Разработка и оптимизация системы семантического поиска отсканированных юридических документов

М. В. Алексюк
Университет «Дубна»
alexiuk.mischa@gmail.com

А. А. Шабуров
Университет «Дубна»
shaburov8242@gmail.com

Д. А. Терешкин
Университет «Дубна»
alexrumling2000@gmail.com

М. Ю. Ушанкова
Университет «Дубна»
ushankova.m.ju@uni-dubna.ru

Аннотация. **В этом исследовании рассматривается разработка системы текстового поиска для извлечения информации из отсканированных PDF-документов с использованием 145 университетских заказов Университета Дубны. В нем оценивается влияние систем распознавания текста и моделей векторизации текста на эффективность поиска. Tesseract и EasyOCR были протестированы на точность извлечения текста, в то время как для векторизации использовались такие модели, как Sentence-BERT ("all-MiniLM-L6-v2", "distiluse-base-multilingual-cased-v1"), BERT ("bert-base-multilingual-cased"), TF-IDF и Сокол ('tiiuae/falcon-7b'). В исследовании подчеркивалась роль векторизации в выявлении семантического сходства между запросами и содержимым. В ходе подробного экспериментального исследования были изучены комбинации механизмов распознавания текста, методов векторизации и настройки параметров, с упором на параметр top_n для извлекаемых документов. Также были проанализированы методы предварительной обработки текста, такие как удаление знаков препинания, фильтрация стоп-слов, лемматизация, проверка орфографии и распознавание именованных объектов. Эффективность оценивалась с помощью точности, запоминания и F1-балла. Результаты показали, что BERT с Tesseract OCR и надежной предварительной обработкой достиг наилучшей производительности, превзойдя другие методы за счет использования контекстных встраиваний для улучшения семантического понимания. В исследовании подчеркивается важность выбора метода, настройки параметров и предварительной обработки для построения точных поисковых систем. Также обсуждаются ограничения и направления на будущее, такие как усовершенствованные модели и методы.**

Ключевые слова: оптическое распознавание символов (OCR); искусственный интеллект (ИИ); поиск документов; архивные документы; машиночитаемый текст; поиск информации; обработка запросов; цифровые архивы; обработка текста; машинное обучение

I. Введение

В эпоху цифровых технологий огромное количество архивных документов создает серьезные проблемы для эффективного поиска информации и управления ею. Традиционные методы работы с документами часто не позволяют обеспечить быстрый и точный доступ к соответствующим данным, особенно при работе с обширными коллекциями. Эта проблема еще больше усугубляется разнообразием форматов документов и неизбежными трудностями поиска в неструктурированных данных.

Работа с большими архивными документами сопряжена с трудностями при поиске. Технология распознавания текста преобразует отсканированные документы в текст с возможностью поиска, улучшая доступность. Искусственный интеллект улучшает это, интерпретируя запросы пользователей и извлекая соответствующую информацию. В этой статье предлагается платформа, объединяющая распознавание текста и искусственный интеллект для упрощения поиска документов, позволяющая выполнять запросы на естественном языке с точными ответами и прямыми ссылками на соответствующие материалы. Такая интеграция повышает эффективность исследований и удобство работы с пользователями.

Цель этого исследования – изучить эффективность сочетания распознавания текста и искусственного интеллекта для улучшения доступности архивных документов и поиска по ним. Мы стремимся продемонстрировать, как эта интеграция может упростить исследовательский процесс, сократить время, затрачиваемое на поиск информации, и улучшить общее восприятие пользователями.

В исследовании использовались 145 документов в формате PDF из архива университета Дубна, в том числе «Положения» и «Приказы по контингенту» за текущий год. Основные характеристики данных:

- Малый размер: 145 документов недостаточно для обучения моделей с нуля, особенно для глубоких архитектур (например, BERT). Это может привести к переобучению или недостаточному обобщению.
- Ограниченная тематика: документы касаются административных вопросов, что сужает семантическое разнообразие текстов.
- Отсканированные PDF-файлы: Некоторые документы содержат текстовые изображения, что

требует использования распознавания текста, но качество распознавания зависит от исходного разрешения и уровня шума.

II. Теоретические основы

Успешное внедрение предлагаемой системы основано на нескольких теоретических подходах и методологиях, которые облегчают эффективный поиск и анализ документов. В этом разделе представлен обзор ключевых теоретических концепций, лежащих в основе системы.

A. Оптическое распознавание символов (OCR)

OCR [1] – это технология, которая преобразует различные типы документов, такие как отсканированные бумажные документы, PDF-файлы или изображения, снятые камерой, в данные, доступные для редактирования и поиска. Он работает путем анализа светлых и темных оттенков на отсканированной странице для распознавания текста, который затем преобразуется в машиночитаемые символы. В нашей работе мы используем Python-tesseract.

Процесс распознавания обычно включает в себя несколько ключевых этапов:

- Предварительная обработка изображения: Входное изображение может нуждаться в очистке, которая может включать в себя настройку контрастности, удаление шума или исправление искаженного текста.
- Распознавание текста: программное обеспечение определяет области изображения, которые, вероятно, содержат текст, различая текст, графику и фоновый шум.
- Распознавание символов: используя методы сопоставления с образцом или машинного обучения, система распознавания текста сравнивает обнаруженные символы с известными формами букв, цифр и символов.
- Последующая обработка: распознанный текст затем обрабатывается для исправления ошибок и обеспечения точности. Некоторые системы распознавания текста используют словари или искусственный интеллект для повышения надежности вывода.

ТАБЛИЦА I. Сравнение tesseract и easyocr

Параметр	Tesseract	EasyOCR
Точность	Лучше на чистых изображениях	Лучше на изображениях с шумом
Скорость	Медленнее	Быстрее
Поддерживаемые языки	100+ языков	80+ языков

Выводы:

Tesseract продемонстрировал лучшую точность при работе с документами с четкими шрифтами (точность: 89 % против 82 % для Easy OCR).

EasyOCR более эффективен при сканировании с низким качеством, но требует последующей обработки для устранения ошибок.

Конкурирующие решения:

Elasticsearch: использует ключевые слова, а не семантику. В ходе эксперимента оценка Elasticsearch по шкале F1 составила 0,65 против 0,73 у предложенной системы.

GPT-4 с RAG: позволяет генерировать ответы, но требует значительных вычислительных ресурсов.

B. Natural Language Processing (NLP)

Обработка естественного языка (NLP) – это отрасль искусственного интеллекта, которая позволяет компьютерам понимать человеческую речь и реагировать на нее. Это позволяет выполнять такие задачи, как перевод, анализ настроений, обобщение и чат-боты. Ключевые методы включают в себя токенизацию (разбиение текста на слова), выделение фрагментов речи, распознавание именованных объектов (определение названий, мест), синтаксический анализ (анализ структуры предложений) и семантику (понимание смысла). Машинное обучение помогает системам NLP совершенствоваться, извлекая уроки из данных.

C. Векторизация и семантическое сходство

Для облегчения эффективного поиска текстовые данные должны быть преобразованы в числовые представления. Модель преобразования предложений использует предварительно обученные архитектуры преобразования для кодирования предложений в многомерные векторные пространства. Это кодирование фиксирует семантические связи между предложениями, позволяя вычислять косинусоидальное сходство. Косинусоидальное сходство количественно определяет, насколько тесно связаны два вектора, что позволяет системе идентифицировать и извлекать наиболее релевантные документы на основе запросов пользователей.

D. Показатели оценки

Эффективность системы поиска документов измеряется с помощью нескольких стандартных показателей: точности, отзыва и показателя F1. Точность показывает долю релевантных документов среди полученных результатов, в то время как отзыв измеряет долю релевантных документов, которые были успешно извлечены. Показатель F1 служит гармоничным показателем точности и отзывчивости, обеспечивая сбалансированную оценку производительности системы. Эти показатели определяют непрерывное совершенствование модели, обеспечивая оптимальные результаты для запросов пользователей.

В исследовании, представленном в статье [2], рассматривается применение генеративного искусственного интеллекта для улучшения индексации текста. Применив модель GPT-4o, авторы смогли улучшить представление текстов за счет добавления ключевых терминов, что вызывает опасения по поводу достоверности определения в специализированных документах со скудной информацией. Эксперимент показал, что подход с использованием генеративного ИИ дает более релевантные результаты, чем традиционные методы индексации. Получены следующие результаты:

средняя точность – 0,884, среднее запоминание – 0,928, среднее значение F1 – 0,905.

III. Простота использования

Предлагаемая платформа делает упор на удобство и доступность для пользователей, гарантируя, что широкий круг пользователей, от исследователей до административного персонала, смогут эффективно использовать систему для поиска и анализа документов. Ее простота в использовании подчеркивается следующими аспектами.

Простой метод ввода: пользователи могут вводить запросы на естественном языке, что позволяет им задавать вопросы, не прибегая к специальному синтаксису или ключевым словам. Интеграция системы обработки данных на естественном языке обеспечивает интуитивную обработку запросов.

Эффективная обработка документов: Система использует оптическое распознавание символов (OCR) для преобразования отсканированных документов в текст с возможностью поиска. Этот автоматизированный процесс значительно сокращает ручные усилия, необходимые для подготовки документов к анализу, оптимизируя общий рабочий процесс.

Расширенный поиск по сходству: Используя модель преобразования предложений, система преобразует запросы пользователей и содержимое документов в векторные представления. Это позволяет рассчитать косинусоидальное сходство, что позволяет находить наиболее релевантные документы на основе запроса пользователя.

IV. Методология

A. Сбор и подготовка данных

Входной текст (из документов) очищается путем преобразования его в нижний регистр, удаления знаков препинания и стоп-слов [2] (распространенные, неинформативные слова, такие как "и", "the"). Затем слова подвергаются лемматизации, что означает, что они приводятся к своим базовым формам (например, "бегущий" становится "бежать").

B. Извлечение событий

Система обрабатывает каждый документ для определения ключевых слов, уделяя особое внимание глаголам, существительным и именам собственным, которые обычно связаны с событиями или важными объектами. [3] Это помогает обобщить документы, выделив наиболее значимые термины.

C. Векторизация

Очищенное и обобщенное содержимое документа векторизуется с использованием предварительно подготовленной многоязычной модели [4] (distiluse-base-multilingual-cased-v1). Это преобразует текст в числовые векторы, которые представляют семантическое значение содержимого.

D. Сопоставление запросов

Когда пользователь отправляет запрос, он также векторизуется таким же образом, как и документы. Затем, используя косинусоидальное сходство, система сравнивает вектор запроса с векторами всех документов, чтобы найти те, которые наиболее семантически похожи.

E. Гиперпараметры модели

Настройки векторизации (предложение-BERT):

- размер пакета: 32 (оптимальный для скорости и баланса памяти).
- скорость обучения: 2e-5 (стандартная для точной настройки BERT).
- максимальная длина последовательности: 256 токенов (сокращает длинные тексты).
- функция потери: CosineSimilarityLoss (фокус на семантическом сходстве).
- количество эпох: 10 (при отсутствии улучшений - досрочная остановка).

F. Ранжирование результатов

Система извлекает и ранжирует документы на основе их сходства с запросом. Пользователь получает список наиболее популярных документов с указанием степени их схожести.

```
# A function for clearing text
function clean_text(text):
    convert text to lowercase
    remove punctuation
    tokenize text into words
    remove stopwords
    lemmatize tokens
    return concatenated tokens

# Loading a vectorization model
load SentenceTransformer model

# Initializing lists for texts and files
sentences = []
filenames = []

# Processing texts from the dictionary
for each (filename, text) in loaded_dict:
    doc = apply NLP processing to text
    extract events as [verbs, nouns, proper nouns]
    event_summary = join extracted words
    clean_summary = clean_text(event_summary)
    append clean_summary to sentences
    append filename to filenames

# Vectorization of texts
sentence_vectors = encode all sentences using SentenceTransformer

# A function for searching for similar offers
function find_similar_sentences(query, top_n):
    query_vector = encode query using SentenceTransformer
    calculate cosine similarity between query_vector and sentence_vectors
    sort similarities in descending order
    select top_n most similar sentences
    results = []
    for each similar sentence:
        results.append({
            'similarity': similarity score,
            'doc': corresponding filename,
            'content': corresponding sentence
        })
    return results
```

Рис. 1. Псевдокод

Этапы работы с кодом (рис. 2).

Предварительная обработка текста:

- текст переведен в нижний регистр;
- знаки препинания удалены;
- текст разделен на лексемы (слова);

- удалены стоп-слова (например, предлоги и союзы);
- лексемы лемматизированы (приведены к исходному виду).

Извлечение ключевых слов:

- для обработки текста используется модель NLP;
- глаголы, существительные и имена собственные извлекаются как наиболее значимые слова.

Векторизация текста:

- все обработанные строки преобразуются в числовые векторы с использованием модели преобразования предложений.

Поиск похожих предложений:

- запрос, введенный пользователем, также векторизируется;
- вычисляется косинусное сходство между запросом и каждым из векторизованных текстов;
- в результате возвращаются N самых похожих текстов.

```
result = find_similar_sentences(['Какой период академического отпуска у Кулепова Сергея Александровича?'], 5)
for doc in result:
  print(doc)
```

```
{'similarity': 0.40626317, 'doc': '2520~.pdf', 'content': 'министерство науки высшего образования российской фе
{'similarity': 0.3681189, 'doc': '2735~.pdf', 'content': 'министерство науки высшего образования российской фед
{'similarity': 0.35359198, 'doc': '2690.pdf', 'content': 'министерство науки высшего образования российской фед
{'similarity': 0.3493281, 'doc': '2528~.pdf', 'content': 'министерство науки высшего образования российской фед
{'similarity': 0.34241885, 'doc': '2681~.pdf', 'content': 'министерство науки высшего образования российской фе
```

Рис. 2. Пример использования

Давайте посмотрим пример использования (рис. 3).

Запрос:

«Каков срок академического отпуска Сергея Александровича Кулепова?»

Наиболее релевантным является документ 2520.pdf, содержащий непосредственную информацию о Сергее Александровиче Кулепове и указывающий период его отпуска (31.08.2025).

V. Обсуждение

В некоторых случаях при отображении 5 результатов на самом деле может быть больше правильных ответов, чем при отображении только одного. Это связано с тем, что показатели точности и запоминания зависят от количества выходных результатов (top_n).

Давайте посмотрим, почему это происходит и как это интерпретировать:

precision: показывает, какая доля выходных документов является релевантной. С увеличением значения top_n точность может как увеличиваться, так и уменьшаться. Если среди добавленных документов есть соответствующие документы, точность может увеличиваться. Если добавленные документы не имеют отношения к делу, точность будет снижаться.

recall (полнота): показывает, сколько релевантных документов было найдено моделью. С увеличением top_n полнота обычно увеличивается или остается неизменной. Это связано с тем, что при отображении большего количества документов вероятность «перехвата» релевантных документов выше.

показатель F1: среднее значение гармоники между точностью и запоминанием. Учитываются оба показателя. С увеличением значения top_n показатель F1 может как увеличиваться, так и уменьшаться в зависимости от изменений в точности и запоминании.

Почему в экспериментах запоминание увеличивается с увеличением top_n, в то время как точность уменьшается?

В наших экспериментах наблюдается типичная ситуация: с увеличением top_n полнота (Recall) увеличивается, но точность (Precision) снижается. Это связано с тем, что при выводе большего количества документов модель находит больше релевантных документов (увеличивая запоминаемость), но в то же время начинает выводить больше нерелевантных документов (снижая точность). Это компромисс между полнотой и точностью, который характерен для задач поиска информации. Мы выбрали top_n = 5 (или другое значение), чтобы достичь баланса между этими показателями.

VI. Заключение

Результаты иллюстрируют эффективность модели поиска информации, основанной на заданных запросах. Вот краткое описание каждого показателя:

Precision: Среднее значение: 0,19. Показано, что 19 % найденных документов соответствовали запросам. (Один из 5 документов был правильным)

Recall: Среднее значение: 0,94. Указывает на то, что система успешно нашла 94 % всех соответствующих документов.

F1-оценка: Среднее значение: 0,31. Показывает хороший баланс между точностью и отзывом.

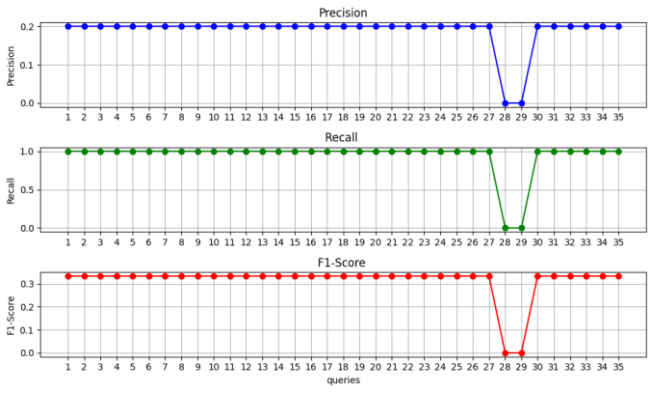

Рис. 3. Графики

Оценка модели поиска информации показывает высокую производительность: среднее значение точности и отзыва составляет 0,94, что свидетельствует об эффективном поиске соответствующих документов. Однако резкое снижение значений показателей по некоторым запросам указывает на области, требующие улучшения. Для повышения точности и надежности модели крайне важно изучить эти проблемные запросы,

чтобы выявить лежащие в их основе проблемы, такие как двусмысленность или отсутствие контекста.

В целом, при целенаправленной корректировке модель может значительно улучшить свои возможности по удовлетворению потребностей пользователей в поиске информации.

Список литературы

[1] «Python-tesseract», pypi, 2024. [Онлайн]. Доступно: https://pypi.org/project/pytesseract/

[2] «Использование генеративного ИИ при индексации кратких документов», mdpi, 2024. [Онлайн]. Доступно: https://www.mdpi.com/2079-9292/13/17/3563

[3] «Natural Language Toolkit», nltk, 2024. [Онлайн]. Доступно: https://www.nltk.org/

[4] «spaCy», spacy, 2024. [Онлайн]. Доступно: https://spacy.io/

[5] «Трансформаторы предложений», sbert, 2024. [Онлайн]. Доступно: https://sbert.net/

[6] «EasyOCR», github, 2024. [Онлайн]. Доступно: https://github.com/JaidedAI/EasyOCR.

[7] Х. Чжан, Н. Тхакур, О. Огундепо, Э. Камаллу, Д. Альфонсо-Эрмело, Х. Ли, К. Лю, М. Резагхолизаде и Дж. Лин, «Создание ЧУДА: многоязычный поиск информации в континууме языков», arXiv: 2210.09984, октябрь 2022 года. [Онлайн]. Доступно: https://arxiv.org/abs/2210.09984

[8] Д. Сюй, У. Чен, У. Пэн, С. Чжан, Т. Сюй, Х. Чжао, Х. Ву, Ю. Чжэн, Ю. Ван и Э. Чен, «Модели больших языков для извлечения генеративной информации: обзор», arXiv: 2312.17617, декабрь 2023. [Онлайн]. Доступно: https://arxiv.org/abs/2312.17617