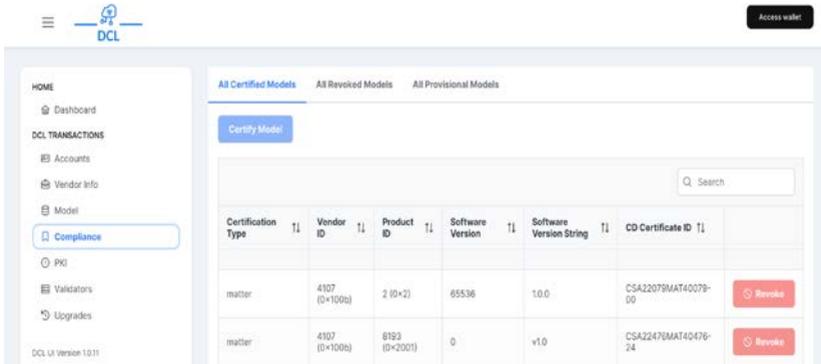


Рис. 3. Список сертифицированных устройств в Web-интерфейсе

Общий формат доступных команд [2] представлен в листингах 4-6.

Листинг 4

CLI-команда добавления модели

```
dclid tx model add-model --vid=<uint16> --pid=<uint16> \
--deviceTypeID=<uint16> --productName=<string> \
--productLabel=<string> --partNumber=<string>
```

CLI-команда добавления версии модели

```

dclid tx model add-model-version --vid=<uint16> --pid=<uint16>
--softwareVersion=<uint32> --softwareVersionString=<string> \
--cdVersionNumber=<uint32> \
--minApplicableSoftwareVersion=<uint32> \
--maxApplicableSoftwareVersion=<uint32> --from=<account>

```

CLI-команда сертификации модели

```

dclid tx compliance certify-model --vid=<uint16> --pid=<uint16>
--softwareVersion=<uint32> --softwareVersionString=<string> \
--certificationType=<matter> \
--certificationDate=<rfc3339 encoded date> \
--reason=<string> --from=<account>

```

Заключение

В результате выполнения работы было реализовано блокчейн-хранилище, отвечающее необходимым требованиям. На данный момент оно используется такими компаниями, как Apple, Google, Amazon, Samsung и т.д. и хранит информацию о:

- 83 активных аккаунтах;
- 1164 устройствах;
- 297 сертифицированных устройствах.

Список литературы

1. DCL whitepaper [Электронный ресурс]. – Режим доступа : https://csa-iot.org/wp-content/uploads/2022/09/DCL_Whitepaper.pdf
2. Документация GitHub-репозитория проекта [Электронный ресурс]. – Режим доступа : <https://github.com/zigbee-alliance/distributed-compliance-ledger/tree/master/docs>

Генерация игрового поля для пошаговых настольных игр

Н. Ю. Савина

Студент бакалавриата

Н. К. Самойлов

Старший преподаватель

Введение

Настольные игры являются одним из популярных и полезных видов досуга. Они позволяют не только приятно проводить время в кругу семьи или друзей, но и развивать логику, воображение, скорость реакции.

Целью данной работы является разработка пошаговой настольной игры для платформы Android. Разрабатываемая игра основывается на игре «Чудовище Джо-Джанги», автором которой является кандидат педагогических наук Л.А.Кутырёв-Трапезников. Игра была создана в 1992 году, является самой известной игрой Кутырева-Трапезникова. В 1993 году игре была присуждена Золотая медаль на Международной ярмарке игрушек в Петербурге.

В игре принимают участие от двух до шести игроков. Игроки отправляются в неизвестную местность – Джо-Джангу. Им необходимо пройти все испытания путешествия: сначала нужно умело собраться в дорогу, потом преодолеть трудности на пути в Джо-Джангу, а уже там сфотографировать или заснять на киноплёнку опасных животных, которые способны напасть. Но главная цель – вернуться в лагерь живым и богатым. Игра завершается, когда вернется хотя бы один из игроков, а победитель определяется по количеству денег, полученных за фотографии, видеозаписи, проданные шкуры животных и предметы снаряжения.

Игра состоит из игрового поля, карточек ситуаций и предметов, игровых денег, игровых кубиков.

1. Архитектура приложения

Разрабатываемое приложение является мобильным приложением для платформы Android, его архитектура состоит из трех основных компонентов, описанных в [1, 2]:

- game – реализует логику игры;

- view – пользовательский интерфейс приложения;
- application – отвечает за взаимодействие игры и отображения.

2. Механика игры

Ядром видеоигры является игровой движок, и при реализации рассматриваемой игры одной из задач была разработка собственного движка для мобильных настольных игр, который можно было бы использовать повторно и позволял бы создать не только «Чудовище Джо-Джанги», но и «Монополию» и другие настольные игры подобного жанра.

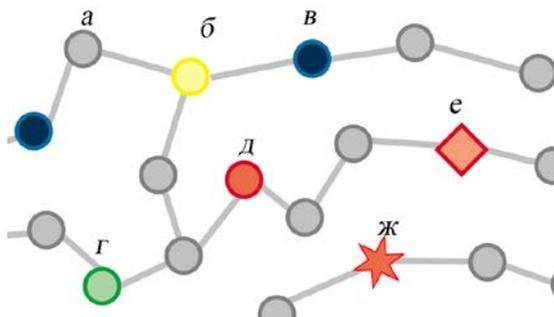
Основными компонентами движка стали объектная модель игрового поля, модель игрока, модель игрового кубика и интерфейс самой игры. На рис. 1 изображена диаграмма классов игрового движка.

Рис. 1. Диаграмма классов игрового движка

1. Игровое поле

Игровое поле может быть смоделировано с помощью графов, где каждая клетка – это вершина графа, а переходы между клетками – ребра.

Подробное описание графов можно найти в [3]. На рис. 2 приведен фрагмент игрового поля.



а – клетка без действий, б – клетка использования карточек ситуаций, в – клетка переправы через реку, г – клетка перехода через болото, д – красная клетка встречи с туземцами, е – красная клетка встречи с гангстерами, животными, ж – красная клетка встречи с чудовищами

Рис. 2. Фрагмент игрового поля

Для создания игрового поля используются классы `Field` – поле, `Place` – клетка, `Transition` – переход между клетками. У каждой клетки есть входящие и исходящие переходы, а у каждого перехода есть источник и цель. В данной работе используется модель ориентированного графа.

Правила игры предусматривают различные типы клеток, от которых зависят действия игроков. Разные события на клетках поля реализованы с помощью интерфейса `IPlaceAction`. Вызов метода `executePlaceAction` позволяет выполнить необходимое действие на клетке. На рис. 3 изображена диаграмма классов игрового поля.

2. Игрок

Моделирование игрока происходит за счет создания объекта, наследующего `Player`. Игрок содержит информацию об игровых предметах, которыми он владеет, количестве денег персонажа.

3. Игральный кубик

Для реализации собственного кубика, или кубиков, необходимо реализовать интерфейс `IDice` с методом `rollDice()`. Различные реализации этого метода позволят моделировать выброс кубиков, необходимый по правилам игры. В данной работе используется кубик с шестью гранями.

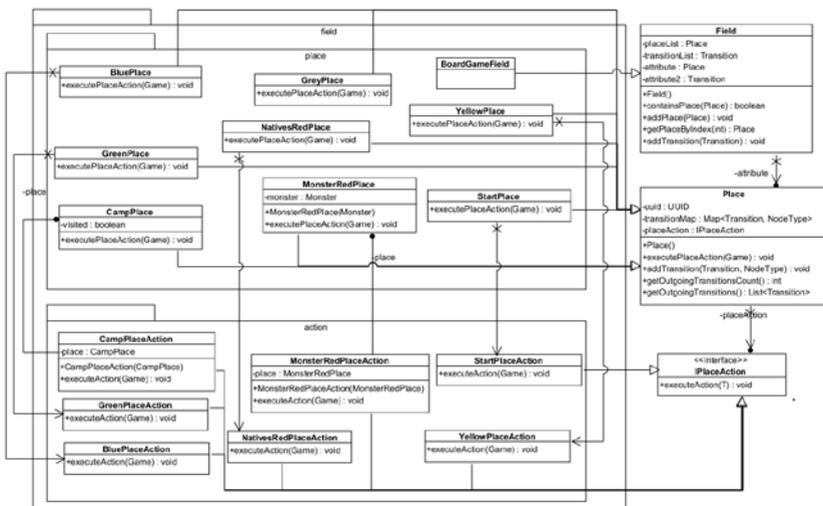


Рис. 3. Диаграмма классов игрового поля

4. Основной игровой цикл

Игра начинается с распределения между всеми участниками 15000 единиц внутриигровой валюты. Каждый игрок тратит эти деньги на снаряжение в экспедицию, также деньги можно выделить для оплаты переправы через реку, на покупку лошадей или на эвакуацию обратно в лагерь.

Игровой цикл – зацикленный набор игровых механик, который использует пользователь. Игровой цикл, как шаблон проектирования, описан в [4]. В разрабатываемой игре основной игровой цикл состоит из нескольких шагов:

1. Выбор игрока для хода;
2. Выбрасывание кубиков;
3. Перемещение игрока по полю в соответствии с выпавшим числом;
4. Выполнение действий на клетке.

Действия на клетке зависят от типа клетки и могут включать в себя:

- для желтых клеток – получение карточки ситуации;
- для синих клеток – использование средств переправы (лодка, плот) и бросок кубика, чтобы определить исход переправы;
- для зеленых клеток – бросок кубика и применение защиты от москитов или попытки выбраться из болота, теряя предметы;

- для красных клеток – торговля с туземцами или отпугивание их, защита от гангстеров и животных, съемка чудовищ или защита от их нападения;
- для клеток привала, монастыря, хижины – покупка или продажа предметов.

В зависимости от числа, выпавшего на кубике, а также снаряжения игрока, действия на клетках могут привести к гибели путешественника, и игрок должен будет начать игру сначала.

Игровой цикл выполняется до тех пор, пока игра не завершится: пока не вернется хотя бы один игрок.

5. Генерация игрового поля

Использование графа в качестве модели игрового поля позволяет создавать новые поля, используя уже известные алгоритмы генерации графов.

На карте игры присутствуют различные области, которые влияют на типы клеток поля: равнины, болота, реки и Джио-Джанга. На равнинах встречаются серые, желтые клетки, красные клетки туземцев и животных, привалы, монастыри. Болота – местонахождение зеленых клеток, реки – местонахождение синих клеток. В Джио-Джанге встречаются клетки чудовищ, которых можно фотографировать или атаковать. На болотах, реках и в Джио-Джанге также встречаются серые и желтые клетки.

Такое разбиение игрового поля на области позволяет использовать процедурную генерацию карты. При процедурной генерации карты необходимо:

1. Построить диаграмму Вороного;
2. Распределить полученные области между областями игровой карты;
3. Сгенерировать граф игрового поля на основании полученного разбиения.

В разрабатываемом приложении для генерации диаграммы Вороного используется алгоритм Форчуна. Подробное описание диаграммы Вороного и алгоритма Форчуна можно найти в [5].

Далее в каждой из областей необходимо сгенерировать граф игрового поля в соответствии с клетками, подходящими области. В разрабатываемом приложении узлы полигонов диаграммы Вороного используются в качестве вершин графа и соединяются ребрами. Полученный граф используется в игре в качестве поля.

3. Пользовательский интерфейс

Разрабатываемое приложение является мобильным и состоит из двух экранов:

- экрана стартового меню;
- экрана игры.

Экран стартового меню позволяет настроить игру перед началом (добавить игроков в команду, распределить игровые предметы), а также непосредственно начать игру.

Экран игры (рис. 4) отвечает за отображение состояния текущей игры, игрового поля, персонажей игроков, а также выводит на экран диалоговые окна, необходимые при выборе игроком предметов или пути перемещения по игровому полю.

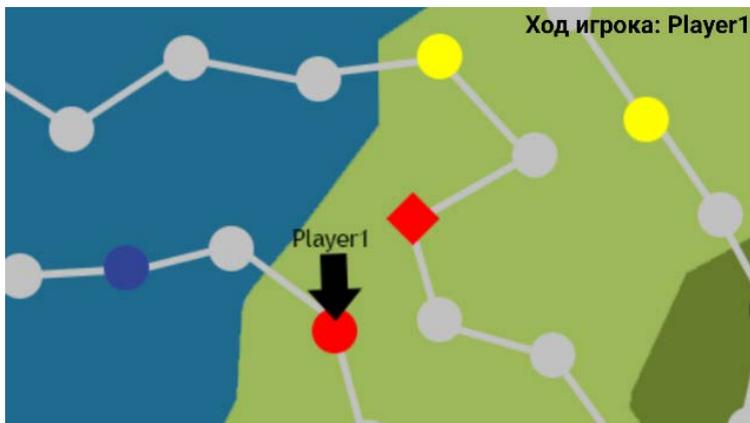


Рис. 4. Пользовательский интерфейс, отображение игрового поля, активного игрока

Для отрисовки игровых объектов используются «обертки» для поля и его компонентов, а также персонажей: `FieldWrapper`, `PlaceWrapper`, `TransitionWrapper`, `CharacterWrapper`. Это позволяет определять дополнительные параметры для игровых объектов, которые не влияют на игру, но используются для отрисовки, например, координаты клеток на поле или размер поля. Применение средств платформы Android для отображения игровых компонентов описано в [6].

Заключение

В данной работе рассмотрена разработка пошаговой настольной игры для платформы Android.

Приложение основано на настольной игре «Чудовище Джо-Джанги».

- В функциональность игры входит:
- Генерация игрового поля;

- Прохождение игроков по игровому полю;
- Применение стратегий игровых клеток в зависимости от типа клетки;
- Применение игровых предметов;
- Применение игровых карточек ситуаций;
- Использование игрового кубика;
- Добавление новых игроков и игра с ними.

Список литературы

1. McShaffry, M. Game Coding Complete / M. McShaffry, D. Graham. – Cengage Learning, 2012. – 911 p.
2. Gregory, J. Game Engine Architecture / J. Gregory. – 3rd edition. – New York: A K Peters/CRC Press, 2018. – 864 p.
3. Касьянов, В. Н. Графы в программировании: обработка, визуализация и применение / В.Н. Касьянов, В.А. Евстигнеев – СПб.: БХВ-Петербург, 2003. – 1104 с.
4. Nystrom, R. Game Programming Patterns / R. Nystrom. – Genever Benning, 2014. – 354 p.
5. Computational Geometry: Algorithms and Applications / M. Berg [et al.]. – 3rd edition. – Heidelberg: Springer Berlin, 2008. – 386 p.
6. Zechner, M. Beginning Android Games / M. Zechner, J. F. DiMarzio, R. Green. – Berkeley, CA: Apress, 2011. – 688 p.

Использование Debezium в микросервисной архитектуре для реализации паттерна захвата изменения данных

А. Р. Сашина

Студент бакалавриата

А. Л. Гавшин

Старший преподаватель

Введение

Современные веб-приложения используют микросервисную архитектуру. Внутри них происходит взаимодействие сервисов и баз данных друг с другом. Приложения с такой архитектурой проще обслуживать и тестировать благодаря независимости отдельных компонент системы.

Однако в микросервисных приложениях может возникнуть проблема поддержки согласованности информации в базе данных, когда изменение состояния отношения должно привести к выполнению некоего события, например, оповещение других систем. Для отслеживания изменения данных применяется шаблон разработки «Change Data Capture» (захват изменения данных). Данная работа посвящена описанию работы платформы Debezium для реализации вышеупомянутого паттерна.

1. Описание паттерна захвата изменения данных

Захват изменения данных является программным процессом, который идентифицирует и отслеживает изменения информации в базе данных [1]. Данный процесс обеспечивает перемещение данных в реальном или почти реальном времени, непрерывно обрабатывая информацию по мере возникновения новых событий в базе данных [2].

Реализация данного шаблона разработки состоит из трех этапов: обнаружение, захват и распространение данных. Паттерн захвата изменения данных используют в современных облачных и микросервисных архитектурах для высокоэффективного перемещения информации по глобальной сети, аналитике в реальном времени, обеспечении синхронизации систем и надежной репликации данных.

2. Различные подходы для отслеживания изменения данных

Пусть существует информационная система, в которой при обновлении информации в отношении базы данных срабатывал триггер, оповещающий другое приложение об изменении. Если рассматриваемая система внутри себя имеет только один сервис, который взаимодействует с базой данных, то реализация триггера была бы описана внутри бизнес-логики данного сервиса. Схема взаимодействия приложения и базы данных показана на рис. 1.

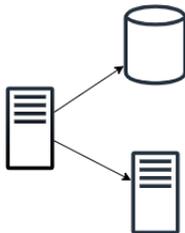


Рис. 1. Один сервис изменяет отношение в базе данных и оповещает другое приложение

Но если с отношением взаимодействует не один сервис, а несколько, то в каждом сервисе необходимо будет продублировать вышеописанное поведение. Альтернативным решением является использование брокера сообщений, например, Apache Kafka. При данном подходе создается определенная группа (топик), в которую отправляются сообщения об изменении данных в отношении базы данных. Для чтения сообщений создается новый сервис - потребитель (консьюмер), читающий содержимое топика [3]. Потребитель во время обработки сообщения об изменении данных также осуществляет реализацию и вызов триггера. Схема взаимодействия показана на рис. 2.

В большинстве случаев использование брокера сообщений является оптимальным решением. Однако, если обновление информации в отношении будет происходить в самой базе данных, а не через сервис, то брокер не получит соответствующего сообщения об изменении. Также такой подход не сработает в случае, когда с базой данных взаимодействует стороннее приложение, не обладающее информацией о топиках рассматриваемой системы.

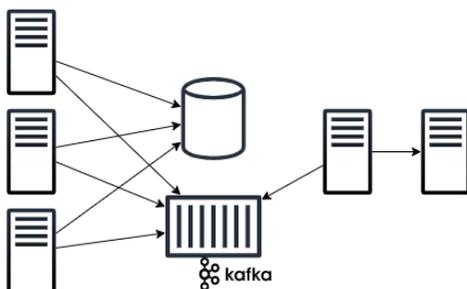


Рис. 2. Несколько сервисов изменяют базу данных и отправляют сообщения в брокер, который читает другой сервис для оповещения

3. Использование Debezium для захвата изменения данных

Для реализации триггера рассматриваемого приложения подходит распределенная платформа Debezium, предоставляющая библиотеку коннекторов к различным системам управления баз данных [4]. Эти коннекторы могут отслеживать и записывать изменения информации на уровне строк в отношениях, реализуя шаблон разработки захвата изменения данных. Debezium можно использовать в связке с рассмотренной выше Apache Kafka или подключить как библиотеку в самом приложении.

Если использовать платформу Debezium, то можно создать сервис, следящий за изменениями в базе данных и отправляющий оповещение другому приложению. Таким образом, реализация триггера будет описана в отдельном сервисе, который не зависит от работы других систем. Схема взаимодействия показана на рис. 3.

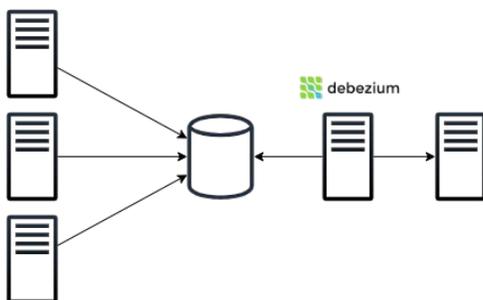


Рис. 3. Несколько сервисов изменяют базу данных, а сервис с использованием Debezium реагирует на изменения и оповещает другое приложение

Заключение

Данная статья посвящена описанию использования платформы Debezium для реализации шаблона разработки захвата изменения данных. Был описан сам паттерн захвата изменения данных, а также продемонстрированы различные решения проблемы разработки триггера на основе изменения состояния отношения в базе данных. Использование подхода с платформой Debezium позволяет спроектировать систему, которая способна при изменении отношения выполнять некоторую логику, например, оповещение другого приложения, в независимости от количества интегрируемых сервисов.

Список литературы

1. Захват изменения данных [Электронный ресурс] : статья компании Qlik. – Режим доступа: <https://www.qlik.com/us/change-data-capture/cdc-change-data-capture>
2. Клеппман, М. Высоконагруженные приложения. Программирование, масштабирование, поддержка / М. Клеппман. – СПб. : Питер, 2018. – 640 с.
3. Нархид, Н. Apache Kafka. Поточковая разработка и анализ данных / Н. Нархид, Г. Шапира, Т. Палино. – СПб. : Питер, 2019. – 320с.
4. Документация Debezium [Электронный ресурс] : документация платформы Debezium. – Режим доступа: <https://debezium.io/documentation/reference/2.2/index.html>

Разработка онлайн-портала для вышивок

Е. А. Селиванова

Студент бакалавриата

П. С. Лысачев

Старший преподаватель

Введение

Каждый человек когда-то сталкивался с проблемой, что придуманные идеи в теории не всегда оказываются такими же на практике. И чтобы протестировать задуманное, необходимо каждый раз создавать реальную модель, тратить на это свои ресурсы, которые в большинстве случаев могут и вовсе не окупиться. В частности, такое затруднение касается дизайнеров вышивок, так как на данный момент не существует таких платформ, которые позволили бы людям этой профессии отображать свои работы на 3D-модели элемента одежды.

Целью работы является создание клиент-серверного приложения, где сервер – Spring Boot приложение [1], а клиент – это веб-страница, управляемая компонентами React [2]. Приложение позволит дизайнерам быстро и удобно делиться своими работами, а также виртуально примерять свои работы на модели.

1. Постановка задачи

Реализовать веб-приложение, которое позволит:

- создавать посты с дизайнами вышивок;
- оставлять комментарии и ставить отметку «Нравится» под постами;
- организовывать хранение дизайнов в папках;
- отображать дизайн вышивок на 3D-модели в области грудной клетки.

2. Анализ предметной области

Чтобы провести анализ предметной области, были выбраны три различные площадки, на которых в данный момент активно собираются люди, занимающиеся не только вышивкой, но и другими видами творчества. Это платформа «Ярмарка Мастеров», сайт «Free Embroidery Design» и группы в социальной сети «ВКонтакте». Конечно же каждая

площадка обладает своими преимуществами и недостатками, но необходимо рассмотреть те критерии, которые важны для целей данной работы. На основе этого была составлена таблица.

Таблица

Сравнение площадок

Площадка	Публикации вышивок	Сохранение файлов вышивок в профиль	3D-модель
Ярмарка Мастеров	+	–	–
Free Embroidery Design	+	+	–
ВКонтакте	+	+	–

Исходя из данных таблицы, можно сделать вывод, что основным недостатком всех аналогов является отсутствие 3D-модели, чем и будет выделяться данная работа на фоне всех рассмотренных площадок.

3. Основная задача

Основная задача, которую решает онлайн-портал для вышивок – это возможность разместить вышивку на 3D-модели. В данном случае модель представляет из себя футболку светло-серого цвета.

Всё решение задачи опирается на выбор фреймворка, который позволяет создавать виртуальную реальность. В данном случае рассматриваются фреймворки React 360 и A-Frame. Проанализировав их преимущества и недостатки, выбор пал в пользу фреймворка A-Frame, так как он является бесплатным, не требует дополнительной установки, работает с HTML и активно набирает популярность [3].

4. Метод решения задачи

Перед тем как перейти к методу решения задачи нужно обратить внимание на то, как пользователь взаимодействует с системой, чтобы получить отображение вышивки на модели. На рис. 1 наглядно продемонстрировано это взаимодействие.

На диаграмме последовательности можно увидеть два объекта: клиент и система. Клиент или по-другому пользователь пытается получить страницу поста. Система на этот запрос отвечает ему HTML-страницей. Получив страницу с постом, клиент может нажать на кнопку «Показать Модель» для того, чтобы увидеть изображение вышивки на ней. Если система при проверке файлов поста нашла хотя бы один файл с расширением jpg/jpeg/png, то она возвращает пользователю 3D-модель

с вышивкой. В противном случае пользователь получит модель без неё. Теперь можно перейти к самому методу решения поставленной задачи.

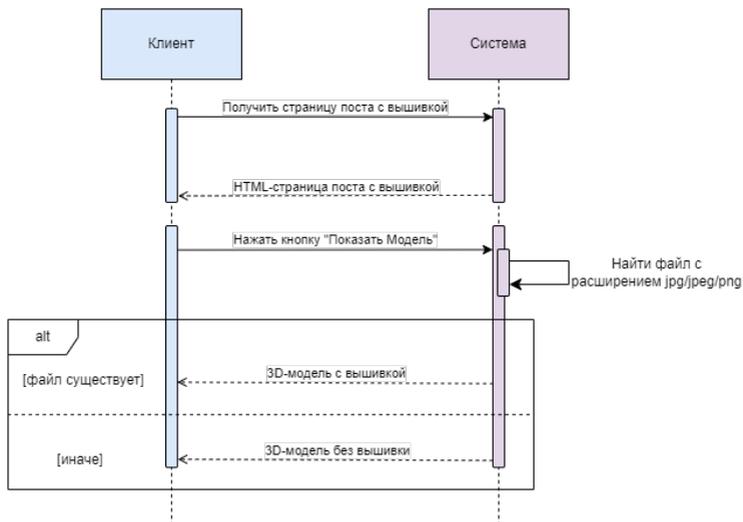


Рис. 1. Диаграмма последовательности

В целом создание любого объекта в фреймворке A-Frame происходит путём добавления тега, как в HTML, только не обычного тега, а специального. В листинге 1 приведен пример тега `a-asset-item`, который определяет ресурс и к которому можно обратиться из другого тега через решетку, указав его `id` [3]. Данный ресурс предоставляет доступ к файлу формата `obj`, который представляет из себя модель футболки.

Листинг 1

Создание ресурса

```

...
<a-asset-item id="object"
              src="/models/t-shirt/Tshirt.obj">
</a-asset-item>
...

```

В листинге 2 тег `a-entity` занимается отображением объекта в виртуальной реальности [3]. Атрибут `obj-model` как раз обращается к ресурсу, где и лежит модель.

Отображение 3D-модели

```
...  
<a-entity id="target"  
  obj-model="obj: #object;"  
  position="0 3 -15"  
  rotation="0 -90 0"  
  scale="10 10 10"></a-entity>  
...
```

Так как в фреймворке A-Frame наложение текстуры происходит на всю модель, как показано на рис. 2, а не конкретную область, то есть на переднюю часть футболки в области грудной клетки, то такое решение не будет удовлетворять поставленной задаче.



Рис. 2. Наложение текстуры на всю 3D-модель

Поэтому следующим этапом решения было продублировать файл модели и методом подбора удалить те полигоны футболки, которые не находятся в области грудной клетки. На рис. 3 изображены по отдельности 3D-модели футболки без текстуры и грудной клетки с текстурой.



Рис. 3. 3D-модели футболки и грудной клетки

Последним этапом является наложение двух объектов друг на друга и получение итогового результата. На рис. 4 можно увидеть итоговую модель.



Рис. 4. Итоговая 3D-модель

Заключение

В результате работы было написано веб-приложение для дизайнеров вышивок, где они могут делиться своими работами и самое главное – виртуально размещать их на элементе одежды. Все поставленные цели были достигнуты.

Приложение имеет дальнейшие перспективы развития: в будущем можно будет реализовать возможность размещать вышивку на любой части элемента одежды, например, на рукавах, на спине; свободно передвигать её по всей модели, делать скриншот и сохранять результат отображения на компьютер.

Исходный код как приложения, так и 3D-модели размещен в публичном доступе [4, 5].

Список литературы

1. Крейг, У. Spring в действии : книга для разработчиков приложений на Java / У. Крейг ; перераб. и доп. – Москва: ДМК Пресс, 2015. – 754 с.
2. Томас, М. React в действии : книга для frontend-разработчиков / М. Томас ; перераб. и доп. – СПб: Питер, 2019. – 368 с.
3. A-Frame [Электронный ресурс] : Официальный сайт фреймворка. – Режим доступа: <https://aframe.io>
4. Git Hub [Электронный ресурс] : Backend-часть проекта. – Режим доступа: <https://github.com/katseliv/portal-for-embroidery>
5. Git Hub [Электронный ресурс] : Frontend-часть проекта. – Режим доступа: <https://github.com/katseliv/portal-for-embroidery-frontend>

Разработка модуля дополнительной отчетности для CRM-системы в виде расширения браузера

С. М. Синяков

Студент бакалавриата

Д. И. Соломатин

Старший преподаватель

Введение

В настоящее время существует множество CRM-систем, которые помогают организациям улучшить процессы продаж и управления клиентами. Одной из таких систем является Параплан – CRM для бизнеса. CRM-система Параплан является одним из наиболее востребованных и популярных решений для автоматизации управления взаимоотношениями с клиентами, в частности Параплан активно используется организациями, работающими в сфере дополнительного образования.

В таких организациях для автоматизации деятельности часто используется только система Параплан и для них является проблемой отсутствие в системе Параплан ряда отчётов, необходимых для анализа и улучшения работы компании. В данной работе будет рассмотрен процесс создания расширения для Параплана, которое добавит недостающие отчеты и упростит работу с системой.

1. Анализ задачи

В настоящий момент в России детские кружки, секции и в целом дополнительное образование могут как финансироваться из бюджетов различного уровня, так и работать на коммерческой основе. Руководителям частных форм дополнительного образования, как и в любой коммерческой организации, необходима финансовая отчетность, чтобы контролировать доходы и расходы, управлять финансами и планировать будущую деятельность.

Но, как уже было сказано, для организации, которые используют для автоматизации деятельности только CRM Параплан, существующей в данной системе финансовой отчетности часто оказывается недостаточно. Для решения этой проблемы в данной работе

рассматривается создание недостающего модуля финансовой отчетности для системы "Параплан".

Из недостающей отчетности в CRM-системе «Параплан» можно выделить 2 наиболее востребованных пользователями отчета – отчет о движении денежных средств и отчет о прибылях и убытках.

Отчет о движении денежных средств – это отчет о транзакциях компании, связанных с ее денежными средствами за определенный период времени. Он показывает изменения в денежных потоках, которые произошли за этот период.

Отчет о прибылях и убытках – это отчет, который дает представление о доходах и расходах компании за определенный период времени. Этот отчет показывает, сколько заработала компания, сколько было потрачено, а также сколько осталось в виде прибыли или убытка.[1]

CRM Параплан предоставляет собой типичное SaaS-решение, соответственно, доработать эту систему локально не представляется возможным, API для взаимодействия с данной системой также не предоставляется. Поэтому было принято решение разработать модуль недостающей отчетности в виде расширения web-браузера.

2. Реализация

Для получения финансовых данных из личного кабинета CRM-системы, учитывая отсутствие публичного API, используется технология *web-скрейпинга (web scraping)*, т.е. извлечения данных из web-страниц CRM Параплан [2]. Для этого используется широко известная JavaScript-библиотека jQuery.

Для автоматизации процесса построения Excel-таблицы с отчетами используется библиотека ejsExcel, одним из главных преимуществ которой является возможность использования Excel-шаблонов для создания документов, с необходимым форматированием и расположением элементов. Пример такого шаблона для одного из ответов приведен на рис. 1.

Единственное, что нужно для формирования готового документа – передать библиотеке необходимые для заполнения шаблона данные в виде JavaScript-объекта. Для рассматриваемого отчета пример таких данных приводится в листинге ниже.

	A	B	C	D	E
1					
2					12.2022
3		Отчет о Прибылях и Убытках			
4	Показатели ↓		Продажа абонементов	Продажа комплектов формы	Итого
5	Общая выручка	<%"#="C8">	<%"#="D8">	0	
6	Средний чек(стоимость одного занятия)	<%~_data_.pro 0			#ДЕЛ/0!
7	Количество проведенных занятий/сделок	<%~_data_.pro 0		0	
8	Выручка	<%"#="C6*C7">	<%"#="D6*D7">	0	
		%>	%>		
	Кoeffициент относительно выручки по направлению 1,2,3	<%"#="C8/\$E\$8">	<%"#="D8/\$E\$8">	0,00%	
9					
10	Переменные расходы	<%"#="CYMM(C11:C29)"%>	<%"#="CYMM(D11:D29)"%>	0	
11	<%forRow v in _data_.variable_costs%><%=v.<%~_data_.variable_costs%><%=v.<%~_data_.variable_costs%>				
29					
30	Маржинальная прибыль	<%"#="C5-C10">	<%"#="D5-D10">	0	
31	Маржинальность, %	<%"#="C30/C5">	<%"#="D30/D5">	#ДЕЛ/0!	
32		%>	%>		

Рис. 1. Пример Excel-шаблона

Листинг

JSON-представление данных для заполнения шаблона отчета

```

_data_ = {
  "profit": [875, 68],
  "variable_costs": [
    {"name": "НДФЛ", "cost": 2362},
    {"name": "начисление зарплаты", "cost": 17600},
    ...
  ],
  "fixed_costs": [
    {"name": "аренда зала", "cost": 40800},
    {"name": "расходы на услуги банка", "cost": 987},
    ...
  ],
  ...
}

```

Здесь:

- variable_costs, fixed_costs – статьи расходов;
- profit – фиксированные числовые данные, которые относятся к выручке;
- и т.д. (приведены не все разделы данных).

В результате будет сформирован отчет, пример которого показан на рис. 2.

A	B	C	D	E
1				
2				5.2023
3	Отчет о Прибылях и Убытках			
4	Показатели	Продажа абонементов	Продажа комплектов формы	Итого
5	Общая выручка	131 250	10 000	141 250
6	Средний чек(стоимость одного занятия)	875	5 000	929
7	Количество проведенных занятий/сделок	150	2	152
8	Выручка	131 250	10 000	141 250
9	Козффициент относительно выручки по направлению 1,2,3	92,92%	7,08%	100,00%
10	Переменные расходы	58 173	6 000	64 173
11	НДФЛ	2362		
12	начисление зарплаты	19700		
13	Реклама, продвижение	30625		
14	взносы с з/п	5486		
15	пошив формы		6 000	
17	Маржинальная прибыль	73 077	4 000	77 077
18	Маржинальность, %	56%	40%	55%
19	Постоянные расходы	74 461	0	74 461
21	услуги связи	0		
22	комиссионное вознаграждение(роялти)	25000		
23	аренда зала	48450		
24	расходы на услуги банка	1011		
26	Store-level EBITDA	-1 384	4 000	2 616
27	Управление	0	0	0
28	Операционная прибыль (EBITDA)	-1 384	4 000	2 616
29	Рентабельность по операционной прибыли, %	-1%	40%	2%
30	Налоги	0	0	0
31	Кредиты только %	0	0	0
32	Амортизация	1 600	0	1 600
34	Чистая прибыль за период	-2 984	4 000	2 616
35	Рентабельность по чистой прибыли, %	-1%	40%	2%
38	Полная себестоимость заказа	875,00	5 000,00	912,07

Рис. 2. Пример сгенерированный с помощью расширения PNL-отчета

Для оформления разработанного решения в виде расширения браузера был создан файл manifest.json, включающий следующие разделы:

- "name": "report_plugin" – название расширения;
- "version": "2.0" – версия издания;
- "description": "модуль финансовой отчетности будет актуальным для бизнеса, использующего CRM-систему «Параплан» в своей деятельности... " – краткое описание, которое включает в себя информацию, полезную для пользователя;
- "manifest_version": 2 – версия манифеста, данный параметр обязателен для указания и помогает правильно интерпретировать манифест;
- "content_scripts" – дает возможность установить скрипты, которые могут быть внедрены на страницы сайтов, заданных в параметрах. С помощью этого параметра можно добавить новые функциональные возможности на страницу сайта.

В результате установки данного расширения в браузере, с ним можно взаимодействовать так же, как и с другими расширениями, что показано на рис. 3.

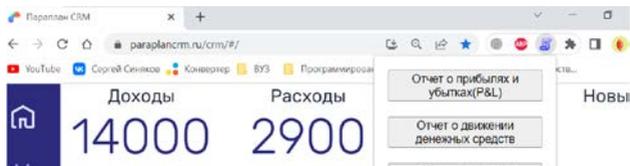


Рис. 3. Всплывающее окно для выбора отчета

Заключение

Разработанный модуль отчетности для CRM Параплан в настоящий момент используется в нескольких организациях в сфере дополнительного образования. В дальнейшем планируется разработка еще нескольких отчетов, в частности отчет о посещаемости обучаемых, отчет налоговых отчислений и др.

Подход в виде разработки расширения браузера может быть использован в случаях, когда необходима доработка закрытых для модификации конечными пользователями web-приложений.

Список литературы

1. Григорьева, Т.И. Финансовый анализ для менеджеров: оценка, прогноз: учебник для вузов / Т. И. Григорьева. – 3-е изд., перераб. и доп. – Москва: Издательство Юрайт, 2019. – 314 с.
2. Крокфорд, Д. JavaScript. Сильные стороны / Д. Крокфорд. – М.: Питер, 2018. – 151 с.

Оптимизация SELECT запросов в СУБД PostgreSQL

И. Ю. Сметанин

Студент бакалавриата

Н. К. Самойлов

Старший преподаватель

Введение

PostgreSQL – это одна из классических систем управления базами данных, которой более 20 лет. Она продолжает активно развиваться и привлекать все больше пользователей. Несмотря на свой возраст, PostgreSQL постоянно совершенствуется, чтобы решить проблемы, связанные с устаревшими схемами дизайна, которые были заложены при ее создании.

На текущий момент при разработке активно используется микросервисная архитектура, когда приложение делится на небольшие компоненты с базами данных несложными по структуре, но большими по объёму хранения данных, поэтому к ней важно писать оптимизированные запросы, выполняющиеся быстро.

В данной работе будут рассмотрены методики, с помощью которых СУБД может более эффективно использовать ресурсы процессора и памяти при выполнении запросов.

1. Этапы обработки запроса

В PostgreSQL существуют следующие стадии обработки запроса от клиента:

1. Разбор (parse). Запрос, приходящий в текстовом виде на сервер базы данных, проверяется на синтаксическую корректность и наличие опечаток. После этого по текстовому запросу строится синтаксическое дерево запроса, в котором содержится информация об объектах, к которым обращается запрос, информация о которых берётся из системных каталогов PostgreSQL. При наличии ошибок возвращается исключение.

2. Переписывание (rewrite). Дерево запроса дополняется информацией, согласно правилам, содержащимся в PostgreSQL, а также раскрываются запросы, лежащие за представлениями. На этом этапе также может выполняться разбор запроса.

3. Планирование (plan). Планировщик перебирает разные планы выполнения запроса, при этом он может менять порядок соединения таблиц, способы доступа к данным, методы соединения, чтобы получить оптимальный план с наименьшей стоимостью. Оценка стоимости зависит от типов узлов, а также от числа строчек, которые требуется обработать.

4. Выполнение (execute). Происходит выполнение плана, в результате получается набор строк, которые сразу отдаются клиенту.

2. Планирование запроса

Оптимизация запросов в PostgreSQL основана на стоимости операций (cost). Данные затраты являются безразмерными величинами, которые нужны для оценки запросов. Оценка стоимости каждого узла в дереве зависит от типа узла, а также от числа строчек, которые требуется обработать.

Селективностью выборки называется доля выбираемых строк. Для запроса с выборкой 10 из 100 строк, селективность равна 0.1.

Кардинальностью выборки называется итоговое количество выбранных строк. Для запроса, выбирающего 10 из 100 строк, кардинальность равна 10 [1].

Если планировщик имеет статистику для данной таблицы, он может правильно вычислить кардинальность, после этого вычислить стоимость и выбрать план с наименьшей стоимостью.

В PostgreSQL существует три вида затрат: start-up, run и total. В зависимости от способа получения данных планировщик будет сортировать планы по определённой стоимости start-up или run. Total это общая сумма start-up и run затрат. Независимо оцениваются только затраты на запуск или исполнение. Команда EXPLAIN показывает как начальные, так и общие затраты на каждую операцию.

План читается снизу вверх, он изображён на рис. 1. В данном примере выполняется последовательное чтение таблицы (узел Seq Scan) с применением фильтрации по полю id. Подготовительные затраты для получения первой строки start-up равны 0 у.е. Общие затраты total займут 170 у.е. Далее выполняется сортировка по полю data (узел Sort).

План выбирается в зависимости от типа запроса. При обычном SELECT запросе выбирается план по минимальной общей стоимости, при запросе курсора план выбирается по времени получения первой строки [2].

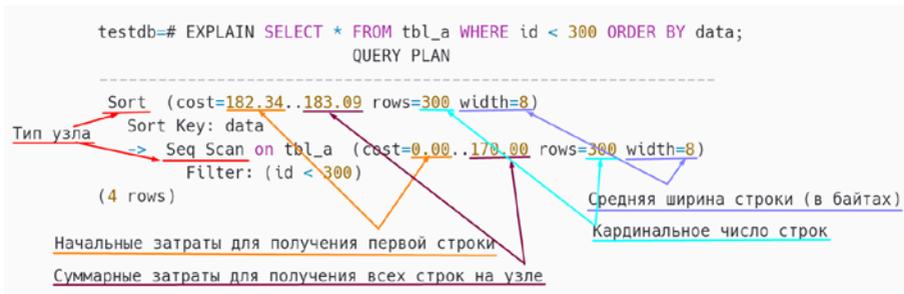


Рис. 1. План выполнения SELECT запроса

3. Типы узлов EXPLAIN запроса

Существуют разные способы доступа к данным, которыми может воспользоваться планировщик при построении плана запроса.

- Последовательное сканирование (Sequential scan). Этот способ доступа, который читает последовательно все страницы с данными, которые относятся к условию выборки. Эффективно при полном чтении или значительной части данных из таблицы.

- Индексное сканирование (Index scan). Этот способ доступа, при котором поиск осуществляется по индексной структуре. Эффективно при условии, если селективность достаточно высокая, так как для каждой записи страница читается заново. Из-за того, что данные в индексе отсортированы, не требуется дополнительных затрат на сортировку.

- Полное индексное сканирование (Index only scan). Имеет те же условия, что и индексное сканирование, за исключением того, что не требуется доступ к таблице, так как все данные содержатся в индексе. При этом может происходить обращение к таблице для проверки актуальности данных в индексе.

- Сканирование по битовой карте (Bitmap Index/Heap Scan). Строит битовую карту по индексу (Bitmap Index Scan) и последовательно сканирует таблицу (Bitmap Heap Scan), при этом каждая страница с данными читается ровно 1 раз. Необходимы затраты на сохранение в оперативной памяти битовой карты. Эффективно, если число записей в выборке примерно равно половине от размера таблицы.

Соединение таблиц всегда осуществляется попарно, поэтому порядок соединения важен для выбора оптимального плана. Существуют следующие виды соединений таблиц.

- Вложенный цикл (Nested loop). Данный метод для каждой строки из левой таблицы производит поиск подходящей строки в правой

таблице. Если размер левой таблицы небольшой, временная сложность является линейной. Если размер таблиц примерно равен, сложность является квадратичной. Метод не требует дополнительных затрат по памяти. Эффективно для небольших выборок и в случае, когда доступ для правой таблицы осуществляется по индексу.

– Соединение хэшированием (Hash join). Данный метод строит хэш таблицу по ключу соединения для левой таблицы. Временная сложность зависит от размеров обеих таблиц и является линейной. Требуются дополнительные затраты на создание хэш таблицы. Эффективно для большого числа строк.

– Соединение слиянием (Merge join). Данный метод предварительно сортирует обе таблицы и методом двух указателей производит слияние двух таблиц. Имеет линейную временную сложность, если данные уже отсортированы. Эффективно для большого числа строк.

4. Способы оптимизации запросов

Разработчик должен руководствоваться следующими критериями для оценки оптимальности запроса.

1. Проверка, что созданный индекс используется. Для этого в EXPLAIN запросе должен присутствовать узел INDEX SCAN.

2. Явно описывать столбцы, которые требуются в итоговой выдаче, не использовать * после SELECT.

3. При ошибке планировщика в определении кардинальности для условий, создавать статистику для группы таких столбцов. Она представлена на рис. 2.

```
testdb=# EXPLAIN SELECT * FROM flights
WHERE flight_no = 'PG0007' AND departure_airport = 'VKO';
          QUERY PLAN
-----
Bitmap Heap Scan on flights (cost=10.49..816.84 rows=14 width=63)
  Recheck Cond: (flight_no = 'PG0007'::bpchar)
  Filter: (departure_airport = 'VKO'::bpchar)
-> Bitmap Index Scan on flights_pkey (cost=0.00..10.49 rows=276 width=0)
    Index Cond: (flight_no = 'PG0007'::bpchar)
(5 rows)
```

Рис. 2. Запрос с поиском всех авиарейсов с определённым номером рейса и аэропортом отправления

Оценка оказывается заниженной. Номер рейса однозначно определяет аэропорт отправления и условие на аэропорт не может уменьшить выборку, если аэропорт указан верно. Добавление отдельной

статистики на два этих столбца исправит заниженную оценку. Результат показан на рис. 3.

```
testdb=# CREATE STATISTICS flights_dep(dependencies)
ON flight_no, departure_airport FROM flights;
testdb=# ANALYZE flights;
testdb=# EXPLAIN SELECT * FROM flights
WHERE flight_no = 'PG0007' AND departure_airport = 'VK0';
QUERY PLAN
-----
Bitmap Heap Scan on flights (cost=12.09..1225.67 rows=473 width=63)
  Recheck Cond: (flight_no = 'PG0007'::bpchar)
  Filter: (departure_airport = 'VK0'::bpchar)
-> Bitmap Index Scan on flights_pkey (cost=0.00..11.97 rows=473 width=0)
    Index Cond: (flight_no = 'PG0007'::bpchar)
(5 rows)
```

Рис. 3. Добавление статистики на два столбца, изменение статистики таблицы полётов, и повторение запроса на поиск

Теперь оценка кардинальности считается верно.

4. Учитывание количества пустых значений в поле при создании индекса по нему. Статистика по полю представлена на рис. 4.

```
testdb=# SELECT * FROM pg_stats WHERE tablename = 'pgconf' and attname =
'fk_id';
```

tablename	pgconf
attname	fk_id
null_frac	0.92943335
n_distinct	-0.070566654
most_common_vals	
most_common_freqs	
correlation	0.0095442245

Рис. 4. Запрос для получения статистики поля определённой таблицы

Если при получении статистики поля видно, что количество его пустых значений близко к единице, это означает, что для уменьшения размеров индекса можно добавить к нему условие отсутствия пустых значений WHERE fk_id IS NOT NULL. Время исполнения операций, задействующих поле этого индекса, не изменится, а размер сократится в разы [3].

5. Проверка наличия индекса у внешнего ключа, так как он не создаётся автоматически.

6. Если запрос требует вывода небольшого числа столбцов и содержит небольшое число условий, создание или изменение

существующего индекса, путём добавления туда столбцов из запроса при помощи INCLUDE, которые не участвуют в условии выборки, и добавление условия в индекс.

Из статистики поля state (рис. 5), видно, что оно чаще всего содержит значение «обработано». Если не предполагается получение записей по другим значениям, можно добавить это условие в индекс, а также добавить ссылку на поле amount через ключевое слово INCLUDE, чтобы работал INDEX ONLY SCAN [4].

Заключение

В результате проделанной работы были выявлены и реализованы эвристики, помогающие в написании оптимальных SELECT запросов, в виде библиотеки. Дальнейшим развитием является её внедрение в системы сборки Java приложений Maven/Gradle.

```
SELECT sum(amount)
FROM pgconf
WHERE state = 'обработано' AND
      item = 'анельсин' AND
      created_at between '2021-10-23 00:00' AND '2021-10-24 00:00';

CREATE index special ON pgconf(created_at, item)
INCLUDE (amount) WHERE state = 'обработано';

testdb=# SELECT * FROM pg_stats WHERE tablename = 'pgconf' and attname
= 'state';

attname          | state
-----+-----
null_frac        | 0
n_distinct       | 3
most_common_vals | {обработано,ождает,ошибка}
most_common_freqs| {0.89863336,0.09996667,0.0014}
correlation       | 0.81656545
```

Рис. 5. Пример добавления дополнительного поля и условия в индекс

Список литературы

1. Рогов, Е. В. PostgreSQL 15 изнутри / Е.В. Рогов. – М.: ДМК Пресс, 2023. – 662 с.
2. Ригс, С. Администрирование PostgreSQL 9. Книга рецептов. / С. Ригс, Х. Кросинг. – М.: ДМК Пресс, 2012. – 368 с.
3. Домбровская, Г. Оптимизация запросов в PostgreSQL. Полное руководство по созданию эффективных запросов / Г. Домбровская, Б. Новиков, А. Бейликова. – М.: ДМК Пресс, 2022. – 278 с.
4. Le, Q. H. Developing Modern Database Applications with PostgreSQL / Q.H. Le, M. Diaz. – Packt Publishing, 2021. – 440 p.

Двухфакторная аутентификация пользователя с использованием системы распознавания лиц

Э. А. Стешенко

Студент бакалавриата

А. В. Швырева

Ассистент

Введение

С целью защиты пользовательских данных от несанкционированного доступа во многих системах используется парольная защита, которая обладает рядом недостатков, таких как недостаточная стойкость, простота взлома, возможность передачи третьим лицам.

Для обеспечения большей надежности активно развиваются методы двухфакторной аутентификации.

Под аутентификацией будем понимать процедуру проверки подлинности введенных данных о пользователе. Двухфакторная аутентификация заключается в вводе пользователем двух независимых факторов аутентификации, например, пароль и код подтверждения.

Целью данной работы является разработка системы двухфакторной аутентификации с использованием биометрических данных пользователя, а именно системы распознавания лиц.

1. Парольные методы защиты информации

Аутентификация по паролю подразумевает ввод пользователем уникальных логина и пароля. Далее эти данные сверяются с хранящимися в системе. После проверки возможны три разных варианта ответа системы:

1. Совпадения данных не обнаружено – данный пользователь не зарегистрирован в системе, также возможен неправильный ввод логина.
2. Введен неправильный пароль.
3. Данные совпали – успешный вход в систему.

При первых двух ответах система информирует пользователя с указанием причины.

Данные в системе принято хранить в виде базы данных, в которой уникальным идентификатором является логин, а каждая строка – информация об одном пользователе.

Парольной защите характерны следующие уязвимости:

- отсутствие проверки личности вводящей логин и пароль;
- возможность подбора логина и пароля;
- возможность хищения информации из базы данных и ее дальнейшее использование;
- возможность использования уязвимостей в системе передачи информации от пользователя к запросу к базе данных;
- возможность получения пароля из другого ресурса.

Усиление парольной защиты имеет несколько направлений. Первое связано с уменьшением информации доступной неавторизованному пользователю, например, указание в каком именно поле ввода допущена ошибка.

Второе направление защиты связано с увеличением уровня безопасности системы. Под этим подразумевается усиление надежности хранения данных, например, отказ от хранения паролей в открытом виде.

Однако для полноценной защиты системы от проникновения злоумышленника недостаточно одного фактора защиты, поэтому следует добавить еще один слой в систему безопасности. Подобным слоем может стать биометрическая аутентификация.

2. Методы и алгоритмы идентификации лиц

Рассмотрим обобщенный алгоритм процесса распознавания лиц, IDF0 диаграмма которого представлена на рис. 1 и 2.

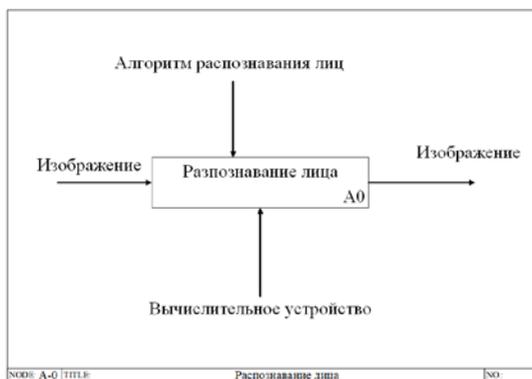


Рис. 1. Диаграмма IDEF0 первого уровня «Распознавание лица»

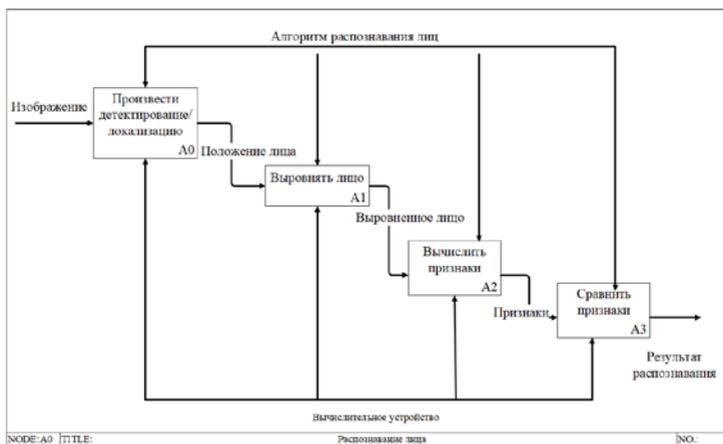


Рис. 2. Диаграмма IDEF0 второго уровня «Распознавание лица»

1. Детектирование и локализация заключается в подаче на вход изображения или кадра видео, на котором происходит выделение лица.

2. Выравнивание лица. Производится геометрическое и яркостное выравнивание изображения лица.

3. Вычисление признаков. Алгоритм вычисляет необходимые для распознавания лица признаки.

4. Сравнение признаков. На данном этапе и происходит распознавание лица путем сравнения вычисленных признаков с заложенными в базу данных эталонами.

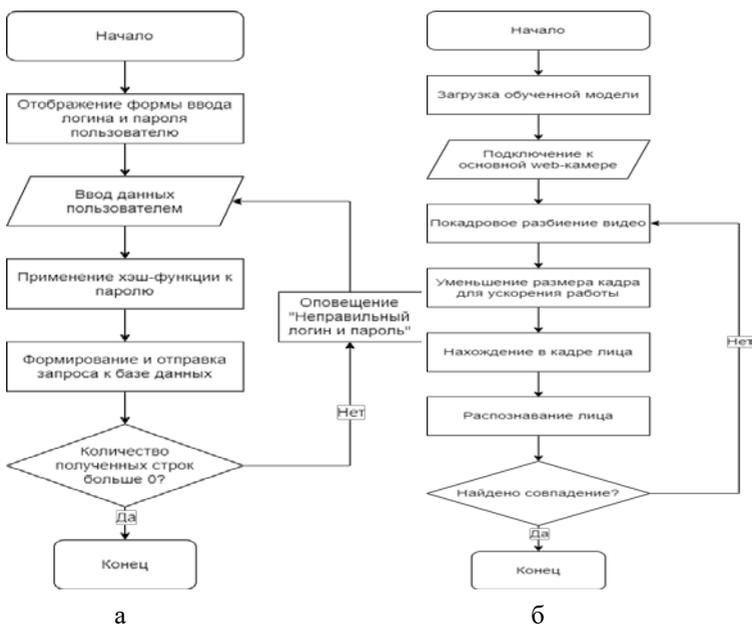
К наиболее известным методам распознавания лиц относятся:

- Метод гибкого сравнения на графах.
- Алгоритмы с использованием нейронных сетей.
- Скрытые марковские модели.
- Метод главных компонент.
- Активные модели внешнего вида.

В качестве метода для реализации был выбран алгоритм с использованием нейронных сетей, так как он имеет достаточную точность и различающую способность при относительной легкости реализации.

3. Разработка алгоритма двухфакторной аутентификации с использованием системы распознавания лиц

Алгоритм двухфакторной аутентификации включает в себя два последовательных блока проверок и представлен на рис. 3.



а – парольная защита, б – система распознавания лиц

Рис. 3. Блок-схемы алгоритмов двухфакторной аутентификации

Блок парольной защиты представляет собой получение и сравнение введенных пользователем логина и пароля после применения хэш-функции, с записанными в базу данных значениями.

Система распознавания лиц в свою очередь состоит из двух частей: предобработки полученного изображения и распознавания лица.

Алгоритм мониторинга нахождения авторизованного пользователя перед устройством после успешной аутентификации представлен на рис. 4.



Рис. 4. Алгоритм проверки пользователя после аутентификации

4. Разработка и тестирование программного обеспечения

При разработке программного обеспечения использовался язык Python и библиотеки face_recognition, dlib, cv2, sklearn, tkinter. Для распознавания лиц использовалась предобученная сверточная нейронная сеть Face Recognition, архитектура которой представлена на рис. 5.

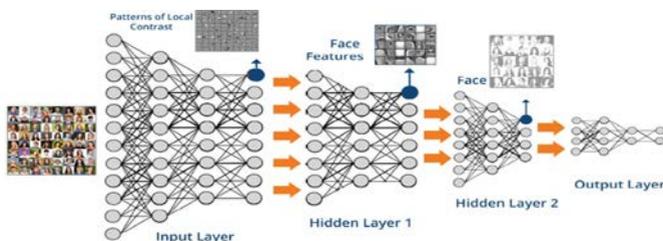


Рис. 5. Архитектура сверточной нейронной сети для распознавания лиц

Интерфейс разработанного программного обеспечения представлен на рис. 6 и 7.

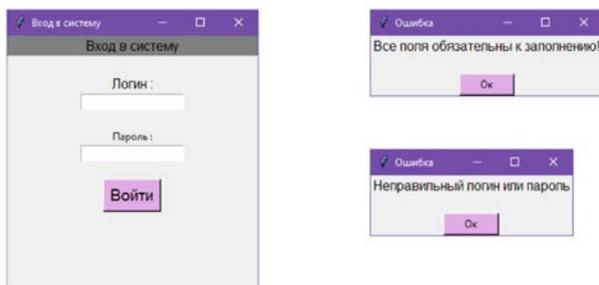


Рис. 6. Интерфейс парольной защиты

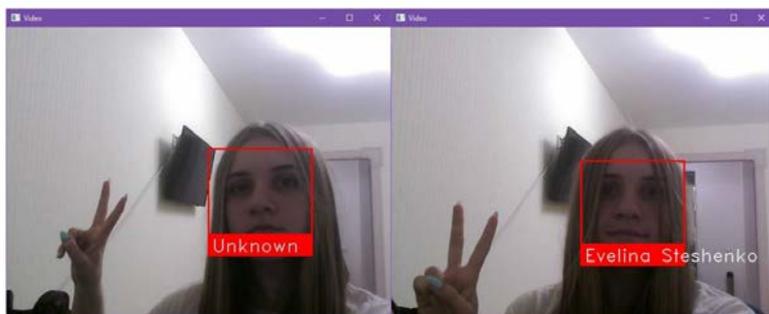
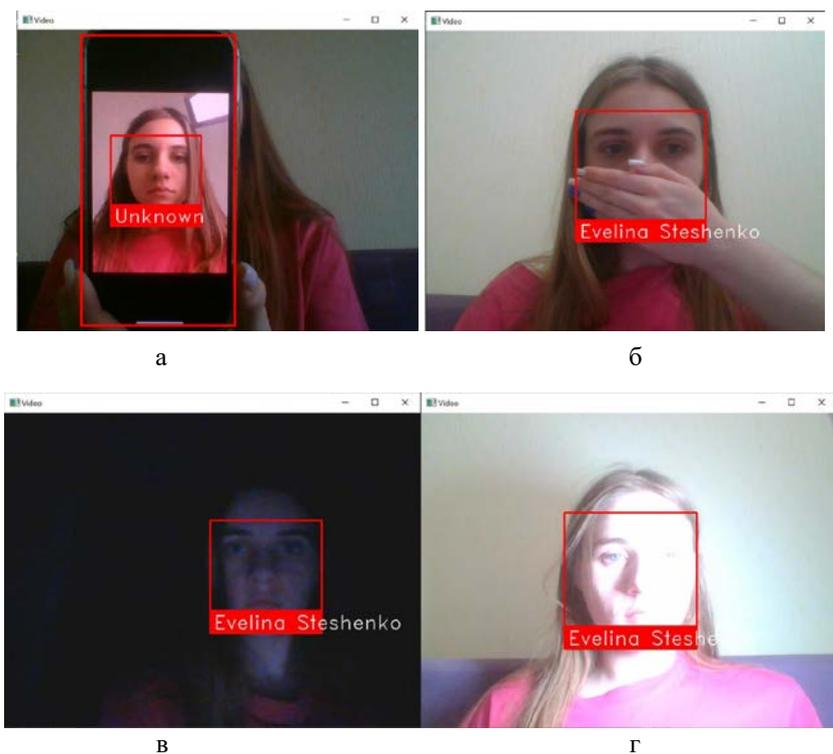


Рис. 7. Интерфейс системы распознавания лиц

Реализованная программа представляет собой форму ввода логина и пароля, при успешном вводе которых появляется окно с изображением веб-камеры и системой распознавания лиц.

При тестировании парольной защиты система оказалась устойчива к уязвимостям: взлом хэша пароля, SQL-инъекции.

Система распознавания лиц устойчива к: яркому освещению, недостаточному освещению, неполной видимости лица, использованию статичных изображений (рис. 8).



а – проверка на использование фотографии, б – проверка при неполной видимости лица, в – проверка при недостаточном освещении, г – проверка при ярком освещении

Рис. 8. Тестирование системы распознавания лиц

Заключение

В данной статье рассмотрена задача разработки программного обеспечения двухфакторной аутентификации пользователя с системой распознавания лиц в реальном времени. Был рассмотрен обобщенный алгоритм распознавания лиц в биометрических системах, а также предложен алгоритм двухфакторной аутентификации пользователя. В разработанном программном обеспечении для распознавания лиц использовалась предобученная нейронная сеть Face Recognition, которая показала свою устойчивость к проблемам освещения и открытости лица при тестировании.

Список литературы

1. Сравнительный анализ двухфакторной аутентификации [Электронный ресурс]. – Режим доступа: <https://cyberleninka.ru/article/n/sravnitelnyy-analiz-dvuhfaktornoy-autentifikatsii/viewer>
2. Алгоритмы на графах [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/post/65367/>
3. Биометрия [Электронный ресурс]. – Режим доступа: <https://www.onespan.com/ru/topics/biometrics>
4. Face Recognition by Elastic Bunch Graph Matching [Электронный ресурс]. – Режим доступа: <https://www.face-rec.org/algorithms/ebgm/wisfelkrue99-facerecognition-jainbook.pdf>
5. Нейронные сети для начинающих [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/post/312450/>
6. Гулятьева, Т.А. Применение скрытых марковских моделей с одномерной топологией к задаче распознавания лиц / Т.А. Гулятьева // Российская научно-техническая конф. «Информатика и проблема телекоммуникаций». – Новосибирск, 2006. – Т. 1. – С. 150-154.
7. Анализ существующих подходов к распознаванию лиц [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/company/synesis/blog/238129/>
8. Face Alignment Using Active Shape Model And Support Vector Machine [Электронный ресурс]. – Режим доступа: <https://arxiv.org/ftp/arxiv/papers/1209/1209.6151.pdf>
9. Документация Face Recognition [Электронный ресурс]. – Режим доступа: https://github.com/ageitgey/face_recognition/tree/55b5c136292dcd4a1f5953f3eb3181235086efab

SaaS-решение для геймификации сервисов

Д. А. Тишанский

Студент бакалавриата

Д. И. Соломатин

Старший преподаватель

Введение

В современном мире информационных технологий пользователи сталкиваются с огромным разнообразием сервисов и приложений, и отсутствие мотивации пользоваться ими становится все более актуальной проблемой. В рамках проведенных исследований выявлено, что свыше половины пользователей приложений и сервисов прекращают их использование в течение первых нескольких дней после установки, а более восьмидесяти процентов отказываются от них в первые две недели [1].

Одним из потенциальных решений данной проблемы является геймификация – технология, позволяющая внедрить игровые элементы в контекст, не связанный с играми [2]. Эти элементы включают в себя рейтинги, испытания, турнирные таблицы, вознаграждения и т.д. Внедрение игровых элементов в сервисы и приложения делает их более привлекательными и интересными для пользователей, которые получают возможность заработать награды и достижения, выполняя определенные задачи, что стимулирует их продолжать использовать сервис [2].

1. SaaS-решение

Однако разработка собственных игровых механик зачастую является затратным и трудоемким процессом, требующим значительных ресурсов со стороны компаний. В качестве альтернативы может быть предложено SaaS-решение, которое предоставит организациям инструменты и функциональность для внедрения игровых механик в их собственные сервисы и приложения.

Организация, являясь клиентом данной системы, получает доступ к платформе геймификации, которая предлагает гибкий набор инструментов для настройки игровых механик. С помощью этой платформы разработчики могут определить различные виды рейтингов,

испытаний и достижений, а также задать правила и условия их получения.

Конечные пользователи, в свою очередь, взаимодействуют с сервисами и приложениями клиентов системы, в которых внедрена геймификация. Они получают доступ к различным игровым элементам, таким как рейтинги, испытания и достижения, которые были сконструированы организацией в рамках данной системы для своих сервисов.

2. Основные сущности системы

Геймификация включает в себя несколько ключевых элементов, способных реализовать игровые механики вне игрового контекста. Среди них основными для данной системы были выбраны следующие сущности.

1. События представляют собой сигналы о выполненных действиях пользователей в рамках клиентского сервиса. Они действуют как элементарные строительные блоки геймификации, отражая важные этапы взаимодействия пользователя.

2. Рейтинги представляют собой количественные характеристики, позволяющие пользователям сравнивать свои результаты с другими участниками. Они служат не только инструментом оценки прогресса, но и стимулируют пользователей к улучшению своих результатов и конкуренции с другими участниками [3].

3. Испытания предоставляют пользователям возможность участвовать в соревнованиях и достигать определенных целей. Эти задачи могут быть структурированы по уровням сложности или разделены на разные этапы, предоставляя постепенно возрастающие вызовы для пользователя. Испытания предоставляют дополнительный стимул к активности, поскольку достижение конкретных целей становится ценной наградой для пользователей [3].

4. Вознаграждения являются ещё одним способом мотивации пользователей через геймификацию. Они представляют собой конкретные призы или поощрения, которые пользователи могут получить за успешное выполнение определенных задач, и также являются мощным стимулом для участия и вовлечения пользователей в сервис [3].

3. Архитектура

Архитектура системы, представленная на рис. 1 включает в себя следующие компоненты. Клиентский сервис имеет доступ к SDK или публичному API, которые обеспечивают взаимодействие с игровыми сущностями. С помощью этих инструментов клиентский сервис может

получать информацию о пользовательских рейтингах, испытаниях, а также описания самих сущностей и их игровых механик, которые хранятся в базе данных.

Важной частью взаимодействия с системой является обработка событий, поступающих от клиентского сервиса. События передаются в систему через шлюз и накапливаются в брокере сообщений. Затем они передаются обработчику событий, который применяет правила геймификации к рейтингам, испытаниям и достижениям, в зависимости от конкретного события.

Окончательно, обработанное событие и данные о сущностях сохраняются в отдельном хранилище, которое не только собирает данные, но также предоставляет аналитические возможности для изучения этих событий и игровых сущностей.

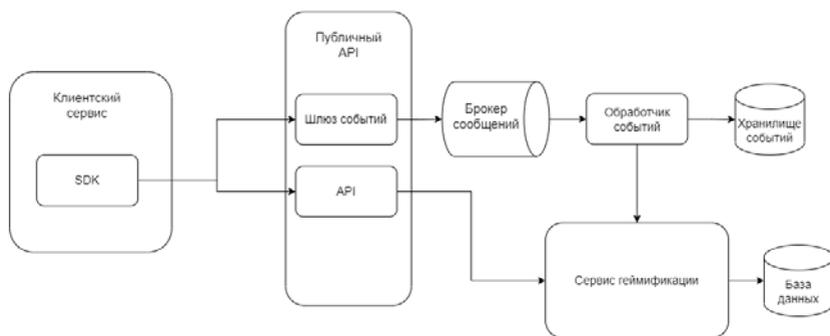


Рис. 1. Архитектура сервиса

4. События

Событие представляет собой определенное действие, выполненное пользователем в клиентском сервисе, которое затем передается в систему геймификации. Каждое событие всегда связано с конкретным пользователем и имеет определенный тип, который указывает на характер этого события. События создаются из административной панели системы. На рис. 2 показаны события, созданные в личном кабинете сервиса геймификации для клиентского интернет-магазина, – событие "Покупка", которое описывает действие покупки товара в интернет-магазине, и "Возврат" соответствует возврату товара пользователем.

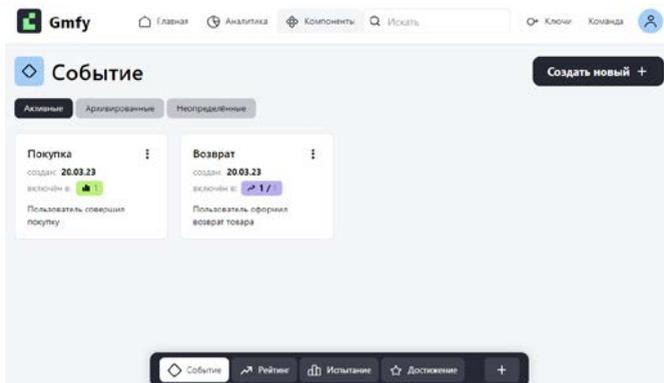


Рис. 2. Список событий, созданных для клиентского интернет-магазина

5. Рейтинги

Рейтинг является количественной характеристикой, которая отражает поведение пользователя. Значения пользовательских рейтингов изменяются в зависимости от действий пользователя и определенных правил рейтинга, которые конструируются на основе событий в административной панели системы. Например, в контексте интернет-магазина может существовать рейтинг "Покупатель", для которого установлены следующие правила.

Событие "Покупка" отражает факт совершения пользователем покупки, и в результате его рейтинг покупателя увеличивается на 5 единиц. Событие "Возврат", в свою очередь, указывает на возврат товара, и по данному событию рейтинг покупателя уменьшается на 1 единицу. На рис. 3 представлен интерфейс, который видит конечный пользователь для ознакомления со своими рейтингами в интернет-магазине клиента.

6. Испытание

Испытание представляет собой качественную характеристику поведения пользователя и включает несколько шагов, каждый из которых может иметь свое значение и место в структуре испытания. Шаги формируются на основе событий, которые отражают определенные действия пользователя. Например, испытание "Первая покупка" состоит из двух шагов: первым шагом является пополнение баланса кошелька, а вторым шагом – совершение покупки. При успешном выполнении этих двух шагов испытание считается пройденным.

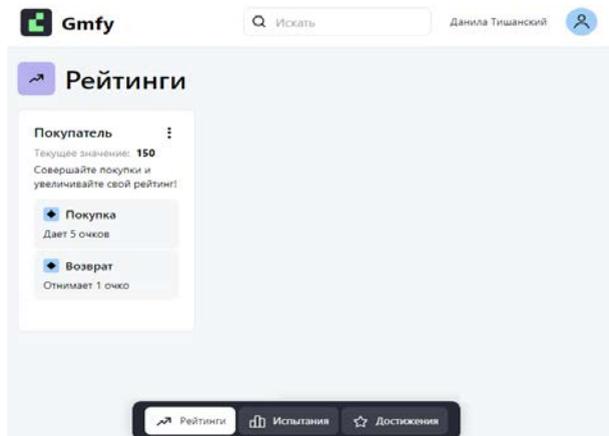


Рис. 3. Рейтинг “Покупатель” пользователя

На рис. 4 в интерфейсе пользователя интернет-магазина представлено испытание "Первая покупка" с необходимыми шагами и прогрессом данного пользователя.

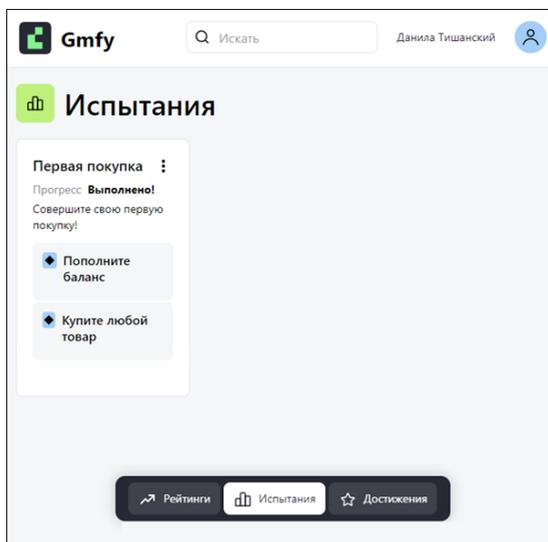


Рис. 4. Прогресс испытания “Первая покупка” у пользователя

7. Достижение

Достижение представляет собой конкретное вознаграждение, которое пользователь получает при выполнении определенных условий. Условия получения достижения определяются логическим выражением, операндами которого выступают рейтинги и испытания. Например, достижение "Шопоголик" может быть получено пользователем в случае, если он успешно выполнил испытание "Первая покупка" на 100% и имеет рейтинг покупателя, превышающий 50 единиц. Т.к. с каждой покупкой рейтинг покупателя повышается на 5 единиц, то для получения достижения нужно совершить минимум 10 покупок, что и видит конечный пользователь в условиях получения достижения (рис. 5).

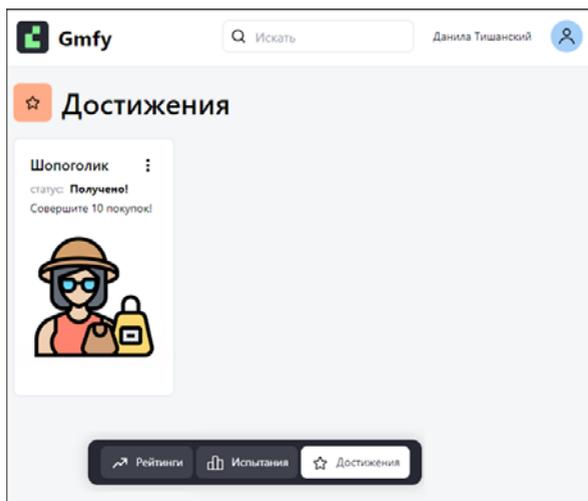


Рис. 5. Достижение “Шопоголик” для пользователя

8. Аналитика

Таким образом, указанные выше сущности играют важную роль в конструировании игровых механик для клиентского сервиса и предоставляют возможность для аналитики действий пользователей. Например, система может предоставить несколько графиков, отражающих динамику рейтинга "Покупатель". Эти графики позволяют проанализировать распределение событий в рейтинге по дням и часам, а также идентифицировать топ пользователей с наивысшим рейтингом.

При изучении распределения событий по часам (рис. 6) можно заметить, что наиболее активные покупки осуществляются в

промежуток времени с 13:00 до 16:00. Такая информация может быть использована, например, для проведения акций или специальных предложений в указанный период времени с целью стимулирования продаж.

Проведение аналитики на основе этих данных позволяет более эффективно понимать поведение пользователей и адаптировать стратегии маркетинга и взаимодействия с клиентами.



Рис. 6. Распределение событий по часам

Заключение

В итоге предлагаемое решение на основе архитектуры SaaS представляет собой универсальный сервис для геймификации клиентских сервисов. Данное решение обеспечивает возможность настройки собственных игровых механик со стороны клиентов, без необходимости их реализации внутри собственной системы. Также решение позволяет проводить аналитику действий пользователей на основе игровых сущностей и, при необходимости, корректировать поведение пользователей путем изменения различных игровых механик.

Список литературы

1. Retention rate on day 30 of mobile app installs worldwide in 3rd quarter 2022, by category [Электронный ресурс]. – Режим доступа : <https://www.statista.com/statistics/259329/ios-and-android-app-user-retention-rate/>
2. Zichermann, G. Gamification by Design: Implementing Game Mechanics in Web and Mobile Apps / G. Zichermann, C. Cunningham. – 1st ed. – Cambridge: O'Reilly Media, Inc., 2011. – 182 p.
3. What is Gamification? [Электронный ресурс]. – Режим доступа : <https://www.gamify.com/what-is-gamification>

Влияние эпохи анализа и децимации на значения коэффициентов корреляции электроэнцефалограммы.

Р. А. Токарев

Студент бакалавриата

Е. В. Жидовленков

Студент бакалавриата

Я. А. Туровский

Профессор

Введение

В ходе различных экспериментов, связанных, в основном, с нейрофизиологией, необходимо записывать электроэнцефалограмму, представляющую собой исследование головного мозга посредством записи с него электрических импульсов. Одним из методов анализа электроэнцефалограммы является подсчет коэффициентов корреляции между парами каналов, записанных с разных отделов мозга, показывающих степень связи этих отделов [1, 2].

В общем случае известно, что на значение коэффициентов корреляции влияют значения децимации и эпохи анализа. Эпохой анализа электроэнцефалограммы является сам интервал времени, в котором производится обработка ЭЭГ-сигнала. В настоящее время не существует достаточно достоверных моделей ЭЭГ-сигнала, следовательно, задачей исследования являлось определение наличия предпосылок выбора определенных параметров для подсчета коэффициентов корреляции на основе эмпирического сигнала.

1. Используемые формулы и основные параметры.

В исследовании использовались коэффициенты корреляции по Спирмену. Они считаются с помощью следующей формулы:

$$r = 1 - 6 \frac{\sum d^2}{n^3 - n},$$

где d^2 – квадраты разностей между рангами; N – количество признаков, участвовавших в ранжировании.

Основными параметрами для подсчета коэффициентов корреляции по Спирмэну являются эпоха анализа, шаг и размер децимации. Децимация нужна для сокращения исходного сигнала в N раз. Шаг (или перекрытие) необходим для установления связи между текущей эпохой анализа и предыдущей. Сама по себе эпоха анализа является ничем иным, как областью подсчета коэффициентов [3–5].

2. Влияние параметров подсчета коэффициентов корреляции на результат.

Для проведения качественного эксперимента необходимо учитывать то, насколько статистически значимы различия коэффициентов корреляции при разных параметрах.

Для нахождения зависимости между этими характеристиками на основе экспериментальных данных была получена таблица, представленная ниже. В ней показаны средние значения коэффициентов корреляции данных ЭЭГ с 32 испытуемых со значениями децимации 1,5 и 10 и размерами эпохи анализа 0,1, 0,25, 0,5 и 1 секунда. Верхняя часть таблицы была создана на основе фоновой ЭЭГ, записанной без каких-либо стимуляций, часть ниже – на основе ЭЭГ со звуковыми стимуляциями, подаваемыми в случайном порядке, третья – со стимуляциями, воспроизводимым при попадании значений коэффициентов корреляции в верхний квартиль, а последняя – в нижний.

Таблица

Коэффициенты корреляции одной и той же пары каналов ЭЭГ в зависимости от децимации и эпохи анализа исходного сигнала.

Фоновая ЭЭГ:	Эпоха анализа			
Децимация	0,1с	0,25с	0,5с	1с
1	0.713	0.700	0.686	0.670
5	0.686	0.655	0.644	0.636
10	0.669	0.644	0.636	0.633
Рандомные стимуляции:	Эпоха анализа			
Децимация	0,1с	0,25с	0,5с	1с
1	0.750	0.721	0.697	0.675
5	0.697	0.668	0.661	0.652
10	0.675	0.661	0.652	0.638
Стимуляции на увеличение коэффициентов корреляции:	Эпоха анализа			
Децимация	0,1с	0,25с	0,5с	1с
1	0.773	0.753	0.735	0.705
5	0.735	0.701	0.687	0.676
10	0.705	0.687	0.676	0.673

Стимуляции на уменьшение коэффициентов корреляции:	Эпоха анализа			
	0,1с	0,25с	0,5с	1с
Децимация				
1	0.776	0.753	0.729	0.707
5	0.729	0.697	0.679	0.672
10	0.707	0.679	0.672	0.668

Исходя из данных этой таблицы, видно, что различия между средними значениями коэффициентов статистически незначимы. Таким образом, с точки зрения обработки сигналов, нет предпосылок ни к выбору эпохи анализа, ни к выбору децимации. Следовательно, децимация, в свою очередь, определяется ресурсоемкостью вычислений, а эпоха анализа – нейрофизиологическими аспектами.

3. Заключение

Данная статья посвящена исследованию зависимости коэффициентов корреляции от эпохи анализа и децимации и нахождению предпосылок для выбора определенных параметров. В ходе обработки данных, полученных в ходе эксперимента, было установлено, что фактор децимации и размера эпохи анализа значимо не влияет на полученные значения коэффициентов корреляции. Таким образом, выбор децимации и эпохи анализа определяется исключительно ресурсоемкостью системы и нейрофизиологическими аспектами.

Список литературы

1. Андреевко, А. С. Обзор методов математической обработки электроэнцефалограмм / А. С. Андреевко // Известия ТРТУ. – 2000. – № 4(18). – С. 111–112.
2. Бахарев, Б. В. Корреляционно-декрементный анализ энцефалограмм на основе моделирования биопотенциалов коры головного мозга / Б. В. Бахарев // Биофизика. – 2016. – № 61(4). – С. 793–798.
3. Туровский, Я. А. Обработка сигнала электроэнцефалограммы на основе анализа частотных зависимостей и вейвлет-преобразования / Я. А. Туровский, С. Д. Кургалин, А. А. Вахтин // Биомедицинская радиоэлектроника. – 2012. – №12. – С. 39–45.
4. Туровский, Я. А. Типологизация локальных максимумов в спектрах электроэнцефалограмм, полученных путем непрерывного вейвлет-преобразования / Я. А. Туровский, С. Д. Кургалин,

А. Г. Семенов // Программные продукты и системы. – 2013. – №4. – С. 57.

5. Туровский, Я. А. Моделирование процессов выделения частотных локальных минимумов в сигналах электроэнцефалограмм / Я. А. Туровский, С. Д. Кургалин, А. В. Максимов // Вестник Тамбовского государственного технического университета. – 2012. – Т. 18, №4. – С. 827–833.

Разработка системы мониторинга веб-приложений

П. А. Толстых

Студент бакалавриата

Е. М. Михайлов

Доцент

Введение

Ежедневно тысячи веб-приложений по всему миру оказываются недоступными для пользователей. Согласно исследованию компании International Data Corporation (IDC), которая является общемировым поставщиком статистических данных о рынке информационных технологий, телекоммуникаций и потребительских технологий, среди 1000 самых больших компаний в мире, ранжированных по уровню дохода, средняя стоимость незапланированных простоев приложений в год составила от 1,25 до 2,5 млрд долларов. Корпорация Siemens Building Technologies в своем исследовании выяснила, что 33% компаний не могут посчитать влияние одного дня простоя на их бизнес в денежном эквиваленте [1].

При создании сайта большинство крупных компаний стремится использовать его не в качестве визитной карточки, а с целью получения коммерческой прибыли. Создав веб-приложение, владелец бизнеса может заняться непосредственной продажей товаров и услуг через интернет. Еще несколько лет назад покупки в интернете считались опасными, теперь же, по данным исследования платежной системы МИР, около 40% россиян оплачивают покупки онлайн. Ежегодное увеличение количества разработчиков только способствует реализации конкурентоспособного веб-ресурса, который быстро принесет выгоду его владельцу [2].

Однако многие интернет-пользователи ежедневно жалуются на низкий уровень поддержки веб-приложений или ее полное отсутствие. Люди вынуждены терять свое время и деньги, не имея возможности оплатить заказ, воспользоваться бонусными баллами, получить своевременную доставку товара. Для предотвращения этих и многих других проблем была разработана система мониторинга.

Различные сервисы для минимизации ошибок стали появляться по мере появления проблем у пользователей. Существуют целые методы

мониторинга веб-сайтов и сервисов, которые наиболее тщательно проверяют доступность и работоспособность систем.

Поэтому с точки зрения бизнеса компаниям, владеющим сайтом, который создает доход, необходимо иметь «под рукой» инструмент, который будет предупреждать о возможных сбоях и проблемах, контролировать возникающие нарушения и сообщать об их решении, а также предоставлять статистические данные [3].

1. Постановка задачи

Целью данной работы является разработка системы мониторинга веб-приложений, которая предназначена для системного администрирования. К разрабатываемой системе выдвинуты следующие требования:

- Мониторинг проблем с DNS-сервером, когда в определенные интервалы времени адрес сайта не может быть определен, хотя сам сайт физически доступен.
- Мониторинг проблем, связанных с большим временем ответа сайта.
- Мониторинг проблем доступности.
- Мониторинг работы с базой данных и проблем подключения к ней.
- Своевременное оповещение пользователей о проблемах, обнаруженных системой мониторинга, через различные каналы уведомлений.
- Составление подробных отчетов за выбранный промежуток времени и их визуализация с использованием статистических методов и математических характеристик.

2. Архитектура системы

Система мониторинга состоит из двенадцати компонентов, которые изображены на рис.1.

Пользовательский интерфейс, реализованный на языке программирования Python с помощью библиотеки Flask, обеспечивает взаимодействие между пользователем и системой мониторинга, а также позволяет пользователю адаптировать систему под свое веб-приложение.

Программа для создания метрик, написанная на языке программирования Python с использованием библиотеки Prometheus-client, для подсчета количественных значений отправляет метрики на сервер Prometheus с помощью компонента Pushgateway, установленного дополнительно.

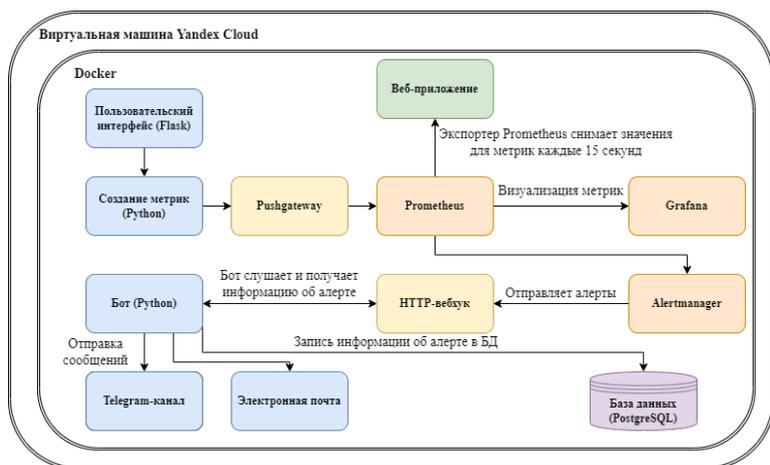


Рис. 1. Архитектура системы мониторинга

Веб-приложение содержит в себе клиентскую и серверную части, на которые и настраивается мониторинг.

Сервер Prometheus с установленным blackbox-экспортером каждые 15 секунд получает значения для метрик у веб-приложения через HTTP, HTTPS, DNS, TCP и ICMP. Также сервер Prometheus содержит конфигурационный файл с информацией об алертах, пороговых значениях метрик и адресе Alertmanager, чтобы связать между собой два компонента.

Alertmanager получает предупреждения от Prometheus и отправляет их с помощью запроса методом POST на HTTP-вебхук, указанный в конфигурационном файле Alertmanager.

Grafana визуализирует полученные от Prometheus метрики с помощью диаграмм, графиков, таблиц, предупреждений.

HTTP-вебхук является прослойкой между Alertmanager и ботом. Python-бот, написанный с использованием библиотек python-telegram-bot, mimetypes, smtplib и email получает POST-запрос на HTTP-вебхук с помощью веб-фреймворка Flask. Для записи информации об алерте в базу данных используется библиотека psycopg2.

В Telegram-канал отправляются сообщения от Python-бота. Электронная почта получает письма с информацией об алертах.

База данных PostgreSQL применяется для хранения информации о сработавшем алерте.

Docker используется для запуска выше перечисленных компонентов, каждый из которых изолирован в контейнере.

3. Реализация интерфейса

Для начала работы с системой мониторинга пользователь должен добавить бота в Telegram-канал, в который будут отправляться алерты при срабатывании. После установки Docker и запуска контейнеров в нем, необходимо настроить конфигурационные файлы сервера Prometheus и Alertmanager. После того как система будет готова к работе, Python-бот отправит тестовое сообщение в Telegram-канал. При срабатывании алерта бот отправляет в канал следующую информацию о нем:

- статус алерта;
- название алерта;
- название сервиса, на котором сработал алерт;
- название домена, на котором сработал алерт;
- описание алерта;
- ссылка на веб-приложение;
- значение метрики;
- тип (важность) алерта;
- действия, которые нужно предпринять;
- время срабатывания алерта.

Пример отправленного сообщения изображен на рис. 2.



Рис. 2. Пример отправленного сообщения

Система мониторинга позволяет отслеживать:

- рост четырехсотых ошибок состояния HTTP;
- всплеск пятисотых ошибок состояния HTTP;
- высокое время ответа сервера;
- падение уровня двухсотых кодов состояния HTTP;
- доступность веб-приложения;
- ошибки при подключении к базе данных;
- падение контейнера Docker;
- проблемы с DNS-сервером.

Заключение

В результате выполнения данной работы была разработана система мониторинга с оповещением в Telegram-канал и на электронную почту. Усовершенствованный аналог разработанной системы мониторинга используется компанией М.Видео в целях предупреждения сбоев и простоев веб-сайта и мобильного приложения.

Список литературы

1. It is good [Электронный ресурс]. – Режим доступа: <https://itisgood.ru/2018/12/26/chto-takoe-monitoring-veb-sajtov/>
2. ТАСС [Электронный ресурс]. – Режим доступа: <https://tass.ru/ekonomika/12030409>
3. Крол, А. Комплексный веб-мониторинг/ А. Крол, Ш. Пауэр. – М.: "Эксмо", 2011. – 768 с.

Решение переопределённых систем нелинейных уравнений

Ю. В. Третьякова

Студент бакалавриата

А. В. Атанов

Доцент

Введение

Переопределённые системы нелинейных уравнений, то есть системы, в которых количество уравнений превышает количество неизвестных, находят широкое применение в различных математических и прикладных задачах – в частности, в задачах оптимизации, в машинном обучении, в задачах обработки сигналов.

Переопределённую систему можно представить следующим образом:

$$\begin{cases} f_1(x_1, x_2, \dots, x_n) = 0, \\ f_2(x_1, x_2, \dots, x_n) = 0, \\ \dots \\ f_m(x_1, x_2, \dots, x_n) = 0, \end{cases}$$

Здесь f_1, f_2, \dots, f_m – заданные нелинейные вещественнозначные функции n вещественных переменных x_1, x_2, \dots, x_n , причём $m > n$.

Если для решения систем линейных и нелинейных уравнений, в которых число уравнений и неизвестных совпадает, разработано достаточно много методов (см., например, [1]), то методы решения переопределённых нелинейных систем остаются крайне немногочисленными и слабо изученными.

Данная работа посвящена изучению и реализации сравнительно нового подхода к решению переопределённых систем нелинейных уравнений, получившего название метода Гаусса–Якоби, а также его сравнению с модифицированным методом Ньютона–Рафсона.

1. Метод Ньютона–Рафсона для переопределённых систем нелинейных уравнений

Рассмотрим наиболее часто использующийся метод решения переопределённых СЛУ, а именно метод Ньютона–Рафсона [2].

Метод Ньютона–Рафсона является обобщением классического метода Ньютона и может быть применен для решения переопределенных систем нелинейных уравнений. В этом случае формула для вычисления очередного приближения к решению будет выглядеть следующим образом:

$$\bar{x}_{n+1} = \bar{x}_n - J^+(\bar{x}_n)f(\bar{x}_n), \quad (1)$$

где \bar{x} – вектор неизвестных, $f(\bar{x})$ – вектор-функция из системы нелинейных уравнений $f(\bar{x})=0$, $J^+(\bar{x}_n)$ – псевдообратная матрица для матрицы Якоби $J(\bar{x}_n)$, составленной из частных производных функций $f_i(\bar{x})$ по переменным \bar{x}_j .

На первом этапе метода задаются начальные значения – это размерность системы, начальное приближение и максимальное число итераций. Далее, начиная с нулевой итерации, производится расчет матрицы Якоби, псевдообратной матрицы для матрицы Якоби и значений функции для очередного приближения. После проверки заданного условия цикл либо повторяется, либо выдается результат в виде приближенного решения, значения функции и количества выполненных итераций.

2. Решение переопределённых систем нелинейных уравнений методом Гаусса–Якоби

Ещё один подход к решению переопределённых систем нелинейных уравнений, названный методом Гаусса–Якоби, был представлен в работе [2].

Рассмотрим частный случай переопределенной системы, в которой количество уравнений $m = 3$, а количество неизвестных $n = 2$:

$$\begin{cases} f_1(x, y) = 0, \\ f_2(x, y) = 0, \\ f_3(x, y) = 0. \end{cases}$$

Составим всевозможные подсистемы из $n = 2$ уравнений данной системы. В рассматриваемом случае таких подсистем всего три. Решим эти подсистемы любым методом (например, методом Ньютона, методом продолжения по параметру и т. д.). Пусть решения указанных подсистем имеют вид

$$\eta_{12} = (x_{12}, y_{12}), \quad \eta_{13} = (x_{13}, y_{13}), \quad \eta_{23} = (x_{23}, y_{23})$$

Вычислим среднее арифметическое полученных решений:

$$\eta = \frac{\eta_{12} + \eta_{13} + \eta_{23}}{3}. \quad (2)$$

Линеаризуем переопределенную систему в точке η . В линеаризованной форме j -е нелинейное уравнение в точке η имеет вид

$$f_j(\eta) + \frac{\partial f_j}{\partial x}(\eta)(x - x_s) + \frac{\partial f_j}{\partial y}(\eta)(y - y_s) = 0$$

или

$$\frac{\partial f_j}{\partial x}(\eta)x + \frac{\partial f_j}{\partial y}(\eta)y + \left(f_j(\eta) - \frac{\partial f_j}{\partial x}(\eta)x_s - \frac{\partial f_j}{\partial y}(\eta)y_s \right) = 0.$$

Находим коэффициенты π_{ij} по следующим формулам:

$$\pi_{12}(\eta) = \left| \begin{array}{cc} \frac{\partial f_1}{\partial x}(\eta) & \frac{\partial f_1}{\partial y}(\eta) \\ \frac{\partial f_2}{\partial x}(\eta) & \frac{\partial f_2}{\partial y}(\eta) \end{array} \right|^2 = \left(\frac{\partial f_1}{\partial x}(\eta) \frac{\partial f_2}{\partial y}(\eta) - \frac{\partial f_2}{\partial x}(\eta) \frac{\partial f_1}{\partial y}(\eta) \right)^2,$$

$$\pi_{13}(\eta) = \left| \begin{array}{cc} \frac{\partial f_1}{\partial x}(\eta) & \frac{\partial f_1}{\partial y}(\eta) \\ \frac{\partial f_3}{\partial x}(\eta) & \frac{\partial f_3}{\partial y}(\eta) \end{array} \right|^2 = \left(\frac{\partial f_1}{\partial x}(\eta) \frac{\partial f_3}{\partial y}(\eta) - \frac{\partial f_3}{\partial x}(\eta) \frac{\partial f_1}{\partial y}(\eta) \right)^2,$$

$$\pi_{23}(\eta) = \left| \begin{array}{cc} \frac{\partial f_2}{\partial x}(\eta) & \frac{\partial f_2}{\partial y}(\eta) \\ \frac{\partial f_3}{\partial x}(\eta) & \frac{\partial f_3}{\partial y}(\eta) \end{array} \right|^2 = \left(\frac{\partial f_2}{\partial x}(\eta) \frac{\partial f_3}{\partial y}(\eta) - \frac{\partial f_3}{\partial x}(\eta) \frac{\partial f_2}{\partial y}(\eta) \right)^2.$$

Комбинаторное решение переопределённой системы будет иметь вид

$$x_{GJ} = \frac{\pi_{12}x_{12} + \pi_{13}x_{13} + \pi_{23}x_{23}}{\pi_{12} + \pi_{13} + \pi_{23}}, \quad y_{GJ} = \frac{\pi_{12}y_{12} + \pi_{13}y_{13} + \pi_{23}y_{23}}{\pi_{12} + \pi_{13} + \pi_{23}}.$$

Для рассмотренного метода Гаусса–Якоби был разработан алгоритм, включающий следующие шаги.

На первом этапе задаются начальные значения – это размерность системы, функция невязки и значение, указывающее на заданную точность. Далее с учетом размерности системы составляются комбинации подсистем и находятся все решения каждой подсистемы. После этого применяется функция для нахождения решения, дающего

минимальную невязку, затем рассчитывается среднее арифметическое для переменной η по x , y и находятся коэффициенты π_{ij} . Проверяется условие превышения заданной точности и находится комбинаторное решение переопределенной системы.

3. Программная реализация

Реализации программного кода осуществлялась с использованием языка программирования Python со встроенными библиотеками NumPy и SymPy.

Python – это интерпретируемый, высокоуровневый язык программирования с динамической типизацией [3]. Он предлагает простой и понятный синтаксис, а также обширную стандартную библиотеку, которая позволяет решать множество задач, включая работу с сетью, базами данных, научными вычислениями, веб-разработкой и многим другим.

NumPy – это библиотека для работы с массивами и матрицами в языке программирования Python. Она предоставляет эффективные функции для математических операций над массивами. NumPy также обеспечивает высокую скорость выполнения операций благодаря тому, что большинство функций написаны на языке C и предоставляются в виде библиотеки, которая может быть вызвана из Python.

SymPy – это библиотека для символьных вычислений в языке программирования Python [4]. Она позволяет работать с символьными выражениями, решать алгебраические уравнения и дифференциальные уравнения, производить символьное дифференцирование и интегрирование, а также создавать символьные матрицы и оперировать ими.

При помощи библиотеки SymPy для метода Гаусса–Якоби велся расчет матрицы Якоби, а также при помощи встроенной функции производилось нахождение всех решений для каждой комбинации подсистем. При помощи библиотеки NumPy в методах осуществлялось умножение матриц, нахождения определителя матрицы, а также производился расчет псевдообратной матрицы для матрицы Якоби.

4. Пример решения переопределённой системы

Рассмотрим переопределённую систему $m = 3$ нелинейных уравнений с $n = 2$ неизвестными:

$$\begin{cases} x^2 - xy + y^2 + 2x = 0, \\ x^2 - 2xy + y^2 + x - y - 2 = 0, \\ xy + y^2 - x + 2y - 3 = 0. \end{cases}$$

Сначала решим эту систему методом Ньютона-Рафсона.

Зададим начальные приближения, например, $x_0 = -3$, $y_0 = -1$. Основываясь на формуле (1), находим очередное приближенное решение на соответствующей итерации. В табл. 1 представлены результаты решения методом Ньютона-Рафсона для $N = 10$ итераций.

Таблица 1

Результаты решения системы методом Ньютона-Рафсона

Номер итерации	x_i	y_i
1	-2.57329843	-0.60209424
2	-2.47021291	-0.50609503
3	-2.46438189	-0.50051186
4	-2.4644302	-0.50054700
5	-2.46442961	-0.50054653
6	-2.46442962	-0.50054654
7	-2.46442962	-0.50054654
8	-2.46442962	-0.50054654
9	-2.46442962	-0.50054654
10	-2.46442962	-0.50054654

Решим теперь эту же систему методом Гаусса-Якоби.

Составим комбинации подсистем по размерности системы, то есть (f_1, f_2) , (f_2, f_3) , (f_1, f_3) . Решения подсистем данной системы записаны в табл. 2.

Таблица 2

Решения подсистем

Комбинация подсистем	Решения
(f_1, f_2)	$(-2; 0)$ $(-0.5 - 0.866025 i ; -1.5 - 0.866025 i)$ $(-0.5 + 0.866025 i ; -1.5 + 0.866025 i)$
(f_2, f_3)	$(-2.5; -0.5), (-1; -2), (-1; 1), (2; 1)$
(f_1, f_3)	$(-2.434258; -0.565741)$ $(-1.232408; -1.767591)$ $(-0.5 - 0.866025 i ; 1)$ $(-0.5 + 0.866025 i ; 1)$

Из полученных решений отбираем только вещественные, а затем для каждой подсистемы выбираем единственное решение, дающее минимальную невязку, вычисляемую по формуле

$$\sum_{i=1}^3 (f_i^2(x, y))^2 \rightarrow \min_{x, y}.$$

Результаты этого этапа представлены в табл. 3.

Таблица 3

Решения с минимальной невязкой

Комбинация подсистем	Решения
(f_1, f_2)	$(-2; 0)$
(f_2, f_3)	$(-2.5; -0.5)$
(f_1, f_3)	$(-2.434258; -0.565741)$

Далее вычисляем значения η_x и η_y по формуле (2):

$$\eta_x = -2.311419, \quad \eta_y = -0.355247.$$

После этого находим

$$\pi_{12} = 3.769667,$$

$$\pi_{23} : 47.929483,$$

$$\pi_{13} : 20.132566.$$

Окончательно получаем комбинаторное решение Гаусса–Якоби:

$$(x_{GJ}, y_{GJ}) = (-2.455335, -0.492186).$$

5. Сравнение методов

Сравним решение системы, полученное методом Гаусса–Якоби, с решениями, полученными методом Ньютона–Рафсона при различных начальных приближениях. В табл. 4 приведены решения системы методом Ньютона–Рафсона с начальным приближением $x_0 = -3$, $y_0 = -1$, а также с начальным приближением $x_0 = 1$, $y_0 = 0$. Для сравнения приводится также решение, полученное методом Гаусса–Якоби.

Сравнение решений, полученных методом Ньютона–Рафсона при различных начальных приближениях с решением, полученным методом Гаусса–Якоби

Метод	Решение	Невязка
Ньютона-Рафсона с $x_0 = -3, y_0 = -1$	(-2.464429; -0.500547)	0.040316
Ньютона-Рафсона с $x_0 = 1, y_0 = 0$	(-1.096753; 0.835606)	0.631509
Гаусса-Якоби	(-2.455334; -0.492186)	0.041089

Как видно, начальное приближение критически влияет на точность решения, получаемого методом Ньютона–Рафсона. При этом метод Гаусса–Якоби не требует задания начального решения, что является существенным преимуществом при решении систем.

Заключение

В работе рассмотрены два метода решения переопределённых систем нелинейных уравнений – метод Ньютона–Рафсона и метод Гаусса–Якоби. Для указанных методов были разработаны программы, позволяющие решать переопределённые системы нелинейных уравнений. Было проведено сравнение методов и на примере установлено, что метод Гаусса–Якоби имеет преимущество по сравнению с методом Ньютона–Рафсона, так как не требует задания начального приближения.

Список литературы

1. Вержбицкий, В. М. Основы численных методов / В. М. Вержбицкий. – Москва : Директ-Медиа, 2013. – 847 с.
2. Palancz, B. Mathematical Geosciences / B. Palancz [et al.] // Hybrid Symbolic-Numeric Methods. – 2018. – P. 77–110.
3. Dawson, M. Python Programming for the Absolute Beginner / M. Dawson. – O’Reilly, 2014. – 416 p.
4. Sandona, D. Symbolic Computation with Python and SymPy / D. Sandona. – 2021. – 441 p.

Разработка программы для корректировки доз лекарственных препаратов

Т. А. Улезько

Студент бакалавриата

Р. С. Шевцов

Студент бакалавриата

С. Д. Кургалин

Профессор

Г. А. Батищева

Профессор ВГМУ

С. В. Борзунов

Доцент

Введение

Под автоматизированным рабочим местом (АРМ) врача – клинического специалиста понимают набор программных продуктов и технических средств, обеспечивающих автоматизацию его работы. В зависимости от профиля врача, к функционалу АРМ предъявляется определенный набор требований. В качестве одной из важных компонент такого набора может выступать информационная система для корректировки доз используемых лекарственных препаратов.

Пациенты медицинских стационаров нередко страдают от нарушения функций почек. Это связано с тем, что назначаемые им лекарственные препараты довольно часто выводятся из организма почками. Если у пациента имеются нарушения функций почек, то препарат не выводится должным образом, что приводит к нежелательным побочным эффектам и наносит вред здоровью человека. В силу этого, врачам необходимо проводить корректировку доз назначаемых лекарственных препаратов с учетом их фармакологических особенностей. При этом специалисты обращаются к инструкциям по медицинскому применению препаратов и к клиническим рекомендациям. Однако проводить вручную поиск

необходимой информации во время врачебного приема – долго и неудобно. Нужно учитывать и ограниченность времени врачебного приема. Поэтому в настоящей работе предлагается решение указанной проблемы, заключающееся в автоматизации процесса выполнения рутинных расчетов и поиска сведений с целью обеспечить врача необходимой информацией о требуемой корректировке доз применяемых препаратов.

1. Характеристики функции почек

Для оценки функции почек используются показатели: клиренс креатинина (КК) и скорость клубочковой фильтрации (СКФ). Эти величины позволяют оценить, насколько хорошо почки выполняют свою главную функцию. Они могут быть посчитаны для конкретного пациента по определенным формулам [1, 2], при подсчете также учитывается возраст, рост, вес и некоторые другие показатели. Рекомендации и ограничения дозирования препаратов в случае почечной недостаточности опираются именно на эти две величины. Оба показателя важны, поэтому в инструкциях по применению препаратов можно увидеть информацию, относящуюся как к клиренсу креатинина, так и к скорости клубочковой фильтрации.

2. Блок-схема алгоритма работы программы

Упрощенная блок-схема алгоритма работы программы представлена на рис. 1.

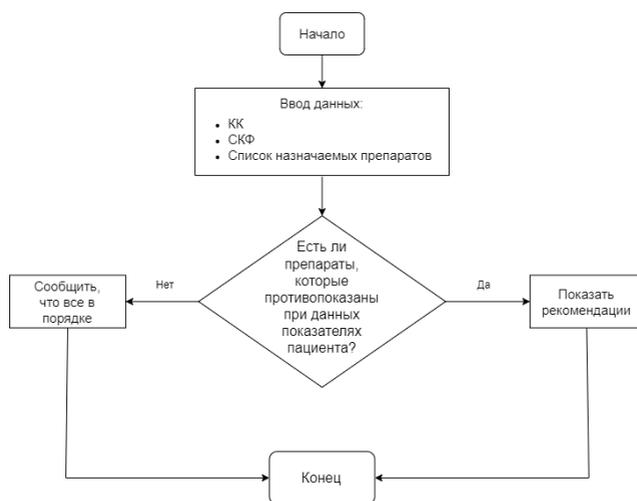


Рис. 1. Упрощенная блок-схема алгоритма работы программы

После начала выполнения программы ожидаются следующие входные данные: показатели функции почек пациента (КК, СКФ) и список лекарственных препаратов, которые ему планируется назначить. Далее производится проверка, какие есть у указанных препаратов инструкции по применению и клинические рекомендации, действующие именно для рассматриваемых значений показателей функции почек пациента. Если таковые имеются, то программа дает врачу рекомендации, иначе ему сообщается об их отсутствии. Далее, при необходимости, процесс может повториться.

3. ER-диаграмма предметной области

ER-диаграмма предметной области представлена на рис. 2.

Каждый лекарственный препарат имеет международное название, и для него может существовать целый ряд торговых названий, под которыми он поступает в продажу. Торговые названия зачастую удобнее использовать для идентификации препарата. Инструкция по применению препарата содержит также некоторое количество правил приема. В контексте применения программы под правилом приема понимается текстовое сообщение, содержащее рекомендации по дозированию препарата, а также числовые интервалы для значений КК и СКФ. Если хотя бы один из этих показателей пациента попадет в соответствующий интервал, то врачу будет выдана рекомендация. Стоит отметить, что допустимо отсутствие одного из интервалов, так как инструкции по дозированию препаратов часто опираются только на один из показателей.

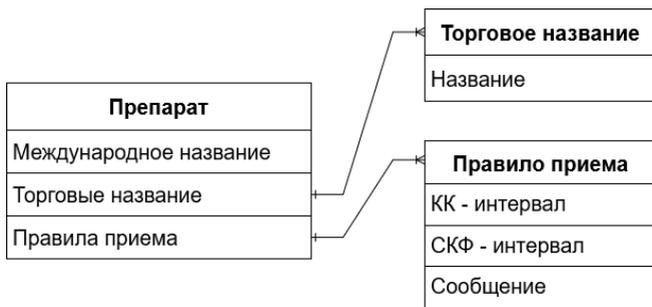


Рис. 2. ER-диаграмма предметной области

4. Формат хранения данных

Изначально был сделан упор на предложение специалистов-медиков работать с данными в виде таблиц (см. рис. 3) и корректировать информацию с помощью стандартных графических редакторов. В таком

случае не пришлось бы реализовывать процесс редактирования данных в самой программе. Однако вскоре стало ясно, что работать с данными в таком формате будет не совсем удобно. Это связано с тем, что искать и изменять информацию в большой таблице довольно сложно, к тому же не будет никакой защиты от разного рода ошибок. Так, неправильно указанное неравенство может повлечь за собой некорректную работу всей программы. Чтение такого файла программным путем тоже неудобно, так как потребует реализации нетривиальных алгоритмов извлечения и валидации данных [3].

INN	IN	Скорректир	Режим доз
'ликлазид	Гликлазид, Глидиаб, Гли	>15	Применени недостаточ
		<15	Согласно к Применени
'либенкламид	Манинил, Стагиглин	> 60	снижения
		< 60	Согласно к данным ин
'лимепирид	Амарил, Глемауно, Гли	>15	Коррекция
		<15	Согласно к
'ликваллон	Юглин, Глюбенлом	>15	Коррекция приема

Рис. 3. Формат XLSX

В силу этого было принято решение перейти к формату XML, который лучше подходит для наших задач (рис. 4). Был создан программный интерфейс, позволяющий управлять данными, относящимися к лекарственным препаратам. Основное преимущество такого подхода состоит в том, что файлы с данными генерируются исключительно в рамках программного комплекса, поэтому при последующем чтении данных из файла отпадает необходимость их проверки на отсутствие ошибок, связанных с влиянием человеческого фактора. Кроме того, поскольку интерфейс программы создавался специально для работы с данными этой предметной области, поиск информации является значительно более удобным, чем это было бы при использовании стандартных табличных форматов.

```

<intl-name>Инсулин Аспарт</intl-name>
<trade-names>
  <value>НовоРапид</value>
  <value>РинФаст</value>
  <value>Росинсулин аспарт P</value>
</trade-names>
<rules>
  <rule>
    <ccr-interval>
      <begin>0</begin>
      <end>15</end>
    </ccr-interval>
    <message>
      Использовать с осторожностью.
    </message>
  </rule>

```

Рис. 4. Формат XML

5. Интерфейс программы

Для разработки пользовательского интерфейса была выбран интерфейс программирования приложений Windows Forms, широко применяемый для создания разнообразных клиентских приложений рабочего стола.

В окне, показанном на рис. 5, можно редактировать информацию, связанную с определенным препаратом.

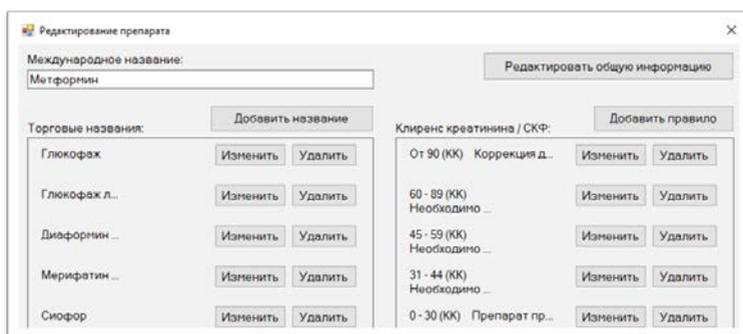


Рис. 5. Окно редактирования препарата

В окне, представленном на рис. 6, есть возможность редактировать информацию, связанную с определенным правилом. Здесь можно выбрать показатели функции почек, от которых будет зависеть данное правило, и ввести сообщение, которое будет показано врачу.

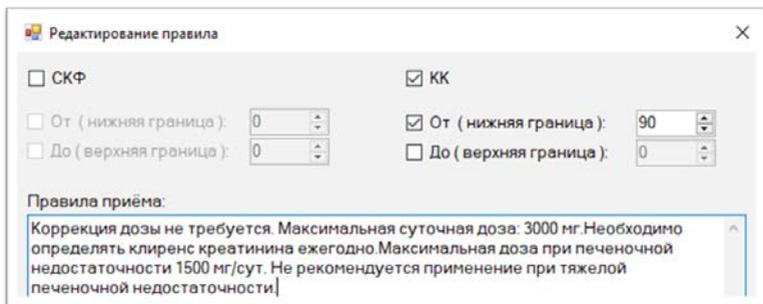


Рис. 6. Окно редактирования правила

На рис. 7 показан программный калькулятор, который используется для расчета показателей функции почек (КК, СКФ) на основе некоторых данных пациента. В текстовое поле выводятся результаты расчетов вместе с дополнительной информацией.

Рис. 7. Программный калькулятор для расчета КК и СКФ

Посчитанные результаты с помощью специальной кнопки интерфейса можно подставить в соответствующие поля в секции

параметров проверки (рис. 8). Дело в том, что иногда у врача есть заранее рассчитанные показатели, которые нужно поместить сразу в секцию проверки, минуя калькулятор.

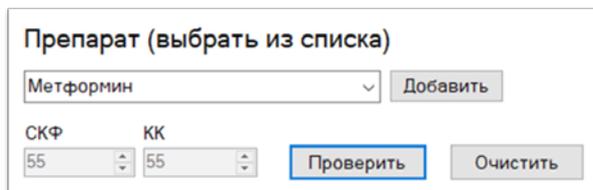


Рис. 8. Параметры проверки

Выбор препаратов происходит с помощью выпадающего списка. Все добавленные для проверки препараты отображаются в соответствующем списке (рис. 9).

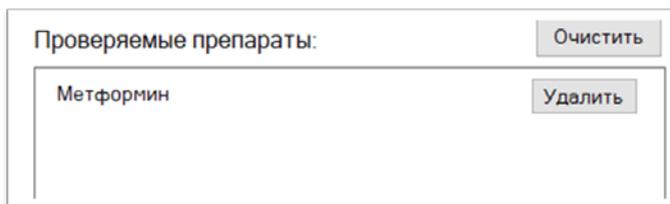


Рис. 9. Список проверяемых препаратов

Если программа обнаружила рекомендации, они будут выведены в специальном окне, как показано на рис. 10.

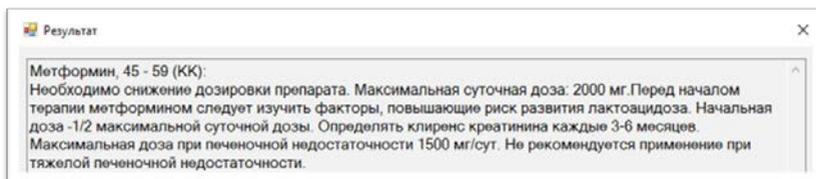


Рис. 10. Результат проверки дозировки препарата

Заключение

В практическом здравоохранении врачу на прием пациента отводится определенное количество времени. Важно, чтобы врачебный прием был полноценным без отвлечения специалиста на поиск информации в инструкциях по применению препаратов и в клинических

рекомендациях. Разработанная программа в рамках поддержки принятия решений позволяет врачу быстро оценить стадию нарушения функции почек пациента и подобрать соответствующий корректный режим дозирования препарата.

Предполагается, что представленная программа в дальнейшем станет составной частью программного комплекса для АРМ. В первую очередь, планируется внедрить созданное программное обеспечение в эндокринологических отделениях стационаров, так как именно там, как правило, есть необходимость в коррекции режима дозирования препаратов в связи с особенностями профильного заболевания. Представленная разработка, вероятно, будет востребована среди специалистов практического здравоохранения, так как ее применение обеспечит сокращение времени поиска информации и рутинных вычислений.

Среди направлений развития проекта укажем также на возможность переноса текущего функционала в мобильное приложение, что позволит заметно повысить удобство использования программного обеспечения и окажет неоценимую помощь врачу в его повседневной деятельности.

Список литературы

1. Cockcroft, D. W. Prediction of Creatinine Clearance from Serum Creatinine / D. W. Cockcroft, M. H. Gault // *Nephron*. – 1976. – V. 16. – P. 31–41.
2. Levey, A. S. A New Equation to Estimate Glomerular Filtration Rate / A. S. Levey, L. A. Stevens, C. H. Schmid [et al.] // *Annals of Internal Medicine*. – 2009. – V. 150, № 9. – P. 604–612.
3. Price, M. J. C# 8.0 and .NET Core 3.0 : Modern Cross-Platform Development / M. J. Price. – Fourth ed. – Packt, 2019. – 818 p.

Автоматическое задание классов сущностей на основе выделения из текстов синонимов, гипонимов, гиперонимов

А. А. Ушакова

Студент бакалавриата

В. В. Гаршина

Доцент

Введение

В наше время пользователи Интернет каждый день генерируют гигабайты данных, которые несут в себе множество полезной информации. Научные работы и всевозможные исследования пополняют всемирную сеть, добавляя в нее новые сведения, значительная часть которых представлена на естественном языке (ЕЯ). Зачастую такие данные являются неструктурированным текстом. Люди не могут эффективно анализировать такой объем с помощью традиционных алгоритмов и структур данных из-за того, что они не позволяют использовать семантические связи.

Таким образом, назрела необходимость в таком представлении информации, которое бы позволяло бы с одной стороны хранить огромное количество объектов и связей между ними, а с другой предоставляло высокоскоростной доступ к хранящимся данным, и, кроме того, сохраняло семантику. Одной из самых эффективных структур данных, позволяющей решать задачи подобного класса, является граф знаний, который относительно недавно появился и стал предметом исследований в последние годы

Для построения графа знаний используется онтология. В информационных технологиях онтология представляет собой структуру знаний, состоящую из терминов, их определений и связанной информации в одной или нескольких областях. Однако по мере того, как преобладают большие данные, ручное расширение существующих онтологий актуальной информацией становится сложной задачей из-за трудоемкого процесса и высокой стоимости ручного труда экспертов. Чтобы облегчить эту проблему, в этой работе делается попытка полностью автоматического расширения онтологии предметной области мероприятий на основе текстов и баз знаний.

1. Обзор существующих решений

Методы расширения онтологий можно разделить на ручные и автоматические.

1. Расширение онтологии вручную

Один из самых технически простых способов расширения онтологии. Методы наполнения онтологий, предполагающие ручной ввод данных предполагают использование пользователем универсальных или специальных редакторов онтологий, таких как универсальные редакторы онтологий Protege, KAON и OntoSaurus[1].

Semantic Wiki- ещё один метод расширения онтологий с помощью знаний. Онтология извлекается из Wiki-систем и сохраняется в формате OWL.

2. Автоматическое расширение онтологии

Наиболее интересный и технически непростой метод расширения онтологии – автоматическое расширение онтологии. Одним из популярных методов является обновление онтологии алгоритмом на основе текстового корпуса.

Плюс извлечения знаний из текста заключаются в легкости составления необходимой терминологической модели и её стандартизации. Однако подобный подход не является совершенным и нуждается в доработке, чтобы повысить его эффективность[2].

Существуют методы, основанные на применении лексико-синтаксических шаблонов. Расширение онтологии в этом случае базируется на многоуровневом анализе естественного языка, а точнее, его морфологии, синтаксиса и семантики. Данный метод имеет максимальную эффективность при наличии довольно большого объёма коллекции шаблонов. Однако существенным недостатком является отсутствие возможности работы со сложными шаблонами или их группами.

Онтология может автоматически расширяться с помощью SemanticWeb. Данный подход заключается в разработке автоматического построения семантической карты (на примере Web-ресурса), на основе анализа текста. Для этого формируется ряд правил, которые направлены на работу со словами в предложениях. В качестве достоинства данного метода можно выделить возможность автоматического создания классов, их свойств и отдельных экземпляров для реализации структуры онтологии.[3].

2. План решения

Для выполнения поставленной задачи требуется:

- 1) Выделить именованные сущности из исходного текста;
- 2) Извлечение отношений между сущностями;

3)Связать сущности с онтологией.

Выделение именованных сущностей

Поиск именованных сущностей (NER) является одной из рядовых задач обработки естественного языка. Она заключается в том, чтобы найти в исходном предложении все объекты, принадлежащие к одному из предопределенных классов.

Задача распознавания именованных сущностей может быть поставлена следующим образом: для заданной последовательности слов (токенов) предсказать последовательность тегов (меток). Каждому входному токеноу сопоставляется тег из заданного множества тегов. Например:

Алиса в Стране чудес

PER O LOC LOC

здесь PER - персона, LOC - локация. Однако, представленная разметка не позволяет разделять подряд идущие сущности. Для разделения таких сущностей используют префиксы B и I перед каждым тегом кроме O. Префикс B обозначает начало сущности, а I - продолжение. Тогда для примера выше будет следующая разметка:

Алиса в Стране чудес

B-PER O B-LOC I-LOC

Разметка с префиксами B и O - наиболее распространённый способ разметки данных. Данный тип разметки часто называют BIO или IOB.

В библиотеке DeepPavlov есть ряд моделей которые решают данную задачу. Была выбрана мультиязычная модель BERT, позволяющая выполнять переход с одного языка на другой с нуля. Модель `ner_ontonotes_bert_mult` обучалась на корпусе OntoNotes (английский язык), который имеет 18 тегов именованных сущностей.

Производительность мультиязычной модели оценивалась на четырех тестовых наборах NER для немецкого, русского, китайского и вьетнамского языков [5]. Результаты переноса представлены на рис. 1.

	German	Russian	Chinese	Vietnamese
PER	87.21	95.74	84.12	83.30
LOC	69.54	82.62	60.83	60.99
ORG	52.95	55.68	54.34	38.92
Total	70.71	79.39	64.44	68.20

Рис. 1. Производительность мультиязычной модели при тестировании на четырех тестовых наборах

Извлечение отношений между сущностями

Извлечение отношений (RE) — это задача извлечения семантических отношений из текста, которые обычно возникают между двумя или более объектами. Эти отношения могут быть разных типов. Например, «Париж находится во Франции» указывает на отношения «находится в» между Парижем и Францией. Это можно обозначить с помощью триплета (Париж, находится в, Франции).

В библиотеке DeepPavlov есть модель, решающая задачи обнаружения и классификации отношений между двумя объектами в тексте. DeepPavlov обеспечивает извлечение отношения на уровне документа, что означает, что отношение может быть обнаружено между сущностями, которые не находятся в одном предложении. Модель RE для русского языка обучена с помощью RuRED на базе новостного корпуса Lenta.ru [6].

3. Выбор средств реализации

Предобученные модели: RE, ner_ontonotes_bert_mult

Язык программирования : python

Среда разработки : Google Colaboratory (GPU ускоритель)

Список используемых библиотек: DeepPavlov, nltk, sparql-client, ruwordnet, SPARQL Burger, rumorphy2

Для токенизации исходного текста используется библиотека Natural language toolkit (NLTK) - популярный открытый пакет библиотек, используемых для разного рода задач NLP.

Отправка SPARQL-запросов к графовым базам данных осуществляется с помощью библиотеки sparql-client. sparql-client — это библиотека SPARQL -запросов, которая выполняет запросы к эндпоинту SPARQL через HTTP. Она автоматически преобразует литералы в соответствующие типы Python [7].

Для создания SPARQL-запросов используется библиотека SPARQL Burger. Данная библиотека конструирует SPARQL-запросы на Python и автоматизирует создание шаблонов Select и Update SPARQL-запросов [8].

Для определения места сущности в онтологии используется тезаурус русского языка RuWordNet. Он был создан на основе автоматизированной трансформации тезауруса RuThes в формат WordNet, который является одним из самых известных ресурсов в области автоматической обработки текстов [9].

4. Реализация

1) Выделение именованных сущностей из исходного текста

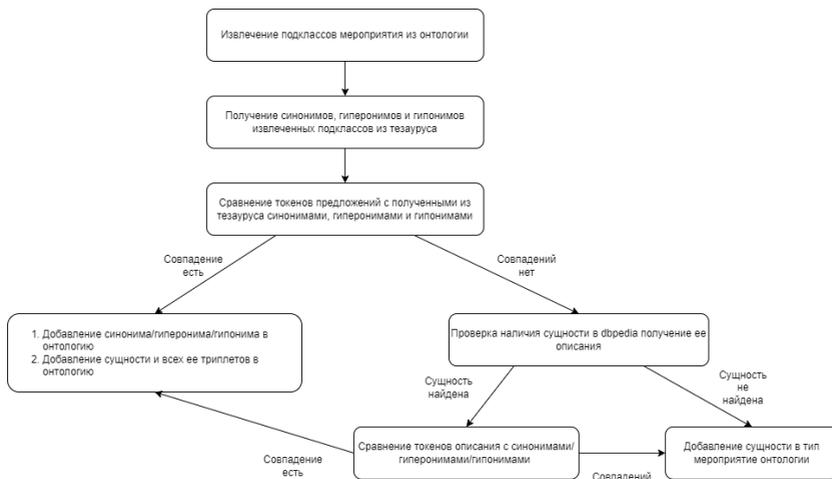


Рис. 5. Алгоритм связывания сущностей с онтологией

```

<?xml version="1.0" encoding="UTF-8"?>
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
  <head>
    <variable name="o"/>
  </head>
  <results>
    <result>
      <binding name="o">
        <uri>http://www.owl-ontologies.com/ontology1680179164.owl#Спортивное_мероприятие</uri>
      </binding>
    </result>
  </results>
</sparql>

```

Рис. 6. Результат SELECT SPARQL-запроса

[[СПОРТИВНАЯ ВСТРЕЧА, СПОРТИВНОЕ СОРЕВНОВАНИЕ, СПОРТИВНОЕ СОСТЯЗАНИЕ, ТУРНИР, ФИЗИКУЛЬТУРНО-СПОРТИВНОЕ МЕРОПРИЯТИЕ, СПОРТИВНО-МАССОВОЕ МЕРОПРИЯТИЕ, СПОРТИВНЫЙ ПРАЗДНИК, СПОРТИВНОЕ МЕРОПРИЯТИЕ, ФИЗИКУЛЬТУРНО-СПОРТИВНЫЙ ПРАЗДНИК, ВСТРЕЧА, СОРЕВНОВАНИЕ, ПЕРВЕНСТВО, БАТАЛИЯ, СПОРТИВНЫЙ ТУРНИР, КОНКУРЕНЦИЯ, СОПЕРНИЧЕСТВО, СОРЕВНОВАНИЕ, СОСТЯЗАНИЕ]]

[[КРОСС ПО ПЕРЕСЕЧЕННОЙ МЕСТНОСТИ, КРУГОВОЙ ТУРНИР, СПОРТИВНЫЙ ФИНАЛ, ИГРЫ ДОБРОЙ ВОЛИ, УНИВЕРСАДА, РЫВОК (УПРАЖНЕНИЕ), МАТЧ, ОДИНОЧНЫЙ РАЗРЯД, ПАРНЫЙ МАТЧ, БОКСЕРСКИЙ МАТЧ, СПОРТИВНАЯ ГОНКА, МНОГОБОРЬЕ, ТОВАРИЩЕСКИЕ СОРЕВНОВАНИЯ, ОБЩЕКОМАНДНЫЙ ЗАЧЕТ, ЧЕМПИОНАТ (СПОРТИВНОЕ СОРЕВНОВАНИЕ), КОРОТКАЯ ПРОГРАММА, КОНКУР (КОННЫЙ СПОРТ), КУБОВОЕ СОРЕВНОВАНИЕ, ЧЕТВЕРТЬФИНАЛ, ОЛИМПИЙСКИЕ ИГРЫ, ПЯТИБОРЬЕ, ШАХМАТНЫЙ ТУРНИР, ПРОИЗВОЛЬНАЯ ПРОГРАММА, ИНДИВИДУАЛЬНОЕ СОРЕВНОВАНИЕ, ПОЛУФИНАЛ, ТУРНИР НА ВЫБЫВАНИЕ, ТИТУЛЬНЫЙ БОЙ, ЗАЗЕД СОРЕВНОВАНИЯ, КВАЛИФИКАЦИОННЫЙ ЭТАП, ПРИКИДКА (ПРЕДВАРИТЕЛЬНОЕ СОРЕВНОВАНИЕ), ЗАПЛЫВ, СПОРТИВНЫЙ ЗАБЕГ, ПРОИЗВОЛЬНЫЙ ТАНЕЦ, ДВОБОРЬЕ, МЕЖДУНАРОДНОЕ СПОРТИВНОЕ СОРЕВНОВАНИЕ, СПАРТАКИАДА, КОНКУРС, КОНКУРЕНЦИЯ НА РЫНКЕ, ЧЕТВЕРТЬФИНАЛ, РЫЦАРСКИЙ ТУРНИР, ПОЛУФИНАЛ, ОСПОРИТЬ (ДОБИВАТЬСЯ, СОПЕРНИЧАЯ), СПАРРИНГ (ТРЕНИРОВОЧНЫЙ БОЙ), СПОРТИВНОЕ СОРЕВНОВАНИЕ, СПОР (ОБСУЖДЕНИЕ)]]

[[СОРЕВНОВАНИЕ, СОСТЯЗАНИЕ, ЗАНЯТИЕ, ДЕЯТЕЛЬНОСТЬ, УЧАСТИЕ, УЧАСТВОВАТЬ, БОРОТЬСЯ С СОПЕРНИКОМ]]

Рис. 7. Полученные из тезауруса синонимы, гиперонимы и гипонимы

3) Для упрощения поиска совпадений токенов предложений с синонимами, гиперонимами и гипонимами, полученными из тезауруса, используется лемматизация текста. Слова приводятся к именительному падежу единственному числу с помощью морфологического анализатора `rumorphy2` [10].

При наличии совпадений дополнительно формируются отношения синоним/гипероним/гипоним с соответствующим подклассом онтологии, и задается тип сущности, определяющий ее место в онтологии (рис. 8).

```
[[['соревнование', 'синоним', 'спортивное мероприятие'], ['лыжня россия', 'type', 'соревнование'],  
['лыжня россия', 'type', 'спортивное мероприятие'], [['соревнование', subClassOf, мероприятие]]]
```

Рис. 8. Дополнительные отношения

Полученные отношения добавляются в онтологию посредством INSERT SPARQL-запрос к графу в GraphDB. Пример такого запроса:

```
prefix ex:<http://www.owl-ontologies.com/Ontology1680179164.owl#>  
insert data  
{ex:Соревнование ex:синоним ex:Спортивное_мероприятие.  
}
```

Если же совпадений не было обнаружено осуществляется поиск сущности и ее описания в базе знаний DBpedia. Для этого к DBpedia отправляется SELECT SPARQL-запрос следующего типа:

```
SELECT *  
WHERE  
{  
?event rdfs:label "Лыжня России"@ru .  
?event dbo:abstract ?abstract .  
filter(langMatches(lang(?abstract),"ru"))  
}  
LIMIT 1
```

При совпадении токенов описания с синонимами, гиперонимами и гипонимами, полученными из тезауруса, формируются отношения описанные ранее.

Если сущность не была найдена в DBpedia или описание сущности не содержало синонимы, гиперонимы или гипонимы, то сущность добавляется в онтологию с типом мероприятие.

5. Результат

Исходный текст:

11 февраля 2023 года в Санкт-Петербурге состоится открытая массовая лыжная гонка «Лыжня России». Соревнование «Лыжня

России» основано в 1982 году и традиционно собирает поклонников одного из самых популярных и массовых видов спорта.

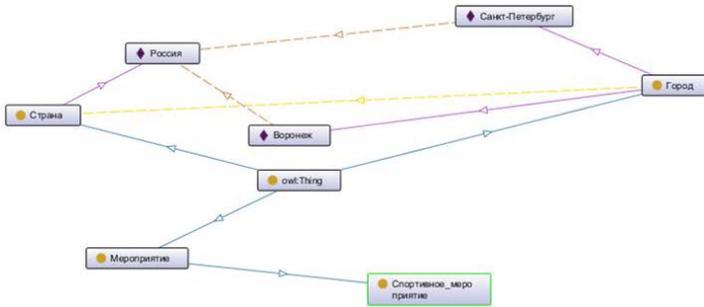


Рис. 9. Исходная онтология

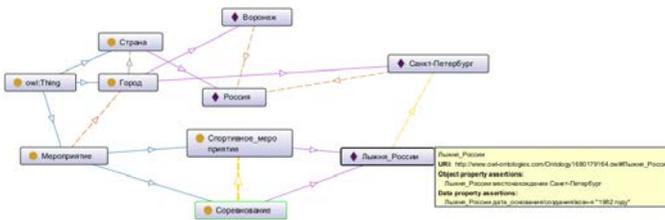


Рис. 10. Дополненная онтология

На рис. 10 видно, что исходная онтология была расширена новыми сущностями Лыжня России и Соревнование. Сущность Соревнование встроилась как синоним сущности Спортивное мероприятие и подкласс сущности мероприятие. Для сущности Лыжня России были добавлены все выделенные триплеты и определен тип.

Таким образом, решена задача по автоматическому заданию классов сущностей на основе выделения из текстов синонимов, гиперонимов и гипонимов.

Заключение

В ходе данной работы был проведен анализ существующих методов пополнения онтологии. Был составлен алгоритм и разработаны процедуры автоматического расширения онтологии. Использование данного решения упрощает процесс обновления онтологии в

соответствии с обновлением данных о предметной области в сети Интернет.

Список литературы

1. Гервик, Е. М. Обзор методов заполнения онтологий / Е. М. Гервик, А. В. Сеньков // Международный журнал информационных технологий и энергоэффективности. – 2018. – Т. 3, № 3(9). – С. 10-18.
2. Ермаков, А. Е. Автоматизация онтологического инжиниринга в системах извлечения знаний из текста / А.Е. Ермаков // Труды международной конференции «Диалог» (Бекасово, 4–8 июня 2008 г.) – Бекасово, 2008. – С. 154-158.
3. Рабчевский, Е. А. Автоматическое построение онтологий / Е. А. Рабчевский // Труды 13 Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции». – Переславль-Зелеский, 2007. – 4 с.
4. Named Entity Recognition model – Руководство пользователя [Электронный ресурс]. – Режим доступа: <https://docs.deeppavlov.ai/en/latest/features/models/NER.html#7.-Models-list>
5. 19 entities for 104 languages: A new era of NER with the DeepPavlov multilingual BERT [Электронный ресурс]. – Режим доступа: <https://towardsdatascience.com/19-entities-for-104-languages-a-new-era-of-ner-with-the-deeppavlov-multilingual-bert-1bfa6d413ea6>
6. Relation Extraction model – Руководство пользователя [Электронный ресурс]. – Режим доступа: <https://docs.deeppavlov.ai/en/latest/features/models/re.html>
7. sparql-client – Руководство пользователя [Электронный ресурс]. – Режим доступа: <https://pypi.org/project/sparql-client/>
8. SPARQL-Burger – Руководство пользователя [Электронный ресурс]. – Режим доступа: <https://github.com/panmitz/SPARQL-Burger>
9. Тезаурус русского языка RuWordNet [Электронный ресурс]. – Режим доступа: <https://ruwordnet.ru/ru>
10. pymorphy2 – Руководство пользователя [Электронный ресурс]. – Режим доступа: <https://pymorphy2.readthedocs.io/en/stable/>

Разработка web-сервиса для анализа ЭКГ на примере распознавания инфаркта миокарда

П. О. Филонов

Студент бакалавриата

А. Л. Гавшин

Старший преподаватель

Введение

Сердечно-сосудистые заболевания являются одной из главных причин смертности во всем мире, а инфаркт миокарда - одна из наиболее распространенных форм таких заболеваний, которая требует быстрой диагностики и лечения. Для распознавания инфаркта миокарда используется метод ЭКГ, который является доступным и широко распространенным способом исследования сердца. Однако интерпретация ЭКГ может быть сложной и требует высокой квалификации специалиста. В данной статье рассматривается разработка web-сервиса, который может использоваться для анализа ЭКГ и распознавания инфаркта миокарда.

Целью работы является создание автоматизированной системы, которая позволит быстро и точно диагностировать инфаркт миокарда на основе анализа ЭКГ. Разработка такой системы может значительно улучшить качество диагностики сердечно-сосудистых заболеваний и способствовать сохранению здоровья и жизни людей.

1. Архитектура приложения

При реализации приложения необходимо построить и придерживаться определённой архитектуры, позволяющей сделать разработку понятной с возможностью гибкого расширения дополнительного функционала в будущем. При грамотном подходе в данной части закладывается надёжный фундамент для успешной реализации сервиса. Как отмечалось ранее, открываются возможности расширения функционала, например, создания приложения общего назначения анализа ЭКГ различных заболеваний.

Итоговая архитектура web-приложения представлена на рис. 1.

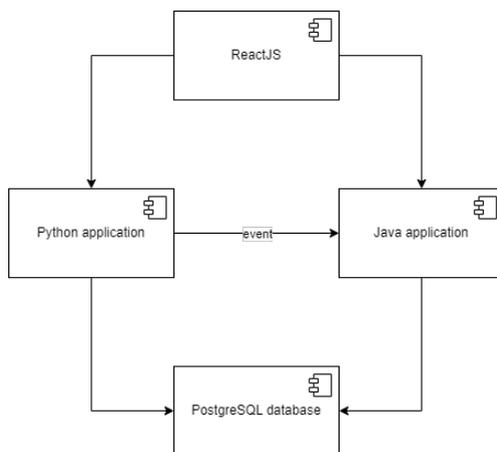


Рис. 1. Архитектура web-приложения

Охарактеризуем выделенные модули.

Python application (сервис обработки сигнала) – backend сервис на языке программирования python. Используется для загрузки файла ЭКГ и преобразования его в первичные «сырые» данные, которые записываются в базу данных и далее будут использоваться в анализе.

Java application (сервис анализа сигнала) – backend сервис на языке программирования java. Используется для анализа данных обработанного в предыдущем сервисе ЭКГ и идентификации заболевания.

ReactJS – клиентская часть приложения, демонстрирующая работу программы. Взаимодействует с обоими backend сервисами для разных нужд (загрузка файлов в «Python application» и получение заключения в «Java application»).

PostgreSQL database – база данных PostgreSQL, используемая для хранения информации об ЭКГ. Рассмотрим подробнее структуру таблиц, отображённую на рис. 2.

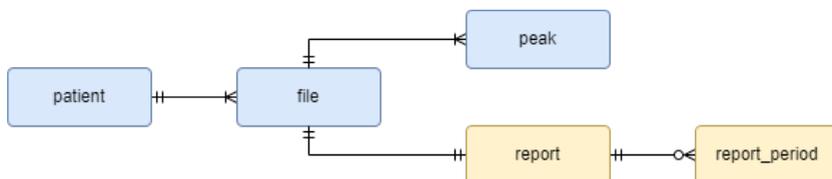


Рис. 2. Структура таблиц базы данных

Блоки с названием таблиц выделены разным цветом. Таким образом, демонстрируется место формирования и записи данных этих таблиц. «patient», «file» и «peak», обозначающие, соответственно, пациента, файл и пик одного из зубца, определяются и сохраняются в «python application» и далее используются в «java application» в готовом виде. Второй сервис формирует «report» и «report_period» - отчёт и период отчёта, в котором указываются промежутки предполагаемого нарушения сигнала.

2. Сервис обработки сигнала

В данном разделе разбирается backend часть, реализованная на языке программирования python. Как уже сказано ранее, она используется для обработки загружаемого файла ЭКГ и формирования данных для анализа. Прежде всего изменяется вид представления сигнала. Он конвертируется в формат JSON и хранится в виде тройки значений: x-координаты, y-координаты и переменной display_order – порядке отображения текущей точки. Таким образом, используя базу данных PostgreSQL и её специальный тип данных – json, появляется возможность хранить сигнал в удобном для всех инструментов виде.

Важная роль сервиса обработки – идентификация пиков зубцов, показывающих деполяризацию и реполяризацию предсердий и желудочков сердца [1]. Преимущество используемого языка программирования в существующих библиотеках обработки ЭКГ сигнала. Для данной цели используется библиотека «neurokit2», которая предоставляет функционал поиска пиков зубцов в зависимости от выбранного метода. В рамках данной задачи поиск основан на формировании пороговых значений, которые определяются на базе сигнала и его изменчивости. Метод является эмпирическим [2].

Архитектура сервиса в виде диаграммы представлена на рис. 3.



Рис. 3. Архитектура сервиса «python application»

Переданный сигнал и информация о нём, в частности, о пациенте, попадает в контроллер – API. На этом этапе есть возможность обработать разные виды сигналов ЭКГ. Для повышения точности информации на текущей стадии за основу взят формат EDF, используемый для хранения и обмена медицинскими данными разного

типа. Он является стандартом в Европе и широко используется в медицинских исследованиях и клинической практике.

Контроллер вызывает сервис, в котором обнаруживаются необходимые данные, описанные выше. Далее сигнал приводится в удобный для обработки вид. Сервисы, в свою очередь, для записи обращаются к репозиторию, содержащему прямой доступ к базе данных.

3. Сервис анализа сигнала

Сервис, реализованный на языке программирования java, отвечает за часть анализа ЭКГ и поиск возможных нарушений сигнала. Он основан на фреймворке «Spring Boot» с архитектурой, слою которой распределены так же, как и в сервисе обработки сигнала (рис. 3). Добавляются дополнительные идентификаторы для обнаружения отклонений в ЭКГ, характерных признакам инфаркта миокарда. Диаграмма их классов представлена на рис. 4.

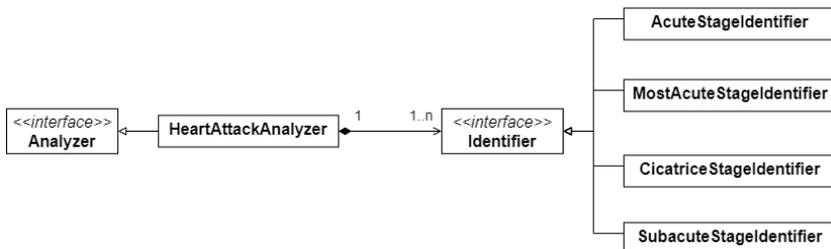


Рис. 4. Диаграмма классов идентификаторов

Интерфейс анализатора имеет единственную реализацию, отражающую поставленную задачу в распознавании инфаркта миокарда. Имплементация обладает списком идентификаторов, каждый из которых выражает одну из стадий болезни [3]:

- острейшая;
- острая;
- подострая;
- рубцовая.

Анализ осуществляется по следующим признакам:

- формирование патологического зубца Q;
- подъём сегмента ST.

Для распознавания признаков как нарушений они должны удовлетворять ряду критериев:

- наклон сегмента ST;
- амплитуда зубца Q и сегмента ST;

- продолжительность зубца Q;
- вид отведения;
- пол пациента;
- возраст пациента.

При соблюдении факторов какого-либо идентификатора сохраняется информация в отчёт с периодами (временем) нарушения для дальнейшего пояснения заключения.

В качестве примера рассмотрим алгоритм распознавания патологического зубца Q в виде блок-схемы на рис. 5.

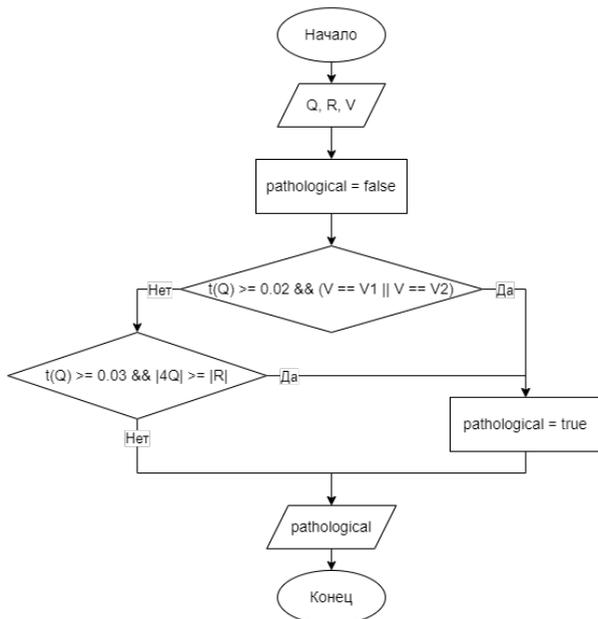


Рис. 5. Блок-схема алгоритма распознавания патологического зубца Q

Как видно из блок-схемы, на вход подаются зубцы Q и R, а также тип отведения рассматриваемого сигнала. Результатом является логическая переменная. Проверяется условие, на длительность Q зубца и отведение V1 или V2. Другое условие – продолжительность Q зубца и сравнение его амплитуды пика с амплитудой пика R зубца. Зубец Q считается патологическим, если верно хотя бы одно из условий.

4. Тестирование

Тестирование приложения проведено на тестовой базе ЭКГ от РОХМИНЭ – общества, которое ведёт работу по созданию инструментария для объективного тестирования автоматического анализа ЭКГ [4]. Их выборка содержит сигналы на различные случаи заболевания, в том числе инфаркта миокарда. Удобство данной выборки также заключается в формате представления ЭКГ, а именно в EDF. Проведённое тестирование, результаты которого представлены в таблице, охватывает распознавание признаков без учёта стадий.

Таблица

Результаты тестирования

Распознано/Признак	Подъём сегмента ST	Патологический зубец Q	Другой признак/отсутствует
Верно распознано	8	20	16
Ошибочно распознано	2	0	4

Общая тестовая выборка:

- 10 с подъёмом сегмента ST;
- 20 с патологическим зубцом Q;
- 20 с другим признаком или его отсутствием (не должно распознавать инфаркт).

Таким образом, из проведённого тестирования вышел результат удачного распознавания 88%. Наибольшее число ошибок приходится на неверную интерпретацию других признаков. Всё же, итог оценивается положительно, так как получилось обнаружить почти все случаи инфаркта миокарда.

Необходимо с развитием сервиса проводить тестирование с выборками большего объёма, учитывая стадии, а также применяя на реальных сигналах медицинской практики.

Заключение

В данной статье был описан процесс разработки web-сервиса для анализа ЭКГ на примере распознавания инфаркта миокарда. Изучены возможные признаки и критерии, реализованы алгоритмы на их основе. Проведено тестирование, в результате чего выявлена точность распознавания инфаркта миокарда на уровне 88%. Поставлены планы дальнейшего развития. Разработанный web-сервис может быть полезен в медицинской практике для быстрой диагностики инфаркта миокарда.

Список литературы

1. Электрокардиограмма. План расшифровки ЭКГ [Электронный ресурс]: статья. – Режим доступа: <https://www.happydoctor.ru/info/536/>
2. Большаков, Т. Ю. Особенности электрокардиографии при сочетании синдрома ранней реполяризации желудочков и инфаркта миокарда / Т. Ю. Большаков, В. А. Шульгин // Кардиология. – 1988. – №11. – С.116-119.
3. К вопросу о трудностях ЭКГ диагностики инфаркта миокарда [Электронный ресурс]: статья. – Режим доступа: <https://cyberleninka.ru/article/n/k-voprosu-o-trudnostyah-ekg-diagnostiki-infarkta-miokarda>
4. Базы данных РОХМИНЭ [Электронный ресурс]: база данных. – Режим доступа: <http://rohmine.org/baza-dannykh-rokhmine/>

Визуализация семантики текста на основе графа знаний

В. А. Филонова

Студент бакалавриата

В. В. Гаршина

Доцент

Введение

Вследствие увеличения информационного обмена данными выросла необходимость использовать автоматическую обработку текста, имеющую в качестве преимущества высокую скорость, возможность обрабатывать большие данные при малом количестве затраченных усилий.

Одной из основных проблем при работе с данными является выбор структур для их организации. Существует множество различных способов хранения данных, которые используются в информационных технологиях: массивы, списки, таблицы. Для анализа подобных структур необходимо писать запросы к базе данных, дополнительные поисковые методы.

Гораздо удобнее представить граф, узлами которого являются сущности и их характеристики, а ребрами – отношения между ними. При таком способе хранения информации сразу можно увидеть кластеры наиболее часто взаимодействующих друг с другом сущностей, перейти от одной сущности к другой, выявив их отношения. Графовое представление помогает лучше понять процессы, происходящие в предметной области, оно является самым удобным инструментом навигации и аналитики.

Данная работа посвящена реализации визуализации графа знаний для семантического анализа текстов о договорной деятельности компаний.

1. Анализ предметной области

Тексты в области договорной деятельности компаний включают большие объемы слабоструктурированной информации, которая может быть представлена в виде договоров, соглашений, контрактов, актов и других документов. Эти документы содержат различные данные,

включая правовые условия, должности сотрудников, информацию о компаниях, сроки исполнения, цены, платежи и т.д.

Для компьютерной обработки и анализа такой информации необходимы специальные алгоритмы и методы, которые позволяют извлечь из документов нужную информацию и представить ее в удобном виде. [1] Решение данной задачи может быть использовано для обработки договорной документации компаний, позволяя автоматически анализировать и классифицировать документы, извлекать ключевые слова и фразы, а также определять связи между документами и компаниями и информацию о них.

Полученная информация может быть использована для различных целей, включая мониторинг исполнения договорных обязательств, анализ деятельности компаний, определение рисков и многие другие. При этом важно обращать внимание на правовые и конфиденциальные аспекты обработки договорной информации, а также использовать специализированные инструменты и программы для обработки такого рода данных.

2. Методы и технологии обработки текста

Решение практически любой задачи обработки текста так или иначе включает в себя проведение анализа текста на нескольких уровнях представления:

1. Графематический анализ.
2. Морфологический анализ.
3. Синтаксический анализ.
4. Семантический анализ.

Для автоматического анализа текстов существует множество методов, выбор конкретного метода зависит от заданных требований и предметной области.

Например, есть методы, основанные на математической статистике и используются для анализа статистических закономерностей в тексте.

Применение технологий искусственного интеллекта имеет большой потенциал в обработке текста. Они могут быть использованы для анализа и классификации текстов, поиска информации, извлечения знаний.

Обширный подраздел искусственного интеллекта, изучающий методы построения алгоритмов, способных обучаться, т.е. улучшать свое поведение в процессе накопления информации - машинное обучение, используется для анализа и классификации текстов, генерации текстовых данных, автоматического перевода и других задач.

Нейронные сети являются разновидностью методов машинного обучения, которые используются для анализа больших объемов данных

выполняют задачи, такие как распознавание образов, классификация и генерация текста.

Правила и грамматики используют заранее заданные правила для извлечения информации из текста и приведения его к нужному формату.

3. Сравнительная характеристика решений

Первым шагом модель обнаружение ладони выполняет точную сегментацию ладони на входном изображении. Затем модель ключевых точек рук предсказывает координаты 21 ключевой точки руки по сегментированной области ладони. Последний шаг состоит в распознавании жеста моделью классификатора, которая принимает на вход образ жеста, представленного в виде вектора признаков (координат ключевых точек) и возвращает метку класса для данного образа.

В табл. 1 представлено сравнение библиотек, способных решать задачу выделения фактов из текста.

Все перечисленные библиотеки являются доступными и эффективными. Natasha и Yargy Parser более легкие и простые в использовании. Но Tomita Parser и DeepPavlov имеют больший функциональный спектр и могут быть использованы для более сложных задач.

Для задачи обработки слабоструктурированных текстов, какими являются документы в области договорной деятельности компаний, проблема скорости распознавания больших данных нивелируется.

Таблица 1

Сравнение библиотек для выделения фактов из текста

Характеристика	DeepPavlov	Natasha	Tomita parser	Yargy parser
Доступность	++	++	++	++
Качество	+	++	-	++
Скорость	+	++	+	+
Простота	-	+	-	++

Наиболее подходящим вариантом для данной предметной области является применение yargy parsera[2].

В табл. 2 представлено сравнение библиотек для визуализации онтологий.

Из представленной таблицы можно сделать вывод, что Ontodia является лучшим выбором среди рассмотренных библиотек для визуализации графов [3].

Сравнение библиотек для визуализации онтологий

Характеристика	Ontodia	WebOwl	VisuOWL	VOWL
Поддерживаемые стандарты	OWL RDF	OWL RDF	OWL	OWL RDF
Наличие интерактивного редактора	++	+	+	++
Визуальные возможности	++	+	+	++
Импорт и экспорт	++	+	+	+
Простота	++	+	++	++

Ontodia обладает следующими преимуществами по сравнению с другими библиотеками:

1. Она имеет высокую оценку по легкости использования, что делает ее очень привлекательной для начинающих пользователей.
2. Она также обладает интерактивным редактором, который позволяет изменять и модифицировать графы в режиме реального времени.
3. Ontodia поддерживает стандарты OWL и RDF, что делает ее полезной для работы с онтологиями и связанными с ними данными.
4. Библиотека обладает высокими визуальными возможностями, что позволяет использовать ее для решения многих задач.
5. Наконец, Ontodia также обеспечивает удобный импорт и экспорт данных, что позволяет эффективно работать с различными форматами данных.

4. Применяемые технологии

В качестве языка программирования был выбран язык python. Разработка приложения будет производиться в среде google colabatory. Используемой библиотекой для выделения фактов из текста стал yargy parser. Библиотека Networkx будет использоваться для формирования графа, а для работы с файлами rdflib. Визуализация графа знаний будет производиться с помощью библиотеки Ontodia.

5. Реализация

На вход система получает текст, содержащий информацию о договоре между компаниями, который затем будет подвергаться обработке. С помощью правил выделяются факты о компаниях, из них формируется структурированная информация, являющаяся основой для

графа. Созданный граф записывается в файл и отправляется для визуализации. На рис. 1 изображен процесс обработки текстовых данных в системе.



Рис. 1. Процесс обработки текстовых данных в системе

Было реализовано выделение фактов из текста. На рис. 2 представлен простой пример выделения названий организаций.

```
TYPE_ORG = morph_pipeline([
    'организация',
    'компания',
    'контора',
    'фирма',
    'ПАО',
    'АО',
    'ИП',
    'ООО',
    'ЗАО',
    'ОАО'
]).interpretation(Organization.type)
QUOTE = or_(eq('«'), eq('»'), eq('\''), eq('\"'), eq('<<'), eq('>>'))
NAME_ORG = rule(
    QUOTE,
    RU.repeatable(),
    QUOTE
).interpretation(Organization.name.normalized())
ORG = rule(
    TYPE_ORG.optional(),
    NAME_ORG
).interpretation(Organization).interpretation(Full.org)
```

Рис. 2. Грамматика для парсера на примере поиска названий организаций

Для этой задачи были созданы правила на основе грамматических конструкций и шаблонов и интерпретаторы, создающие структурированную информацию на основе выделенных знаний. Затем эти правила с помощью парсера были применены к тексту.

В результате этих действий из текста выделились структурированные факты, являющиеся основой для построения графа (рис. 3).

```
org=Org(  
  type='ООО',  
  name='«юнитрест»'  
),  
  
org=Org(  
  type='ОАО',  
  name='«викитэк»'  
),
```

Рис. 3. Созданная структурированная информация на основе извлеченных знаний

Теперь происходит наполнение графа, добавляются необходимые ребра и вершины (рис. 4).

```
for match in matchesOrgInfo:  
    # Выделение информации об организации  
    org_name = match.fact.org.type+' '+match.fact.org.name  
    if org_name is not None:  
        G.add_node(org_name, node_class='organisation')
```

Рис. 4. Наполнение графа

В результате получается граф знаний, который включает в себя выделенную информацию. Далее этот граф необходимо сохранить в файл формата owl. После создания онтологии ее надо загрузить в локальную конечную точку SPARQL, после чего запустить библиотеку Ontodia с входными параметрами, содержащими путь до конечной точки (рис. 5).

```
set SPARQL_ENDPOINT=https://library-ontodia-org.herokuapp.com/sparql  
npm run demo
```

Рис. 5. Запуск Ontodia

После настройки подключения к конечной точке SPARQL в Ontodia должна отобразиться онтология в виде графа. Можно использовать интерактивный редактор для изменения и модификации онтологии.



Рис. 6. Визуализация графа в Ontodia

Заключение

Данная статья посвящена реализации системы для обработки и визуализации текстовых данных. Описаны преимущества использования графового представления данных, методы и технологии в области обработки естественного языка, программные средства для выделения фактов из текста и создание системы. В итоге данной разработки получилось рабочая программа для визуализации семантики текста на основе графа знаний. Использование данного решения упрощает дальнейший анализ и изучение документов в области договорной деятельности компаний.

Список литературы

1. Автоматическая обработка текстов на естественном языке и анализ данных : книга / Е. И. Большаков[и др.]. – М. : Изд-во НИУ ВШЭ, 2017.– 269с.
2. Yargy – Руководство пользователя [Электронный ресурс] : Документ является составной частью проекта Natasha. – Режим доступа: <https://github.com/natasha/yargy>
3. Ontodia – Руководство пользователя [Электронный ресурс]. – Режим доступа: <https://github.com/metaphacts/ontodia>

Управление системой полива тепличного комплекса с учетом погодных условий

Д. Е. Хныкин

Студент бакалавриата

Н. К. Самойлов

Старший преподаватель

Введение

подавляющее большинство тепличных комплексов не оснащены логикой, с помощью которой можно было бы управлять автоматикой теплиц дистанционно, без физического участия оператора. Таким образом, разработка информационной системы и функционирующего макета устройства, обеспечивающего сбор данных, их передачу, обработку, анализ и управление автоматикой теплиц, представляет собой актуальную задачу. Устройство (управляющая плата и набор периферийных устройств) должно быть выполнено в компактном корпусе с учетом особенностей места расположения его в теплицах, т.е. должно быть устойчиво как минимум к брызгам воды и условиям повышенной влажности, что соответствует международному стандарту степени защиты (International Protection) IP54 (ГОСТ 14254-2015) [1]. При этом необходимо предусмотреть своевременный сбор и обработку информации с периферийных устройств с целью прогнозирования климата внутри теплиц, в том числе и для экономии ресурсов.

1. Цель исследования

Разработка функционирующего макета устройства, которое обеспечило бы сбор, обработку, передачу данных и управление теплицами в тепличном комплексе.

2. Принцип работы информационной системы

Принцип работы информационной системы основывается на реализации и взаимодействии трех ключевых частей.

- Аппаратная часть: выполненное на печатной плате устройство, управление которым осуществляет контроллер на базе процессора с RISC-архитектурой, и его обвязка в виде периферийных датчиков для сбора информации.

- Серверная часть: набор микросервисов, реализованный на базе Spring Framework для обработки информации.
- Клиентская часть: клиент для управления системой в виде Android-приложения для просмотра состояний и управления теплицами.

Печатаная плата для аппаратной части информационной системы разработана самостоятельно, разведена в среде автоматизации проектирования электроники, включающая в себя редактор принципиальных схем – EasyEDA, выполнена на фольгированном стеклотекстолите, места под компоненты помечены с помощью шелкографии, как показано на рис. 1.

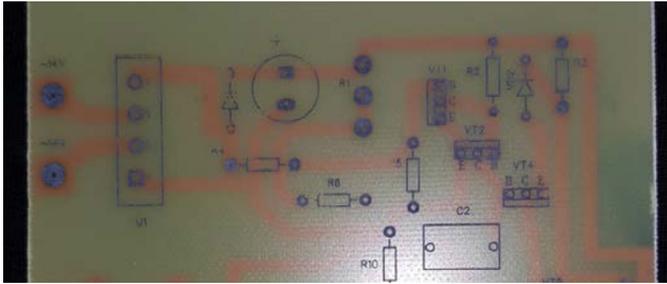


Рис. 1. Шелкография на печатной плате

Серверная часть приложения написана на языке Java с использованием Spring Boot Framework, включая технологии Spring Security и Spring Data.[2-3] Внутренняя архитектура приложения состоит из трех уровней: слоя контроллеров, слоя сервисов и слоя репозитория.

Слой контроллеров обеспечивает взаимодействие сервера с клиентом с помощью REST API, как показано на рис. 2. Задача контроллера – получить HTTP-запрос от клиента, передать его в слой сервиса. После необходимой обработки данных запроса, контроллер формирует и возвращает клиенту HTTP-ответ. Слой сервисов отвечает за всю бизнес-логику приложения. В нём реализована вся функциональность, используемая приложением для манипуляции данными, как входящих из запросов клиента, так и из базы данных.[4] На рис. 3 показана схема архитектуры приложения.

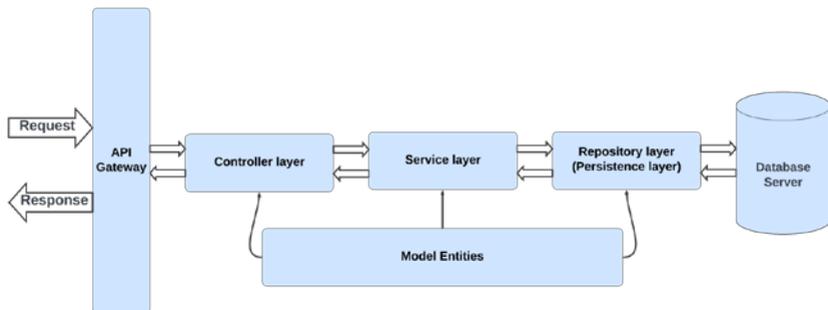


Рис. 2. Схема внутренней REST API архитектуры приложения

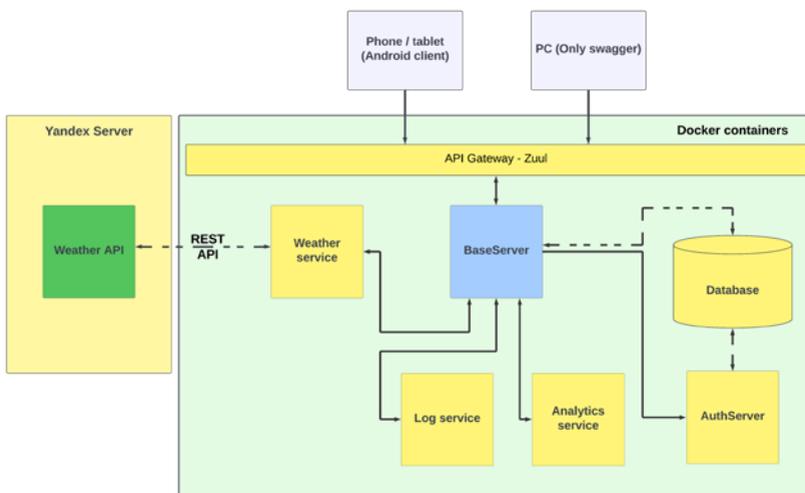


Рис. 3. Схема архитектуры приложения

Описание компонентов:

- Weather API – api от компании Яндекс, для получения информации о погоде;
- Weather Server – внутренний сервис для работы с данными о погоде;
- Auth Server – решение на основе сервера Keycloak;
- API Gateway – основанный на JVM маршрутизатор и серверный балансировщик нагрузки. Необходим для получения единой точки входа для сервисов;

- Database – в качестве базы данных будет использоваться PostgreSQL.

Каждый микросервис и/или компонент имеет свою схему для минимальной связности.

Клиентская часть информационной системы, выполнена в виде android-приложения.

Рассмотрим вариант использования информационной системы. На комплекс теплиц, или на единичных теплицах, а так же на открытых участках почвы, с засеянными на них культурами, за которыми требуется уход, размещается одно управляющее устройство, выполненное в рамках данной работы, которое имеет корпус, защищенный по стандарту IP54, во избежание попадания влаги на электронные компоненты. И опциональный набор дополнительных устройств, таких как: насос, ультрафиолетовая лампа, увлажнители воздуха и т.д. Управляющее устройство имеет набор датчиков (конфигурируется в зависимости от места применения) и доступ к сети интернет. Устройство получает данные с датчиков, которые обрабатываются микроконтроллером и отправляются на общий сервер, где происходит их анализ и обработка.[5-6]

Таким образом, на сервере осуществляется конвергенция двух потоков информации:

- поток информации с датчиков теплицы;
- поток информации с серверов сервиса Яндекс.Погода.

В результате обработки на стороне сервера информации из этих двух потоков, появляется возможность прогнозировать и выстраивать план работы устройства для создания благоприятной среды внутри теплицы.

Для управления системой предусмотрена клиентская часть в виде android-приложения для конечного пользователя. В приложении реализован полный цикл использования от регистрации нового пользователя и регистрации им конечного устройства, до просмотра статистики и управления устройством и подключенных к нему дополнительных устройств. Несмотря на наличия автоматического режима работы системы, пользователь вправе вручную изменять состояния устройств.

3. Особенности работы с API Яндекс.Погода

Для работы с API нужно сначала получить в личном кабинете разработчика уникальный ключ. Чтобы получить информацию о погоде на сервер яндекса нужно отправить следующий GET-запрос, представленный в листинге. Параметры запроса к API Яндекс.Погода представлены в табл. 1.

Формат запроса к API Яндекс.Погода

```
GET /v2/forecast?X-Yandex-API-Key=key HTTP/1.1
Host: api.weather.yandex.ru
```

Таблица 1

Параметры запроса к API Яндекс.Погода

Поле	Описание	Тип
lat	Широта в градусах. Обязательное поле.	double
lon	Долгота в градусах. Обязательное поле.	double
lang	Сочетания языка и страны, для которых будут возвращены данные погодных формулировок.	String
limit	Количество дней в прогнозе, включая текущий.	integer
hours	Для каждого из дней ответ будет содержать прогноз погоды по часам.	boolean
extra	Расширенная информация об осадках.	boolean

Такой запрос позволяет узнать фактическое значение и прогноз погоды для указанного населенного пункта.

Далее, на серверной стороне необходимо описать класс, который обрабатывает ответы API и сопоставляет поля ответа, с необходимыми полями в классе. Это нужно для того, чтобы далее с информацией о погоде можно было работать, как с обычными Java-объектами.

Ответ на запрос возвращается в формате JSON. Информация, которая содержится в ответе, представлена в табл. 2.

Формат ответа API Яндекс.Погода

Поле	Описание	Формат
now	Время сервера в формате Unixtime.	Число
now_dt	Время сервера в UTC.	Строка
info	Объект информации о населенном пункте.	Объект
fact	Объект фактической информации о погоде.	Объект
forecasts	Объект прогнозной информации о погоде.	Объект

4. Конструкционные особенности устройства

Аппаратная часть информационной системы состоит из следующих частей:

- печатной платы, на который размещен микроконтроллер и необходимая для него обвязка из электронных компонентов, набор реле, для управления нагрузкой и их обвязка, а так же компоненты для реализации питания платы и необходимые разъемы для подключения периферийных устройств (датчиков и модулей);
- датчика температуры, давления, влажности и освещенности;
- блока питания.

Обвязка микроконтроллера подразумевает использование wi-fi модуля, который необходим для передачи данных на сервер. Для этого, соответственно, устройству необходимо предоставить доступ к сети интернет.

Сигнал с датчиков приходит в микроконтроллер, который записывает информацию о показаниях и отправляет её на сервер.

Заключение

Разработана информационная система, которая обеспечивает сбор, обработку, передачу данных для управления климатом теплиц в тепличном комплексе или участками, которые требуют ухода и поддержания благоприятной среды почвы. Первый этап развития системы представлен на GitLab.[7] Перспективы дальнейшей разработки связаны с созданием системы интегрированной в тепличный комплекс, повышением отказоустойчивости всей системы, повышением качества системы для конечного пользователя.

Список литературы

1. Межгосударственный стандарт ГОСТ 14254-2015 (МЭК 60529-2013) «Степени защиты, обеспечиваемые оболочками (код IP)» [Текст] : ГОСТ 14254-2015 от 10 декабря 2015г. № 48-2015. – Взамен ГОСТ 14254-96 (МЭК-529-89); введ. 01.03.2017г. – Москва: Стандартинформ, 2016. – 39 с.
2. Раджпут, Д. Spring. Все паттерны проектирования / Д. Раджпут. – Санкт-Петербург : Питер, 2019. – 320 с.
3. Карнелл, Д. Микросервисы Spring в действии / Д. Карнелл, У. Санчес Иллари, пер. с англ. А. Н. Киселева. – Москва: ДМК Пресс, 2022. – 490 с.
4. Ahmed, I. PostgreSQL 10 High Performance / I. Ahmed, G. Smith, E. Pirozzi. – Birmingham: Packt Publishing, 2018. – 508 с.
5. Водолазов, А. Микроконтроллеры для систем автоматки. Учебное пособие / А. Водолазов. – Вологда : Инфра-Инженерия, 2022. – 168 с.
6. Петин, В. Новые возможности Arduino, ESP, Raspberry Pi в проектах IoT / В. Петин. – Санкт-Петербург: БХВ-Петербург, 2022. – 319 с.
7. Программный код серверной и клиентской части информационной системы [Электронный ресурс] : GitLab-проект – Режим доступа: <https://gitlab.com/khnykin-greenhouse>

Разработка проекта «List_of_documents» для реализации программ дополнительного образования ФГБОУ ВО «Воронежского государственного университета»

Т. А. Шорохова

Студент бакалавриата

Е. А. Копытина

Старший преподаватель

Введение

Современный ритм жизни подразумевает выполнение большого количества рутинных задач, которые занимают много времени и сил, убивают производительность и мотивацию. Люди перегружают свой график задачами и не уделяют достаточно времени для отдыха и восстановления сил. Это может привести к усталости, стрессу, плохому качеству работы, серьезным смысловым или орфографическим ошибкам. Что при работе с документами вышеуказанное может повлечь за собой серьезные проблемы как для исполнителя: выговоры, материальные потери, а также вероятность увольнения; так и для всего учебного заведения: штрафы, для финансовых документов – расхождения с бюджетом. Проект, описанный в данной статье, разработанный для дополнительного образования ФГБОУ ВО «ВГУ», является крайне актуальным и востребованным, так как он способен решить столь серьезную в наше время проблему, проблему тайм менеджмента и минимизации ошибок при работе с текстовыми документами. Аналогичные подходы к решению такого рода задач были подробно рассмотрены в [1-6].

1. Постановка задачи

К описываемому в статье приложению «List_of_documents» были выдвинуты следующие требования:

- Аутентификация пользователя.
- Регистрация пользователя
- Возможность создания на основе данных из excel-таблицы документа зачисления на курс для выбранного направления;
- Возможность создания на основе данных из excel-таблицы документа отчисления с курса для выбранного направления.

- Возможность создания на основе данных из excel-таблицы документа служебная записка сразу для выбранных направлений подготовки.
- Сохранение необходимого документа в зависимости от выбранного вида документа и направления для зачисления/отчисления.

2. Реализация логики

Приложение написано на объектно-ориентированном языке Python. Для реализации были использованы следующие модули:

- Flask,
- SQLAlchemy,
- python-docx,
- openpyxl.

Разработанное приложение имеет клиент-серверную архитектуру.

Клиент и сервер взаимодействуют посредством обмена HTTP-запросов.

Серверная часть представлена следующими модулями, содержащимися в папках `list_of_documents/controllers`, `list_of_documents/entities` и `list_of_documents`:

- `forms.py`, где расположены классы форм регистрации, логина,
- `UserController.py`, где реализована логика регистрации и авторизация пользователя,
- `FileController.py`, где реализована логика сохранения необходимого нам документа в зависимости от параметров,
- `config.py`,
- `main_config.py` служит для инициализации и подключения БД,
- `users.py`, где реализованы классы задач, пользователя.

Также разработан модуль, запускающий приложение – `main.py`.

В основном модуле `UserController.py` были реализованы функции-запросы, взаимодействующие с клиентской частью. Например, функция `api_user_auth()`, которая выполняет проверку аутентификации и позволяет выполнить выход на главную страницу.

Клиентская часть представлена следующими JavaScript скриптами, содержащимися в папке `list_of_documents/templates`:

1. `authorization.html`,
2. `registration.html`.

При их реализации была применена библиотека для работы с БД – `sqlalchemy`.

Также были разработаны следующие представления пользовательского интерфейса, содержащиеся в папке `list_of_documents/templates`:

1. index.html,
2. authorization.html,
3. registration.html,
4. header.html,
5. conections.html.

И CSS стили, содержащиеся в папке list_of_documents/static - main.css

3. Реализация интерфейса

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

Зарегистрироваться'." data-bbox="327 287 645 457"/>

Рис. 1. Страница авторизации пользователя

Чтобы авторизоваться предварительно необходимо зарегистрировать в системе при помощи почты Воронежского Государственного университета и пароля. Вид окна регистрации представлен на рис. 2.

Авторизоваться'." data-bbox="332 592 661 787"/>

Рис. 2. Страница регистрации пользователя

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

Главная страница Выход

Выберите тип документа
На выбрано

Выберите направление
На выбрано

Сохранить

Рис. 3. Главная страница настроек документов

Для сохранения документов отчисления/зачисления необходимо выбрать тип документа и вид направления пример выбора показан на рис. 4., после чего необходимо нажать кнопку сохранить и файл будет скачан на ПК пользователя как показана на рис. 5.

Главная страница Выход

Выберите тип документа
Зачисление на курс

Выберите направление
Рубль и анализ данных

Сохранить

Рис. 4. Параметры для документов зачисления/отчисления

Главная страница Выход

Выберите тип документа
Зачисление на курс

Выберите направление
Рубль и анализ данных

Сохранить

enrollment_analy...docx Показать все

Рис. 5. Сохранение документа

Сгенерированный с помощью разработанного приложения документ зачисления на курс имеет вид представленный на рис. 6.

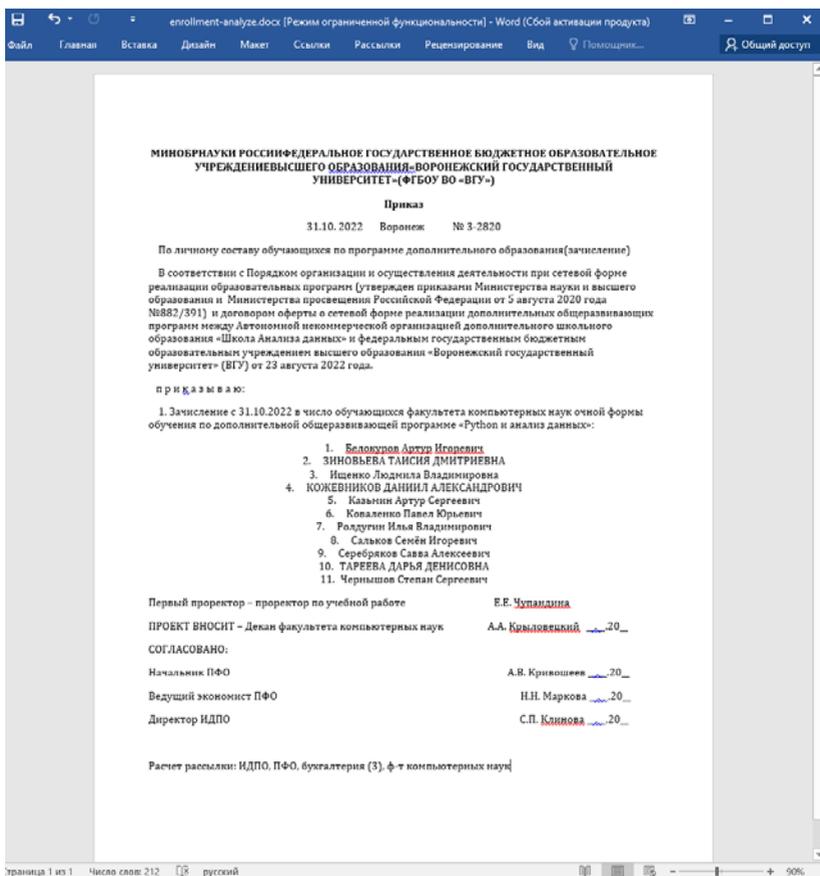


Рис. 6. Документ зачисления на курс «Python и анализ данных»

Закключение

В данной статье была рассмотрена разработка веб-приложения «List_of_documents» для реализации программ дополнительного образования ФГБОУ ВО «ВГУ». Все поставленные в начале разработки приложения функциональные требования были выполнены. Приложение является востребованным и полностью работоспособным.

Список литературы

1. 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».

2. 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. Алгазина, Воронеж, 10–12 февраля 2022 года / Под редакцией Д.Н. Борисова. – Воронеж: Общество с ограниченной ответственностью "Вэлборн", 2022. – С. 990-996.

3. 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.

4. Копытина, Е. А. Реализация программы расчета для оптимизации трассировки тепловых сетей / Е. А. Копытина, Н. А. Петрикеева, Д. М. Чудинов // Жилищное хозяйство и коммунальная инфраструктура. – 2022. – № 2(21). – С. 63-73. – DOI 10.36622/VSTU.2022.21.2.007.

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. Алгазина, 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.

Реализация одной задачи оптимизации на языке программирования Python

Е. С. Юревич

Студент бакалавриата

М. Е. Залыгаева

Старший преподаватель

Введение

Оптимизация и проблема выбора являются самыми обсуждаемыми и актуальными темами в современном мире. Они важны как в обычной жизни людей, так и в различных профессиональных сферах.

Людям часто приходится сталкиваться с проблемой выбора. И принять правильное решение бывает очень сложно. Особых проблем может не возникнуть, когда можно спокойно рассмотреть все варианты, обдумать и выбрать лучшее. Но, к сожалению, так бывает не всегда. Иногда невозможно вернуться к варианту, который уже отклонён, а сделав свой выбор, приходится на нём остановиться [1–3].

Например:

1. Поиск и покупка редкого товара в ограниченном количестве, но желанного многими.
2. Желание нанять на работу хорошего сотрудника, который при отказе захочет уйти в другую компанию.
3. Решение снять подходящую квартиру в городе, где много желающих.
4. Выбор в приложениях на сайтах знакомств.

В данной статье с целью автоматизации принятия подобных решений рассматривается реализация задачи остановки случайных процессов на основе метода решения, который предложил С. М. Гусейн-Заде, на языке программирования Python. Также проведён анализ результатов, полученных с помощью реализованной программы.

2. Условие задачи, сформулированной Мартином Гарднером

Рассмотрим задачу, которая сначала кажется детской, но на самом деле положила начало теории оптимальной остановки случайных процессов. В 1960 году её придумал Мартин Гарднер. Это задача о

разборчивой невесте. На самом деле речь идет о том, чтобы найти наилучшую стратегию выбора из последовательности.

Условия задачи:

- Существует единственное вакантное место;
- Есть известное число кандидатов: n ;
- Общение с кандидатами происходит в случайном порядке, с каждым не более одного раза;
- О каждом кандидате известно, лучше он или хуже любого из предыдущих;
- После общения с кандидатом происходит сравнение его с предыдущими, и либо производится отказ, либо принимается предложение. Если предложение принято, то процесс останавливается. Если происходит отказ, то вернуться к этому варианту уже невозможно.

Цель – максимизировать вероятность выбора наилучшего из n объектов.

3. Описание метода решения

В 1966 г. данный метод предложил С.М. Гусейн-Заде [4, 5]. В основе лежит метод «динамического программирования».

g_t – вероятность «победы» в случае выбора t -го объекта при условии, что элемент t лучше всех предыдущих;

h_t – вероятность получения наилучшего элемента при пропуске первых t элементов.

На рис. 1 представлен процесс поиска зависимости g и h от t .

g_t			$\frac{n-2}{n}$	$\frac{n-1}{n}$	1
t		$n-3$	$n-2$	$n-1$	n

а)

h_t			$\frac{(n-2) + (n-1)}{n(n-1)}$	$\frac{1}{n}$	0
t		$n-3$	$n-2$	$n-1$	n

б)

а) g_t , б) h_t

Рис. 1. Поиск зависимости:

Находим зависимость g от t :

$$g_t = \frac{t}{n}, \quad (1)$$

Находим зависимость h от t

$$h_t = \frac{t}{n} \cdot \left(\frac{1}{t} + \frac{1}{t+1} + \dots + \frac{1}{n-1} \right) \quad (2)$$

Наша цель: на каждом шаге t сравнивать эти вероятности. Если $g(t) > h(t)$, то мы останавливаемся на t -ом объекте. Если $g(t) < h(t)$, то мы пропускаем t -ый объект и переходим к рассмотрению $t+1$ -го. Если $g(t) = h(t)$, то нет разницы, останавливать выбор или пропускать объект, поэтому будем считать, что мы останавливаемся на t -ом объекте.

$$h_t / g_t = \frac{1}{t} + \frac{1}{t+1} + \dots + \frac{1}{n-1} \quad (3)$$

Определяем по графику на рис. 2, что

$$h_t / g_t = 1$$

при

$$t = \frac{n}{e}.$$

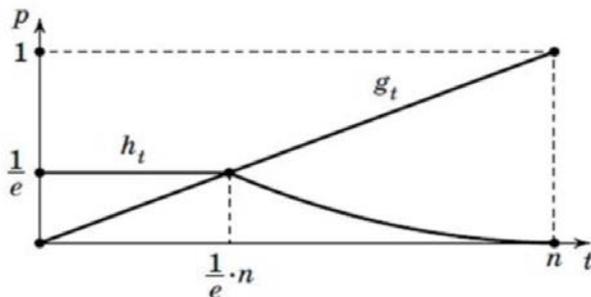


Рис. 2. График $g(t)$ и $h(t)$

$$t^* = \frac{n}{e}, \quad (4)$$

$$h_t = g_t = \frac{1}{e} \approx 0,368 \quad (5)$$

Определяем, что оптимальное число объектов t , которое следует пропустить, равно n/e . Вероятность равна примерно 0,368.

Метод решения: сначала следует пропустить первую $1/e$ часть, только запоминая лучший вариант для будущего сравнения, а дальше выбирается первый элемент, который больше выбранного. При этом вероятность получить в конце концов самый лучший вариант из всех n равна примерно 36,8% [6, 7].

4. Реализация задачи на Python

Представим сравниваемые объекты в виде массива чисел, характеризующих их качество. Возьмём количество объектов, равное 1000. Худший по качеству объект – 1, лучший – 1000.

На рис. 3 представлена блок-схема алгоритма для реализации задачи на Python.

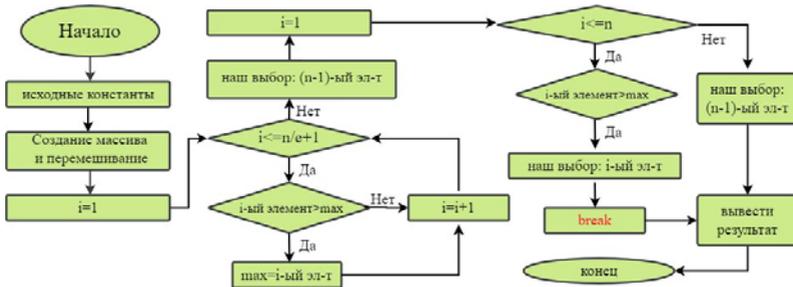


Рис. 3. Блок-схема алгоритма для реализации задачи на Python

Вначале мы вводим необходимые нам константы и переменные, формируем массив объектов и перемешиваем его, потому что очередность случайна. Случайность генерируется с помощью перемешивания массива функцией `shuffle()` модуля `random`. Отвергается первая $1/e$ часть объектов и запоминается лучший из них, то есть максимальное число для будущего сравнения.

Затем, проходя по остальным элементам массива, мы находим первый объект, который больше, чем найденное нами максимальное число, то есть обладающий тем свойством, что он лучше всех предшествующих объектов.

Если находится такое число, то программа прерывает цикл, то есть производит оптимальную остановку и выдаёт нам выбранный объект.

Если же такого числа не находится, программа автоматически выбирает последний элемент исходного массива и выдаёт результат.

5. Анализ результатов

Проведём анализ результатов также с помощью Python. Прогоним реализованную программу 1000 раз и найдём процентную часть определенных результатов: наилучший выбор, топ-3 ниже лучшего и все остальные ниже качеством. Сделаем так 10 раз для усреднения результатов. На рис. 4 представлена диаграмма с процентным содержанием определенных результатов.

В среднем в 1000 прогонок программы:

Наилучший выбор - 36.77 %

Топ-3 ниже лучшего - 23.06 %

Меньше качеством - 40.17 %

Вероятность выбрать один из 4-х лучших объектов - ~ 60%

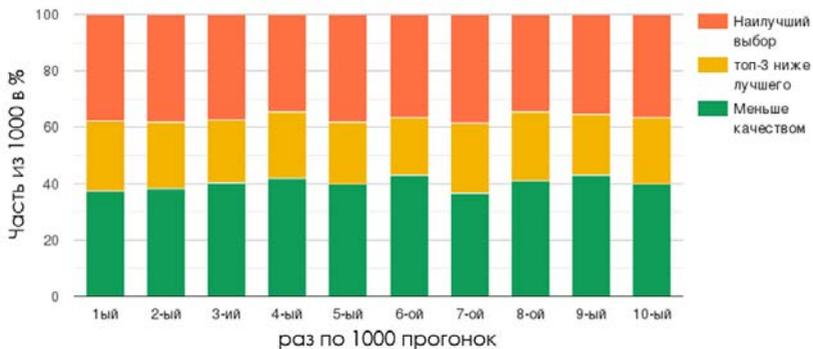


Рис. 4. Диаграмма процентного содержания результатов

Программа работает и в том случае, если в начальном массиве есть одинаковые числа. Также можно рассмотреть эту программу с различным распределением случайных величин.

Заключение

В рамках данной статьи, после изучения механизма решения задачи методом, предложенным С. М. Гусейном-Заде, была построена блок-схема алгоритма для реализации задачи оптимальной остановки случайных процессов, была реализована данная задача оптимизации на языке программирования Python, и также был проведён анализ результатов работы реализованной программы.

Список литературы

1. Вальд, А. Последовательный анализ / А. Вальд ; пер. с англ. П. А. Бакута [и др.] ; под ред. Б. А. Севастьянова. – М. : Гос. изд-во физ.-мат. лит., 1960 . – 327 с.
2. Arrow, K. J. Bayes and minimax solutions of sequential decision problems / K. J. Arrow, D. Blackwell, M. A. Girshick // *Econometrica*. – 1949. – No 17. – P. 213–244.
3. Snell, L. J. Applications of martingale system theorems / L. J. Snell // *Trans. Amer. Math. Soc.* – 1952. – No 73. – P. 293–312.
4. Гусейн-Заде, С. М. Разборчивая невеста / С. М. Гусейн-Заде. – М. : МЦНМО, 2003.
5. Гусейн-Заде, С. М. Задача выбора и оптимальное правило остановки последовательности независимых испытаний / С. М. Гусейн-Заде // *ТВП*. –1966. – 11:3. С. 534–537.
6. Chow, Y. S. *Great Expectations: The Theory of Optimal Stopping* / Y. S. Chow, H. Robbins, D. Siegmund. – Boston : Houghton Mifflin Company, 1971. — 534 P.
7. Ширяев, А. Н. Статистический последовательный анализ: Оптимальные правила остановки / А. Н. Ширяев. – М. : Наука, 1969 . – 230 с.

Статьи студентов магистратуры

Защита программы в памяти микроконтроллера для предотвращения клонирования устройства

В. М. Беспалов

Студент магистратуры

А. Ю. Савинков

Профессор

Введение

В настоящее время, практически все сферы нашей жизни в той или иной степени зависят от микроконтроллеров. Эти устройства используются повсеместно, например, в бытовых приборах, таких как стиральные машины и холодильники или в компонентах компьютера, таких как жесткие диски. Микроконтроллеры получили широкое распространение из-за низкой цены, малого энергопотребления и небольших габаритов.

Обычно, стоимость микроконтроллера и окружающих его компонентов в устройстве значительно ниже цены разработки программного обеспечения для работы устройства. Из-за этого стремительно развивается направление защиты данных, записанных в памяти устройства.

Существует множество разнообразных атак [1], позволяющих как получать пользовательские данные, так и целиком клонировать устройство. Клонирование устройства – это копирование программы, сохраненной в памяти одного микроконтроллера, в память другого. В большинстве случаев, клонирование производится с целью дальнейшей продажи, так как затраты на клонирование программного обеспечения значительно ниже затрат на разработки программного обеспечения, что позволяет продавать подделанные устройства значительно дешевле.

В связи с этим, многие производители начали задумываться о том, как защитить память устройства, чтобы было невозможно как произвести клонирование, так и просто прочитать данные из памяти устройства. Примерами таких защит является «Readout protection», которая встроена в некоторые микроконтроллеры производства

компании ST Microelectronics [2], а также защита от подключения отладочного интерфейса, которая используется производителем SiLabs.

Вышеперечисленные механизмы предотвращают чтение данных из памяти устройства. Но, подобные механизмы защиты существуют не на всех существующих микроконтроллерах. В таком случае, предотвращение клонирования становится нетривиальной задачей.

1. Постановка задачи

В данной работе рассматривается возможность защиты программы в памяти микроконтроллера только от клонирования. При этом предполагается, что злоумышленник, производящий копирование не занимается реверс-инжинирингом ввиду низкой экономической целесообразности. Это связано с тем, что реверс-инжиниринг значительно сложнее и, следовательно, дороже, чем разработка программного обеспечения.

Данная работа нацелена на дешевые микроконтроллеры, не имеющие встроенных средств защиты от копирования. В связи с этим, прошивка должна запускаться лишь на том экземпляре микроконтроллера, в память которого программа была записана на заводе изготовителе.

К тому же, предполагается, что разрабатываемый модуль, включенный в состав какой-либо программы, должен быть унифицирован для максимальной простоты прошивки в заводских условиях. Также данный модуль должен иметь минимальное влияние на производительность основной программы.

Дополнительным условием является наличие устройств, выполняющих чтение программы из памяти, подключенной к микроконтроллеру по интерфейсу SPI.

Ввиду описанного выше, к разрабатываемому модулю выдвигаются следующие требования:

- Возможность запуска программы только на том экземпляре микроконтроллера, на который программа была записана на заводе.
- Максимальная унификация прошивки.
- Высокая производительность.
- Возможность чтения программы, через интерфейс SPI.

2. Предполагаемый способ прошивки

Ввиду наличия требования по невозможности запуска одной прошивки на двух разных экземплярах прошивки, а также требования по унификации прошивки, то было принято решение шифровать отдельные части готовой прошивки или всю прошивку в памяти при

первом запуске устройства используя уникальный идентификатор процессора, так как многие процессоры имеют регистр, содержащий такой идентификатор. Такое решение позволяет использовать одну и ту же прошивку для всех устройств одного типа, не позволяя при этом запуск одной и той же прошивки на разных устройствах одного типа.

Соответственно, предполагается следующий алгоритм прошивки устройства на заводе-изготовителе устройства:

1. Запись скомпилированной программы в память микроконтроллера

2. Первый запуск микроконтроллера, выполняющий частичное (или полное) шифрование прошивки.

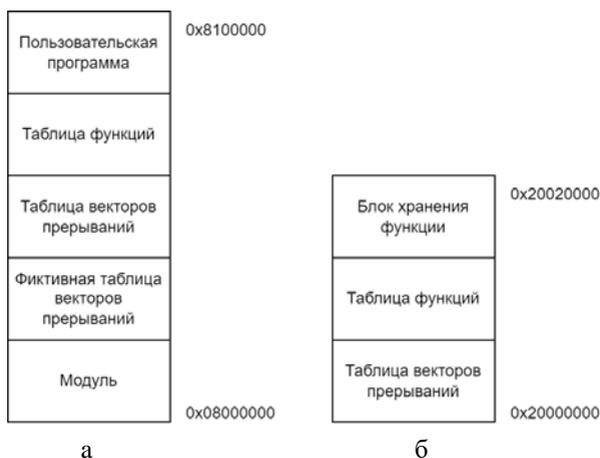
Такой подход позволяет запретить запуск копии прошивки на микроконтроллере, имеющем уникальный идентификатор процессора, отличный от того, который имело устройство, на котором данная программа была впервые запущена. Также данная идея позволяет использовать лишь одну копию прошивки на заводе-изготовителе и не препятствует использованию flash-памяти, подключенной к микроконтроллеру при помощи интерфейса SPI. Высокая производительность обеспечивается полным или частичным копированием расшифрованной программы в оперативную память.

Для обеспечения более гибкой работы модуля, предусматривается шифрование отдельных функций. Такой подход позволяет регулировать производительность устройства, при этом пользователь модуля сам выбирает, какие функции необходимо шифровать. Для этого создана таблица, в которую записаны имена функций и их размеры.

3. Расположение программы в памяти устройства

Ввиду того, что в настоящее время, многие микроконтроллеры способны производить исполнение программы из энергонезависимой памяти, то, при разработке модуля, наличие подобной функции подразумевается.

В соответствии с этим было решено хранить в flash-памяти модуль, а также фиктивную таблицу векторов прерываний, которая необходима для первичной инициализации программы. Вторая таблица векторов прерываний, которая должна использоваться при нормальной работе программы, шифруется при первом запуске так как является одним из важнейших компонентов практически любой встраиваемой системы. Хранение двух экземпляров векторов прерываний позволяет запретить корректный вызов всех прерываний после клонирования. На рисунке обозначено расположение программы в памяти микроконтроллера STM32F407VG [3], на котором проводилась первичная проверка работоспособности модуля.



а – программа в энергонезависимой памяти, б – в оперативной памяти

Рис. 1. Расположение программы в памяти с начальными и конечными адресами

4. Реализация

На текущий момент, для реализации модуля, используется плагин для компилятора gcc, который позволяет заменять заранее заданные функции на вызов основной функции модуля, запускающей расшифровывание целевой функции и последующий вызов по заданному адресу в оперативной памяти.

При таком способе вызова функции была решена проблема передачи аргументов при помощи создания дополнительного стека, который хранит в себе необходимые аргументы, передаваемые в функцию, их количество, а также индекс вызвавшей функции. Такой индекс необходим для расшифровки вызываемой функции до возврата. Как только удаляется последняя запись из дополнительного стека, используется обычный оператор возврата и продолжается выполнение программы в обычном режиме.

Заключение

Данная статья посвящена рассмотрению модуля программного обеспечения, который предназначен для предотвращения клонирования устройства. Описаны архитектура программы, её расположение в памяти, а также основные принципы. Была проведена первичная проверки работоспособности разработанного модуля

Данная реализация имеет следующие достоинства:

- Высокая производительность за счет шифрования отдельных функций.
- Унификация прошивки для выпускаемых устройств за счет шифрования при первом запуске

К недостаткам данного модуля программного обеспечения следует отнести использование плагина для компилятора gcc, что не предполагает компиляцию приложения разработчика иным инструментом. Также, одним из недостатков является недостаточное тестирование модуля, так как он проверялся лишь на трех сравнительно небольших утилитах.

В будущем планируется заменить компилятор gcc на инструмент, получающий на вход скомпилированный elf-файл с отладочной информацией, в который будут вноситься определенные изменения, позволяющие сохранить текущую архитектуру. Также планируется создать набор тестов, содержащий различные сценарии использования, для всеобъемлющей проверки корректности работы.

Список литературы

1. Hezam, A. A. A Comprehensive IoT Attacks Survey based on a Building-blocked Reference Model / D. Konstantas, M. Mahyoub // International Journal of Advanced Computer Science and Applications – 2018 – 9(3) – P. 355-373
2. Proprietary code read-out protection on STM32L4, STM32L4+, STM32G4 and STM32WB Series MCUs [Электронный ресурс]. – Режим доступа: https://www.st.com/resource/en/application_note/an4758-proprietary-code-readout-protection-on-stm32l4-stm32l4-stm32g4-and-stm32wb-series-mcus-stmicroelectronics.pdf
3. DS8626 [Электронный ресурс]. – Режим доступа: <https://www.st.com/resource/en/datasheet/stm32f407vg.pdf>

Специализированный отладчик для программ цифровой обработки сигналов

Е. П. Булавина

Студент магистратуры

А. Ю. Савинков

Профессор

Введение

Несмотря на то, что методы цифровой обработки сигналов базируются на математических принципах, известных миру уже несколько веков, это направление активно развивается и используется в настоящее время. Оно является основой для различных отраслей, таких как системы связи, обработки аудио и видео сигналов, распознавания речи, радиолокации и множества других. [1] Программы цифровой обработки сигналов работают с такими данными, как аудио, видео, радио, к обрабатываемым сигналам так же могут относиться различные данные, преобразованные в цифровой вид. Цифровая обработка сигналов решает задачи спектрального анализа, фильтрации и обработки сигнала. Исходя из задач следует одна из основных особенностей работы программ ЦОС – поточный характер обработки больших объемов данных. Несмотря на то, что операции, проводимые над сигналом, являются достаточно простыми в математическом представлении, объем данных, который требуется обработать программе, может существенно усложнить разработку приложений. В процессе создания программы, разработчики неизбежно допускают ошибки в исходном коде, из-за чего становится необходимым процесс отладки. В программах цифровой обработки сигналов отладка становится трудоемким процессом из-за характера данных. Проблема не так велика если её можно отследить на небольшой выборке, но зачастую неполадка может возникнуть только после прохождения определенного количества времени или же только при работе с большим объемом данных. Поточковый характер так же влияет на сложность отладки, так как для проверки корректности может потребоваться вся последовательность данных, обработка которых вручную может занять часы реального времени. В связи с этим, предлагается разработать

специализированную систему для отладки для программ цифровой обработки сигналов, которая могла бы автоматизировать часть задач процесса поиска ошибок, что могло бы существенно уменьшить время, затрачиваемое разработчиками на создание и наладку работы программного обеспечения.

1. Постановка задачи

Целью данной работы является разработка системы, позволяющей автоматизировать процесс отладки программ цифровой обработки сигналов. Для упрощения выполнения работы были выделены следующие этапы:

- анализ традиционных способов отладки;
- выделение требований к системе отладки программ цифровой обработки сигналов;
- реализация сбора отладочных данных;
- автоматизация анализа данных;
- интеграция системы отладки в среду разработки.

В данной работе затрагиваются первые три этапа.

2. Пошаговая отладка

Один из основных подходов к отладке программ – пошаговая отладка, позволяющая проанализировать состояние системы во время остановки её работы с выполнением одной операции за шаг. Данный способ предполагает использование точек останова для выбора места, с которого начнется отладка.

Данный процесс можно описать следующим способом. Перед началом отладки разработчик выбирает место, в котором будет располагаться точка останова, уже после чего происходит запуск программы с подключенным отладчиком. Затем, по достижению точки останова и поставленных условий, выполнение программы приостанавливается и её состояние замораживается. [2] В данный момент можно свободно просматривать содержимое стека и регистров, а также переходить на следующий шаг выполнения программы.

Несмотря на то, что данный способ отладки является наиболее наглядным и дает наиболее полное понимание работы программы, в нем можно выделить несколько недостатков, из-за которых затрудняется отладка программ цифровой обработки сигналов. Среди них можно отметить:

- значительное время, требуемое для отладки в случае обработки большого количества последовательных данных;

- отсутствие данных о предыдущем состоянии программы, что может потребоваться в случае, если необходимо знать всю предыдущую последовательность информации;
- нарушение таймингов работы программы, предотвращающие возможность отладки приложений, получающих и обрабатывающих данные в реальном времени.

3. Трассировка

Другой подход к отладке предполагает использование трассировки, позволяющей собирать отдельные данные во время выполнения программы и анализировать их.

По сравнению с пошаговой отладкой, данный способ решает часть проблем. Так, трассировка сохраняет всю последовательность данных, что дает представление о работе программы в целом, а не в отдельный момент выполнения. Нивелируется и время, затраченное на достижение точки выполнения программы, в которой возникла проблема. Несмотря на то, что ведение трассировки также оказывает влияние на тайминги приложения, оно все еще будет продолжать работу, позволяя наблюдать за ним в реальном времени.

Стоит отметить, что при этом у данного способа имеются и свои недостатки:

- для работы трассировки как правило требуется написание дополнительного кода;
- трассировка может не сохранить информацию, требуемую для локализации проблемы;
- изменение данных, выводимых трассировкой, предполагает перекомпилирование программы и её перезапуск, что может занять большое количество времени.

4. Требования к системе отладки

Исходя из характера работы программ цифровой обработки сигналов и особенностей использования различных способов отладки, можно выделить следующие требования для разрабатываемой системы:

- отсутствие необходимости вносить изменения в исходный код работы программы для каждой итерации отладки, чтобы предотвратить перекомпилирование приложения;
- повышенная скорость отладки, позволяющая обрабатывать большой массив данных за время, близкое к реальному;
- автоматизация анализа данных для предотвращения ручной обработки получаемой информации;
- сохранение предыдущих значений наблюдаемых параметров для просмотра и анализа изменяемой во времени информации;

- возможность остановки выполнения программы в произвольный момент времени для перехода к пошаговой отладке, с возможностью вернуться к предыдущему режиму;
- возможность визуализации обрабатываемых данных;
- интеграция системы отладки со средой разработки для удобства использования.

5. Реализация

Предполагается, что программа цифровой обработки сигналов, требующая отладки, написана на языке C, как одном из наиболее популярных для решения подобных задач. Из-за этого, основой для данной системы служит уже существующий отладчик – GDB.

GDB предоставляет множество средств для расширения функциональности отладчика. Среди них стоит уделить особое внимание Python API – интерфейс, позволяющий переопределять и расширять поведение GDB, а также создавать скрипты для автоматизации работы отладчика и управлять его поведением. Для этого в Python был предоставлен модуль `gdb`, реализующий методы и классы для работы с GDB. [3]

Для того, чтобы выполнить требования с разрабатываемой системе, потребовалось добавить новый тип точек останова, переопределяющий и расширяющий стандартное поведение. Так, наследуя класс `gdb.Breakpoint`, данная точка останова выполняет следующие действия при её прохождении:

- значения отслеживаемых переменных перенаправляются в отдельный канал, где они могут быть либо записаны в файл, либо отправлены в другую программу, отвечающую за обработку данных, не влияющую на выполнение программы в целом (например, визуализация данных);
- проводится анализ отслеживаемых данных (находится ли значение в пределах ожидаемого);
- при выявлении ошибочных значений, работа программы приостанавливается, после чего поведение точки останова переходит к стандартному до повторного прохождения точки либо завершения работы программы;
- если заданное условие остановки не выполнено при прохождении, программа не останавливает свое выполнение, при этом сохраняя требуемые значения для возможного анализа.

Для работы с данными точками останова требуется возможность создавать объекты данного класса, не внося изменения в исходный код расширения. Для этого была создана команда `st_breakpoint`, позволяющая создавать точки останова данного типа исходя из

параметров, переданных в параметры команды. На текущем этапе работы она принимает следующие параметры:

- название файла и номер строки для создания точки останова;
- список отслеживаемых переменных, которые требуется сохранять для последующего анализа;
- условие в формате строки, при котором выполнение программы будет остановлено.

Так как данные точки останова создаются с помощью команды, добавленной в консоль GDB, это позволит создавать их из других программ, обращаясь к терминалу. Для этого требуется предварительно загрузить разрабатываемое расширение в текущей сессии GDB, что возможно делать в автоматическом режиме.

Данные решения упрощает работу с отладчиком извне, что позволит интегрировать данную систему отладки в среду разработки, поддерживающую расширения. А из-за того, что значения отслеживаемых переменных перенаправляются в отдельно выделенный канал, то визуализация данных может быть выполнена в отдельной программе, что позволит ускорить обработку точек останова из-за проведения визуализации данных в другом процессе, не влияющем на выполнение отладки.

Заключение

Данная статья посвящена первым этапам разработки специализированной системы отладки программ цифровой обработки информации. На данный момент выполнены работы по анализу существующих систем, а также создано расширение, позволяющее сохранять и анализировать массив данных. В последующем предстоит работа по разработке модуля, автоматизирующего собранные данные, как отдельные значения, так и всю последовательность. Будет добавлена визуализация собранных данных для ручного анализа. Для удобства работы с данным расширением планируется интеграция со средой разработки, в частном случае с VS Code, поддерживающей расширения и предоставляющей для этого множество инструментов.

Стоит отметить, что данная система сможет упростить отладку не только программ цифровой обработки сигналов, но и других приложений, в которых требуется работать с большим объемом потоковых данных.

Список литературы

1. Smith, S. W. The Scientist and Engineer's Guide to Digital Signal Processing / S. W. Smith – 2nd ed. – San Diego : California Technical Publishing, 1999. – 663 p.

2. Basic debugging procedure [Электронный ресурс]. – Режим доступа : https://querix.com/go/beginner/Content/05_workbench/01_ls/04_how_to/05_debug/01_dbg_alg.htm

3. Extending GDB [Электронный ресурс]. – Режим доступа : <https://sourceware.org/gdb/onlinedocs/gdb/Extending-GDB.html#Extending-GDB>

Решение задачи описания изображений с помощью нейронных сетей

П. С. Бурдюг

Студент магистратуры

А. В. Акимов

Старший преподаватель

Введение

Машинное описание изображения – это довольно новая задача в области машинного обучения. 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.

Описание изображения довольно простая задача для человека, но очень нетривиальная для машинных алгоритмов. Человеку достаточно беглого взгляда на изображение, чтобы охарактеризовать множество деталей и событий, но для моделей машинного обучения эта способность – невероятно сложная задача [1].

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 [2].

Генерация подписей к изображению возможна в трёх различных вариантах:

- Классификация изображения – каждой картинке присваиваются известные метки классов.
- Описания изображения – генерация осмысленного текста, описывающего всё изображение.
- Аннотация изображения – выделение и генерация описания областей изображения.

Будем рассматривать только второй вариант задачи.

Конечной целью работы является анализ, реализация и сравнение подходов и архитектур для решения задачи описания изображения, а

также выявление закономерностей эволюции нейросетевых моделей на примере данной задачи. В этой статье рассмотрим часть таких подходов.

1. Архитектура энкодер-декодер

Простейшей архитектурой для решения задачи описания изображения является энкодер-декодер (рис. 1).

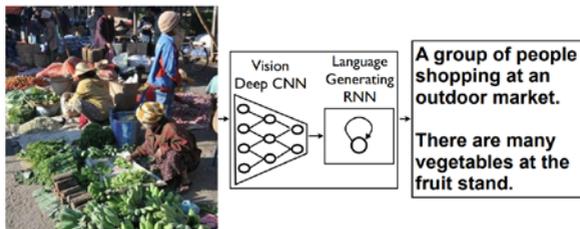


Рис. 1. Иллюстрация архитектуры энкодер-декодер на примере задачи описания изображения [2]

В качестве энкодера используется любая свёрточная нейронная сеть без последнего слоя для классификации, на выходе она представляет изображение в виде вектора фиксированной длины.

Декодер представляет собой языковую модель, состоящую из набора последовательных блоков рекуррентных сетей (рис 2) (в данном примере используются блоки LSTM). Каждый из блоков генерирует слово из результирующего описания изображения (выдает распределение вероятности p на все слова из словаря, на основе которого происходит выбор слова).

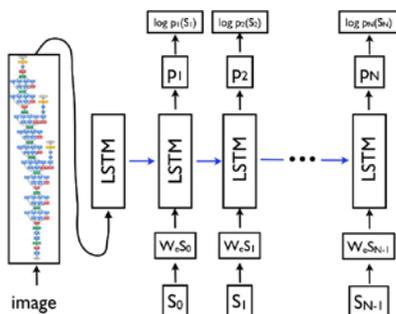


Рис. 2. Устройство декодера и энкодера на примере GoogleNet и LSTM [2]

Слова из словаря представляются в виде векторов унитарного кода (one-hot) S (рис 2). На вход языковой модели при обучении подаются эмбединги слов $W_e S$ – векторные представления слов с учётом их совместной встречаемости в контексте описаний изображений. Для построения эмбедингов используется специальная модель W_e , которая реализует алгоритм кодирования слов.

Перед началом обучения модели по описаниям изображений из датасета строится словарь. Для этого текст описаний токенизируется (разбивается на отдельные слова), затем каждое слово нормализуется (приводится к единой начальной форме), часть служебных мало важных слов удаляется из словаря.

Блоки RNN используют информацию, полученную из изображения с помощью энкодера (вектор представления) и слово, сгенерированное предыдущим блоком (специальное слово <start>, если генерируемое слово первое в тексте).

В итоге обе модели: энкодер и декодер обучаются совместно.

2. Проблемы модели энкодер-декодер

Описанная выше модель имеет ряд недостатков, чтобы выявить их, для начала рассмотрим устройство рекуррентной сети.

В рекуррентной нейронной сети на каждом шаге t вычисляется скрытое состояние h на основе нелинейной функции $f()$ от линейного преобразования входного значения x и скрытого состояния на предыдущем шаге h_{t-1} [4]:

$$h_t = f(Vx_t + Wh_{t-1} + b), \quad (1)$$

где V , W и b – коэффициенты линейного преобразования.

Выходное значение \hat{y}_t вычисляется как нелинейная функция $g()$ от линейного преобразования скрытого состояния h_t с коэффициентами U и \hat{b} :

$$\hat{y}_t = g(Uh_t + \hat{b}). \quad (2)$$

Для обучения сети минимизируются суммарные потери, функция $L_t(y_t, \hat{y}_t)$ показывает, насколько сильно предсказание сети \hat{y}_t отличаются от эталона y_t . Запишем градиент от функции потерь L_t .

Так как градиент от линейного преобразования – это линейное преобразование градиентов, то рассчитаем его только по параметрам W :

$$\frac{dL_t}{dW} = \frac{\partial L_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial h_t} \frac{\partial h_t}{\partial W}; \quad \frac{dh_t}{dW} = \sum_{k=1}^t \left(\prod_{i=k+1}^t \frac{\partial h_i}{\partial h_{i-1}} \right) \frac{\partial h_k}{\partial W}. \quad (3)$$

Можно заметить, что градиент содержит произведение большого количества элементов (равного количеству ячеек RNN), это может вести к затуханию или взрыву градиента.

Способы решения этой проблемы рассмотрим чуть позже, сейчас к перейдем к другим проблемам модели.

Одной из таких проблем является «жадное» декодирование. Когда декодер при генерации следующего слова выдает распределение вероятности по словам в словаре, модель выбирает слово с наибольшей вероятностью и подаёт его на вход следующему блоку для генерации еще одного слова, но такой метод является не оптимальным. Дело в том, что, выбирая слово с максимальной вероятностью, мы отсекаем все остальные варианты, которые могли оказаться более удачными.

Следующая проблема – «забывчивость» декодера при генерации текста описания. Если текст описания длинный, декодер перестает учитывать сгенерированные ранее в начале предложения слова, это связано с особенностями рекуррентной сети (при подсчете результата используются скрытые состояния лишь небольшого количества предыдущих ячеек RNN, значения, которые были посчитаны раньше теряются).

Последняя проблема – вектор представления изображения имеет фиксированную длину и декодер не может описать картинку с большим количеством контента. Если на картинке присутствует много объектов или происходит много действий, модель не может описать их все, так как каждый раз при генерации слова рассматривает один и тот же вектор представления изображения.

В рамках работы была реализована и обучена простая модель энкодер-декодер для проведения экспериментов и выявления проблем. В качестве энкодера была выбрана сеть ResNet-101, а декодера модель RNN. Для обучения использовался датасет MS COCO 2014 и фреймворк PyTorch. В ходе экспериментов на тестовой выборке было обнаружено, что сеть верно описывает простые изображения с краткими описаниями, но имеет трудности с описанием более сложных изображений, содержащих несколько сцен или объектов, для описания которых используется больше текста. Этот результат подтверждает перечисленные выше недостатки.

Рассмотрим возможные улучшения архитектуры для решения этих проблем.

3. Улучшение архитектуры энкодер-декодер

Для решения проблемы взрыва градиента используется метод ограничения градиента (gradient clipping) [5]. Он заключается в искусственном ограничении нормы градиента. Если норма градиента

становится больше заданного значения C , мы присваиваем ему это значение (рис. 3).

$$\text{Если } \|\nabla L\|_{clipped} \geq C, \text{ то } \nabla L \leftarrow \frac{C}{\|\nabla L\|} \cdot \nabla L. \quad (4)$$

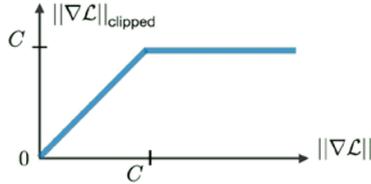


Рис. 3. Метод ограничения градиента [5]

Для борьбы с затуханием градиента используется сети долгой краткосрочной памяти (Long Short Term Memory или LSTM).

Основная идея – добавление в RNN ещё одного скрытого состояния C , предназначенного для дополнительной передачи информации через сеть, что добавляет дополнительный путь «течения» градиента.

В качестве входа на каждом шаге t используется конкатенация скрытого состояния h и входного значения x :

$$z_t = [h_{t-1}, x_t]. \quad (5)$$

Для подсчета выходного значения используется вентиль забывания f – значение, которое характеризует, насколько мы должны учитывать предыдущее состояние C и входной вентиль i – насколько учитывается текущее сгенерированное значение. Оба значения представляют собой сигмоиду от линейного преобразования входного значения [4]:

$$f_t = \sigma(W_f z_t + b_f); \quad i_t = \sigma(W_i z_t + b_i), \quad (6)$$

где W_f, b_f, W_i, b_i – соответствующие коэффициенты линейного преобразования. Текущее значение \hat{C} генерируется аналогично скрытому состоянию в обычной RNN, как гиперболический тангенс от линейного преобразования входного значения z_t с коэффициентами W_c, b_c :

$$\hat{C}_t = th(W_c z_t + b_c). \quad (7)$$

Дополнительное скрытое состояние C формируется, как сумма предыдущего и текущего состояния, взвешенная с использованием коэффициентов выше:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \hat{C}_t. \quad (8)$$

Для получения скрытого состояния h формируется коэффициент на выходе o – выходной вентиль, аналогично другим вентилям это сигмоида от линейного преобразования входного значения с коэффициентами W_o, b_o :

$$o_t = \sigma(W_o z_t + b_o); h_t = o_t \cdot \tanh(C_t). \quad (9)$$

LSTM не только борется с затуханием градиента, но и частично решает проблему «забывчивости» декодера, так как дополнительная передача информации позволяет учитывать более длинный контекст, но проблема всё ещё остаётся.

Перейдем к проблеме «жадного» декодирования. Для её решения используется алгоритм лучевого поиска (beam search) [6]. Его суть заключается в том, чтобы поддерживать в памяти фиксированное количество гипотез k . Запустим генерацию следующего слова для всех вариантов из памяти, а затем рассчитаем условную вероятность для каждого из полученных слов. Она вычисляется как произведение вероятностей каждого из слов в гипотезе:

$$p(w_t) = p(w_{t-1}) \cdot p(w_{t-2}) \cdot \dots \cdot p(w_1). \quad (10)$$

Из всех гипотез мы выбираем k самых вероятных, а остальные отбрасываем. После генерации последнего слова (специального слова $\langle \text{end} \rangle$) выбираем самую вероятную гипотезу.

В результате, удастся не потерять удачные гипотезы из-за «жадного» декодирования. Оптимальным считается число k в пределах от 4 до 8, при больших k качество декодера падает [7].

4. Модули внимания

Наконец перейдем к двум последним проблемам: «забывчивости» декодера и невозможности описать большое число объектов в разных частях изображения.

Для их решения используется механизм внимания. Он заключается в добавлении функции (обычно в виде одного или нескольких полно-связных линейных слоев) в модель, на выходе которой модуль должен отвечать на вопрос – насколько важно «обратить внимание» на ту или иную часть данных.

Рассмотрим механизм самовнимания для декодера. В данном случае модуль внимания должен возвращать информацию о том, насколько важно обращать внимание на каждое из предыдущих слов в предложении при генерации следующего (рис 4).

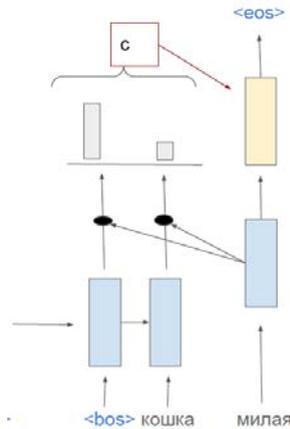


Рис. 4. Механизм самовнимания

Этот механизм реализуется в виде специального коэффициента, показывающего для каждого из предыдущих слов, насколько важно учесть скрытое состояние ячейки, которая его сгенерировала. На каждом шаге t скрытые состояния h с соответствующими коэффициентами вероятностями w суммируются в вектор контекста c :

$$c_t = w_0 h_0 + w_1 h_1 + \dots + w_{t-1} h_{t-1}. \quad (11)$$

Затем этот вектор подаётся на вход текущей ячейке вместе с входным значением. Это решает проблему «забывчивости» декодера, теперь модель может учитывать контекст всего предложения, вне зависимости от того, как далеко стоят слова.

Аналогичным образом модуль внимания может быть применён при обработке изображения. Главная его задача в этом случае – выделить область изображения, на которую важно обратить внимание при генерации следующего слова в описании, тем самым связывая слова с конкретными частями картинки [3]. На вход модуль внимания принимает карту признаков изображения, полученную из энкодера, и сгенерированное на предыдущем шаге слово. Результатом работы модуля внимания является взвешенная карта изображения, где каждое значение – коэффициент важности соответствующего пикселя на изображении для генерации следующего слова (рис 5) [8]. В сумме все коэффициенты составляют единицу.

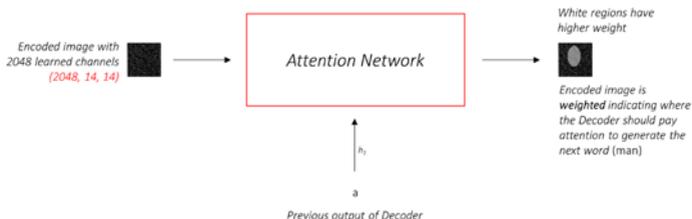


Рис. 5. Механизм attention для изображения [8]

Таким образом, модель учится описывать изображение по частям также, как это делал бы человек, что позволяет генерировать описания для картинок с большим числом объектов и действий.

Заключение

В рамках данной работы были исследованы методы решения задачи описания изображения, а именно архитектура энкодер-декодер, выявлены недостатки данной архитектуры, а также рассмотрены алгоритмы её улучшения, главным из которых является механизм внимания. Был реализован простейший вариант модели.

В качестве дальнейшего направления работы планируется сравнить эффективность обычной модели энкодер-декодер, с моделью с механизмом внимания, используя метрики BLEU, METEOR, CIDEr, а также рассмотреть архитектуру трансформер для решения данной задачи. Дело в том, что модель энкодер-декодер имеет еще один изъян: это рекуррентная сеть в декодере, которая работает последовательно, архитектура трансформер лишена этого недостатка [9].

Список использованной литературы

1. Karpathy, A. Deep Visual-Semantic Alignments for Generating Image Descriptions / A. Karpathy, Li Fei-Fei // CVPR. – 2015. – DOI: 10.48550/arXiv.1412.2306.
2. Show and Tell: A Neural Image Caption Generator / Oriol Vinyals, [et. al.] // CVPR. – 2015. – DOI: 10.48550/arXiv.1411.4555.
3. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention / Kelvin Xu [et.al.]// PMLR. – 2015. – DOI: 10.48550/arXiv.1502.03044.
4. Математические методы анализа текстов (курс лекций, К.В.Воронцов, А.А.Потапенко) [Электронный ресурс]. – Режим доступа: <http://bit.ly/2LFhfCA>

5. Recurrent Neural Networks cheatsheet [Электронный ресурс]. – Режим доступа: <https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks#architecture>
6. Graves, A. Sequence Transduction with Recurrent Neural Networks / A. Graves // ICML. – 2012. – DOI: 10.48550/arXiv.1211.3711.
7. Freitag, M. Beam Search Strategies for Neural Machine Translation / M. Freitag, Y. Al-Onaizan // First Workshop on Neural Machine Translation. – 2017. – DOI: 10.48550/arXiv.1702.01806.
8. Show, Attend, and Tell | a PyTorch Tutorial to Image Captioning [Электронный ресурс]. – Режим доступа: <https://github.com/sgrvinod/a-PyTorch-Tutorial-to-Image-Captioning>
9. Attention Is All You Need / Ashish Vaswani [et.al.]// NIPS. – 2017. – DOI: 10.48550/arXiv.1706.03762.

Нейросетевой и МНК алгоритмы совместного определения ориентации и оптических параметров плоской диффузной поверхности по данным измерений характеристик светорассеяния

В. В. Василега

Студент магистратуры

В. Д. Попело

Профессор

Введение

К числу нормируемых свойств объектов техносферы относят их заметность в оптическом диапазоне длин волн электромагнитного излучения. Это свойство отражает возможность человека или технических средств наблюдения извлекать информацию о наличии, местоположении (траектории движения), типе и режиме функционирования объекта, содержащуюся в структуре оптических полей (сигналов) рассеянного, собственного и функционального излучения. Возможны различные стратегии управления заметностью предусматривающие как максимизацию (например, для объектов повышенной опасности), так и минимизацию (например, для образцов вооружения и военной техники) количественных показателей этого свойства за счет совершенствования формы объекта, применения специальных материалов и покрытий, регулирования уровня функциональных излучений.

Образы объектов при их наблюдении в оптическом диапазоне длин волн, представляющие пространственные распределения яркости, обладают большой изменчивостью. Поэтому для нормирования оптических свойств объектов часто удобно использовать более стабильные параметры, например, значения коэффициента отражения и углы наклона отдельных элементов поверхности. При этом дистанционное определение таких параметров в настоящее время встречает значительные затруднения.

Целью работы является сравнение методов совместной оценки оптических и пространственных параметров поверхности по данным

наблюдений под различными углами, включающих метод наименьших квадратов и использование нейронной сети с прямой связью.

1. Анализ предметной области

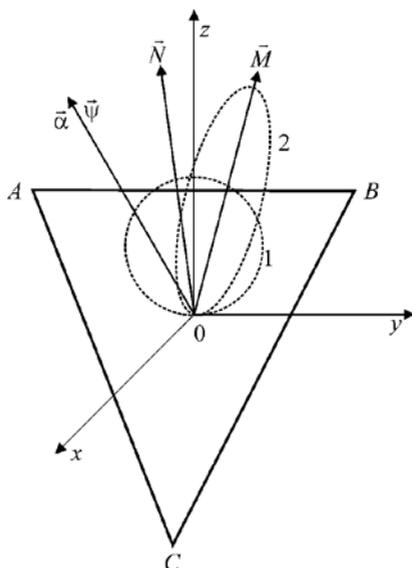


Рис. 1. Геометрическая схема задачи

Пусть участок плоской диффузной поверхности ABC облучается направленным оптическим излучением, а рассеянное регистрируется приёмником. Рассмотрим наиболее простую ситуацию совмещенного приема, когда облучение объекта и его наблюдение осуществляют с близких направлений.

Геометрию задачи определяют четыре вектора, заданные в трехмерной декартовой системе координат $XYZO$ такой, что оси OZ и Ox лежат в горизонтальной плоскости, а ось OZ совпадает с нормалью к этой плоскости. Вектор \vec{N} перпендикулярный к поверхности ABC , характеризует ее наклон. Вектор $\vec{\psi}$ характеризует направление на источник излучения. Вектор $\vec{\alpha}$ характеризует направление в точку наблюдения. Для частного случая, когда облучение объекта и его наблюдение осуществляют с близких направлений, этот вектор совпадает с вектором $\vec{\psi}$. Геометрическая схема приведена на рис. 1.

Пусть имеется возможность изменять положение источника и приемника относительно поверхности ABC (или поворачивать эту поверхность). При этом будет изменяться положение векторов $\vec{\psi} = \vec{\alpha}$ относительно нормали \vec{N} , но сохраняется значение расстояния между приемником и поверхностью. В каждом положении приемник регистрирует отраженный сигнал U_i .

2. Алгоритм оценки параметров с помощью МНК

Для каждого измерения известно направление $\vec{\psi} = \vec{\alpha}$ облучения и наблюдения поверхности ABC и значение сигнала U_i . Неизвестными являются оптические параметры и ориентация поверхности, характеризуемая положением вектора \vec{N} .

Индикатриса рассеяния (силы $I(\theta)$ излучения) для диффузной поверхности подчиняется закону Ламберта

$$I(\theta) = I_0 \cos \theta, \quad (1)$$

где I_0 – максимальная сила отраженного излучения в направлении вектора \vec{N} нормали к поверхности; $\theta = (\vec{N}, \vec{\psi})$ – угол между нормалью к поверхности направлением на приемник рассеянного излучения.

Поэтому в случае диффузного рассеяния единственным оптическим параметром поверхности будет коэффициент отражения ρ .

Зависимость между реальным значением сигнала и теоретическим определена как

$$u_i - \rho(l_{\psi i} l_N + m_{\psi i} m_N + n_{\psi i} n_N) = v_i \quad (2)$$

где u_i – относительное (безразмерное) значение отраженного сигнала в момент времени t_i ; $\cos \psi_{xi} = l_{\psi i}$, $\cos \psi_{yi} = m_{\psi i}$, $\cos \psi_{zi} = n_{\psi i}$; v_i – невязка i -ого условного уравнения, обусловленная тем, что значения u_i , $l_{\psi i}$, $m_{\psi i}$, $n_{\psi i}$ измерены с погрешностями; $i = \overline{1, W}$, W – количество проведенных измерений; $\cos \eta_x = l_N$, $\cos \eta_y = m_N$, $\cos \eta_z = n_N$ – искомые значения направляющих косинусов нормали к плоской поверхности объекта; ρ – искомое значение коэффициента отражения поверхности.

Метод наименьших квадратов предусматривает поиск решения, удовлетворяющего минимуму суммы квадратов невязок условных уравнений, то есть когда выполняется условие

$$S = \sum_{i=1}^W (v_i - l_{\psi i} p - m_{\psi i} q - n_{\psi i} h)^2 \rightarrow \min, \quad (3)$$

где $\rho l_N = p$; $\rho m_N = q$; $\rho n_N = h$ – новые переменные.

Необходимые условия достижения минимума функционала S имеют вид

$$\frac{\partial S}{\partial p} = 0, \frac{\partial S}{\partial q} = 0, \frac{\partial S}{\partial h} = 0. \quad (4)$$

С использованием этих выражений формируем систему нормальных уравнений, включающую три линейных уравнения с тремя неизвестными

$$\begin{aligned} p \sum_{i=1}^n l_{\psi i}^2 + q \sum_{i=1}^n l_{\psi i} m_{\psi i} + h \sum_{i=1}^n l_{\psi i} n_{\psi i} &= \sum_{i=1}^n v_i l_{\psi i}; \\ p \sum_{i=1}^n l_{\psi i} m_{\psi i} + q \sum_{i=1}^n m_{\psi i}^2 + h \sum_{i=1}^n m_{\psi i} n_{\psi i} &= \sum_{i=1}^n v_i m_{\psi i}; \\ p \sum_{i=1}^n l_{\psi i} n_{\psi i} + q \sum_{i=1}^n m_{\psi i} n_{\psi i} + h \sum_{i=1}^n n_{\psi i}^2 &= \sum_{i=1}^n v_i n_{\psi i}. \end{aligned} \quad (5)$$

Найти решения этой системы уравнений можно, например, методом Крамера, после чего они подставляются в следующее выражение для определения коэффициента отражения

$$\rho = \sqrt{p^2 + q^2 + h^2} \quad (6)$$

Направляющие косинусы вектора \vec{N} вычисляются как

$$l_N = \frac{p}{\sqrt{p^2 + q^2 + h^2}}, m_N = \frac{q}{\sqrt{p^2 + q^2 + h^2}}, n_N = \frac{h}{\sqrt{p^2 + q^2 + h^2}} \quad (7)$$

3. Оценка параметров поверхности с помощью нейронной сети прямой связи

Для использования нейронной сети необходимо провести ее обучение.

Для этого использовался набор данных из 400 тренировочных групп измерений. В каждой группе каждые два последовательных измерения были сделаны из точек, отстоящих друг от друга на фиксированное угловое расстояние, единое для всего набора данных.

Для каждой группы измерений указано значение сигнала, отраженного поверхностью, с пятнадцати направлений, и значение коэффициента отражения поверхности.

Сама нейронная сеть представлена следующей моделью.

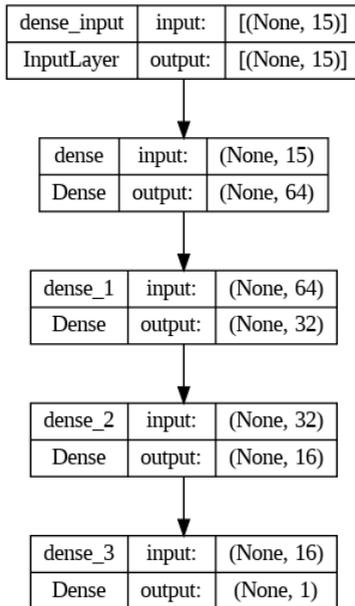


Рис. 2. Модель нейронной сети

Модель имеет входной слой из 15 узлов, 3 скрытых слоя и один выходной слой. Скрытые слои имеют функции активации ReLU, функцией активации выходного слоя является сигмоид. Выходной величиной является оценка коэффициента отражения поверхности.

4. Результаты использования алгоритмов

Для оценки эффективности реализованных методов был собран набор групп измерений отраженного света. Этот датасет содержит 20 групп измерений.

Для оценки эффективности работы алгоритмов используется метрика средней абсолютной процентной ошибки, которая определяется следующим образом:

$$M = \frac{1}{n} \sum_{i=1}^n \left| \frac{A_i - F_i}{A_i} \right| \quad (8)$$

где A_t - реальное значение предсказываемого параметра, F_t - предсказание параметра, n – число предсказаний.

Процесс обучения нейронной сети изображен на графиках. график “loss” соответствует функции потери, на основе которой калибровались параметры нейронной сети.

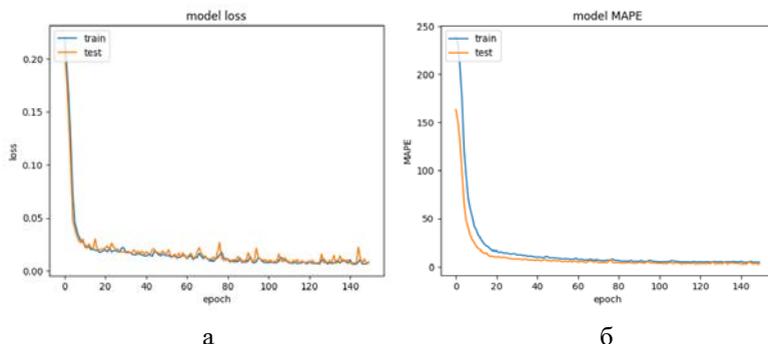


Рис. 3. Графики зависимостей: а – средней абсолютной процентной ошибки; б – функции потерь от эпохи обучения, синей линией отмечены результаты обучающей выборки, оранжевой – валидационной

Таблица 1

Результаты регрессии

Метод регрессии	Ошибка регрессии(MAPE)
Метод наименьших квадратов(МНК)	6.5795
Нейронная сеть прямой связи	3.7862

Наилучшие результаты классификации методом случайного леса были достигнуты при следующих значениях гиперпараметров: 150 эпох, 15 проходов сети за эпоху, функция потерь – средняя абсолютная ошибка. Для сравнения, при использовании данных измерений, не имеющих помех, МНК показал результаты, идентичные реальным значениям коэффициента отражения, в то время как нейронная сеть показала значение MAPE равное ~1.5%. время работы методов при вычислении одной группы измерений представлены в табл. 2.

Время вычисления

Метод регрессии	Время работы (в мс)
Метод наименьших квадратов(МНК)	0.0039
Нейронная сеть прямой связи	6.1281

5. Обсуждение результатов

Как видно из результатов, рассмотренные методы позволяют достичь достаточно высокой точности оценки коэффициента отражения поверхности. Но у применения этих двух методов есть свои особенности.

Как видно из результатов, нейронная сеть показала большую устойчивость к помехам, и, как следствие, большую точность. В то же время, выполнения вычислений при использовании МНК значительно меньше, чем у нейронной сети. Это вызвано сравнительно меньшим числом необходимых вычислений по сравнению с использованной моделью нейронной сети.

Ещё одной особенностью является природа модели нейронной сети. Это проявляется в двух аспектах: ограничение на входные данные, и процесс обучения сети. В текущей реализации, модель сети предполагает на вход фиксированного числа измерений, последовательной отстоящих друг от друга на фиксированное угловое расстояние, потому что таким ограничениям отвечал обучающая выборка. Оценка на основе каких-либо других данных может привести к непредсказуемым результатам. Это приводит к проблеме формирования обучающей выборки, отвечающей этим ограничениям. Возможным вариантом частичного преодоления этой проблемы является изменение модели сети для включения направлений измерений в процесс регрессии, аналогично с методом наименьших квадратов.

Заключение

В ходе работы реализовано математическое и программное обеспечение для оценки оптических и пространственных параметров поверхности. Регрессия методом наименьших квадратов достигнута средняя абсолютная процентная ошибка в 6.5%. Ошибка регрессии с помощью нейронной сети прямой связи – 3.7%.

Список литературы

1. Обнаружение и координатометрия оптико-электронных средств, оценка параметров их сигналов / под ред. Ю.Л. Козирацкого. – М.: Радиотехника, 2015. – 456 с.
2. Попело, В.Д. Оптико-локационные характеристики объектов различной размерности / В. Д. Попело, И. Р. Фахуртдинов // Метрология. – 2012. – № 7. – С. 9-18.
3. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems / Aurélien Géron. – 2nd Edition. – O'Reilly Media, 2019. – 856 p.

Поиск закономерностей в поведении клиентов телекоммуникационной компании

М. С. Верещагина

Студент магистратуры

В. В. Гаршина

Доцент

Введение

Большинство крупных корпоративных компаний, вынужденных в ходе своей работы обрабатывать большие объемы клиентских данных, обращаются к технологиям машинного обучения и анализу больших данных. При таких объемах разнородной (и далеко не всегда «чистой») информации компании не всегда имеют возможность следить за ситуацией на рынке и вовремя реагировать на негативные события без использования дополнительных средств анализа. Так, один из самых популярных примеров использования технологий Data Mining в реальной жизни – это прогнозирование оттока клиентов.

Задача прогнозирования оттока клиентов (churn prediction) заключается в том, чтобы заранее найти и выявить категорию пользователей, склонных через определенный промежуток времени отказаться от использования услуг и продуктов компании. Точное и заблаговременное обнаружение таких клиентов позволяет более эффективно бороться с их оттоком – например, выявлять причины их ухода и принимать соответствующие меры по их удержанию. Эта задача особенно актуальна для организаций, работающих в сегменте B2C («business to consumer» или «бизнес — потребителю»), а в особенности для компаний, работающих в областях, где распространение предлагаемых компанией услуг близко к 100%. [1] В такой ситуации задача удержания клиентов в компании и налаживания с ними взаимоотношений выходит на первый план, поэтому в данной работе будет рассмотрена методика прогнозирования оттока клиентов в B2C сегменте.

1. План исследования

В результате вышеописанного анализа предметной области и подходов к анализу оттока клиентов был составлен план исследования

для разработки сценария проведения бизнес-анализа модели поведения расторгнувших контракты для задачи управления отношениями с клиентами телекоммуникационной компании.

Для достижения поставленной цели предполагается решение следующих задач:

1. Определить признаки классификации поведения абонентов.

2. Сформировать дата сеты из истории взаимодействия компании с клиентами. Для оценки вероятности отключения того или иного абонента компании используют большое количество метрик: количество совершенных звонков в месяц, пользовательские подписки, давность последней покупки и многие другие. Поэтому мы хотим проклассифицировать отключившихся за определенный период абонентов по ряду признаков - это будет происходить в рамках итерационного процесса с корректировкой классифицирующих признаков в зависимости от полученных результатов на последующих шагах.

3. Выбрать набор алгоритмов Data Mining для решения поставленных задач.

4. Разработать сценарии для Системы Анализа Данных для выявления клиентов “в группе риска” (т.е. тех клиентов, которые в ближайшем будущем способны покинуть компанию) и наиболее влиятельных классифицирующих признаков и их значений, наибольшим образом способствующих оттоку абонентов разных групп.

На основе полученных результатов сформировать предложения для удержания клиентов (в первую очередь наиболее значимых для компании групп).

2. Исходные данные для анализа

Данные анализируемой компании интернет-провайдера содержат 3 следующие связанные между собой таблицы Excel с информационной выборкой о 24 617 клиентах этой компании:

1. Building – таблица с характеристиками места подключения клиентов;
2. Client – список клиентов;
3. Period – история расчетных периодов клиентов с 1 января 2008 года по 31 декабря 2022 года.

Так как в задаче речь идет о клиентских данных - при подготовке датасета учитывалось, что данные должны быть предварительно анонимизированы и обезличены, т.е. не должны содержать любую персональную информацию, позволяющую идентифицировать пользователей – и этот датасет удовлетворяет этому условию. Этот

набор данных отслеживает отток клиентов телекоммуникационной компании на основе множества возможных факторов.

Далее мы сфокусировали свое внимание на клиентах, обладающих адресом места подключения. Подходящих записей в таблице Client нашлось в количестве 3 476.

3. Разведывательный анализ

Существуют разные подходы к анализу оттока клиентов. Большинство из них основано на интеллектуальном анализе данных, показывающих в современном мире высокую эффективность и производительность. [2]

В качестве основного инструмента для проведения анализа после обзора рынка инструментов было решено выбрать аналитическую платформу Logiном. Данное решение отчасти обусловлено доступностью бесплатной полнофункциональной версии программы.

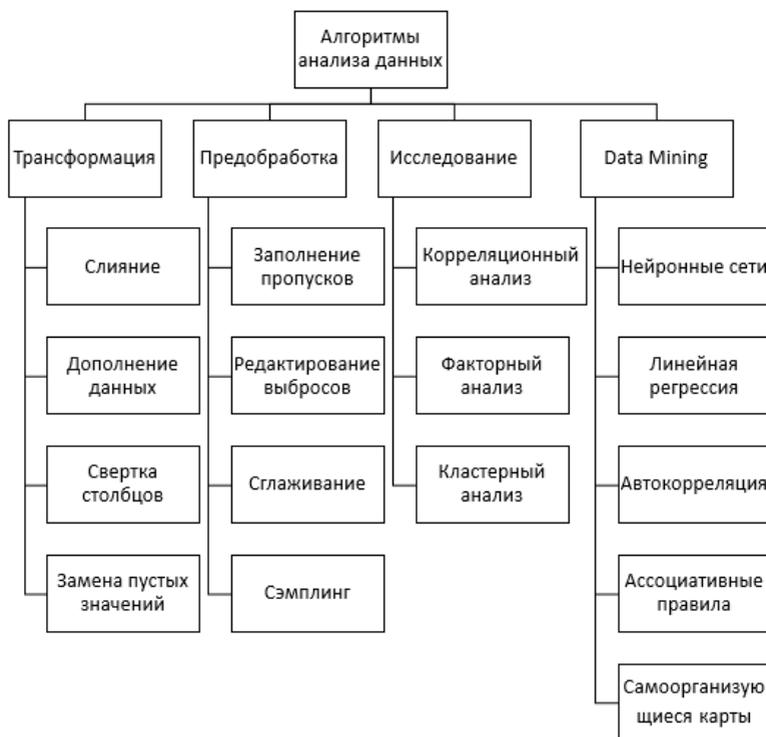


Рис. 1. Классификация алгоритмов анализа данных

На рис. 1 показаны различные алгоритмы обработки данных, которые доступны в программе Loginom для нахождения и извлечения скрытых закономерностей из информации. [3]

В ходе данного исследования применялись такие методы, как слияние нескольких таблиц в одну, заполнение пропусков, сэмплинг, внутреннее обогащение данных и другие. К примеру, *sdate* – начало расчетного периода клиента является непрерывным атрибутом, который надо дискретизировать, чтобы получить категориальный атрибут для дальнейшей классификации. Для каждого из клиентов по истории расчетных периодов были выявлены дополнительные плоские признаки для анализа данных – например, дата прихода и ухода пользователя из компании. На основе этих данных были построены графики притока/оттока клиентов по годам и планам, представленные на рис. 2-3.

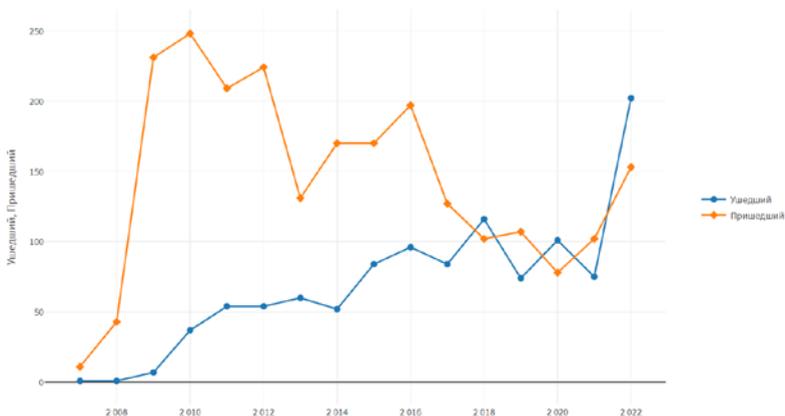


Рис. 2. Общий приток/отток по годам

Можно заметить, что с 2016 года постепенно снижается прирост клиентской базы и наблюдается увеличение оттока клиентов.

При этом анализ планов (рис. 3) показал, что наибольший отток в последние годы наблюдался с 4 архивных планов, которые больше недоступны для подключения. В то же время это одни из самых популярных тарифов Архивный Тариф №2 до сих пор имеет самое большое количество подключений (даже в сравнении с новыми тарифами).

Дальнейшее сравнение «среднего» чека ушедших и пришедших клиентов показало, что средняя цена, которую платили ушедшие клиента за свои планы, гораздо выше средней цены только что

пришедших клиентов, даже несмотря на рост значения накопительной скидки, что и могло стать причиной оттока клиентов.

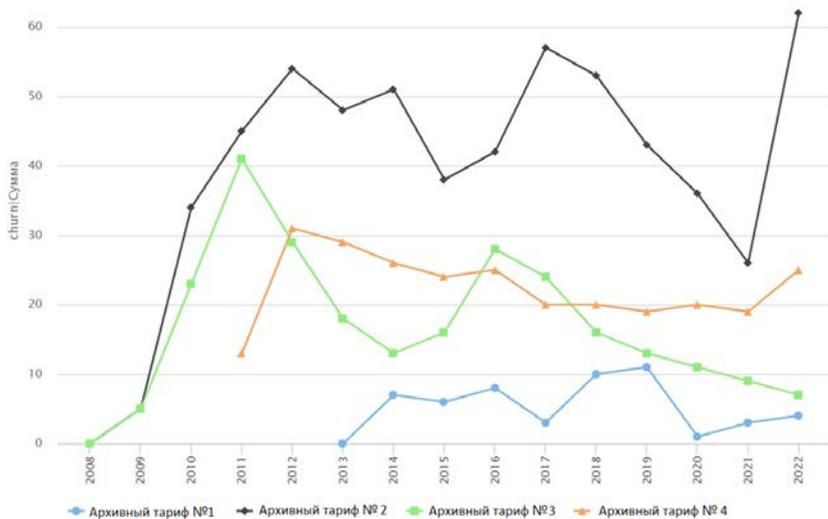


Рис. 3. Отток с архивных тарифов по годам

4. Описание данных для анализа методами Data Mining

Сегментирование клиентов в зависимости от их профилей в БД проводилось с использованием аппарата самоорганизующихся карт Кохонена, основанных на обучении нейронной сети Кохонена (без учителя) для решения задач классификации и кластеризации. Карты Кохонена являются методом проецирования многомерного пространства в пространство с более низкой размерностью (чаще всего, двумерное) и применяется также для решения задач моделирования, прогнозирования и др. В программе Loginom карта Кохонена это еще один настраиваемый компонент, который можно связывать с другими.

Размер карты Кохонена выбран опытным путем и составил 2 на 2 шестиугольников. Каждый из шестиугольников является нейроном сети. При данном варианте сеть состоит из 4 нейронов, обучение проходит без появления «пустых» нейронов, негативно сказывающихся на оценке полученной информации.

В нижеприведенной таблице представлены входные атрибуты, использованные для обучения нейронной сети.

Используемые для анализа атрибуты

Атрибут	Значение
district	Квартал
moving_exists	Был ли переезд
tp_name	Тарифный план
tv_tp_packs	ТВ-пакеты, которые входят в тарифный план
local_bonus	«Хороший» ли РП, т.е. добавляет ли этот РП накопительную скидку клиенту
action_bonus_k	Скидка итоговая, включая акции
discount_sum	Общая сумма списаний с клиента в РП

#	Метка поля	Тип	Итого	Кластер 1	Кластер 2	Кластер 3	Кластер 0
			100%	34%	24%	23%	18%
1	ab district_name	⊗	100,00	99,39	100,00	99,99	100,00
2	12 moving_exists	⊗	99,47	88	80	84	80
3	ab tp_name	⊗	100,00	99,94	99,95	99,99	98,9
4	ab tv_tp_packs	⊗	100,00	99,94	97,6	97,6	91,6
5	12 local_bonus	⊗	95,8	75	70	75	72
6	12 action_bonus_k	⊙	56	49	61	54	75
7	12 discount_sum	⊗	100,00	99,06	100,00	100,00	0,09

Рис. 4. Профили кластеров ушедших клиентов

Результатом обучения нейронной сети является построение кластеров и карты сегментирования клиентов по их профилям, представленные на рисунке 4. По каждому из полученных кластеров приведена информация о его мощности и уровнях значимости для каждого из входных полей, а также и оценка общей мощности того или иного кластера. По построенным картам можно оценить лояльность клиентов к той или иной услуге, увидеть связи ранее не заметные во влиянии услуг друг на друга.

В результате обучения нейронная сеть обнаружила 4 независимых кластера с типичными поведением клиентов в использовании услуг:

- 11 кластер (34%) – клиенты с самым высоким «средним» чеком;
- 22 кластер (24%) – ушли с архивного тарифа со скидкой;
- 33 кластер (23%) – ушли с архивного тарифа без скидки;
- 40 кластер (18%) – клиенты с самой большой скидкой.

Заключение

Loginom открывает широкие возможности для оперативного анализа больших ненормализованных наборов данных. Так в ходе исследования был нами выявлен ряд закономерностей, которые

предстоит проанализировать и глубже изучить, продолжить работу над профилированием клиентов с помощью карты Кохонена с попыткой более тщательного подбора входных полей для анализа. Этот подход представляется перспективным с точки зрения анализа поведения существующей клиентской базы на предмет предсказания потенциального оттока. Разработанный сценарий после тестирования затем можно будет многократно применять к однотипным данным и использовать для анализа поведения и прогнозирования оттока клиентов в B2C сфере.

Список литературы

1. Кудинов, А. Т. CRM: российская практика эффективного бизнеса / А. Т. Кудинов, Е. Гольшева, М. Сорокин. – М. : Издательство ИС-Публишинг, 2009. – 374 с.
2. Методы и модели анализа данных: OLAP и Data Mining / А. А. Барсемян [и др.]. – СанктПетербург : БХВ Петербург, 2012. – 335 с.
3. Loginom – Руководство пользователя [Электронный ресурс]. – Режим доступа: <https://help.loginom.ru/userguide/>

Обзор технологий голосового ввода для анализа сообщений

В. С. Господарикова

Студент магистратуры

В. В. Гаршина

Доцент

Введение

Сфера компьютерных технологий с каждым годом становится всё более обширной и разнообразной. Изобретаются новые способы обработки, хранения и создания информации. И в то же время компьютерные технологии стремятся к расширению своей аудитории пользователей. Этому способствует развитие дизайна интерфейсов, новые способы анализа опыта взаимодействия с конечной программой различных людей, заинтересованных в ней.

Всё это приводит к открытию новых технологий для людей без специального компьютерного образования. Безусловно, это очень важно, ведь эксперты в различных не технологических областях обладают огромным количеством знаний и умений, которые также должны быть накоплены и использованы при решении сложных, комплексных задач.

Однако если мы посмотрим на такую область, как онтологии – формализованные базы знаний, то сможем заметить, что процесс их создания очень труден сейчас для человека без специального образования. А ведь, с одной стороны, именно такой формат наиболее понятен и доступен для анализа и выводов в конкретной ситуации. И в то же время создание онтологии не специалистом в предметной области, к которой будет относиться конкретная база знаний, невозможно.

До текущего момента наиболее часто для создания подобных графов использовалась совместная работа компьютерного специалиста и эксперта, располагающего теоритическими знаниями. Однако, это часто приводит к непониманию между людьми, ошибками в данных базы и прочим оплошностям.

Но со временем люди, так или иначе, задумываются о том, что этот процесс необходимо облегчить и сделать доступным не техническим специалистам. И в этот момент становится вопрос о том, какой формат внесения данных наиболее удобен для любого человека. И тут вспоминается то, что каждый из нас ежедневно разговаривает на родном

для него языке. Голосовой ввод был придуман достаточно давно и с тех пор рекомендует себя наилучшим образом. Поэтому для создаваемого мной редактора онтологий было принято условие, что команды пользователя распознаются из его речи.

Но дальше передо мной встал другой трудноразрешимый вопрос: какое средство распознавания сообщений подходит лучше всего. Далее мы проведём анализ необходимых для нашего редактора онтологий параметров голосового ввода. После чего рассмотрим и сравним подходящие средства распознавания и выберем наилучшее из них для прикладной задачи.

1. Требования к средству распознавания речи

Чтобы обозначить необходимые требования к средству распознавания речи, необходимо сначала задать функции, которые оно должно выполнять. В нашей программе это распознавание речи. Также нужно перечислить требования к обрабатываемым входным и выходным данным голосового интерфейса:

- На вход должна быть возможность подавать команды на русском языке;
- На выходе должен быть текст на русском языке;
- Полученный текст желательно должен быть заранее разделён знаками препинания;
- Конечные команды должны передавать смысловую составляющую максимально точно.

Таким образом, мы можем сказать, что средство распознавания речи должно работать с русским языком, иметь свой словарь и, желательно, работать с пунктуацией.

Также рассматривая голосовой интерфейс с точки зрения удобства пользователя нужно упомянуть следующее. Поскольку, команды пользователя могут быть длиной в одно или несколько предложений, то вся программа будет дожидаться полной версии сказанного перед началом работы с онтологией. Это вызвано тем, что если разделить команду на смысловые единицы раньше окончания предложения, то можно потерять часть названия узла или текста атрибута. И это накладывает свои требования на голосовой интерфейс: его скорость распознавания должна быть как можно ближе к скорости речи или меньше. Такое требование отразится в параметре SF(Speed factor), для которого Трасп – это время распознавания сигнала, Т – длительность сигнала:

$$SF = T_{расп} / T \quad (1)$$

Тогда данный параметр в наших требованиях должен быть меньше 1. Тогда распознавание речи займёт времени не больше, чем у пользователя её произнесение. При этом стоит отметить, что для нас обязательно иметь этот показатель минимальным из всех, потому что это не ускорит работу всего приложения из-за ожидания окончания команды.

Следующим требованием стоит упомянуть открытый доступ к рассматриваемой технологии. Это упростит её подключение и дальнейшее использование. Также нужно упомянуть, что эксперты предметной области нередко пользуются компьютером без доступа в интернет для сохранения интеллектуальной собственности. Поэтому стоит искать средства распознавания, которые могут работать в офлайн режиме.

Ещё один важный пункт подбора – возможность выбора используемой языковой модели. Это также опирается на необходимость работы с различными предметными областями. Не имея возможности обновить модель самостоятельно, нам пришлось бы ждать, пока новые научные термины станут широко использоваться, что неприемлемо.

И последним требованием к голосовому интерфейсу необходимо указать максимальную точность распознавания. Для измерения точности можно использовать такую характеристику, как SER – это метрика, определяющая точность на основе отношения количества неправильно распознанных предложений к общему количеству предложений. Однако, нам необходимо распознавать не тексты, а одно или несколько предложений за раз. Поэтому данная характеристика не так актуальна для нас как следующая.

Вторая распространённая метрика для измерения точности системы распознавания речи называется WER. Это производная от величины, называемой «расстояние Левенштейна».

Расстояние Левенштейна, или редакционное расстояние, — метрика сходства между двумя строковыми последовательностями. Рассчитывается оно, как минимальное количество операций вставки, удаления и замены одного символа для превращения одной строки в другую. Чем больше расстояние, тем сильнее различаются сравниваемые строки. Для одинаковых последовательностей символов расстояние равно нулю. WER (Word error rate) – процент неправильно распознанных слов. Она вычисляется, как сумма всех необходимых операций замены (S), удаления (D), добавления (I) символов в распознанную строку для получения правильного результата, делённая на количество слов (N) в конечной строке:

$$WER = (S + D + I) / N \quad (2)$$

Для нашего голосового интерфейса метрика WER должна быть минимальна для наилучшего распознавания команд.

Итак, подведём итоги данного раздела. Искомое средство распознавания речи должно удовлетворять следующим требованиям:

1. Возможность распознавания фраз на русском языке;
2. Открытый доступ к коду;
3. Офлайн режим работы;
4. Возможность выбора языковой модели;
5. Метрика SF должна быть меньше или равна 1;
6. Метрика WER должна быть минимальна.

С этими данными мы перейдём к следующему разделу, где рассмотрим существующие голосовые интерфейсы.

2. Сравнение различных средств распознавания

В наши дни существует огромное разнообразие коммерческих систем распознавания речи. Свои услуги предлагают Google, Yandex, Amazon и многие другие компании. Однако под наши требования они не подходят. Поэтому мы рассмотрим только следующие три системы, удовлетворяющие нашим потребностям:

1. CMU Sphinx – группа систем распознавания речи разработанных университетом Карнеги Меллон. Она включает в себя серию систем распознавания (Sphinx 2 – 4) и акустическую тренировочную модель (SphinxTrain).

2. Kaldi Vosk – это набор инструментов для распознавания речи, написанный на языке C++, имеющий лицензию Apache v2.0. Одной из целей Kaldi было иметь современный и гибкий код, написанный на C++, который можно было бы легко расширить и изменять.

3. Mozilla DeepSpeech – это система с открытым исходным кодом, для преобразования речи в текст. Для обучения используется модель, обученная методами машинного обучения, на основе глубоких речевых исследований Байдю. DeepSpeech использует проект TensorFlow, чтобы облегчить реализацию.

Все отобранные голосовые интерфейсы уже удовлетворяют первым четырём требованиям из списка, представленного в конце предыдущего раздела. Проведём теперь их сравнение по метрикам WER и SF. Данные метрики были получены распознаванием тестового текста всеми средствами распознавания. Результаты сравнения представлены в табл. 1.

Результаты сравнения метрик

Название	WER, %	SF
CMU Sphinx	21.4	0.5
Kaldi Vosk	6.5	0.6
Mozilla DeepSpeech	29.96	0.9

Для наглядности отобразим полученные результаты на гистограммах (рисунок 1, 2).

WER, %

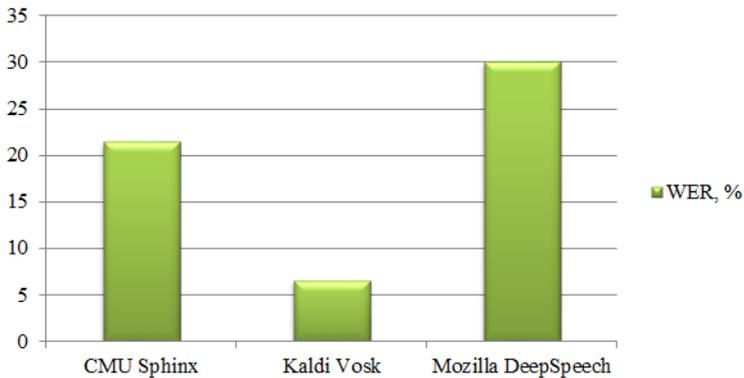


Рис. 1. Сравнение результатов расчёта WER

SF

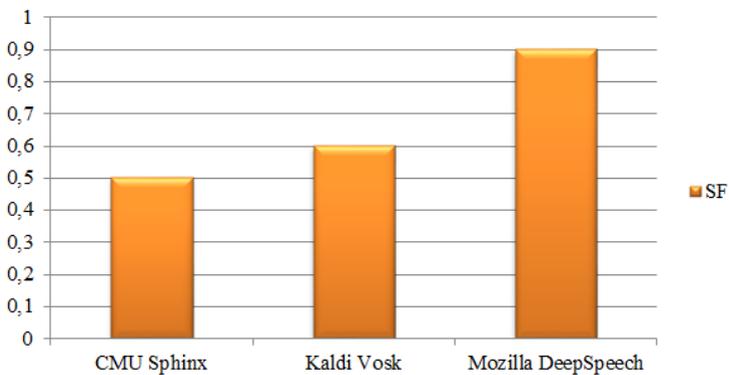


Рис. 2. Сравнение результатов расчёта SF

Из графиков видно, что самый быстрый результат показал СМУ Sphinx. Однако требованию по скорости удовлетворили все три системы.

Также, если рассматривать характеристику точности распознавания, то на первом месте по количеству ошибок оказалась Kaldi Vosk. Это приводит нас к выводу, что самая подходящая система распознавания именно эта.

Заключение

В ходе работы был проведён анализ требований к системе распознавания речи. Также были рассмотрены существующие варианты решения проблемы и выбран наиболее подходящий из всех. Kaldi Vosk удовлетворяет следующим требованиям:

- Возможность распознавания фраз на русском языке;
- Открытый доступ к коду;
- Офлайн режим работы;
- Возможность выбора языковой модели;
- Метрика SF меньше 1 и равна 0.6;
- Метрика WER минимальна для рассматриваемых средств распознавания речи и равна 6.5%.

Это самая подходящая система для решения прикладной задачи распознавания команд.

Список литературы

1. Карпов, А. А. Методология оценивания работы систем автоматического распознавания речи / А. А. Карпов, И. С. Кипяткова // Известия высших учебных заведений. Приборостроение. – 2012. – Т. 55, №. 11. – С. 38-43.

2. Беленко, М. В. СРАВНИТЕЛЬНЫЙ АНАЛИЗ СИСТЕМ РАСПОЗНАВАНИЯ РЕЧИ С ОТКРЫТЫМ КОДОМ / М. В. Беленко, П. В. Балакшин // Международный научно-исследовательский журнал. – 2017. – №. 4(58), Часть 4. – С. 13-18.

3. Жевнеров, К. Оценка качества работы систем Speech to Text / К. Жевнеров // NTA. – 2023 [Электронный ресурс]. – Режим доступа: <https://newtechaudit.ru/ocenka-kachestva-raboty-sistem-speech-to-text/>

4. Хабр [Электронный ресурс] : Расстояние Левенштейна для чайников. – Режим доступа: <https://habr.com/ru/articles/676858/>

Использование нейронных сетей в задаче распознавания сцен

В. О. Григорьев

Студент магистратуры

А. В. Акимов

Старший преподаватель

Введение

Распознавание сцен – одно из важных направлений в классификации изображений, поскольку технология распознавания сцен используется в приложениях взаимодействия человека с компьютером, интеллектуальной робототехники, интеллектуального видеонаблюдения и автономного вождения. Это также считается предварительным условием или предварительными знаниями для других сложных задач компьютерного зрения, таких как поиск изображений по текстовому запросу и обнаружение объектов в контексте сцены. [1]

Основной целью распознавания сцен является присвоение семантических меток данным изображениям, эти семантические метки определяются человеком, включая различные виды природы, сцены в помещении, городскую среду и т.д. Большое количество разнообразных объектов, различная компоновка и семантическая неоднозначность делают распознавание сцены более сложной проблемой по сравнению с другими задачами общей классификации изображений, такими как распознавание объектов.

В настоящей работе исследуется применение различных архитектур нейронных сетей для задачи распознавания сцен.

В качестве основной архитектуры рассматривается недавно появившейся [2] подход к распознаванию Semantic Aware Recognition, основанный на мультимодальной сети, состоящей из семантической и RGB ветвей и модуля внимания, который с помощью этой архитектуры объединяет информацию об изображении и контексте.

Исследуется применение различных моделей сверточных нейронных сетей в качестве RGB-ветви данной архитектуры. А также сравниваются результаты данной сети с классическими сверточными нейронными сетями.

1. Сверточные нейронные сети

За последние несколько лет алгоритмы распознавания сцен претерпели значительную эволюцию в результате развития машинного обучения и глубоких сверточных нейронных сетей.

Нейронные сети значительно повысили эффективность распознавания сцен, хотя она все еще значительно ниже, чем в других задачах распознавания из-за таких проблем как большое количество разнообразных объектов, различной компоновки и семантической неоднозначности.

Были рассмотрены следующие сверточные нейронные сети:

- AlexNet;
- VGG;
- GoogLeNet;
- ResNet.

В последние годы основная тенденция повышения эффективности распознавания сцен была направлена на увеличение числа блоков сверточной нейронной сети. Однако их производительность не растет линейно с увеличением числа параметров сети. Например, VGG имеет в двадцать раз больше блоков, чем GoogleNet, но производительность VGG для распознавания сцен лишь на 2,9% и 1,6% лучше, чем у AlexNet и GoogleNet, соответственно. [3]

2. Сети внимания

В работе [2] была представлена новая стратегия улучшения распознавания сцен: использование информации объектного уровня для управления процессом обучения сцен.

Известно, что, при обучении моделей на ImageNet, существует определенная парадигма тонкой настройки, которая может привести к смещению моделей в сторону пространственного центра визуального представления. [4] Это означает, что модели становятся более способными распознавать объекты, которые находятся в центре изображения, но менее способными распознавать объекты, которые находятся на краях изображения. Это может привести к тому, что модели будут ошибаться при распознавании объектов в реальном мире, где объекты могут находиться в различных частях изображения.

Предлагаемый подход опирается на механизмы внимания, управляемые семантикой, для нахождения наиболее важных объектов сцены, вне зависимости от их расположения.

Основная идея, лежащая в основе этой модели, заключается в том, что признаки, извлеченные из областей RGB и семантической сегментации, дополняют друг друга для распознавания сцен. Если обучить сверточную нейронную сеть распознавать сцены только на

основе семантической сегментации входного изображения RGB, эта сеть, как правило, фокусируется на репрезентативных и различающихся объектах сцены. Поэтому эти представления нужно использовать для тренировки карты внимания. Затем эта карта используется для управления извлеченными из RGB представлениями, тем самым переориентируя их на изученное содержимое сцены.

Предлагаемая архитектура сети представлена на рис. 1. Пусть $I \in R^{w \times h \times 3}$ – нормализованное RGB изображение, и пусть $M \in R^{w \times h \times L}$ тензор оценок, представляющий семантическую сегментацию, где $M_{i,j} \in R^{1 \times 1 \times L}$ представляет собой распределение вероятности для i -го, j -го пикселя на L -множестве изученных семантических меток.

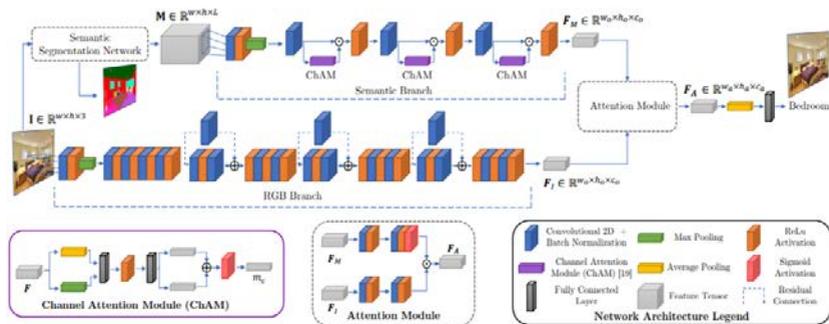


Рис. 1. Архитектура мультимодальной модели Semantic Aware Recognition

Сеть состоит из семантической ветви (рис. 1 сверху) и ветви RGB (рис. 1 внизу). Их выход представляет собой пару тензоров признаков, F_M и F_I соответственно, которые передаются в модуль внимания для получения окончательного тензора признаков F_A . Он подается в линейный классификатор для получения окончательного прогноза для задачи распознавания сцен класса K .

RGB-ветвь состоит из архитектуры сверточной нейронной сети. В эту ветвь подается изображение I и возвращается набор признаков на основе RGB $F_I \in R^{w_o \times h_o \times c_o}$, где w_o, h_o и c_o – ширина, высота и количество каналов выходной карты признаков. Эта ветвь включает оригинальные базовые блоки сверточной нейронной сети. Получившийся набор признаков F_I затем направляется в модуль внимания.

Семантическая ветвь вначале содержит сеть семантической сегментации, использующуюся для вывода тензора баллов семантической сегментации M , который затем подается в семантическую ветвь. M кодирует информацию о наборе значимых и репрезентативных объектов, их пространственных отношениях и расположении в сцене.

Семантическая ветвь берет тензор баллов семантической сегментации M , и возвращается набор семантических признаков $F_M \in R^{w_o \times h_o \times c_o}$, которые моделируют набор семантических меток и их семантические взаимозависимости в пространственном и канальном измерениях.

3. Модуль внимания

Модуль внимания используется для получения набора семантически взвешенных признаков $F_A \in R^{w_o \times h_o \times c_o}$, которые передаются линейному классификатору для получения конечной апостериорной вероятности сцены. Архитектура и параметры предлагаемого модуля семантического внимания изображены на рис. 2.

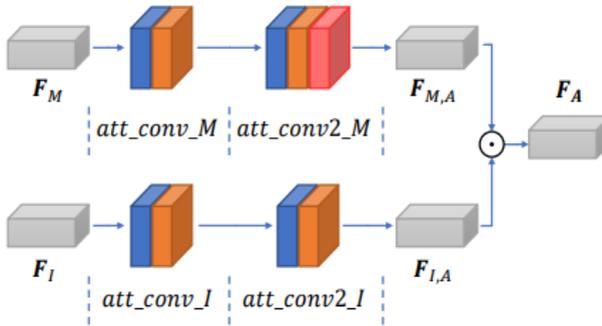


Рис. 2. Архитектура модуля внимания

Семантические представления $F_{M,A}$ получаются из выхода семантической ветви F_M с помощью двойного сверточного блока, за которым следует сигмоидный модуль, это позволяет усилить семантические классы, релевантные для данного изображения:

$$F_{M,A} = \sigma \left(\phi \left(W_M^A \left(\phi \left(W_M^A F_M + b_M^A \right) \right) + b_M^A \right) \right), \quad (1)$$

где σ и ϕ - сигмоидная и ReLU функции активации соответственно, W_M^A , b_M^A и W_M^A , b_M^A – веса и смещения двух сверточных слоев.

Аналогично, RGB признаки $F_{I,A}$ получаются из выходных данных RGB ветви F_I путем:

$$F_{I,A} = \phi \left(W_I^{A_2} \left(\phi \left(W_I^{A_1} F_I + b_I^{A_1} \right) \right) + b_I^{A_2} \right), \quad (2)$$

где $W_I^{A_1}$, $b_I^{A_1}$ и $W_I^{A_2}$, $b_I^{A_2}$ – это веса двух сверточных слоев данной ветви.

Формирование F_A выполняется простым перемножением этих двух представлений:

$$F_A = F_{I,A} \odot F_{M,A}. \quad (3)$$

Затем F_A направляется в классификатор, состоящий из среднего пула, отсева и полносвязанного слоя, генерируя вектор признаков $f \in R^K$. Вывод апостериорных вероятностей сцены $y \in R^K$ по f достигается с помощью логарифмической нормализованной экспоненциальной (логарифмической softmax) функции $\gamma(f) : R^K \rightarrow R^K$:

$$y_k = \gamma(f) = \log \left(\frac{\exp(f_k)}{\sum_i \exp(f_i)} \right), \quad \forall i = 1, \dots, K, \quad (4)$$

где y_K – апостериорная вероятность для класса K при заданном векторе признаков f .

4. Реализация

Для проведения исследований были реализованы следующие нейронные сети:

- AlexNet;
- VGG;
- GoogLeNet;
- ResNet;
- Semantic Aware Recognition с различными вариантами RGB-ветви.

Для реализации был использован набор данных MIT Indoor-67, фреймворк для глубокого обучения PyTorch и библиотека TorchVision.

Разделение данных происходило с помощью функции `random_split`, которая разделяет набор данных на части случайным образом. В данном случае 15620 изображений были поделены на три части:

- 13 000 для обучающей выборки;
- 2 000 для валидационной выборки;
- 620 для тестовой выборки.

Модели для сверточных нейронных сетей импортировались с помощью модуля `TorchVision`.

Модель `Semantic Aware Recognition` для нашей задачи строилась на основе соответствующего исходного кода данной модели с последующими заменами RGB ветви модели. Например, для использования в качестве RGB ветви сети `ResNet-18` необходимо импортировать эту модель, заменить начальный слой и энкодер (листинг 1).

Листинг 1

RGB ветвь для сети ResNet-18

```
# RGB ветвь
if arch == 'ResNet-18':
    # Импорт модели ResNet-18
    base = resnet.resnet18(pretrained=True)

    size_fc_RGB = 512
    sizes_lastConv = [512, 512, 512]

self.in_block = nn.Sequential(
    nn.Conv2d(3, 64, kernel_size=7, stride=2, padding=3,
    bias=False),
    nn.BatchNorm2d(64),
    nn.ReLU(inplace=True),
    nn.MaxPool2d(kernel_size=3, stride=2, padding=1,
    return_indices=True)
)
self.encoder1 = base.layer1
self.encoder2 = base.layer2
self.encoder3 = base.layer3
self.encoder4 = base.layer4
```

5. Результаты

В таблице 1 представлены результаты сравнения различных вариантов сверточных нейронных сетей между собой. В таблице 2 представлены результаты тестирования модели `Semantic Aware Recognition` для различных вариантов RGB-ветвей.

Таблица 1

*Результаты тестирования различных вариантов сверточных
нейронных сетей*

Нейронная сеть	Точность на тестовом наборе (%)	Время обучения (с)	Размер (МБ)
AlexNet	53,6	2 487	233
VGG	58,3	5 524	528
GoogLeNet	55,1	1 286	127
ResNet	76,5	1 481	45

Таблица 2

*Результаты тестирования модели Semantic Aware Recognition
для различных вариантов RGB-ветвей*

Вариант RGB-ветви	Точность на тестовом наборе (%)	Время обучения (с)	Размер (МБ)
AlexNet	62,1	20 083	475
VGG	70,3	41 712	762
GoogLeNet	67,7	10 368	368
ResNet	85,6	11 986	285

На основе полученных результатов, самой точной сверточной нейронной сетью для данной задачи среди остальных сверточных сетей оказалась сеть ResNet с процентом точности на тестовом наборе 76,5. Ее использование в качестве RGB-ветви в модели Semantic Aware Recognition дало точность 85,6 процента, что на 9,1 процентов больше, чем точность классической версии сети. Для остальных моделей точность в случае использования их в качестве RGB-ветви в модели Semantic Aware Recognition тоже оказалась выше, чем в классической версии сети.

Взамен увеличения точности сети увеличилось ее время обучения и итоговый размер. Увеличение точности нейронной сети требует более сложных вычислительных операций, что занимает больше времени для выполнения. Кроме того, при улучшении сети ее размер может увеличиться, поскольку ей может потребоваться больше памяти для хранения весов и параметров.

Заключение

В рамках данной работы было исследовано применение различных архитектур нейронных сетей для задачи распознавания сцен.

В качестве основной архитектуры рассматривался недавно появившийся подход к распознаванию Semantic Aware Recognition, основанный на мультимодальной сети, состоящей из семантической и RGB ветвей и модуля внимания, который с помощью этой архитектуры объединяет информацию об изображении и контексте.

Исследовалось применение различных моделей сверточных нейронных сетей в качестве RGB-ветви данной архитектуры.

В результате механизм внимания и управление семантикой позволил увеличить точность распознавания по сравнению с классическими моделями сверточных нейронных сетей, взамен на увеличение времени обучения и размера сети.

Список литературы

1. Quattoni, A. Recognizing indoor scenes / A. Quattoni, A. Torralba // Conference on Computer Vision and Pattern Recognition. – 2009. – № 5. – P. 413-420.
2. Lopez-Cifuentes, A. Semantic-aware scene recognition / A. Lopez-Cifuentes, M. Escudero-Vinoloa // International Conference on Computer Vision. – 2019. – № 2. – P. 106-125.
3. Zhou, B. Places: A 10 million image database for scene recognition / B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, A. Torralba // Transactions on Pattern Analysis and Machine Intelligence. – 2018. – № 11. – P. 1452–1464.
4. He, K. Rethinking imagenet pre-training / K. He, R. Girshick // International Conference on Computer Vision. – 2019. – № 13. – P. 1918–1927.

Обзор методов обучения с подкреплением для решения бизнес-задач на основе имитационного моделирования

Д. А. Денисова

Студент магистратуры

В. В. Гаршина

Доцент

Введение

Применение систем искусственного интеллекта в промышленности и для задач бизнеса связано с определенными трудностями – архивные данные могут не соответствовать текущему и будущему положению дел, прямое взаимодействие с контроллерами может быть трудно осуществимым, дорогостоящим или вовсе невозможным. Здесь и находит применение сочетание обучения с подкреплением и имитационного моделирования. В зависимости от требуемых результатов, могут использоваться подходы, основанные на оптимизации политик, функции награды или, если с помощью нейросети создается модель, оптимизация действий может проводиться с помощью внешнего планировщика.

Применение моделирования для виртуальных тренировок политик и в системах реального времени в настоящее время находит все новые применения. Методы глубокого обучения (deep learning, DL) и обучения с подкреплением (reinforcement learning, RL) становятся неотъемлемой частью систем для задач широкого применения, хотя и не единственной – другие подходы, такие, как байесовский вывод и методы, основанные на символьных вычислениях, также играют важную роль [1].

Цель данного исследования – провести обзор существующих методов использования имитационного моделирования для тренировки систем искусственного интеллекта, различных подходов к модернизации этих методов и их применения для задач реального мира.

1. Обучение с подкреплением и имитационное моделирование

Обучение с подкреплением (RL) – метод машинного обучения, в котором агенты обучаются, взаимодействуя со средой и получая вознаграждение (положительное или отрицательное) в качестве

обратной связи. Его цель – выработка агентами стратегии, обеспечивающей максимальную суммарную награду.

Основные элементы:

- Политика – определяет, как агент реагирует на различные состояния среды и как он выбирает действия в соответствии с ними;
- Награды – численные значения, получаемые агентом из среды в ответ на пару «состояние-действие»;
- Функция стоимости – функция, определяющая награду в долгосрочной перспективе; подсчитывается, начиная с некоторого определенного состояния, следует определенной политике;
- Модель среды определяет поведение окружения, которое позволяет алгоритму показывать лучшие результаты, исходя из его особенностей [2]. Например, AlphaGo от DeepMind пыталась имитировать уже сыгранные партии, в то время как AlphaGo Zero не подражала игрокам, разрабатывая собственные, неизвестные людям стратегии, многократно разыгрывая позиции сама с собой [3].

Для промышленности среда создается сочетанием модели цифрового двойника процесса, логов и архивных данных. В процессе работы модели осуществляются наблюдения, которые с помощью механизмов оптимизации поведения агентов формализуются в виде политик, применяющихся в решениях производственных задач или использующихся для дополнительного обучения и расширения модели (рис. 1).

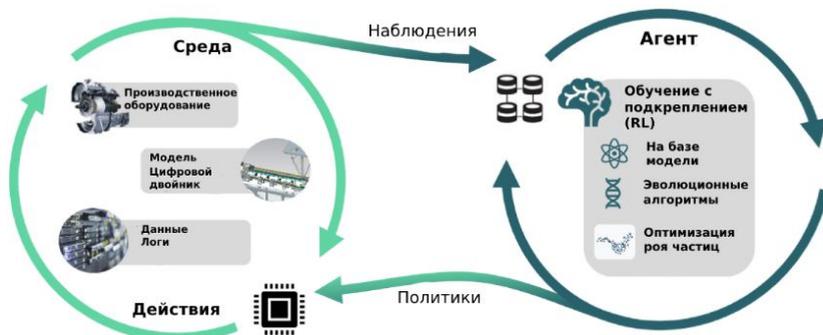


Рис. 1. Схема применения RL и моделирования в промышленности [4]

Дальнейшее развитие подход получил в глубоком обучении с подкреплением (deep reinforcement learning, DRL). В то время, как предыдущие реализации обучения с подкреплением требовали аккуратного подбора признаков, основанного на человеческом знании и опыте, глубокие нейронные сети могут автоматически находить признаки малой размерности в многомерных данных. Это привело к расширению применения RL, сначала с использованием одного агента – в таких задачах, как навигация робота внутри помещения и управления ресурсами, а в последнее время и с использованием мультиагентного подхода – для управления дорожным движением, автономного вождения, эффективного энергопотребления, а также для стратегических игр, таких, как Dota2 и Starcraft [5].

2. Подходы

Самым элементарным подходом представляется сочетание модели и алгоритма по отдельности. Изначально проводится запуск модели, затем через определенный промежуток времени она останавливается, данные передаются в модуль ИИ, затем параметры модели корректируются, и процесс продолжается до получения требуемого результата. Расширение имитационного моделирования с помощью машинного обучения делает возможной реализации адаптивного поведения с минимальным вмешательством человека, что позволяет создавать более эффективные динамические системы для стремительно изменяющихся сред [6,7].

Среди современных активно применяющихся и модернизирующихся подходов RL основными можно выделить следующие:

1. Q-обучение

Проблема RL может быть сформулирована как марковский процесс принятия решений (MDP), который включает в себя множество состояний S , множество действий A , функцию награды $R : S \times A \rightarrow \mathbb{R}$ и вероятностную функцию перехода $P : S \times S \times A \rightarrow [0, \infty)$. На каждом шаге агент обозревает состояние $s \in S$, производит действие $a \in A$, получает награду $r \in R$ и переходит в состояние $s' \in S$. Главная задача метода – найти оптимальную политику $\pi : S \rightarrow A$, которая максимизирует долгосрочную награду. Q-функция (от англ. Quality) используется, чтобы оценить «качество» действия в данных обстоятельствах. Она определена как:

$$Q(s, a) = E \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \mid s_0 = s, a_0 = a, \pi \right], \quad (1)$$

где γ - коэффициент дисконтирования, R – функция награды, t – шаг и π - используемая политика. Таким образом, агент оценивает ценность каждого из рассматриваемых действий и выбирает одно из них – максимальное или случайное в случае с эpsilon-жадным алгоритмом.

Для создания Deep Q-learning алгоритма глубокая нейронная сеть (сверточная, CNN) используется как нелинейное приближение Q-функции. DQN обновляется с помощью минимизации функции потерь:

$$L(\theta) = E_{e \sim D} \left[(y - Q(s, a | \theta))^2 \right], \quad (2)$$

где θ - параметры сети, $e = \langle s, a, r, s' \rangle$ - опыты агента в буфере D, и y – аппроксимация оптимального значения действия, основанная на уравнении Беллмана:

$$y = r + \gamma \max_{a'} Q'(s', a' | \theta'), \quad (3)$$

где r – награда, s' - следующее состояние и a' - соответствующее ему действие. Q' представляет целевую сеть, чьи параметры θ' скопированы из предыдущей итерации. Градиент потерь такой функции:

$$\nabla_{\theta} L(\theta) = E_{e \sim D} \left[(Q(s, a | \theta) - y) \nabla_{\theta} Q(s, a | \theta) \right] \quad (4)$$

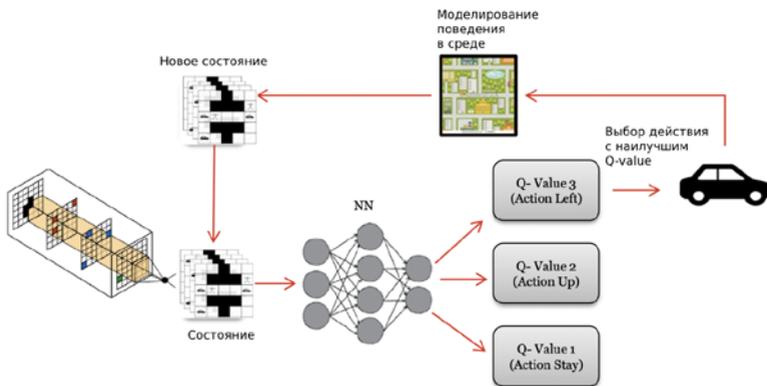


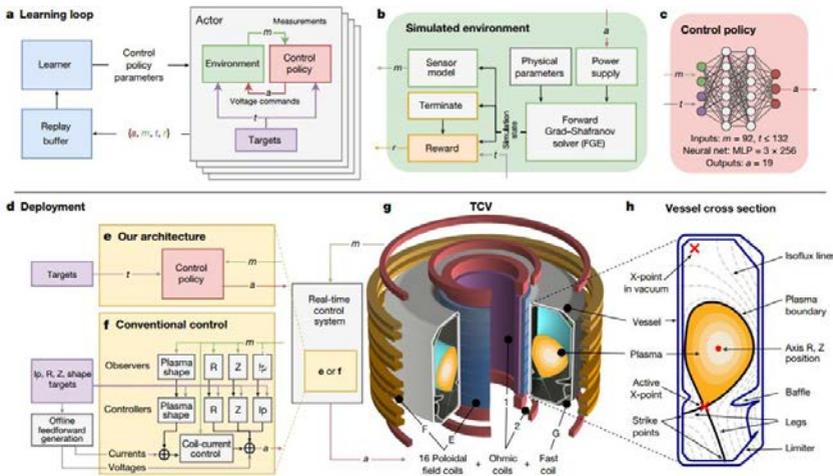
Рис. 2. Пример применения DQN для автономного вождения [8]

Параметры Q оптимизируются с помощью стохастического градиентного спуска с применением адаптивных коэффициентов скорости обучения (Adam, RMSProp и т.д.).

2. Методы «актор-критик»

В данных методах применяется актер, представленный стратегией π_θ с набором параметров θ , который опирается на оценку критика, представляющего значение функции полезности $U(s)$, $Q(s,a)$ или $A(s,a)$ с параметрами ϕ . Актер выбирает действия, учитывая свою ценность и стремясь получить награду от критика, а критик оценивает действия актора, основываясь на Q-функции. Но из-за того, что мы выбираем наборы стохастически, может возникать высокая дисперсия градиентов, и распределение политики может отклоняться в неоптимальном направлении [9]. Для модернизации подхода применяются различные версии метода – актер-критик с преимуществом, разные виды pre-trained данных и т.д.

В качестве примера промышленного применения данного метода можно привести работу по контролю плазмы в токамаке (рис. 3):



а) цикл обучения, б) цикл взаимодействия с моделью среды, в) политика контроля, д-ф) взаимодействие модели с системой контроля реального времени, г) отображение контроллера, х) схема резервуаров и плазмы [10]

Рис. 3. Архитектура дизайна разработанного контроллера

В данном исследовании был применен модернизированный метод актора-критика; несмотря на то, что он проводился с ограничениями, авторы утверждают, что этот подход применим в дальнейших испытаниях и что система с перемещением контроллеров может быть

эффективной и безопасной, и позволяет открывать новые способы манипуляции плазмой в резервуаре.

Преимущества данных методов:

- оптимальные действия ищутся с помощью градиентного спуска, показывают лучший результат;
- можно использовать небольшие, быстро обучаемые сети;
- эффективны, если можно выделить ключевые для решения задачи факторы.

Недостатки:

- требуется много обучающих примеров;
- неэффективны для задач с большим числом степеней свободы;
- могут эксплуатировать уязвимости в модели, если градиентный спуск сходится к неоптимальному действию;
- для любой новой задачи нужно полностью заново обучать модель.

3. Модернизация подходов для бизнес-задач

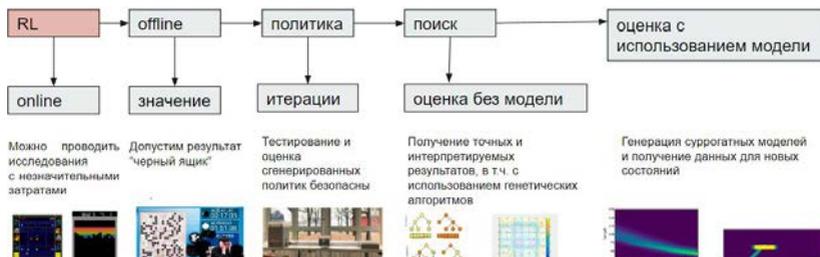


Рис. 4. Направления использования в промышленности на основе данных [4]

Требования промышленности приводят к следующим вариациям: online-обучение, если оно возможно с незначительными затратами (пример – тестовые алгоритмы игры в Atari); направленное на значения, если допустим черный ящик (AlphaGo); итеративное применение политик, если их тестирование и оценка возможны и безопасны (контроль газовых турбин), поиск и оценка без модели, если в результате мы получим точные и интерпретируемые результаты (энергоэффективное расписание поездов) или же создание суррогатных моделей и испытание найденных политик для новых условий на “испытательном полигоне” (рис.4).

Заключение

Данная статья посвящена обзору базовых методов и подходов использования обучения с подкреплением в сочетании с имитационным моделированием для задач производства. Имитационное моделирование переживает возрождение, расширяя концепцию до цифрового двойника предприятия и участвуя уже во всем жизненном цикле производства в реальном времени, и методы машинного обучения делают это возможным. Корпорации, такие, как концерн Siemens, успешно применяют методы в системах управления, добиваясь экономически эффективных результатов. В настоящий момент направление активно развивается, публикуется огромное количество разработок для широкого применения в большом количестве сфер – от робототехники и управления роем дронов до кибербезопасности.

Список литературы

1. Deep Learning, reinforcement learning, and world models / Matsuo Yutaka [et al.] // *Neural Networks*. – 2022. – Vol. 152. – Issue C. – P. 267-275.
2. Николенко, С. Глубокое обучение. Погружение в мир нейронных сетей / С. Николенко, А. Кадурын, Е. Архангельская. – СПб: Питер, 2018. – 480 с.
3. Mastering the game of go without human knowledge / David Silver [et al.] // *Nature*. – 2017. – Vol. 550. – P. 354-359.
4. Sterzing, V., Weber, M. Reinforcement Learning in the Physical World [Электронный ресурс] : Siemens, Data Analytics & AI, 2022. Режим доступа: <https://www.anyscale.com/events/2022/03/29/reinforcement-learning-in-the-physical-world>
5. Deep Multiagent Reinforcement Learning: Challenges and Directions / Annie Wong [et al.] // *Artificial Intelligence Review*. – 2022. – Vol. 56 – P. 5023-5056.
6. Elbattah, M. ML-Aided Simulation: A Conceptual Framework for Integrating Simulation Models with Machine Learning / M. Elbattah, O. Molloy. // *Principles of Advanced Discrete Simulation (SIGSIM-PADS'1)*. – 2018. – P. 33-36.
7. Борщев, А. В. Имитационные модели как виртуальная среда для обучения и тестирования искусственного интеллекта для бизнес-приложений / Борщев А. В., Mahdavi A. // *Имитационное моделирование. Теория и практика (ИММОД-2019)*. – 2019. – Т.1 – С. 20-29.
8. Wallis, L., Paich, M., Ma, S. Integrating Artificial Intelligence and Simulation Modeling [Электронный ресурс]. – Режим доступа:

<https://www.anylogic.com/blog/integrating-artificial-intelligence-with-simulation-modeling/>

9. Кохендерфер, М. Алгоритмы принятия решений / М. Кохендерфер, Т. Уилер, К. Рэй ; пер. с англ. В. С. Яценкова. – М.: ДМК Пресс, 2023. – 684 с.

10. Magnetic control of tokamak plasmas through deep reinforcement learning / Degraeve J. [et al] // Nature – 2022. – Vol. 02. – P. 414-419.

Математическое и программное обеспечение для автоматизации идентификации типа тары в задаче переработки вторсырья

Н. В. Дешин

Студент бакалавриата

М. А. Дрюченко

Доцент

Введение

Одной из проблем в современном мире, является скорость получения информации. Одна из причин этого - недоступность старых данных в цифровом формате. Для ввода данных в электронную таблицу или цифровой документ требуется ручной труд. Если данные имеют определенную структуру, к примеру, заданы в табличном формате, на это может уходить десятки минут, что в рамках тысяч документов превращается в месяцы работы. Изображение документа занимает гораздо больше места для хранения, чем тот же документ в цифровом формате. Широкое распространение цифровых камер, средств оцифровки и сканирования привело к активному развитию методов детектирования и распознавания объектов на изображениях. Текстовая информация по-прежнему является наиболее надежным признаком для сопоставления, классификации и описания изображений. Тонны архивных бумаг, чеков и счетов проходят сканирование и оцифровку во многих отраслях: в розничной торговле, логистике, банковских услугах и т. п. Компании получают конкурентное преимущество, если быстро оцифровывают и находят нужную информацию. Оцифровка позволила нам автоматически заполнять юридические документы и заявки на услуги, а также открыла доступ к аналитике фискальных чеков, отслеживанию динамики цен и суммарных трат.

В настоящее время наиболее успешные системы оптического распознавания символов (OCR-системы) распознают тексты на изображениях, составленных из стандартных шрифтов, не подвергнутых шуму и искажениям, с достаточно высокой точностью.

Основная задача OCR-систем заключается в назначении фрагменту изображения текста соответствующей символьной информации.

Основные требования к OCR-системам включают возможности сохранения форматирования исходных документов: шрифтов, абзацев, таблиц, графиков и изображений.

1. Типовая схема обработки изображений и распознавания текста

Обобщенная схема обработки изображений объектов вторичного сырья при распознавании типа материала, из которого они изготовлены, представлена на рисунке 1.

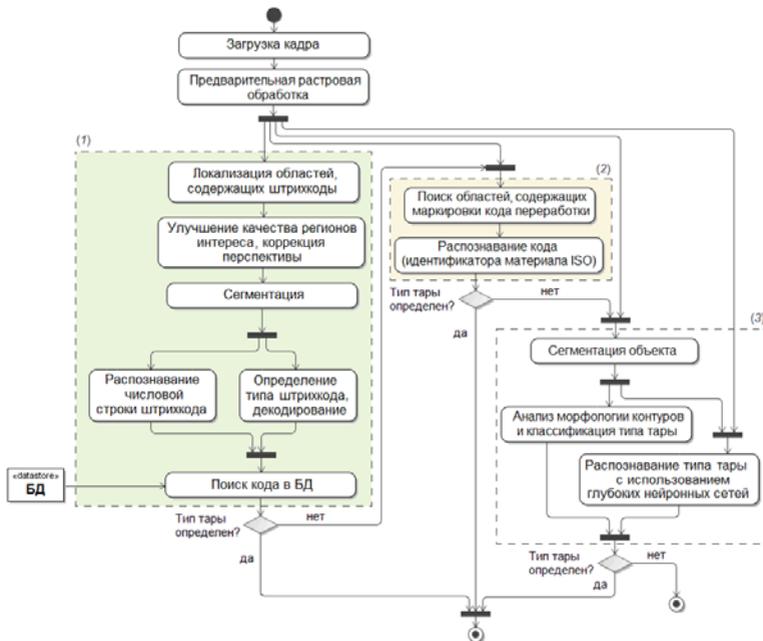


Рис. 1. Общая схема обработки изображения при распознавании материалов вторичного сырья

Схема предполагает этап предварительной растровой обработки, направленной на улучшение качества изображения, избавление от шумов и улучшение точности распознавания, а также последовательное выполнение трех вариантов алгоритмов идентификации типа тары, основанных на:

1. распознавании и извлечении информации из штрихкодов,
2. распознавании и анализе маркировки кодов переработки,
3. анализе формы и внешнего вида тары.

В ходе данной работы блок анализа формы и внешнего вида тары не рассматривается, так как это другая область, которую не затрагивает тема работы.

2. Предварительная растровая обработка

Процесс обработки изображений состоит из ряда этапов, среди которых одним из наиболее важных является предварительная обработка изображений, которая представляет собой самостоятельный процесс. Предварительная обработка изображений отвечает за фильтрацию шумов, помех и искажений на изображении. К задачам предварительной обработки изображений относятся:

1. подавление шумов,
2. изменение диапазона яркостей,
3. изменение резкости и коррекция цветов,
4. бинаризация,
5. морфологические преобразования.

Данный этап отвечает за нормализацию входных параметров с целью увеличения точности распознавания информации на изображении. Наборы методов предварительной обработки изображения могут различаться исходя из ситуации применения.

3. Извлечение информации из штрихкодов

Текущий вариант алгоритмов идентификации типа тары на основе анализа штрихкодов ориентирован на работу с линейными штрихкодами, наносимыми на этикетки или саму упаковку. В данных штрихкодах зашифрован уникальный код, по которому запрашивается в специализированных БД связанная с ним информация, такая как тип товара, цена и т.п.

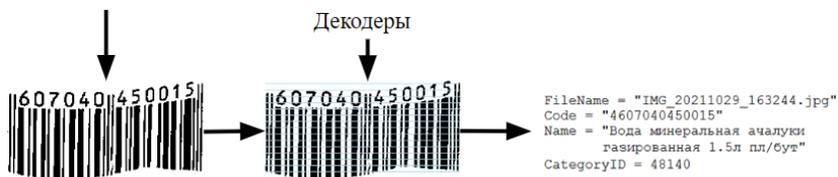


Рис. 2. Общая схема обработки штрихкода

Возможны случаи, когда нет возможности извлечь информацию из полос штрихкода. Такое бывает, когда штрихкод затирается, или полосы накладываются друг на друга. В таком случае стоит воспользоваться алгоритмом распознавания числовой последовательности.

Данную задачу можно решать с помощью метрик или с использованием нейронных сетей.

В первом случае числовая последовательность разбивается на фрагменты, содержащие цифры, происходит сбор метрик и их сравнение со значением из словаря.

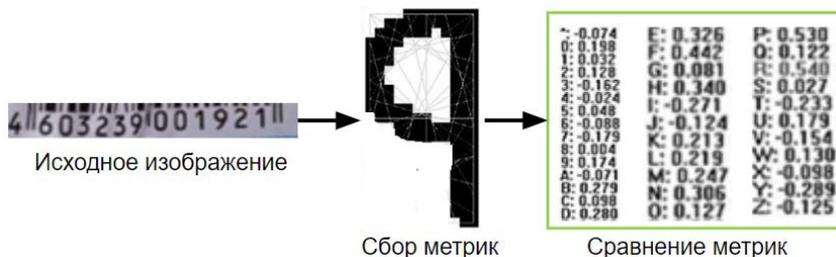


Рис. 3. Сравнение метрик для символа

Данный способ прост в реализации, но очень сильно зависим от шумов и искажений изображения. Даже, если изменить шрифт или размер символа, точность классификации сильно упадет.

В таком случае стоит воспользоваться нейронными сетями, которые выделяют на изображении общие структурообразующие признаки, после чего на их основе производят классификацию.

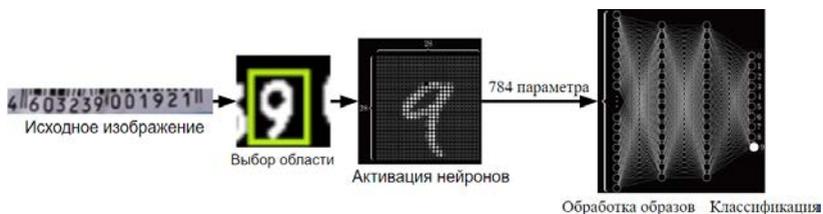


Рис. 4. Схема работы свёрточной нейронной сети для распознавания числовой последовательности

4. Извлечение информации из кодов переработки

Следующий блок алгоритмом направлен на обработку изображения с целью обнаружения кодов переработки. Петля Мебиуса – стандартный маркер/символ кода переработки. Тип материала определяется по числу внутри/рядом с треугольником или по буквенным обозначениям.

Данную задачу также можно решить с помощью метрик или нейронных сетей. В случае метрик стоит обратить внимание, что с петля Мебиуса похожа на треугольник, можно обнаружить его с помощью

сравнения таких параметров, как периметр/площадь фигуры, количество углов.

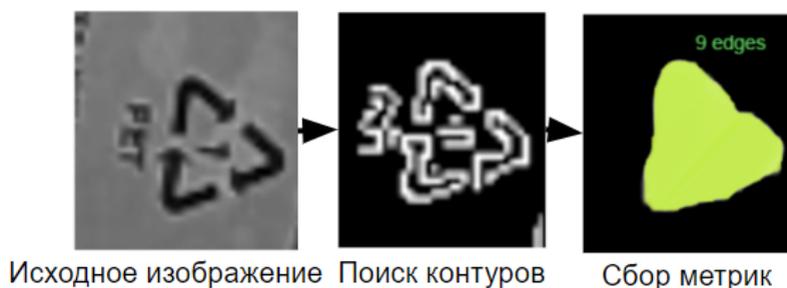


Рис. 5. Сбор метрик из изображения, содержащего коды переработки

Но стоит отметить, что точная форма фигуры не определена и может отличаться в зависимости от производителей, от чего может сильно уменьшаться точность определения. В таком случае можно воспользоваться нейронными сетями для обнаружения и классификации

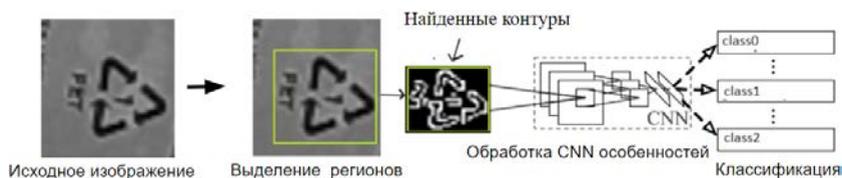


Рис. 6. Схема работы нейронной сети для классификации кодов переработки

Заключение

Данная статья посвящена разработке математического и программного обеспечения для автоматизации идентификации типа тары в задаче переработки вторсырья. Описаны алгоритмы предварительной обработки изображения, позволяющие избавиться от шумов и сегментировать текст. Также описаны алгоритмы, направленные на извлечение информации из штрихкодов и числовых последовательностей, кодов переработки.

Список литературы

1. Болотова, Ю. А. Обзор алгоритмов детектирования текстовых областей на изображениях и видеозаписях / Ю. А. Болотова,

В. Г. Спицын, П. М. Осина // Компьютерная оптика. – 2017. – Т. 41. – № 3. – С. 441-452.

2. О локализации плоских объектов на изображениях со сложной структурой проективных искажений / Д. В. Тропин [и др.] // Информационные процессы. – 2019. – Т. 19, № 2. – С. 208-229.

3. Фисенко, В. Т. Компьютерная обработка и распознавание изображений : учебное пособие / В. Т. Фисенко, Т. Ю. Фисенко ; СПбГУ ИТМО, 2008. – 192с.

4. Гонсалес, Р. Цифровая обработка изображений / Р. Гонсалес, М. Вудс. – М.: Техносфера, 2005. – 1072 с.

5. Исафилов, Х. С. Исследование методов бинаризации изображений / Х.С. Исафилов // Вестник науки и образования. – 2017. – Т.2. – № 6. – С. 43-50.

Разработка системы управления отоплением с использованием нечеткого логического

Д. П. Донских

Студент магистратуры

Я. А. Воронцов

Доцент

Введение

В настоящее время большое внимание уделяется охране условий труда рабочих. Согласно исследованиям, микроклимат в производственных помещениях оказывает существенное влияние на здоровье работников [1]. Работа в условиях низкой температуры приводит к снижению иммунитета, что делает человека уязвимым к инфекционным заболеваниям [7][8]. В большой степени негативному влиянию низких температур в помещении подвержены офисные работники. Они вынуждены подолгу находиться в малоподвижном положении, сидя за рабочим столом, и зачастую не могут восполнить теплотери физической работой.

Повышенная температура так же негативно влияет на состояние работника. Она может приводить к недостаточному насыщению организма кислородом, что может привести к нарушению активности головного мозга, снижению концентрации и производительности.

Влажность воздуха так же является важным показателем. Низкая или высокая относительная влажность в помещении может так же приводить к нарушениям работы сердца, почек, ухудшению общего самочувствия.

Таким образом необходимо поддерживать некоторую комфортную температуру и влажность в помещении. Но комфортная величина может варьироваться в зависимости от времени суток, сезона, температуры и скорости ветра на улице. Именно определению и поддержанию комфортных условий в помещении посвящена данная работа.

1. Переменные

Понятие комфортной температуры является примером нечеткой величины, то есть величины без четко заданного числового значения, а

находящегося в некотором интервале. Такая величина называется лингвистической переменной. Ее можно определить как пятерку

$$\{x, T(x), X, G, M\} \quad 1$$

где x — имя переменной; $T(x)$ — некоторое множество значений лингвистической переменной, каждое из которых является нечеткой переменной на множестве X ; G есть синтаксическое правило для образования имен новых значений; M есть семантическая процедура, позволяющая преобразовать новое имя, образованное процедурой, в нечеткую переменную (задать вид функции принадлежности), ассоциирует имя с его значением, понятием.

Время суток, сезон, температура и скорость ветра на улице рассматриваются как нечеткие лингвистические переменные:

1. Температура в помещении: прохладно, нормально, умеренно тепло, тепло.
2. Температура на улице: холодно, тепло, жарко.
3. Скорость ветра на улице: слабый, умеренный, сильный.
4. Время суток: утро, день, вечер, поздний вечер, ночь.
5. Влажность: низкая, нормальная, высокая.

В качестве функции принадлежности могут использоваться различные функции, определяющие степень принадлежности переменной к нечеткому множеству. Например, для переменной «температура в помещении» используется колоколообразная функция, заданная формулой:

$$f(x; a, b, c) = \frac{1}{1 + \left| \frac{x - c}{a} \right|^{2b}} \quad 2$$

где a, b, c — некоторые коэффициенты. Графически данная функция принадлежности может быть представлена в следующем виде [6].

Кроме колоколообразной функции принадлежности для задания других переменных используются треугольная, трапециевидная, s- и z-образные [5].

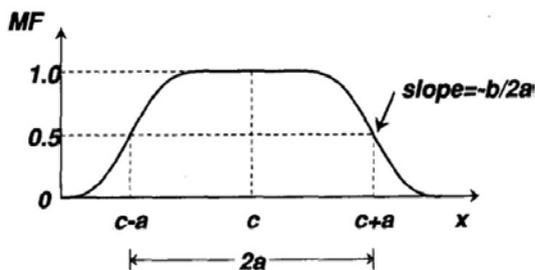


Рис. 1. Колоколообразная функция принадлежности

2. Набор правил

Для представленных выше переменных формируется база правил, позволяющих определять результирующие значения. В случае исследуемой задачи результирующим значением является величина комфортной температуры в помещении. База правил формируется на основании знаний и опыта эксперта в данной области. Для разрабатываемой системы были использованы следующие правила:

1. Если на улице «холодно», то в помещении должно быть «умеренно тепло»
2. Если на улице «тепло», то в помещении должно быть «нормально»
3. Если на улице «жарко», то в помещении должно быть «прохладно»
4. Если влажность «низкая», то в помещении должно быть «умеренно тепло»
5. Если влажность «нормальная», то в помещении должно быть «нормально»
6. Если влажность «высокая», то в помещении должно быть «прохладно»
7. Если сейчас «утро» или «поздний вечер», то в помещении должно быть «умеренно тепло»
8. Если сейчас «ночь», то в помещении должно быть «прохладно»
9. Если сейчас «день», то в помещении должно быть «нормально»
10. Если сейчас «вечер», то в помещении должно быть «тепло»
11. Если на улице «жарко» и ветер не «слабый», то в помещении должно быть «прохладно»
12. Если на улице «холодно» и ветер не «слабый», то в помещении должно быть «умеренно тепло».

3. Фаззификация, аккумуляция, дефаззификация

Чтобы работать с разработанными выше правилами, необходимо представить числовые значения переменных в нечетком виде. Данный процесс называется фаззификацией и заключается в определении степени принадлежности числового значения переменной каждому из понятий логико-лингвистической переменной [3][4].

Далее необходимо произвести нечеткий логический вывод. Для этого был использован метод MAX-MIN. В результате обработки всех правил по каждой выходной переменной образуется совокупность результирующих множеств, которые в конечном итоге объединяются (накладываются друг на друга). После этого осуществляется операция дефаззификации. В данной работе применялся метод нахождения центра масс (CoG) объединенного множества, который рассчитывается по формуле [3]:

$$y = \frac{\int_{Min}^{Max} x\mu(x)dx}{\int_{Min}^{Max} \mu(x)dx} \quad 3$$

где y – результат дефаззификации, x - переменная, соответствующая выходной лингвистической переменной, $\mu(x)$ - функция принадлежности нечеткого множества, соответствующего выходной переменной после этапа аккумуляции, Min и Max — левая и правая точки интервала носителя нечеткого множества рассматриваемой выходной переменной.

После выполнения описанных выше операций получим числовое значение целевой температуры, которую необходимо поддерживать.

4. PID регулятор

Для поддержания полученного значения целевой температуры используется ПИД регулятор. Регулятор формирует управляющий сигнал для нагревательных элементов, являющийся суммой трёх слагаемых, первое из которых пропорционально разности входного сигнала и сигнала обратной связи (сигнал рассогласования), второе — интегралу сигнала рассогласования, третье — производной сигнала рассогласования

$$u(t) = P + I + D = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de}{dt} \quad 4$$

где K_p , K_i , K_d — коэффициенты усиления пропорциональной, интегрирующей и дифференцирующей составляющих регулятора соответственно, $e(t)$ — величина отклонения текущего значения от целевого.

Для поддержания температуры, полученной после обработки входных данных контроллером, ее значение передается на вход PID регулятору. На выходе получаем значение в диапазоне $[0..1]$, которое используется для управления конвектором посредством ШИМ.

5. Разработка аппаратной части

В качестве контроллера для датчиков температуры используется ESP8266, смонтированный на плате Wemos D1 mini (рис. 2).



Рис. 2. Wemos D1 mini

Данный модуль имеет встроенный Wi-Fi, что позволяет отправлять собранные данные по протоколу HTTP главному управляющему устройству. В качестве непосредственно датчика использован компактный АНТ10 (рис. 3), имеющий разрешающую способность 0.01°C и точность $\pm 0.3^{\circ}\text{C}$, что значительно лучше, чем у широко распространенных АМ2302, часто использующихся для построения домашних метеостанций [2].

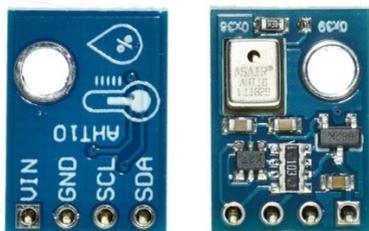


Рис. 3. АНТ10

Устройство в нескольких экземплярах собрано на печатной плате и смонтировано на стенах в нескольких помещениях.

Для сбора данных и управления несколькими конвекторами используется одноплатный компьютер Raspberry PI 4 с операционной системой на базе Debian. Он выступает и в качестве точки доступа Wi-Fi для датчиков температуры.

Для управления непосредственно мощной нагрузкой в виде конвекторов используются твердотельные реле SSR40, позволяющие пропускать до 40А. Данные реле ощутимо нагреваются при эксплуатации – необходимо использовать радиаторы для их охлаждения.

6. Разработка программной части

Программная часть системы реализована с использованием платформы .NET 7 и Blazor. В качестве СУБД применяется SQLite.

Разработанный веб-сервис позволяет подключать или отключать использование нечеткого логического вывода и поддерживать температуру, заданную вручную. При этом реализована возможность изменения параметров для каждого из помещений в отдельности.

Для сбора статистики о потреблении электроэнергии используются данные от датчиков тока, установленных на каждой из цепей управления конвекторами. На рис. 4 можно видеть почасовой график работы конвектора в одном из помещений и величину потребленной электроэнергии:



Рис. 4. График работы конвектора и потребленная мощность

Программная часть системы, в отличие от готовых решений, проектировалась с возможностью функционировать в автономном режиме без доступа к сети интернет, поскольку обработка информации и принятие решений происходит непосредственно на локальном устройстве.

Заключение

Данная статья посвящена разработке и реализации системы управления отоплением с использованием нечеткого логического вывода. Описаны алгоритмы определения целевой поддерживаемой температуры в зависимости от условий окружающей среды. В результате получилась полностью работоспособная система, состоящая из сети датчиков температуры и влажности, аппаратных средств управления нагревательными элементами с использованием ШИМ и веб-сервиса для управления параметрами системы и сбора статистики. Использование данного решения позволяет в автоматическом режиме поддерживать комфортную температуру в помещении, а также снизить энергозатраты на отопление за счет понижения температуры в ночной период и поддержания постоянной температуры без значительных колебаний в дневной. Согласно собранной статистики, уменьшение потребления электроэнергии составило около 3-5% по сравнению с поддержанием температуры, заданной вручную.

Список литературы

1. Васильев, А. В. Влияние микроклимата на производительность труда рабочих в РМЦ / А. В. Васильев, Е. П. Одноров. — Текст : непосредственный // Молодой ученый. — 2016. — № 1 (105). — С. 138-141. — URL: <https://moluch.ru/archive/105/24773/> (дата обращения: 11.05.2023).

2. Ушаков В. А., Донских Д. П., Теплякова М. А., Зуев С. А., Создание домашней метеостанции с низкой себестоимостью // сборник студенческих научных работ факультета компьютерных наук ВГУ. - Воронеж: Издательский дом ВГУ, 2018. - С. 335-341.
3. Леоненков А. В. Нечеткое моделирование в среде MATLAB и fuzzyTECHN. — СПб.: БХВ Петербург, 2005. — 736 с.
4. Простой регулятор на базе нечеткой логики. Создание и настройка // Хабр [Электронный ресурс] Режим доступа: URL: <https://habr.com/ru/articles/413539/> (дата обращения: 11.05.2023).
5. Основы нечеткой логики: Учебно-методическое пособие к практическим занятиям и лабораторным работам / Д.Р. Григорьева, Г.А. Гареева, Р.Р. Басыров - Набережные Челны: Изд-во НЧИ КФУ, 2018. - 42 с.
6. Nonlinear System VS Linear System // Research hubs URL: <https://researchhubs.com/post/mathsfundamentals/bell-shaped-function.html> (дата обращения: 11.05.2023).
7. Афанасьева Р. Ф., Прокопенко Л. В., Константинов Е. И. Тепловое состояние работающих в нагревающем микроклимате в теплый и холодный периоды года // Научно технический сборник «Вести газовой науки» 2013 № 2: [Электронный ресурс]. - Режим доступа: <http://cyberleninka.ru/article/n/teplovoe-sostoyanie-rabotayuschih-v-nagrevayuschemmikroklimate-v-teplyu-i-holodnyu-periody-goda>.
8. Кожевникова Н. Ю. Влияние метеорологических условий внутренней среды помещений на здоровье человека // Сборник материалов научно-практической конференции 4 февраля 2010 г. Екатеринбург: - УрГСХА, 2010. Ч. 1 С. 161- 165.

Адаптация веб-приложения «SQL портал» под мобильные устройства

Е. В. Елфимова

Студент магистратуры

Д. И. Соломатин

Старший преподаватель

Введение

В настоящее время известной проблемой, как на факультете компьютерных наук, так и в целом в любом процессе обучения IT-специалистов, является небольшое количество учебных баз данных, находящихся в открытом доступе. Выполнив поиск по сети Интернет, вы едва ли сможете найти около десятка таких учебных материалов. Проблема состоит в том, что данные из действующих баз данных не могут быть непосредственно выложены в общий доступ, так как практически во всех случаях они содержат конфиденциальную информацию, которую нельзя подвергать разглашению. Для использования таких данных в качестве учебного материала, они должны быть подвергнуты анонимизации, что является не самой тривиальной задачей.

С другой стороны, даже обезличенная база данных, готовая стать ресурсом для обучения и тренировки, скорее всего будет использована довольно узким кругом лиц, по причине отсутствия в сети Интернет веб-сервиса, позволяющего пользователям, не являющимся высококвалифицированными специалистами в области баз данных, опубликовать учебную базу данных, тем самым поделиться ей со всеми желающими. Отсутствие такого ресурса приводит к необходимости обучения на одних и тех же учебных базах в течение долгого времени, что не способствует развитию учебного процесса.

Ранее для решения описанных задач было разработано веб-приложение «SQL портал», позволяющее публиковать учебные базы данных и предоставляющее возможность исполнения запросов к ним на стороне клиента.

Благодаря сервису «SQL портал» можно собрать большое количество учебного материала и использовать его в академических

целях, например, проводить практические занятия по языку SQL на факультете компьютерных наук с использованием упомянутого ранее сервиса. Это предоставит выбор среди разнообразия опубликованных в приложении баз данных, как для преподавателей, так и для студентов, что будет способствовать поддержанию интереса к учебному процессу с обеих сторон.

В данной работе будет рассмотрен способ адаптации данного приложения под мобильные устройства, включающий в себя, как архитектурные и компонентные изменения, так и высокоуровневые изменения в интерфейсе пользователя.

1. Принцип работы веб и мобильной версии приложения «SQL портал»

Ключевым элементом легкости системы в обращении является отсутствие необходимости развертывания локальных или распределенных СУБД. Такая легкость достигается за счет использования AlaSql – легковесной клиентской базы данных, выполняемой в среде JavaScript, структуры которой размещаются в оперативной памяти. В предложенном решении SQL-запросы выполняются на клиенте, т.е. в браузере, общая схема работы с клиентской базой данных из веб-приложения представлена на рис. 1.

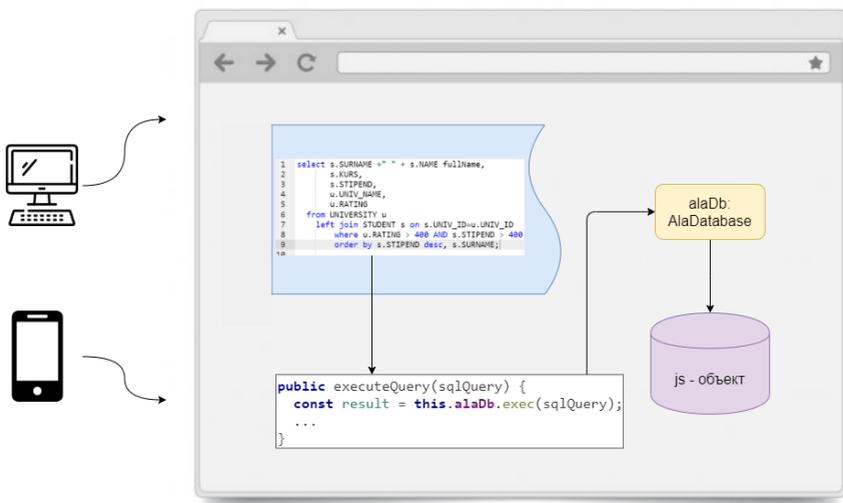


Рис. 1. Описание взаимодействия с клиентской базой данных

При адаптации приложения под мобильное устройство данный принцип был сохранен. При исполнении SQL запроса из мобильного клиента последовательность работы приложения остается неизменной за исключением того, что в качестве браузера выступает браузер мобильного устройства. Заданный пользователем запрос выполняется на мобильном устройстве к ранее загруженной и располагающейся в оперативной памяти мобильного устройства базе данных.

2. Изменение архитектуры веб-приложения «SQL портал» для адаптации под мобильные устройства

Для реализации веб-клиента приложения был выбран фреймворк Angular. При реализации веб-клиента была использована трехуровневая архитектура, характерная для выбранного фреймворка [1]. Схема архитектуры Angular приложения представлена на рис. 2.

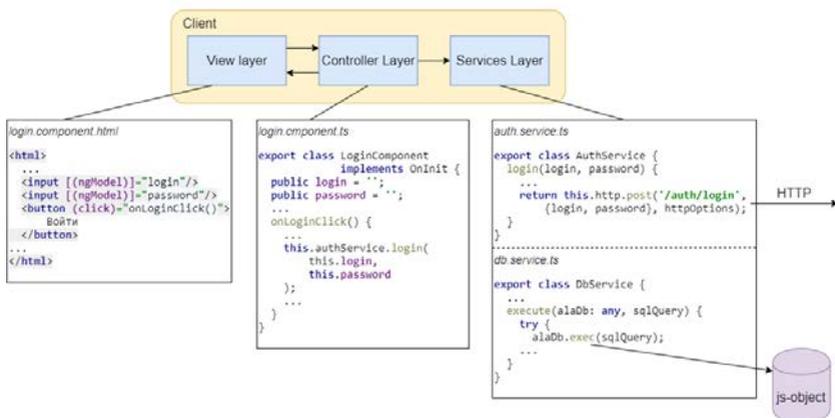


Рис. 2. Архитектура веб-клиента приложения

Рассмотрим более подробно уровни указанной схемы:

- уровень представления (View Layer) состоит из HTML-шаблонов, таблиц CSS соответствующих различным компонентам пользовательского интерфейса;
- уровень управления (Controller Layer) состоит из контроллеров Angular, склеивающих вместе данные, извлекаемые, соответственно, с сервера и из представления. Контроллер инициализирует модель представления и определяет, как представление должно реагировать на изменения модели и наоборот;
- уровень сервисов (Frontend Services Layer) – набор сервисов Angular, реализующих бизнес-логику и способных осуществлять

взаимодействие с REST API сервера, а также осуществляющих взаимодействие с клиентской базой данных.

Для создания приложения под платформу Android был использован кроссплатформенный фреймворк – Apache Cordova. Он позволяет использовать стандартные веб-технологии – HTML5, CSS3 и JavaScript для кроссплатформенной разработки [2]. Apache Cordova выступает в роли посредника или оболочки, которая оборачивает исходный код веб-приложения, позволяя ему исполняться на мобильном устройстве, а также обеспечивает возможность обращаться к API конкретных платформ (Android, iOS) через плагины [3].

Таким образом, благодаря Apache Cordova адаптация веб-клиента Angular под мобильные устройства была произведена без необходимости полного переписывания программного кода на другом языке или фреймворке, что видно из рис. 3.

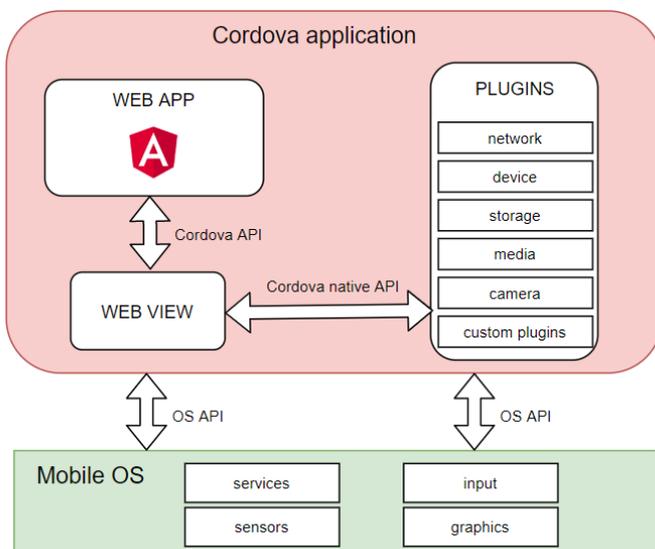


Рис. 3. Архитектура мобильного клиента приложения

Приложения с такой архитектурой называются гибридными.

3. Адаптация интерфейсов веб-приложения «SQL портал» под мобильные устройства

Основным недостатком использования уже существовавшего веб-интерфейса, представленного на рис. 4, в мобильной версии был ввод громоздкого SQL запроса с клавиатуры, что абсолютно неудобно.

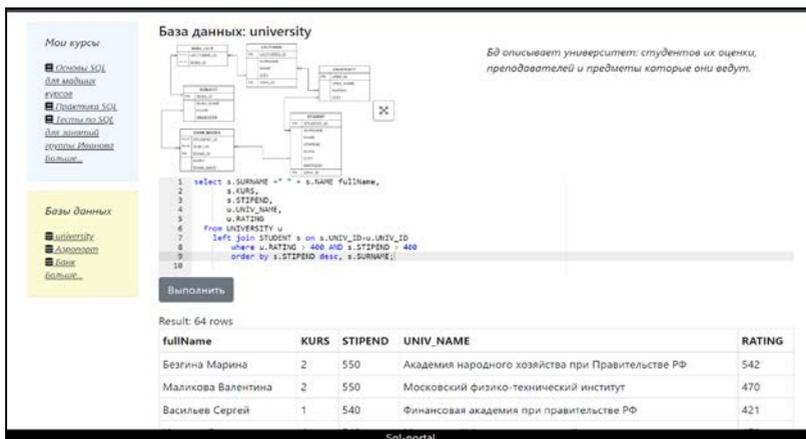


Рис. 4. Веб-интерфейс страницы исполнения запроса к базе данных

Поэтому в мобильном интерфейсе пользователю предоставляется словарь, содержащий все необходимые для составления запроса элементы – ключевые слова, выражения и т.д. Пользователь может перемещать соответствующие «блоки» по областям «Запрос» и «Словарь» с помощью механизма Drag-and-Drop, а также изменять порядок «блоков» в рамках одной области.

Мобильные интерфейсы приложения приведены на рис. 5.

Заключение

Данная статья посвящена деталям адаптации клиента веб-приложения «SQL портал» под мобильные платформы. В описанном подходе подчеркнуты технологии, позволяющие преобразовывать веб-приложение в мобильное гибридное приложение с минимальными изменениями исходного кода приложения. Такой подход позволяет одновременно поддерживать два и более вида клиентов, основанных на одной и той же кодовой базе, что облегчает поддержку приложения в будущем.

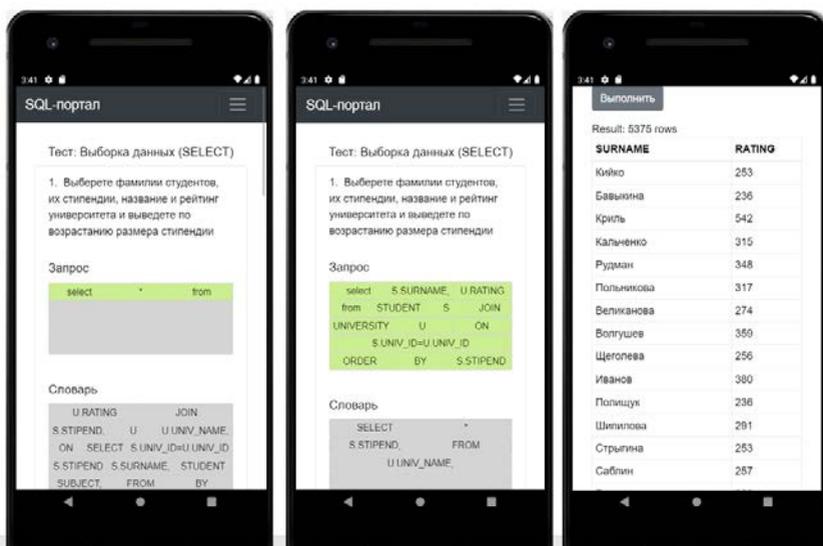


Рис. 5. Мобильный интерфейс исполнения запроса к базе данных

При этом, несмотря на легкость переносимости приложения на мобильную платформу, в пользовательский интерфейс приложения были внесены значительные изменения, так как из-за небольшого экрана мобильных устройств и отсутствия обособленной физической клавиатуры ввод SQL запроса может быть затруднительной задачей для пользователя.

Список литературы

1. Файн, Я. Angular и TypeScript. Сайтостроение для профессионалов / Я. Файн, А. Моисеев – М.: Питер, 2018. – 464 с.
2. Официальный сайт документации по фреймворку Apache Cordova [Электронный ресурс] : сайт. – Режим доступа : <https://cordova.apache.org/docs/en/latest/>
3. Васильев, Н. П. Введение в гибридные технологии разработки мобильных приложений. Учебное пособие / Н. П. Васильев, А. М. Заяц. – М.: Лань, 2020. – 160 с.

Анализ угроз и тенденций рынка информационных технологий

Жехангер Али Кадхем Жехангер

Студент магистратуры

А. Ю. Иванков

Доцент

Введение

Количество кибератак растёт каждый день. Далее перечислено несколько крупнейших кибератак первый квартал 2021 года [1].

Фонд Harris Federation, который управляет 50 начальными и средними школами в Лондоне и Эссексе, стал жертвой кибератаки 27 марта. Злоумышленники получили доступ к информационной системе и зашифровали данные. Телефонная связь также была отключена. 36.000 студентов остались без доступа к электронной почте.

Одна из крупнейших страховых компаний в США CNA подверглась кибератаке с использованием нового варианта вредоносного ПО Phoenix CruptoLocker. 21 марта злоумышленники развернули в сети CNA программу-вымогатель, зашифровавшую 15 тыс. устройств. Вирус также зашифровал компьютеры сотрудников, работающих удаленно, которые во время атаки были подключены к корпоративной виртуальной сети VPN.

Не менее 60000 компаний и организаций по всему миру подверглись хакерской атаке из-за уязвимости в Microsoft Exchange Server.

Асег подверглась атаке с использованием программы-вымогателя, в результате которой злоумышленники потребовали выкуп 50 миллионов долларов – самая крупная сумма на сегодняшний день. Злоумышленники поделились несколькими изображениями предполагаемых документов, которые им удалось украсть в ходе атаки, включая финансовые таблицы, банковские балансы и банковские сообщения.

Данные преступления не были связаны с непосредственным нанесением вреда здоровью людей, в отличие от следующего. Хакеры взломали компьютерную систему водоочистой станции во Флориде и

попытались отравить воду для 15 тысяч жителей, попытавшись увеличить содержание гидроксида натрия в воде более чем в 100 раз – с нормальных 100 ppm до 11100 ppm. Инцидент произошел утром 5 февраля. Хакеры сначала получили удаленный доступ к компьютеру работника предприятия через TeamViewer, а затем перехватили контроль над внутренними компьютерными системами контроля станции. В департаменте шерифа округа не подтвердили информацию о том, что именно ПО TeamViewer открыло путь хакерам.

Все эти события сигнализируют о возрастающей актуальности проблем обеспечения безопасности информационных систем как в коммерческом сегменте, так и в государственных учреждениях.

1. Анализ угроз

Современный цифровой мир внес в информационную безопасность и информационные технологии два элемента, которые существенно меняют представление о безопасности. Первый – люди – сотрудники, клиенты, пользователи, которые являются неотъемлемой частью любой ИТ-технологии. Второй элемент – различные персональные устройства (гаджеты) – мобильные телефоны, смартфоны, персональные компьютеры, которые обеспечивают взаимодействие людей с цифровой инфраструктурой.

Для любой организации можно выделить внешние факторы, которые могут нарушить стабильность и безопасность ее работы:

- регуляторы, которые предъявляют различные требования;
- клиенты и партнеры, которые подключаются к инфраструктуре организации и тем самым начинают прямо или косвенно влиять на технические системы организации;
- дочерние компании, которые образуют экосистему, являются потенциальной угрозой атаки через цепочки поставок;
- злоумышленники, которые используют различные технические каналы и средства, чтобы нарушить деятельность компании.

Сегодня безопасность перемещается в сторону сервисной модели, в которой обеспечение безопасности осуществляется аутсорсинговой компанией, а на специалиста по безопасности возлагается контроль данных процессов. Такая модель позволяет использовать привычные технологии даже в условиях нехватки персонала.

В финансовых организациях на кибербезопасность влияют [2-6]:

- угрозы;
- законодательство;
- информационные технологии;
- финансовые технологии (финтех).

Изменения в бизнесе автоматически влекут изменения кибербезопасности. Так, внедрение виртуальных помощников подразумевает решение вопросов защиты самих помощников, интернета вещей и технологий машинного обучения.

Существуют технологии, которые защищают ИТ инфраструктуру (файрволы на периметре, VPN для защиты каналов связи, DLP-системы для мониторинга коммуникаций в поисках утечек информации), а есть более крупный пласт задач, технологий и механизмов защиты, интегрированных в бизнес. Они имеют схожую структуру:

- угрозы;
- законодательство;
- информационные технологии;
- бизнес-процессы.

Безопасность сегодня является интегрированной, что больше способствует обеспечению информационной безопасности. В первую очередь это касается угроз. Необходимо знать, как действуют хакеры, среди которых можно выделить четыре типа злоумышленников:

- инсайдеры – сотрудники, желающие нанести ущерб организации, в которой работают по причине личной обиды, с целью обогащения или иным причинам;
- хакеры-любители – одиночки, действующие в основном из интереса, чтобы испытать себя;
- киберкриминал – преступные организации;
- хакеры на службе государства.

Очевидно, что хакерство в настоящее время является полноценным бизнесом. Данный бизнес занимает третье место по объему рынка и имеет:

- различные механизмы генерации прибылей;
- собственные валюты и инструменты ее обмена и обналичивания;
- различные экономические агенты (поставщики, производители, покупатели);
- широкий продуктовый ряд;
- особые экономические зоны;
- найм персонала;
- собственные торговые площадки;
- глобальные механизмы дистрибуции;
- саморегулирование;
- специальный инструментарий (техподдержка, обучение и повышение квалификации);

– психологи и физиологи.

Рынок киберпреступности активно развивается и будет развиваться. Основной сложностью борьбы с ним на уровне государства являются, во-первых, бюрократизированность, во-вторых, соблюдение требований законодательства, в отличие от преступников.

В ближайшем будущем можно выделить следующие актуальные угрозы в области информационных технологий.

1. Интернет вещей как площадка для реализации атак (маршрутизаторы, видеорекамеры, мобильные приложения).

2. Мобильные приложения как разносчики вредоносного ПО и «воры» данных.

3. Программы-вымогатели и сопутствующие им технологии.

4. Атаки «третьего поколения» (использование протокола DNS для скрытия активности ВПО, CDN, BGP).

5. Целенаправленные угрозы, реализующие полный цикл «kill chain».

6. Атаки на цепочки поставок (внедрение вредоносного кода в программное обеспечение в OpenSource).

7. Кража данных известных лиц с последующим шантажом.

8. Criminal-as-a-Service.

9. «Призраки интернета прошлого» (угрозы от устаревшего программного и программноаппаратного обеспечения, которое находится в интернете).

10. Нехватка людей для реализации возрастающего числа задач.

Реализация данных угроз может приводить к широкому числу последствий, среди которых можно выделить:

- взлом кардиостимулятора;
- смерть пациентов;
- падение курса акций;
- кража интеллектуальной собственности;
- снижение объема обогащенного топлива;
- влияние на результаты выборов;
- нарушение работы сети интернет;
- блокирование дверей;
- отказ тормозов в автомобиле;
- атака на систему управления полетами и транспортом.

2. Тенденции развития кибербезопасности

В области сетевых технологий в настоящее время существуют следующие тенденции развития.

1. Централизованное управление, включая смешанное (cloud + on-premise).

2. Интеграция с решениями по расследованию сетевых инцидентов.

3. Инспектирование зашифрованного трафика путем: расшифрования и работы с зашифрованным трафиком через гомоморфное шифрование или использование машинного обучения для обнаружения смыслов.

4. Микросегментация.

5. Интеграция с Threat Intelligence.

6. Переход на SaaS-модель.

7. Использование виртуальных межсетевых экранов.

8. Использование облачных межсетевых экранов с готовыми сценариями для облаков (AWS, Azure, GCP, Sber).

9. Оркестрация и автоматизация процессов, которые объединят в единую инфраструктуру безопасности множество продуктов.

10. Мониторинг аномалий. Корректнее учить аномалиям, когда они неизвестны или их сложно описать. Нужно анализировать другие виды данных (трафик, логи более всеобъемлющего состава, бизнес-активность организации).

Возрастает потребность в инструментах фильтрации и анализа зашифрованного трафика, который составляет более 50-60% от общего объема трафика. Необходимо уметь его мониторить, выявлять вредоносный код, защищать.

Тенденции защиты хостов:

- использование EDR решений (Gartner) или STAP (IDC).

Российские вендоры (Касперский, Юзергейт) активно входят в этот сегмент;

- все в одном (шифрование, DLP, сканер);

- локальная или облачная песочница для анализа подозрительных файлов;

- интеграция с сетевыми решениями за счет обмена данными или резолюциями;

- поддержка SaaS-модели;

- решения класса NG Endpoint Security (несколько антивирусов, antimalware-движков, контроль репутации файлов, корреляция IP и URL).

Тенденции развития сервисов управления идентификацией и контролем доступа (Identity & Access Management) [7]:

- управление привилегированными пользователями (PAM);

- многофакторная аутентификация;

- софт-токены, позволяющие реализовать второй фактор без дорогостоящих решений;

- биометрия;
- динамическая (контекстная) аутентификация;
- поддержка модели IaaS;
- интеграция с другими средствами защиты (МСЭ, IPS, SIEM);
- идентификация сотрудников и клиентов;
- федеративные системы;
- контроль поведения пользователей с помощью UEBA (UBA).
- Тенденции развития систем предотвращения утечки данных (DLP):
- контроль облаков и мобильных устройств;
- акцент на compliance (GDPR);
- автоматизация и простота управления;
- дополнительный мониторинг сотрудников, улучшенный workflow и управление инцидентами;
- интеграция с другими решениями по ИБ (средства маскирования данных, шифрование файлов, файрволлы для баз данных, шифрование баз данных).

В области информационной безопасности мобильных устройств прослеживаются следующие тенденции:

- интеграция с инфраструктурой ИБ;
- обмен информации об угрозах (Threat Intelligence);
- поддержка SaaS-модели;
- уклон в Detection & Response (EDR / STAP);
- тенденции хостовой ИБ.

Тенденции развития безопасности облачных технологий:

- активное внедрение CASB (Cloud Access Security Broker);
- смешанное управление (облачное + on-premise);
- поддержка в модели SaaS;
- переход к аутсорсингу / облачной ИБ по причине нехватки персонала;
- переход от модели MSSP к MDR (позволяет передать функцию обеспечения информационной безопасности аутсорсинговой компании).

В настоящее время явно заметна всё более возрастающая роль облачных технологий. Важно отметить, что при их использовании объект защиты находится за пределами корпоративной или ведомственной сети. Это приводит к повышению актуальности модели безопасности «нулевого доверия» (Zero Trust) [8] – полного отсутствия доверия кому-либо – даже пользователям внутри периметра. Модель подразумевает, что каждый пользователь или устройство должны

подтверждать свои данные каждый раз, когда они запрашивают доступ к какому-либо ресурсу внутри или за пределами сети.

Так же в настоящее время возрастает важность аналитики, которая становится неотъемлемой функцией продуктов информационной безопасности. Аналитика позволяет описать ситуацию (обнаружить атаку и её тип), выявить причину (какая часть инфраструктуры позволила реализовать атаку), предсказать последствия атаки и определить меры по реагированию.

Заключение

Развитие цифровых технологий и их активное внедрение в бизнес-процессы организаций на всех уровнях порождает широкий спектр угроз и требует развития средств защиты. Среди основных тенденций развития рынка информационных технологий можно отметить широкое использование мобильных устройств, облачных вычислений и технологий искусственного интеллекта. Внедрение данных технологий должно быть неразрывно связано с интеграцией соответствующих средств обеспечения информационной безопасности.

Список литературы

1. 10 крупнейших кибератак в первом квартале 2021 [Электронный ресурс]. – Режим доступа: <https://10guards.com/ru/uncategorized/10-major-cyber-attacks-in-q1-2021/>
2. Семяко, Г. В. Информационная безопасность в финансовом секторе: киберпреступность и стратегия противодействия / Г. В. Семяко // Социальные новации и социальные науки. – 2020. – №1 (1).
3. Салкуцан, А. А. Анализ влияния компьютерных атак на банковскую систему Российской Федерации / А. А. Салкуцан // Вестник РЭА им. Г. В. Плеханова. – 2019. – №6 (108).
4. Левина, А. Ю. Защита информации в автоматизированных банковских системах / А. Ю. Левина // Хроноэкономика. – 2019. – №1 (14).
5. Нестерова, Д. А. Риски информационной безопасности коммерческих банков в условиях новой экономической и технологической реальности / Д. А. Нестерова // Инновации и инвестиции. – 2020. – №5.
6. Борисова, Е. С. Инновации как инструмент обеспечения информационной безопасности и повышения эффективности деятельности банковской системы / Е. С. Борисова, А. Л. Белоусов // Russian Journal of Economics and Law. – 2019. – №3.

7. Ванин, М. В. Что такое Identity & Access Management и как его выбрать / М. В. Ванин // Информационная безопасность. – 2020. – № 4. – С. 10-11.

8. Rose, S. Zero Trust Architecture [Электронный ресурс]. – Режим доступа: <http://zhumal.mipt.rssi.ru>

Искусственный интеллект в кибербезопасности

Жехангер Али Кадхем Жехангер

Студент магистратуры

А. Ю. Иванков

Доцент

Введение

В настоящее время широкое распространение получили технологии машинного обучения. Это связано как с развитием аппаратных мощностей, так и с развитием математического обеспечения, в частности, аппарата искусственных нейронных сетей. Разработанные за последние несколько лет алгоритмы позволили значительно повысить точность решения задач классификации и предсказания, и эти результаты активно используются в различных областях кибербезопасности.

В данной работе проведен анализ существующих задач в области кибербезопасности и рассмотрены алгоритмы машинного обучения, применяемые для решения этих задач. Так же проведено сравнение классических и современных методов машинного обучения на примере задачи распознавания спама в сообщениях электронной почты.

1. Информационная безопасность и кибербезопасность

В настоящее время активное распространение получил термин «кибербезопасность». Среди нормативных документов, регулирующих данную область в России, можно выделить Концепцию стратегии кибербезопасности Российской Федерации [1], а также международный стандарт ISO/IEC 27032. Следует определить разницу между понятиями «информационная безопасность» (ИБ) и «кибербезопасность».

Кибербезопасность, согласно [1], подразумевает обеспечение совокупности условий, при которых все составляющие киберпространства [1] защищены от максимально возможного числа угроз и воздействий с нежелательными последствиями, в то время как информационная безопасность – состояние защищенности личности, организации и государства и их интересов от угроз, деструктивных и иных негативных воздействий в информационном пространстве. Таким

образом, кибербезопасность является одной из составляющих информационной безопасности.

Информационная безопасность статична. Есть соответствующие документы регуляторов, которые определяют требования к ее организации, составу системы управления, необходимости реализации тех или иных мер. Основными регуляторами в этой области являются Федеральная служба по техническому и экспортному контролю (ФСТЭК) и Федеральная служба безопасности (ФСБ) России.

Развитие цифровых технологий и их активное внедрение в бизнес-процессы организаций на всех уровнях порождает широкий спектр угроз и требует развития средств защиты. Среди основных тенденций развития рынка информационных технологий можно отметить широкое использование мобильных устройств, облачных вычислений и технологий искусственного интеллекта. Внедрение данных технологий должно быть неразрывно связано с интеграцией соответствующих средств обеспечения информационной безопасности.

2. Технологии искусственного интеллекта в задачах кибербезопасности

Искусственный интеллект (ИИ) помогает решать задачи машинного обучения (МО). Интеллектуальным называется такой алгоритм, который в ходе накопления данных (обучения) улучшает своё поведение (точность). Реализация интеллектуальных алгоритмов основана на методах машинного обучения [2].

МО – класс методов ИИ, характерной чертой которых является не прямое решение задач, а обучение за счет применения решений множества сходных задач.

Основная задача машинного обучения – разработка алгоритма, который работает с данными для принятия каких-либо решений. Основное свойство интеллектуальных систем – выполнять творческие функции, которые традиционно считаются прерогативой человека.

Ключевое отличие разработки моделей машинного обучения и систем принятия решений от традиционной разработки программного обеспечения в том, что человек не закладывает определенную логику и не реализует ее в программе принятия решений. Вместо этого используются алгоритмы, которые на основании данных строят решающие правила. То есть после подачи на вход в систему новых данных уже сами алгоритмы выносят вердикты на основании виденных ими ранее примеров, использованных для обучения. Человек не влияет на принятие решения. Он влияет только на подбор примеров, на которых алгоритм обучается, а также алгоритмов, которые должны будут решать задачи.

Выделяют четыре класса задач машинного обучения.

1 Обучение с учителем, в котором можно выделяются следующие задачи:

- задача классификации – определение категории объекта на основе описания категорий (классов) и их основных свойств;
- задача регрессии – предсказание числового значения, числовая оценка вещественной функции на основе значений аргумента (заданного вектора признаков).

2 Обучение без учителя:

- кластеризация – разделение объектов массива данных на кластеры с выявлением особенностей каждого, чтобы вести более точечную работу с ними, т.е. группирование по схожести признаков;
- поиск аномалий в данных: сначала строится модель нормальности данных, затем выявляются отклонения и проводится дополнительный анализ.

3 Обучение с частичным привлечением учителя: под имеющийся большой массив данных есть правила разметки только для части данных. В этом случае применяются специальные алгоритмы, которые пытаются разметить остальные данные.

4 Обучение с подкреплением: машина с ИИ автоматически учится в среде взаимодействия, соблюдая определенные правила.

В кибербезопасности чаще всего возникают задачи обучения с учителем и в меньшей степени задачи кластеризации и поиска аномалий. При решении задач в машинном обучении необходимо определить класс задачи и категорию данных. Исходя из этого выбираются алгоритмы и подходы.

Выделяют 2 категории данных:

- структурированные: можно естественным образом представить в виде таблицы в базе данных, каждый элемент данных обладает общей структурой (например, транзакции клиентов, сетевой трафик);
- неструктурированные: каждый элемент данных уникален (например, видео, изображения, текст, запись голоса).

Ключевые области применения ИИ в кибербезопасности:

- противодействие мошенничеству (антифрод): оценка риска транзакций (насколько операция может быть мошеннической, а клиент - мошенником);
- расследования: по уже известным или предотвращенным случаям мошенничества выявляются дополнительные связи мошенников;

- киберзащита: выявление аномалий, заражений хостов, DDos-атак, их предотвращение, выявление фишинговых доменов;
- контроль конфиденциальной информации: нельзя допустить, чтобы информация, содержащая банковскую или коммерческую тайну, переместилась за периметр обработки либо попала в домены, где ее не должно быть.

Для противодействия социальной инженерии используются схожие модели ИИ, что и для противодействия мошенничеству в целом, но они изучают другие признаки операций.

Зрелость open source библиотек машинного обучения позволяет использовать готовый продукт для решения сложных практических задач. Текущие data science-технологии построены на open source-решениях. Они эффективны для разработки моделей машинного обучения. Рынок open source-решений активно развивается благодаря тому, что промышленные корпорации публикуют внутренние инструменты, а сообщество разработчиков совершенствует их.

3. Пример задачи кибербезопасности по распознаванию спама

Одной из задачи кибербезопасности является распознавание спама в сообщениях электронной почты. Спам – это массовая рассылка рекламного или иного характера, распространяемая без согласия получателя. Спам может принести реальный ущерб, связанный с хищением личных данных, например, путем распространения ссылок на фишинговые сайты [3].

Постановка задачи классификации заключается в том, чтобы для каждого нового сообщения электронной почты принять решение, относящее его либо к классу «спам», либо к классу полезных сообщений.

В рамках данной задачи было проведено сравнение нескольких подходов – алгоритмов неглубокого и глубокого машинного обучения [4]. Из неглубоких алгоритмов были использованы наивный байесовский классификатор, метод опорных векторов с линейным ядром и случайный лес с количеством деревьев равным 100, из алгоритмов глубокого обучения – рекуррентная нейронная сеть.

Прежде, чем переходить к работе с алгоритмами классификации текста, документ необходимо представить очищенным набором слов. Для этого выполняется процедура индексации, состоящая из трех этапов [5].

1. Токенизация:
 - приведение текста к нижнему регистру;
 - замена всех знаков препинания на пробелы;
 - разделение строки на отдельные слова по пробелам.

2. Очистка данных:
 - анализ данных;
 - определение порядка и правил преобразования данных;
 - преобразование данных.
3. Нормализация:
 - стемминг;
 - лемматизация.

Лемматизация исходных текстов проводилась с помощью обучающего набора эмбедингов библиотеки Word2Vec [6]. Все алгоритмы были также протестированы на взятом из открытого доступа датасете англоязычных писем [7]. Полученные результаты показаны в сводной таблице 1.

Таблица 1

Точность алгоритмов классификации спам сообщений

Классификатор	Точность
Наивный байесовский классификатор	0.91
Метод опорных векторов	0.93
Случайный лес	0.95
Рекуррентная нейронная сеть	0.97

Среди классических алгоритмов лучшие показатели у метода опорных векторов и случайного леса. Нейронная сеть показала наибольшую точность среди всех испытанных алгоритмов.

Заключение

Развитие цифровых технологий и их активное внедрение в бизнес-процессы организаций на всех уровнях порождает широкий спектр угроз и требует развития средств защиты. Использование технологий машинного обучения в задачах кибербезопасности является перспективным направлением развития сферы информационной безопасности. Внедрение в системы кибербезопасности алгоритмов глубокого обучения позволяет повысить их эффективность, что показано на примере классификации спам-сообщений, где наибольшую точность показала глубокая рекуррентная нейронная сеть.

Список литературы

1. Концепция стратегии кибербезопасности Российской Федерации [Электронный ресурс]. – Режим доступа: <http://council.gov.ru/media/files/41d4b3dfbdb25cea8a73.pdf>

2. Сирота, А. А. Методы и алгоритмы анализа данных и их моделирование в MATLAB: учеб. пособие. – СПб.: БХВ-Петербург, 2016. – 384 с.
3. Спам и фишинг в 2021 году. [Электронный ресурс]. – Режим доступа: <https://securelist.ru/spam-and-phishing-in-2021/104407/>
4. Складенко, Н.С. Обзор алгоритмов машинного обучения, решающих задачу обнаружения спама / Н. С. Складенко // Новые информационные технологии в автоматизированных системах. – 2017. – №20.
5. Автоматическая обработка текстов на естественном языке и компьютерная лингвистика: учеб. пособие / Е. И. Большакова [и др.]. – М.: МИЭМ, 2011. – 272 с.
6. Word2vec Project [Электронный ресурс]. – Режим доступа: <https://code.google.com/archive/p/word2vec/>
7. Spam Mails Dataset [Электронный ресурс]. – Режим доступа: <https://www.kaggle.com/datasets/venky73/spam-mails-dataset>

Разработка программного обеспечения для анализа наполненности и целостности медицинских ампул

А. С. Зверева

Студент магистратуры

А. В. Швырева

Ассистент

Введение

Растворы для инъекций представляют собой стерильные водные или неводные растворы действующего вещества (веществ) в соответствующем растворителе, предназначенные для инъекционного введения в определенные ткани или органы или в сосудистое русло, что подразумевает нарушение целостности кожных покровов. В связи с этим возникает опасность внесения в организм инфекций или пирогенных веществ. Поэтому растворы должны быть свободными от механических включений и стерильными (невскрытыми). Для хранения растворов используются медицинские ампулы – это герметически запаянные сосуды, предназначенные для хранения химически чистых веществ или стерильных лекарственных препаратов, шовного материала.

Проверка качества изготовленной ампулы в ручном режиме требует много ресурсов. Также может влиять человеческий фактор – некоторая степень ошибки при принятии решения о качестве ампулы. Применение методик компьютерного зрения поможет увеличить количественные и качественные характеристики выпускаемых растворов.

1. Анализ целостности

Лекарственный препарат должен иметь упаковку, которая защищает раствор от воздействия факторов окружающей среды и обеспечивает целостность ампулы [1].

В рассматриваемом случае характеристикой целостности является скручивание колпачка ампулы. При слишком сильном скручивании носик колпачка вскрывает ампулу, что становится причиной непригодности использования содержащегося в ней раствора (рис. 2).

Определение размера ампулы позволит детектировать скручивание колпачка.



а

б

Рис. 1. Положение колпачка на ампуле: а – нормальное положение, б – скрученное положение

Детектор Кэнни

Детектор границ Кэнни предназначен для поиска границ на изображении [2]. Алгоритм можно представить следующим образом:

- подготовка изображения,
- поиск градиентов,
- применение операции замыкания,
- связывание краев в контур.

Подготовка изображения включает в себя сглаживание фильтром Гаусса и преобразование к градациям серого (рис. 2а).

Для поиска градиентов используется оператор Собеля, который в каждой точке изображения рассчитывает вектор яркости. Ядра фильтров представляют собой (1) и (2).

$$MG_x = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} \quad (1)$$

$$MG_y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & 2 & -1 \end{pmatrix} \quad (2)$$

После воздействия на изображение оператором Собеля производится операция замыкания (дилатация и эрозия).

Результатом работы оператора Кэнни является бинарное изображение с выделенными границами (рис. 2б).

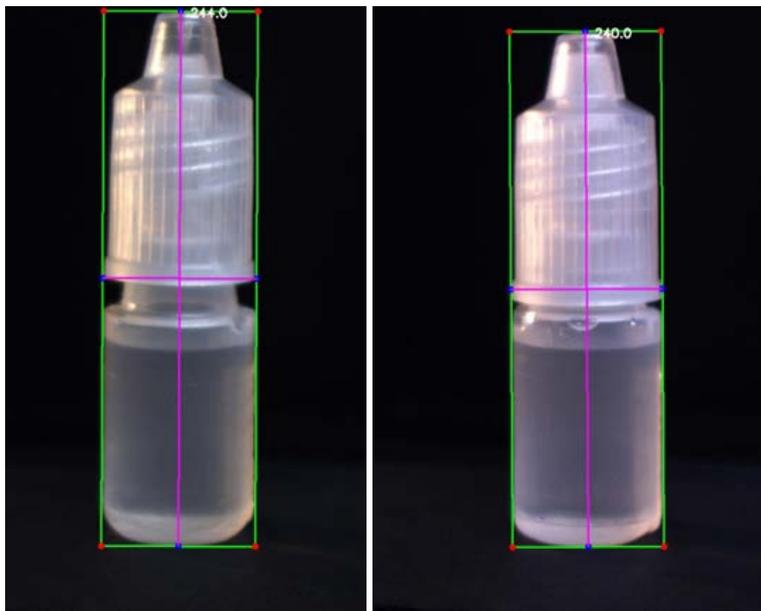


а

б

Рис. 2. Процесс обработки изображения: а – подготовленное изображение, б – изображение после воздействия оператора Кэнни

После определения границ производится расчет длины ампулы (рис. 3).



а

б

Рис. 3. Результат вычисления длины ампул: а – ампула нескрученная, б – ампула скрученная.

2. Анализ взвеси

Под взвесью понимаются механические включения (МВ), случайно присутствующие в лекарственных средствах. Для их детектирования была выбрана нейронная сеть YOLOv7. Принцип работы этой НС можно представить следующим образом [3]: изображение делится на квадратную сетку. Для каждой ячейки сети выводится вероятность определяемого класса. Ячейки, имеющие вероятность класса выше порогового значения, выбираются и используются для определения местоположения объекта на изображении.

Обучение YOLOv7

Имеющийся набор изображений был размечен с выделением двух классов – clear (чистый) и suspension (содержащий взвесь) и далее разделен на обучающую, валидационную и тестовую выборки.

Пусть (1) – есть набор входных обучающих данных, где X – необработанные входные данные, Y – соответствующая метка.

$$S = \{(X_i, y_i), i = 1..N\} \quad (3)$$

Обозначим рекурсивную функцию $Z^{(m)}$ для каждого слоя как (4)

$$Z^{(m)} = f(Q^{(m)}), Z^{(0)} = X, \quad (4)$$

где

$$Q^{(m)} = W^{(m)} * Z^{(m-1)}, \quad (5)$$

где $W^{(m)}$ – веса для обучения, $Z^{(m-1)}$ – карта признаков предыдущего слоя, $f(Q^{(m)})$ – пуллинговая функция.

Целевая функция принимает вид (6):

$$\|w^{(out)}\|^2 + L(W, w^{(out)}) + \sum_{m=1}^{M-1} \alpha_m \left[\|w^{(out)}\|^2 + l(W, w^{(out)}) - \gamma \right], \quad (6)$$

где $L(W, w^{(out)})$ – общие потери, $l(W, w^{(out)})$ – сопутствующие потери.

Результаты обучения представлены на рис. 3.

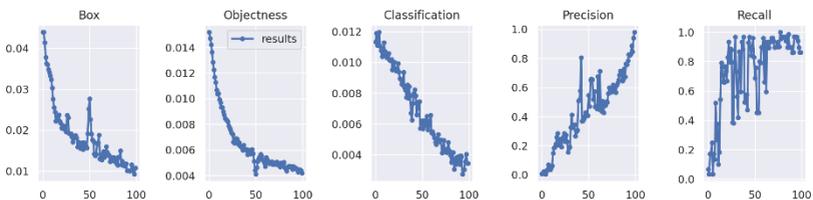


Рис. 4. Результаты обучения нейронной сети

Результаты тестирования представлены в таблице 1 и на рис. 7:

Таблица 1

Метрики тестирования нейронной сети

P	R	mAP@.5	mAP@.5:.95:
0.843	0.927	0.986	0.794



Рис. 5. Результат тестирования нейронной сети

Заключение

В результате выполнения данной работы был проведен анализ задачи определения длины ампулы на изображении и детектирования механических включений.

В качестве алгоритмов были рассмотрены детектор Кэнни и нейронная сеть YOLOv7. Также эти методы были реализованы программно, что позволило проверить их работу на наборе изображений..

Список литературы

1. ОФС.1.4.2.0005.15 Видимые механические включения в лекарственных формах для парентерального применения и глазных лекарственных формах [Текст] : – Введ. 01.12.2018.
2. Canny J. A Computational Approach to Edge Detection / J. Canny // TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE – 1986. – №6. – С. 679-697.
3. Chen-Yu L. Deeply-Supervised Nets / L. Chen-Yu [et.al.] // PMLR – 2015. – P. 1-9.

Разработка приложения для разметки данных на основе микросервисной архитектуры

Н. А. Зеленев

Студент магистратуры

А. А. Сирота

Профессор

Введение

Машинное обучение используется для решения широкого спектра задач таких, как классификация, обнаружение объектов, сегментация изображений и анализ естественного языка. Разметка данных всегда была одной из самых важных и первостепенных задач в создании моделей машинного обучения. Маркировка или разметка данных, является частью этапа предварительной обработки. Данный процесс требует точной идентификации исходных данных (например, изображений, текстовых файлов и видео), а затем добавления одной или нескольких меток к этим данным для определения их контекста, что позволяет модели машинного обучения решать поставленную задачу [1]. Маркировка данных лежит в основе различных вариантов использования машинного обучения и глубокого обучения, включая компьютерное зрение и обработку естественного языка. Компании объединяют программное обеспечение, процессы и людей для фильтрации, структурирования и разметки данных. Эти обучающие данные становятся основой для последующих этапов разработки. Метки позволяют аналитикам выделить переменные в наборе данных, что, в свою очередь, позволяет выбрать оптимальные предикторы данных для моделей машинного обучения [2]. Метки определяют соответствующие векторы для обучения модели, где алгоритм учится делать наилучшие прогнозы. Однако, аннотирование большого объема данных требует значительных человеческих и денежных затрат [1]. Для построения надежных моделей и повышения точности прогнозирования требуется большое количество аннотированных данных. Объемы обучающий данных ежедневно увеличиваются, требуя все больше и больше человеко-часов и вычислительных мощностей. Поэтому, специалисты в

данной области уделяют особое внимание автоматизации этапа разработки моделей машинного обучения [3].

1. Основные подходы к решению задачи разметки данных

Разметка данных – важнейший шаг в разработке высокопроизводительной модели машинного обучения. Хотя данный этап кажется простым, его не всегда легко реализовать [4]. В результате разработчикам приходится учитывать множество факторов и методов, чтобы определить наилучший подход к разметке. Поскольку каждый метод имеет свои плюсы и минусы, рекомендуется детально оценить сложность задачи, а также размер, объем и продолжительность проекта. Рассмотрим основные подходы к маркировке.

Внутренняя маркировка – использование штатных специалистов по анализу данных упрощает отслеживание этапов разработки, обеспечивает большую точность и повышает качество. Однако такой подход обычно требует больше времени и предпочтителен для крупных компаний, которые готовы выделять ресурсы для решения задачи.

Синтетическая маркировка – этот подход позволяет генерировать новые данные для проекта из уже существующих наборов данных, что повышает качество данных и экономит время. Однако синтетическая маркировка требует значительных вычислительных мощностей, что может привести к росту стоимости проекта.

Автоматизированная маркировка – этот процесс разметки данных использует скрипты для сокращения временных затрат и необходимости аннотирования человеком. Однако возможность возникновения технических проблем требует, чтобы участие человека оставалось неотъемлемой частью процесса обеспечения качества.

Использование аутсорсинга может быть оптимальным выбором для большинства современных проектов, но разработка и управление рабочими процессами, ориентированным на фрилансеров, также может отнимать много ресурсов. Хотя платформы для фрилансеров предоставляют полную информацию о кандидатах для облегчения процесса проверки, наем управляемых команд по разметке данных обеспечивает предварительно проверенный персонал и готовые инструменты.

Краудсорсинг – данный подход является более быстрым и экономически эффективным благодаря возможности выполнения небольших задач. Однако экспертиза специалистов, контроль качества и управление проектом варьируются в зависимости от платформы краудсорсинга. Одним из самых известных примеров краудсорсинговой маркировки данных является Recaptcha. Этот проект был двойным, поскольку он контролировал ботов и одновременно улучшал аннотацию

данных на изображениях. Например, запрос Recaptcha просил пользователя определить все фотографии, содержащие автомобиль, чтобы доказать, что он человек, а затем эта программа могла проверить себя, основываясь на результатах других пользователей. Данные, полученные от этих пользователей, обеспечивают базу данных меток для массива изображений.

Основной компромисс при разметке данных заключается в том, что, хотя она может сократить время масштабирования бизнеса, это, как правило, обходится недешево [5]. Более точные данные обычно улучшают прогнозы моделей, поэтому, несмотря на высокую стоимость, ценность, которую она обеспечивает, обычно оправдывает вложения. Поскольку аннотирование данных обеспечивает больший контекст наборов данных, оно повышает производительность анализа данных, а также приложений на основе машинного обучения и искусственного интеллекта. Например, маркировка данных позволяет получить более релевантные результаты поиска на платформах поисковых систем и лучшие рекомендации по товарам на платформах электронной коммерции.

2. Проблемы разметки данных

Этап разметки данных не обходится без трудностей. В частности, к наиболее распространенным проблемам относятся стоимость и человеческий фактор [1].

Хотя аннотирование данных имеет решающее значение для моделей машинного обучения, данный этап может быть дорогостоящим как с точки зрения ресурсов, так и времени [5]. Если компания использует более автоматизированный подход, инженерным группам все равно придется создавать конвейеры данных перед их обработкой, а ручная маркировка почти всегда будет дорогой и трудоемкой.

Помимо высокой стоимости подходы к разметке данных также подвержены человеческим ошибкам (например, ошибки кодирования или ошибки ручного ввода), что может снизить качество данных. Это, в свою очередь, приводит к неточной обработке данных и моделированию. Проверки обеспечения качества необходимы для поддержания качества данных.

3. Современные методы разметки данных

Независимо от подхода, существуют передовые методы, оптимизирующие точность и эффективность маркировки данных. Первым и самым базовым шагом в сторону упрощения этапа разметки данных являются интуитивно понятные интерфейсы, которые

минимизируют когнитивную нагрузку и переключение контекста для людей, занимающихся аннотированием [6].

Подходы на основе консенсуса измеряют степень согласия между несколькими маркировщиками (человеческими или машинными). Оценка консенсуса рассчитывается путем деления суммы согласованных меток на общее количество меток на единицу данных.

Дополнительным шагом может являться аудит меток, который позволяет проверить точность меток и их обновление их по мере необходимости. К данному классу алгоритмов относятся, например, синтез запроса на членство, который генерирует синтетический экземпляр и запрашивает для него метку. Выборка на основе пула – ранжирование всех неразмеченных экземпляров в соответствии с измерением информативности и выбор лучших запросов для аннотирования. Селективная выборка на основе потока – выборка немаркированных экземпляров по одному и их маркировка или игнорирование в зависимости от их информативности или неопределенности [4].

Трансферное обучение берет одну или несколько предварительно обученных моделей из одного набора данных и применяет их к другому. Это может включать многозадачное обучение, при котором несколько задач изучаются в тандеме.

Активное обучение – это отдельная категория алгоритмов машинного обучения, которые помогают человеку определить наиболее подходящие наборы данных. К данному классу алгоритмов относятся, например, синтез запроса на членство, который генерирует синтетический экземпляр и запрашивает для него метку. Выборка на основе пула – ранжирование всех неразмеченных экземпляров в соответствии с измерением информативности и выбор лучших запросов для аннотирования. Селективная выборка на основе потока – выборка немаркированных экземпляров по одному и их маркировка или игнорирование в зависимости от их информативности или неопределенности.

4. Области применения

Хотя разметка данных может повысить точность, качество и удобство использования в различных задачах и отраслях, можно выделить отдельные области исследований, где данный подход применяется чаще всего [4].

Например, компьютерное зрение – область искусственного интеллекта, использующая обучающие данные для построения модели компьютерного зрения, которая позволяет сегментировать изображения и автоматизировать категоризацию, идентифицировать ключевые точки

на изображении и определять местоположение объектов. Компьютерное зрение используется во многих отраслях промышленности - от энергетики и коммунального хозяйства до производства и автомобилестроения.

Обработка естественного языка (НЛП) – отрасль искусственного интеллекта, объединяющая вычислительную лингвистику со статистическими моделями, моделями машинного обучения и глубокого обучения для определения и маркировки важных фрагментов текста, генерирующих обучающие данные для анализа настроений, распознавания имен сущностей и оптического распознавания символов. НЛП все чаще используется в таких корпоративных решениях, как обнаружение спама, машинный перевод, распознавание речи, резюмирование текста, виртуальные помощники и чат-боты, а также системы GPS с голосовым управлением. Это сделало НЛП важнейшим компонентом в эволюции критически важных бизнес-процессов.

5. Обзор микросервисной архитектуры

Микросервисная архитектура – новый тренд в организации компонентов программных систем, позволяющий разбить архитектуру приложения на независимые компоненты, которые могут быть быстро развернуты на любом инфраструктурном ресурсе. Микросервисы запускаются независимо друг от друга и обычно поддерживаются механизмом развертывания и оркестрации, например, в облаке. Исследования микросервисов касаются как архитектурных принципов и поддержки, так и применения архитектурных шаблонов. В облаке микросервисы связаны с контейнерами, которые представляют собой легкий механизм виртуализации, используемый для сборки, распределения и управления приложениями. Исследования микросервисов касаются как архитектурных принципов и поддержки, так и применения архитектурных шаблонов. В облаке микросервисы связаны с контейнерами, которые представляют собой легкий механизм виртуализации, используемый для сборки, распределения и управления приложениями.

Микросервисная архитектура – это подход к функциональной декомпозиции компонентов системы в контексте проектирования, ориентированного на домен (область знаний). Компоненты характеризуются четко определенными и явно опубликованными внутренними интерфейсами. Каждый сервис является полностью автономным. Следовательно, изменение реализации сервиса не влияет на другие сервисы, так как взаимодействие происходит только через интерфейсы. Функциональная декомпозиция приложения является ключом к построению успешной архитектуры микросервисов. При этом

достигается слабая связность (например, RESTful-интерфейсы) и высокая согласованность (несколько сервисов могут компоноваться друг с другом для определения сервисов более высокого уровня или приложения).

Микросервисы — это архитектурный паттерн, зародившийся на основе сервис-ориентированной архитектуры (SOA), в котором особое внимание уделяется автономности и легкости. Хотя это архитектурный принцип, параллельно в облачных вычислениях появилась технология контейнеризации, обеспечивающая простой механизм виртуализации, который может служить механизмом упаковки приложений для платформ облачных вычислений.

6. Основные компоненты предлагаемой системы

Применение микросервисной архитектуры позволяет декомпозировать систему на функциональные блоки, которые могут разрабатываться независимо, с использованием различных технологий и быть взаимозаменяемыми. Основные компоненты системы представлены на рис. 1.

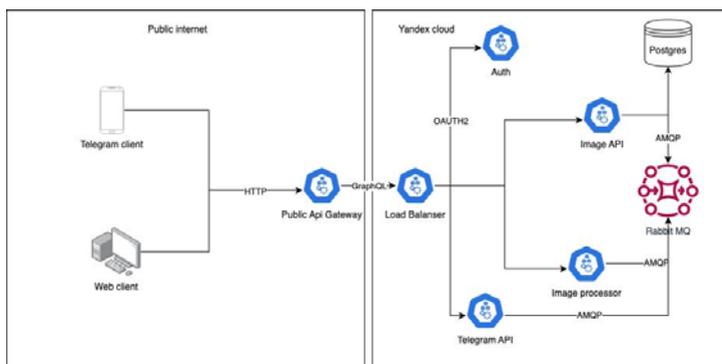


Рис. 1. Целевая архитектура системы

Центральной решаемой задачей остается разметка данных, поэтому ядром системы является микросервис, решающий задачу аннотирования данных. Благодаря тому, что данный компонент может быть легко заменен, можно использовать различные автоматизированные подходы, что позволяет дорабатывать систему под конкретную решаемую задачу, добиваясь максимальной точности. Также, данный архитектурный стиль позволяет добиться независимости от природы данных (изображения, текст, видео, аудио). Неизменным остается одно единственное требование – поддержание контракта, определенного интерфейсом

конкретного микросервиса. Отдельной частью остается контроль качества автоматически размеченных данных. Для этого необходимо разграничивать права различных групп пользователей к ресурсам системы. Данную задачу решает микросервис аутентификации. Аутентификация – это процесс проверки личности пользователей или других субъектов в рамках системы контроля доступа. Приложение микросервисов имеет множество независимых сервисов, каждый из которых выполняет определенную заданную функцию. Его функциональность позволит отделить пользователей системы от группы модераторов, которая будет ответственной за верификацию и редактирование результатов разметки, путем изменения параметров меток.

Взаимодействие между микросервисами, входящими в состав системы, будет реализовано асинхронно с использованием брокеров сообщений. Брокер сообщений — это программное обеспечение, которое позволяет приложениям, системам и службам взаимодействовать друг с другом и обмениваться информацией. Брокер сообщений делает это путем перевода сообщений между формальными протоколами обмена сообщениями. Это позволяет взаимозависимым сервисам обмениваться данными друг с другом напрямую, даже если они написаны на разных языках или реализованы на разных платформах. Асинхронный обмен сообщениями относится к типу межприкладной коммуникации, которую делают возможной брокеры сообщений. Он предотвращает потерю ценных данных и позволяет системам продолжать функционировать даже в условиях прерывистого соединения или задержек, характерных для сетей общего пользования. Асинхронный обмен сообщениями гарантирует, что сообщения будут доставлены один раз (и только один раз) в правильном порядке относительно других сообщений.

Неотъемлемой частью любой информационной системы является хранение обрабатываемых данных. Для решения этой задачи используется реляционная СУБД. PostgreSQL — это мощная объектно-реляционная система баз данных с открытым исходным кодом, которая активно разрабатывается уже более 35 лет и заслужила репутацию надежной, функционально устойчивой и производительной системы. Данный элемент позволит хранить любой тип информации, поступающей на вход системы, например, текст, изображения и видео.

Заключение

Разметка данных является важной задачей машинного обучения. Оптимизированное аннотирование данных, включающее

автоматические и полуавтоматические подходы, повышает эффективность аннотирования данных и снижает стоимость задач машинного обучения. Использование микросервисной архитектуры позволит создать масштабируемое решение, которое можно будет развивать для любой прикладной задачи или интегрировать в уже существующие системы на основе моделей машинного обучения. Стоит отметить, что интеграция данного решения сводится к решению относительно несложной инженерной задачи – реализации интерфейса, позволяющего готовым компонентам системы общаться с новой системой, решающей задачу разметки данных. Автоматизация данного фундаментального этапа разработки моделей машинного обучения позволит снизить денежные и, что не менее важно, временные затраты. Исследования в данной области позволят значительно увеличить качество готовых решений, расширить области применения машинного обучения и искусственного интеллекта, и сделать данные технологии более доступными.

Список литературы

1. Боим, Р. Задавая правильные вопросы при поиске данных / Р. Боим, О. Гриншам // 28-я Международная конференция по инженерии данных (ICDE). – Брюссель, 2012. – С. 1261–1264.
2. Брю, Э. Использование краудсорсинга и активного обучения для отслеживания настроений в онлайн-СМИ / Э. Брю, Д. Грин // 19-я Европейской конференции по искусственному интеллекту. – Стокгольм, 2018. – С. 145-150.
3. Дэвид, П. Оценка методом максимального правдоподобия коэффициентов ошибок. Прикладная статистика / П. Дэвид, А. М. Скин. – 1979. – № 1. – С. 20-28.
4. Демартини, Г. Использование вероятностных рассуждений и методов краудсорсинга для крупномасштабного связывания сущностей / Г. Демартини, Д. Эддин // 21-я международная конференция «World Wide Web». – Лион, 2012. – С. 469-478.
5. Фэн, Ю. Адаптивная система краудсорсинга / Ю. Фэнг, Г. Ли, Б. Оой // Международная конференция ACM SIGMOD по управлению данными. – Мельбурн, 2015. – С. 1015-1030.
6. Фан, М. Самообучающееся активное обучение из толпы / М. Фан, С. Жу, Б. Ли, В. Динг. // 12-я международная конференция по сбору данных (ICDM). – Брюссель, 2012. – С. 858-863.

Прослеживание границ объектов бинарных изображений.

К. Д. Иванов

Студент магистратуры

В. В. Фертиков

Доцент

Введение

Обработка изображений является одной из самых актуальных областей компьютерной науки и находит применение во многих отраслях, от медицинской диагностики до промышленности и науки о материалах. С развитием компьютерных технологий и алгоритмов, возможности обработки изображений значительно расширились, что позволяет автоматизировать рутинные процессы и ускорить выполнение задач. Процесс прослеживания границ объектов на непрерывных бинарных изображениях является одним из ключевых этапов при анализе изображений. Он позволяет определить контуры объектов на изображении и выделить их от фона, что является необходимым для дальнейшей обработки и распознавания объектов на изображении. В этой работе мы рассмотрим и разработаем модуль для прослеживания границ объектов на непрерывных бинарных изображениях и протестируем разработанный модуль.

1. Понятие фигуры на изображении

Фигура – это связанная замкнутая область на плоскости, ограниченная конечным числом непересекающихся жордановых кривых [1].

Иначе, можно сказать, что под Фигурой будем понимать – связанную часть плоскости, ограниченную одной или несколькими замкнутыми линиями. Таким образом, фигура, это сплошной объект на плоскости с конечным числом дыр.

В рамках этого определения, не любая замкнутая область является фигурой. Одним из самых важных моментов, является отсутствие самопересечений у кривых, определяющих форму объекта.

На рис. 1 изображен квадрат с треугольным отверстием, а на рис. 2 квадрат и треугольник имеют общие точки, что означает, что эти объекты имеют пересечения.

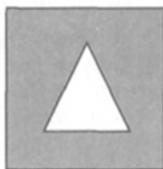


Рис. 1. Фигура квадрата с треугольным отверстием

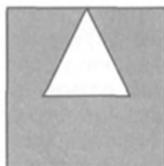


Рис. 2. Изображение, на котором нельзя выделить фигуру по определению

2. Граничное представление фигуры

Точка фигуры является граничной, если в ее окрестности находится любое количество точек, не принадлежащих фигуре. Границей фигуры будем называть множество всех граничных точек фигуры. Граница фигуры может быть односвязной – если ее граница состоит из одного контура и многосвязной – если граница фигуры состоит из нескольких несвязанных контуров. В этом случае один контур описывает внешнюю границу фигуры, а все остальные находятся внутри него.

3. Граничные коридоры дискретной сцены

Для построения границ фигур изображения необходимо выявить все граничные пары точек в изображении. Граничная пара – это пара разноцветных соседних точек. Будем использовать треугольную структуру соседства.

Тогда граничные пары и граничные треугольники имеют следующие свойства:

1. Каждая граничная пара является общей стороной пары граничных треугольников;
2. Каждый граничный треугольник имеет две стороны, состоящие из граничных пар.

Граничным коридором будем называть объединенную цепочку из граничных треугольников. У граничного коридора будет две границы – внешняя и внутренняя, причем одна граница будет состоять из точек первого цвета, а другая из точек второго цвета.

Задачу нахождения всех граничных коридоров будем называть прослеживанием или трассировкой границы дискретной сцены.

4. Поиск граничных коридоров.

Задача поиска состоит в том, чтобы найти хотя бы одну граничную пару, входящую в граничный коридор. В любом коридоре будут находиться точки, которые находятся вертикально и горизонтально. Поэтому можно производить поиск только по одному направлению, например горизонтально. Поиск горизонтальных граничных пар будет происходить посредством построчного сканирования бинарного изображения. Построчное сканирование – это циклический проход по изображению сверху вниз и слева направо. Пример циклического прохода по изображению приведен на рисунке 3.

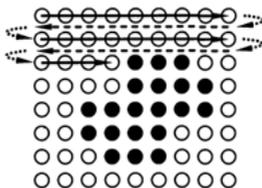


Рис. 3. Построчное сканирование изображения

Все точки, находящиеся вне изображения, являются белыми. Граничная пара будет найдена, если в строке встретится пара точек, имеющая разные цвета. Как только граничная пара найдена, можно начинать трассировку граничного коридора. При окончании прослеживания граничного коридора, необходимо продолжить поиск граничных пар с того же самого места. При полном построчном сканировании, будут найдены граничные пары, входящие в уже прослеженные граничные коридоры. Для того, чтобы не прослеживать одни и те же коридоры, создадим матрицу такого же размера, как и исходное изображения, в которой будем отмечать все точки, входящие в найденные граничные коридоры. Таким образом, граничный коридор считается обнаруженным, если при построчном сканировании найдена граничная пара и хотя бы одна точка из граничной пары не отмечена в дополнительной матрице.

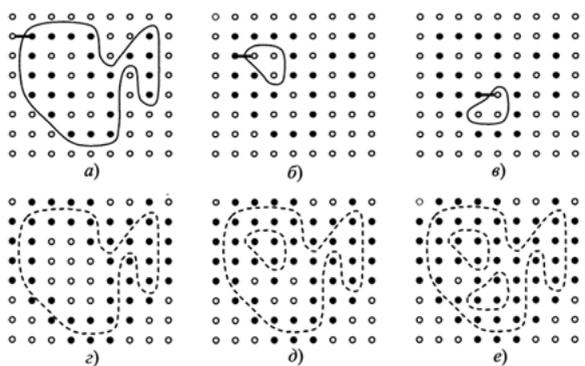


Рис. 4. Пример поиска граничных коридоров с занесением отметок в дополнительную матрицу

Отметим, что порядок точек в найденной граничной паре позволяет определить внешнему или внутреннему коридору она принадлежит.

5. Симплексное прослеживание граничного коридора

Симплексное прослеживание граничного коридора можно представить как движение человека с завязанными глазами вдоль границы лужи. Человеку необходимо одну ногу всегда держать в луже, а вторую на берегу, чтобы не потерять лужу и не зайти обеими ногами вглубь [2]. Иллюстрация этого подхода приведена на рис. 5.

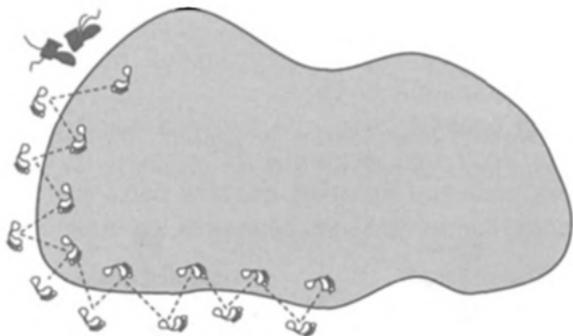


Рис. 5. Прослеживание пешеходом границы лужи

Алгоритм прослеживания состоит из следующих шагов:

1. Построение начального треугольника по граничной паре;

2. Основная часть: последовательные построения новых треугольников;
3. Проверка окончания прослеживания. Если новый треугольник совпадает с начальным треугольником, то конец цикла, иначе переход к шагу 2.

6. Построение начального треугольника по граничной паре

После нахождения разноцветной горизонтальной граничной пары необходимо построить треугольник, с которого начнется прослеживание граничного коридора. Будем строить граничный коридор, у которого фигура, состоящая из черных точек, находилась слева от коридора, а фон справа. В связи с этим введем обозначения: L – левая черная точка, R – правая белая точка. Третью вершину назовем T и будем выбирать ее с учетом того, что вершины треугольника должны составлять правую тройку. Треугольник должен быть прямоугольным, в соответствии со всеми правилами, угол при вершине R должен быть прямым, а вершина T должна быть выше или ниже вершины R в соответствии с рисунком 6.



Рис. 6. Построение начального треугольника

Таким образом, координаты третьей точки находятся по следующим формулам:

$$T.x=R.x, T.y=R.y+(R.x-L.x), . \quad (1)$$

7. Последовательное построение новых треугольников

При построении нового треугольника, производится поворот предыдущего треугольника через одну из его сторон. Построение нового треугольника зависит от цвета точки T.

Опишем два правила по которым производится поворот треугольника:

1. Новый и старые треугольники должны иметь общую сторону LT или RT, выбор её производится с условием, что концевые точки разноцветные.
2. Старый и новый треугольники центрально симметричны относительно их общей стороны.

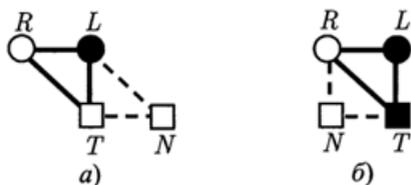


Рис. 7. Варианты переворотов треугольника

Таким образом, новый треугольник отличается от старого только одной вершиной. Приведем формулы получения координат новой точки.

$$N_m = \begin{cases} T_m + (L_m - R_m), & \text{если } T_m \text{ черная,} \\ T_m + (R_m - L_m), & \text{если } T_m \text{ белая.} \end{cases} \quad (2)$$

В конце необходимо обозначить вершины нового треугольника на основе вершин старого треугольника и новой точки.

$$\begin{aligned} L_{m+1} &= \begin{cases} L_m, & \text{если } T_m \text{ белая,} \\ T_m, & \text{если } T_m \text{ черная;} \end{cases} \\ R_{m+1} &= \begin{cases} T_m, & \text{если } T_m \text{ белая,} \\ R_m, & \text{если } T_m \text{ черная;} \end{cases} \\ T_{m+1} &= N_m \end{aligned} \quad (3)$$

8. Проверка окончания прослеживания

Процесс трассировки граничного коридора заканчивается, когда новый треугольник совпадает с треугольником, с которого началось прослеживание. Отметим, что в ходе прослеживания мы получили граничные коридоры в формате структуры данных, в котором хранились точки граничных коридоров, также мы получили все точки, входящие в граничные коридоры на дополнительной матрице, на которой отмечались все задействованные точки при прослеживании граничных коридоров.

9. Пример работы алгоритма прослеживания границы

В ходе работы был разработан алгоритм прослеживания границ бинарного изображения на языке python.

На вход алгоритма поступает бинарное изображение. Изначально трехцветное изображение необходимо преобразовать к одноцветному изображению и выбрать пороговое значение, ниже которого все пиксели будут белыми, а выше черными. Итогом работы алгоритма будет список граничных коридоров, находящихся на изображении, граничный коридор является списком треугольников, полученных в ходе симплексного прослеживания границ.

На рис. 8 и 9 приведены примеры работы алгоритма.

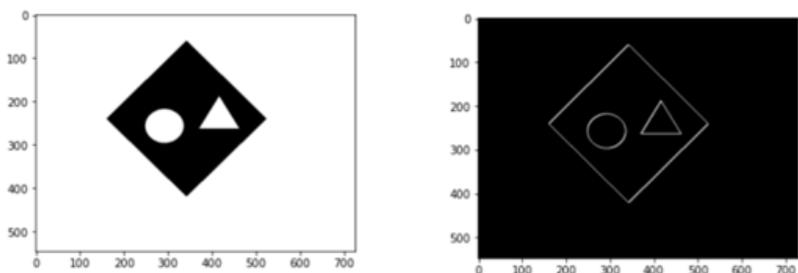


Рис. 8. Найденные границы простой фигуры с отверстиями

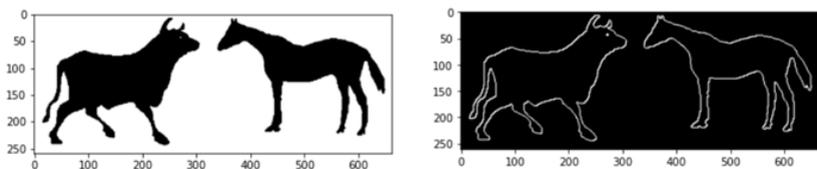


Рис. 9. Найденные границы сложной фигуры с отверстиями

На данных рисунках видно, что были найдены как внешние границы фигур, так и границы отверстий в фигурах. Также, видно, что границы простых объектов, представляющие отрезки были прослежены без искажений, а границы сложных объектов, с постоянными поворотами, так же были найдены без искажений.

Заключение

Данная работа посвящена разработке программного модуля для прослеживания границы фигуры бинарного изображения. Описано, понятия фигуры, граничных пар и граничных коридоров. Разработан и протестирован алгоритм прослеживания граничных коридоров. Использование данного решения позволят проводить анализ изображений на основании формы объекта, строить скелеты изображений.

Список литературы

1. Гонсалес Р. Цифровая обработка изображений / Р. Гонсалес, Р. Вудс. – М. : Техносфера, 2005. – 1072 с.
2. Местецкий Л. Непрерывная морфология бинарных изображений / Л. Местецкий М.: Физмолит, 2009. – 288 с.

Методы оптимизации работы нейронных сетей

К. А. Капкин

Студент магистратуры

А. В. Акимов

Старший преподаватель

Введение

Современные методы на основе нейронных сетей относятся к кластеру наиболее востребованных и постоянно развивающихся алгоритмов машинного обучения, применяемых в различных сферах практической деятельности. Постоянно формируются разнообразные задачи, отличающиеся постановкой и типами входных данных: распознавание изображений, звуковых сигналов, синтаксический анализ текстов, диагностика заболеваний и так далее. Область применения таких алгоритмов невероятно широкая, на сегодняшний день обычный человек пользуется плодами этих технологий и в повседневной жизни. В связи с непрекращающимся совершенствованием существующих решений с использованием нейронных сетей, различающихся множеством свойств и особенностями реализации, возникает проблема определения наиболее эффективного метода минимизации функции ошибки, гарантирующего наилучшие результаты при решении конкретной задачи.

Известно, что наиболее используемым методом обучения нейронных сетей является алгоритм обратного распространения ошибки [6. 1], в котором минимизация целевой функции производится методом градиентного спуска [2]. Но обратное распространение не вносит в работу сети оптимизацию: оно считает значения градиента на внутренних узлах сети.

Целью данной статьи является анализ основных особенностей алгоритмов оптимизации работы нейронных сетей.

Стоит добавить, что, не умея пользоваться методами оптимизации, иначе как черным ящиком, практически невозможно повторить эксперименты по тренировке больших современных сетей, однако даже при умении грамотного использования методами оптимизации тоже приходится потрудиться, чтобы получить схожие значения метрик точности.

1. Градиентный спуск и его проблемы минимизации ошибок

Градиентный спуск — это алгоритм оптимизации, используемый для минимизации ошибок в модели машинного обучения. Он работает путем итеративной корректировки параметров модели в направлении отрицательного градиента функции потерь (которая представляет ошибку), чтобы уменьшить ошибку и найти оптимальные параметры, которые дают наилучшие результаты прогнозирования. Алгоритм продолжает этот процесс до тех пор, пока он не достигнет минимума или не будет выполнен заранее определенный критерий остановки.

Представим формулы для обычного градиентного спуска в обобщенном виде:

$$\Delta\theta = -\eta\nabla_{\theta}J(\theta_t), \quad (1)$$

$$\theta_{t+1} = \theta_t + \Delta\theta = \theta_t - \eta\nabla_{\theta}J(\theta_t), \quad (2)$$

где θ_t — параметры сети, $J(\theta_t)$ — целевая функция или функция потерь в случае машинного обучения, а η — скорость обучения. Со стороны может показаться, что в этих формулах нет ничего сверхъестественного, однако наибольший интерес в рамках статьи вызывает $\nabla_{\theta}J(\theta_t)$ — именно эта часть формулы с оператором набла обновляет параметры выходного слоя. Сложность заключается в том, чтобы добраться до параметров слоёв в скрытых слоях нейронной сети, приходится проходить через нелинейности, производные от которых вносят свой вклад. Здесь мы опираемся на принцип обратного распространения ошибки. Однако следует отметить, что обратное распространение не вносит в работу сети оптимизацию. Оно перемещает неверную информацию с конца сети на все веса внутри, чтобы другой алгоритм уже смог оптимизировать эти веса так, чтобы они соответствовали нашим данным.

Явно расписанные формулы для обновления весов в середине сети поражают объемом, ведь каждый нейрон зависит от всех нейронов, с которыми он связан, а те — от всех нейронов, с которыми связаны они, и так далее. При этом даже в не самых больших учебных нейронных сетях может быть порядка 100 слоев, а среди наиболее точных и производительных сетей количество слоев на несколько порядков больше. Каждый вес — переменная в $J(\theta_t)$. Такое невероятное количество степеней свободы позволяет находить наиболее сложные признаки и зависимости, но также создает некоторый набор проблем.

– Застревание в локальных минимумах или седловых точках, коих для функции множества переменных может быть очень много.

– Сложный ландшафт целевой функции: плато чередуются с регионами сильной нелинейности: производная на плато практически равна нулю, а внезапный обрыв, наоборот, может остаться незамеченным для алгоритма.

– Некоторые параметры обновляются значительно реже других, особенно когда в данных встречаются информативные, но редкие признаки, что плохо сказывается на нюансах обобщающего правила сети, с другой стороны, придание слишком большой значимости вообще всем редко встречающимся признакам может привести к переобучению.

– Слишком маленькая скорость обучения заставляет алгоритм сходиться очень долго и застревать в локальных минимумах, слишком большая — не замечать узкие глобальные минимумы или вовсе расходиться.

Вычислительной математике известны продвинутые алгоритмы второго порядка, которым под силу найти хороший минимум и на сложном ландшафте, но тут удар снова наносит количество весов. Чтобы воспользоваться методами второго порядка, необходимо посчитать гессиан $J(\theta_i)$ — матрицу производных по каждой паре параметров пары параметров, в некоторых случаях и обратную к ней. Рабочие оптимизаторы второго порядка существуют [3], но в рамках статьи сконцентрируемся на вычислительно подъёмном решении задачи с помощью методов первого порядка.

2. Методы оптимизации. Импульсные методы

Один из наиболее интуитивно понятных методов оптимизации поиска глобального минимума является идея методов с накоплением импульса [4]. Сформулировать ее можно следующим образом: если мы некоторое время движемся в определённом направлении, то, вероятно, нам следует туда двигаться и дальше без резкой смены направления движения. Для этого нужно уметь обращаться к недавней истории изменений каждого параметра. Можно хранить последние n экземпляров $\Delta\theta$ и на каждом шаге считать среднее, но такой подход занимает слишком много памяти для большого количества экземпляров. Так как в такого рода методах нужно не точное среднее, а оценка, поэтому используется экспоненциальное скользящее среднее.

Чтобы накапливать информацию о последних местоположениях, будем умножать уже накопленное значение на коэффициент сохранения $0 < \gamma < 1$ и прибавлять очередную величину, умноженную на $1 - \gamma$. Чем ближе γ к единице, тем больше окно накопления и сильнее сглаживание — история перемещения начинает влиять сильнее, чем каждое

очередное местоположение. Если мы попадаем на плато, значение v_t последних n экземпляров $\Delta\theta$ затухают экспоненциально, отсюда и название экспоненциального скользящего среднего. Применим экспоненциальное бегущее среднее, чтобы накапливать градиент целевой функции сети:

$$v_t = \gamma v_{t-1} + \eta \nabla_{\theta} J(\theta_t), \quad (3)$$

$$\theta_{t+1} = \theta_t - v_t, \quad (4)$$

Обычно γ берётся порядка 0,9. В формуле 3 выражение $1-\gamma$ входит в состав скорости обучения η . Чем меньше γ , тем больше алгоритм ведёт себя как обычный стохастический градиентный спуск [5]. Чтобы получить популярную физическую интерпретацию уравнений, представьте как шарик катится по холмистой поверхности. Если в момент времени t под шариком был ненулевой уклон $\nabla_{\theta} J(\theta_t)$, а затем он попал на плато, он всё равно продолжит катиться по этому плато. Более того, шарик продолжит двигаться пару обновлений в ту же сторону, даже если уклон изменился на противоположный. Тем не менее, на шарик действует вязкое трение и каждую секунду он теряет $1-\gamma$ своей скорости.

Заметим, что накопленное в v_t значение может очень сильно превышать значение каждого отдельного $\eta \nabla_{\theta} J(\theta_t)$. Простое накопление импульса уже даёт хороший результат, но метод Нестерова [6] идёт дальше и применяет следующую идею: игра на опережение по вектору обновления. Такое изменение позволяет быстрее «катиться», если в стороне, куда мы направляемся, производная увеличивается, и медленнее, если наоборот. Мы знаем, что на следующем шаге хотим сместиться на v_{t-1} , поэтому посчитаем градиент функции потерь не в точке θ_t , а в $\theta_t - \gamma v_{t-1}$. Преобразуем (3):

$$v_t = \gamma v_{t-1} + \eta \nabla_{\theta} J(\theta_t - \gamma v_{t-1}). \quad (5)$$

3. Адаптивные методы

Не все признаки, подаваемые на вход нейронной сети, имеют одинаковое влияние на конечный результат. Одни параметры могут содержать много шума и меняться чаще других с различной амплитудой. В выборке каких-то параметров могут встречаться редкие значения, которые в процессе обучения с фиксированным коэффициентом обучения нейронной сети будут не замечены. Одним из недостатков рассмотренного ранее метода стохастического градиентного

спуска является отсутствие механизмов оптимизации на подобных выборках. И как следствие, процесс обучения может остановиться в локальном минимуме. Для решения подобной проблемы были предложены адаптивные методы обучения нейронных сетей, которые позволяют динамически изменять скорость обучения в процессе обучения нейронной сети. Существует целый ряд подобных методов и их разновидностей. Рассмотрим наиболее часто встречаемые.

4. Adagrad

Для решения проблемы будем хранить для каждого параметра сети сумму квадратов его обновлений. Если параметр принадлежит цепочке часто активирующихся нейронов, значит сумма быстро накапливается. Используя замены в (6), перепишем формулу обновления следующим образом:

$$g_t = \nabla_{\theta} J(\theta_t), \quad \theta_{t+1} = \theta_t - \gamma v_{t-1}, \quad (6)$$

$$G_t = G_t + g_t^2, \quad \theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_t + \varepsilon}} g_t, \quad (7)$$

где G_t — сумма квадратов обновлений, а ε — сглаживающий параметр. У часто обновлявшегося в прошлом параметра большая G_t , значит большой знаменатель в (7). Параметр, изменившийся всего раз или два, обновится в полную силу. Сглаживающий параметр играет роль только в самом начале поиска глобального минимума.

Таким образом и работает метод адаптивного градиента Adagrad [7]. Ещё одно его достоинство — отсутствие необходимости точно подбирать скорость обучения. Достаточно выставить её в меру большой, чтобы обеспечить хороший запас, но не такой громадной, чтобы алгоритм расходился. По сути, мы автоматически получаем затухание скорости обучения.

5. RMSProp и Adadelta

Однако, на этом трудности не заканчиваются. Недостатком вышеизложенного метода является то, что G_t в (7) может увеличиваться сколько угодно, что через некоторое время приводит к слишком малым обновлениям и алгоритм не обновляется. Такие методы как RMSProp [8] и Adadelta [9] призваны исправить этот недостаток.

Для этого необходимо немного изменить подход Adagrad: так же меньше обновляем веса, которые слишком часто обновляются, но вместо полной суммы обновлений, будем использовать усреднённый по истории квадрат градиента.

Обозначим бегущее среднее квадрата градиента как $E[g^2]_t$ и перепишем (7). Знаменатель в (9) — корень из среднего квадратов градиентов, отсюда и название метода — root mean square propagation (RMSProp):

$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1-\gamma)g_t^2, \quad (8)$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \varepsilon}} g_t = \theta_t - \frac{\eta}{RMS[g]_t} g_t. \quad (9)$$

Метод Adadelta отличается от RMSProp тем, что мы добавляем в числитель (9) стабилизирующий член $RMS[\Delta\theta]_t$. Такое изменение сделано из соображений, что размерности θ_t и $\Delta\theta$ должны совпадать. На шаге t неизвестно значение $RMS[\Delta\theta]_t$, поэтому обновление параметров происходит в три этапа, а не в два. Сначала накапливаем квадрат градиента, затем обновляем θ_t , после чего обновляем $RMS[\Delta\theta]$.

Заметим, что нам нужен ненулевой $RMS[\Delta\theta]_{t-1}$ для первого шага, иначе все последующие значения будут равны нулю. Но эта проблема решена ранее, добавлением ненулевого сглаживающего параметра ε в (7). Однако следует учесть, что без большого значения $RMS[\Delta\theta]_{t-1}$ мы получим поведение, противоположное Adagrad и RMSProp: частое обновление весов часто встречающихся признаков. Вероятно, авторы методов как раз и добивались такого поведения алгоритмов. Для RMSProp и Adadelta, как и для Adagrad не нужно очень точно подбирать скорость обучения — достаточно прикидочного значения.

6. Adam

Метод Adam (adaptive moment estimation) [10], ещё один оптимизационный алгоритм. Он сочетает в себе и идею накопления движения и идею более слабого обновления весов для типичных признаков. От Нестерова Adam отличается накоплением значения градиента вместо $\Delta\theta$, однако это изменение не несёт больших отличий (13). Кроме того, мы хотим знать, как часто градиент изменяется:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)g_t, \quad (10)$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)g_t^2, \quad (11)$$

где m_t — экспоненциальное среднее градиента и v_t — экспоненциальное среднее квадратов градиента с гиперпараметрами β_1 и β_2 периода усреднения.

Легко заметить, что это уже знакомый нам $E[g^2]_t$, так что здесь практически нет отличий от RMSProp. Важное отличие состоит в начальной калибровке m_t и v_t : они страдают от той же проблемы, что и начальные значения в RMSProp и Adadelta: если задать нулевое начальное значение, то они будут долго накапливаться, особенно при большом окне накопления $0 \ll \beta_1 < 1$, $0 \ll \beta_2 < 1$. Чтобы не добавлять еще два гиперпараметра, предлагается [10] искусственно увеличивать m_t и v_t на первых шагах ($0 < t < 10$ для m_t и $0 < t < 1000$ для v_t). Итоговое правило обновления выглядит следующим образом:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t}, \quad (12)$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \varepsilon}} \hat{m}_t. \quad (13)$$

Авторы Adam предлагают [10] в качестве значений по умолчанию $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$ и утверждают, что алгоритм выступает лучше или примерно так же, как и все предыдущие алгоритмы на широком наборе датасетов за счёт начальной калибровки.

Заключение

В ходе работы были рассмотрены наиболее популярные оптимизаторы первого порядка, применяемые в современных методах обучения нейронных сетей. В рамках исследования был проведен анализ свойств и особенностей рассматриваемых методов, а также были предоставлены рекомендации для наиболее эффективной работы обозреваемых алгоритмов.

Нельзя не отметить, что на практике смена алгоритма обновления весов в один момент не решит все проблемы с нейронной сетью. Конечно, прирост при переходе от стохастического градиентного спуска к современному адаптивному методу будет заметен, однако не станет критичным.

Несмотря на то, что все рассмотренные в рамках этой статьи оптимизаторы могут найти путь к глобальному минимуму даже по плато с очень маленьким уклоном или сбежать из локального минимума, если

до этого они уже набрали импульс, плохая начальная точка может привести к плохим результатам.

Список литературы

1. Understanding Backpropagation Algorithm [Электронный ресурс]. – Режим доступа: towardsdatascience.com/understanding-backpropagation-algorithm-7bb3aa2f95fd/ (дата обращения 10.05.2023).
2. Gradient Descent Algorithm — a deep dive [Электронный ресурс]. – Режим доступа: towardsdatascience.com/gradient-descent-algorithm-a-deep-dive-cf04e8115f21/ (дата обращения 10.05.2023).
3. Transtrum, M. Improvements to the Levenberg-Marquardt algorithm for nonlinear least-squares minimization/ M. Transtrum, J. Sethna// DASP. – 2012. – DOI: 10.48550/arXiv.1201.5885.
4. Momentum in machine learning? What is Nesterov momentum? [Электронный ресурс]. – Режим доступа: medium.com/@abhinav.mahapatra10/ml-advanced-momentum-in-machine-learning-what-is-nesterov-momentum-ad37ce1935fc/
5. Компьютерный практикум «Цифровые технологии и искусственный интеллект» [Электронный ресурс]. – Режим доступа: mech.math.msu.su/~vzb/MasterAI/SGD.html/
6. Botev, A. Nesterov's Accelerated Gradient and Momentum as approximations to Regularised Update Descent / A. Botev, G. Lever, D. Barber // ML. – 2016. – DOI: 10.48550/arXiv.1607.01981.
7. Ward, R. AdaGrad stepsizes: Sharp convergence over nonconvex landscapes, from any initialization/ R. Ward, X. Wu, L. Bottou // ML. – 2016. – DOI: 10.48550/arXiv.1806.01811.
8. Kurbiel, T. Training of Deep Neural Networks based on Distance Measures using RMSProp/ T. Kurbiel, S. Khaleghian // ML. – 2017. – DOI: 10.48550/arXiv.1708.01911.
9. Zeiler, M. ADADELTA: An Adaptive Learning Rate Method / M. Zeiler // ML. – 2012. – DOI: 10.48550/arXiv.1212.5701.
10. Kingma, D. Adam: A Method for Stochastic Optimization / D. Kingma, J. Ba // ML. – 2015. – DOI: 10.48550/arXiv.1412.6980.

Автоматизированное тестирование бизнес-процессов ChatOps Telegram

А. Н. Квасов

Студент магистратуры

И. В. Илларион

доцент

Введение

В наше быстро меняющееся время разработка программного обеспечения становится все более сложной, а необходимость в высококачественном, надежном ПО никогда не была столь актуальной. Одним из ключевых элементов обеспечения качества программного продукта является тестирование, которое позволяет проверить и подтвердить соответствие характеристик ПО заявленным требованиям. Сейчас тестирование зачастую рассматривается как отдельный процесс, выполняющийся после завершения процесса разработки, но все большее количество компаний осознает, что запуск процессов тестирования на ранних этапах разработки ПО может принести значительные преимущества. На этих стадиях тестирования можно обнаружить и устранить многие проблемы, которые могут возникнуть позже в процессе разработки, что позволяет сократить время и затраты на исправление ошибок. Например, согласно исследованию IBM, исправление дефектов в производственной среде может стоить в 100 раз дороже, чем в случае, если ошибка была обнаружена и исправлена на ранних этапах разработки.

Для внедрения тестирования необходимо применять современные методы и подходы, одними из таких подходов являются: ChatOPS, юнит-тестирование и непрерывная интеграция.

Целью настоящей работы является: предложить технологию непрерывного тестирования на самых ранних этапах разработки программного обеспечения, позволяющую автоматизировать процесс тестирования и своевременно реагировать на возникающие ошибки для сокращения трудозатрат на разработку продукта.

1. Подходы к автоматизированному тестированию

Одним из наиболее распространенных подходов к автоматизированному тестированию является пирамида тестирования, включающая в себя юнит-тестирование, интеграционное тестирование, системное тестирование и сквозное тестирование.

Пирамида тестирования – это структура, показывающая, какие типы тестирования нужно проводить на разных уровнях разработки программного обеспечения. Она состоит из четырех уровней, каждый из которых имеет свой уровень абстракции, цели и технологии.

1. Юнит-тестирование – тестирование отдельных модулей программы для проверки их корректности работы. Модуль в данном случае представляет собой отдельную часть программы, которая может быть протестирована независимо от других модулей. Юнит-тестирование позволяет выявить и исправить ошибки на ранних этапах разработки, что уменьшает затраты на тестирование. Юнит-тестирование проводится на уровне исходного кода с помощью специальных фреймворков, таких как JUnit, NUnit, PHPUnit и т.д. В ходе тестирования проверяются отдельные методы, классы или компоненты программы.

2. Интеграционное тестирование – тестирование взаимодействия различных модулей программы, чтобы убедиться в их совместной работе без ошибок. Подход используется для тестирования сложных программных систем, состоящих из большого числа компонентов. В ходе интеграционного тестирования проверяется правильность взаимодействия между различными модулями, корректность передачи данных и т.д. Для проведения интеграционного тестирования используются специальные фреймворки, такие как Selenium, Appium, TestNG и др.

3. Системное тестирование – тестирование системы в целом с целью убедиться в ее соответствии требованиям и оценить ее работоспособность. Проводится после успешного юнит и интеграционного тестирования и может включать в себя тестирование функциональности, производительности, надежности, безопасности и других аспектов системы. Для проведения системного тестирования используются такие методы и инструменты, как функциональное тестирование, нагрузочное тестирование, тестирование безопасности и т.д. Всё это позволяет проверить систему на разных уровнях и выявить возможные проблемы и недостатки.

4. Сквозное тестирование – тестирование системы с точки зрения конечного пользователя, моделируя реальные сценарии использования. Этот вид тестирования позволяет выявить ошибки и проблемы,

связанные с необходимостью улучшения пользовательского опыта, отсутствием функциональности, которую пользователь ожидает, и другими недостатками, которые могут быть выявлены только при тестировании системы в реальных условиях.

2. Проектирование юнит-тестов на основе паттерна AAA

AAA (Arrange, Act, Assert) – паттерн, представляющий собой методологию проектирования юнит-тестов, включает три основных шага, которые необходимо выполнить при написании каждого теста: подготовка данных (Arrange), вызов методов (Act) и проверка результатов (Assert).

Первый шаг – Arrange – отвечает за подготовку начальных данных для тестирования. Здесь определяются все необходимые переменные, объекты и другие ресурсы, которые будут использоваться в тесте.

Второй шаг – Act – представляет собой вызов метода, которые необходимо протестировать. В этом шаге происходит выполнение тестируемого кода с переданными ему параметрами.

Третий шаг – Assert – отвечает за проверку полученных результатов тестирования. В этом шаге определяются ожидаемые значения и сравниваются с фактическими.

Преимущества паттерна AAA в том, что он позволяет значительно упростить процесс проектирования юнит-тестов и повысить их качество. Некоторые из преимуществ данного подхода:

1. Ясность. AAA позволяет четко разделить каждый тест на три части, что облегчает понимание его структуры и назначения.

2. Модульность. Каждый шаг теста является отдельной функцией, что упрощает поддержку и обновление кода.

3. Устойчивость. Применение паттерна AAA обеспечивает стабильность тестов, поскольку каждый шаг выполняется в строгом порядке и не зависит от других частей теста.

3. ChatOps в Telegram: Интеграция с CI/CD Github Actions

Одним из наиболее распространенных способов автоматизации процессов CI/CD является использование сервисов непрерывной интеграции, таких как Github Actions. Эти инструменты позволяют разработчикам автоматизировать процессы сборки, тестирования и развертывания приложения, что значительно сокращает время и уменьшает вероятность ошибок.

Однако, чтобы мониторить результаты этих процессов и быстро реагировать на возникшие проблемы, можно использовать ChatOPS в мессенджере Telegram. Для этого необходимо настроить интеграцию

между Github Actions и Telegram, чтобы уведомления об изменениях в статусе процессов отправлялись в определенный чат в Telegram.

Для начала необходимо создать Telegram-бота и получить его токен. Для этого нужно обратиться к боту @BotFather и выполнить несколько простых команд. Созданный бот будет служить посредником между Github Actions и чатом в Telegram.

Далее нужно создать чат в Telegram, в который будут отправляться уведомления о процессах CI/CD. Для этого достаточно создать новый чат и добавить в него созданного ранее бота.

После этого можно настроить интеграцию между Github Actions и Telegram. Для этого создается новый workflow в репозитории Github и указывается следующий код:

Листинг

Пример yaml.main

```
name: CI

on: [push]

jobs:
  build:
    runs-on: ubuntu-latest

    steps:
      - name: Checkout repository
        uses: actions/checkout@v2

      - name: Build and test
        run: |
          echo "Building and testing..."

      - name: Send notification
        uses: appleboy/telegram-action@master
        with:
          to: ${{ secrets.TELEGRAM_CHAT_ID }}
          token: ${{ secrets.TELEGRAM_BOT_TOKEN }}
          message: "CI/CD build was successful!"
```

В этом примере после того, как происходит push в репозиторий, запускается задание build, которое выполняет сборку и тестирование приложения. После успешного завершения задания, Github Actions использует плагин Telegram-action для отправки уведомления в указанный чат в Telegram.

Важно отметить, что для использования плагина необходимо предварительно создать секреты в репозитории, в которых хранятся токены для доступа к Telegram-боту и чату в Telegram.

Заключение

В рамках данной статьи был рассмотрен пример непрерывного тестирования программного обеспечения с использованием паттерна проектирования юнит-тестов AAA и интеграции Telegram с Github Actions. Данный подход позволяет управлять сборками из Telegram и при каждом изменении в коде проверять его на работоспособность и соответствие заявленным требованиям.

Список литературы

1. Ron Patton. Software Testing / Ron Patton. - Sams Publishing, 2005. - 408p.
2. ISTQB [Электронный ресурс]. – Режим доступа: <https://www.istqb.org/>
3. GitHub docs [Электронный ресурс]. – Режим доступа: <https://docs.github.com/>

Использование систем целочисленных сдвигов базисных сплайнов в методе Адамса

Е. А. Килигин

Студент магистратуры

И. А. Кольцова

Студент магистратуры

Е. А. Киселев

Доцент

Введение

В настоящее время дифференциальные уравнения и их системы часто возникают в различных математических моделях. Поскольку точное решение удастся найти далеко не всегда, развитие численных методов является актуальной задачей.

При решении задачи Коши для обыкновенных дифференциальных уравнений первого порядка довольно часто используется метод Адамса. Он основан на процедуре интерполяции. В его классическом варианте используется интерполяционный многочлен Лагранжа, но у этого подхода есть ряд недостатков. Во-первых, пригодным для практики является многочлен Лагранжа относительного небольшого порядка. При большом количестве узлов в качестве его альтернативы были придуманы сплайны [1, 2]. Во-вторых, переход к расчетной схеме более высокого порядка требует использования совершенно другого набора коэффициентов метода. Это довольно неудобно с точки зрения программной реализации.

В одной из недавних статей [3] была разработана модификация метода Адамса, которая позволяет преодолеть указанные недостатки. Вместо многочлена Лагранжа в ней предлагалось использовать интерполяцию по системам целочисленных сдвигов. В качестве примера были рассмотрены системы сдвигов, порожденные функциями Гаусса и Лоренца. В данной работе мы используем ту же идею, но предлагаем несколько иные семейства функций. Как показывают вычислительные эксперименты, базисные сплайны позволяют достичь более высокой

точности нахождения численного решения, чем при использовании функций Гаусса и Лоренца.

1. Общие формулы

Дадим краткий обзор основных идей и формул статьи [3]. Рассматривается задача Коши

$$y'(x) = F(x, y(x)), y(x_0) = y_0, \quad (1)$$

где функция $F(x, y(x))$ и начальное условие $y(x_0) = y_0$ заданы. Предполагается, что решение (1) в исследуемой области существует и единственно. Выберем дискретные точки $x_k = x_0 + kh$, $k = 0, 1, 2, \dots$ с некоторым постоянным шагом h . Значения решения в этих точках обозначим для краткости как y_k . Для модификации явной схемы метода Адамса расчетные формулы, полученные в статье [3], выглядят следующим образом:

$$y_{k+1} = y_k + h \sum_{m=0}^{M-1} u_m F(x_{k-m}, y_{k-m}), \quad (2)$$

где параметр M задает количество предыдущих значений решения y_{k-m} , которые используются для вычисления следующего y_{k+1} , а u_k – некоторые числовые коэффициенты, о которых мы поговорим позднее. Важным обстоятельством сейчас является только то, что набор значений u_k не зависит от M . Следовательно, при попытке повысить точность расчетной схемы за счет увеличения M пересчитывать u_k не потребуется. Аналогично выглядит и модификация неявной схемы:

$$y_{k+1} = y_k + h \sum_{m=-1}^{M-2} u_m F(x_{k-m}, y_{k-m}), \quad (3)$$

Отметим, что в этом равенстве $u_{-1} = u_0$.

Прежде чем привести формулу для коэффициентов u_m , сформулируем одно вспомогательное определение. Пусть $\varphi(x)$ – заданная функция. Как мы увидим, ее выбор играет очень важную роль.

Определение 1. Функция $\tilde{\varphi}(x)$, являющаяся линейной комбинацией $\varphi(x-k)$, $k \in \mathbb{Z}$,

$$\tilde{\varphi}(x) = \sum_{k=-\infty}^{\infty} d_k \varphi(x-k), \quad (4)$$

называется узловой, если она удовлетворяет условию

$$\tilde{\varphi}(m) = \delta_{0,m}, m \in \mathbb{Z}. \quad (5)$$

Отметим, что подобного рода конструкции обычно используются для решения задачи об интерполяции [4, 5].

В терминах коэффициентов узловой функции d_k , следуя [3], запишем формулу для u_k :

$$u_k = \sum_{m=-\infty}^{\infty} d_m \int_0^1 \varphi(t+k-m) dt. \quad (6)$$

Отметим, что в данном исследовании мы взяли несколько иное определение узловой функции, чем в [3], благодаря чему некоторые формулы теперь выглядят проще.

2. Функции Гаусса и Лоренца

Пусть $\varphi(x)$ – это функция Гаусса, т. е.

$$\varphi(x) = \exp\left(-\frac{x^2}{2}\right), \quad (7)$$

Этот случай подробно рассмотрен в статье [3], но отсутствовали примеры практического применения. В качестве развития этого направления мы решили исследовать также функции вида

$$\varphi(x) = x^{2N} \exp\left(-\frac{x^2}{2}\right), N = 1, 2, 3, \dots \quad (8)$$

В силу определенных причин построить узловую функцию удастся только при четных значениях степени x . Коэффициенты рассчитывались численно по методике, указанной в [4, 5]. Подробное описание этой громоздкой процедуры не является целью данной работы, поэтому сразу перейдем к вычислительным экспериментам, связанным с решением дифференциальных уравнений. Используя приведенные в [3] коэффициенты u_k , мы рассмотрели несколько дифференциальных уравнений, для которых известно точное решение.

На рис. 1 представлен пример приближенного решения задачи Коши $y' = y - x$, $y(0) = 0$. Ее точное решение имеет вид $y(x) = e^{-x} + x - 1$ (на рис. 1 оно показано жирными точками). Пунктир – это численное решение, полученное классическим методом Адамса второго порядка точности, а жирная линия – решение, полученное с помощью интерполяции по функциям Гаусса также с использованием только двух слагаемых. Как видим, точность при относительно малом

количестве точек ($h = 0.1$) невысокая. Еще хуже результаты для функций вида (8), поэтому мы не будем их приводить.

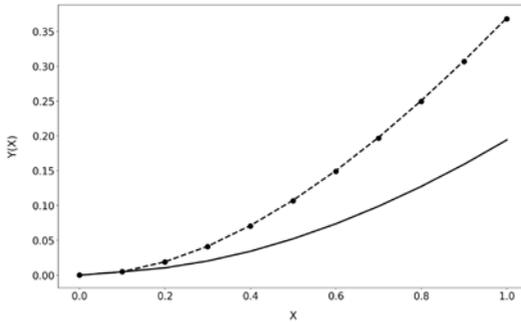


Рис. 1. Пример приближенного решения дифференциального уравнения с использованием функции Гаусса

Пусть теперь $\varphi(x)$ – это функция Лоренца, т. е.

$$\varphi(x) = \frac{\sigma^2}{\sigma^2 + x^2}. \quad (9)$$

За исключением примеров практического применения, этот случай также подробно рассмотрен в статье [3]. На рис. 2 изображен пример численного решения. Точность также довольно плохая. Таким образом, функции, предложенные в статье [3], оказываются малоприспособленными для построения алгоритмов численного решения.

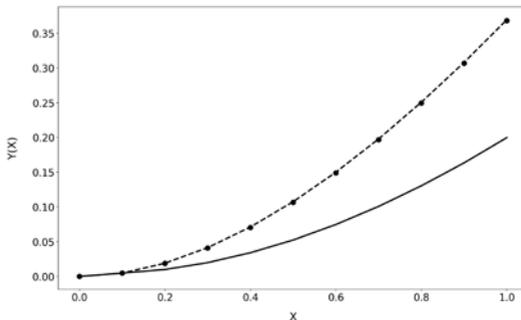


Рис. 2. Пример приближенного решения дифференциального уравнения с использованием функции Лоренца

3. Базисные сплайны

Пусть $\chi_{[a,b]}(x)$ – характеристическая функция отрезка $[a,b]$, т. е.

$$\chi_{[a,b]}(x) = \begin{cases} 1, & x \in [a,b], \\ 0, & x \notin [a,b]. \end{cases} \quad (10)$$

Зададим центрированные базисные сплайны $B_n(x)$ рекуррентно с помощью свертки [2, 4]:

$$B_1(x) = \chi_{[-0.5,0.5]}(x), \quad (11)$$

$$B_{n+1}(x) = \int_{-\infty}^{\infty} B_n(x-t)B_1(t)dt, n > 1. \quad (12)$$

Случаи $n=1$ и $n=2$ тривиальны:

$$d_k = \delta_{0,k}, u_k = \frac{1}{2}\delta_{0,k}, k \in Z. \quad (13)$$

Поскольку $u_{-1} = u_0$, это соответствует неявной схеме метода Адамса второго порядка точности.

При $n=3$ известен следующий результат [4]:

$$d_k = (-1)^k (3 - 2\sqrt{2})^{|k|}. \quad (14)$$

Для коэффициентов u_k мы получили формулу

$$u_k = d_{k-1} \frac{1}{48} + d_k \frac{23}{48} + d_{k+1} \frac{23}{48} + d_{k+2} \frac{1}{48}. \quad (15)$$

При $n=4$ справедливо соотношение [4]:

$$d_k = (-1)^k \sqrt{3}(2 - \sqrt{3})^{|k|}. \quad (16)$$

Для u_k тогда имеем

$$u_k = d_{k-1} \frac{1}{24} + d_k \frac{11}{24} + d_{k+1} \frac{11}{24} + d_{k+2} \frac{1}{24}. \quad (17)$$

В случае сплайнов более высоких порядков формулы для d_k и u_k очень быстро становятся весьма громоздкими. Кроме того, имеет место предельное соотношение

$$\lim_{n \rightarrow \infty} \left(B_n(x) - \frac{1}{\sigma_n \sqrt{2\pi}} \exp\left(-\frac{x^2}{2\sigma_n^2}\right) \right) = 0, \quad (18)$$

где $\sigma_n = \sqrt{n/12}$, причем предел практически достигается уже при $n = 10$ [6]. Поэтому мы ограничимся детальным рассмотрением только сплайнов порядков $n = 3$ и $n = 4$.

На рис. 3 и 4 представлены примеры численного решения того же дифференциального уравнения с помощью интерполяции на основе $B_3(x)$ и $B_4(x)$. Наилучших результатов при использовании двух слагаемых, как видим, удастся достичь с помощью $B_3(x)$. Правда, классический метод Адамса по-прежнему справляется со своей задачей лучше. В связи с этим, актуальной задачей будет поиск других типов функций, которые, возможно, позволят достичь более хороших результатов.

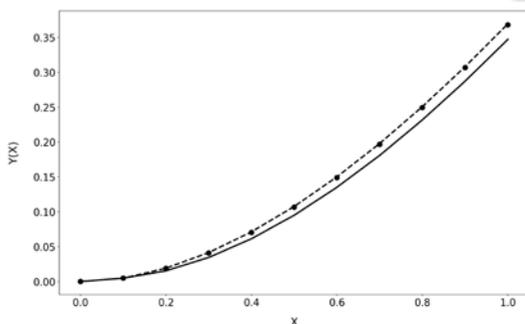


Рис. 3. Пример приближенного решения дифференциального уравнения с использованием $B_3(x)$

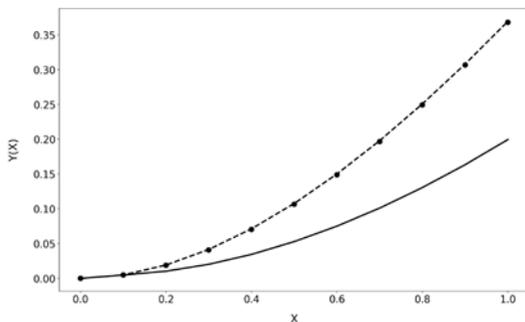


Рис. 4. Пример приближенного решения дифференциального уравнения с использованием $B_4(x)$

Заключение

В данной работе рассмотрены возможности практического применения модифицированного метода Адамса для обыкновенных дифференциальных уравнений первого порядка на основе базисных сплайнов. Он позволяет строить расчетные схемы разных порядков, используя одинаковый набор коэффициентов u_k . Это делает возможным достаточно просто реализовать алгоритм численного решения с постепенно увеличивающимся количеством слагаемых, которому планируется посвятить дальнейшие исследования. В настоящий момент наилучших результатов в плане точности удалось достичь с помощью сплайнов третьего порядка. В будущем планируется провести вычислительные эксперименты с другими типами функций, чтобы выяснить, как их следует выбирать в зависимости от вида $F(x, y)$ в правой части уравнения (1).

Список литературы

1. Бахвалов Н. С. Численные методы : учебное пособие / Н. С. Бахвалов, Н. П. Жидков, Г. М. Кобельков. – Москва : Наука, Физматлит, 1987. – 600 с.
2. Завьялов Ю. С. Методы сплайн-функций / Ю. С. Завьялов, Б. И. Квасов, В. Л. Мирошниченко. – Москва : Наука, 1980. – 352 с.
3. Киселев Е. А. Обобщение метода Адамса путем использования интерполяции с помощью систем равномерных сдвигов / Е. А. Киселев, В. А. Ушаков // Физико-математическое моделирование систем : Материалы XXII Международного семинара, Воронеж, 27–28 ноября 2020 года. – Воронеж : Воронежский государственный технический университет, 2021. – С. 107–111.
4. Чуи Ч. Введение в вейвлеты : учебное пособие для студентов вузов / Ч. Чуи ; пер. с англ. Я. М. Жилейкина. – Москва : Мир, 2001. – 412 с.
5. Maz'ya V. Approximate approximations / V. Maz'ya, G. Schmidt. – AMS Mathematical Surveys and Monographs, 2007. – V. 141. – 350 p.
6. Unser M. On the Asymptotic Convergence of B-Spline Wavelets to Gabor Functions / M. Unser, A. Aldroubi, M. Eden // IEEE Transactions on Information Theory. – 1992. – V. 38, № 2. – P. 864–872.

Разработка генератора REST API сервисов на основе модели данных

И. И. Клишин

Студент магистратуры

Е. Н. Десятирикова

Профессор

Введение

На сегодняшний день скорость разработки и поддержка качества продукта являются одними из основных проблем, возникающих при разработке любого продукта. В случае, когда необходимо разработать один-два сервиса, данные проблемы можно решить расширением команды разработчиков, однако в случае, когда необходимо разработать несколько десятков однотипных сервисов в короткие сроки – необходимо другое решение. Таким решением может стать генератор REST API, позволяющий создавать REST API сервисы на основе модели данных, используемой в сервисе.

В процессе работы, генератор REST API сервисов создает полную структуру сервиса – слой доступа к данным, логический уровень, а также сам API, предоставляющий доступ к ресурсам для дальнейшего использования сервиса. Полученный сервис готов к развертыванию в рабочем окружении и использованию непосредственно после генерации, а при необходимости может быть дополнен необходимыми методами для реализации бизнес-требований.

При использовании данного генератора сокращается время на разработку базового «каркаса» практически любого современного приложения – серверной части. А благодаря автоматической генерации Open API документации упрощает интеграцию как на стороне клиента, так и на стороне сторонних сервисов [1].

1. Описание модели данных

Модель данных представляет собой описание сущностей и связей, используемых в сервисе. В самом простом случае модель данных можно представить в виде XML-документов, в которых описаны сущности. На рис. 1 представлен пример схемы базы данных, используемый для описания модели данных. Схема содержит три таблицы – Customers

(покупатели), Orders (Заказы) и Shipments (доставки). В данной схеме присутствует только один вид связи – один ко многим.

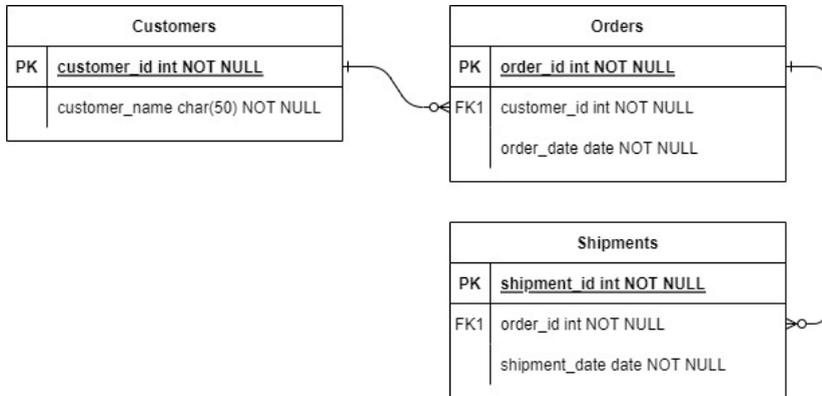


Рис. 1. Пример схемы базы данных

Данной схеме базы данных соответствует описание модели данных, приведенное в листингах 1, 2, 3.

Листинг 1

Описание сущности Customers

```

<entity name="customer" datasource="postgres">
  <field name="customer_id" type="int" nullable="false">
  <field name="customer_name" type="char" max="50"
nullable="false">
</entity>
    
```

Листинг 2

Описание сущности Shipments

```

<entity name="shipment" datasource="postgres">
  <field name="shipment_id" type="int" nullable="false">
  <field name="order_id" type="int" nullable="false"
foreign="order.order_id">
  <field name="shipment_date" type="date"
nullable="false">
</entity>
    
```

Листинг 3

Описание сущности Orders

```

<entity name="order" datasource="postgres">
  <field name="order_id" type="int" nullable="false">
    
```

```

    <field name="customer_id" type="int" nullable="false"
foreign="customer.customer_id">
    <field name="order_date" type="date" nullable="false">
</entity>

```

Описанные сущности представлены в виде XML-документов и предназначены непосредственно для обработки генератором REST API сервисов. Формат XML не является безальтернативным средством представления модели данных и для данной задачи может быть использован любой другой формат данных.

2. Архитектура разрабатываемого генератора

Разрабатываемый генератор представляет собой многоуровневую систему, состоящую из различных компонентов. На рис. 2 изображена диаграмма компонентов.

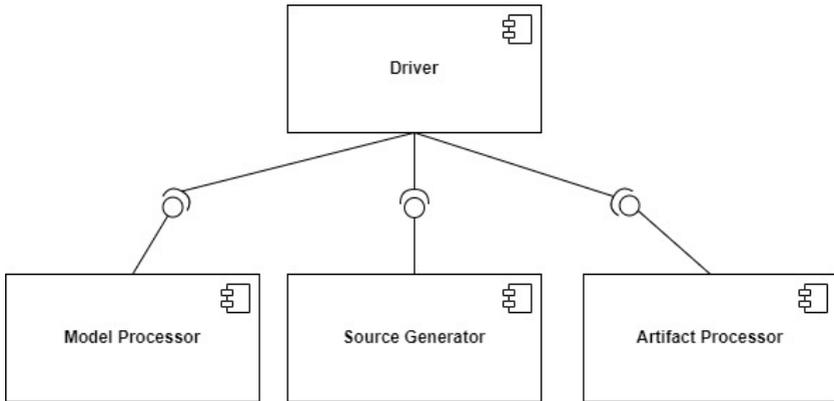


Рис. 2. Диаграмма компонентов разрабатываемого генератора

Компонент Driver представляет собой «обертку» для основных компонентов – Model Processor, Source Generator, Artifact Processor. Задача компонента Driver – корректный вызов функций и процедур, расположенных в компонентах.

Компонент Model Processor отвечает за первоначальную обработку модели данных – валидацию, как самой модели (формат, атрибуты

сущностей), так и связей, а также конвертирование модели во внутреннее представление для дальнейшей работы. На рис. 3 продемонстрирована диаграмма классов компонента Model Processor.

Компонент Model Processor предоставляет публичный интерфейс IModelProcessor содержащий методы isValid() и toModel(). Данные методы предназначены для валидации модели данных (isValid()), а также для конвертирования XML-модели во внутреннюю модель в виде программных объектов (toModel()).

Интерфейс IModelProcessor позволяет использовать и реализовывать множество реализаций компонента Model Processor. Пользователю данного интерфейса доступно изменение поведения как самого IModelProcessor, так и более низкоуровневых интерфейсов – IXMLModelValidator и IXMLModelConverter, объекты которых используются при валидации и конвертировании модели данных.

Компонент Model Processor предоставляет базовую реализацию интерфейсов IModelProcessor, IXMLModelValidator, IXMLModelConverter в виде классов XMLModelProcessor, XMLModelValidator, XMLModelConverter соответственно.

Интерфейс IXMLModelConverter следует контракту «exception safe», это означает, что если метод isValid() вернул true, значит модель данных корректна и вызов toModel() не приведет к возникновению ошибки, в противном случае, если метод isValid() вернул false – модель некорректна и toModel() не следует вызывать дабы избежать возникновения ошибок. Данный подход позволяет контролировать обработку ошибок на каждом из уровней системы, не прибегая к стандартному способу – механизму исключений, который является хоть и удобным средством работы с ошибками, однако в некоторых ситуациях влияет на производительность системы в целом [2].

Стоит отметить, что пользователя данного интерфейса ничто не ограничивает и он по собственному желанию может использовать механизм исключений, например, как на листинге 4.

Листинг 4

Пример использования механизма исключений

```
if (!processor.isValid(xmlDataModel)) {  
    throw new RuntimeException("XML data model is invalid");  
}
```

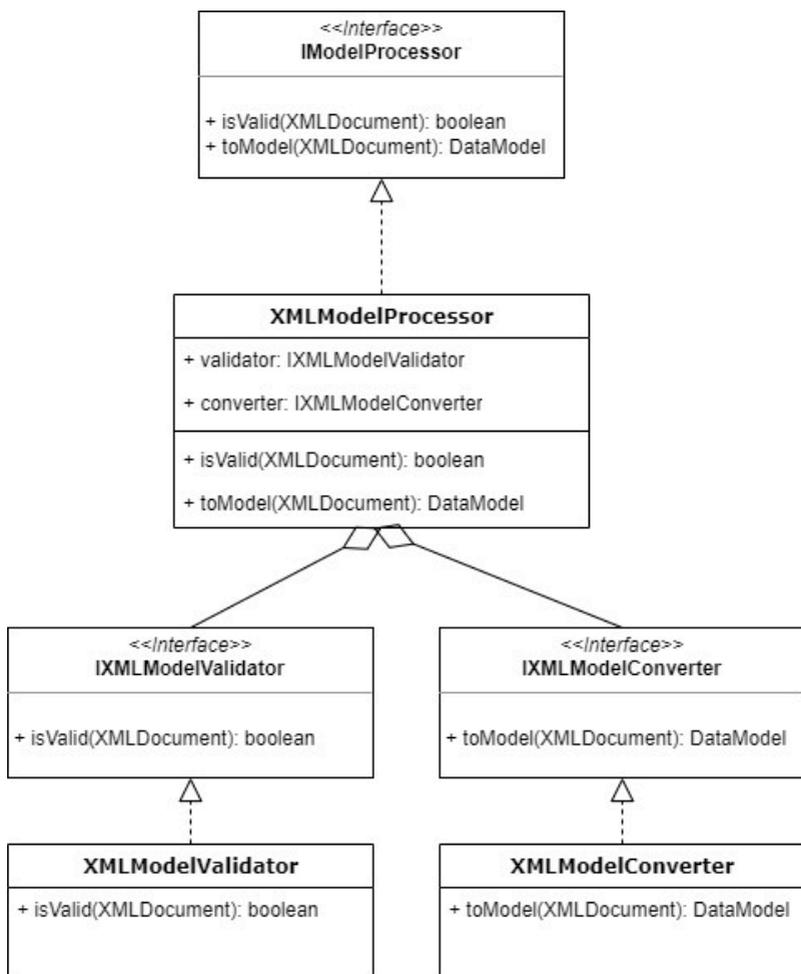


Рис. 3. Диаграмма классов компонента Model Processor

Компоненты Source Generator, Artifact Processor реализованы идентичным образом компоненту Model Processor, что позволяет унифицировать разработку как классов внутри каждого из компонентов, так и других компонентов при необходимости.

3. Генерация REST API сервиса

После того как модель данных описана в виде XML документов её необходимо предоставить драйверу генератора. В качестве драйвера в

базовой реализации представлена утилита командной строки, принимающая на вход путь до модели данных. В процессе работы генератора создается полностью структура сервиса. На рис. 4 продемонстрированы уровни сервиса в виде диаграммы компонентов.

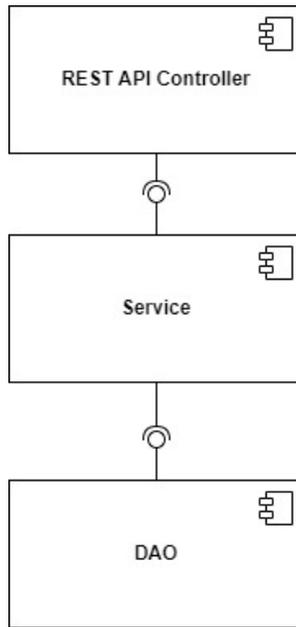


Рис. 4. Диаграмма компонентов полученного сервиса

Полученный в ходе работы генератора сервис построен по принципу MVC и имеет следующую структуру – на самом верхнем уровне представлен REST API Controller – компонент, обеспечивающий обработку HTTP запросов к сервису. В листинге 4 приведен пример обработчика HTTP запроса.

Листинг 5

Обработчик HTTP запроса в REST API Controller

```
@Operation(summary = "Get all order")
@GetMapping(produces = {MediaType.APPLICATION_JSON_VALUE})
@ApiResponse(responseCode = "200", description = "Order
list")
public Mono<ResponseEntity<List<OrderResponse>>>
findAllOrders() {
    return orderService.findAll()
```

```

        .map(orderMapper::toResponse)
        .collectList()
        .map(ResponseEntity::ok);
    }

```

В приведенном листинге описан обработчик HTTP GET запроса, выполняющего выборку всех заказов (сущность Order). Данный обработчик обеспечивает обращение к orderService – компоненту Service, который в свою очередь перенаправляет запрос в компонент DAO – объект доступа к данным.

Данная передача запроса между компонентами необходима для поддержания слабой связи между компонентами сервиса для обеспечения большей гибкости и надежности системы в целом [3]. Кроме того, позволяет инкапсулировать в каждом компоненте логику только свойственную данному компоненту – в Service различную логику, связанную с бизнес-задачами, в DAO – исключительно логику работы с источником данных.

4. Анализ метрик

После того, как сгенерированный сервис развернут в рабочем окружении, возникает необходимость сбора различных показателей (метрик), указывающих на эффективность системы. Для анализа эффективности сгенерированного сервиса, а также самого генератора рассматриваются метрики, представленные в таблице ниже.

Таблица

Метрики полученного сервиса

Вид реализации сервиса	Время процессора (% от доступного)	Время операций ввода/вывода (% от общего времени исполнения)	Потребляемая память (Мб)	Пропускная способность (запросов в секунду)
Ручная	2,36	12,59	382,3	5844
Генератор	3,59	11,3	301,18	5927

Окончание Таблицы

В таблице представлены основные метрики, отвечающие за эффективность работы системы – время процессора, время операций ввода/вывода, потребляемая память, а также более общая

характеристика, пропускная способность, отражающая эффективность системы в целом.

Как видно из таблицы, ручная реализация имеет преимущество в виде меньшего потребления времени процессора, в тоже время, генератор сгенерировал код сервиса с меньшим числом операций ввода/вывода и меньшим количеством потребляемой памяти, в результате чего общая эффективность в виде пропускной способности возросла. Данный результат возможен в связи с использованием генератором низкоуровневых инструментов для доступа к данным, которые позволили реализовать полностью компонент DAO без единой строчки кода, написанной разработчиком.

Заключение

Данная статья посвящена разработке генератора REST API сервисов на основе модели данных. Продемонстрированы архитектура разрабатываемого решения, а также процесс работы генератора. В итоге данной разработки получился полностью рабочий программный модуль, реализованный в виде утилиты командной строки. Данная разработка может быть полезна при разработке прототипов, или разработке при ограниченных ресурсах. В дальнейшем предполагается доработать данное решение до возможности генерации клиентского приложения.

Список литературы

1. Арно Л. Проектирование веб-API / Арно Л. – ДМК Пресс, 2020. – 116 с.
2. Блох Д. Java: эффективное программирование / Блох Д. – Вильямс, 2018. – 359 с.
3. Хо Кларенс, Козмина Юлиана, Шефер Крис. Spring 5 для профессионалов / Хо Кларенс, Козмина Юлиана, Шефер Крис – Диалектика-Вильямс, 2019. – 890-892 с.

Поиск сигнала спутниковой навигационной системы Galileo

С.С. Крупенин

Студент магистратуры

А. Ю. Савинков

Профессор

Введение

Спутниковая система навигации - электронно-техническая комплексная система, состоящая из совокупности космического и наземного оборудования для определения местоположения и параметров движения для наземных, воздушных и водных объектов, а также низкоорбитальных космических аппаратов

В современном мире системы навигаций используются в различных сферах жизни. Системы определения местоположения реализованы во многих устройствах от систем навигации кораблей и самолетов до телефонов и умных часов.

Но реализация для карманных устройствах немного отличается. При определении местоположения устройства используют дополнительные данные, которые устройства получают через сотовую сеть. Но бывают такие ситуации, что недоступна сотовая связь или устройство не имеет модуля связи (различные микроконтроллеры). В таких случаях для получения более точного местоположения необходимо полагаться на все доступные спутниковые системы.

В мире существуют такие системы как: ГНСС ГЛОНАСС (Россия) и GPS (США), Китай и страны Европейского союза проводят работы по развертыванию ГНСС Beidou и Galileo.

Если первые две появились достаточно давно, чтобы для них появились оптимизации, то вот для такой молодой системы как Galileo в текущий момент не так много.

И поэтому в данной работе будет представлено собственное решение поиска сигнала спутниковой навигационной системы Galileo.

1. Особенности системы Galileo

Galileo — это спутники, передающие сигнал с точным временем, а устройство на Земле рассчитывает на основе сигналов от нескольких спутников и данных об их орбитах свое местоположение с точностью, обычно составляющей около нескольких метров. Также Galileo имеет

вторичный код, который может изменить фазу сигнала при переходе через границу символа. Данная особенность позволяет передавать дополнительную информацию. [1]

2. Постановка задачи

Современная спутниковая навигация основывается на использовании принципа беззапросных дальномерных измерений между навигационными спутниками и потребителем. Это означает, что потребителю в составе навигационного сигнала передается информация о координатах спутников. Одновременно (синхронно) производятся измерения дальностей до навигационных спутников. Способ измерений дальностей основывается на вычислении временных задержек принимаемого сигнала от спутника по сравнению с сигналом, генерируемым аппаратурой потребителя. [2]

В данной работе будет рассматриваться решение поиска сигнала для спутниковой навигационной системы Galileo в частотно-временной области.

3. Разработка системы распределения памяти

Возможно два подхода к реализации начального поиска:

- Прямое вычисление свертки во временной области
- Вычисление циклической свертки с использованием FFT-IFFT

Первый способ позволяет использовать всю энергию сигнала. Так как опорный сигнал принимает значения ± 1 , алгоритм реализуется без умножений.

Второй способ основан на известном свойстве преобразования Фурье. Свертка функций во временной области равна обратному преобразованию Фурье от произведения спектров этих функций в частотной области. Это существенно уменьшает вычислительную сложность. Но обратное преобразование Фурье не учитывает вторичный код, который может изменить фазу сигнала при переходе через границу символа. Ситуация поясняется на следующем рис. 1.



Рис. 1. Изменение фазы сигнала при переходе через границу символа

Для решения данной проблемы выборка для FFT может быть разделена на две полувыборки, обрабатываемые независимо. Тогда по крайней мере одна из полувыборок не будет содержать фазового скачка из-за вторичного кода. Для решения проблемы вторичного кода необходимо использовать две выборки с масками. Данная ситуация приводится на рис. 2.

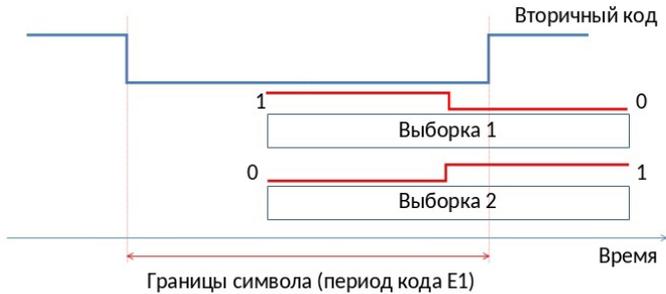


Рис. 2. Применение двух масок для выборки

Но это накладывает сложности так как использование двух выборок повышает в два раза вычислительные затраты и снижает энергию сигнала в накоплении в два раза (3 дБ потери чувствительности).

Также поиск по времени осложняется неопределенностью значения частоты несущей входного сигнала, обусловленной доплеровским сдвигом частоты и погрешностью опорного генератора приемника. Из-за движения спутников по своей орбите наблюдается эффект Доплера. Эффект Доплера - изменение частоты и, соответственно, длины волны излучения, воспринимаемой наблюдателем (приемником), вследствие движения источника излучения относительно наблюдателя (приемника). Поэтому необходимо реализовать проверку гипотез о частотном сдвиге входного сигнала. С учетом скорости движения спутников относительно поверхности земли и характеристик опорного генератора приемника, неопределенность частоты ΔF предполагалась в диапазоне ± 10 кГц.

Поэтому предполагаем гипотезы: При когерентных накоплениях набег фазы из-за несоответствия частот входного и опорного сигнала не должен превысить $\pi/2$, поэтому для 4 мс накоплений частотная ошибка не должна превышать 62.5 Гц.

Ввиду неопределенности частотного сдвига необходимо ввести гипотезы о частотном сдвиге с шагом сетки частот 125 Гц.

Основные шаги алгоритма поиска:

- Resampling для получения частоты дискретизации, кратной чиповой скорости Галилео
- Снятие цифровой модуляции
- Предварительные когерентные накопления
- Гипотезы о частотном сдвиге
- Окончательные когерентные накопления с использованием FFT
- Некогерентные накопления
- Оценка времени и частоты.

Заключение

Таким образом, получилось решение поиска сигнала спутниковой навигационной системы Galileo, который позволяет ускорить процесс поиска первого решения. По сравнению с прямым вычислением свертки во временной области, разработанный алгоритм с использованием циклической свертки с использованием FFT-IFFT отличается большой эффективностью.

Список литературы

1. European GNSS Service [Электронный ресурс]. – Режим доступа: <https://www.gsc-europa.eu/galileo/what-is-galileo>
2. Принципы навигации [Электронный ресурс]. – Режим доступа: <https://glonass-iac.ru/guide/navfaq.php>

Разработка платформы для построения чат-ботов

П. В. Кубленко

Студент магистратуры

Д. И. Соломатин

Старший преподаватель

Введение

В настоящее время наблюдается настоящий рассвет эры чат-ботов. По оценке мировой ассоциации телеком-операторов GSMA, в 2024 году 85% пользовательских обращений будут обрабатываться без помощи человека. Отвечать на них будут чат-боты.

51% пользователей ожидают от брендов немедленного ответа в любое время суток. Для этого бизнесы запускают круглосуточную службу поддержки и тратят на это большие деньги. Но большинство вопросов пользователей типичные, и на них вполне может отвечать правильно настроенная программа — к примеру, чат-бот. И пользователей это устраивает. Недавно Facebook опубликовал множество данных, подтверждающих ценность ботов для бизнеса: 2 миллиарда сообщений отправляются между людьми и предприятиями, ежемесячно 56% людей предпочли бы получить сообщение, чем звонить в службу поддержки клиентов. 53% людей с большей вероятностью совершают покупки в предприятиях, которые могут отправлять сообщения. [1] Чат-бот может принести бизнесу дополнительный канал продаж. Люди проводят в мессенджерах около 80% времени использования смартфона. Поэтому e-commerce-бизнесам стоит смотреть в сторону чат-ботов, как дополнительного канала продаж.

С точки зрения продаж, чат-бот похож на приложение. В том же Telegram можно нативно из бота вызывать оплату с Apple Pay. Самое главное тут — аудитория, которая определяет выбор мессенджера. К примеру, по географии — в одних регионах России более популярен WhatsApp, Viber, в других — Telegram.

Важно помнить, что чат-бот — это не приложение и не сайт. Поэтому не нужно переносить, к примеру, добавление товаров в корзину и оформление заказа в бот. Чат-бот — это про скорость, простоту и удобство. Чем проще диалог, тем выше продажи.

1. Постановка задачи

Реализовать веб-платформу, в которой пользователи могут:

- создавать узконаправленных чат-ботов по своим входным данным;
- интегрировать ботов с мессенджерами;
- управлять созданными ботами.

Основная идея заключается в том, чтобы освободить пользователя от выполнения лишней работы по самостоятельному созданию чат-бота пусть даже и в no-code режиме. Все, что нужно сделать пользователю, это принести в систему текстовый файл, содержащий как можно более подробную информацию о бизнесе или продукте пользователя. Система сама обрабатывает эту информацию, сформулирует как можно больше возможных пользовательских вопросов и на основе этой информации обучит бота анализировать пользовательский ввод и подбирать как можно более подходящий ответ.

Итак, если обобщить описанное выше, то система должна:

1. Принять на вход файл с пользовательской информацией.
2. С помощью нейронной сети сформулировать как можно больше вопросов к этой информации и конкретные ответы на них.
3. С помощью файла, получившегося на этапе 2, обучить нейронную сеть сопоставлять пользовательский ввод с вопросами в файле и выдавать связанный с этим вопросом ответ. По сути, полученная модель и будет готовым чат-ботом, поскольку сможет обрабатывать входные данные (пользовательский ввод) и подбирать для этих данных подходящий ответ (ответ пользователю в чате).
4. С помощью указанных пользователям при создании бота ключах интеграции в мессенджеры, например Telegram, интегрировать и запустить рабочего бота.

Входной файл представляет собой текстовый файл, содержащий некий массив текста. Файл для обучения представляет собой JSON-файл с парами вопрос-ответ.

2. Реализация

Были испробованы следующие подходы к построению файла для обучения:

- дообучение уже существующей модели Transformer, например mt5-base от Google.[2] Из недостатков подхода можно выделить необходимость сборки огромного датасета для проведения качественного обучения и необходимость наличия большого вычислительного ресурса для его осуществления;

- использование полностью готовой языковой модели (GPT-3.5 в ChatGPT).

GPT-3.5 — это нейронная сеть на основе архитектуры Transformer, которая используется для генерации текстов на основе заданного контекста. Она была обучена на огромном объеме текстовых данных, включая книги, статьи, новости и различные текстовые документы, что позволило ей получить широкие знания о языке и контексте.

Архитектура Transformer была представлена в 2017 и стала одной из наиболее успешных моделей в области обработки естественного языка. Архитектура Transformer состоит из двух основных блоков: кодировщика и декодировщика. На рис. 1 показана высокоуровневая архитектура модели.[3] Более глубокое погружение выходит за рамки этой работы.

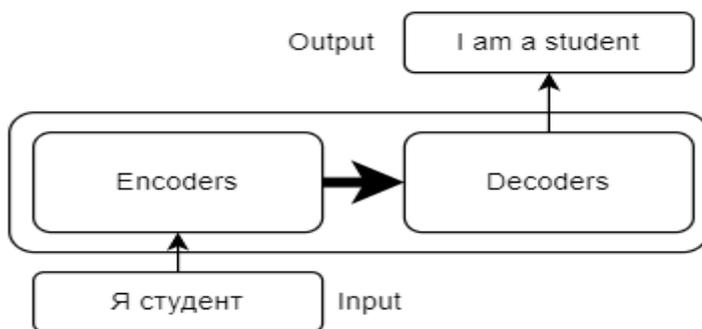


Рис. 1. Высокоуровневая архитектура модели Transformer

3. Прикладное использование ChatGPT

Существует несколько способов использования ChatGPT:

- самый стандартный способ использования ChatGPT – прямое общение с нейросетью с помощью веб-сайта компании OpenAI;
- использование API, с помощью которого программно можно взаимодействовать с ChatGPT.

Как раз последний подход был выбран для решения задачи, связанной с магистерской работой. Рассмотрим более подробно, как использовать ChatGPT программно на языке Python. Для этого потребуется:

- python 3.11;
- библиотека openai;
- наличие аккаунта OpenAI и сгенерированный API-ключ.

Структура программного запроса к ChatGPT показана в листинге 1.

Структура программного запроса ChatGPT

```

completion = openai.ChatCompletion.create(
model="<model_name>" ,
  messages=[
    { "role": "<role_name>", "content": "<some content>" }
  ]
)

```

У каждого запроса есть три основных параметра:

- model - тип используемой модели, на момент написания самая современная это gpt-3.5-turbo;
- content – само сообщение к нейросети;
- role – имеет три доступных варианта. В зависимости от указанной роли сообщение будет по-разному воспринято нейросетью.

Доступные роли описаны ниже:

- system - необязательная роль. Помогает задать нейросети модель поведения, что в текущем разговоре фокус будет на определенной теме, например к генерации вопросов к тексту;
- user – роль, показывающая, что сообщение пишет конечный пользователь, желающий получить ответ;
- assistant – роль, показывающая, что это был один из предыдущих ответов нейросети на запрос от роли user. С помощью этой роли можно ссылаться на предающие ответы нейросети для продолжения разговора.

Также есть некоторые дополнительные необязательные параметры:

- max-tokens – максимальное количество токенов, которые нейросеть может сгенерировать в качестве ответ;
- n – количество ответов, которые сгенерирует нейросеть;
- stream – флаг, определяющий, будет ли ответ послан полностью после генерации или же будет отсылаться частями.

На основе этого был создан программный модуль, который принимает на вход массив текста, задает к нему вопросы, генерирует ответы на эти вопросы и собирает это в JSON файл. Приведем пример входных данных для скрипта:

«Интернет-магазин "Лови Букет" предлагает свежие и красивые цветы, корзинки с подарками, дизайнерские букеты и неординарные композиции. Мы вкладываем все силы и душу в рабочий процесс, ведь для нас это не только бизнес, но и любимое дело. Каждый наш букет неповторим, ведь создается с уважением и любовью!»

Сформированные вопросы.

- Что предлагает интернет-магазин "Лови Букет"?
- Какие услуги являются главными для магазина?
- Как можно сделать заказ в интернет-магазине "Лови Букет"?
- Какие гарантии предоставляются покупателю интернет-магазина?
- Какие персональные параметры нужно указать при заказе букета?
- Какие уникальные композиции может предложить магазин?

Сформированные ответы на вопросы выше.

- Интернет-магазин "Лови Букет" предлагает услуги по заказу и доставке букетов цветов и других подарков.
- Главными услугами магазина являются заказ и доставка букетов цветов в Воронеже.
- Для заказа в интернет-магазине "Лови Букет" нужно выбрать нужный букет или композицию, указать адрес доставки и оформить заказ через сайт или по телефону.
- Покупателю магазина предоставляются гарантии качества цветов, оплаты и доставки.
- При заказе букета нужно указать его вид, размер, цвет и количество, а также адрес доставки и контактные данные получателя.
- Магазин может предложить уникальные композиции, созданные вручную мастерами, а также индивидуально оформленные букеты со смешанными цветами.

4. Обучение бота

Была разработана нейросеть, принимающая на вход файл для обучения, содержащий пары вопрос-ответ. Нейросеть в процессе обучения анализирует вопросы, содержащиеся в файле, и учится сопоставлять их с некоторым пользовательским вводом. После завершения обучения нейросеть готова принимать на вход пользовательский ввод, искать максимальное совпадение этого ввода с вопросами, содержащимися в файле и отдавать ответ, соответствующий максимально похожему на ввод вопросу. Простейший файл для обучения сети представлен в листинге 2.

Таким образом, после проведения обучения на входном файле вопросов и ответов, мы получаем готовую языковую модель, которая умеет анализировать входной текст, находить в файле максимально похожий вопрос и в качестве результата отправлять ответ на этот вопрос. Если интегрировать эту модель с мессенджером, то мы получаем готового чат-бота, способного анализировать

пользовательский ввод и способного подобрать максимально релевантный ответ на этот ввод.

Листинг 2

Пример файла для обучения бота

```
{ "intents": [
  {
    "tag": "greeting",
    "patterns": [
      "Здравствуйте"
    ],
    "responses": [
      "Здравствуйте, как я могу помочь?"
    ]
  },
  {
    "tag": "order",
    "patterns": [
      "Как можно сделать заказ в интернет-магазине \"Лови Букет\"?",
      "Как заказать?"
    ],
    "responses": [
      "Для заказа в интернет-магазине \"Лови Букет\" нужно выбрать нужный букет или композицию, указать адрес доставки и оформить заказ через сайт или по телефону."
    ]
  },
  {
    "tag": "goodbye",
    "patterns": [
      "Досвидания"
    ],
    "responses": [
      "До свидания, спасибо за визит",
      "Приятного дня!",
      "Пока! Возвращайтесь скорее."
    ]
  }
]
}
```

5. Интеграция сTelegram

Для реализации интеграции на языке Python была использована библиотека Telebot. Для интеграции бота нужен API-ключ бота в Telegram, предоставляемый клиентом. Пример работы бота в Telegram показан на рис. 2.



Рис. 2. Пример работы бота в Telegram

Заключение

В ходе данной работы был построен прототип платформы по созданию чат-ботов. В платформу в последствии будут добавляться новые фишки и вноситься улучшения. Например, увеличивать файл для обучения бота путем перефразирования вопросов.

Список литературы

1. Афанасьев Н.Г. Чат-боты и их место в бизнесе [Электронный ресурс]. – Режим доступа: <https://scienceforum.ru/2022/article/2018029603?ysclid=lgla3sjryk711594760>
2. Asking Right Questions: Training a T5 transformer Model for a New task [Электронный ресурс]. – Режим доступа: <https://towardsdatascience.com/asking-the-right-questions-training-a-t5-transformer-model-on-a-new-task-691ebba2d72c>
3. Transformersdocs [Электронный ресурс]. – Режим доступа: <https://huggingface.co/transformers/>

Разработка алгоритма распознавания элементов нотной грамоты с помощью кластерного анализа

Е. С. Макаренко

Студент магистратуры

В. В. Фертиков

Доцент

Введение

На данный момент цифровая обработка, анализ и распознавание изображений является интенсивно развивающимся направлением научной деятельности. Это объясняется высоким интересом к алгоритмам автоматического распознавания объектов, например, печатный текст, мелкие и крупные объекты на изображении, лица или ноты.

Так или иначе каждая задача имеет свое уникальное решение. Распознавание нот, не стало исключением. Исходя из темы магистерской диссертации основным элементом распознавания станет один (или совокупность) метод(-ов) кластерного анализа. Проблемами в работе станут в первую очередь графические артефакты, шумы, геометрическая сложность элементов и качество самого исходного изображения. Работа посвящена разработке алгоритма и сравнению его с аналогами, представленными на рынке.

1. Постановка задачи

Как уже было сказано ранее, процесс распознавания нот качественно отличается от, например, той же задачи распознавания печатного текста. Особенная геометрия, привязка элемента к координатам, множественные мелкие элементы сильно усложняют работу. Исходя из этого, была поставлена задача – создать алгоритм способный распознавать ноты в записях, попадающих под понятие нотной грамоты.

Данная работа будет включать в себя несколько этапов:

1. выбор и реализации метода нахождения образов
2. создание массива данных на основе матрицы расстояний
3. выбор и реализация метода кластеризации

4. определение оптимального числа кластеров и именование элементов
5. получение и анализ результатов проделанной работы

2. Поиск образов на изображении с помощью морфологического анализа

Морфологическая обработка изображений - это метод обработки изображений, который используется для изменения формы, размера и расположения объектов на изображении. Он основан на математической морфологии, которая использует теорию множеств для анализа формы и структуры объектов на изображении. Соответственно, и положение объекта такой анализ нам предоставляет [1].

Морфологическая обработка изображений включает в себя две основные операции - эрозию и дилатацию.

1. Эрозия - это операция, которая уменьшает размер объектов на изображении. Она основана на использовании структурного элемента, который скользит по изображению и удаляет пиксели, которые не соответствуют заданным критериям. Эрозия может использоваться для удаления шума и разрывов в объектах на изображении.

2. Дилатация - это операция, которая увеличивает размер объектов на изображении. Она также основана на использовании структурного элемента, который скользит по изображению и добавляет пиксели, которые соответствуют заданным критериям. Дилатация может использоваться для объединения разрозненных объектов на изображении.

Извлечение информации происходит в несколько этапов:

1. чтение исходного изображения и преобразования его в оттенки серого;
2. измерение расстояния между горизонтальными линиями на изображении путем создания структурного элемента, который скользит по изображению и подсчитывает количество пикселей;
3. извлечение простых графических элементов на изображении, таких как горизонтальные и вертикальные линии, наклонные линии и короткие вертикальные линии;
4. удаление элементов из бинарного изображения, таких как горизонтальные линии и обозначения, кроме коротких вертикальных линий;
5. обработка связных компонент на изображении для нахождения ограничивающих рамок для каждого элемента.

На рисунке представлен пример найденных искомым элементов.

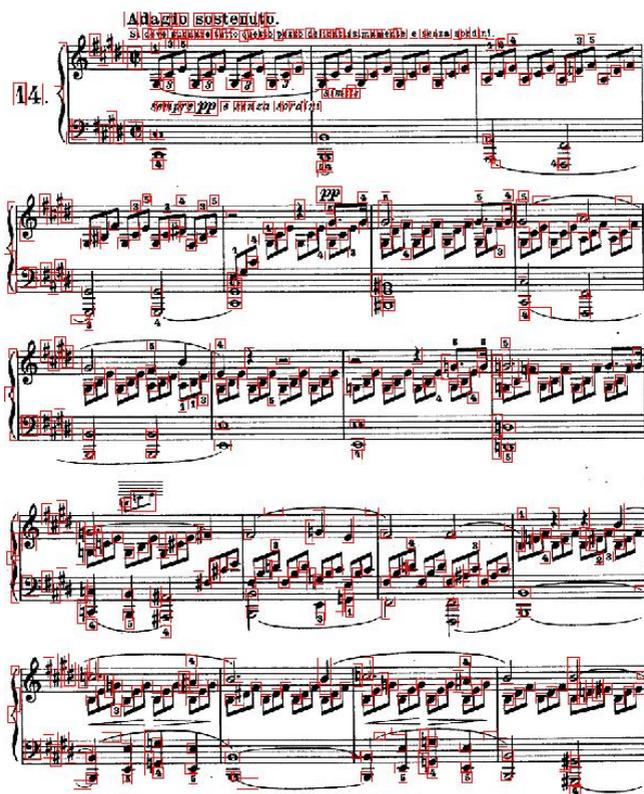


Рисунок. Морфологический анализ изображения

3. Подготовка матрицы сходства и кластеризация

Подготовка матрицы сходства подразумевает под собой следующий порядок действий:

1. вырезанием маленьких изображений из основного по записанным координатам;
2. приведение изображений к единому размеру;
3. перевод изображений в оттенки серого;
4. попарное сравнение изображений методом двумерной взаимной корреляции.

Полученная матрица используется как набор входных данных для кластерного анализа.

Исходя из специфики задачи, было решено использовать в качестве алгоритма кластеризации иерархическое агломеративное кластерное древо [2].

После получения результатов кластеризации остается оценить результат и присвоить вручную имена каждому кластеру. Как говорилось ранее, множество одинаковых элементов можно трактовать различным образом. Для верного распознавания после кластеризации необходимо будет проверять положение элемента на координатной оси, после чего уже окончательно именовать.

Заключение

Данная статья посвящена описанию процесса разработки и тестирования алгоритма распознавания элементов нотной грамоты. Кратко описаны структурные части целого алгоритмы, такие как морфологический анализ, подготовка к кластеризации и непосредственная кластеризация. Результатом работы стал рабочий программный модуль способный распознать искомые элементы. В дальнейшем предполагается применение алгоритма к решению прикладных задач таких как проигрывание композиции по полученным нотам.

Список литературы

1. Пытьев Ю.П. Методы морфологического анализа изображений / Ю.П. Пытьев, А.И. Чуличков М.: Физматлит, 2010 343 с.
2. Гонсалес Р. Цифровая обработка изображений / Р. Гонсалес, Р. Вудс. – М. : Техносфера, 2005. – 1072 с.

Автоматическая оценка дефектов на основе статистических данных однотипных проектов.

С. С. Масленникова

Студент магистратуры

В. В. Коротков

Старший преподаватель

Введение

Оценка времени, затраченного на выполнение задач при разработке проектов, является важным аспектом планирования проекта. Чем точнее оценка, тем лучше происходит управление ресурсами, определение сроков завершения проекта и улучшается процесс разработки. Также она помогает управлять ожиданиями заказчика и команды разработчиков. Несоответствие между ожиданиями и реальностью может привести к задержкам в проекте, неудовлетворенности заказчика и команды разработчиков.

Одной из основных проблем при оценке времени на выполнение задачи является необходимость сбора команды разработчиков и привлечение опытных коллег, а также затраты времени на анализ каждой задачи. Однако по мере разработки появляются и накапливаются дефекты, которые тоже требуют оценки времени на выполнение. В результате команде требуется планировать время выполнения не только сложных и больших задач, но и относительно простых и однотипных. Таким образом, оценку большого количества дефектов можно сократить за счет автоматизации. Данная работа посвящена обзору различных методов оценки времени на выполнение задачи для последующей разработки и программной реализации модели машинного обучения для оценки часов, затраченных на исправление дефектов, и сервиса для ее интеграции с инструментами управления проектами.

1. Agile-разработка, основные понятия

Agile-разработка программного обеспечения — это современный подход к созданию программного обеспечения, который ориентирован на итеративный и постепенный прогресс. Он позволяет командам быстро и эффективно реагировать на меняющиеся требования, создавая качественные продукты с максимальной эффективностью. Эта

методология позволяет непрерывно получать обратную связь от клиента, что ведет к повышению качества и эффективности продукта. Преимущества agile-разработки очевидны: сокращение сроков, улучшение результатов и повышение удовлетворенности клиентов делают ее привлекательной для многих компаний сегодня.

Рассмотрим несколько терминов, которые будут использоваться в дальнейшей работе.

Тикет – это термин, используемый в разработке ПО для обозначения задачи, которую необходимо выполнить. Он может содержать информацию о задаче, ее приоритете, сроке выполнения и ответственном за выполнение.

Дефект – это ошибка в программном обеспечении, которая приводит к некорректной работе приложения. Дефекты могут возникать из-за ошибок в коде, неправильного проектирования или недостаточного тестирования.

Бэклог – это список задач, которые необходимо выполнить в рамках проекта.

Спринт – это итерация в agile, которая длится обычно от одной до четырех недель и включает в себя планирование, выполнение и обзор работ. В начале спринта команда планирует задачи, которые она сможет выполнить за этот период, а в конце проводит обзор проделанной работы.

Сторипойнт – это условная единица, используемая в agile-методологиях разработки ПО для оценки сложности задачи. Она не соотносится с конкретным количеством времени, а используется для оценки сложности задачи относительно других задач в бэклоге. Оценка сторипойнтов производится командой разработчиков на основе их опыта и знаний о проекте.

2. Методы оценки времени в гибких методологиях разработки

Рассмотрим несколько популярных техник для оценивания времени, которые применяются в agile-разработке [1].

Planning poker – это техника командной оценки задач в относительных единицах. Оценка сложности производится с помощью цифровых карт, которые содержат числа Фибоначчи или другие числа. В начале оценки каждый участник команды получает колоду карт и выбирает карту, которая соответствует сложности задачи. Затем участники обсуждают свои оценки и повторяют процесс до тех пор, пока не будет достигнуто согласие.

T-shirt sizing – это техника относительной оценки времени на основе относительной сложности задач. В этой технике задачи оцениваются с помощью размеров футболок, таких как XS, S, M, L, XL,

где каждый размер соответствует определенному диапазону времени или сложности. Например, XS может соответствовать задачам, которые можно выполнить за несколько минут, а XL может соответствовать задачам, которые займут несколько дней.

Affinity mapping system – это метод, который используется в agile-разработке для оценки задач. Этот метод помогает команде организовать большое количество информации в ключевые темы, которые могут помочь команде увидеть общую картину. Оценка задач с помощью affinity mapping system состоит из нескольких этапов. Команда проводит сессию мозгового штурма, после чего собирает все идеи и организовывает их в группы, которые отражают общие темы. Этот метод может быть полезен для оценки задач, которые требуют коллективного решения и организации большого количества информации.

Итак, после рассмотрения нескольких методов, можно заметить, что все они предполагают оценку в относительных единицах времени, а также требуют участия команды [2]. Также все эти методы предполагают более точную оценку для небольших тикетов и менее точную для крупных.

3. Функция ранжирования текстов BM25

Перейдем к рассмотрению методов машинного обучения, которые могут быть использованы для оценки дефектов. Как правило, тикет содержит текстовое описание. В этом случае можно предположить, что выполнение тикетов с похожим описанием занимает примерно одинаковое время. Тогда задачу оценивания времени можно свести к поиску похожих тикетов по их текстам и усреднению времени их выполнения.

Алгоритм BM25 – это алгоритм ранжирования документов, который решает задачу ранжирования документов по их релевантности поисковому запросу в информационном поиске [3]. Он основан на модели пространства векторов и использует меру TF-IDF для оценки важности терминов в документе. Алгоритм BM25 учитывает не только частоту термина в документе, но и его частоту в коллекции документов, а также длину документа. Он используется в поисковых системах, таких как Google и Яндекс, а также в других приложениях, связанных с обработкой естественного языка.

В алгоритме BM25 ранжирование документов зависит от нескольких параметров.

$$score(D, Q) = \sum_{i=1}^n IDF(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{avgdl}\right)} \quad (1)$$

$$IDF(q_i) = \ln \left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} \right) \quad (2)$$

Основными параметрами являются:

- $f(q_i, D)$ – количество раз, когда слово q_i встретилось в документе D , чем чаще слово встречается в документе, тем выше его вес;
- $|D|$ – длина документа D в словах, чем длиннее документ, тем меньше его вес;
- $avgdl$ – средняя длина документа в коллекции;
- k_1 и b – свободные коэффициенты, обычно $k_1 \in [1.2, 2.0]$ и $b = 0.75$;
- N – количество документов в коллекции, чем больше документов в коллекции, тем меньше вес каждого документа;
- IDF – частота употребления слова в коллекции, чем чаще слово встречается в коллекции, тем меньше его вес;
- $n(q_i)$ количество документов, содержащих q_i .

Параметр b определяет, насколько сильно длина документа влияет на его релевантность. Чем выше значение параметра b , тем больше вес у коротких документов. Параметр b используется для учета того, что длинные документы могут содержать много повторяющихся слов, которые не являются релевантными для запроса.

Параметр k_1 определяет, насколько много весят частоты слов в документе. Чем выше значение параметра k_1 , тем больше вес у частот слов в документе. Параметр k_1 позволяет увеличить влияние частот слов в документе на его релевантность, что может улучшить качество ранжирования.

4. Регрессия Хьюбера

Альтернативный подход к определению времени – это решение задачи регрессии по признакам, извлеченным из текста. Для того, чтобы модель была более устойчивой к отклонениям, можно использовать модель, устойчивую к выбросам. Одной из них является регрессия Хьюбера [4].

Функция Хьюбера – это функция потерь, используемая в устойчивой регрессии, которая менее чувствительна к выбросам, чем квадратичная ошибка. Для построения линейной регрессии требуется минимизировать выражение относительно переменной β .

$$\sum_{i=1}^m \phi\left(y_i - x_i^T \beta\right) \xrightarrow{\beta} \min \quad (3)$$

$$\phi(u) = \begin{cases} u^2, & |u| \leq M \\ 2Mu - M^2, & |u| > M \end{cases} \quad (4)$$

Здесь M – это положительно определенный порог, определяющий квадратичную или абсолютную функцию потерь использовать.

Квадратичная функция потерь увеличивает влияние выбросов, так как она учитывает квадрат разности между предсказанным и фактическим значением. Абсолютная функция потерь, напротив, учитывает только абсолютную разность между предсказанным и фактическим значением, что делает ее менее чувствительной к выбросам. Использование регрессии Хьюбера позволит балансировать между этими двумя функциями потерь, получая при этом более точные значения для небольших тикетов и не так сильно штрафовать модель за ошибки на дефектах, занимающих много времени.

5. Применение глубокого обучения для оценки времени

Рассмотрим еще один подход, который был представлен в статье «A deep learning model for estimating story points» [5]. Авторы разработали модель с помощью глубокого обучения для оценки времени в сторипойнтах. Архитектура модели представлена на рисунке.

Подход, описанный в статье, включает в себя несколько этапов:

- получение векторных представлений (эмбеддингов) слов;
- получение предварительного векторного представления документа с помощью LSTM-слоя;
- получение окончательного векторного представления текста с помощью Recurrent Highway Network (RHWN);
- получение оценки в story-points с помощью регрессора.

Модель является модифицированной версией архитектуры LSTM. Несмотря на то, что модель оценивает время в сторипойнтах, а не в часах, она опирается только на текстовые данные из тикета. Таким образом, можно использовать подход и архитектуру, предложенную авторами статьи, чтобы обучить модель делать предсказания в часах.

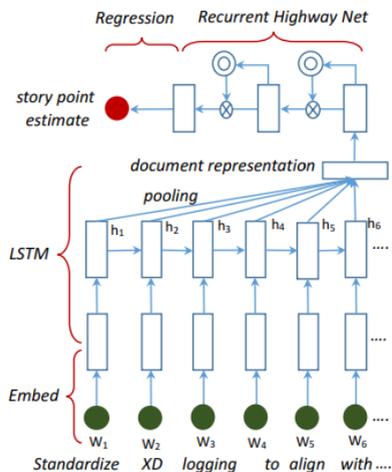


Рисунок. Архитектура модели

Заключение

Итак, в статье были рассмотрены различные методы оценки времени, которое займет выполнение задачи. Были рассмотрены алгоритмы машинного обучения, которые могут быть эффективными при оценивании задач и которые в дальнейшем могут быть использованы для разработки модели по оцениванию дефектов.

Список литературы

1. Top 5 User Story Estimation Techniques [Электронный ресурс] : Электрон. журн. – Режим доступа : <https://www.7pace.com/blog/user-story-estimation-techniques>
2. Story points and estimation [Электронный ресурс] : Электрон. журн. – Режим доступа : <https://www.atlassian.com/agile/project-management/estimation>
3. Robertson, S. Okapi at TREC-3 / S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, M. Gatford. – 1994.
4. Huber, P. J. Robust Estimation of a Location Parameter / P. J. Huber // The Annals of Mathematical Statistics, Ann. Math. Statist. – 1964. – Vol. 35. – Issue 1. – P. 73-101.
5. Choetkiertikul M. A Deep Learning Model for Estimating Story Points / M. Choetkiertikul, H. K. Dam, T. Tran, T. Pham, A. Ghose, T. Menzies // IEEE Transactions on Software Engineering. – 2019. – Vol. 45. – Issue 7. – P. 637-656.

Автоматическое построение таксономии текста

Е. В. Муравьева

Студент магистратуры

В. В. Гаршина

Доцент

Введение

Важная задача обработки естественного языка (Natural language processing, NLP) - автоматическое понимание и извлечение знаний из текстов TextMining. Задачи TextMining заключаются в выделении понятий (именованных сущностей), а так же смысловых и структурных отношений между ними, формирующих семантическое представление текста. В связи с этим, актуальным является представление текстовой информации в формальном и машинно-читаемом формате, отражающем таксономию текста: семантические связи между понятиями предметной области и формирующие смысл текста в целом.

1. Проблема построения таксономии текста

Автоматическое построение таксономии текста является одним из важных инструментов в области обработки естественного языка. Эта технология позволяет автоматически классифицировать тексты по тематикам и создавать иерархическую структуру, которая отражает связи между ними. Процесс построения таксономии текста начинается с анализа большого количества текстовых документов. Алгоритмы машинного обучения используются для выявления ключевых слов и фраз, которые наиболее часто встречаются в текстах. Затем эти ключевые слова и фразы группируются в кластеры, которые представляют собой темы или категории. Для построения иерархической структуры таксономии используется алгоритм дерева решений. Каждый кластер становится узлом дерева, который может иметь несколько подузлов или листьев. Подузлы представляют собой более конкретные темы, связанные с родительским узлом, а листья – конкретные тексты, относящиеся к соответствующей теме. Автоматическое построение таксономии текста имеет множество применений. Оно может быть использовано для классификации новостных статей, блогов, отзывов и других типов текстов. Это

позволяет быстро и эффективно организовывать большие объемы информации и находить нужные данные. Кроме того, автоматическая таксономия текста может быть использована для создания интеллектуальных систем поиска. Такие системы могут быстро находить нужные документы и информацию, используя ключевые слова и категории. Однако необходимо учитывать, что автоматическая таксономия текста не всегда является точной. Алгоритмы машинного обучения могут ошибаться при выявлении ключевых слов и фраз, а также при группировке их в кластеры. Поэтому необходимо проводить регулярную экспертную проверку и корректировку таксономии, чтобы обеспечить ее точность и актуальность. В целом, автоматическое построение таксономии текста является важным инструментом для эффективной обработки больших объемов информации. Оно может быть использовано в различных областях, где необходимо быстро и точно классифицировать тексты и организовывать информацию.

2. Особенности формирования лексических таксономий

Обсудим основные понятия, которые непосредственно связаны с автоматическим построением таксономии текста. Неформализованный текст усложняет алгоритмы анализа, поэтому важно знать основные принципы построения высказываний ЕЯ, особенности обработки лексики и синтаксиса.

Граф знаний является формальным способом представления знаний и связей между ними в машинно-читаемом формате. Он представляет собой сеть узлов (сущностей) и связей между ними, которые отражают отношения между понятиями и фактами в определенной предметной области. Граф знаний используется для хранения и обработки данных, а также для автоматического извлечения знаний из текстов на естественном языке. Он является важным инструментом в области искусственного интеллекта и машинного обучения. При этом необходимо отметить, что определение графов знаний различаются и в некоторых исследованиях говорится о том, что граф знаний не отличается от онтологии и базы знаний.

Следует также отметить, что для структуризации слов (лексики) и для формализации структуры текста важную роль играют гиперонимы, гипонимы и синонимы. Они содержатся в терминологических онтологиях и показывают информацию о терминах естественного языка и связях между ними. Гипероним (в лингвистике) — понятие в отношении к другому понятию, выражающее более общую сущность. В отношении некоторого множества объектов гиперонимом является понятие, отражающее надмножество к исходному. Термин «зверь» является гиперонимом по отношению к термину «собака», а термин

«собака» в свою очередь — гипероним по отношению к термину «бульдог». И наоборот — термин «бульдог» — гипоним по отношению к термину «собака», а «собака» — гипоним по отношению к термину «зверь». Гиперонимы и гипонимы играют важную роль для определения аерархии классов в графе знаний, так как они позволяют организовать информацию в логическую структуру и упростить ее обработку. Например, при поиске информации о фруктах, система может использовать гипероним "фрукты", чтобы найти все связанные с этой темой гипонимы. Также гиперонимы и гипонимы могут быть использованы для автоматической классификации текстов на основе содержания.

Исследования показывают, что автоматическое формирование таксономии предметной области и программная реализация алгоритмов по формированию классов сущностей на основе выделения синонимов, гипонимов и гиперонимов являются эффективными подходами [1,2].

3. Обзор подходов и алгоритмов автоматического формирования таксономии предметной области текста

Рассмотрим некоторые подходы и алгоритмы, используемые для построения таксономии текстов, которые часто используются в современных исследованиях.

Первым рассмотрим проект WordNet [3], который является лексической базой данных. Он содержит семантические отношения между словами английского языка. Он использует гиперонимы и гипонимы для создания семантической сети, которая может быть использована для классификации текстов. Алгоритмы WordNet позволяют определять связи между словами, такие как синонимы, антонимы, гиперонимы, гипонимы, а также ассоциации между словами. Это позволяет улучшить точность классификации текстов путем использования более точных определений слов и их значений.

Примерами семантических таксономий для естественного языка являются международные базы знаний, включающие русскоязычный контент: DBpedia [4], Wikidata [5] и Freebase [6].

Для формирования семантических связей и отношений между словами текста применяется достаточно много алгоритмов.

Нами был рассмотрен алгоритм LSA (Latent Semantic Analysis). Он является методом анализа текстов, который использует матричные вычисления для определения семантических отношений между словами. Он основывается на предположении, что слова, которые часто встречаются вместе в текстах, имеют более сильную связь друг с другом, чем слова, которые редко встречаются вместе.

Для этого алгоритм LSA использует матрицу терминов-документов, которая содержит информацию о том, какие слова встречаются в каких документах. Затем происходит снижение размерности этой матрицы, чтобы уменьшить количество признаков и сделать ее более компактной. После этого происходит вычисление сингулярного разложения матрицы, которое позволяет определить скрытые семантические отношения между словами. Например, если слова "кошка" и "собака" часто встречаются вместе, то они будут иметь более близкое расстояние друг к другу в новом пространстве признаков, которое создается после применения алгоритма LSA.

Алгоритм LSA может быть использован для классификации текстов, поиска похожих документов и создания рекомендаций. Он также может быть комбинирован с другими алгоритмами, такими как WordNet, для улучшения точности классификации и поиска.

В исследовании публикаций по данной тематике можно указать частое использование Алгоритма SVM (Support Vector Machines). Он основан на методах машинного обучения и используется для классификации данных. SVM основывается на поиске гиперплоскости, которая разделяет данные на два класса. Гиперплоскость выбирается таким образом, чтобы максимизировать расстояние между ней и ближайшими точками каждого класса, которые называются опорными векторами. Алгоритм SVM может быть использован для классификации текстов, где каждый документ представлен в виде вектора признаков, например, количество вхождений каждого слова в документе. SVM может обучаться на размеченных данных, где каждый документ имеет метку класса, или на неразмеченных данных, где алгоритм пытается найти структуру в данных самостоятельно. Одним из преимуществ SVM является его способность работать с высоко размерными данными и находить сложные нелинейные зависимости между признаками. Он также имеет хорошую обобщающую способность и может работать с небольшими объемами данных. Алгоритм SVM может быть комбинирован с другими методами обработки текста, такими как TF-IDF или LSA, для улучшения точности классификации. Он также может быть использован для ранжирования документов по их релевантности в поисковых системах.

Стоит еще упомянуть алгоритм KNN (K-Nearest Neighbors), который использует близость между текстами для классификации новых текстов на основе ближайших соседей в обучающем наборе.

Можно выделить основные этапы обработки текста для формирования таксономии по его содержанию.

1. Извлечение ключевых слов: для того чтобы выделить наиболее значимые слова, используются методы статистического анализа и машинного обучения. Для этого используются такие алгоритмы, как частотный анализ, tf-idf (term frequency-inverse document frequency), мультимодельная латентно-семантическая индексация.

2. Затем, решается проблема выделения похожих слов и синонимов. Так как в текстах могут использоваться разные слова, которые имеют схожее значение, то необходимо учитывать их в процессе анализа. Для этого применяются методы семантического анализа, такие как латентное размещение Дирихле (LDA), word2vec и fastText. Проект демонстрирует выявление сходства смысла между выделенными понятиями предметной области RusVectōrēs [7].

3. Следующим шагом является кластеризация: чтобы выделить различные категории и подкатегории в тексте, производится кластеризация ключевых слов и схожих слов. Для этого применяются алгоритмы кластерного анализа, например, k-средних, DBSCAN (density-based spatial clustering of applications with noise) и иерархическая кластеризация.

4. Извлечение отношений между терминами: для того чтобы определить иерархию таксономии, необходимо выделить отношения между терминами. Для этого используются методы ассоциативного анализа, такие как ассоциативные правила и логистическая регрессия [8].

5. Последним этапом выполняется визуализация: для наглядного представления результатов анализа, производится визуализация таксономии. Для этого используются алгоритмы визуализации данных, например, диаграммы Венна, деревья и сетевые графики.

Анализ текста и построение моделей классификации могут помочь оценить эталонные и субъективные знания о предметной области.

Заключение

Данная статья посвящена исследованию подходов и алгоритмов, применяемых для построения таксономий текстов с целью формализации содержания и семантической обработки. Описаны основные языковые индикаторы - лексические характеристики, которые могут быть использованы для решения данной задачи.

Список литературы

1. Автоматическая обработка текстов на естественном языке и анализ данных / Е. И. Большакова [и др.]. – М. : Изд-во НИУ ВШЭ, 2017. – 269 с.

2. Польщикова, О. Н. Гиперо-гипонимические отношения в терминологии компьютерной лингвистики / О. Н. Польщикова // Филологические науки. Вопросы теории и практики. – 2022. – Т. 15. – № 6. – С. 2001-2005.
3. Проект Wordnet [Электронный ресурс] : лексическая база данных для английского языка. – Режим доступа: <https://wordnet.princeton.edu>
4. Проект dbpedia [Электронный ресурс] : графы знаний – Режим доступа: <https://www.dbpedia.org>
5. Проект WikiData [Электронный ресурс] : база знаний – Режим доступа: <https://www.wikidata.org>
6. Проект FreeBase [Электронный ресурс] : база данных – Режим доступа: <https://developers.google.com/freebase>
7. Проект RusVectōrēs: семантические модели для русского языка [Электронный ресурс]. – Режим доступа: <https://rusvectors.org/ru>
8. Тихомиров, М. М. Разработка автоматизированной системы пополнения таксономии на текстах конкретной предметной области / М. М. Тихомиров // Интеллектуальные системы. Теория и приложения. – 2021. – Т. 25. – № 4. – С. 250-254.

Разработка сайта для моделирования искажения акустического сигнала в мелком море

В.С. Назаренко

Студент магистратуры

А.Ю. Малыхин

Старший преподаватель

Введение

В наши дни особенно важны исследования в области акустики, изучающие водные просторы вблизи берега. Данный выбор обусловлен огромным хозяйственным и техническим значением данной части мирового океана для человечества. Изучение именно мелкого моря полезно для рыболовства, нефтедобывающей промышленности, для построения подводных сетей и многих других сфер деятельности человека. Чем больше известно закономерностей воздействия среды на сигнал, тем более точно можно изучить саму среду.

1. Постановка задачи

Целью данной статьи является создание сайта, который позволит пользователю моделировать искажение акустического сигнала при передаче в мелком море, и развёртывание этого сайта. Он позволит пользователю изучить особенности распространения сигнала в океанической среде и воздействие самой среды на сигнал в зависимости от её характеристик.

Средой разработки был выбран Matlab, так как в нем содержатся все необходимые инструменты для реализации выбранной программы, а средой для развёртывания сайта Matlab Web App Server.

2. Алгоритм реализации модового подхода

В данном методе акустический сигнал представляется, как сумма вертикальных мод с амплитудами, медленно меняющимися в горизонтальной плоскости. Мы будем одновременно учитывать и дисперсию, и горизонтальную рефракцию звука. Для моделирования сигнала, приходящего на приёмник, нам необходимо вычислить горизонтальное волновое число q_l (l – номер моды) по формуле[1]:

$$q_l = \sqrt{\frac{(2\pi f)^2}{c^2} - \frac{\pi^2}{H^2} \left(l - \frac{1}{2}\right)^2}, \quad (1)$$

где f - частота (Гц), c - скорость звука (м/с), H - глубина моря (м).

Для нахождения сигналов, пришедших на приемник, необходимо разложить сигнал на гармонические составляющие (получить его спектр) с помощью преобразования Фурье, которое мы будем вычислять по формуле [2]:

$$S(\omega) = \int_{-\infty}^{\infty} s(t)e^{-i\omega t} dt, \quad (2)$$

где $S(\omega)$ - спектр сигнала, $s(t)$ - излучаемый сигнал, ω - угловая частота (Гц), t - время (с). Для нахождения сигналов, соответствующих различным модам, нам необходимо искать фазовые набеги $\varphi_l(\omega)$, считающиеся по формуле. Так как мы рассматриваем сигнал, на который действует не только дисперсия, но и горизонтальная рефракция, нам необходимо учитывать изменение горизонтального волнового числа и фазовых набегов в зависимости от изменения траектории распространения сигнала. Фазовые набеги мы считаем, как криволинейный интеграл вдоль горизонтального луча от источника до приёмника звука:

$$\varphi_l(\omega) = \int_S^R \vec{q}_l(\omega, r) ds, \quad (3)$$

где ω - угловая частота сигнала (Гц), R и S - точки, где находятся приёмник и излучатель звукового сигнала, q_l - волновое число l -й моды на выбранном участке, r - радиус-вектор в горизонтальной плоскости. Для получения спектра сигнала, пришедшего на приемник $S'(\omega)$, мы будем умножать спектр исходного сигнала на $e^{-i\varphi_l(\omega)}$:

$$S'(\omega) = S(\omega)e^{-i\varphi_l(\omega)}, \quad (4)$$

Используя обратное преобразование Фурье, можно получить сигнал, зная его спектр:

$$s'(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} S'(\omega)e^{i\omega t} d\omega, \quad (5)$$

Мы хотим учитывать в программе и эффект горизонтальной рефракции, поэтому используемый нами метод вертикальных мод и

горизонтальных лучей учитывает явление горизонтальной рефракции акустических мод [3].

3. Создание и развёртывание сайта

Для проведения расчётов с использованием модового подхода возьмем сигнал, который будет длиться 0,5 секунды. На примере этой звуковой волны мы сможем наглядно увидеть изменение сигнала в зависимости от частоты и номера моды сигнала. Сигнал будет задаваться формулой:

$$s(t) = \sin(2\pi ft) \quad (6)$$

где f - фиксированная частота сигнала (Гц), t - время излучения сигнала (с). Мы будем проводить моделирование ситуации типичной для мелкого моря, когда глубина при отдалении от берега линейно возрастает. Модель берегового клина представлена на рис. 1.

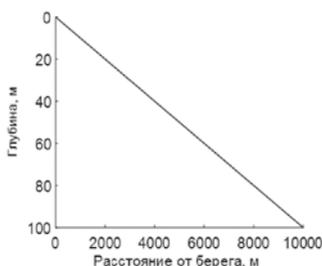


Рис. 1. Модель берегового клина

Так как нам было необходимо создать сайт, способный производить вычисления в среде Matlab, мы воспользовались средствами Matlab Compiler и создали сайт на основе смоделированного в Matlab App Designer интерфейса. На рис. 2 представлен интерфейс сайта.

На рис. 2 можно увидеть траектории распространения сигналов от источника до приёмника. В данном случае приёмник располагается в двухлучевой области, куда может попасть сигнал не только по прямому лучу (испытывающему незначительную горизонтальную рефракцию), но и по рефрагированному (подходящему близко к берегу и под действием рефракции существенно меняющему свою траекторию). С помощью данного графика пользователь может понять, дойдёт ли сигнал до приёмника, и сколько лучей сможет на него попасть: один, два, или звук не попадёт в эту область (зона акустической тени).

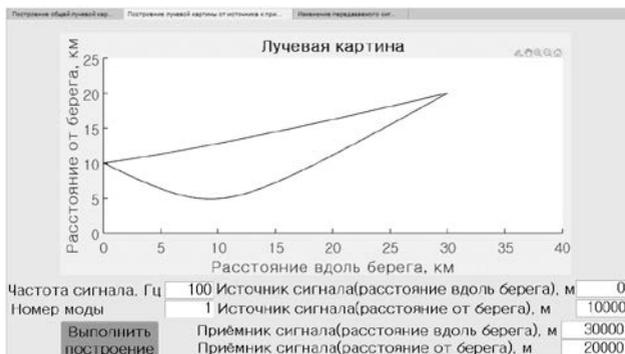


Рис. 2. Графики первой моды сигналов, распространяющихся по прямому и рефрагированному лучу с частотой 100 Гц

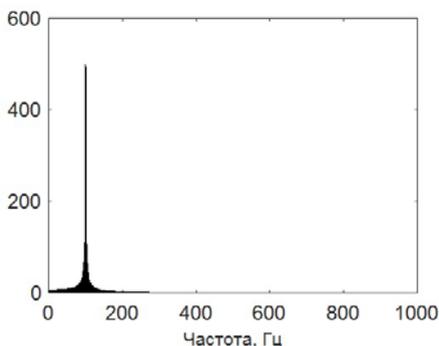


Рис. 3. Спектр сигнала, используемого при моделировании. Частота 100 Гц, длительность 0,5 с.

На рис. 3 представлен график спектра сигнала. Так как мы рассматриваем конечную синусоиду, на графике спектра можно увидеть, что сигнал может быть представлен, как сумма более чем одной гармонической волны. Таким образом, в спектре сигнала присутствуют значимые частоты, отличные от основной. Далее мы рассмотрим интерфейс сайта, отображающий передачу и искажение самого сигнала, представленный на рисунке 4. Из него видно, что сигнал, пришедший на приёмник по прямому лучу, дошёл раньше, чем сигнал, идущий по рефрагированной траектории, прошедший большее расстояние до приемника. Сигналы исказились, потому что разные частоты их спектра

распространялись с разной скоростью и пришли в разное время. Сигнал, идущий по рефрагированной траектории, больше исказился.

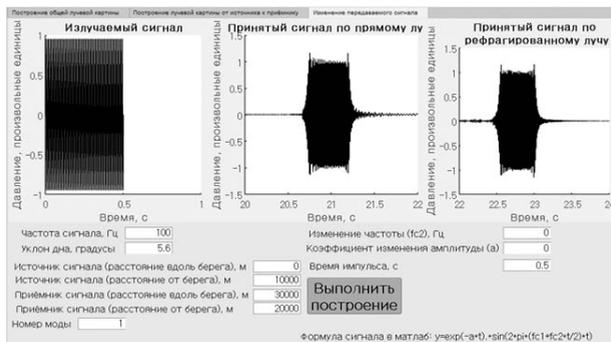


Рис. 4. Графики первой моды сигналов, распространяющихся по прямому и рефрагированному лучу с частотой 100 Гц

На рис. 5 приведён вывод сайта для второй моды того же сигнала. Если сравнивать рис. 4 и рис. 5, видно, что с увеличением номера моды растёт и искажение сигнала. Длительность пришедшего сигнала возросла. Также оба сигнала пришли на приёмник немного позже для второй моды, чем для первой.

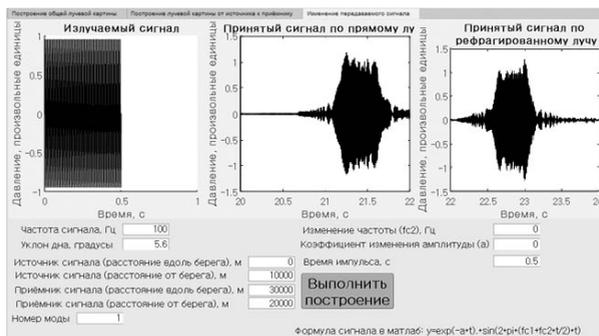


Рис. 5. Графики второй моды сигналов, распространяющихся по прямому и рефрагированному лучу с частотой 100 Гц

Таким образом, можно сделать вывод о том, что по прямому лучу сигнал доходит до приёмника раньше, чем по рефрагированному.

Заключение

Был создан сайт, моделирующий распространение звукового сигнала в мелком море. Он был написан на языке Matlab и позволяет построить графики, пришедшего на приемник сигнала для различных мод. Сайт позволяет пользователю самостоятельно выбирать тип сигнала, номер моды, его частоту и длительность, а также задавать параметры среды: уклон дна, координаты источника и приёмника сигнала.

Список литературы

1. Кацнельсон, Б.Г. Акустика мелкого моря. / Б.Г. Кацнельсон, В.Г Петников – М.: Наука, 1997. – 191 с.
2. Кацнельсон, Б.Г. Горизонтальная рефракция в мелком море и ее экспериментальные наблюдения / Б.Г. Кацнельсон, М. Бади, Дж. Линч // Акустический журнал. – 2007. – Т.53. №3. – С. 362-376.
3. Бреховских, Л. М., Лысанов, Ю. П. Теоретические основы акустики океана. / Л М. Бреховских, Ю.П. Лысанов – М.: Наука, 2007. – 370 с.

Моделирование в пакете MATLAB передачи акустических сигналов в океаническом волноводе

Е. И. Олейник

Студент магистратуры

А. Ю. Малыхин

Старший преподаватель

Введение

Практический интерес к изучению акустических волн и особенностям их распространения в водной среде обусловлен тем, что это единственный тип волн известной физической природы, способный распространяться под водой на расстояние многих сотен километров [1]. А так как океанский шельф важен для жизнедеятельности человека (освоение нефтегазовых ресурсов, возведение прибрежной инфраструктуры, мониторинг гидродинамических возмущений и т.д.), он интересен для исследования учёных и инженеров самых разных специальностей, а разработка ПО для помощи этим специалистам не теряет актуальности.

Целью данной работы является создание веб-приложения с помощью инструментария среды Matlab, способного моделировать искажение акустических сигналов при распространении в мелком море. В основные задачи проекта входит реализация программного модуля для расчёта горизонтальных волновых чисел, реализация модуля расчёта принятого сигнала, тестирование работы приложения при различных параметрах исходного сигнала и параметров среды, а также реализация пользовательского интерфейса.

1. Модель волновода и параметры исходного сигнала

Поле точечного акустического источника в океаническом волноводе можно описать с помощью модового подхода. Его суть состоит в том, что звуковую волну можно представить в виде суперпозиции мод, распространяющихся в горизонтальном направлении независимо друг от друга [2].

В нашем случае в математической модели акустического волновода ключевую роль играет понятие горизонтального волнового числа,

выражающего скорость изменения фазы волны для конкретной моды и частоты в зависимости от пространственной координаты. Далее приведена формула вычисления горизонтального волнового числа для l-ой моды. Эта приближённая формула применяется для волновода Пекериса - волновода, в котором скорость звука с глубиной не меняется.

$$q_l = \sqrt{\frac{(2\pi f)^2}{c^2} - \frac{\pi^2}{H^2} \left(l - \frac{1}{2}\right)^2}, \quad (1)$$

где H - глубина моря в метрах, f - частота исходного сигнала в Гц, c – скорость звука в воде в м/с.

Но так как разработанное приложение должно учитывать изменение скорости звука в воде с глубиной, для расчёта горизонтальных волновых чисел мод необходимо решать задачу Штурма-Лиувилля. Необходимо численно решить дифференциальное уравнение второго порядка с двумя граничными условиями (для поверхности волновода и его дна соответственно) [3]. Плотность водной среды и плотность дна будут считаться постоянными.

Само дифференциальное уравнение для нахождения горизонтальных волновых чисел выглядит следующим образом:

$$\frac{d^2\psi_l(z)}{dz^2} + [k^2(z) - q_l^2]\psi_l(z) = 0, \quad (2)$$

Граничные условия выглядят следующим образом:

$$\psi_l(z)_{z=0} = 0, \quad (3)$$

$$\left[\psi_l(z) + g(q_l) \frac{d\psi_l(z)}{dz} \right]_{z=H} = 0 \quad (4)$$

где
$$g(q) = \frac{m_1}{\sqrt{q^2 - k_1^2}}, \quad m_1 = \frac{\rho_1}{\rho}, \quad k(z) = \frac{2\pi f}{c(z)}.$$

Здесь $c(z)$ – это скорость звука в воде на заданной глубине, м/с; ρ – плотность водного слоя, кг/м³; ρ_1 – плотность грунта на дне, кг/м³; q – горизонтальное волновое число, м⁻¹.

2. Предлагаемый подход

Алгоритм решения задачи Штурма-Лиувилля на практике кратко приведён на блок-схеме на рис.1. Для заданного профиля скорости звука с определённым шагом перебираются горизонтальные волновые числа q и подставляются в дифференциальное уравнение (2). Далее уравнение численно решается, и если для текущего q граничные условия

соблюдаются, значит выбранное значение и есть искомое горизонтальное волновое число, иначе перебор продолжается.



Рис. 1. Блок-схема алгоритма решения задачи Штурма-Лиувилля

3. Сравнительный анализ результатов работы программы

В рассматриваемом далее примере для моделирования исходного сигнала используются коэффициент изменения амплитуды (a) и частота (f), имеющие следующие значения: $a = 10 \text{ с}^{-1}$, $f = 100 \text{ Гц}$. Длительность сигнала составляет $0,5 \text{ с}$. Формула исходного синусоидального сигнала:

$$y = e^{-at} \sin(2\pi ft) \quad (5)$$

Для последующих расчётов горизонтальных волновых чисел мод использовалась формула (1). Далее приведён соответствующий график исходного сигнала (рис.2).

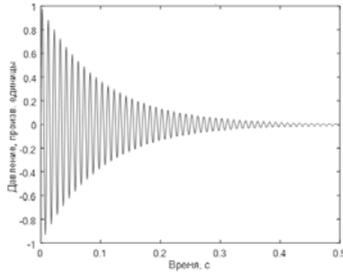


Рис. 2. Исходный сигнал (частота 100 Гц)

На примере второй моды рассмотрим её изменение в зависимости от глубины моря (рис.3).

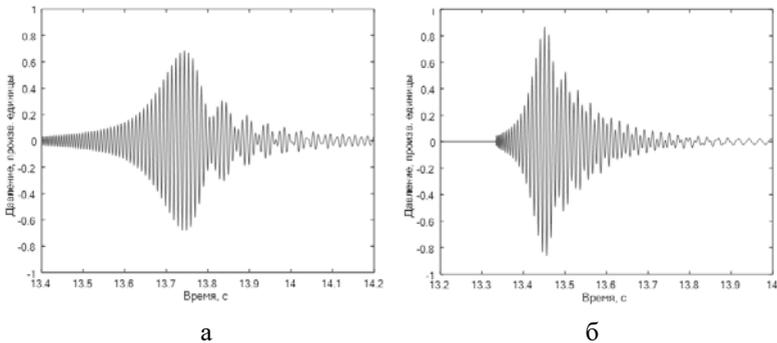


Рис. 3. Вторая мода полученного сигнала (частота исходного сигнала 100 Гц, длина акустической трассы 20 км, а – при глубине 50 м, б – при 100 м)

Из приведённых выше графиков можно сделать вывод, что с увеличением глубины используемого волновода уменьшается искажение получаемого сигнала и его время распространения. А при уменьшении глубины сигнал всё больше растягивается во времени из-за увеличения разности групповых скоростей для разных частот волны.

Сравнивая подобным образом моды полученного сигнала при разных частотах исходного сигнала, можно сделать вывод, что с увеличением частоты форма принятого сигнала и его длительность всё больше соответствуют параметрам исходного сигнала. И такой же результат получается при уменьшении длины преодолеваемой акустической трассы.

Сравним моды полученного сигнала для случая, когда скорость в волноводе остаётся неизменной и равна 1500 м/с и для случая, когда скорость изменяется в зависимости от глубины волновода и соответствует профилю скорости звука, приведённому на рис.4.

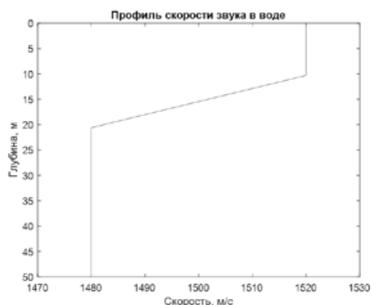
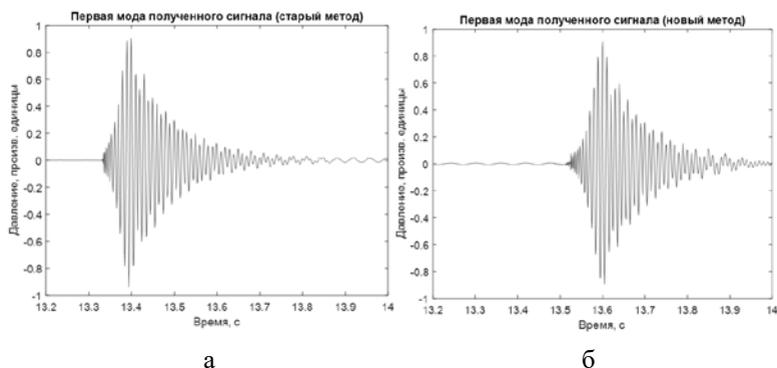


Рис. 4. Пример профиля скорости звука



а – посчитанных по формуле (1), б – полученных численно в ходе решения задачи Штурма-Лиувилля

Рис. 5. Первая мода полученного сигнала, построенная на основе горизонтальных волновых чисел

Как можно заметить на рис.5, во втором случае, так как с глубиной скорость звука в воде уменьшается и начиная с глубины 20 метров остаётся равной 1480 м/с, основная энергия сигнала приходит примерно на две десятые секунды позже, что отличается от случая, когда скорость звука постоянна. А это означает, что изменение скорости звука с глубиной заметно влияет на время прихода сигнала на приёмник.

4. Архитектура разработанного приложения

Диаграмма на рис.6 демонстрирует архитектуру приложения в целом и описывает взаимодействия между клиентским уровнем, включающим в себя файлы графического интерфейса и модуль вызова функций, предоставляющих приведённую к нужному формату информацию; уровнем логики, включающего в себя модуль расчёта горизонтальных волновых чисел и модуль расчёта принятого сигнала; и уровнем хранилища данных, представляющим собой базы данных горизонтальных волновых чисел для конкретных волноводов, управление которыми происходит посредством СУБД PostgreSQL.

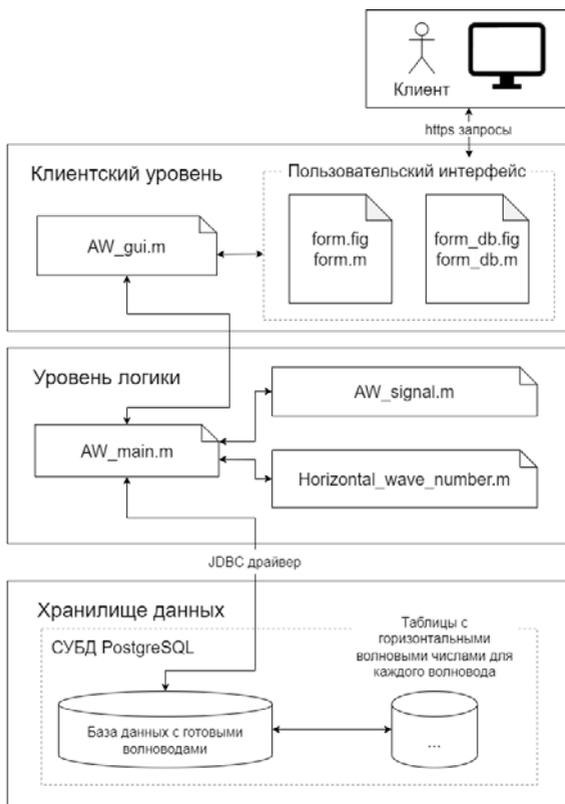
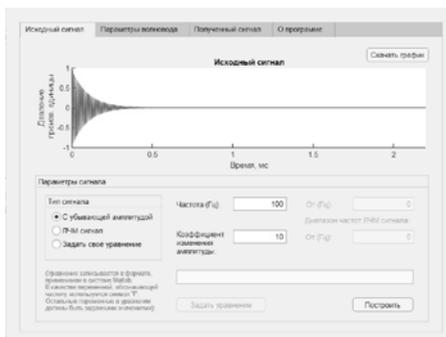


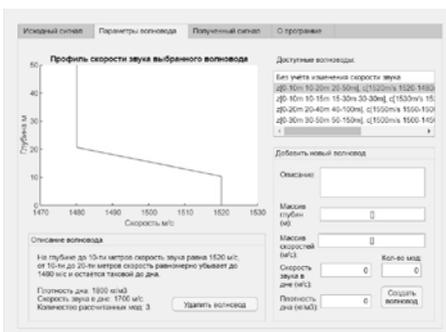
Рис. 6. Архитектура приложения

Интерфейс разработанного приложения показан на рис. 7. Пользователь может настроить параметры исходного сигнала, длину

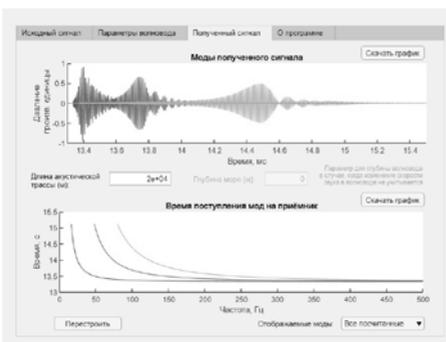
акустической трассы и глубину моря, а также выбрать, будет ли учитываться в волноводе изменение скорости звука в зависимости от глубины, и установить нужный профиль скорости звука.



а



б



в

Рис. 7. Интерфейс веб-приложения, где, а – вкладка с параметрами исходного сигнала, б – параметры волновода, в – полученный сигнал

В более расширенных настройках волновода можно прочитать подробное описание каждого волновода из базы данных, а также добавить волновод со своими параметрами. В таком случае расчёт горизонтальных волновых чисел займёт некоторое время, в зависимости от того, какое количество мод для расчёта указал пользователь. Затем в базе данных будет создана новая таблица с горизонтальными волновыми числами для нового волновода, и он появится в списке доступных волноводов.

Заключение

Разработанное в среде Matlab приложение позволяет графически отображать поведение мод сигнала при различных входных данных. Оно учитывает особенности входного сигнала и свойства волновода. Также удалось подтвердить корректность результатов работы программы, проверив их на соответствие следующим закономерностям:

- В областях с большей глубиной сигнал искажается меньше;
- С увеличением номера моды его форма искажается сильнее;
- С повышением частоты исходного сигнала принимаемый сигнал точнее соответствует излучённому;
- Профиль скорости звука заметно влияет на время прихода мод на приёмник.

Список литературы

1. Рабинович М.И., Трубецков Д.И. Введение в теорию колебаний и волн / М.И. Рабинович, Д.И. Трубецков – НИЦ «Регулярная и хаотическая динамика», 2000. – 560 с.
2. Кацнельсон Б.Г., Петников В.Г. Акустика мелкого моря / Б.Г. Кацнельсон, В.Г. Петников – М.: Наука, 1997. – 191 с.
3. Бреховских Л.М., Лысанов Ю.П. Теоретические основы акустики океана / Л.М. Бреховских, Ю.П. Лысанов – М.: Наука, 2007. – 370 с.

Разработка инструментария для исследования и отладки сетей Zigbee

Н.В. Пауков

Студент магистратуры

Д. И. Соломатин

Старший преподаватель

Введение

В настоящее время одной из самых передовых и активно развивающихся сфер технологий является разработка устройств с применением микроконтроллеров. В частности, активно развиваются технологии, позволяющие делать такие устройства беспроводными и объединять их в сети. Большое распространение получили так называемые «умные» устройства, такие как управляемые лампочки, розетки, датчики, детекторы звука или движения, с помощью которых можно автоматизировать различные домашние или производственные процессы. В число наиболее известных технологий для работы с беспроводными сетями входит технология Zigbee [1].

В ходе разработки и эксплуатации устройств и сетей Zigbee как конечные пользователи, так и разработчики сталкиваются с целым рядом сложностей при исследовании поведения устройств в сети и их отладке. Данные сложности связаны, прежде всего, с отсутствием привычных пользовательских интерфейсов для взаимодействия с устройствами в реальном времени и для автоматизации такого взаимодействия.

В ходе данной работы был разработан специальный инструментарий, целью которого стало повышение эффективности исследования, настройки и отладки как сетей Zigbee в целом, так и отдельных сетевых узлов в их составе.

Для достижения данной цели были поставлены задачи: изучить причины сложностей отладки беспроводных устройств, собрать требования к разрабатываемому инструментарию, а затем спроектировать, разработать и протестировать его необходимые компоненты.

1. Определение сложностей отладки устройств и сетей Zigbee

Технология Zigbee предназначена для объединения в сети недорогих устройств с низким энергопотреблением в случаях, когда не требуется высокая скорость передачи данных, но важна надежность и устойчивость сети. Одной из наиболее частых областей применения таких устройств является всевозможная домашняя автоматизация и построение систем для оповещения о различных событиях [2].

Устройства Zigbee как правило являются малогабаритными и предназначены для встраивания в домашний интерьер. Конструкция многих устройств предполагает наличие специального корпуса, внутри которого находится микроконтроллер. Данные особенности во многих случаях не позволяют использовать традиционное проводное подключение устройств к персональному компьютеру для их отладки или настройки. Кроме того, устройства обычно не предоставляют какого-либо графического интерфейса, поэтому для их настройки необходимо прибегать к использованию дополнительных инструментов, например, сетевых шлюзов.

Тем не менее, несмотря на то, что шлюзы упрощают настройку и диагностику сетей для конечного пользователя, их использование не всегда является желательным или возможным. В частности, не всегда новые устройства сразу поддерживаются шлюзами, а в случае большого количества устройств может оказаться, что часть устройств поддерживается только одним шлюзом, часть – другим, а их совместное использование не допускается. Кроме того, шлюзы предназначены для интеграции с готовыми устройствами и имеют низкую применимость в случаях, когда необходимо отлаживать поведение устройства в момент разработки.

Так, разработчику устройства часто требуется моделировать различные топологии сетей, варьировать роли устройств в них, менять характеристики устройств, воспроизводить сетевые конфликты, отправлять произвольные сетевые пакеты и их последовательности, а затем проверять реакцию устройства.

В таких случаях необходимо использование специальных инструментов, предназначенных не только для настройки сетей конечным пользователем, но и для их отладки и исследования в ходе разработки.

В качестве существующих решений для выполнения перечисленных задач можно выделить такие инструменты как Ember Application Framework Command Line Interface и Zigbee CLI Agent в составе nRF52 SDK for Thread and Zigbee. Данные инструменты позволяют запускать устройства с произвольными параметрами и

управлять их сетевым поведением, например, инициировать подключение к сети и выход из нее, а также отправку различных пакетов, однако, они обладают рядом недостатков при описанных сценариях использования.

Во-первых, пользовательские интерфейсы данных инструментов являются достаточно низкоуровневыми и сложными в развертывании и применении при необходимости многократно выполнять какие-либо операции с устройствами в сети. Во-вторых, они предназначены для ручного управления устройствами и не предоставляют возможностей для автоматизации. Кроме того, выполнение некоторых настроек, в частности, объявление конечных точек Zigbee и списка поддерживаемых кластеров может требовать написания вспомогательного кода и перепрошивки устройства при каждом изменении данных списков.

Кроме того, для повышения эффективности исследования и отладки поведения устройства в сети требуется иметь возможность не только управлять устройствами по отдельности, но и разворачивать произвольные сети с заданной топологией с помощью заранее разработанных скриптов.

Для подтверждения представленных утверждений был проведен опрос среди инженеров, занимающихся разработкой устройств на основе сетевого стека Zigbee.

2. Определение требований к инструментарию

В результате проведенного анализа предметной области и опроса разработчиков приложений Zigbee был сформулирован набор основных требований к инструментарию для исследования и отладки сетей.

- Возможность управлять поведением отдельных устройств в сети (запускать с произвольными параметрами, инициировать сетевые события и процедуры в соответствии со спецификацией Zigbee).
- Возможность автоматизировать действия с устройствами с помощью скриптов на языке программирования Python.
- Возможность интегрировать инструментарий с существующими приложениями, позволяя управлять их поведением в сети.
- Наличие графического интерфейса для выполнения команд, а также возможность работы с инструментарием с помощью командной строки.

3. Разработка инструментария

Для реализации поставленных требований в качестве основы при проектировании инструментария была выбрана клиент-серверная архитектура.

На рис.1 представлена диаграмма, демонстрирующая полученную в результате проектирования архитектуру инструментария.

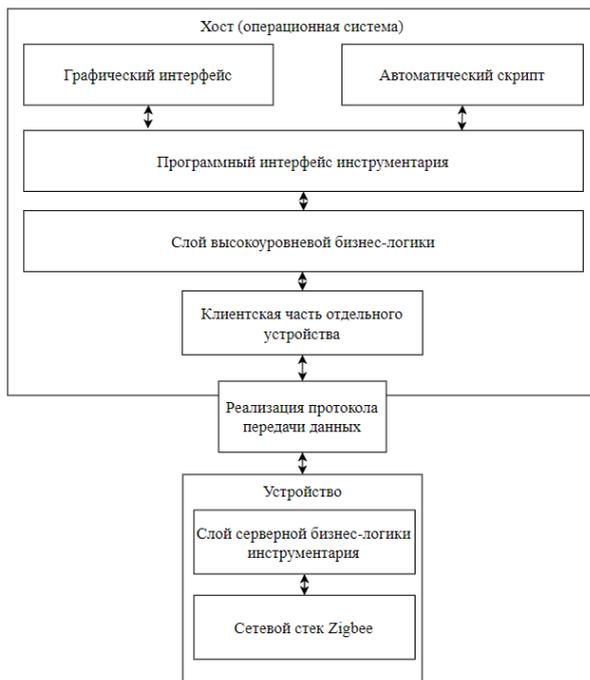


Рис. 1. Архитектура инструментария

В качестве серверной стороны выступает платформа, которая отвечает за запуск сетевого стека Zigbee и взаимодействие с беспроводной сетью. В данном проекте в качестве основной платформы выступает отладочная плата nRF52840 DK, а в качестве сетевого стека Zigbee – стек ZBOSS, входящий в состав nRF5 SDK for Thread and Zigbee [3].

Для программирования серверной бизнес-логики инструментария, располагающейся поверх сетевого стека, был выбран язык программирования Си в силу кроссплатформенности и широких возможностей оптимизации программного кода [4].

Клиентская сторона, в свою очередь, располагается на персональном компьютере или ноутбуке в рамках операционной системы Windows или Linux. Для реализации клиентской бизнес-логики, графического интерфейса и программного интерфейса для

автоматических скриптов используется язык программирования Python и набор библиотек PyQt5.

Взаимодействие клиента и сервера организовано посредством специального текстового протокола передачи данных, который позволяет клиенту отправлять команды для их выполнения на сервер, получать результаты выполнения, а также уведомления о различных сетевых событиях.

Внутренняя реализация протокола может использовать по выбору пользователя проводное соединение через последовательный порт, технологию Bluetooth или передачу команд через специальный пользовательский кластер уровня ZCL в составе приложения Zigbee.

На рис.2 можно увидеть демонстрацию графического интерфейса разработанного инструментария. В соответствии с рисунком, интерфейс включает в себя основное меню с подменю для загрузки скриптов и выполнения действий с сетью в целом или отдельными устройствами, а также вкладки, позволяющие просматривать состояние запущенных устройств и управлять ими.

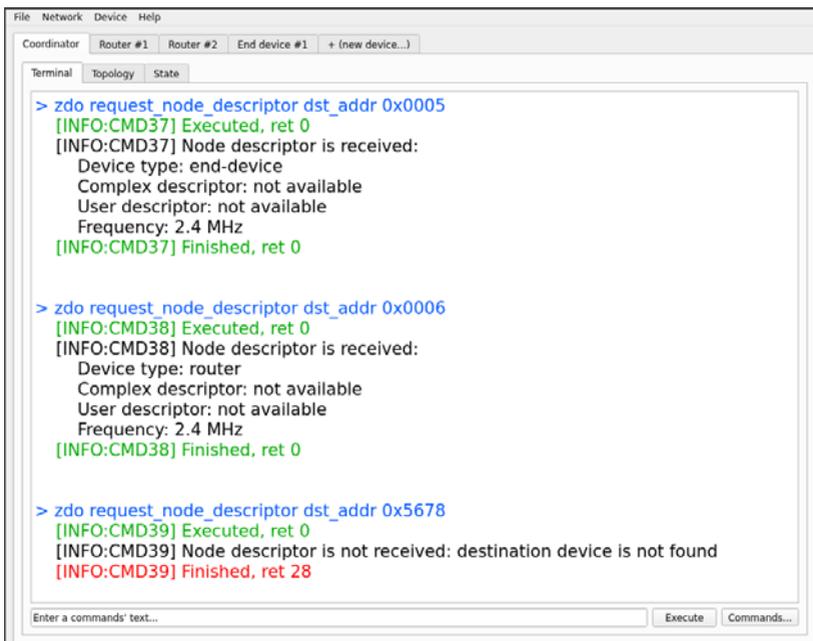


Рис. 2. Главное окно графического интерфейса инструментария

Вкладка устройства, в свою очередь, содержит вложенные вкладки для просмотра истории выполненных команд, визуализации топологии сети, доступной из выбранного устройства и вкладку для просмотра и настройки параметров устройства, продемонстрированную на рис. 3.

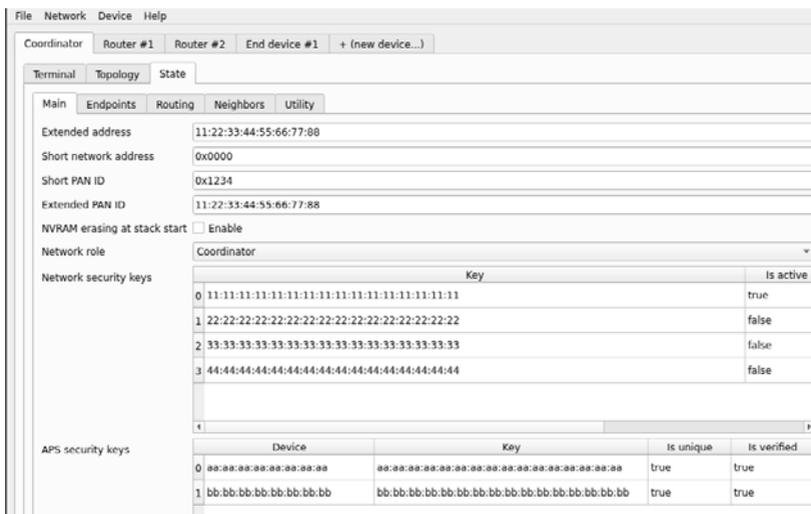


Рис. 3. Вкладка настройки сетевого устройства

Разработанный инструментарий позволяет просматривать основные параметры, задающие работу устройства, а также редактировать их значения. Так, например, для проверки реакции устройств в сети на смену сетевого ключа, пользователь инструментария может добавить произвольный ключ в настройки устройства, а затем отправить команду на инициирование процедуры замены ключа для всей сети.

Заключение

В ходе работы был разработан инструментарий, предназначенный для исследования и отладки беспроводных сетей, использующих сетевой стек Zigbee.

Инструментарий спроектирован на основе клиент-серверной архитектуры и включает в себя платформу, которая отвечает за запуск сетевого стека, а также графический интерфейс и программный интерфейс для разработки автоматических скриптов.

Для дальнейшего развития инструментария запланировано расширение списка доступных команд и настроек, а также поддержка

новых типов устройств Zigbee, таких как устройства с использованием технологий Green Power и Smart Energy.

Список литературы

1. Gislason, D. Zigbee Wireless Networking / D. Gislason. – New York: Newnes, 2008. – 448 p.

2. Беспроводные сети Zigbee [Электронный ресурс]. – Режим доступа : <https://habr.com/ru/companies/efo/articles/281048/>

3. Справочник по API сетевого стека ZBOSS [Электронный ресурс] : база данных. – Режим доступа : https://developer.nordicsemi.com/nRF_Connect_SDK/doc/zboss_api_doc/modules.html

4. Документация по языку программированию Си [Электронный ресурс] : база данных. – Режим доступа : <https://en.cppreference.com/w/c/>

Исследования трафика корпоративной сети с применением машинного обучения (в задачах оптимизации использования ресурсов)

Ю. В. Петров

Студент магистратуры

А. С. Коваль

Старший преподаватель

Введение

В наши дни вполне естественным считается прохождение через компьютерные сети гигантских объемов информации. Происходит это как в глобальной сети Интернет, так и в корпоративных сетях, причем нет большого значения, крупная это фирма или небольшая организация.

В отличие от сравнительно недавнего прошлого, проблема с лимитированностью трафика не стоит так уж остро. Тем не менее, внутри корпоративных сетей помимо полезного трафика есть трафик, который может быть для конечного пользователя и необходимых ему приложений бесполезным, он может быть вызван, например, неправильной настройкой сетевых приложений, а также принадлежать приложениям вредным, будь то вирусные приложения, или малополезные системные (как, например, телеметрия, встроенная в современные версии Windows). В связи с этим возникает потребность в анализе трафика внутри сети с целью определения его составляющих. Также это можно производить с целью предотвращения угроз, связанных с перехватом данных и т.д.

Из вышеизложенного следует, что требуется выполнять классификацию сетевого трафика для выявления бесполезного или вредоносного. В ее основе – определение отличий в трафике, показывающих уникальные характеристики у тех или иных приложений. Это не требует больших затрат ресурсов, что с учетом цели в виде оптимизации их использования в ходе работы сети особенно актуально.

Одной из целей работы можно выделить определение эффективности использования машинного обучения для классификации трафика. Дело в том, что нейросети позволят куда более эффективно, чем классические методы, такие как сигнатурный, выявить паттерны, соответствующие тому или иному типу трафика, а также запомнить большее их количество и т. п. Также простой анализ передаваемых

пакетов по заголовкам – ненадежный способ из-за как динамических портов транспортных протоколов, так и мер борьбы разработчиков приложений с такой идентификацией.

В связи с этим планируется опробовать использование машинного обучения с целью оценки его эффективности в рамках задач классификации сетевого трафика.

Целью работы является апробирование методов машинного обучения для оптимизации использования ресурсов корпоративной компьютерной сети на примере сети ФКН ВГУ. Каналы связи сети имеют неограниченную пропускную способность, потому паразитный трафик (что бесполезный, что вирусный), забивает ее, оставляя меньше ресурсов для трафика полезного, снижая тем самым их работу и нарушая стабильность передачи данных. Также работа ненужных приложений нагружает компьютеры (что клиентские, что серверные), замедляя их работу. Эти проблемы в той или иной степени могут присутствовать внутри сети ФКН, потому в целях их выявления и дальнейшего устранения данная работа будет актуальна.

К задачам можно отнести:

1. изучение теоретических основ классификации и применимости машинного обучения к ней;
2. выбор и работа с моделью данных, сбор датасетов, полезных для выявления бесполезного и вредного трафика;
3. работа с трафиком сети ФКН ВГУ с целью выявления бесполезного и вредного трафика, а также борьба с ним.

На текущем этапе работы пока еще не было произведено практического изучения трафика реальной сети и, соответственно, дальнейших действий, связанных с этим, все это находится в процессе выполнения.

1. Понятие классификации трафика

Классификация сетевого трафика – это определение типа трафика, который проходит через сеть. Это имеет большое значение в первую очередь для защиты передаваемой информации и управления трафиком сети передачи данных [1–3]. Значительную роль в этом процессе играет математическое моделирование, решающее следующие задачи [4-7]:

1. прогнозирование сетевого трафика для оценки ресурсов, которые требуются для получения необходимого качества обслуживания;
2. оценка влияния алгоритмов управления сетевым трафиком на характеристики сети;
3. исследование процессов, происходящих в сети;

4. генерация трафика для проведения имитационного моделирования;

5. определение источников сетевого трафика на основе его характеристик.

В основе же моделей, используемых в ходе имитационного моделирования, лежат:

6. различные законы распределения;

7. стохастические временные ряды;

8. теория фракталов;

9. цепи Маркова.

К сожалению, описанные выше модели имеют значительный недостаток: они моделируют сетевой трафик по примеру уже существовавшего ранее, т.е. новое поведение получить в данных моделях будет попросту невозможно. Также с их помощью невозможно моделирование работы в режиме реального времени. Тем не менее, их применение тоже важно, потому что при наличии достаточного количества примеров поведения реального сетевого трафика работа модели будет достаточно реалистичной, чтобы взаимодействие с ней было сколь-либо продуктивным.

2. Машинное обучение в рамках задачи классификации трафика

Машинное обучение в задаче классификации сетевого трафика подразделяется на два вида:

1. методы обучения с учителем;

2. методы обучения без учителя.

Методы обучения с учителем в своей основе имеют статистику и вероятность, т.е. обучение и, следовательно, прогнозирование происходит на каком-то объеме уже существующих данных. Эти методы позволяют приложениям обучаться без специального программного обеспечения. Обучение без учителя может подразумевать использование нескольких уровней данных. Т.е., после обработки исходных данных результат выносится на следующий уровень и т.д.

Методы обучения с учителем делятся на:

1. задачи классификации;

2. задачи регрессии.

Задача классификации в свою очередь разделяется на:

3. метод опорных векторов;

4. дискриминантный анализ;

5. байесовские методы;

6. метод k-ближайших соседей;

7. нейронные сети (глубокое обучение).

Задача регрессии же делится на:

1. линейную регрессию;
2. метод опорных векторов;
3. ансамблевые методы;
4. дерево принятия решений;
5. нейронные сети (глубокое обучение).

К методам машинного обучения без учителя относится задача кластеризации, которая делится на:

1. метод k-средних;
2. иерархические модели;
3. смешанную Гауссовскую;
4. скрытую Марковскую;
5. нейронные сети (глубокое обучение).

Процесс анализа сетевого трафика вполне может использовать различные из описанных выше методов. К примеру, на этапе фильтрации трафика методы машинного обучения без учителя могут хорошо помочь с классификацией данных по уровню их важности и отсеять большую часть шума. На этапе же с классификацией можно задействовать и методы машинного обучения с учителем.

3. Данные для машинного обучения

В сети Интернет на различных ресурсах, таких как, например, kaggle.com, для обучения нейросетей представлены различные датасеты, содержащие данные о сетевом трафике, в том числе и о трафике внутрикорпоративных сетей. Одним из них является датасет, собранный в компьютерной сети Университета Кауки в Колумбии (рис. 1), который представлен в виде csv-таблицы.

Flow_ID	Source_IP	Source_Port	Destination_IP	Destination_Port	Protocol	Timestamp	Flow.Duration	Total.Fwd.Packets
172.19.1.46-10.200.7.7-52422-3128-6	172.19.1.46	52422	10.200.7.7	3128	6	26/04/2017:11:11:17	45523	22
172.19.1.46-10.200.7.7-52422-3128-6	10.200.7.7	3128	172.19.1.46	52422	6	26/04/2017:11:11:17	1	2
10.200.7.217-50.31.185.39-38848-80-6	50.31.185.39	80	10.200.7.217	38848	6	26/04/2017:11:11:17	1	3
10.200.7.217-50.31.185.39-38848-80-6	50.31.185.39	80	10.200.7.217	38848	6	26/04/2017:11:11:17	217	1
192.168.72.43-10.200.7.7-55961-3128-6	192.168.72.43	55961	10.200.7.7	3128	6	26/04/2017:11:11:17	78068	5
172.19.1.56-10.200.7.6-50004-3128-6	10.200.7.6	3128	172.19.1.56	50004	6	26/04/2017:11:11:17	105069	136
192.168.72.43-10.200.7.7-55963-3128-6	192.168.72.43	55963	10.200.7.7	3128	6	26/04/2017:11:11:17	104443	5
192.168.10.47-10.200.7.6-1849-3128-6	192.168.10.47	51848	10.200.7.6	3128	6	26/04/2017:11:11:17	11002	3
10.200.7.217-68.67.178.197-57300-443-6	68.67.178.197	443	10.200.7.217	57300	6	26/04/2017:11:11:17	106503	10
192.168.72.43-10.200.7.7-55977-3128-6	192.168.72.43	55977	10.200.7.7	3128	6	26/04/2017:11:11:17	118415	7
192.168.180.51-10.200.7.4-57740-3128-6	10.200.7.4	3128	192.168.180.51	57740	6	26/04/2017:11:11:17	205118	32
192.168.180.51-10.200.7.4-57740-3128-6	10.200.7.4	3128	192.168.180.51	57740	6	26/04/2017:11:11:17	3	5
192.168.180.51-10.200.7.4-57740-3128-6	10.200.7.4	3128	192.168.180.51	57740	6	26/04/2017:11:11:17	131	3
192.168.180.51-10.200.7.4-57740-3128-6	10.200.7.4	3128	192.168.180.51	57740	6	26/04/2017:11:11:17	6	3
172.19.1.45-10.200.7.6-50227-3128-6	10.200.7.6	3128	172.19.1.45	50227	6	26/04/2017:11:11:17	108338	123
212.124.124.94-10.200.7.194-443-44447-6	212.124.124.94	443	10.200.7.194	44447	6	26/04/2017:11:11:17	202096	4
192.168.180.51-10.200.7.4-57741-3128-6	10.200.7.4	3128	192.168.180.51	57741	6	26/04/2017:11:11:17	202151	31
192.168.180.51-10.200.7.4-57741-3128-6	10.200.7.4	3128	192.168.180.51	57741	6	26/04/2017:11:11:17	1	2
192.168.180.51-10.200.7.4-57741-3128-6	10.200.7.4	3128	192.168.180.51	57741	6	26/04/2017:11:11:17	1	2

Рис. 1. csv-таблица датасета от Университета Кауки

Так как этот датасет содержит данные о работе реальной университетской сети, он выглядит подходящим вариантом для выбора данных, на которых планируется обучать нейросеть. К сильным его

сторонам также можно отнести большое количество признаков представленных сущностей, так как позволит при анализе сети ФКН более точно производить классификацию трафика, чем если бы их количество было заметно ограничено. Еще одно свойство – большое количество записей, поэтому в данном случае возможно использование различных выборок из него, что в свою очередь позволит реализовать вариативность при обучении нейросети.

Еще один вариант – датасет от Канадского Института Кибербезопасности (рис. 2).

SourceIP	DestinationIP	SourcePort	DestinationPort	Timestamp	Duration	FlowBytesOut	FlowBytesIn	FlowBytesReverse	FlowBytesTotal	
72.31.91.42	192.168.20.191	443	51041	14.01.2020 15:49 4.544040	1125631	2477163	71334759377181591711	159324	20262	191539493490051
192.168.20.191	195.201.169.48	51043	443	14.01.2020 15:49 8.171041	930	113.8165871398760574081074599		11936	1460.	70485850554415278
192.168.20.191	96.17.115.57	51021	443	14.01.2020 15:49 0.033776	162	4796.3050686887825670535528186		138	4085.	74135480818477979
192.168.20.191	96.134.14.140	50308	443	14.01.2020 15:49 0.060723	108	1778.56823256691863050244553		120	1976.	180947265212181
192.168.20.191	23.78.199.190	50983	443	14.01.2020 15:49 0.019545	55	2814.61891067708034433589153		66	3376.	82277480778676132
192.168.20.191	66.218.84.45	49762	443	14.01.2020 15:49 0.029580	55	1855.564454394341545163684922		60	2028.	397565922020820
192.168.20.191	151.101.124.237	50649	443	14.01.2020 15:49 0.025021	55	2108.153551017145599709793736		66	2637.	784261220574737
192.168.20.191	151.101.124.237	50650	443	14.01.2020 15:49 0.025077	55	2193.1448059957276184045066423		66	2631.	4937671970331379
192.168.20.191	104.244.42.195	49745	443	14.01.2020 15:49 0.044585	55	1233.59874397218795559044517		66	1480.	3184627566255465
192.168.20.191	151.101.124.84	49872	443	14.01.2020 15:49 0.025090	55	2192.42840977489902348798644886		66	2630.	13007166998404346
31.18.80.12	192.168.20.191	443	50794	14.01.2020 15:49 0.027098	213	7860.358608688807738596723005		108	3085.	533087781748123268
54.210.11.71	192.168.20.191	443	50882	14.01.2020 15:49 0.028243	245	8674.715484796870020800132068		108	3823.	9563785176814781
192.168.20.191	213.180.193.105	49676	443	14.01.2020 15:49 0.139417	35	394.489955377279823730737818		66	473.	33994405273880462
192.168.20.191	151.101.124.238	50646	443	14.01.2020 15:49 0.025122	55	2189.3161172952189311725022		66	2627.	1794647002027179
31.18.80.36	192.168.20.191	443	50797	14.01.2020 15:49 0.027173	213	7838.4663379089537408456924153		108	3974.	5354443270893002
192.168.20.191	151.101.124.238	50676	443	14.01.2020 15:49 0.025168	55	2185.11468331468314685146845		66	2622.	1776223776223778
192.168.20.191	82.202.190.90	85017	443	14.01.2020 15:49 0.138894	55	396.556448007844607533188170		66	432.	60703419037593955
192.168.20.191	104.16.198.5	50676	443	14.01.2020 15:49 0.015271	55	3602.5977997511621333813742739		66	4321.	9173597039480808
192.168.20.191	151.101.126.110	50596	443	14.01.2020 15:49 0.025058	55	2194.90781387181738366885865		66	2633.	8893766461808004

Рис. 2. csv-таблица датасета от Канадского Института Кибербезопасности

Этот датасет содержит меньшее количество признаков сущностей, чем предыдущий, и меньший объем, однако он имеет в себе примеры заведомо вредоносного и заведомо доброкачественного трафика через протоколы DNS и DoH (DNS over HTTPS), что в свою очередь может оказаться полезным, например, при рассмотрении роли DNS-сервера ВГУ в рамках работы сети ФКН (которая является частью всей сети ВГУ).

Еще один датасет на сайте kaggle.com под названием Internet Traffic Data Set (рис. 3) представлен сотрудником пакистанского Института компьютерных наук Аль-Хаварезми Мухаммадом Асфанд Яром. Он содержит данные об интернет-трафике, собранные местным интернет-провайдером, у более чем 2000 пользователей на протяжении нескольких месяцев. В отличие от вышерассмотренных датасетов, этот представляет из себя не csv-таблицу, а собранные данные захваченного трафика в исходном виде, что позволяет изучить их через анализаторы пакетов вроде Wireshark.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	103.12.198.162	39.62.1.38	UDP	1274	443 → 45841 Len=1232
2	-0.005975	39.62.1.38	142.250.181.54	TCP	78	2655 → 443 [ACK] Seq=1 Ack=1 Win=3708 Len=0
3	-0.005898	39.62.1.38	103.12.198.146	TCP	66	44538 → 443 [ACK] Seq=1 Ack=1 Win=1736 Len=0
4	-0.005898	39.62.1.38	103.12.198.146	TCP	66	44540 → 443 [ACK] Seq=1 Ack=1 Win=1736 Len=0
5	-0.005889	39.62.1.38	103.12.198.146	TCP	66	44540 → 443 [ACK] Seq=1 Ack=2761 Win=1736 Len=0
6	-0.005888	39.62.1.38	103.12.198.146	TCP	66	44538 → 443 [ACK] Seq=1 Ack=1 Win=1736 Len=0
7	-0.005881	39.62.1.38	103.12.198.146	TCP	66	44538 → 443 [ACK] Seq=1 Ack=6901 Win=1736 Len=0
8	-0.005881	39.62.1.38	103.12.198.146	TCP	66	44540 → 443 [ACK] Seq=1 Ack=6901 Win=1736 Len=0
9	-0.005875	39.62.1.38	103.12.198.146	TCP	66	44540 → 443 [ACK] Seq=1 Ack=13801 Win=1730 Len=0


```

> Frame 1: 1274 bytes on wire (10192 bits), 1274 bytes captured (10192 bits) on interface 0
> Ethernet II, Src: JuniperX_7b:41:7c (48:0f:9d:7b:41:7c), Dst: Datalink (08:00:27:00:00:00)
> Internet Protocol Version 4, Src: 103.12.198.162, Dst: 39.62.1.38
> User Datagram Protocol, Src Port: 443, Dst Port: 45841
Data (1232 bytes)
  Data: 496414bf671966a13d61680318875a7b18af8eef523eca82fa97c81232
  [Length: 1232]
0000 b8 ca 3a b5 4b 0c 40 8f 9d 7b 41 7c 08 00 45 00
0010 04 ec 00 03 40 00 5e 11 c1 eb 67 0c c6 a2 27 3e
0020 01 26 01 bb b3 11 04 d8 a9 e1 49 64 14 bf 67 19
0030 66 a1 3d e1 68 03 18 87 5a 7b 18 af 8e ef 52 3e
0040 ca 82 fa 07 c8 b5 69 32 92 9b 89 fb d6 b3 c1 50
0050 bf b5 34 d3 98 bb d2 71 9c 28 78 af 12 24 38 33
0060 50 db a3 40 bd 10 c6 f7 87 21 62 a5 a8 26 39 fd
0070 96 76 6a 69 66 96 69 40 e8 8c 68 9d 3d b3 20 76
0080 08 6e 96 78 c5 9a 4a 22 b6 b7 1a cd 2a aa 8f 1e
0090 cf c5 39 2e f3 55 0d 6f 41 d9 8a 50 10 0c f8 b0
00a0 6d 8c 56 18 aa 77 ef c6 4f 60 ce 44 4d 9b 14 eb
00b0 fa 89 a8 7a 7d 77 19 ab dc 56 ae 10 fb 0c 09 c1

```

Рис. 3. выдержка из трафика от пакистанского Института компьютерных наук Аль-Хаварезми в анализаторе пакетов Wireshark

Рассмотренные выше датасеты, а также другие, подобные им, могут быть использованы для обучения нейросети, которая впоследствии будет производить анализ трафика сети ФКН ВГУ.

Заключение

Как следует из поставленных задач, в дальнейшем будут реализованы классификаторы для выявления бесполезного и вредного трафика, сформированные на основе обучения на сформированных датасетах и моделях данных. Классификаторы будут апробированы на трафике сети факультета компьютерных наук, с целью выявления и последующего устранения бесполезного и вредоносного трафиков и, таким образом, повышения эффективности использования ресурсов сети. На текущий момент были рассмотрены основные идеи, связанные с применением машинного обучения в рамках изучения трафика сети с задачей оптимизации использования ресурсов, а также материалы, которые будут использованы в этом процессе.

Список литературы

1. Кузьмин, В.В. Модели и процедуры управления трафиком в мультисервисной сети оператора связи: диссертация: дис. ... канд. техн. наук / В.В. Кузьмин. – Нижний, 2015. – 189 с.
2. Sperotto, A. Hidden markov model modeling of ssh brute-force attacks / A. Sperotto, R. Sadre, P. de Boer, A. Pras // Integrated Management of Systems, Services, Processes and People in IT. – 2009. – С. 164–176.
3. Velan, P. A survey of methods for encrypted traffic classification and analysis / P. Velan, M. Čermák, P. Čeleda // International Journal of Network Management. – 2015. – № 25(5). – С. 355–374.

4. Благов, А.В. Исследование адекватности моделей самоподобного трафика, используемых для оценки качества обслуживания в сети / А.В. Благов // Электронный журнал: Труды МАИ. – 2014. – № 76. – С. 17-23.

5. Соловьев, А.Ю. О задаче прогнозирования самоподобных сетевых процессов / А.Ю. Соловьев // II Международная научная конференция «Современные проблемы информатизации в системах моделирования, программирования и телекоммуникациях». – 2009. – С. 3-12.

6. Box, G.E.P. Time Series Analysis, Forecasting and Control. Revised ed. San Francisco / G.E.P. Box, G.M. Jenkins. – CA: Holden-Day, 1976 – 112 p.

7. Muscariello, L. Markov models of internet traffic and a new hierarchical MMPP model / L. Muscariello, M. Mellia, M. Meo, M.A. Marsan, R. Lo Cigno // Computer Communications Journal. – 2005. – № 28(16). – С. 1835–1851.

Анализ плоских движений фигур сложной формы

М. О. Саввин

Студент магистратуры

А. В. Лобода

Профессор

Введение

Рассматривается двухзвенная равнобедренная ломаная M (далее ломаная), состоящая из двух отрезков одинаковой длины, соединенных под некоторым углом. Целью данной работы является поиск максимальной длины ломаной $M = M_{ABC}$, которая может быть перемещена через Г-образный коридор, образованный тоннелями единичной ширины (см. рис. 1). Эта работа перекликается с оптимизационной задачей о перемещении дивана [1–2], сформулированной в 60-х годах прошлого века и до сих пор являющейся нерешенной геометрической проблемой (см. [3]).

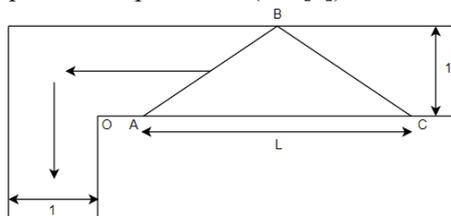


Рис. 1. Постановка задачи

Сразу считаем, что длина основания AC обсуждаемой ломаной фигуры будет превосходить 2 (т.к. иначе длина ломаной будет не больше отрезка предельной длины $2\sqrt{2}$, проходящего через коридор), и, следовательно, фигура должна будет совершать поворот из горизонтального в вертикальное положение при прохождении коридора. Высота рассматриваемой фигуры M будет фиксированной и равной 1.

1. Выделение этапов движения

Обозначим угол наклона отрезка AC к горизонтали через φ , а угол при основании нашей фигуры M ($\angle BAC$) через β . В случае

с движением через симметричный угол достаточно будет рассмотреть поворот фигуры в промежутке $\varphi \in \left[0, \frac{\pi}{4}\right]$ и обсудить размещение фигуры

в коридоре на всем этом промежутке.

Мы рассматриваем здесь частную траекторию движения фигуры M , считая, что в каждый момент такого движения на сторонах коридора находятся по крайней мере две из трех точек A, B, C , задающих эту ломаную. В связи с этим интересующий нас промежуток изменения угла φ предлагается разбить на две части (два этапа): $0 \leq \varphi \leq \beta$ и $\beta \leq \varphi \leq \frac{\pi}{4}$.

Разделяющее эти два промежутка значение $\varphi = \beta$ соответствует горизонтальному положению стороны AC . При $\varphi < \beta$ самой верхней точкой ломаной является точка B ; при $\varphi > \beta$ верхней является точка C .

По окончании второго этапа фигура будет находиться в симметричном положении, а точка B будет расположена на оси симметрии коридора.

2. Первый этап поворота фигуры

На первом этапе нас будет интересовать расположение отрезка AB относительно точки O . Движение отрезка двумя точками по сторонам описано в книге [4] как «скольжение лестницы по сторонам угла». Известно, что каждая из точек отрезка будет двигаться по эллипсу, и наша задача сводится к исследованию семейства этих эллипсов.

Для удобства обсуждения перейдем к новой системе координат, в которой центром является внешний угол коридора, а сам коридор повернут на 90 градусов. Тогда точка $O(0,0)$ превратится в $Q(1,1)$, Выберем точку N на стороне AB , задав соотношение длин AN и NB , равное λ . Такая ситуация показана на рис. 2.

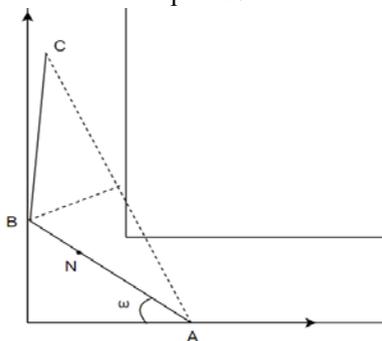


Рис. 2. Движение фигуры на первом этапе

Тогда в соответствии с [4], мы можем выписать уравнение, которое связывает координаты точки N с длиной отрезка АВ (l) и параметром λ :

$$x^2(1+\lambda)^2 + y^2\left(\frac{1+\lambda}{\lambda}\right)^2 = l^2. \quad (1)$$

Уравнение (1) описывает семейство эллипсов, по которым движутся точки отрезка АВ. Параметр λ будет определять конкретную точку на прямой и уравнение эллипса, соответствующее этой точке.

Для прохождения фигуры через коридор точка Q должна быть внешней для любого из этих эллипсов. Для проверки этого свойства достаточно найти огибающую семейства (1). Известно (см. [5]), что огибающая определяется как решение системы вида:

$$\begin{cases} F(x, y, \lambda) = 0 \\ F'_\lambda(x, y, \lambda) = 0 \end{cases}. \quad (2)$$

Дифференцируя уравнение (1) по λ , получим систему уравнений:

$$\begin{cases} (\lambda(1+\lambda))^2 x^2 + (1+\lambda)^2 y^2 = (\lambda l)^2 \\ (1+2\lambda) \cdot 2\lambda(1+\lambda)x^2 + 2(1+\lambda)y^2 = 2\lambda l^2 \end{cases}. \quad (3)$$

Решение системы (3) дает параметрическое описание огибающей:

$$x = \frac{l}{\sqrt{(1+\lambda)^3}}, \quad y = l \sqrt{\frac{\lambda^3}{(1+\lambda)^3}}. \quad (4)$$

Подставим точку Q(1,1) в систему (4) и найдем параметры ломаной M и точки N на ней, при которых происходит «захват» точки Q.

Решением будут значения $\lambda = 1$ (т. е. точка N будет центром прямой АВ) и $l = |AB| = 2\sqrt{2}$. Соответствующая максимальная длина ломаной может быть найдена как $2l = 4\sqrt{2}$.

3. Второй этап поворота фигуры

На втором этапе по стенкам коридора будут двигаться точки А и С, но нас будет все так же интересовать расположение точки Q относительно звеньев ломаной. Однако мы будем рассматривать только движение отрезка АВ, так как точка В в процессе движения со стороны коридора до оси симметрии OQ будет находиться левее точки Q, а значит, и отрезок ВС не будет пересекать угол на всем втором этапе.

Выберем точку N на отрезке АС аналогичным первому этапу образом и отложим перпендикуляр к прямой АС в этой точке (рис. 3). Точку пересечения перпендикуляра с отрезком АВ обозначим как G. Тогда множество точек G будет зависеть от множества точек N, движущихся по эллипсам.

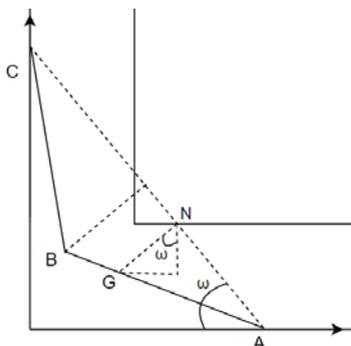


Рис. 3. Движение фигуры на втором этапе

Утверждение. При движении точек A и C по сторонам коридора орбитой любой точки G является наклонный эллипс.

Для данного этапа введем следующие обозначения: $l = |AC|$ и $\lambda = \frac{AN}{NC}$. Также отметим, что здесь значение параметра $\lambda \in (0,1]$, так как соотношение, большее единицы, поместит нашу точку G на отрезок BC, а не AB.

Выразим координаты точки N через угол ω наклона отрезка AC и его длину (см. рис. 3):

$$x_N = \frac{x_A}{1+\lambda} = \frac{l}{1+\lambda} \cos(\omega), \quad y_N = \frac{\lambda y_C}{1+\lambda} = \frac{\lambda l}{1+\lambda} \sin(\omega). \quad (5)$$

Тогда смещения точки G относительно точки N можно выразить через угол наклона фигуры и длину отрезка GN:

$$x_G = x_N - |GN| \sin(\omega), \quad y_G = y_N - |GN| \cos(\omega). \quad (6)$$

Объединяя (5) и (6), получаем описание движения точки G:

$$\begin{cases} x_G = x_N - |GN| \sin(\omega) = \frac{l}{1+\lambda} \cos(\omega) - \frac{2\lambda}{1+\lambda} \sin(\omega) \\ y_G = y_N - |GN| \cos(\omega) = \frac{\lambda l}{1+\lambda} \sin(\omega) - \frac{2\lambda}{1+\lambda} \cos(\omega) \end{cases}. \quad (7)$$

Систему (7) можно записать в следующем общем виде:

$$\begin{cases} x = A \cos(\omega) + B \sin(\omega) \\ y = C \cos(\omega) + D \sin(\omega) \end{cases}, \quad (8)$$

где коэффициенты A, B, C, D выражаются через длину l и параметр λ .

Выражая из (8) $\cos(\omega)$ и $\sin(\omega)$, получим следующие формулы:

$$\cos(\omega) = \frac{Dx - By}{AD - BC}, \quad \sin(\omega) = \frac{-Cx + Ay}{AD - BC}. \quad (9)$$

Тогда по основному тригонометрическому тождеству:

$$(Dx - By)^2 + (-Cx + Ay)^2 = (AD - BC)^2. \quad (10)$$

Подставим сюда в качестве коэффициентов исходные значения из системы (7) и получим уравнение кривой второго порядка.

$$(\lambda(lx + 2y))^2 + (2\lambda x + ly)^2 = \left(\frac{\lambda(l^2 - 4\lambda)}{1 + \lambda}\right)^2. \quad (11)$$

Полученное уравнение и будет являться уравнением семейства наклонных эллипсов. Убедимся в этом, построив их графически при некоторых фиксированных значениях параметра λ .

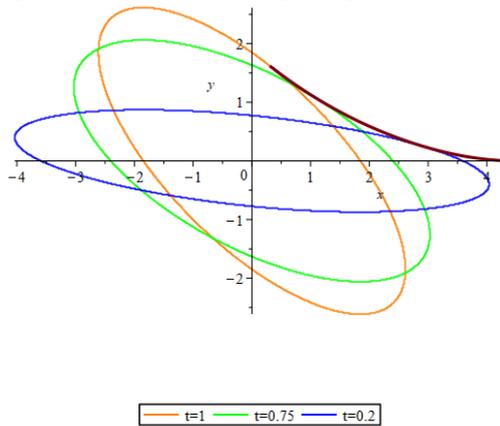


Рис. 4. Кривые из семейства наклонных эллипсов и огибающая

Здесь, как и на первом этапе, нас интересует захват точки $Q(1;1)$. Подставим координаты этой точки в уравнение семейства эллипсов (11) и сгруппируем его по степеням l , предварительно раскрыв скобки:

$$-l^4 \lambda^2 + (\lambda^4 + 10\lambda^3 + 2\lambda^2 + 2\lambda + 1)l^2 + (4\lambda^4 + 12\lambda^3 + 12\lambda^2 + 4\lambda)l - 8\lambda^4 + 16\lambda^3 + 8\lambda^2 = 0. \quad (12)$$

Уравнение (12) описывает кривую, на графике которой (рис. 5) видна интересующая нас точка плоскости $(l_{\text{крит}}, \lambda)$: при $l < l_{\text{крит}}$ уравнение (12) не имеет решений, а при $l > l_{\text{крит}}$ мы получаем уже два значения параметра λ .

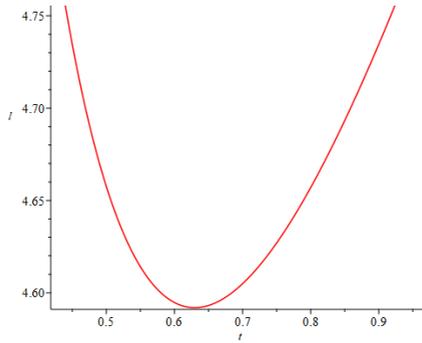


Рис. 5. Решения (l, λ) для захвата точки Q

Для нахождения $l_{\text{крит}}$ учтем, что частная производная левой части (12) в этой точке обращается в ноль. Тогда продифференцируем уравнение (12) по параметру λ и получим систему из двух полиномов:

$$\begin{cases} -l^4\lambda^2 + (\lambda^4 + 10\lambda^3 + 2\lambda^2 + 2\lambda + 1)l^2 + (4\lambda^4 + 12\lambda^3 + 12\lambda^2 + 4\lambda)l - 8\lambda^4 + 16\lambda^3 + 8\lambda^2 = 0, \\ -2l^4\lambda + (4\lambda^3 + 30\lambda^2 + 4\lambda + 2)l^2 + (16\lambda^3 + 36\lambda^2 + 24\lambda + 4)l - 32\lambda^3 + 48\lambda^2 + 16\lambda = 0 \end{cases} \quad (13)$$

Для решения такой системы полиномиальных уравнений воспользуемся базисом Грёбнера (набор процедур Groebner представлен в пакете Maple). Лексикографический порядок построения такого базиса для системы от двух переменных дает следующий многочлен 19-й степени, зависящий только от одной переменной:

$$l^2(l^9 - 2l^8 - 2l^7 - 32l^6 - 39l^5 - 86l^4 - 100l^3 + 56l^2 + 32l - 64)(l + 2)^2(l^2 + 4)^3. \quad (14)$$

Выпишем отдельно второй сомножитель и построим его график:

$$l^9 - 2l^8 - 2l^7 - 32l^6 - 39l^5 - 86l^4 - 100l^3 + 56l^2 + 32l - 64. \quad (15)$$

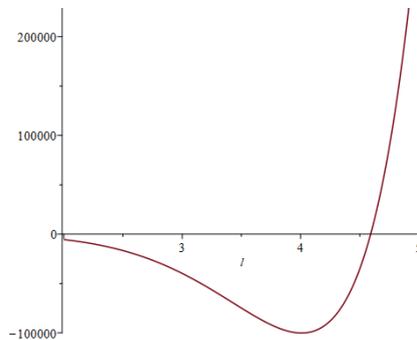


Рис. 6. График функции $f(l)$ в области положительных l

На рис. 6 мы можем видеть, что в области положительных l существует наше искомое значение $l_{крит}$, которое является корнем полиномиального уравнения (15).

Корнем этого уравнения будет $l \approx 4.592$. Это значение будет являться максимальным, при котором наша фигура может пройти коридор на втором этапе. Соответствующую длину ломаной можно найти по теореме Пифагора:

$$L = 2|AB| = 2\sqrt{1 + \left(\frac{l}{2}\right)^2} \approx 5.009.$$

Предельное значение длины фигуры, проходящей оба этапа, будет определяться как наименьшее из двух максимальных значений. Напомним, что на первом этапе мы получили значение $L = 4\sqrt{2} \approx 5.657$, что больше, чем оценка, полученная на втором этапе, а значит, максимальной длиной ломаной будет $L \approx 5.009$.

Заключение

Описан подход к исследованию плоских движений фигур сложной формы. Выписаны уравнения, описывающие поведение точек фигуры на всех этапах специальной траектории ее движения по коридору. Можно показать, что к такой специальной траектории сводится любое движение ломаной M , проходящей через коридор. Тем самым, получена точная оценка максимальной длины фигуры (ломаной), удовлетворяющей условиям задачи. Найденное значение является (единственным положительным) корнем выписанного многочлена 9-й степени.

Список литературы

1. Gerver, J. L. On Moving a Sofa Around a Corner / J. L. Gerver // *Geometriae Dedicata*. – 1992. – V. 42. – № 3. – P. 267–283.
2. Kallus, Y. Improved upper bounds in the moving sofa problem [Электронный ресурс] / Y. Kallus, D. Romik. – Режим доступа: arXiv:1706.06630 [math.MG]
3. «Список нерешенных геометрических задач» [Электронный ресурс]: – Режим доступа: https://en.wikipedia.org/wiki/List_of_unsolved_problems_in_mathematics#Euclidean_geometry
4. Васильев, Н. Б. Прямые и кривые / Н. Б. Васильев, В. Л. Гутенмахер. – М. : МЦНМО, 2006. – 128 с.
5. Филипов, А. Ф. Сборник задач по дифференциальным уравнениям / А. Ф. Филипов. – М. : Интеграл-Пресс, 1998. – 208 с.

Использование машинного обучения для решения задачи разделения аудиосигнала

Н. А. Селиверстов

Студент магистратуры

А. Ю. Иванков

Доцент

Введение

В настоящее время наблюдается развитие медиасферы: улучшение доступности сети Интернет обеспечивает рост аудитории, а развитие площадок для цифровой дистрибуции позволяет создателям медиаконтента эффективно взаимодействовать с ней. Следствием данных тенденций является повышение спроса на различное программное обеспечение, упрощающее производство медиаконтента. Повышение же вычислительной мощности компьютеров и развитие технологий цифровой обработки сигналов, позволяет разрабатывать эффективные решения, удовлетворяющие этот спрос.

Одним из видов программного обеспечения, которое широко используется для производства медиаконтента, являются программы, плагины и Интернет-сервисы, выполняющие различные действия с аудиосигналами: от редактирования аудиодорожек до генерации аккомпанемента.

Среди перспективных направлений в обработке аудиосигналов можно выделить разработку программных средств, выполняющих разделение аудиосигналов – выделение сигналов, воспроизводимых различными источниками, из смешанного сигнала.

Разделение аудиосигнала может применяться для решения различных задач, например, выделения специфических шумов, однако наиболее популярными сферами использования данной технологии являются музыка и обработка человеческой речи. В области музыки задача разделения аудиосигнала применяется для выделения партий отдельных инструментов [2] из музыкальных произведений, а в области обработки человеческой речи – для выделения речи из зашумленного сигнала [8] или выделения речи отдельных людей из сигнала, содержащего разговор нескольких человек.

Поскольку задача разделения аудиосигнала является нетривиальной, существует большое количество различных подходов к ее решению. Тем не менее в настоящее время для данной цели наиболее широко применяют алгоритмы машинного обучения, что обусловлено успехами, которых они достигли в цифровой обработке сигналов различных размерностей.

В рамках настоящей работы производится обзор алгоритмов машинного обучения, используемых для решения задачи разделения аудиосигнала, рассматриваются их особенности, а также перспективные направления в данной области.

1. Постановка задачи разделения аудиосигнала

Согласно определению, разделение аудиосигнала – это процесс выделения из смешанного аудиосигнала отдельных сигналов, воспроизводимых различными источниками [1]. Таким образом, алгоритм, реализующий разделение аудиосигнала, должен принимать на вход звуковой сигнал, представляющий собой сумму сигналов некоторых источников, а результатом его работы должен быть набор сигналов, как можно сильнее приближенных к тем, из которых был составлен входной сигнал.

Исходя из постановки задачи разделения аудиосигнала, наиболее популярным методом ее решения в настоящее время является фильтрация смешанного сигнала таким образом, чтобы каждому выделяемому источнику соответствовал уникальный фильтр. При использовании данного подхода в большинстве реализаций смешанный сигнал преобразуется в спектрограмму при помощи прямого оконного преобразования Фурье, после чего к ней применяется фильтрация, а из полученных после этого спектрограмм восстанавливаются выделяемые сигналы при помощи обратного оконного преобразования Фурье. Пример разделения аудиосигнала при помощи фильтрации спектрограммы приведен на рисунке 1.

В связи с наличием большого количества различных алгоритмов, реализующих разделение аудиосигналов, существует также большое количество их классификаций, однако в данном разделе следует рассмотреть классификацию алгоритмов разделения по способности работать с произвольным количеством источников. По данному критерию можно разделить алгоритмы на две категории:

- Алгоритмы, способные выделять сигналы только заранее заданных источников;
- Алгоритмы, способные выделять сигналы произвольного количества различных источников.

Так как существует огромное множество различных источников звука, чьи сигналы обладают уникальными амплитудно-частотными характеристиками, в настоящее время не существует универсального решения, способного выделять любые источники, а разработка алгоритмов второй категории является одним из перспективных направлений. Большинство же решений жестко специализированы на выделении определенных источников, которые определяются нуждами предметной области, в рамках которой данные решения существуют.

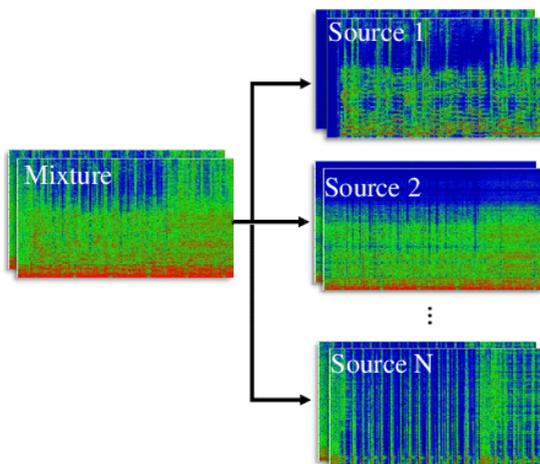


Рис. 1. Разделение аудиосигнала при помощи фильтрации спектрограммы

2. Алгоритмы машинного обучения, используемые для решения задачи разделения аудиосигнала

Простейшим способом фильтрации аудиосигнала является применение к аудиосигналу линейных фильтров. Очевидно, что данный метод неприменим для решения задачи разделения аудиосигнала, так как способен достаточно эффективно разделять лишь весьма простые сигналы, которые не встречаются на практике.

Таким образом, возникает необходимость в алгоритмах, способных формировать такие фильтры, применение которых сможет обеспечить достаточную степень соответствия выделенного сигнала идеальному. Для этих целей хорошо подходят алгоритмы машинного обучения, которые могут итеративно приближаться к идеальному результату разделения.

Существует несколько алгоритмов машинного обучения, не использующих нейронные сети, которые применимы для решения задачи разделения аудиосигнала:

- Неотрицательное матричное разложение (non-negative matrix factorization);
- Сингулярное разложение (singular value decomposition);
- Метод главных компонент (principal component analysis);
- Анализ независимых компонент (independent component analysis) и др.

Несмотря на то, что алгоритмы машинного обучения демонстрируют значительно лучшее качество разделения аудиосигналов по сравнению с использованием линейных фильтров, результаты их работы довольно далеки от таких, которые можно использовать для эффективного решения практических задач.

Следующим шагом в решении задачи разделения стало использование нейронных сетей. Как было отмечено выше, в большинстве случаев при разделении аудиосигнала алгоритмы работают с его спектрограммой, которая является двумерным сигналом – изображением. В настоящее же время существует большое количество обладающих различными моделями и архитектурами нейронных сетей, которые способны эффективно обрабатывать изображения, что и сделало их лучшим вариантом для решения задачи разделения аудиосигнала.

Среди наиболее часто используемых для данных целей разновидностей нейронных сетей следует выделить сверточные нейронные сети (convolutional neural network или CNN) и рекуррентные нейронные сети (recurrent neural network или RNN), а также нейронные сети, обладающие архитектурой encoder-decoder [2, 3, 4, 7, 8]. Однако, кроме вышеуказанных технологий, для разделения аудиосигналов может применяться множество других, например, модули внимания и сети долгой краткосрочной памяти (long short term memory или LSTM) [4].

В нижеследующих разделах рассмотрены алгоритм неотрицательного матричного разложения, как алгоритм, демонстрирующий наилучшее качество разделения аудиосигнала без использования нейронных сетей, а также нейронная сеть U-Net, как часто используемая для разделения сверточная сеть, обладающая архитектурой encoder-decoder.

3. Использование неотрицательного матричного разложения для разделения аудиосигнала

Неотрицательное матричное разложение – это группа алгоритмов машинного обучения, которые разлагают заданную матрицу V на две

матрицы W и H , при этом на все три матрицы накладывается ограничение: они не могут содержать отрицательных элементов [5]. Данное ограничение необходимо в рамках задачи разделения аудиосигнала, так как ни спектрограммы, ни амплитудно-частотные характеристики и точки активации сигналов не могут иметь отрицательных значений.

Существует множество различных типов матричного разложения, однако в настоящей статье рассматриваются лишь его базовые принципы и их применение для разделения аудиосигналов.

Пусть на входе алгоритма задана некоторая матрица $V \in \mathbb{R}^{p \times n}$, а также определен ранг факторизации r , тогда результатом его работы будут являться матрицы $W \in \mathbb{R}^{p \times r}$ и $H \in \mathbb{R}^{r \times n}$ такие, что:

$$V \approx \hat{V} = WH. \quad (1)$$

Одной из наиболее распространенных метрик аппроксимации является норма Фробениуса:

$$\|V - WH\|_F^2 = \sum_{i,j} (V - WH)_{ij}^2. \quad (2)$$

В случае применения неотрицательного матричного разложения для решения задачи разделения аудиосигнала матрица V представляет собой спектрограмму входного смешанного сигнала, ранг факторизации r – количество выделяемых источников, матрица W – амплитудно-частотные характеристики выделенных сигналов, а матрица H – точки активации соответствующих источников [6]. Пример разделения аудиосигнала с использованием неотрицательного матричного разложения приведен на рисунке 2.

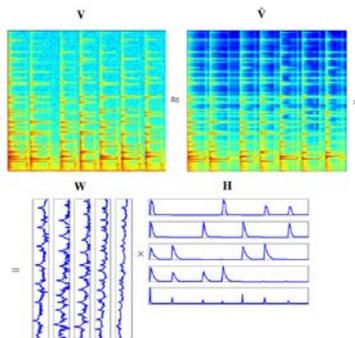


Рис. 2. Разделение аудиосигнала с использованием неотрицательного матричного разложения [6]

4. Использование сети U-Net для разделения аудиосигнала

U-Net – это сверточная нейронная сеть, разработанная для эффективной сегментации биомедицинских изображений [7]. Поскольку фильтрацию спектрограммы смешанного сигнала с целью выделения сигналов различных источников можно представить, как сегментацию данной спектрограммы, сеть U-Net стала также эффективным средством, используемым для разделения аудиосигналов [8].

Сеть U-Net является полносвязной нейронной сетью, обладающей архитектурой encoder-decoder. В роли кодировщиков выступают слои прямой свертки, каждый из которых уменьшает размер изображения вдвое, но удваивает количество каналов. В роли же декодеров выступают слои обратной свертки, при этом для сохранения высокого уровня детализации в сети U-Net добавлены дополнительные пропускные соединения между слоями кодировки и декодировки одного уровня иерархии. Выходной слой сети предоставляет маску, умножение которой на спектрограмму смешанного сигнала позволяет получить спектрограмму выделенного сигнала, из которой он может быть восстановлен.

Схематичное изображение архитектуры сети U-Net, применяемой для разделения аудиосигналов и работающей со спектрограммами, приведено на рисунке 3.

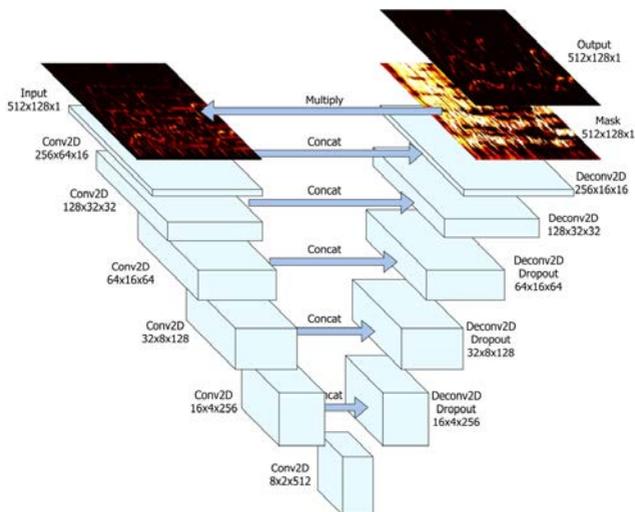


Рис. 3. Архитектура сети U-Net [8]

5. Перспективные направления в области разделения аудиосигнала

Несмотря на то, что в настоящее время, с развитием нейронных сетей, достигнуты значительные успехи в решении задачи разделения аудиосигнала, в данной области проводится большое количество исследований и по развитию существующих алгоритмов и разработке новых [9, 10]. Далее приводится краткий обзор некоторых перспективных направлений таких исследований.

– Поиск и реализация универсальных решений. Как было отмечено ранее в разделе постановки задачи, в настоящее время высокое качество разделения достигается за счет применения специализированных решений, однако создание универсального решения позволит значительно расширить область применения задачи разделения аудиосигналов [11];

– Разработка решений, работающих с самим сигналом, а не его спектрограммой. Этапы преобразования сигнала в спектрограмму и обратно накладывают некоторые ограничения на эффективность разделения, поэтому существуют работы, авторы которых фокусируются на создании решений, способных разделять аудиосигнал, без использования оконных преобразований Фурье [12];

– Переход от фильтрации сигнала к синтезу. Задача разделения аудиосигнала в настоящее время решается при помощи фильтрации смешанного сигнала, однако выделенные таким образом сигналы неизбежно обладают помехами и искажениями, ухудшающими их восприятие. Одним из вариантов решения данной проблемы является добавление этапа генерации сигналов, которые синтезируются на основе выделенных, но лишены искажений [13];

– Разработка и использование метрик, лучшим образом коррелирующих с человеческим восприятием звука. Механизмы восприятия разницы между двумя аудиосигналами у человека отличаются от классических метрик, используемых для этих целей, поэтому перспективным направлением является создание метрик, основанных на особенностях человеческого восприятия звука [14];

– Интеграция задач разделения аудиосигнала и их классификации. Задача разделения аудиосигнала может являться частью комплексных алгоритмов классификации звуковых сигналов. Например, такие алгоритмы могут применяться для формирования субтитров с указанием ролей или генерации партитур или набора редактируемых MIDI-файлов на основе музыкального произведения.

Заключение

В рамках настоящей работы была описана задача разделения аудиосигнала, произведен анализ алгоритмов машинного обучения, используемых для ее решения, а также исследованы перспективные направления в данной области.

Список использованной литературы

1. Audio Source Separation [Электронный ресурс]. – Режим доступа: <https://paperswithcode.com/task/audio-source-separation>
2. Stripe-Transformer: deep stripe feature learning for music source separation / J. Qian [и др.] // EURASIP Journal on Audio, Speech, and Music Processing. – 2023. – DOI: 10.1186/s13636-022-00268-1.
3. Efthymios T. Efficient Networks for Universal Audio Source Separation / T. Efthymios, Z. Wang, P. Smaragdis // IEEE International Workshop on Machine Learning for Signal Processing (MLSP). – 2020. – DOI: 10.1109/MLSP49062.2020.9231900.
4. Schulze-Forster, K. Informed audio source separation with deep learning in limited data settings / K. Schulze-Forster. – Paris: Institut polytechnique de Paris, 2021. – 110 p. – HAL Id: tel-03511031.
5. The why and how of nonnegative matrix factorization [Электронный ресурс]. – Режим доступа: <https://blog.acolyer.org/2019/02/18/the-why-and-how-of-nonnegative-matrix-factorization>
6. Févotte, C. Single-channel audio source separation with NMF: divergences, constraints and algorithms / C. Févotte, E. Vincent, A. Ozerov // Audio Source Separation. – 2018. – HAL Id: hal-01631185.
7. Ronneberger, O. U-Net: Convolutional Networks for Biomedical Image Segmentation / O. Ronneberger, P. Fischer, T. Brox // Medical Image Computing and Computer-Assisted Intervention (MICCAI). – 2015. – DOI: 10.48550/arXiv.1505.04597.
8. Singing voice separation with deep U-Net convolutional networks / A. Jansson [et.al.] // International Society for Music Information Retrieval Conference (ISMIR). – 2017. – ORCID: 0000-0001-8028-9905.
9. Current Trends in Audio Source Separation [Электронный ресурс]. – Режим доступа: https://sigsep.github.io/AES2020_CurrentTrendsInSourceSeparation.pdf
10. Trends in Audio Source Separation [Электронный ресурс]. – Режим доступа: <https://mip-frontiers.eu/2019/11/13/waspaa.html>
11. Universal Sound Separation / I. Kavalerov [et.al.] // IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA). – 2019. – DOI: 10.48550/arXiv.1905.03330.
12. 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.

13. Generative Speech Enhancement Based on Cloned Networks / M. Chinen [et.al.] // IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA). – 2019. – DOI: 10.48550/arXiv.1909.04776.

14. Kastne, T. An Efficient Model for Estimating Subjective Quality of Separated Audio Source Signals / T. Kastner, J. Herre // IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA). – 2019. – DOI: 10.1109/WASPAA.2019.8937179.

Интеграция технологий для реализации электродинамического моделирования

Д. А. Семечев

Студент магистратуры

Д. Н. Борисов

Доцент

Введение

В настоящее время расчет сложных электрических крупных объектов является крайне ресурсозатратной задачей, проблематика которой обусловлена колоссальным различием размеров расчетной области и геометрических размеров объекта в исследуемых моделях. Причем при наличии целого ряда алгоритмов расчета электродинамических структур, необходимо осуществлять выбор оптимального из них. В связи с этим для адекватности получаемых результатов при расчетах электродинамических характеристик необходимо использовать несколько численных методов [1].

Одним из современных лидеров расчётов математического моделирования является Mathworks Matlab. Данный пакет позволяет решать задачи цифровой обработки сигналов, сложные оптимизационные задачи для многих областей техники, в том числе для проектирования систем связи, устройств передачи информации и т. д.

Специалисты по высокоточному электродинамическому моделированию чаще всего для расчетов используют другие пакеты программ, что обусловлено сложностью исследования предметной области. В настоящее время наиболее пригодные для высокоточного математического моделирования электродинамических структур являются программы FEKO, CST Microwave Studio, HFSS [2]. При анализе пригодного инструмента в данной предметной области критерием являются адекватность получаемых результатов и используемые ресурсы (компьютерные мощности, время) [3]. В связи с этим как наиболее пригодный выбран CST Microwave Studio [4, 5]. Программа предназначена для быстрого и точного численного моделирования высокочастотных устройств (антенн, фильтров, ответвителей мощности, планарных и многослойных структур), а также

анализа проблем целостности сигналов и электромагнитной совместимости во временной и частотных областях с использованием прямоугольной или тетраэдральной сеток разбиения.

1. Возможности CST Microwave Studio

В основе работы CST Microwave Studio лежат следующие вычислители:

1. Вычислитель во временной области.
2. Вычислитель в частотной области.
3. Вычислитель собственных мод.
4. Интегральный вычислитель.

Вычислитель во временной области моделирует распространение электромагнитного (ЭМ) поля во времени и в пространстве. С его помощью рассчитывается передача энергии между портами или другими источниками возбуждения и/или свободным пространством. Временной расчётный модуль (солвер) подходит для моделирования большинства высокочастотных объектов: коннекторов, полосковых линий, антенн т. д., позволяя получить результаты во всем частотном диапазоне за один вычислительный цикл средствами преобразования Фурье [5].

Вычислитель в частотной области выполняет расчет трехмерной электродинамической задачи методом конечных элементов (FEM). Причем моделирование осуществляется на определенных частотных точках, а для получения полного спектра используется перебор дискретных частотных точек (сви́пирование по частоте) с возможностью интерполяции. Дискретизация расчетной области может выполняться как гексагональными, так и тетрагональными элементами объемного разбиения. Вычислитель в частотной области универсален, т. е. применим для решения широкого класса задач. При этом необходимо отметить, что анализ электрически малых (высокорезонансных) или узкополосных задач будет выполняться частотным вычислителем с максимальной эффективностью [5].

2. Использование Matlab совместно с CST Microwave Studio

Расчёт основных параметров излучающих структур и их дальнейшая оптимизация является очень ресурсозатратным процессом. Как правило, для реальных задач необходимо использовать мощности сервера или мощных рабочих станций. В связи с этим управлять ресурсами чаще всего приходится дистанционно. Поэтому актуальным является рассмотрение технологий интеграции Matlab и Microwave Studio (MWS) с возможностью расчёта необходимых характеристик в Microwave Studio под управлением Matlab, а также с дальнейшим

использованием результатов в других прикладных пакетах. Для реализации такой интеграций будем использовать фреймворк ActiveX.

ActiveX – фреймворк для определения программных компонентов, пригодных к использованию для программ, написанных на разных языках программирования. Программное обеспечение может собираться из одного или нескольких компонентов, чтобы совместно использовать их функциональность.

Впервые эта технология была внедрена в 1996 году компанией Microsoft как развитие технологий Component Object Model (COM) и Object Linking and Embedding (OLE) и теперь она широко используется в операционных системах семейства Microsoft Windows, хотя сама технология и не привязана к операционной системе.

Анализ документации CST Microwave Studio показывает, что любой ее компонент может выступать в качестве управляемого OLE сервера. OLE – технология связывания и внедрения объектов в другие документы и объекты, разработанная корпорацией Майкрософт.

CST Microwave Studio поддерживает связь между Matlab и CST MWS с использованием языка программирования Microsoft Visual Basic.

К основным базовым функциям, необходимым для работы с интерфейсом ActiveX, можно выделить: `actxserver` – создает локальный или удаленный сервер; `invoke` – вызывает метод для объекта ActiveX [6].

3. Реализация антенны в CST Microwave Studio с помощью Matlab

Рассмотрим на примере создание модели антенны в CST MWS с помощью инструментов Matlab. Для этого необходимо с помощью команд `actxserver` и `invoke` подключиться из Matlab к приложению CST MWS и создать пустой проект. Далее необходимо задать базовые единицы изменения, указанные на рис. 1, и диапазон изменяемых частот (в нашем примере – 1.5-3.5 ГГц). После этого формируется параметризация геометрического описания объекта. Затем необходимо сформировать геометрию антенны посредством работы с компонентами (создание, наложение, удаление компонентов, с возможностью использования логических операции объединения, вычитания, пересечения).

В листинге указан пример кода для создания модели противовеса антенны. Для этого указывается: название и тип компонента, в котором реализована структура (для примера взят `component1`), материал, из которого состоит объект (медь), размеры путем задания координат X, Y, Z, а также функция вызова создания объекта в CST MWS. На рис. 2 показано описание параметров среды, из которых состоит исследуемый объект, а также других используемых физических характеристик (часть из них задается по умолчанию).

```

Geometry = 'mm';
Frequency = 'GHz';
Time = 'ns';
TemperatureUnit = 'Kelvin';
Voltage = 'V';
Current = 'A';
Resistance = 'Ohm';
Conductance = 'S';
Capacitance = 'PikoF';
Inductance = 'NanoH';

```

Рис. 1. Базовые единицы изменения

Листинг

```

Name = 'Groundplane';
component = 'component1';
material = 'Copper (annealed)';
Xrange = [-0.5*Wg 0.5*Wg];
Yrange = [-0.5*Lg 0.5*Lg];
Zrange = [0 Ht];
Cstbrick(mws, Name, component, material, Xrange, Yrange,
Zrange)

```

```

W = 28.45; %Ширина патча
L = 28.45; %Длина патча
Fi = 9; %Толщина разреза
Wf = 1.137; %Длина запитки
Grf=1; %Ширина пустого пространства
Lg = 2*L; %Длина земли
Wg = 2*W; %Ширина земли
Ht = 0.035; %Толщина меди
Hs = 1.6; % Толщина подложки

```

Рис. 2. Параметры антенны

Посредством комбинации выше указанных функций и параметров в Matlab можно получить структуру, показанную на рис. 3.

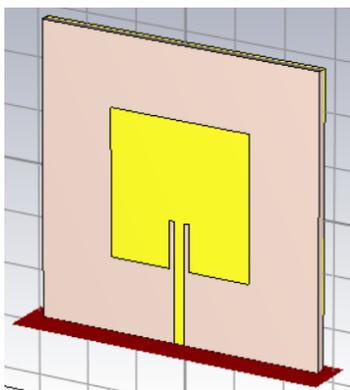


Рис. 3. Сформированная имитационная модель антенны в программе CST MWS

После этапа построения имитационной модели можно произвести расчет её характеристик: согласование с линией питания (S_{11} , КСВН), диаграмму направленности (ДН) антенны, коэффициент усиления (КУ), коэффициент направленного действия (КНД). Такой подход позволяет с помощью Matlab получить необходимые характеристики вычисленные в CST MWS. На рис. 4 показан коэффициент отражения S_{11} полученный в Matlab с использованием интеграции с CST MWS.

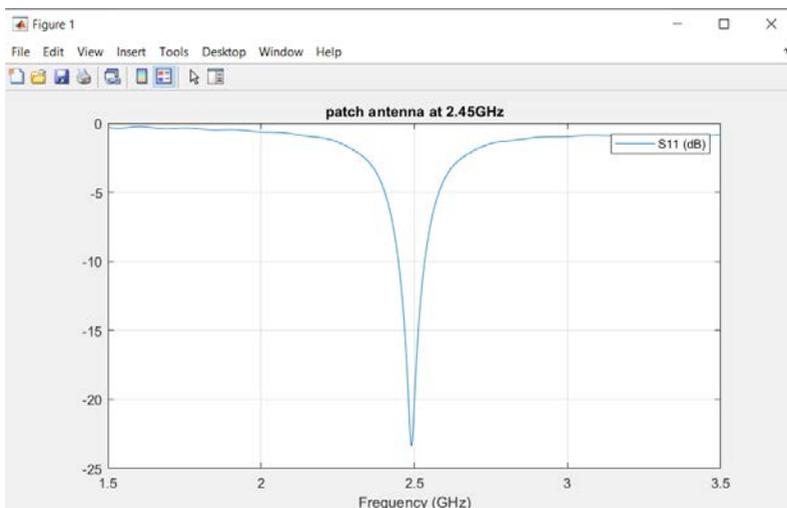


Рис. 4. Зависимость S_{11} от частоты

Заключение

Рассмотрен подход, позволяющий реализовать интеграцию Matlab и CST Microwave Studio, построенный на использовании фреймворка ActiveX и технологии OLE. Данный подход позволяет управлять расчетом электродинамических характеристик в пакете CST MWS с последующей обработкой полученных результатов в Matlab (Simulink Matlab). На основе данного подхода можно осуществлять синтез излучающих структур на основе заданных параметров, используя удаленные вычислительные мощности сервера или рабочей станции.

Список литературы

1. Hybrid Modeling [Электронный ресурс]. – Режим доступа : <https://www.g2.com/glossary/hybrid-modeling-definition>
2. Банков, С. Е. Электродинамика для пользователей САПР / С. Е. Банков, А. А. Курушин, 2009. – Изд. «Электронная книга», издание ИПЭ АН [Электронный ресурс]. – Режим доступа : <http://jre.cplire.ru/jre/library/>
3. Горбачев, А. П. Проектирование печатных фазированных антенных решеток в САПР «CST Microwave Studio» / А. П. Горбачев, Е. А. Ермаков : учебное пособие. – Новосибирск : Изд-во НГТУ, 2008. – 88 с.
4. CST MICROWAVE STUDIO – система моделирования СВЧ трехмерных структур [Электронный ресурс]. – Режим доступа : https://eurointech.ru/eda/microwave_design/cst/CST-MICROWAVE-STUDIO.shtml
5. Официальный сайт компании CST [Электронный ресурс]. – Режим доступа : www.cst.com.
6. Using CST with Matlab [Электронный ресурс]. – Режим доступа : https://www.researchgate.net/publication/330520619_Using_CST_with_Matlab

Влияние расположения узлов расчётной области на решения уравнений в частных производных методом коллокаций

Е. П. Скородед

Студент магистратуры

А. В. Атанов

Доцент

Введение

При моделировании различных явлений в физике, химии, биологии, экономике и в других областях возникает необходимость решать уравнения в частных производных. Как правило, такие уравнения решаются численными методами. Большую популярность последние годы приобрели бессеточные методы – в частности, основанные на использовании радиальных базисных функций. Одним из наиболее популярных среди этих методов является метод коллокаций Кансы [1, 2].

В методах радиальных базисных функций (РБФ) распределение и количество точек оказывают значительное влияние на точность, стабильность и вычислительные затраты. Существует много разных способов расположения узлов, используемых для решения уравнений с помощью указанного метода. Целью работы является изучение влияния расположения узлов расчётной области на решения уравнений в частных производных методом коллокаций.

1. Метод коллокаций Кансы

Функция $\varphi: R^d \rightarrow R$, значение которой в любой точке $x \in R^d$ зависит только от расстояния до некоторой фиксированной точки $c \in R^d$, то есть $\varphi = \varphi(\|x - c\|)$, называется радиальной базисной функцией (РБФ); c – центр РБФ.

В работе используется мультиквадратичная РБФ

$$\phi(r) = \sqrt{1 + (\varepsilon r)^2}, \text{ где } \varepsilon > 0 \text{ и } r = \|x - c\|.$$

Рассмотрим метод коллокаций Кансы, в котором для решения уравнений в частных производных используются РБФ.

Рассматривается следующая краевая задача:

$$Lu(x) = f(x), \quad x \in \Omega, \quad (1)$$

$$Bu(x) = g(x), \quad x \in \Gamma, \quad (2)$$

где L – это дифференциальный оператор, B – граничный дифференциальный оператор, f и g – известные функции, Ω – вычислительная область, Γ – граница области Ω .

Пусть $\{x_j, f(x_j)\}_{j=1}^{N_i}$ – различные точки из Ω в количестве N_i штук и $\{x_j, g(x_j)\}_{j=N_i+1}^N$ – граничные точки в количестве N_b . Общее число точек (узлов коллокаций) равно $N = N_i + N_b$.

Метод коллокаций предполагает сведение краевой задачи (1)–(2) к дискретной задаче следующего вида:

$$Lu(x_k) = f(x_k), \quad k = 1, 2, \dots, N_i,$$

$$Bu(x_k) = g(x_k), \quad k = N_i + 1, \dots, N.$$

При этом функция решения аппроксимируется с помощью линейной комбинации РБФ:

$$\hat{u}(x) = \sum_{j=1}^N \alpha_j \varphi(\|x - x_j\|), \quad (3)$$

где $\{\alpha_j\}_{j=1}^N$ – неизвестные коэффициенты, подлежащие определению.

Применяя операторы L и B к (3), получаем:

$$L\hat{u}(x) = \sum_{j=1}^N \alpha_j L\varphi(\|x - x_j\|), \quad (4)$$

$$B\hat{u}(x) = \sum_{j=1}^N \alpha_j B\varphi(\|x - x_j\|). \quad (5)$$

Подставляя (4) в (1) и (5) в (2), получаем уравнения коллокаций:

$$\sum_{j=1}^N \alpha_j L\varphi(\|x - x_j\|) = f(x), \quad x \in \Omega,$$

$$\sum_{j=1}^N \alpha_j B\varphi(\|x - x_j\|) = g(x), \quad x \in \Gamma.$$

В результате получаем систему линейных уравнений с неизвестными α_j :

$$\sum_{j=1}^N \alpha_j L \varphi(\|x_k - x_j\|) = f(x_k), \quad k = 1, 2, \dots, N_i,$$

$$\sum_{j=1}^N \alpha_j B \varphi(\|x_k - x_j\|) = g(x_k), \quad k = N_i + 1, \dots, N.$$

Решив эту систему и подставив найденные значения α_j в (3), получим аппроксимированную функцию решения $\hat{u}(x)$.

2. Распределения узлов расчётной области

Перейдём к вариантам расположения узлов в расчётной области, которые были использованы в работе. Самым простым способом распределения узлов является регулярное с добавлением дополнительных узлов вблизи границы, представленное на рис. 1а. Вторым способом является триангуляция Делоне, показанная на рис. 1б. Другие типы расположения узлов, использованные в работе: узлы Холтона (рис. 1в), узлы Соболя (рис. 1г) и узлы Падуи (рис. 1д).

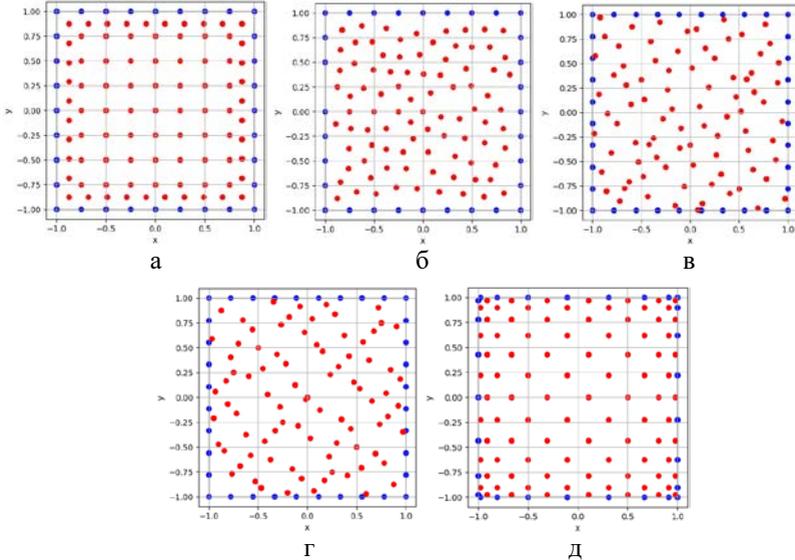


Рис. 1. Распределения узлов расчётной области: а – регулярное, б – триангуляция Делоне, в – Холтона, г – Соболя, д – Падуи

Точки Падуи – первый и пока единственный пример набора точек, обеспечивающих минимальный рост константы Лебега (константа

Лебега показывает, насколько хорош интерполянт по сравнению с полиномом наилучшего приближения). Координаты точек Падуи вычисляются по следующим формулам:

$$x_m = \cos\left(\frac{(m-1)\pi}{n}\right), \quad y_k = \begin{cases} \cos\left(\frac{(2k-1)\pi}{n+1}\right), & m = 2r + 1, \\ \cos\left(\frac{(2k-2)\pi}{n+1}\right), & m = 2r, \end{cases}$$

в которых $1 < k < \frac{n}{2}$ и $n > 0$.

3. Программная реализация

В ходе выполнения работы на языке Python [3] были реализованы библиотеки для генерации расчётных узлов пяти вариантов распределений и для решения уравнений в частных производных методом коллокаций Кансы.

Перечислим основные шаги разработанного алгоритма.

1. Определение разработанных библиотек:
 - `points_generation.py` – библиотека для генерации расчётных узлов пяти вариантов распределений;
 - `kansa.py` – библиотека, реализующая метод коллокаций Кансы.
2. Генерация точек расчётных областей для всех рассматриваемых распределений.
3. Решение уравнения в частных производных методом коллокаций с подбором оптимального параметра ε .
4. Расчёт среднеквадратичной ошибки (RMSE) по формуле (6) для разного количества точек для каждого распределения и построение графика зависимости.

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N |u(z_i) - \hat{u}(z_i)|^2}. \quad (6)$$

4. Пример решения уравнения

Рассмотрим уравнение Пуассона:

$$u_{xx} + u_{yy} = f(x, y),$$

$$u|_{\partial\Omega} = g(x, y).$$

Для проверки выберем следующее тестовое решение рассматриваемого уравнения:

$$u(x, y) = 6x^2y - 2y^3 - 3x^2 + \sin\left(\frac{x}{2}\right).$$

На рис. 2 изображена поверхность $u(x, y)$, то есть график точного решения, и значения функции решения, полученные методом коллокаций для прямоугольной (рис. 2а) и круговой (рис. 2б) областей.

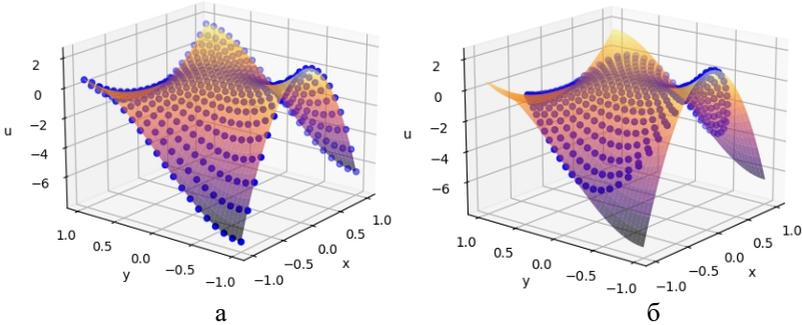


Рис. 2. Точное решение уравнения и точки, соответствующие численному решению в прямоугольной (а) и круговой (б) областях.

На рис. 3 показана зависимость ошибки от способа расположения узлов и их количества в прямоугольной области. В данном случае узлы Падуи показали наилучший результат.

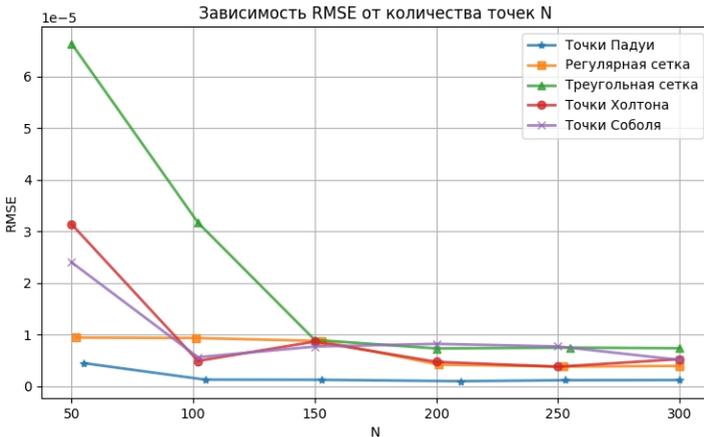


Рис. 3. Зависимость RMSE от способа расположения узлов и их количества в прямоугольной области

Далее мы снова рассматриваем зависимость ошибки от способа расположения узлов и их количества, но уже в круговой области. В данном случае лучший результат показало распределение Холтона, однако узлы Падуи выигрывают при малом количестве узлов (рис. 4).

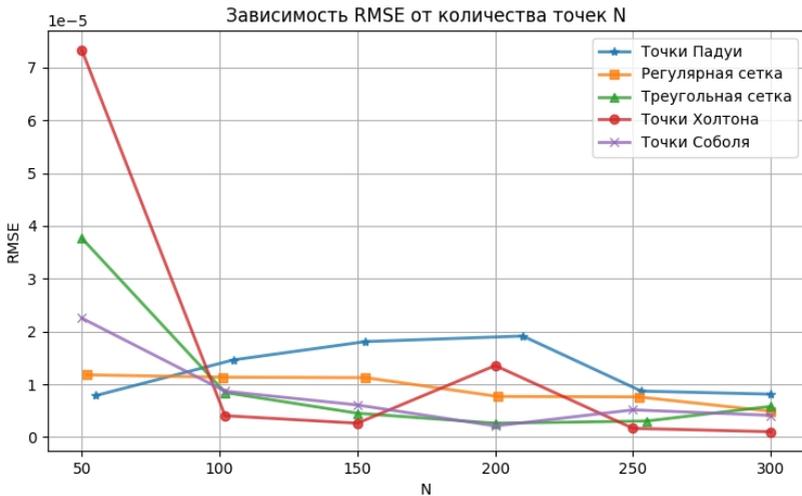


Рис. 4. Зависимость RMSE от способа расположения узлов и их количества в круговой области

Заключение

Данная статья посвящена изучению влияния распределения узлов расчётной области на решения уравнений в частных производных методом коллокаций, а также разработке и реализации программного модуля на языке программирования Python для решения уравнений в частных производных методом коллокаций с возможностью выбора типа расположения узлов расчётной области и их количества. Описаны алгоритм решения уравнений в частных производных методом коллокаций и создание программного модуля. В итоге данной разработки получился полностью рабочий программный модуль, реализованный в виде библиотек и тестовых задач. Использование данного решения позволяет решать уравнения в частных производных методом коллокаций с возможностью выбора типа распределения узлов расчётной области, их количества и формы границы.

Список литературы

1. Kansa, E. J. Multiquadrics – a scattered data approximation scheme with applications to computational fluid dynamics – I / E. J. Kansa // *Comput. Math. Applic.* – 1990. – V. 19, № 8/9. – P. 127–145.
2. Biancolini, M. E. *Fast Radial Basis Functions for Engineering Applications* / M. E. Biancolini; Springer, 2018. – 372 p.
3. Лутц, М. Изучаем Python : авторитетный курс объектно-ориентированного программирования / Марк Лутц. – Т. 1, 5-е изд. : Пер. с англ. – СПб. : ООО «Диалектика», 2019. – 832 с.

Использование метода максимумов модуля для оценки характерных значений параметров электрокардиографических сигналов

А. А. Сурков

Студент магистратуры

Е. А. Киселев

Доцент

Введение

В настоящее время продолжается широкое внедрение цифровых технологий в медицинскую область. Одно из важных направлений связано с созданием и совершенствованием систем автоматизированной обработки результатов электрокардиографических исследований (ЭКГ).

Данная работа посвящена решению одной из проблем, связанных с аппроксимацией форменных элементов ЭКГ. В серии недавних статей [1, 2] была предложена математическая модель, основанная на функции Гаусса, которая помогла повысить уровень сжатия сигналов ЭКГ без потери клинически значимой информации [3]. Трудность заключается в алгоритме расчета параметров этой модели: имеется возможность автоматически определить положения зубцов ЭКГ, а также их амплитуды, но не характерную ширину. Для решения этой проблемы мы предлагаем проводить предварительный статистический анализ сигнала ЭКГ, используя для нахождения параметра ширины метод максимума модулей. Данный подход был недавно успешно применен для аналогичных целей при исследовании электроэнцефалограмм (ЭЭГ) [4].

1. Теоретическая часть

Пусть s_n , $n = 0, 1, \dots, N-1$, – исследуемый цифровой сигнал ЭКГ, состоящий из N значений. Опишем коротко, следуя [4], метод максимумов модуля. Он состоит из следующих этапов.

1. Сигнал разбивается на блоки длиной K точек. Общее количество таких блоков обозначим L .

2. В пределах каждого блока с помощью среднего значения сигнала, взятого по 5 крайним точкам слева и справа, ищется изолиния

и вычитается. Сигнал, который получается после удаления изолинии, мы обозначим как \tilde{s}_n .

3. В пределах каждого блока вычисляется свертка сигнала \tilde{s}_n с функцией Гаусса

$$W(a_i, b_j) = \frac{1}{\sqrt{a_i}} \sum_{k=0}^{K-1} \tilde{s}_k \exp\left(-\frac{(k-b_j)^2}{2a_i^2}\right), \quad (1)$$

где $a_i = (i-1)h$, $b_j = jh$, $i, j = 0, 1, \dots, K-1$, h – шаг дискретизации.

4. При каждом значении b_j для каждого фрагмента ЭКГ отыскивается значение a_i , при котором достигается максимальное значение величины $|W(a_i, b_j)|$.

5. Для каждого значения a_i рассчитывается количество случаев $m(a_i)$, когда на него выпадал максимум величины $|W(a_i, b_j)|$.

6. По формуле

$$n(a_i) = \frac{m(a_i)}{m}, \quad (2)$$

вычисляется относительная частота выпадения максимума $|W(a_i, b_j)|$ на a_i . Здесь $m = KL$. По максимумам в распределении $n(a_i)$ можно судить о характерных значениях параметра ширины функции Гаусса.

Модель, предложенная в [2], заключается в следующем. Фрагмент ЭКГ, содержащий какие-либо зубцы, аппроксимируется комплексом из $P = 1, 2, 3$ функций Гаусса:

$$\tilde{s}_n = \sum_{p=1}^P A_p \exp\left(-\frac{(nh - M_p)^2}{2\sigma^2}\right), \quad (3)$$

где A_p , M_p , σ – неопределенные параметры. Величину σ мы предлагаем оценить с помощью метода максимумов модуля, описанного выше. Тогда оставшиеся параметры можно найти с помощью системы уравнений, приведенной в [2]. Соответствующие соотношения довольно громоздки, поэтому мы не будем подробно здесь их описывать.

2. Вычислительные эксперименты

Для исследования мы брали образцы ЭКГ, содержащие $N = 8192$ отсчетов с шагом дискретизации $h = 1$ мс. Каждый образец разбивался на $L = 16$ блоков длиной $K = 256$ отсчетов.

На рис. 1 и 2 представлены несколько примеров реализации метода максимумов модуля. В обоих случаях наблюдается серия ярко выраженных максимумов частоты $n(a_i)$. На рис. 1 первые три располагаются в точках $a=10$, $a=34$, $a=64$, а на рис. 2 в точках $a=15$, $a=37$, $a=117$. Мы полагаем, что это вызвано тем, что в ЭКГ наблюдаются форменные элементы разных типов, а следовательно, и разного характерного масштаба: Р-волна, Т-волна и QRS-комплекс. Наименьшей шириной, как правило, обладают зубцы QRS-комплекса [5]. Им тогда должен отвечать первый по счету максимум на рис. 1 и 2. В этом случае следующие два максимума связаны с Р- и Т-волной, соответственно.

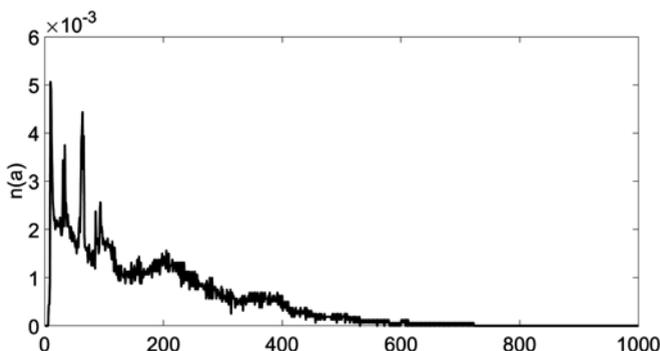


Рис. 1. Относительная частота (2) для первого образца ЭКГ

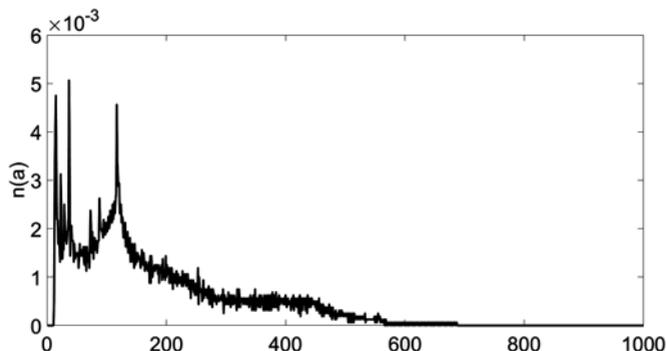


Рис. 2. Относительная частота (2) для второго образца ЭКГ

Далее на рис. 3 и 4 приведены несколько примеров обработки QRS-комплекса с помощью метода моментов с применением описанной выше

технологии предварительной оценки параметра σ . Качество восстановления практически не уступает результатам статей [2, 3], где параметр выбирался вручную: в обоих случаях отклонение не превышает 1% по среднеквадратичной норме.

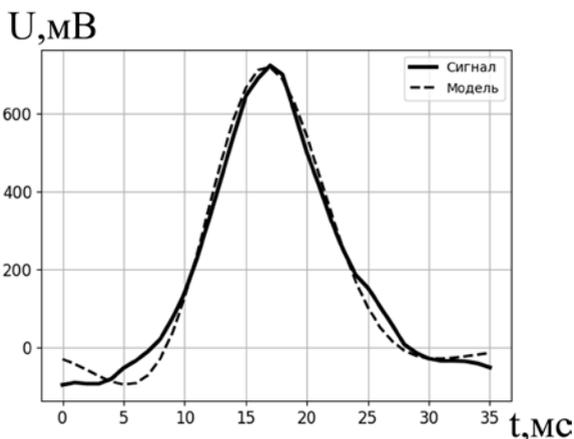


Рис. 3. Моделирование QRS-комплекса, полученного из первого образца ЭКГ

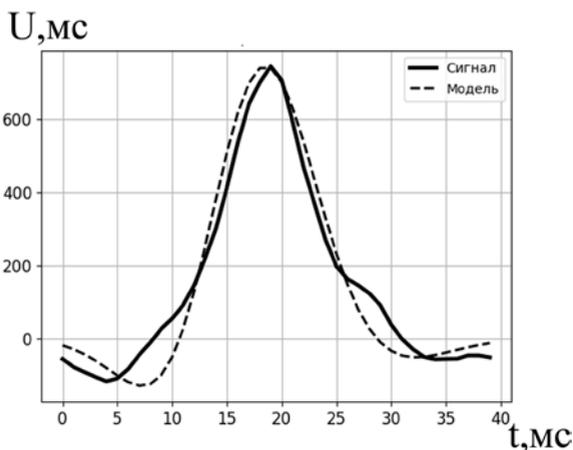


Рис. 4. Моделирование QRS-комплекса, полученного из второго образца ЭКГ

Заключение

В данной работе предложено усовершенствование алгоритма моделирования форменных элементов ЭКГ, предложенного в статье [3]. Ключевой идеей является применение метода максимумов модуля для предварительной оценки характерной ширины зубцов ЭКГ. Прежде всего, отметим, что в отличие от ЭЭГ, в распределении частоты $n(a_i)$ наблюдается не один, а целая серия максимумов. Мы считаем, что это вызвано различием в масштабах Р-, Т-волны и QRS-комплекса. Отметим, что, согласно результатам проведенных экспериментов, наиболее сильно отличается у разных образцов именно характерная длительность Т-волны. Этот параметр является очень важным в клинической диагностике [5], поэтому в будущем метод максимумов модуля может стать новым эффективным инструментом автоматизированного анализа ЭКГ.

Список литературы

1. Никифоров П. Л. Модель электрокардиографического сигнала на основе совокупности колокольных импульсов / П. Л. Никифоров // Вестник молодых ученых. Серия: Технические науки, 1998. – № 1. – С. 64–68.
2. Насер Нихад. Аппроксимация пиков в электрокардиограммах комбинацией сдвигов функции Гаусса / Нихад Насер // Системы управления и информационные технологии, 2015. – № 1(59). – С. 77–80.
3. Киселев Е. А. Комбинированный алгоритм сжатия сигнала электрокардиограммы с помощью всплесков Добеши и функции Гаусса / Е. А. Киселев, Нихад Махмуд Насер, Е. Г. Супонев // Системы управления и информационные технологии, 2017. – № 3(69). – С. 53–56.
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). – Moscow : IEEE, 2022. – P. 1–4.
5. Зудбинов Ю. И. Азбука ЭКГ / Ю. И. Зудбинов. – Ростов-на-Дону : «Феникс», 2003. – 160 с.

Использование алгоритмов машинного обучения для планирования рабочей нагрузки высокопроизводительных вычислительных систем

К. А. Турченко

Студент магистратуры

С. В. Борзунов

Доцент

Введение

В настоящее время высокопроизводительные вычисления и обработка больших данных являются одними из наиболее важных направлений в информационных технологиях. Это связано с потребностью проводить моделирование сложных систем, анализировать криптографические данные и решать другие ресурсоемкие задачи. Одним из важных способов организации высокопроизводительных вычислений являются расчеты на суперкомпьютерных системах.

Суперкомпьютерные центры часто создают в крупных университетах, где в них над своими проектами будут работать множество исследователей. По этой причине для каждого пользователя высокопроизводительных вычислительных систем необходимо выделять некоторое количество вычислительных ресурсов на определенное время. Для этого существуют различные планировщики и менеджеры управления ресурсом. В качестве примера приведем планировщик SLURM, который используется на вычислительном кластере факультета компьютерных наук Воронежского государственного университета [1].

В ходе работы на вычислительном кластере было замечено, что стандартный планировщик не всегда оптимально распределяет ресурсы и формирует очередь задач. Целью настоящего исследования является разработка алгоритма на основе методов машинного обучения для составления очереди задач и создания расписания для их выполнения. В работе используется «глубокая Q-сеть», которая является искусственной нейронной сетью с целевой Q-функцией для оптимизации [2]. Такой подход хорошо зарекомендовал себя при решении задач составления расписаний.

1. Модель глубокого Q-обучения

Прежде чем рассматривать данный метод, мы должны определить набор терминов, которые будут использованы в дальнейшем:

– Состояние (State): Состояние S указывает на конкретное положение в среде.

– Действие (Action): Действие A описывает, что делает агент, находясь в определенном состоянии S .

– Награды: Агент будет получать определенные награды (положительные или отрицательные) в зависимости от совершаемых действий.

– Эпизоды: Агент не может совершать действия после наступления терминального (конечного) состояния.

– Q-значения (Q-values): Мера качества действия A , выполненного в конкретном состоянии S .

Модель глубокого Q-обучения ищет оптимальную функцию $Q(A, S)$. Этот метод основан на совмещении Q-обучения и нейронной сети.

При глубоком Q-обучении прошлый опыт сохраняется в памяти, а будущее действие зависит от выходных данных Q-сети. Именно так Q-сеть вычисляет значение ценности в состоянии $S(t)$, где t – дискретное время. Точно так же целевая сеть (нейронная сеть) определяет значение Q для следующего состояния $S(t+1)$ для стабилизации обучения.

Алгоритм глубокого Q-обучения опирается на нейронные сети и Q-обучение с подкреплением. В этом случае нейросеть хранит опыт в кортежах вида <Состояние, Следующее состояние, Действие, Награда>, которые используются для определения значения Q .

Случайная выборка предыдущих данных повышает стабильность обучения нейронной сети. В результате глубокое Q-обучение использует другую концепцию для повышения производительности агентов – воспроизведение опыта или сохранение опыта из прошлого.

Вычисление потерь становится проще, когда известна квадратичная разница между целевым и прогнозируемым значениями Q .

Для нашей задачи нецелесообразно представлять Q-функцию в виде таблицы для каждой комбинации S и A . Вместо этого мы будем использовать нейронную сеть для аппроксимации функции с параметром, которая оценивает Q-значения. Общая схема данного подхода показана на рис. 1. Это можно сделать, обновляя значения Q согласно уравнению Беллмана [2]:

$$Q^{new}(S_t, A_t) = (1 - \alpha)Q(S_t, A_t) + \alpha(R_t + \lambda \max_a Q(S_{t+1}, a)), \quad (1)$$

где S – состояние или наблюдение; A – действие агента; R – награда за предпринятое действие; t – дискретное время; α – скорость обучения; λ – коэффициент дисконтирования, из-за которого вознаграждения со временем теряют свою ценность.



Рис. 1. Общая схема глубокого Q-обучения

2. Применение модели глубокой Q-сети для составления расписания задач на кластере

Для того чтобы применить алгоритм, нам необходимо реализовать среду для обучения. Будем использовать автоматическую генерацию пула задач с определенными ограничениями на количество задач и время их выполнения [3].

Модель кластера представляет собой двумерный массив NumPy, каждый элемент которого обозначает занятость вычислительного ресурса в определенный момент времени (временная координата определяется строками массива).

Для эмулятора кластера был реализован стандартный алгоритм составления расписания в пакете SLURM, который называется BackFill. Данный алгоритм последовательно рассматривает входные задачи в порядке приоритета и распределяет их на ближайшие свободные вычислительные ресурсы.

Агент получает награды в процессе обучения. Награда

$$R(A) = 1 \quad (2)$$

выдается в случае, если наше действие (размещение задачи на позицию в расписании) привело к меньшему суммарному времени обработки всех предыдущих задач, чем стандартный алгоритм SLURM. В противном случае агент получает награду

$$R(A) = -1. \quad (3)$$

Процесс обучения агента прекращается при достижении агентом определенного количества суммарных вознаграждений, равных 10. Количество таких вознаграждений можно задать через параметр алгоритма, который напрямую влияет на скорость обучения.

На рис.2 показано время выполнения групп задач. Мы видим, что для многих групп задач обученная модель справилась лучше, чем стандартный алгоритм BackFill.

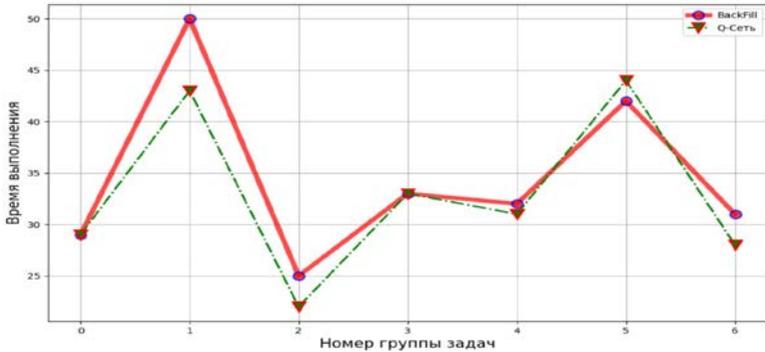


Рис. 2. Сравнение времени выполнения списка из 15 групп по 10 задач

Теперь сравним полученные расписания для отдельно взятой группы задач (рис. 3, 4):

task_id	time	used_threads	current_cpu_time
0	1	4	160
1	2	6	38
2	3	13	48
3	4	9	120
4	5	13	153
5	6	13	118
6	7	7	157
7	8	7	156
8	9	4	95
9	10	9	239

Рис. 3. Расписание на основе алгоритма BackFill

Мы видим, что суммарное время выполнения уменьшилось на 2 единицы в случае составления расписания задач на основе обученной модели рис. 4., по сравнению с алгоритмом BackFill рис. 3.

task_id	time	used_threads	current_cpu_time
0	4	9	120
1	7	7	157
2	8	7	156
3	3	13	48
4	6	13	118
5	10	9	239
6	5	13	153
7	9	4	95
8	1	4	160
9	2	6	38

Рис. 4. Расписание на основе обученной модели

Заключение

С использованием методов машинного обучения разработан алгоритм решения задачи распределения вычислительных ресурсов и составления расписания задач на вычислительном кластере. В ряде модельных примеров достигнуты лучшие результаты по сравнению со стандартным встроенным алгоритмом планирования пакета SLURM. В таких случаях использование обученной модели позволяет выполнять наборы задач за меньшее время, при этом не используя алгоритм полного перебора, который находит оптимально распределение задач, но при этом имеет высокую временную сложность.

Список литературы

1. Кургалин, С. Д. Using the resources of the Supercomputer Center of Voronezh State University in learning processes and scientific researches / С. Д. Кургалин, С. В. Борзунов // Труды международной конференции «Суперкомпьютерные дни в России», 24-25 сентября 2018 г. – Москва : Изд-во МГУ, 2018. – С. 972-977.
2. Sutton, Richard S. Reinforcement Learning / Richard S. Sutton, Andrew G. Barto : The MIT Press, 2018.
3. Баранов, А. В. Методы и средства моделирования системы управления суперкомпьютерными заданиями / А. В. Баранов, Д. С. Ляховец // Программные продукты и системы. – 2019. – Т. 32, № 4. – С. 581–594.

Разработка мобильного графического редактора с функцией распознавания контуров на платформе ios

Р. П. Филипенко

Студент магистратуры

Е. Н. Десятерикова

Профессор

Введение

С уходом зарубежных компаний с рынка появилась острая необходимость в замещении программного обеспечения. Исходя из этого, необходимо рассматривать процесс создания программного обеспечения, которое могло бы импортозаместить ушедшие продукты. Мной предложены следующие этапы разработки мобильного приложения.

Одним из этапов для успешной реализации графического приложения, рассчитанного на любого типа пользователей, является составление технической документации с описанием того, что приложение должно уметь делать. Здесь же проводится верхнеуровневое проектирование взаимодействия мобильного приложения с какой-то удаленной базой данных (или бэкендом). В нашем случае в качестве такой базы будет служить Yandex Object Storage [1].

1. Этапы разработки и проектирования

Разработка дизайна мобильного приложения. На данном этапе стоит продумать такой UI, чтобы даже неподготовленный пользователь мог использовать мобильное приложение. Для этого в приложение необходимо заложить функционал онбординга пользователя. Также приложение должно содержать в себе красивые анимации, чтобы пользователю было приятнее им пользоваться. Стоит проработать использование виброотклики, проанализировав ожидания пользователя от мобильного приложения на то или иное действие.

Создание и оформление репозитория. Для того, чтобы приложение можно было поддерживать и развивать, необходимо контролировать его версии. Также в приложении обязательно должна присутствовать техническая документация с описанием переиспользуемых

компонентов, инструкций по сборке и запуску приложения, описания архитектуры и т.д.

Проектирование дизайн-системы. Для интегрирования готовых элементов, которые нарисовал дизайнер, необходимо разработать дизайн-систему. С точки зрения программирования стоит оформить его отдельным SPM пакетом для дальнейшего переиспользования. Также дизайн-система решает проблемы унификации элементов интерфейса. Дизайнер обязан консультироваться с разработчиком такой системы, что ведет к более тесной интеграции различных ролей при создании приложения [2].

Проектирование сетевого слоя мобильного приложения. Для обмена информацией с бэкендом посредством АПИ, описанного в техническом задании, необходимо создать слой, отвечающий за отправку запросов и обработку ответов с сервера. Предполагается реализация как SPM-пакет. С точки зрения инструментов проектирования можно использовать: - встроенный класс URLSession; - сторонние библиотеки, например, Alamofire С точки зрения архитектуры предполагается поделить использовать такую логическую абстракцию, как Node (или нода). Каждая нода отвечает за определённую роль - составление схемы запроса - инъект заголовков - инъект тела запроса - отправка готово запроса - получение результата - парсинг результата - обработка ошибок Количество нод может быть добавлено, или уменьшено [3].

Проектирование архитектуры мобильного приложения. Так как базовая архитектура является фундаментом приложения, на котором будут строиться дальнейшие логические слои, необходимо выбрать самое удобное решение. Я решил использовать MVVM вместе с координаторами, внутри которых находится роутер. MVVM расшифровывается как Model-View-ViewModel. Такая архитектура позволяет быстро реализовывать новые экраны. На данном этапе необходимо добавить их кодогенерацию с помощью SwiftGen. Такая архитектура позволяет держать в модели всю бизнес-логику и проводить с ней манипуляции. View позволяет отображать все элементы графического интерфейса, исключая в этом слое какую-то логику, кроме логики отображения. ViewModel позволяет провести манипуляции над бизнес-данными так, чтобы они были готовы для презентации. То есть во ViewModel находится презентационная логика. Для навигации между экранами приложения предполагается использовать координаторы. Через них же удобно передавать данные экранами. Вся логика показа модуля должна быть инкапсулирована в роутер, чтобы не допустить утечки UIKit - ых методов в координаторы.

2. Используемые библиотеки

Инициализация проекта Создание проекта для iPhone, подключение всех SPM-пакетов или дополнительных библиотек, например, OpenCV. Написание Make File, по возможности подключение и настройка CI/CD.

Изучение базовых библиотек Metal, CoreAnimation, CoreGraphics, UIKit, Accelerate С помощью Metal мы можем получить низкоуровневый доступ к рендеру изображений на видеокарте. CoreAnimation, CoreGraphics предоставляют удобное АПИ для работы с изображениями и его слоями. UIKit позволяет удобно сверстать программный интерфейс [4].

Изучение и применение библиотек Vision, CoreML, OpenCV. Необходимо изучение возможностей каждой из этих библиотек для детектирования контуров на изображении. Стоит использовать стандартные библиотеки Vision и CoreML, и если их возможностей уже не будет хватать, стоит обратить внимание на OpenCV.

3. Разделы приложения

После составления технической документации у нас выделятся основные разделы приложения, но некоторые уже можно назвать и сейчас - это холст для рисования и настройки приложения. На этом этапе необходимо спроектировать и разработать экран, на котором будет присутствовать палитра инструментов, а также область, где можно будет рисовать. В настройках приложения нужно отобразить табличную форму, где будет переходит к экранам, на котором можно будет регулировать пользовательский выбор той или иной функциональности.

Подключение к хосту функционала распознавания контуров. На данном этапе необходимо детектить контуры на холсте. Необходимо предусмотреть обработку ошибок, если контур не был найден. Необходимо добавить возможность показать, какой контур выбран, или отменить выбор.

Использование контуров для реализации дополнительной функциональности. Например, можно перетаскивать контуры, переворачивать их, видоизменять.

Покрытие кодовой базы тестами. Предполагается, что код будет писаться по TDD (вначале тест - потом функционал). Это позволит запросить разработку на “длительной дистанции”.

Заключение

Целью моей работы было разработать и выпустить конкурентоспособное приложение для рынка графического редактирования. Для этого были составлены этапы разработки,

спроектированы дизайн-системы, привлечены базовые и дополнительные библиотеки, а так же будет добавлена камера глубин для реализации размытия или замены элементов изображений.

Список литературы

1. Компьютерные сети / Эндрю Таненбаум // Изд-во Вашингтонского университета, Вашингтон. - 2011. - 12-13 стр.
2. Чистый код / Роберт Мартин // Изд-во: Библиотека программиста, Санкт - Петербург - 2020. - С. 164-170.
3. SwiftUI By Tutorials / Antonio Bello, Bill Morefield // Razeware LLC, Hawksbill Rd McGaheysville, Virginia. - 2019. - С. 120-142.
4. Основы разработки приложений под iOS, iPadOS и macOS / Усов В. // Изд-во: Питер, г. Санкт-Петербург. - 2019. - С. 52-56.

Извлечение данных о технологии проведения химических реакций из неструктурированных PDF-файлов

П. А. Хоменко

Студент магистратуры

В. В. Гаршина

Доцент

Введение

Научная литература, например, журнальные статьи и патенты является одним из важнейших источником информации для специалистов из области химии в поиске способов проведения конкретных химических реакций или синтетических процедур, представляющих интерес. Существующие базы данных реакций обычно заполняются специалистами путем ручной абстракции, что отнимает много времени и сил, особенно с учетом экспоненциального роста специализированной литературы за последние годы. Эта проблема мотивирует разработку автоматизированных методов извлечения информации о реакциях из неструктурированных текстовых данных.

Для того чтобы получить значимые данные из текста, необходимо следовать методу, который называется Интеллектуальный анализ текста или Text Mining, что представляет собой одну из разновидностей методов Data Mining и подразумевает процессы извлечения знаний и высококачественной информации из текстовых массивов.

Ознакомление с технологией Text Mining необходимо начинать с изучения задачи извлечения именованных сущностей (Named-entity recognition, NER).

Данное исследование сосредоточено на разработке автоматизированных методов извлечения реакций из химической литературы. В качестве целевого источника информации рассматриваются журнальные публикации в формате PDF, которые отражают последние достижения в области химии.

1. Рассматриваемые номенклатуры

ИЮПАК (IUPAC) — Международный союз теоретической и прикладной химии (International Union of Pure and Applied Chemistry). В качестве одного из основных направлений своей деятельности ИЮПАК

разрабатывает Рекомендации по установлению однозначной, единообразной и последовательной номенклатуры и терминологии для конкретных научных областей.

При использовании систематической номенклатуры ИЮПАК, называя соединения, следует учитывать местоположение атомов углерода в молекуле и структурных элементов: выделяется базовая основа молекулы, в которой атомы водорода замещены на какие-либо структурные звенья или атомы. [2]

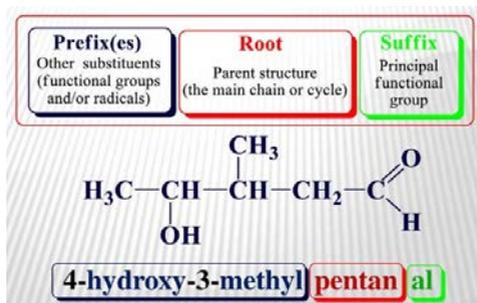


Рис. 1. Примеры образования наименования молекул согласно номенклатуре IUPAC

SMILES (Simplified Molecular Input Line Entry System) - это линейная номенклатура (метод с использованием печатных символов) для ввода и представления молекул и реакций.

SMILES - это настоящий язык, хотя и с простым словарным запасом (наименование атома и символы связи) и только несколько грамматических правил. SMILES, в свою очередь, может использоваться в качестве «слов» в словаре других языков предназначен для хранения информации о химических веществах и химических реакциях). [3]

SMILES	Name	SMILES	Name
CC	ethane	[OH3+]	hydronium ion
O=C=O	carbon dioxide	[2H]O[2H]	deuterium oxide
C#N	hydrogen cyanide	[235U]	uranium-235
CCN(CC)CC	triethylamine	F/C=C/F	E-difluoroethene
CC(=O)O	acetic acid	F/C=C\F	Z-difluoroethene
C1CCCCC1	cyclohexane	N[C@@H](C)C(=O)O	L-alanine
c1ccccc1	benzene	N[C@H](C)C(=O)O	D-alanine

Рис. 2. Примеры образования наименования молекул согласно номенклатуре SMILES

2. Open Source Chemistry Analysis Routines (OSCAR)

Программное обеспечение Open-Source Chemistry Analysis Routines (OSCAR), инструментарий для распознавания именованных сущностей и данных в химических публикациях, разрабатывается с 2002 года. Недавнее обновление привело к разделению основной функциональности OSCAR и ее выпуску в виде библиотеки OSCAR4. Данная библиотека имеет модульный API, что позволяет разработчикам включать ее во внешние приложения. OSCAR4 реализует независимую от домена архитектуру, на основе которой могут быть построены новые инструменты для химического анализа текста. [5]

OSCAR4 можно вызывать из внешнего программного обеспечения, как показано в примерах на Рисунке 3.

```
String text = "The quick brown ethyl acetate jumps over the lazy bromine";
Oscar oscar = new Oscar();
List <NamedEntity> nelist = oscar.findNamedEntities(text);
```

Рис. 3. Java-код с использованием API OSCAR4 для идентификации химических именованных сущностей (CNEs) в блоке текста

Приведенный выше пример показывает, как можно использовать OSCAR4 без необходимости какого-либо понимания базовой технологии или реализации.

3. ChemDataExtractor

Набор инструментов ChemDataExtractor(CDE) представляет собой усовершенствованный конвейер обработки естественного языка для извлечения информации о химических свойствах из научной литературы. [4]

Общий процесс извлечения информации из научного текста выглядит следующим образом:

- Разбить документ на составные элементы (заголовков, параграфы, предложения, таблицы, рисунки);
- Токенизация текста для выделения отдельных лексем;
- Применить тегирование части речи для определения семантической роли каждой лексемы;
- Обнаружение химических именованных сущностей с помощью машинного обучения;
- Разбор текста и таблиц с помощью вложенных правил для выявления химических связей;

– Устранение взаимозависимостей между различными элементами;

– Вывод набора взаимно согласованных химических записей.

ChemDataExtractor, аналогично библиотеке OSCAR4, позволяет извлекать химическую информацию в полностью независимо от домена.

Хотя данный инструмент поддерживает анализ документов разнообразных форматах, авторы просят по возможности избегать использования PDF-версии статьи.

4. Определение отношений

Синтаксические анализаторы строятся на помеченных лексемах для присвоения синтаксической структуры тексту. Цель разбора фразы будет заключаться в построении химического эквивалента древовидной структуры предложения Хомского. [3]

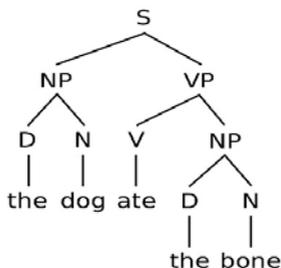


Рис. 4. Основное английское синтаксическое дерево.

В этой древовидной модели:

S — предложение, D — определитель, N — существительное, V — глагол, NP — именная словосочетание, VP — глагольное словосочетание.

Формальная структура химической области имеет высокую вероятность того, что будет найден только один анализ. Поэтому для парсинга фраз выбран формальный подход, с использованием ANTLR.

ANTLR (ANOther Tool for Language Recognition) — это генератор синтаксических анализаторов, использующий синтаксический анализ LL для автоматизации построения распознавателей языка. Он был разработан для создания грамматик для формальных языков программирования, но применим к любой области, где существует базовая неявная грамматика.

5. Программные результаты обнаружения наименований молекул

Аннотирование наборов данных о химических реакциях требует больших трудозатрат и специальных знаний, поэтому необходимо

уменьшить зависимость от большого количества размеченных данных, обычно необходимых для контролируемого обучения глубоких нейронных моделей. Для этого было выбрана стратегия дополнительного обучения ChemBERT v3.0 - предварительно обученный текстовый кодер на основе Transformer (Bidirectional Encoder Representations from Transformers) на молекулярных данных для обнаружения объединенных молекулярных представлений.

BERT реализует механизм самоконтроля, который изучает контекстуальные отношения между словами (или подсловами) в тексте. В общем представлении соединяет в себе два отдельных компонента - кодировщик, который считывает входной текст, и декодировщик, который производит предсказание для задачи. Поскольку целью BERT является создание языковой модели, необходим только механизм кодировщика.

Затем этот кодировщик ChemBERT был соединен с декодерами для конкретных задач (детектирование и аннотирование химических сущностей) и настроен с использованием ограниченного количества информации для каждой конечной задачи. Кроме того, входные тексты для маркировки ролей в рамках реакции должны были быть релевантными - описывать по крайней мере одну химическую реакцию и ее основной продукт, таким образом, образуя гораздо более ограниченное подпространство текстов общей химической литературы. ChemBERT использует моделирование языка с маской (masked-language modelling - MLM) и многозадачную регрессию (multi-task regression - MTR) на большом корпусе из 77 миллионов строк SMILES и исследует гипотезу масштабирования о том, что предварительное обучение на больших наборах данных может повысить производительность.

Для достижения поставленной задачи было проведено дополнительное обучение модели на пользовательских данных. Обучающие данные должны были 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
378	[PyH]+[2'cu4	B-Prod
379	@ I-Prod	
380	-C1)4 I-Prod	
381	@ I-Prod	
382	~ I-Prod	
383	-CL)- I-Prod	
384	(q	
385	4 B-Prod	
386) 0	
387	and 0	
388	[PyH]+[2.Ag4	B-Prod
389	@ I-Prod	
390	-C1)4 I-Prod	
391	@ I-Prod	
392	4- I-Prod	
393	CL])- I-Prod	
394	(0	
395	5 B-Prod	
396). 0	

Рис. 5. Разметка в стиле CoNLL и маркировка ролей в реакции.

Для тестирования полученных результатов был использован небольшой кусок текста.

```

"tokens": [
  "it",
  "was",
  "interesting",
  "to",
  "note",
  "that",
  "3a",
  "hydroxysaxatol[3,2-
  a]isoindol-8(3aH)-one",
  "derivative",
  "was",
  "not",
  "found",
  "and",
  "the",
  "methylmisoindolin-1-one",
  "4a",
  "was",
  "formed",
  "in",
  "good",
  "yield",
  "beside",
  "a",
  "little",
  "amount",
  "of",
  "(2)-3a",
  "methylmisoindolin-1-
  one",
  "5a",
  "was",
  "the",
  "loading",
  "of",
  "the",
  "catalyst",
  "Bu3P",
  "was",
  "increased",
  "to",
  "20",
  "or",
  "30",
  "mol",
  "of",
  "methylmisoindolin-1-
  one",
  "2",
  "and",
  "3",
  "of",
  "the",
  "Folin",
  "1",
  "1",
  ],
"reactions": [
  {
    "Reactants": [
      {
        "id": "3a",
        "text": "3a",
        "x": 5,
        "y": 9,
        "z": 6
      }
    ],
    "Product": [
      {
        "id": "(K)-3",
        "text": "methylmisoindolin-1-one",
        "x": 15,
        "y": 15,
        "z": 15
      }
    ],
    "Catalyst_Reagents": [
      {
        "id": "Bu3P",
        "text": "Bu3P",
        "x": 36,
        "y": 36,
        "z": 36
      }
    ]
  },
  {
    "Reactants": [
      {
        "id": "3a",
        "text": "3a",
        "x": 5,
        "y": 9,
        "z": 6
      }
    ],
    "Product": [
      {
        "id": "(E)-3",
        "text": "methylmisoindolin-1-one",
        "x": 27,
        "y": 27,
        "z": 27
      }
    ],
    "Catalyst_Reagents": [
      {
        "id": "Bu3P",
        "text": "Bu3P",
        "x": 36,
        "y": 36,
        "z": 36
      }
    ]
  },
  {
    "Reactants": [
      {
        "id": "3a",
        "text": "hydroxysaxatol[3,2-
        a]isoindol-8(3aH)-one",
        "x": 5,
        "y": 9,
        "z": 6
      }
    ],
    "Product": [
      {
        "id": "4a",
        "text": "4a",
        "x": 18,
        "y": 18,
        "z": 18
      }
    ],
    "Catalyst_Reagents": [
      {
        "id": "Bu3P",
        "text": "Bu3P",
        "x": 36,
        "y": 36,
        "z": 36
      }
    ]
  }
]

```

Рис. 6. Результат работы дообученной модели на тексте, используемого для экспертной оценки.

It was interesting to note that 3a- [hydroxyisoxazolo\[3,2-a\]isoindol-8](#) (3aH)-one derivative 3 was not found and the (E)-3-methyleneisoindolin-1-one 4a was formed in good yield beside a little amount of (Z)-3-methyleneisoindolin-1-one 5a when the loading of the catalyst Bu3P was increased to 20 or 50 mol % (entries 2 and 3 , Table 1). To our delight , an [excellent](#) yield of 5a was obtained when the reaction was carried out at 150 °C (entry 8 , Table 1).

Рис. 7. Текст, используемый для проведения экспертной оценки.

Оценка производительности моделей по маркированию ролей реакции проводилась на отдельных тестовых наборах с использованием стандартных метрик: Precision, Recall и F1- score (F1).

$$Pr\ precision = \frac{TruePositives}{(TruePositives + FalsePositives)}, \quad (1)$$

$$Re\ call = \frac{TruePositives}{(TruePositives + FalseNagatives)}, \quad (2)$$

$$F_1 = 2 \cdot \frac{Pr\ precision \cdot Re\ call}{(Pr\ precision + Re\ call)}, \quad (3)$$

Были получены следующие результаты:

Таблица

Результаты оценки моделей с использованием стандартных меток.

	Pr <i>recision</i> (%)	Re <i>call</i> (%)	F ₁ (%)
BERT	69.2	69.2	69.2
ChemBERT	77.0	76.4	76.7
OUR_ChemBERT_add	79.3	78.1	78.7

На уровне предложений ChemBERT достиг значительного превосходства над BERT, а адаптивный к задачам Our_ChemBERT_add дает дополнительные 2% улучшения в метрике F1- score (F1).

Заключение

Данная статья посвящена анализу готовых средств и разработке методик по выделению именованных сущностей типа «молекула–реактив».

Были рассмотрены готовые решения - OSCAR4, что реализует независимую от домена архитектуру, на основе которой могут быть

построены новые инструменты для анализа, а также ChemDataExtractor, что в свою очередь реализует усовершенствованный конвейер обработки естественного языка статей из области химии. Каждое из существующих решений обладает недочетами, например, авторы ChemDataExtractor просят по возможности избегать использования PDF-файлов в качестве источника данных.

Для решения задачи детектирования молекулярных наименований в форме номенклатуры ИЮПАК и маркирования ролей сущностей в рамках рассматриваемой реакции была использована предобученная модель ChemBERT v3.0. Данная модель была выбрана за универсальный формат разметки текстового корпуса в стиле CoNLL. В рамках экспертной оценки полученных результатов разбиения на токены, а также маркирования ролей лексических единиц наблюдается дробление на составные части наименований молекул, однако данные элементы все еще опознаны входящими в состав реакций.

Список литературы

1. The CHEMDNER corpus of chemicals and drugs and its annotation principles / Martin Krallinger [et.al.] // Journal of Cheminformatics. – 2015. – № 1. – P. 1-17.
2. Learning continuous and data-driven molecular descriptors by translating equivalent chemical representations. / Robin Winter [et.al.] // Chemical Science. — 2019. – № 10. – P. 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. OSCAR: an extensive repository of chemically and functionally diverse organocatalysts. / Simone Gallarati [et.al.] // Chemical Science. — 2022. – № 13. – P. 13782-13794.

Прогнозирование визуального контекста изображения за его пределами

А. С. Цыбульская

Студент магистратуры

В. С. Тарасов

Старший преподаватель

Введение

В настоящее время генерация изображений или их фрагментов с помощью нейронных сетей получила достаточно широкое распространение. Создание новых изображений и визуальных моделей на основе вводных параметров, устранение шумов и повышение четкости используется не только в сфере искусства и дизайна, но и в сферах медицины и науки. Частными случаями задачи генерации являются задачи интерполяции и экстраполяции визуального контекста.

Интерполяция изображения представляет собой восстановление утраченного фрагмента внутри изображения с использованием нейронных сетей. Такое решение используется в программах-редакторах, например, Photoshop, а также в компьютерной графике, для изменения размера изображения, повышения его разрешения или изменения формата. В медицине интерполяция может использоваться для увеличения разрешения медицинских изображений, таких как рентгеновские снимки или МРТ-сканы.

Экстраполяция, или же прогнозирование визуального контекста изображения за его пределами, пока не имеет такого широкого распространения. Но, во-первых, эта задача имеет схожую реализацию с задачей интерполяции, то есть может быть решена путем изменения уже существующего алгоритма машинного обучения. Во-вторых, она может быть применена для преобразования вертикального видео в горизонтальное положение, изменение размера изображения без потери качества, генерации текстур и моделей. Таким образом, можно сделать выводы об актуальности данной задачи.

1. Особенности реализации задачи экстраполяции

Задачи интерполяции и экстраполяции являются похожими как по смыслу, так и по реализации, но все же имеют различия. Используя

разницу в назначении алгоритмов можно применить готовую идею реализации интерполяции с соответствующими изменениями к задаче экстраполяции.

Во-первых, в задаче предсказания визуального контекста за пределами изображения неизвестная область может иметь неограниченную площадь, и с увеличением ширины границ качество сгенерированного изображения не должно резко снижаться. Также, если речь идет о неизвестной области по краям изображения (такой как на рисунке 1), вокруг нее будет меньше количество пикселей, несущих полезную информацию, в сравнении с неизвестной областью внутри изображения (рисунок 2). Визуальный контекст, из которого может быть сгенерирована часть изображения, в случае экстраполяции представляет собой только саму границу по краям исходного изображения.



Рис. 1. Неизвестная область при экстраполяции



Рис. 2. Неизвестная область при интерполяции

Второй особенностью задачи экстраполяции является заранее известное положение участков изображения, которые нужно сгенерировать.

2. Генеративно-сопоставительная сеть

Так как необходима генерация фотореалистичных участков изображений, было решено использовать генеративно-сопоставительную сеть (GAN). Этот алгоритм машинного обучения без учителя представляет собой комбинацию генератора и дискриминатора. Генератор генерирует изображения, они подаются на вход дискриминатору вместе с настоящими изображениями. Задача дискриминатора – отличить настоящие от сгенерированных. Такое сопоставительное обучение в общем случае показывает хорошие результаты, неразличимые человеческим глазом.

Для упрощения и стабилизации обучения, было принято решение использовать трехэтапную процедуру обучения, основанную на применении трех функций потерь.

$$L_{mse}(I_n, I_p) = \|M \otimes (G(I_p) - I_n)\|_2^2 \quad (1)$$

$$L_D(I_n, I_p) = -[\log D(I_n) + \log(1 - D(G(I_p)))] \quad (2)$$

$$L_G(I_n, I_p) = L_{mse}(I_n, I_p) - \alpha \log D(G(I_p)) \quad (3)$$

В первой фазе обучения веса генератора регулируются в соответствии с функцией потерь (1). Следующая фаза делает то же самое с дискриминатором в соответствии с функцией потерь (2). Третья фаза является фазой совместного сопоставительного обучения в соответствии с (2) и (3). В последней формуле α - это настраиваемый гиперпараметр, который компенсирует потери MSE потерями GAN стандартного генератора [1].

Для генератора будет использована структура кодер-декодер, в качестве дискриминатора – уравненная компиляция локального и глобального дискриминаторов.

3. Глобальный и локальный дискриминаторы

Принципы работы локальных и глобальных дискриминаторов схожи, за исключением подхода к анализу изображения.

Глобальный дискриминатор рассматривает все изображение целиком. На рисунке 3 показан результат его работы при выполнении задачи интерполяции.



Рис. 3. Результат работы глобального дискриминатора.

Заметно, что генератор смог «обмануть» дискриминатор, заполняя область с утраченным фрагментом расплывчатыми очертаниями, достаточно грубо скомпилированными из граничащих с фрагментом областей.

Локальный дискриминатор анализирует только область в непосредственной близости от вырезанного фрагмента.



Рис. 4. Результат работы локального дискриминатора.

Основной минус работы локального дискриминатора – на целостном изображении сгенерированный фрагмент хорошо заметен.

При поочередном применении обоих дискриминаторов в случае интерполяции удалось достичь хорошего результата (рисунок 5) [2].



Рис. 5. Результат применения обоих дискриминаторов

Однако при применении обоих дискриминаторов к задаче экстраполяции обнаружилось ошибки. Так, глобальный дискриминатор оставляет на изображении «пороги» перехода цвета. При применении локального вместе с глобальным обнаруживаются артефакты. Это происходит из-за упомянутой выше особенности экстраполяции – меньшего количества обрамляющих информативных пикселей.

Следовательно, при решении задачи предсказания визуального контекста за пределами изображения, подход локальных и глобальных дискриминаторов необходимо оставить. Однако, во избежание возникновения артефактов, в зависимости от ширины генерируемых участков, влияние локального дискриминатора необходимо ограничить экспериментально выводимым коэффициентом.

Из-за особенностей задачи экстраполяции не всегда хватает визуального контекста для генерации краевых участков. Эту проблему можно решить, используя подход циклической экстраполяции.

В случае если генерируемый участок достаточно широкий, предлагается генерировать на основе исходного изображения только его часть. Далее необходимо циклически передавать на вход изображение со сгенерированным участком и генерировать на его основе новую часть.

Заключение

Данная статья посвящена поиску подходов к разработке системы, генерирующей визуальный контекст изображения за его пределами. В

качестве основы для такой системы рассмотрены уже готовые решения для интерполяции изображений. В итоге анализа была определена архитектура и метод обучения нейронной сети. Также были выделены основные отличия интерполяции от экстраполяции, на основе этих различий были выбраны векторы изменения исходного алгоритма. Основными идеями для будущей разработки данной системы являются использование GAN, вычисление экспериментального коэффициента уравновешивания локального и глобального дискриминатора, циклическая экстраполяция.

Список литературы

1. Sabini, M. Painting Outside the Box: Image Outpainting with GANs / M. Sabini, G. Rusak // eprint arXiv. – 2018. – DOI: 10.48550/arXiv.1808.08483. – 2 p.
2. Iizuka S. Globally and Locally Consistent Image Completion / S. Iizuka, E. Simo-Serra, H. Ishikawa. // ACM Transactions on Graphics, volume 36, issue 4. – New York, 2017. – 9 p.

Повышение эффективности работы системы адаптивной модуляции в каналах с замираниями

А.В. Шипилов

Студент магистратуры

А.Ю. Савинков

Профессор

Ю.Н. Прибытков

Научный консультант

Введение

В наше время все большую и большую роль в жизни играют беспроводные сети, они получили свое широкое распространение благодаря активно увеличивающемуся количеству мобильных и портативных устройств.

Современные беспроводные системы передачи информации работают в постоянно изменяющихся условиях приема радиосигнала. С одной стороны, переменными являются мощность и степень искажения сигнала, вызванные многолучевым распространением или погодными условиями.

А с другой стороны - варьируется помеховая обстановка. современных системах сотовой связи и беспроводной передачи информации используется адаптивная регулировка мощности и адаптивная модуляция.

Адаптивная регулировка мощности является эффективным средством борьбы с замираниями сигналов, однако приводит к увеличению средней мощности, особенно значительному в условиях глубоких замираний. Повышение средней мощности является нежелательным явлением в системах с множеством преемопередатчиков, таких сотовые системы связи, поскольку приводит к повышению уровня взаимных помех.

Поэтому помимо адаптивной регулировки мощности используется адаптивная модуляция, когда темп передачи данных зависит от состояния канала и задается отношением мощности сигнала к суммарной мощности шума и помехи, а его изменение осуществляется за счет изменения битовой загрузки символов.

1. Системы связи с использованием адаптивной модуляции

В настоящее время предложено и исследовано большое число видов модуляции для цифровых беспроводных систем передачи информации. При разработке системы связи, выбор вида модуляции, как правило, является компромиссным решением между спектральной эффективностью, помехоустойчивостью, устойчивостью к нелинейным искажениям и межсимвольной интерференции. Тем не менее, пока не удастся найти единого универсального метода модуляции, позволяющего обеспечивать требуемые характеристики передачи данных в беспроводных системах связи.

Это объясняется тем, что свойства канала распространения сигнала могут существенно меняться даже в границах применимости конкретной системы передачи данных. Такая проблема особо остро стоит в системах связи с мобильными объектами, где приемник и передатчик могут находиться как в зоне прямой видимости, так и в условиях затенения и многолучевого распространения радиосигнала. В системах космической связи со стационарными объектами условия распространения сигнала также могут существенно варьироваться в зависимости, например, от погодных условий.

В такой ситуации возможно использовать метод модуляции, рассчитанный на наиболее сложные условия распространения. Однако в этом случае снижается спектральная эффективность системы связи в хороших условиях.

Поэтому в современных беспроводных системах передачи информации используются методы адаптивного выбора вида модуляции в зависимости от условия распространения сигнала.

2. Постановка задачи

Принцип работы систем связи с использованием адаптивной модуляции сводится к следующему: приемник сигнала измеряет уровень принятого сигнала и передает его по каналу обратной связи на передатчик. Передатчик же использует полученную оценку для выбора модуляции в следующем пакете.

Становится очевидным, что оценка запаздывает, так как тратится время на передачу по каналу обратной связи, обработку полученных результатов и интерпретацию для последующего выбора модуляции.

Предлагаемым решением, рассмотренным в данной статье, является предсказание уровня сигнала на основе авторегрессии для более точного выбора модуляции, то есть приемник, имеющий всю необходимую для расчета информацию, а именно, все измерения сигнала, предсказывает значения модуляции и передает приемнику по каналу обратной связи.

Авторегрессионная модель – модель временных рядов, в которой значения временного ряда в данный момент времени линейно зависят от предыдущих значений этого же ряда.

Авторегрессионный процесс порядка p определяется выражением (1).

$$X_t = c + \sum_{i=1}^p \alpha_i X_{t-i} + \varepsilon_t, \quad (1)$$

где $\alpha_1, \dots, \alpha_p$ - параметры модели (коэффициенты авторегрессии), c - постоянная (часто для упрощения предполагается равной 0), а $\{\varepsilon_t\}$ - белый шум.

В данной работе использовался метод авторегрессии – скользящего среднего (ARMA), представляющий собой одну из математических моделей, используемых для анализа и прогнозирования стационарных временных рядов.

Данная модель обобщает две более простые модели: модель авторегрессии и модель скользящего среднего.

Моделью авторегрессии – скользящего среднего называется процесс генерации временного ряда, определяемый выражением (2).

$$X_t = c + \varepsilon_t + \sum_{i=1}^p \alpha_i X_{t-i} + \sum_{i=1}^q \beta_i \varepsilon_{t-i}, \quad (2)$$

где c - константа, $\{\varepsilon_t\}$ - белый шум, то есть последовательность независимых и одинаково распределенных случайных величин, с нулевым средним, а $\alpha_1, \dots, \alpha_p$ и β_1, \dots, β_q - действительные числа, авторегрессионные коэффициенты и коэффициенты скользящего среднего, соответственно.

3. Реализация метода и имитационное моделирование

Для подтверждения эффективности метода на основе авторегрессии было проведено имитационное моделирование. Оцениваемой характеристикой являлось количество ошибок в выборе модуляции, то есть разница между предсказанной и оптимальной модуляцией.

На первом этапе были рассчитаны конечные разности на основе автокорреляционной функции $R(\tau)$, определяемой выражением (3) [1].

$$R(\tau) = J_0(2 \cdot \pi \cdot f_d \cdot \tau), \quad (3)$$

где $J_0(x)$ – функция Бесселя первого рода нулевого порядка.

На втором этапе по конечным разностям было получено авторегрессионное выражение с использованием метода линейного предсказания [2] и алгоритма Левинсона-Дурбина [3].

На третьем этапе полученное выражение было проинтегрировано.

Для проведения моделирования были заданы следующие параметры: отношение энергии сигнала, приходящейся на 1 бит принимаемого сообщения, к энергетической спектральной плотности шума (E_b/N_0), равное 18 дБ, относительная скорость передатчика и приемника равная 60 км/ч, время отправки пакетов равное 1 мс.

В ходе имитационного моделирования были получены следующие значения: количество ошибок для системы без предсказания равно 523548, количество ошибок для системы с использованием авторегрессионного метода равно 169590.

Таким образом, сравнив полученные значения, можно сделать вывод об эффективности предложенного метода, так как число ошибок с использованием авторегрессионного метода значительно ниже (более чем в 3 раза) количества ошибок для системы без предсказания.

Предложенный метод может быть использован в пакетных сетях для увеличения эффективности передачи информации в каналах с замираниями.

Заключение

Данная статья посвящена возможности повышения эффективности работы системы с использованием адаптивной модуляции в каналах с замираниями.

Предложен метод предсказания уровня сигнала на основе авторегрессии. В итоге данной работы была дана оценка эффективности предложенного метода, а также описаны возможности прикладного применения указанного алгоритма.

Предложенный способ расчета позволяет осуществлять обоснованный выбор между простыми методами с фиксированной модуляцией и более сложными методами адаптивной модуляции.

Полученные результаты могут быть использованы при проектировании и оценке эффективности систем беспроводной передачи информации через каналы с замираниями.

Список литературы

1. William C. Jakes, Donald C. Cox. Microwave Mobile Communications. Wiley-IEEE Press., 1994. – P. 642.
2. J. Makhoul. Linear prediction: A tutorial review. // Proceedings of the IEEE, 1975, vol. 63, no. 4, pp. 561-580.

3. Pedro A. Morettin. The Levinson Algorithm and Its Applications in Time Series Analysis. // International Statistical Review, 1984, Vol. 52, No. 1, pp. 83-92.
4. John G. Proakis. Digital Communications. McGraw-Hill, 2008. – P. 1150.

Содержание

Статьи студентов бакалавриата

Брюханова А. Ю., Толстобров А. П. Разработка сервиса, автоматизирующего учет посещаемости онлайн-занятий в портале LMS Moodle.....	4
Бурнашев Д. С., Залыгаева М. Е. Реализация рекомендательной системы пользовательских предпочтений на основе сингулярного разложения матриц на языке Python	11
Вакулин Д. А., Иванков А. Ю. Глубокое обучение в задаче стегоанализа цифровых изображений.....	19
Валеев В. И., Тарасов В. С. Анализ уязвимостей смарт-контрактов платформы Ethereum и их устранение на примере языка программирования Solidity	26
Володина А. А., Швырева А. В. Разработка системы мониторинга и встраивания ЦВЗ в измененные текстовые файлы	33
Воронцова М. И., Абрамов И. В. Повышение результативности работы отделов продаж заказчиков	41
Горбачева И. В., Алейникова Н. А. Разработка моделей оптимизации управления проектами	48
Долгова А. В., Алейникова Н. А. Разработка сервиса принятия решений при прогнозировании временных рядов цен акций.....	56
Донцов А. А., Самойлов Н. К. Разработка приложения для организации Mesh-сети посредством Nearby Connections API.....	65
Евсеева А. А., Митрофанова Е. Ю. Методы и алгоритмы машинного обучения для решения задачи распознавания языка жестов	72
Запорожец В. И., Акимов А. В. Сравнение оптимизаторов для обучения нейронных сетей	80
Кимбрис В. В., Клиньских А. Ф. Применение обучения с подкреплением для поиска оптимальных стратегий в играх.....	87
Копытина Е. А. Автоматизация формирования документов основных профессиональных образовательных программ высшего образования факультета компьютерных наук Воронежского государственного университета.....	94
Кравцов Д. Д., Вахтин А. А. Разработка сервиса определения местоположения устройства на основе GSM и Wi-Fi сетей.....	108

Матюхин А. Е., Махно А. А., Киселев Е. А. Приближенные формулы для построения узловых функций на основе систем целочисленных сдвигов функции Гаусса и базисных сплайнов высокого порядка	119
Мелихова В. А., Иванков А. Ю. Глубокое обучение в задаче распознавания онкологии кожи	126
Москаленко А. В., Громковский А. А. Моделирование процессов арбитража при торговле цифровыми активами	133
Мясоедов А. А., Самойлов Н. К. Кодогенерация многопоточных классов с помощью Annotation processor.....	143
Обухова Э. В., Швырева А. В. Использование цифровых следов в интересах решения задачи профориентации	150
Пилеич М. К., Чекмарёв А. И. Исследование быстродействия ORM библиотек	158
Плаксина Е. А., Воронцов Я. А. Формирование публичного расписания с помощью направленной синхронизации календарей	165
Рощупкин В. А., Воронцова Я. А. Практическое применение рекомендаций OWASP для тестирования REST API	173
Рыбас Д. В., Соломатин Д. И. Блокчейн-хранилище сертификатов моделей устройств интернета вещей	182
Савина Н. Ю., Самойлов Н. К. Генерация игрового поля для пошаговых настольных игр.....	188
Сашина А. Р., Гавшин А. Л. Использование Debezium в микросервисной архитектуре для реализации паттерна захвата изменения данных.....	195
Селиванова Е. А., Лысачев П. С. Разработка онлайн-портала для вышивок.....	199
Синяков С. М., Соломатин Д. И. Разработка модуля дополнительной отчетности для CRM-системы в виде расширения браузера	204
Сметанин И. Ю., Самойлов Н. К. Оптимизация SELECT запросов в СУБД PostgreSQL	209
Стешенко Э. А., Швырева А. В. Двухфакторная аутентификация пользователя с использованием системы распознавания лиц	215
Тишанский Д. А., Соломатин Д. И. SaaS-решение для геймификации сервисов	223
Токарев Р. А., Жидовленков Е. В., Туровский Я. А. Влияние эпохи анализа и децимации на значения коэффициентов корреляции электроэнцефалограммы	230

Толстых П. А., Михайлов Е. М. Разработка системы мониторинга веб-приложений	234
Третьякова Ю. В., Атанов А. В. Решение переопределённых систем нелинейных уравнений.....	239
Улезько Т. А., Шевцов Р. С., Кургалин С. Д., Батищева Г. А., Борзунов С. В. Разработка программы для корректировки доз лекарственных препаратов.....	246
Ушакова А. А., Гаршина В. В. Автоматическое задание классов сущностей на основе выделения из текстов синонимов, гипонимов, гиперонимов	254
Филонов П. О., Гавшин А. Л. Разработка web-сервиса для анализа ЭКГ на примере распознавания инфаркта миокарда.....	263
Филонова В. А., Гаршина В. В. Визуализация семантики текста на основе графа знаний	270
Хныкин Д. Е., Самойлов Н. К. Управление системой полива тепличного комплекса с учетом погодных условий.....	277
Шорохова Т. А., Копытина Е. А. Разработка проекта «List_of_documents» для реализации программ дополнительного образования ФГБОУ ВО «Воронежского государственного университета	284
Юревич Е. С., Залыгаева М. Е. Реализация одной задачи оптимизации на языке программирования Python.....	290

Статьи студентов магистратуры

Беспалов В. М., Савинков А. Ю. Защита программы в памяти микроконтроллера для предотвращения клонирования устройства	297
Булавина Е. П., Савинков А. Ю. Специализированный отладчик для программ цифровой обработки сигналов	302
Бурдюг П. С., Акимов А. В. Решение задачи описания изображений с помощью нейронных сетей.....	308
Василега В. В., Попело В. Д. Нейросетевой и МНК алгоритмы совместного определения ориентации и оптических параметров плоской диффузной поверхности по данным измерений характеристик светорассеяния.....	317
Верещагина М. С., Гаршина В. В. Поиск закономерностей в поведении клиентов телекоммуникационной компании	325
Господарикова В. С., Гаршина В. В. Обзор технологий голосового ввода для анализа сообщений	332
Григорьев В. О., Акимов А. В. Использование нейронных сетей в задаче распознавания сцен.....	338

Денисова Д. А., Гаршина В. В. Обзор методов обучения с подкреплением для решения бизнес-задач на основе имитационного моделирования	346
Дешин Н. В., Дрюченко М. А. Математическое и программное обеспечение для автоматизации идентификации типа тары в задаче переработки вторсырья	354
Донских Д. П., Воронцов Я. А. Разработка системы управления отоплением с использованием нечеткого логического	360
Елфимова Е. В., Соломатин Д. И. Адаптация веб-приложения «SQL портал» под мобильные устройства	368
Жехангер Ж. А., Иванков А. Ю. Анализ угроз и тенденций рынка информационных технологий	374
Жехангер Ж. А., Иванков А. Ю. Искусственный интеллект в кибербезопасности.....	382
Зверева А. С., Швырева А. В. Разработка программного обеспечения для анализа наполненности и целостности медицинских ампул	388
Зеленев Н. А., Сирота А. А. Разработка приложения для разметки данных на основе микросервисной архитектуры	393
Иванов К. Д., Фертиков В. В. Прослеживание границ объектов бинарных изображений	401
Капкин К. А., Акимов А. В. Методы оптимизации работы нейронных сетей	408
Квасов А. Н., Илларион И. В. Автоматизированное тестирование бизнес-процессов ChatOps Telegram	416
Килигин Е. А., Кольцова И. А., Киселев Е. А. Использование систем целочисленных сдвигов базисных сплайнов в методе Адамса	421
Клишин И. И., Десятирикова Е. Н. Разработка генератора REST API сервисов на основе модели данных.....	428
Крупенин С. С., Савинков А. Ю. Поиск сигнала спутниковой навигационной системы Galileo	436
Кубленко П. В., Соломатин Д. И. Разработка платформы для построения чат-ботов	440
Макаренко Е. С., Фертиков В. В. Разработка алгоритма распознавания элементов нотной грамоты с помощью кластерного анализа	447
Масленникова С. С., Коротков В. В. Автоматическая оценка дефектов на основе статистических данных однотипных проектов	451
Муравьёва Е. В., Гаршина В. В. Автоматическое построение таксономии текста.....	457

Назаренко В. С., Малыхин А. Ю. Разработка сайта для моделирования искажения акустического сигнала в мелком море	463
Олейник Е. И., Малыхин А. Ю. Моделирование в пакете MATLAB передачи акустических сигналов в океаническом волноводе.....	469
Пауков Н. В., Соломатин Д. И. Разработка инструментария для исследования и отладки сетей Zigbee	478
Петров Ю. В., Коваль А. С. Исследования трафика корпоративной сети с применением машинного обучения (в задачах оптимизации использования ресурсов.....)	485
Саввин М. О., Лобода А. В. Анализ плоских движений фигур сложной формы.....	492
Селиверстов Н. А., Иванков А. Ю. Использование машинного обучения для решения задачи разделения аудиосигнала	499
Семечев Д. А., Борисов Д. Н. Интеграция технологий для реализации электродинамического моделирования	508
Скородед Е. П., Атанов А. В. Влияние расположения узлов расчётной области на решения уравнений в частных производных методом коллокаций	514
Сурков А. А., Киселев Е. А. Использование метода максимумов модуля для оценки характерных значений параметров электрокардиографических сигналов	521
Турченко К. А., Борзунов С. В. Использование алгоритмов машинного обучения для планирования рабочей нагрузки высокопроизводительных вычислительных систем	526
Филипенко Р. П., Десятерикова Е. Н. Разработка мобильного графического редактора с функцией распознавания контуров на платформе ios	531
Хоменко П. А., Гаршина В. В. Извлечение данных о технологии проведения химических реакций из неструктурированных PDF-файлов.....	535
Цыбульская А. С., Тарасов В. С. Прогнозирование визуального контекста изображения за его пределами	543
Шипилов А. В., Савинков А. Ю., Прибытков Ю. Н. Повышение эффективности работы системы адаптивной модуляции в каналах с замираниями	549

Научное издание

**ТРУДЫ МОЛОДЫХ УЧЁНЫХ
ФАКУЛЬТЕТА КОМПЬЮТЕРНЫХ НАУК ВГУ**

В ы п у с к 3

Под редакцией *Д. Н. Борисова*

Издано в авторской редакции