МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

«ТЮМЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ НАУК

Кафедра программного обеспечения

|  |  |
| --- | --- |
|  | РЕКОМЕНДОВАНО К ЗАЩИТЕ В ГЭК  Заведующий кафедрой  К. т. н., доцент  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М. С. Воробьева  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г. |

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

бакалаврская работа

РАЗРАБОТКА СЕРВИСА ДЛЯ АНАЛИЗА ТЕКСТОВ НАУЧНЫХ ПУБЛИКАЦИЙ СБОРНИКА «МАТЕМАТИЧЕСКОЕ И ИНФОРМАЦИОННОЕ МОДЕЛИРОВАНИЕ»

02.03.03 Математическое обеспечение и администрирование информационных систем

Профиль «Технологии программирования и анализа больших данных»

|  |  |  |
| --- | --- | --- |
| Выполнили работу  (групповой проект)  студенты 4 курса  очной формы обучения |  | Загайнова Евгения Олеговна  Калимова Алтынай Есенбаевна |
| Руководитель  Профессор, д. пед. н. |  | Захарова Ирина Гелиевна |

# **ВВЕДЕНИЕ**

В наши дни важной и неотъемлемой частью успешной научной работы является оформление публикуемых статей. Грамотная подготовка статьи оказывает непосредственное влияние на шансы её успешной публикации в требуемом сборнике и способствует улучшению представления информации, тем самым повышая ценность и доступность для широкой аудитории.

Однако взаимодействие между авторами и редакторами научных журналов часто бывает затруднено. Возникают сложности с выбором подходящей структуры статьи и соблюдением специфических требований конкретного издания. Ошибки в оформлении, неполные аннотации, неправильно составленный список литературы и отсутствие или избыток ключевых слов могут снизить шансы на принятие статьи. Эти проблемы особенно заметны в технических и научных областях, где стандарты подачи данных между изданиями могут сильно различаться. Из-за несовпадения между требованиями издателя и тем, как их в результате понимают авторы, статьи могут отклоняться без дальнейшего рассмотрения. Чтобы этого избежать, необходимо тщательное изучение предыдущих выпусков и детальный анализ опубликованных материалов. При этом проведение полного обзора всех номеров издания — достаточно трудоемкий процесс, и авторам зачастую сложно самостоятельно оценить степень соответствия своей работы тематическому фокусу и формату издания.

В связи с этим целью данной работы является разработка веб-сервиса, который поможет исследователям искать и изучать научные статьи. В нем пользователю будет предоставляться аналитическая информация о содержании, возможность ориентироваться в рамках научного издания и эффективно искать необходимую ему информацию. Так автор сможет определить, подходит ли его работа для выбранного сборника, и может ли он адаптировать её в соответствии с требованиями, или будет вынужден отказаться от публикации, избегая изучения большого количества научных статей.

Для достижения цели необходимо выполнить следующие задачи:

1. Изучить методы анализа научных статей;
2. Спроектировать систему хранения данных;
3. Разработать модули для обработки и анализа научных текстов:

3.1. Разработать модуль предобработки данных;

3.2. Разработать модуль извлечения метаданных;

3.3. Разработать модуль визуализации данных;

1. Спроектировать архитектуру приложения;
2. Разработать серверную и клиентскую части приложения;
3. Провести тестирование функционала приложения.

**ГЛАВА 1. анализ ТЕКСТОВ НАУЧНЫХ ПУБЛИКАЦИЙ СБОРНИКА «МАТЕМАТИЧЕСКОЕ И ИНФОРМАЦИОННОЕ МОДЕЛИРОВАНИЕ:**

1.1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

Научные тексты сборника «МАТЕМАТИЧЕСКОЕ И ИНФОРМАЦИОННОЕ МОДЕЛИРОВАНИЕ» представляют собой структурированные документы. Предварительный анализ текстов публикаций данного сборника позволил выявить общие характеристики, которые могут быть присущи к другим научным изданиям. Эти закономерности позволяют применять универсальные подходы к анализу научных текстов и разработке инструментов для такого анализа.

Анализ научных публикаций требует учёта нескольких ключевых аспектов. В первую очередь, научные статьи обычно состоят из типовых разделов: введение, постановка задачи, описание методологии, результаты и заключение, содержат визуальные элементы. Их можно отслеживать и выделять, не только обнаруживая соответствующий заголовок, но и обращая внимание на контекст и стиль написания той или иной части статьи.

Далее, важно заметить, что метаданные, включающие аннотацию, ключевые слова, данные об авторах и список литературы, в рамках цифровых научных изданий имеют большое значение, обеспечивая навигацию в массиве научных данных. Они позволяют ориентироваться в нем без необходимости глубокого ознакомления с полными текстами статей. Благодаря метаданным пользователи могут осуществлять целенаправленный поиск, фильтровать публикации по различным критериям и находить необходимые материалы.

Анализ научных статей формируется из двух пунктов: разделов и метаданных, что позволяет проводить детальное рассмотрение и каждой отдельной статьи отдельно, и сборника, где она опубликована, в целом.

## 1.2 МЕТОДЫ ОБРАБОТКИ ТЕКСТА

Обработка — важный подготовительный этап, его цель заключается в устранении избыточной и нерелевантной информации и обеспечении пригодности данных для различных аналитических задач. Далее будут рассмотрены методы, применяемые в задаче обработки текста на естественном языке[11].

Удаление знаков препинания помогает избавиться от элементов, не несущих смысловой нагрузки. Преобразование текста в нижний регистр устраняет различия между заглавными и строчными буквами, что снижает размер словаря. Удаление лишних пробелов, табуляций и знаков переноса улучшает структуру текста, делая его более удобным для дальнейшей обработки. Также некоторые задачи подразумевают обязательное удаление стоп-слов — часто встречающихся, но не несущих значимой информации, такие как предлоги и союзы[5].

Токенизация — процесс разбиения текста на минимальные смысловые единицы, токены. Это шаг в обработке текстовых данных, который значительно упрощает дальнейшую обработку, например, при подсчёте частоты слов, лемматизации или классификации текста. Он позволяет выделять слова, которые составляют строку символов[12].

Базовые методы реализуются с использованием функции Python split(). Данная функция возвращает список строк после разбиения заданной строки указанным разделителем. По умолчанию split() разбивает строку по пробелам.

Библиотека NLTK (Natural Language ToolKit) предоставляет несколько способов токенизации: она содержит модуль tokenize(), который далее классифицируется на две подкатегории: – Word tokenize, чтобы разбить предложение на токены или слова, и Sentence tokenize, чтобы разбить документ или абзац на предложения.

Лемматизация и стемминг — это методы, применяемые для приведения слов к их основам или начальным формам. Лемматизация преобразует слово с сохранением его контекста. Стемминг, в отличие от лемматизации, просто сокращает слово до его основы, игнорируя грамматические особенности. Формы слов, однокоренные слова создают “ложную” вариативность, хотя часто принадлежат одним лексическим единицам и имеют схожий смысл. Таким образом, приведение к леммам и стамам позволяет избавиться от избыточности данных и их двусмысленности. Могут быть реализованы средствами библиотеки NLTK.

Векторизация — процесс преобразования текста в числовые векторы, которые могут быть использованы для дальнейшего анализа, в алгоритмах, которые работают с числовыми данными.

Методы векторизации могут быть частотными как Bag of Words, который представляет текст как вектор, где каждый элемент — частота слов в документе, TF-IDF — более сложный, учитывает не только частоту слова в документе, но и его значимость в коллекции документов. Word2Vec использует нейронные сети для преобразования слов в многомерные вектора, учитывающие контекст.

Векторизация с помощью трансформеров подразумевает использование архитектуры трансформеров (BERT, RoBERTa) для создания эмбеддингов (векторных представлений) текста. В отличие от вышеупомянутых методов, трансформеры создают более сложные и контекстуально осмысленные представления текста, которые учитывают порядок слов, связи и контекст в предложении

## 1.3 МЕТОДЫ АНАЛИЗА ТЕКСТОВ

## Задача анализа текста состоит в извлечении значимой информации из структурированного предварительной предобработкой текста.

Частотный анализ — метод подсчёта количества упоминаний слов или фраз в тексте, позволяет определить, какие термины текста являются ключевыми. Применяется для выявления тем текста, определения значимости терминов и создания тематических словарей. Так же может быть использован в качестве подготовительного этапа для других методов: тематического моделирования, построения облаков слов.

Регулярные выражения представляют собой инструмент для поиска и выделения текста по заданным шаблонам, используются для сегментации текста, выделения ключевых данных или фильтрации избыточной информации.

Методы выше относят к поверхностным методам анализа текста, основанным на извлечении отдельных частей текста и статистических метриках. Глубокие методы анализа подразумевают болеесложные методы обработки, которые учитывают не только поверхностные характеристики, которые могут быть получены и без предварительной обработки.

Например, тематическое моделирование, как задача глубокого анализа, заключается в автоматическом выявлении скрытых тем в большом наборе текстов. Например, частотный алгоритм LDA (Latent Dirichlet Allocation) определяет темы как группы слов, которые часто встречаются вместе и потенциально имеют общую смысловую нагрузку. Другие подходы, например Word2Vec, позволяют преобразовать слова в векторное представление и оценить их семантическую близость. Для тематической группировки так же может быть использован кластерный анализ. Работая с наукообразными текстами, можно проводить его для разных частей документов коллекции, например, кластеризовать источники литературы — распространенная практика для поиска похожих публикаций[6].

# **ГЛАВА 2. проектирование и разработка КЛИЕНТ-СЕРВЕРНОГО ПРИЛОЖЕНИЯ**

2.1 АРХИТЕКТУРА ПРИЛОЖЕНИЯ

Архитектура Django [7] веб-приложения основана на шаблоне MTV (Model-Template-View), который разделяет приложение на три слоя: Model (Модель) описывает структуру базы данных, отношения между данными и предоставляет доступ к данным через Django ORM, позволяя выполнять операции CRUD с объектами Python вместо SQL-запросов. Однако это не исключает возможности применения нативных запросов в приложении средствами библиотеки psycopg-binary.

Модель Section представляет секции сборника, каждая из которых имеет идентификатор (section\_id) и название (name). Модель Author описывает авторов с их идентификаторами (authors\_id) и именами (name). Модель Article представляет статьи, связывая их с секциями через внешний ключ (section) и с авторами через связь "многие-ко-многим" (authors), реализованную через промежуточную модель ArticleAuthors. В статье указаны заголовок (title), метаданные, количественные характеристики, такие как количество ключевых слов, ссылок, изображений и таблиц. Поле annotation\_embeddings, добавленное в базу данных с помощью миграций, хранит массивы эмбеддингов аннотаций для организации поиска среди статей.

Template отвечает за отображение данных с помощью html-шаблонов. View обрабатывает запросы пользователя, взаимодействует с моделями для получения или изменения данных и передает их в шаблоны для формирования ответа. Views нашего приложения отвечают за обработку запросов, фильтрацию данных, визуализацию и управление списком статей. Основное представление — класс ArticleListView реализует отображение списка статей с пагинацией, фильтрацией и поиском. Оно добавляет в контекст шаблона список секций, авторов и диапазоны числовых фильтров, рассчитываемые с помощью агрегатных функций. Переопределенный метод get\_queryset() фильтрует статьи по выбранным секциям, авторам, диапазонам числовых значений и, при необходимости, то есть получении поискового запроса, выполняет поиск по аннотациям статей с использованием предобученной языковой модели. Все данные передаются в соответствующий шаблон, который поддерживает интерактивную фильтрацию и навигацию по статьям.

Компоненты связываются через систему маршрутизации, которая перенаправляет запросы на представления. Поток выполнения запроса начинается с маршрутизации, переходит к обработке представлением, взаимодействию с моделью и завершается генерацией html-ответа.

2.2 ИЗВЛЕЧЕНИЕ И ОБРАБОТКА ДАННЫХ НАУЧНЫХ СТАТЕЙ

Исходные данные представлены в виде цельного PDF-файла, содержащего научные статьи. Выпуск сборника скачивается и далее обрабатывается для извлечения метаданных и текста каждой статьи. Извлеченные данные, такие как заголовки, авторы, аннотации, ключевые слова, количества изображений, таблиц и источников, сохраняются в базе данных scholar\_db PostgreSQL[8] для последующего использования. Основные этапы этого процесса включают:

1. Загрузка исходных данных — одного PDF-файла;
2. Разбиение на статьи — PDF-файл делится на отдельные статьи, которые обрабатываются независимо. Разбиение происходит согласно страницам, указанным в оглавлении, для правильного разбиения необходимо указать диапазон страниц оглавления;
3. Извлечение данных — из каждой статьи извлекаются такие данные, как заголовки, авторы, аннотации, ключевые слова и другие метаданные;
4. Запись в базу данных — все извлеченные данные записываются в базу данных для дальнейшего анализа и хранения.

Для работы с PDF-файлом используется библиотека PyMuPDF (fitz), которая позволяет извлекать и текст (Рис. 1), и изображения, и таблицы. Изображения и таблицы не извлекаются, в соответствующие поля таблицы записываются только их количества. Так же подсчитывается количество позиций в источнике, количество ключевых слов.

В классе Journal, отображающем сборник,читается оглавление, из которого извлекаются названия секций, номера которой фиксируются для каждой статьи, названия статей, их авторы и страницы, где статьи расположены.

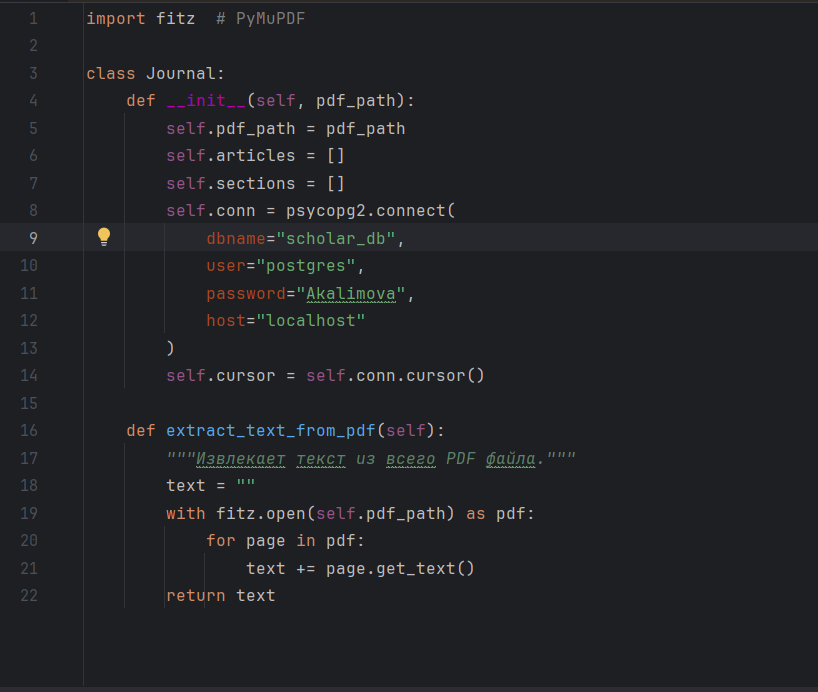


Рис. 1. Извлечение текста из сборника

Далее начинается процесс разбиения общего файла на PDF-файлы отдельный статей(Рис. 2). Создается объект класса Article для статьи, в конструкторе которого содержатся необходимые для извлечения данные. Для того чтобы разделить статью на компоненты(Рис.3), используются методы с регулярными выражениями. Они помогают обнаруживать искомые части по маркерам. Например, для извлечения названий используется регулярное выражение, которое ищет строки в верхнем регистре, для поиска авторов — распознающее имена в формате ФИО с инициалами после фамилии. Структура сборника “МАТЕМАТИЧЕСКОЕ И ИНФОРМАЦИОННОЕ МОДЕЛИРОВАНИЕ” позволяет использование только регулярных выражений, так как каждый раздел помечен соответствующим заголовком, однако это делает инструмент недостаточно универсальным.

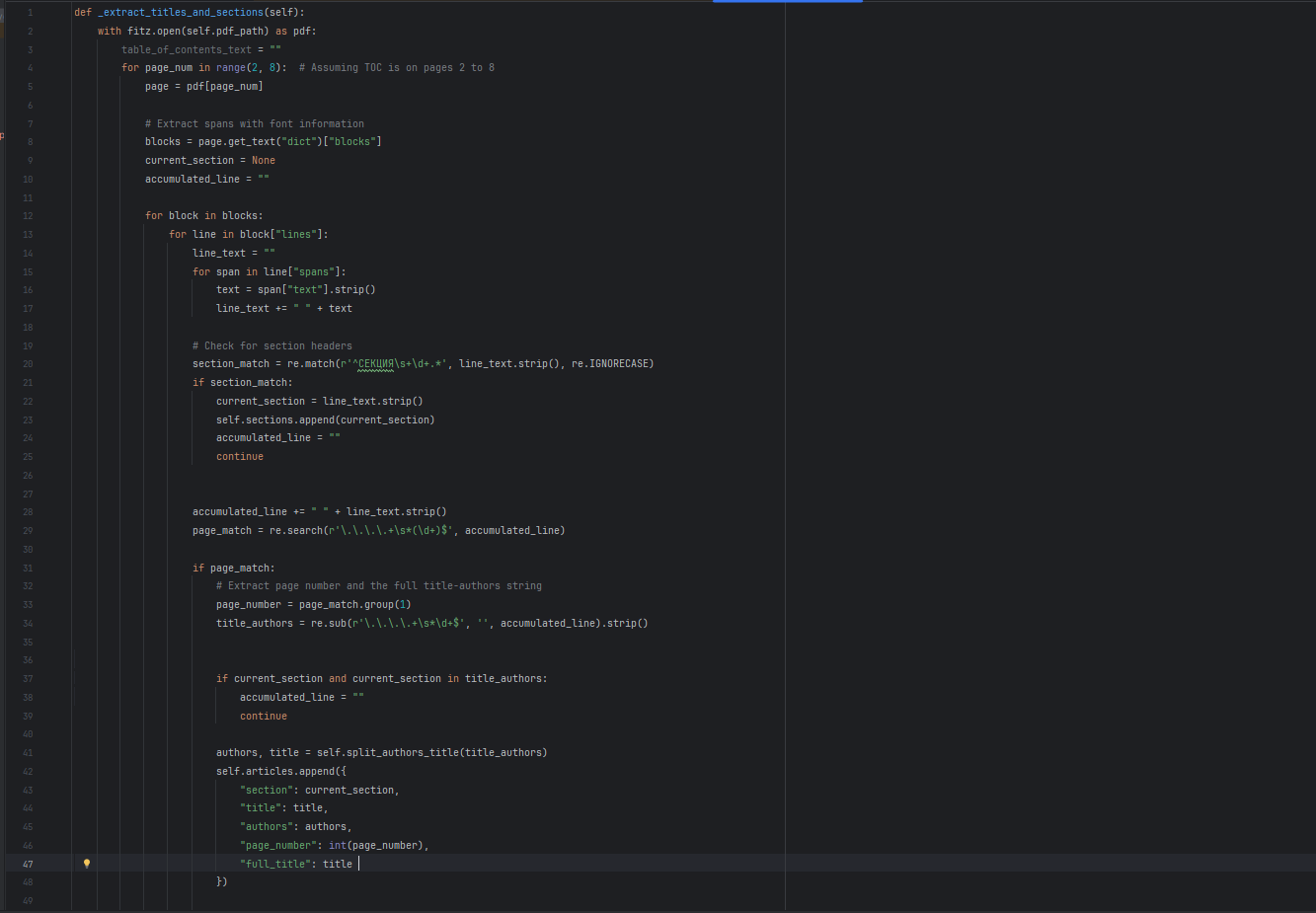


Рис. 2. Разбиение файла на статьи

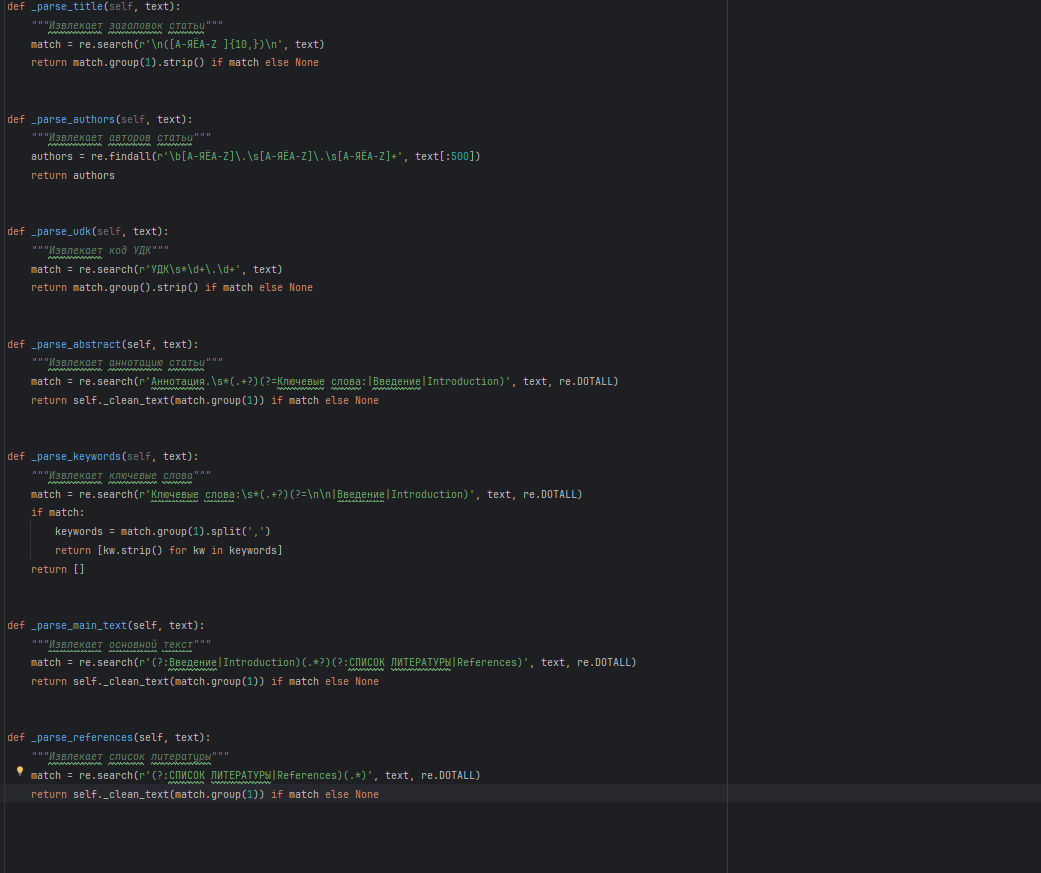


Рис. 3. Разделение статьи на компоненты

Следующим этапом является запись полученных характеристик в базу данных. Такой вариант хранения обеспечивает упрощенную и ускоренную работу таких инструментов, как поиск и фильтрация. Для соединения с выбранной нами СУБД PostgreSQL и управления транзакциями используется библиотека psycopg2. С ее помощью как обеспечивалась работа моделей ORM, так и использовались стандартные SQL-запросы. Взаимодействие между моделями ORM и базой данных организованы миграциями. Они позволяют автоматически создавать, изменять и удалять таблицы, индексы и другие объекты базы данных на основе изменений, внесенных в модели[8].

2.2 БАЗА ДАННЫХ

В этом разделе рассматривается структура базы данных (Рис. 4), включающая ключевые таблицы и связи между ними, а также описание каждой сущности.

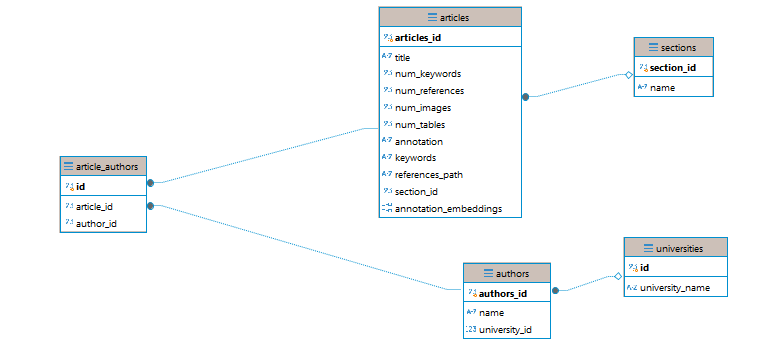


Рис. 4. Схема базы данных

Сущность Articles представляет собой таблицу, содержащую информацию о статьях из научного сборника. Каждая статья имеет уникальный идентификатор `articles\_id`, а также включает в себя метаданные: заголовок, аннотацию, ключевые слова, количество изображений и таблиц. Поле `section\_id` связывает статьи с таблицей Sections, указывая, к какому разделу относится статья.

## «articles» — Сущность статей

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Описание |
| articles\_id | integer | Уникальный идентификатор статьи (PK). |
| title | text | Заголовок статьи. |
| num\_keywords | integer | Количество ключевых слов в статье. |
| num\_references | integer | Количество ссылок в статье. |
| num\_images | integer | Количество изображений в статье. |
| num\_tables | integer | Количество таблиц в статье. |
| annotation | text | Аннотация статьи. |
| keywords | text | Список ключевых слов статьи. |
| references\_path | text | Путь к файлу с библиографическими ссылками. |
| section\_id | integer | Идентификатор раздела, к которому относится статья (FK). |
| annotation\_embeddings | text | Представление аннотации в виде векторных данных (эмбеддингов). |

Сущность Authors содержит информацию об авторах статей. Каждому автору присваивается уникальный идентификатор `authors\_id`. Поле `name` хранит имя автора, а `university\_id` связывает автора с его университетом, который хранится в таблице Universities.

«authors» — Сущность авторов

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Описание |
| authors\_id | integer | Уникальный идентификатор автора (PK). |
| name | text | Имя автора статьи. |
| university\_id | integer | Идентификатор университета, к которому относится автор (FK). |

Отношение "многие ко многим" между сущностями Article и Authorsразделяется на два отношения "многие к одному" с помощью промежуточной сущности**, связующей таблицы** ArticleAuthors.

«article\_authors» — Связь статей и авторов

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Описание |
| id | integer | Уникальный идентификатор записи (PK). |
| article\_id | integer | Идентификатор статьи (FK). |
| author\_id | integer | Идентификатор автора статьи (FK). |

Сущность Sections представляет разделы научного сборника, в которые включаются статьи. Разделы используются для логической группировки статей по темам или областям исследования, что упрощает их структуризацию. Разделы связаны с таблицей Articles через поле `section\_id`, что позволяет легко находить все статьи, относящиеся к конкретному разделу.

«sections» — Сущность разделов

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Описание |
| section\_id | integer | Уникальный идентификатор раздела (PK). |
| name | text | Название раздела. |

Сущность Universities хранит данные об университетах, к которым относятся авторы статей. Каждый университет имеет уникальный идентификатор `id` и название, которое сохраняется в поле `university\_name`. Поле `university\_id` в таблице Authors используется для связи авторов с их университетами.

« universities » — Сущность разделов

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Описание |
| id | integer | Уникальный идентификатор университета (PK). |
| university\_name | text | Название университета. |

## 2.3 ФИЛЬТРАЦИЯ И ПОИСК

Фильтрация и поиск организованы с помощью метода get\_queryset в классе ArticleListView, позволяя пользователю выбирать статьи по секциям, авторам, числовым диапазонам. При обработке GET-запроса из URL извлекаются параметры: sections (список ID секций), author (ID автора), числовые диапазоны для метрик (например, min\_num\_keywords) и строка поиска search\_str Для фильтрации по секциям используется оператор \_\_in, чтобы выбрать статьи, относящиеся к указанным секциям. Фильтрация по авторам реализована через промежуточную таблицу ArticlesAuthors. Числовые диапазоны проверяются условиями \_\_gte (больше или равно) и \_\_lte (меньше или равно), применяемых к таким метрикам, как количество ключевых слов, изображений, таблиц и ссылок.

Семантический поиск построен на основе обработки аннотаций статей с использованием предобученной модели DeepPavlov/rubert-base-cased-sentence, которая преобразует тексты в многомерные вектора (эмбеддинги). Основной этап подготовки заключается в векторизации аннотации каждой статьи и записи результата — списка векторов в базу данных. Аннотации, так же хранящиеся в базе ввиду небольшого объема текста в этом разделе, разбиваются на отдельные предложения, которые затем преобразуются в эмбеддинги с помощью функции embed\_bert\_cls. Эта функция токенизирует текст с использованием AutoTokenizer, передает токены в модель AutoModel, извлекает эмбеддинг , представляющий текст, и нормализует его для дальнейшего использования в вычислении схожести.

Сгенерированные эмбеддинги сохраняются в поле модели Article. Во время выполнения запроса пользователь вводит строку поиска, которая также преобразуется в эмбеддинг. Затем для каждой статьи создается объект Candidate (Рис.7), содержащий заголовок, аннотацию и эмбеддинги. Он вычисляет семантическую схожесть между запросом и статьей, комбинируя два показателя: косинусное сходство эмбеддинга запроса с эмбеддингами предложений аннотации и схожесть заголовков, рассчитываемую с использованием индекса Жаккара (Рис.8).

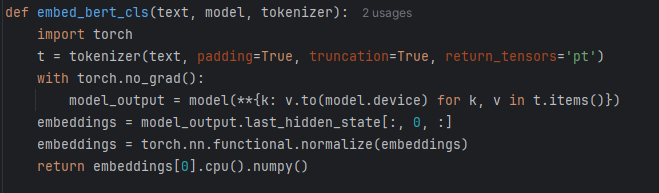
Итоговое значение отражает степень релевантности статьи запросу и используется для сортировки статей. Только статьи с показателем выше заданного порога (например, 0.75) возвращаются как результаты поиска. 

Рис. 6 Метод embed\_dert\_cls

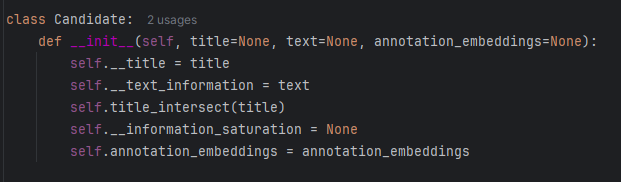


Рис. 7 Класс Candidate

## 2.4 Интерфейс приложения

Интерфейс приложения был спроектирован с целью обеспечения удобства пользователей при взаимодействии с научными статьями и метаданными. Он включает в себя несколько ключевых элементов, которые позволяют пользователям эффективно выполнять задачи поиска, фильтрации и отображения информации. Рассмотрим компоненты интерфейса более подробно.

Основной экран(Рис. 8) приложения состоит из двух разделов: панели поиска и фильтрации и списка публикаций. Панель поиска и фильтрации расположена с левой стороны интерфейса. Она включает различные параметры, которые позволяют пользователю точно настроить поиск. Ключевые параметры фильтрации включают:

* Секции: пользователь может выбрать из предложенного списка секций (например, "Деятельностное математическое образование", "Информационные системы анализа данных"). Это позволяет сузить круг поиска по теме публикации.
* Авторы: возможность выбрать определённых авторов. Функция автодополнения помогает быстро найти нужного автора.
* Количество ключевых слов, изображений, таблиц: фильтрация публикаций по этим критериям. Например, если пользователь ищет статьи с определённым количеством изображений или таблиц, он может задать соответствующие параметры.

Эти фильтры позволяют пользователю ограничить результаты поиска и найти публикации, которые соответствуют его интересам или исследовательским запросам.

Список публикаций отображает все найденные статьи, соответствующие выбранным фильтрам. Каждая публикация в списке включает:

* Название статьи, которое кликабельно и ведет на страницу с подробной информацией.
* Автора и секцию: указаны имена авторов и тема, к которой относится статья, что помогает быстро ориентироваться в материалах.
* Аннотацию: краткое описание содержания статьи.
* Ключевые слова: метки, которые дают представление о теме статьи.
* Количество изображений, таблиц и ссылок: эти поля показывают, сколько визуальных материалов и источников включает статья.

Каждая запись в списке является кликабельной, что позволяет пользователю перейти на страницу с полным описанием статьи(Рис. 9) и метаданными. Когда пользователь переходит по ссылке на статью, открывается страница с подробной информацией о публикации. На этой странице представлено:

* Название статьи — в верхней части страницы, с выделением и возможностью копировать или поделиться ссылкой.
* Секция — тема, к которой относится статья (например, "Математическое моделирование").
* Авторы — имена всех авторов с возможностью перейти к их профилям или работам.
* Ключевые слова — список ключевых терминов, которые связаны с темой статьи, и могут быть использованы для дальнейшего поиска.
* Аннотация — текст, кратко объясняющий содержание и основные моменты исследования.
* Изображения и таблицы — отображаются как количество этих материалов в статье.
* Ссылки — количество источников, на которые ссылается статья, с возможностью перехода по ним.

Этот экран предоставляет пользователю полное представление о статье и её метаданных, что позволяет быстро оценить её релевантность.

Приложение также включает элементы визуализации данных, что помогает пользователям анализировать содержание публикаций на основе графиков и диаграмм. Среди визуальных элементов:

* Диаграмма(Рис. 10) распределения ключевых слов: на этой диаграмме отображаются часто встречающиеся ключевые слова среди всех публикаций, что помогает пользователям понять, какие темы преобладают в выбранном наборе публикаций.
* Гистограммы(Рис. 11): отображают данные о количестве изображений, таблиц и источников, что позволяет визуально оценить содержание публикаций. Гистограммы также могут быть полезны для выявления аномалий, таких как статьи с необычным количеством визуальных материалов или ссылок.

Эти визуализации значительно упрощают анализ данных, помогая быстро выявить закономерности и особенности в коллекции публикаций.

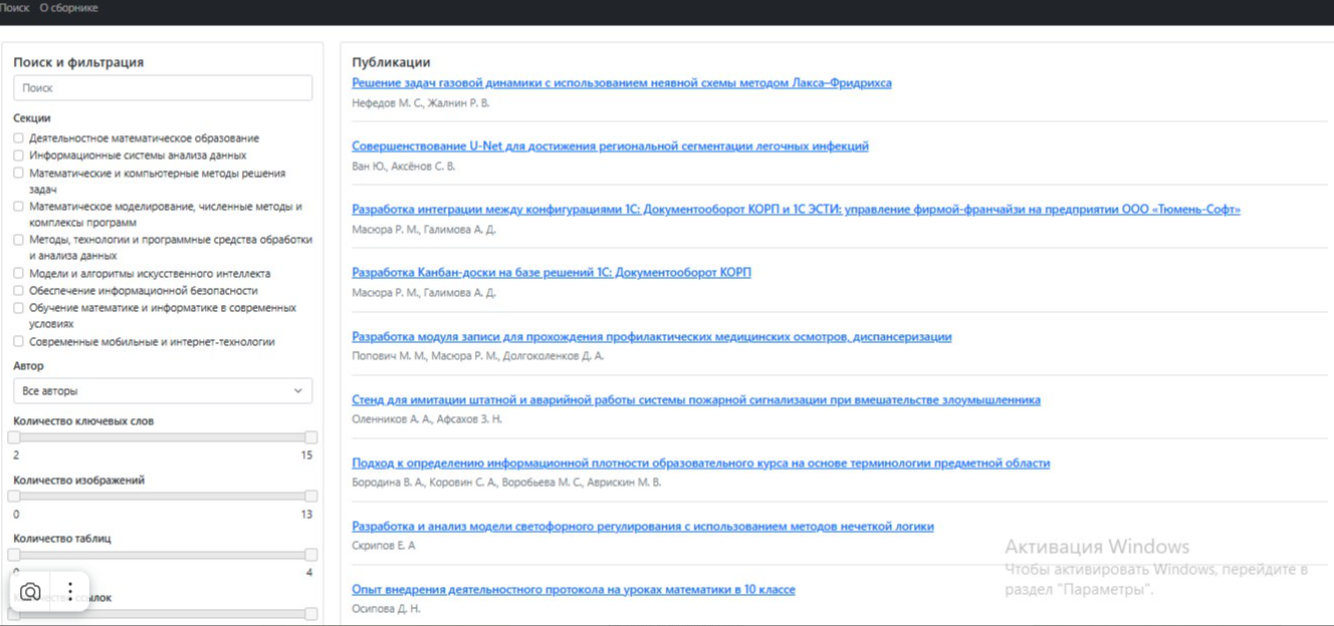


Рис. 8 Основной экран

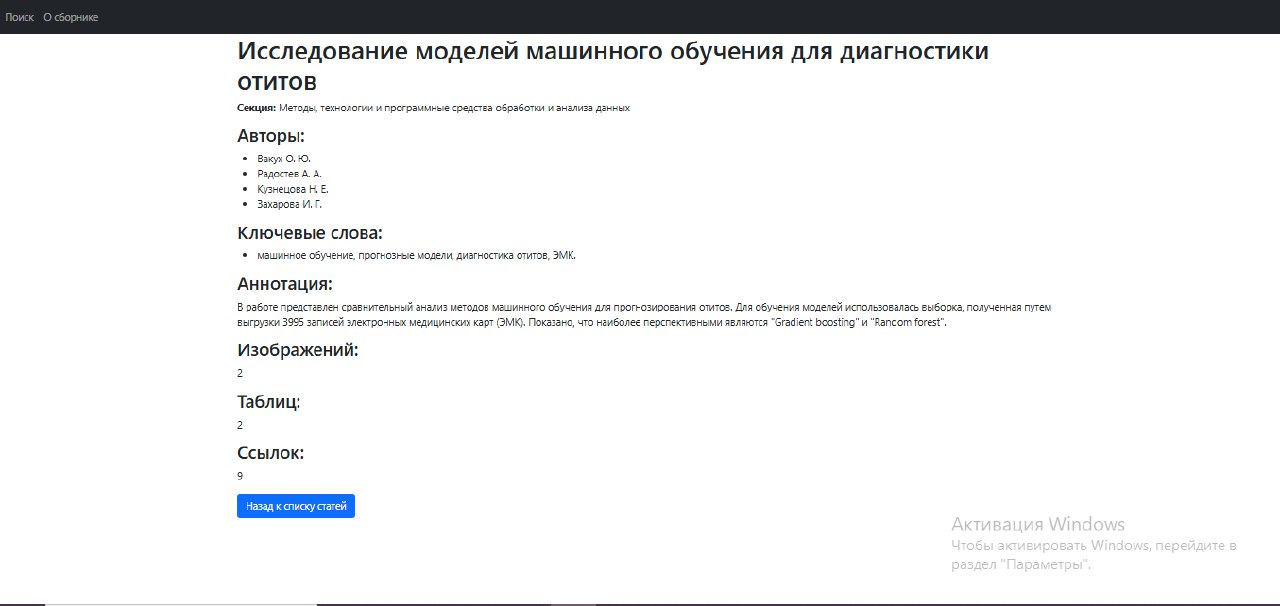
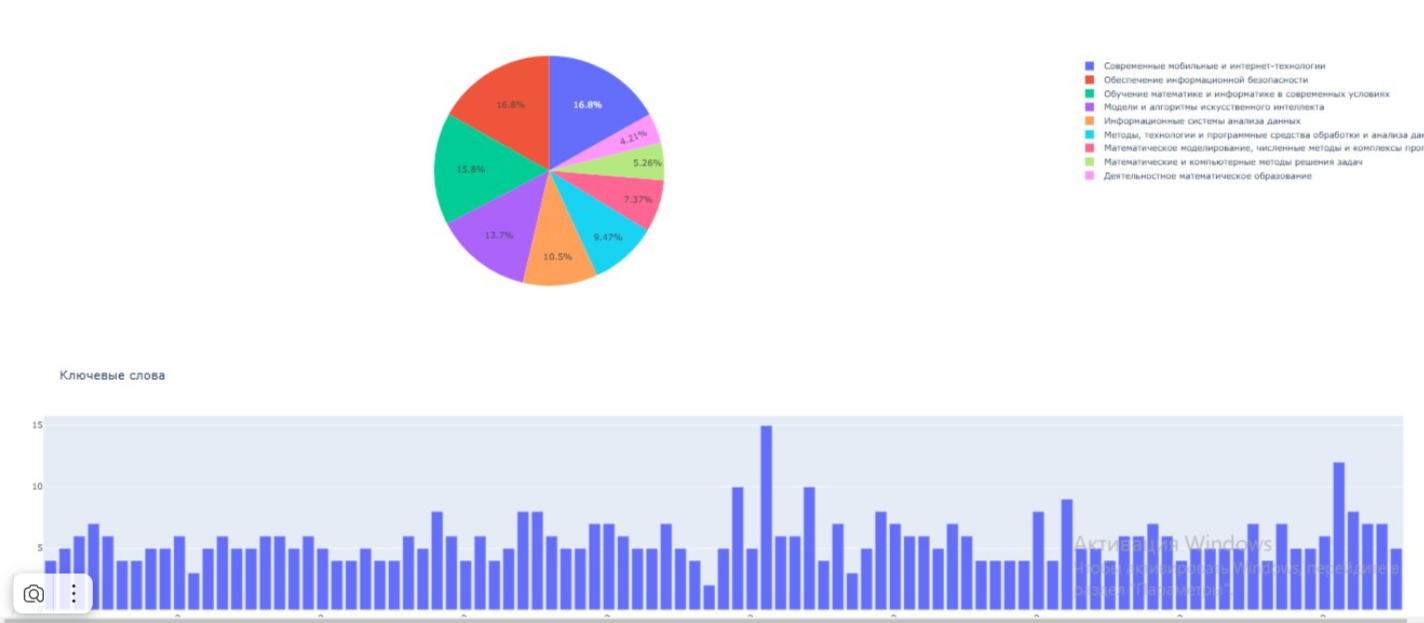
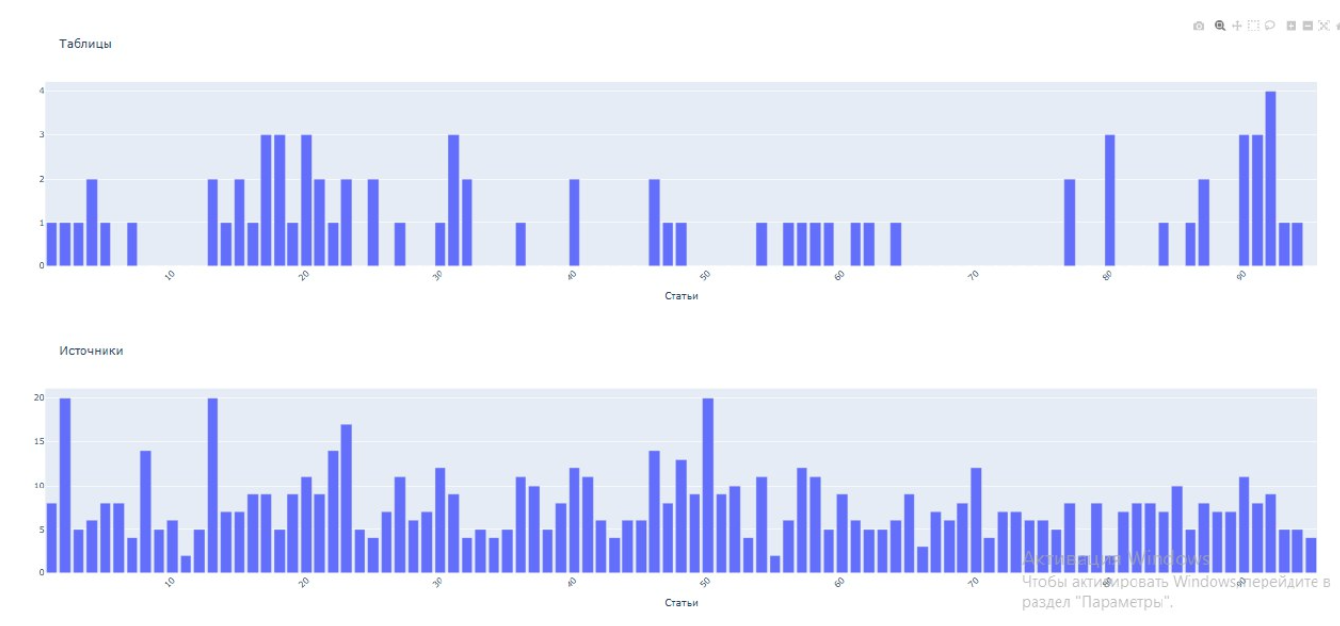


Рис. 9 Информация о статье

Рис. 10 Диаграмма распределения ключевых слов Рис 11. Гистограмма отображающая данные о статьях

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Arxiv.org. Архив научных статей по машинному обучению и обработке текста. [Электронный ресурс]. – URL: <https://arxiv.org/> (дата обращения: 15.11.2024).
2. Scopus. Аналитический инструмент для поиска научных публикаций. [Электронный ресурс]. – URL: <https://www.scopus.com/> (дата обращения: 12.10.2024).
3. Crossref Metadata Search. Сервис для поиска и извлечения метаданных научных статей. [Электронный ресурс]. – URL: <https://search.crossref.org/> (дата обращения: 14.11.2024).
4. PubMed. Репозиторий научных статей в области медицины и биологии. [Электронный ресурс]. – URL: <https://pubmed.ncbi.nlm.nih.gov/> (дата обращения: 10.10.2024).
5. Natural Language Toolkit (NLTK). Инструменты для обработки текста, включая токенизацию, стемминг, лемматизацию. [Электронный ресурс]. – URL: <https://github.com/nltk/nltk> (дата обращения: 16.10.2024).
6. Gensim. Библиотека тематического моделирования. [Электронный ресурс]. – URL: <https://github.com/RaRe-Technologies/gensim> (дата обращения: 14.12.2024).
7. Django Project. Документация. [Электронный ресурс]. – URL: <https://docs.djangoproject.com/> (дата обращения: 11.10.2024).
8. PostgreSQL Documentation. Полный набор документации для работы с PostgreSQL, включая методы работы с JSON и массивами. [Электронный ресурс]. – URL: <https://www.postgresql.org/docs/> (дата обращения: 11.10.2024).
9. Semantic Scholar. Автоматизированный поиск по научным статьям с извлечением метаданных и ключевых данных. [Электронный ресурс]. – URL: <https://www.semanticscholar.org/> (дата обращения: 16.12.2024).
10. SciGuide. Путеводитель по академической среде Высшей Школы Экономики. [Электронный ресурс]. – URL: <https://sciguide.hse.ru/> (дата обращения: 16.12.2024).
11. Яндекс Образование. Методы извлечения данных из текстов. [Электронный ресурс]. – URL: <https://education.yandex.ru/handbook/data-analysis/article/metody-izvlecheniya-dannyh-iz-tekstov> (дата обращения: 09.11.2024).
12. Яндекс Образование. Текстовые данные и способы их обработки. [Электронный ресурс]. – URL: <https://education.yandex.ru/handbook/data-analysis/article/tekstovye-dannye-i-sposoby-ih-obrabotki> (дата обращения: 09.11.2024).
13. НТВПРТ. Архив выпусков научно-технического вестника «Проблемы радиоэлектроники и телекоммуникаций». [Электронный ресурс]. – URL: <https://ntvprt.ru/ru/archive-vypuskov> (дата обращения: 16.12.2024).