Министерство науки и высшего образования Российской Федерации
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ (НИ ТГУ)
Институт прикладной математики и компьютерных наук

ДОПУСТИТЬ К ЗАЩИТЕ В ГЭК
Руководитель ОПОП
д-р техн. наук, профессор
_____ К.И. Лившиц
«06» июня 2024 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА

СОЗДАНИЕ ПРИЛОЖЕНИЯ, АНАЛИЗИРУЮЩЕГО ТЕКСТ

по направлению подготовки 01.03.02 Прикладная математика и информатика,
направленность (профиль) «Прикладная математика и информатика»

Литвак Александра Ильинична

Руководитель ВКР
канд. физ.-мат. наук, доцент
_____ Е.Г. Пахомова
«31» мая 2024 г.

Автор работы
студент группы № 932021
_____ А.И. Литвак
«31» мая 2024 г.

Томск – 2024

Министерство науки и высшего образования Российской Федерации.

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ (НИ ТГУ)
Институт прикладной математики и компьютерных наук

УТВЕРЖДАЮ
Руководитель ОПОП
д-р техн. наук, профессор
_____ К.И. Лившиц
подпись
«13» ноября 2023 г.

ЗАДАНИЕ

по выполнению выпускной квалификационной работы бакалавра обучающемуся
Литвак Александре Ильиничне
Фамилия Имя Отчество обучающегося

по направлению подготовки 01.03.02 Прикладная математика и информатика, направленность (профиль) «Прикладная математика и информатика»

1 Тема выпускной квалификационной работы
Создание приложения, анализирующего текст

2 Срок сдачи обучающимся выполненной выпускной квалификационной работы:
а) в учебный офис / деканат – 31.05.2024 б) в ГЭК – 06.06.2024

3 Исходные данные к работе:
Объект исследования – веб-приложение, анализирующее текст
Предмет исследования – процесс разработки веб-сайта
Цель исследования – разработать функциональный веб-сайт, для анализа текста

Задачи:
1. Изучить современный технологии и инновации, связанные с разработкой веб-приложений;
2. Изучить документацию необходимых NLP-библиотек;
3. Реализовать нужные алгоритмы;
4. Создать веб-сайт для более удобного взаимодействия с пользователем;

Методы исследования:
Изучение документации, методов NLP, разработка веб-сайта, выполняющего задачу

Организация или отрасль, по тематике которой выполняется работа, –
НИ ТГУ, Институт прикладной математики и компьютерных наук, кафедра компьютерной безопасности

4 Краткое содержание работы
В дипломной работе будет разработан и реализован веб-сайт для анализа текста.

Руководитель выпускной квалификационной работы
канд. физ.-мат наук, доцент, ТГУ
должность, место работы
_____ / Е.Г. Пахомова
подпись *И.О. Фамилия*

Задание принял к исполнению
Студентка группы 932021, ТГУ
должность, место работы
_____ / А.И. Литвак
подпись *И.О. Фамилия*

АННОТАЦИЯ

Данная работа затрагивает актуальную тему в эпоху развития количества информации, которую можно анализировать. Сейчас становится все более актуальной разработка инструментов и сервисов, которые смогут автоматически, быстро и бесплатно подготавливать к обработке и непосредственно обрабатывать и анализировать тексты. Одним из ключевых инструментов в данной теме являются методы обработки естественного языка (Natural Language Processing, NLP).

Целью данной работы является создание функционального веб-сайта, способного проводить различный текстовый анализ. Объектом исследования является сам процесс анализа текстовой информации с помощью веб-приложений, а предметом исследования являются методы и технологии, используемые для создания данного приложения. Это включает: алгоритмы обработки естественного языка, технологии веб-разработки, архитектурные решения для веб-приложений, интерфейсы взаимодействия пользователя и методы визуализации.

Результатом работы стал функциональный веб-сайт, обеспечивающий различные возможности анализа текста. В ходе работы были решены все поставленные задачи, были рассмотрены ключевые этапы создания приложения, начиная от изучения алгоритмов обработки и различного анализа текста, до их реализации и внедрения. Также была выполнена интеграция разработанного приложения с веб-интерфейсом для удобного взаимодействия с потенциальными пользователями.

Разработанный проект имеет практическую ценность, предлагая эффективный и удобный механизм для различных исследований.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ 6
1 Обзор 8
1.1 Анализ существующих решений 8
1.1.1 Анализ конкурентов 8
1.1.2 Трудности обработки русского языка в NLP 10
1.2 Стек технологий 11
1.2.1 Технологии и инструменты разработки клиентской части 11
1.2.1.1 HTML 11
1.2.1.2 CSS 11
1.2.2 Технологии и инструменты разработки серверной части 12
1.2.2.1 Jinja2 12
1.2.2.2 Flask 12
1.2.3 Фреймворки, библиотеки, модели и алгоритмы NLP 13
1.2.3.1 pymorphy2 13
1.2.3.2 spaCy 14
1.2.3.3 textBlob 14
1.2.3.4 googletrans 15
1.2.3.5 Расстояние Дамерау-Левенштейна 16
1.2.3.6 rake-nltk 17
1.2.3.7 Модели huggingFace 18
1.2.3.8 API Яндекс.Спеллер 18
2 Интерфейс 20
2.1 Общая информация о сайте и его архитектура 20

2.2 Информация о каждой странице сайта 22
2.2.1 Переводчик 23
2.2.2 Анализ тональности 23
2.2.3 Ключевые слова и суммаризация 24
2.2.4 Сравнения 26
2.2.5 Семантический граф 28
2.2.6 Морфологический анализ 29
ЗАКЛЮЧЕНИЕ 31
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ.. 32
ПРИЛОЖЕНИЕ А Исходный код ресурса 33

ВВЕДЕНИЕ

Объем текстовой информации, доступной для анализа и дальнейших выводов, в современном мире растет очень быстро. Поэтому становится все более актуальной разработка инструментов и сервисов, которые смогут автоматически, быстро и бесплатно подготавливать к обработке и непосредственно обрабатывать и анализировать тексты. Одним из ключевых инструментов в данной теме являются методы обработки естественного языка (Natural Language Processing-NLP).

Цель проекта: создать сайт с требуемой для проведения анализа текста на русском функциональностью, включая: морфологический анализ, проверка орфографии, извлечение именованных сущностей, анализ тональностей, построение поверхностно семантического графа, выделение ключевых слов, перевод, суммаризация и анализ схожести текстов.

Для достижения заданной цели были определены следующие задачи:

- Изучить современный технологии и инновации, связанные с разработкой веб-приложений;
- Изучить документацию необходимых NLP-библиотек;
- Реализовать нужные алгоритмы;
- Создать веб-сайт для более удобного взаимодействия с пользователем.

Прежде чем приступить к созданию проекта, был проведен анализ существующих решений. Основными аналогами являются веб-сайты AOT.ru и retext.AI. Первый сервис может помочь с анализом морфологии слов и построением семантического графа, а второй с суммаризацией и извлечением ключевых слов. Более подробна их функциональность и недостатки описаны в пункте 1.1.1 Анализ конкурентов.

Русский язык в связи со своей долгой историей, обширностью, огромным количеством правил и исключений, множеством падежных форм,

окончаний, различных видов спряжений и склонений, а также отсутствием четкой структуры предложений является очень сложным для изучения. Именно поэтому обработка текстов на русском – такая нетривиальная и интересная задача.

В результате проделанной работы был создан функциональный веб-сайт, предоставляющий широкий спектр возможностей для анализа текста. Все поставленные задачи успешно решены: рассмотрены ключевые этапы создания приложения, начиная с изучения алгоритмов обработки и анализа текста, и заканчивая их реализацией и внедрением. Особое внимание уделено интеграции разработанного приложения с веб-интерфейсом, обеспечивающим удобное взаимодействие с пользователями.

Выбор именно веб-приложения обусловлен удобством и универсальностью взаимодействий. Таким образом, не нужно скачивать отдельные приложения под каждую задачу.

Разработанный сервис может быть полезен лингвистам, людям, которые хотят профессионально изучать русский язык, иностранцам, которые хотят научиться говорить на нем правильно, а также аналитикам данных.

1 Обзор

1.1 Анализ существующих решений

1.1.1 Анализ конкурентов

После анализа рынка выяснилось, что основным конкурентом разработанного веб-ресурса является сайт *«Автоматическая Обработка Текста»* [8]. Данный ресурс предлагает следующую функциональность:

- Морфология:

Пользователь вводит русскую, английскую или немецкую словоформу и получает нормальную форму и морфологические атрибуты либо, по желанию, всю парадигму слова.

- Синтаксис:

Пользователь вводит предложение на русском языке и получает результаты анализа поверхностного синтаксиса в виде системы составляющих.

- Граф:

По одному предложению на русском языке строится поверхностно семантический граф.

- Перевод:

Перевод с русского языка на английский. Используются результаты графематического, морфологического и синтаксического анализаторов.

- Поиск по массиву:

Лингвистический поиск по размеченному морфологическим анализатором массиву. Можно искать по части речи и по морф. Характеристикам. Размеченный корпус состоит из 680 миллионов слов.

- Поиск по биграммам:

Поиск по леммным биграммам (54 млн.). Леммы получены с помощью программы Trigram.

Недостатки данного ресурса:

- Нет обновлений с 11 декабря 2021;
- Недостаточная функциональность (отсутствие возможности провести анализ тональностей, выделение ключевых слов, суммаризацию и анализ схожести текстов);
- Проблемы висячих вершин на некоторых тестовых данных в построении графа;
- Искусственный ограничения на количество знаков для анализа.

Ресурс удобен в использовании, а также является прямым конкурентом, поэтому было решено основываться на его функциональности и исправить его недостатки.

С суммаризацией и проверкой орфографии может помочь узконаправленный сервис *ReText.AI* [6].

Суммаризация ReText.AI поможет сократить текст онлайн до заданного количества знаков без потери смысла. Функция позволяется сделать выжимку из текста для конспекта, уменьшить текст статьи под социальные сети и сжать текст для изложения.

Недостатки ресурса:

- В бесплатном режиме доступна суммаризация только первых 1000 символов и 5 суммаризаций в день, а также для проверки орфографии доступно 1000 символов ежедневно;
- Ограниченная функциональность;
- Стоимость подписки варьируется от 246.58 до 699 рублей в месяц в зависимости от времени подписки (по состоянию на май 2024).

В целом сервис удобен, хорошо проводит суммаризацию, есть большое количество дополнительных функций, например перефразирование.

Разработчики передали, что используют архитектуру – Transformer для работы с текстом. Данные нейронные сети используются в Google переводчике, Яндекс переводчике и GPT-3.

Остальные сервисы обладают единичной функциональностью и не являются прямыми аналогами, поэтому было решено, чтобы проект представлял из себя комбинацию большого количества инструментов, нужных целевой аудитории.

1.1.2 Трудности обработки русского языка в NLP

NLP на данный момент является трендом, поэтому существует огромное количество библиотек, ML-моделей и датасетов. Однако большинство этих ресурсов нацелены на английский и немецкий языки, что создает некоторые сложности для разработчиков, работающих с текстами на других языках, включая русский.

Русский язык отличается высокой сложностью и многообразием грамматических конструкций, что затрудняет его обработку и анализ. В отличие от английского и немецкого, для русского языка гораздо меньше готовых решений и специализированных инструментов.

Сложность русского языка для анализа связано с рядом факторов:

- Большое количество правил и исключений к ним;
- Множество окончаний, различных видов спряжений и склонений, создающих дополнительные сложности при лемматизации и стемминге;
- Омонимы усложняют задачи по семантическому анализу;
- Отсутствие четкой структуры предложений усложняет синтаксический анализ.

1.2 Стек технологий

1.2.1 Технологии и инструменты разработки клиентской части

1.2.1.1 HTML

HTML (HyperText Markup Language) – это язык разметки, используемый для создания и структурирования содержимого веб-страниц. Он позволяет определять заголовки, параграфы, списки, ссылки, изображения и другие элементы, составляющие веб-страницу.

Выбор был остановлен именно на HTML ввиду того, что он является стандартом для всех веб-браузеров, а также из-за его интерактивности и простоты использования и активной поддержки сообщества посредством постоянных обновлений.

1.2.1.2 CSS

CSS (Cascading Style Sheets) – это язык стилей, используемый для описания внешнего вида и форматирования HTML-документов. CSS позволяет изменять шрифты, цвета, отступы, выравнивание, позиционирование и другие аспекты стиля элементов на веб-странице.

CSS был выбран из-за возможности повторно использовать стили, легкой интеграцией с Flask и простотой в использовании. Данный проект не предусматривает сложные стили, ведь каждый элемент функциональности разнесен на отдельные страницы, что позволяет не перегружать каждое окно и повысить читаемость и интуитивность интерфейса. Такой подход облегчает дальнейшее масштабирование проекта.

1.2.2 Технологии и инструменты разработки серверной части

1.2.2.1 Jinja2

Jinja – это быстрый и расширяемый шаблонизатор. Специальные заполнители в шаблоне позволяют писать код, аналогичный синтаксису Python. Jinja позволяет динамически вставлять данные в шаблоны и управлять отображением содержимого с помощью условных конструкций и циклов.

Он компилирует шаблоны в байт-код, что ускоряет обработку. Также он по умолчанию экранирует переменные, выводимые в HTML, что предотвращает атаки XSS (межсайтовый скриптинг).

Jinja позволяет создавать динамические веб-страницы, где содержимое изменяется в зависимости от данный, также легко интегрирует с Python-приложениями.

1.2.2.2 Flask

Flask – это фреймворк для Python, предназначенный для разработки веб-приложений. Flask отличается простотой и гибкостью, предоставляя разработчику возможность выбирать необходимые компоненты и легко расширять функциональность приложения.

- Flask предоставляет минимум предустановленных настроек и компонентов, что делает его легким и простым в использовании;
- Flask позволяет легко интегрировать различные расширения и библиотеки, обеспечивая модульность приложения;
- Flask имеет встроенную поддержку Jinja2;
- Flask имеет обширную документацию и активное сообщество, что облегчает процесс разработки и поиска решений.

1.2.3 Фреймворки, библиотеки, модели и алгоритмы NLP

1.2.3.1 pymorphy2

Pymorphy2 [7] – морфологический анализатор. Его основными функциями является приведение слова к нормальной форме (для глаголов – инфинитив, для существительных – именительный падеж, единственное число), ставить слово в нужную морфологическую форму и возвращать грамматическую информацию о слове (например: число, род, падеж, часть речи, время, наклонение и так далее).

В данном проекте эта библиотека используется для страницы с функциональности морфологического анализа. Ниже приведены функции, использующие pymorphy2:

- Анализ первичной морфологии слова:
  - Получение нормальной формы слова
  - Получение грамматической информации о слове
- Подсчет количества разных частей речи в тексте
- Поиск и сохранение в список существительных и глаголов
- Извлечение именованных сущностей
- Получение всех форм отправленного слова (извлечение всех словоформ)

Pymorphy использует словарь OpenCorpora, который для русского языка содержит около 400 тысяч лексем и 5 миллионов отдельных слов. Для неизвестных слов строятся предположения. Чтобы сэкономить оперативную память и обеспечить быстрый анализ как словарных, так и несловарных слов pymorphy2 извлекает парадигмы из лексем, преобразует информацию в цифры и кодирует слова в Directed acyclic word graph - направленный ациклический граф слов.

1.2.3.2 spaCy

Spacy – популярная библиотека для обработки естественного языка. Она имеет поддержку 75+ языков, в том числе и русского, 84 предобученных моделей для 25 языков, встроенный визуализатор и другие функции, необходимые для анализа.

На рисунке 1 представлена архитектура модели. Конвейеры обучения разработаны с учетом эффективности и возможности настройки. Например, несколько компонентов могут использовать общую модель «токен-вектор», а лемматизатор можно заменять или исключать. В последней версии некоторые компоненты зависят от результата предыдущих моделей.

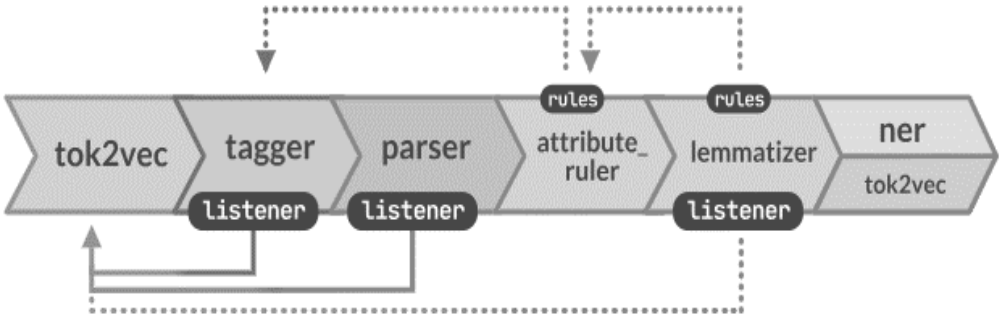

Рисунок 1 – Архитектура модели

В проекте используется модуль displacy, который позволяет построить по предложению семантический граф, который наглядно показывает, как именно компьютер понимает естественный язык.

1.2.3.3 textBlob

TextBlob – это библиотека Python для обработки текстовых данных. Она предоставляет простой API для решения распространенных задач обработки естественного языка (NLP), таких как тегирование частей речи, извлечение фраз существительных, анализ настроения, классификация и многое другое.

Эта библиотека используется в проекте для анализа тональностей текстов из файлов. После обращения к библиотеке функция возвращает именованный кортеж вида: Sentiment(polarity, subjectivity). Оценка полярности (polarity) – вещественное число в диапазоне [-1.0; 1.0], где -1.0 означает резко отрицательную, 0 – нейтральную, а 1.0 – резко положительную тональность. Субъективность (subjectivity) – так же вещественное число, только в диапозоне [0.0; 1.0], где 0.0 – очень объективная оценка (подходящее всем), а 1.0 резко субъективная оценка (что-то очень личное).

Например, предложение «Этот ресторан отличается прекрасным обслуживанием, отличным качеством и свежестью еды, мне очень понравилось проводить время здесь, советую всем!» с помощью библиотеки был оценен, как положительная тональность и высокий уровень объективности: Sentiment(polarity=0.8966666666666666,

subjectivity=0.1333333333333334)

А предложение: «Лично мне фильм показался скучным и растянутым, потому что я не люблю боевики, я едва удерживал интерес до конца.»:

Sentiment(polarity=-0.69791666666666666,
subjectivity=0.7854166666666667)

К сожалению, библиотека textBlob работает исключительно с текстом, написанном на английском языке, поэтому для других функций были использованы другие библиотеки, а для анализа тональностей предварительно выполняется машинный перевод с помощью API googletrans.

1.2.3.4 googletrans

GoogleTrans – это бесплатная и неограниченная библиотека, реализующая API Google Translate. Она не является официальной и использует Google Translate Ajax API для вызова таких методов, как detect и translate, использует те же серверы, что и translate.google.com.

Недавно Google обновили службу перевода, добавив механизм заявок, чтобы предотвратить работу программ-сканеров. Автор библиотеки обошел эту систему путем генерирования тикета с помощью метода обратной разработки.

1.2.3.5 Расстояние Дамерау-Левенштейна

Расстояние Левенштейна – метрика, измеряющая по модулю разность между двумя последовательностями символов. Она определяется как минимальное количество односимвольных операций (вставка, удаление, замена), необходимых для превращения одной строки в другую.

В данном проекте алгоритм используется для сравнения двух строк. Так как чаще всего при компьютерном наборе текста ошибками являются нажатие клавиши клавиатуры, рядом с необходимой (замена), пропуск буквы (вставка) и случайное повторное нажатие клавиш, например при залипании (удаление), то метрика Левенштейна достаточно хорошо проверяет данные тексты.

Если к списку разрешённых операций добавить транспозицию (замена местами двух соседних символов), получается расстояние Дамерау-Левенштейна. Для неё также существует алгоритм, требующий O(M*N) операций. Дамерау показал, что 80% ошибок при наборе текста человеком являются транспозициями.

В работе «Сравнительный анализ алгоритмов редакционного расстояния Левенштейна и Дамерау-Левенштейна» [2] на основе эксперимента было доказано, что несмотря на то, что алгоритм Дамерау-Левенштейна показывает меньшую эффективность на больших строках, часто при решении задачи автоматического исправления ошибок, например, в поисковых запросах, он будет работать эффективнее, чем алгоритм Левенштейна. Это, как раз связано с ошибками-транспозициями.

Для вычисления расстояния используется матричный алгоритм Вагнера-Фишера, представленный на рисунке 2.

$$D(i,j) = \begin{cases} 0, & i=0,\ j=0 \\ i, & j=0,\ i>0 \\ j, & i=0,\ j>0 \\ \min\{ & \\ \quad D(i,j-1)+1, & \\ \quad D(i-1,j)+1, & j>0,\ i>0 \\ \quad D(i-1,j-1)+\mathrm{m}(S_1[i],S_2[j]) & \\ \} & \end{cases}$$

Рисунок 2 – Алгоритм Вагнера-Фишера

Так как нам не нужно хранить все элементы матрицы можно сохранять только текущую и предыдущую строку для повышения производительности.

1.2.3.6 rake-nltk

Алгоритм Rake (Rapid Automatic Keyword Extraction) является одним из гибридных (лингвостатистических) алгоритмов для выделения ключевых слов в текстах. Он основан на предположении о том, что ключевые элементы могут быть неоднословными, не содержат знаков пунктуации, служебных и десемантизированных слов (слова, частично или полностью семантически опустошенные, и, следовательно, малоинформативные или вовсе неинформативные). Ключевые выражения, выделенные в тексте с учетом словаря разделителей, ранжируются по весу, определяемого как сумма трех метрик (частота, степень (мера совместной встречаемости), отношение частоты к степени).

В работе «Эксперименты по автоматическому выделению ключевых выражений в стилистически разнородных корпусах русскоязычных текстов» [3] был проведен анализ методов извлечения ключевых слов и их сравнение. Алгоритм RAKE дал достаточно хорошие результаты, поэтому было решено использовать именно его.

В проекте использовалась библиотека rake-nltk, которая позволяет автоматически производить извлечение ключевых слов и выражений из

текстов естественного языка, библиотека предоставляет удобный интерфейс. Одним из ключевых преимуществ использования rake-nltk является её способность работать с большими объемами текста, обеспечивая быстрое и точное извлечение ключевых слов.

1.2.3.7 Модели huggingFace

HuggingFace – одна из популярных платформ для обработки естественного языка. На этом сайте собрано некоторое количество предобученных моделей, в том числе для суммаризации текстов на кириллице.

Все модели должны быть обучены на каком-либо наборе исходных данных. Самыми популярными являются датасеты, использующие заголовки статей газет в качестве целевой переменной и сами тексты в качестве признаков или входных переменных. Также часто используются собранные посты из мессенджеров и социальных сетей, где целевым являются первые строки или заголовки, а остальной текст поста является признаками.

Выбранная модель обучена на заголовках российских новостных изданий. Именно эта модель показалась мне самой логчиной для тестовых текстов.

1.2.3.8 API Яндекс.Спеллер

Яндекс.Спеллер – сервис проверки правописания, который предлагает разработчикам использовать возможность интерактивной проверки орфографии на страницах своих сайтов. Спеллер анализирует слова, основываясь на правилах орфографии и лексике современного языка, а также использует технологии машинного обучения (ML, Machine Learning). Поддерживает 3 языка: русский, украинский и английский.

Сервис может исправить следующие ошибки:

- неправильные, пропущенные или лишние буквы;

- неправильное употребление прописных и строчных букв;
- повторы слов;
- неправильное написание в зависимости от контекста.

В проекте API используется для выявления орфографических ошибок на странице морфологического анализа.

2 Интерфейс

2.1 Общая информация о сайте и его архитектура

В данном параграфе будет дана классификация проекта по разным аспектам.

1. Назначение:

Назначение сайта – предоставление пользователям возможности анализа текстовой информации с использованием современных методов обработки естественного языка (NLP). Сайт предлагает функции, такие как морфологический анализ, построение семантического графа, сравнение текстов, выделение ключевых слов, суммаризация, анализ тональности и переводчик.

2. Уровень интерактивности:

Сайт является динамическим, поскольку контент и результаты анализа генерируются в процессе работы с пользователем и добавляются по мере готовности. Динамичность сайта достигается за счет использования серверных технологий и взаимодействия с внешними библиотеками для обработки данных. Все результаты показываются на той же странице, не переходя на другие.

3. Целевая аудитория:

Сайт ориентирован на B2C (Business-to-consumer) рынок. Он предоставляет услуги непосредственно конечным пользователям, которые могут быть как частными лицами, так и представителями малого и среднего бизнеса, заинтересованными в анализе текстовых данных для различных целей, таких как маркетинг и SEO (search engine optimization) исследования, анализ отзывов, мониторинг социальных сетей, а также изучения русского языка, как лингвистами на профессиональном уровне, так и людьми, которые в совершенстве не знают язык и хотят лучше и правильнее его понимать и на нем говорить.

4. Структура

Веб-ресурс является многостраничным (MPA, Multi-Page Application). Он включает несколько взаимосвязанных страниц, с которых можно делать взаимосвязанные переходы, таких как:

- Главная страница;
- Переводчик;
- Ключевые слова и суммаризация;
- Сравнения;
- Семантический граф;
- Морфологический анализ

Такая структура не принуждает пользователя каждый раз переходить на главную страницу, если ему понадобилась какая-либо другая функция. Про каждую из этих страниц более подробно написано в параграфах 2.2.

5. Тип хостинга

На данный момент сайт размещен на локальном сервере, так как находится на стадии тестирования. В будущем планируется размещение на хостинге для доступа широкой аудитории.

6. Кастомизация

Веб-ресурс является кастомизированным. Он разработан с нуля и не использует в своей клиентской части каких-либо шаблонов или конструкторов. Это обеспечивает высокую гибкость и возможность адаптации.

7. Архитектура

Серверная часть сайта использует монолитную архитектуру, представленную на рисунке 3, что означает, что все компоненты взаимодействуют напрямую через вызов функций и методов. Запуск файлов обработки из routes.py добавляет модульность.

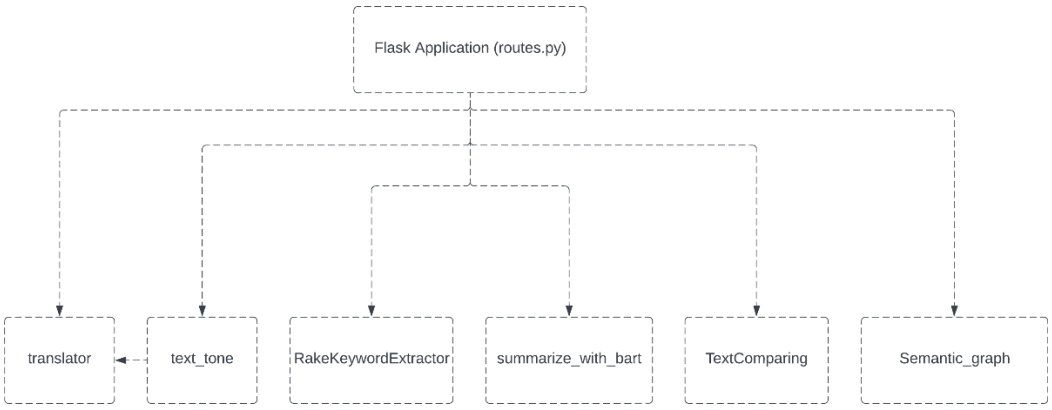

Рисунок 3 – архитектура серверной части

2.2 Информация о каждой странице сайта

Все страницы содержат сайдбар, находящийся в левой части экрана (рисунок 4), для легкого перехода к необходимым функциям, без необходимости возвращаться на начальную страницу. Также, в хедере всех страниц, кроме главной, есть кнопка для перехода к начальной странице.

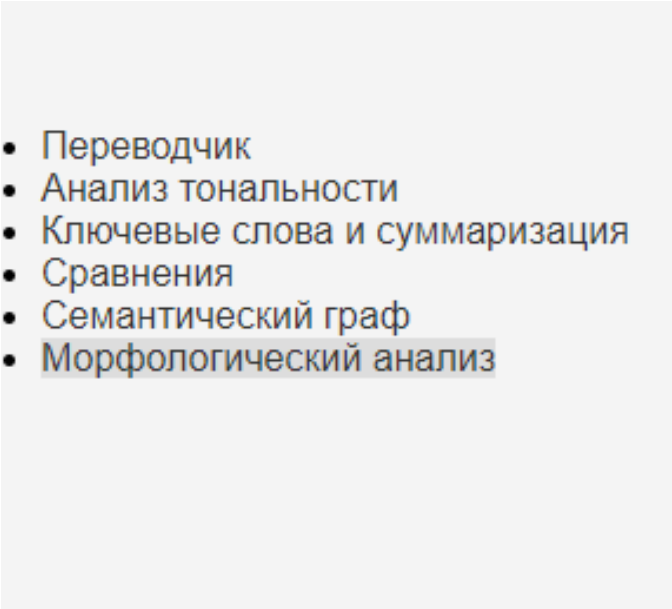

Рисунок 4 – Сайдбар

Настроены необходимые стили и визуализация.

2.2.1 Переводчик

Как показано на рисунке 5 сначала пользователю в специальное окно необходимо ввести текст. После этого при нажатии на кнопку «Перевести», отравляется POST – запрос, в котором в переменную сохраняется значения поля, и она передается в файл translator.py, который, с помощью API googletrans делает автоматический машинный перевод. После этого возвращает уже переведенный текст и динамически, с помощью Jinja встраивает в шаблон. Произведена обработка исключений, связанных с тем, что пользователь нажал кнопку, но никакой текст не был введен.

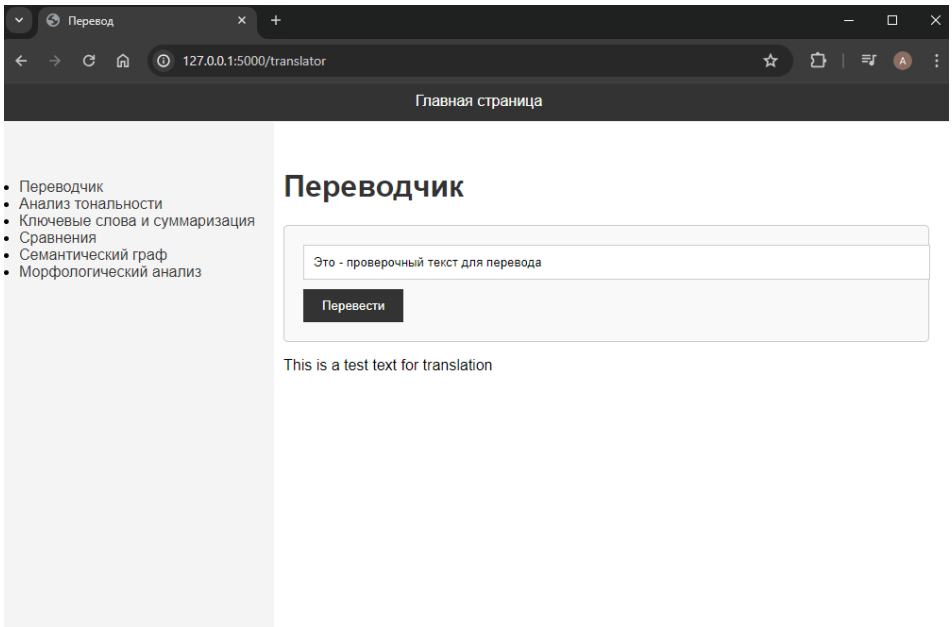

Рисунок 5 – Демонстрация работы страницы «Переводчик»

2.2.2 Анализ тональности

На рисунке 6 представлен интерфейс страницы «Анализ тональности». Пользователю предлагается загрузить файл в формате .txt и нажать на кнопку «Upload and Analyze». Необходимо, чтобы каждое отдельное высказывание, или отзыв было на отдельной строке. После проверяется, что файл был загружен, проводится предобработка текста (разделение по строкам, очистка

и автоматический машинный перевод). После этого список очищенных и переведенных строк отправляется в файл text_tone.py, в котором, с помощью библиотеки TextBlob узнается тональность текста и его субъективность, а также средний показатель тональности.

Так как тональность представляет собой величину в промежутке [-1.0; 1.0], а отзывы чаще всего представлены в промежутке [1.0; 5.0] было принято решение воспользоваться линейной трансформацией. Итоговая формула выглядит: Y = 2 * X + 3, где X – исходная величина, полученная из TextBlob, Y – конечная.

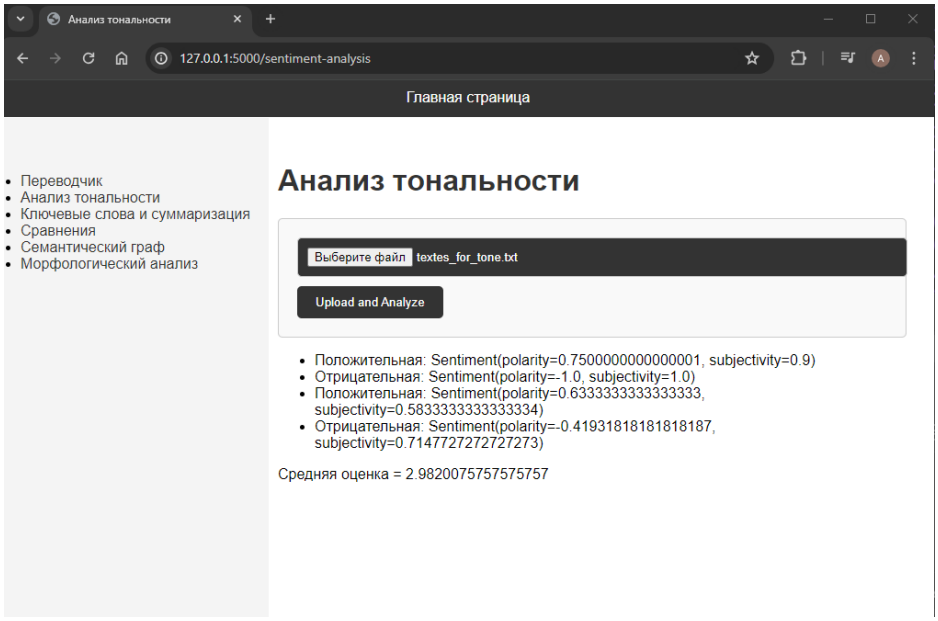

Рисунок 6 – Демонстрация работы страницы «Анализ тональности»

Средний показатель тональности может быть использован для оценки правдивости оценок мест, заведений, фильмов или другим объектам, на которые пользователи ставят оставляют оценку и отзыв.

2.2.3 Ключевые слова и суммаризация

Ключевые слова помогают алгоритмам поисковых систем находить веб-страницы, которые следует показать пользователям, а также часто используются в научных статьях. Правильный подбор ключевых слов

помогает бизнесу лучше продать свои услуги посредством более релевантных названий для продвижения товаров и услуг.

На рисунке 7 продемонстрирован пользовательский интерфейс. Для начала пользователю необходимо прикрепить непустой файл в формате .txt в форму, далее нажать на кнопку «Upload and Analyze». В это время файл открывается, проверяется, что он был добавлен и является непустым, и его содержимое отправляется в файл keywords_and_summarize.py, а именно в класс RakeKeywordExtractor, в котором, с помощью алгоритма RAKE, описанного в подпункте 1.2.3.6 выполняется извлечение ключевых слов. Результат возвращается в виде списка кортежей, содержащих само слово и его «вес» - важность в данном тексте. А далее, уже на странице распаковывается и записывается динамически в виде ненумерованного списка.

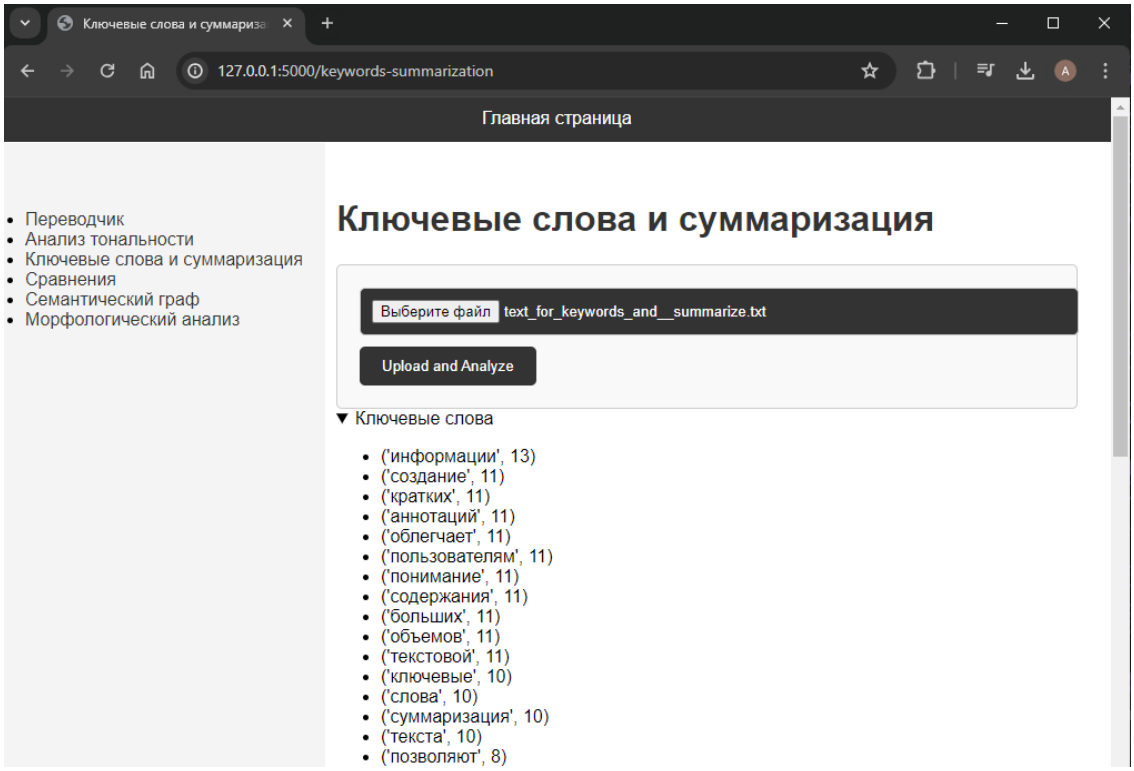

Рисунок 7 – Демонстрация работы извлечения ключевых слов

Суммаризация – удобный инструмент для экономии времени. Он помогает быстро понять основное содержание большого текста без необходимости читать его полностью.

На рисунке 8 представлен пример работы модуля суммаризации. Так как было принято решение проводить извлечение ключевых слов и суммаризацию на одной странице, то и обработка происходит с тем же файлом. Полученный из файла текст напрямую отправляется в файл keywords_and_summarize.py, а именно в функцию summarize_with_bart, которая на вход получает текст и мой личный API_TOKEN, полученный после регистрации на сайте HuggingFace. Далее, после отправки POST запроса напрямую к выбранной модели возвращается уже суммаризованный текст. Готовое предложение возвращается в routes.py, в котором возвращается html шаблон с результатом.

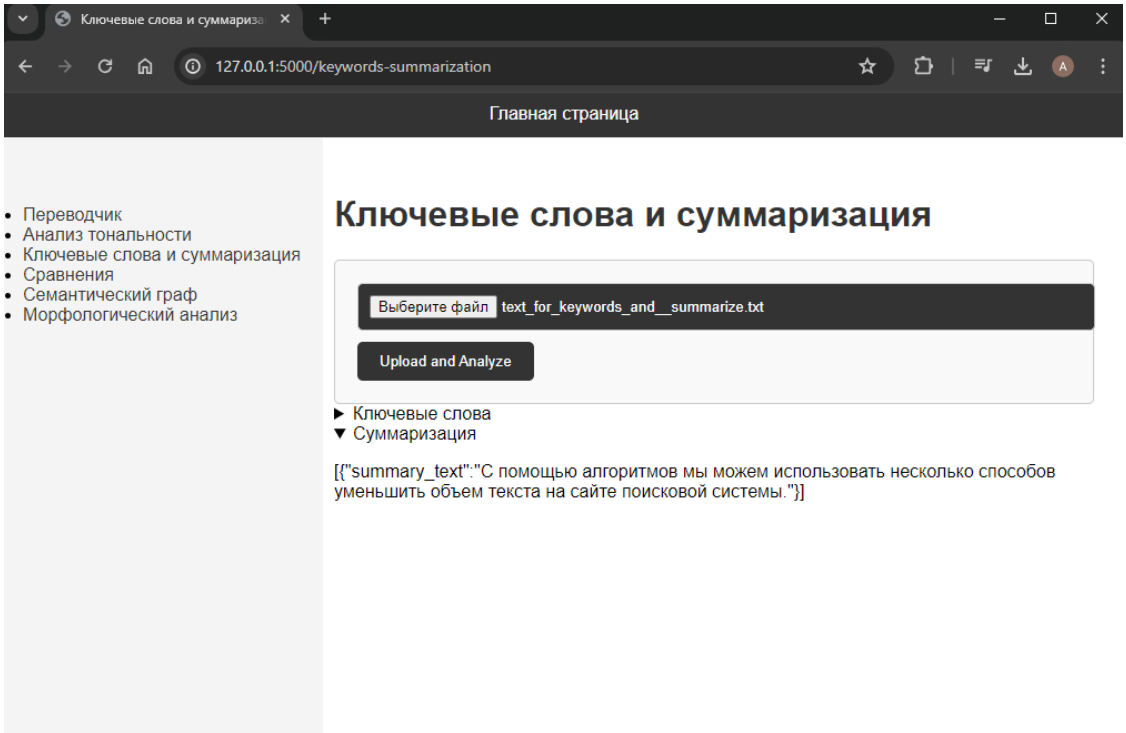

Рисунок 8 – Демонстрация работы суммаризации

2.2.4 Сравнения

На странице «Сравнения» используется метрика Левенштейна. Такой инструмент просто необходим при анализе текстовых данных. На основе такого алгоритма можно создать модуль, который будет помогать исправлять ошибки и опечатки на этапе очистки или предобработки датасета.

На рисунке 9 продемонстрирован результат работы алгоритма Вагнера-Фишера для метрики Левенштейна.

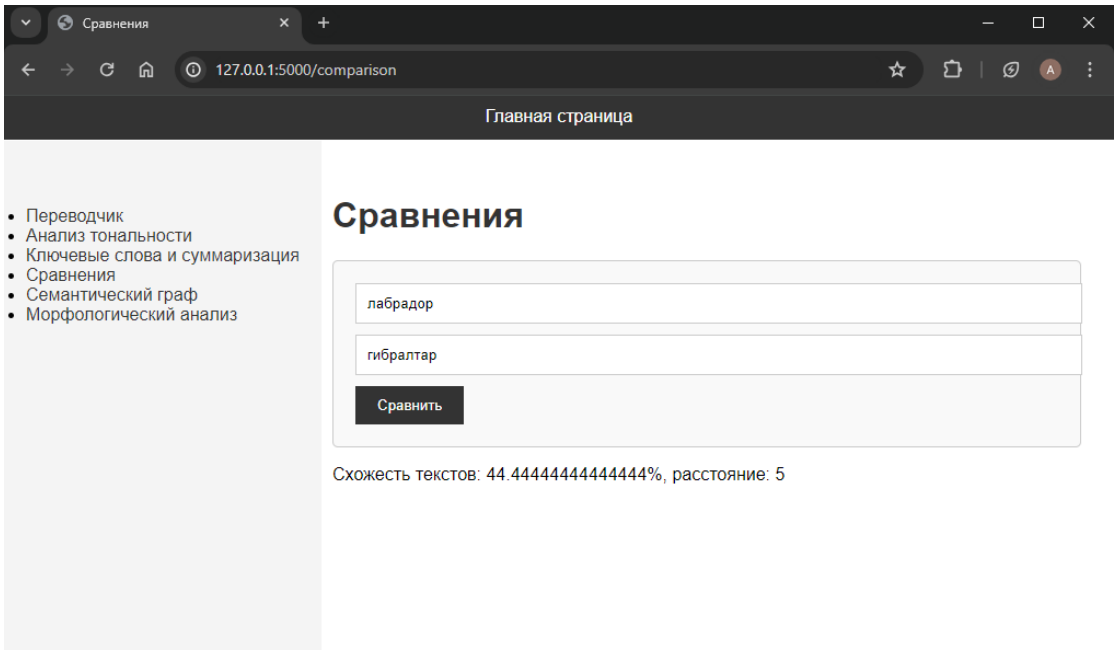

Рисунок 9 – Демонстрация работы страницы «Сравнения»

Для начала пользователь вводит 2 выражения, которые он хочет сравнить. Далее, после нажатия интуитивной кнопки, эти 2 слова сохраняются через форму в переменные в соответствующей функции файла routes.py. Осуществляется проверка на наличие символов в элементах формы, в случае пропуска выводится соответствующее предупреждение. Создается экземпляр класс TextComparing, в который передаются значения.

В классе сначала происходит предобработка текста (приведение к нижнему регистру, удаление лишних знаков с помощью регулярных выражений) и высчитывается в матричном виде расстояние. Значение расстояния и схожести текста (similarity=1-(distance/max_length)*100%, где similarity – значение схожести текста, distance – посчитанная метрика расстояния, max_length – длина наибольшего выражения) возвращаются обратно в routes.py и вставляются в шаблон.

2.2.5 Семантический граф

В общем виде семантическая сеть – множество вершин, каждая из которых соответствует определенному объекту (понятию, факту, явлению или процессу), а между вершинами заданы различные отношения, представляемые дугами.

Достоинства:

- большие выразительные возможности, естественность и наглядность системы знаний, представленной графически;
- близость структуры сети, представляющей систему знаний, семантической структуре фраз естественного языка;
- данная модель более других соответствует современным представлениям об организации долговременной памяти человека.

Недостатки:

- громоздкость и неэффективность представления знаний только аппаратом семантической сети;
- сложность организации процедуры поиска нужного знания (как фрагмента сети).

Страница «Семантический граф» представлена на рисунке 10. Она ожидает от пользователя ввода предложения и нажатия кнопки «Построить граф». После этого введенное предложение отправляется в файл semantic_graph.py. В этом файле с помощью библиотеки spaCy и модели «ru_core_news_sm»[1] создается svg-разметка, в которой заменяются английские названия частей речи и зависимостей на русские для более понятного интерфейса. Перевод зависимостей был выполнен вручную, он включает в себя все возможные слова и аббревиатуры, которые могут использоваться в

[1] Источник тренировочных данных для модели – корпус, содержащий более 700 тыс. новостных заголовков Lenta.ru: https://github.com/natasha/nerus

разметке. Далее, уже переведенная и отредактированная разметка возвращается в semantic_graph.html и вставляется напрямую.

Как видно на примере, под словами указывается их часть речи, а под дугами – отношения между ними.

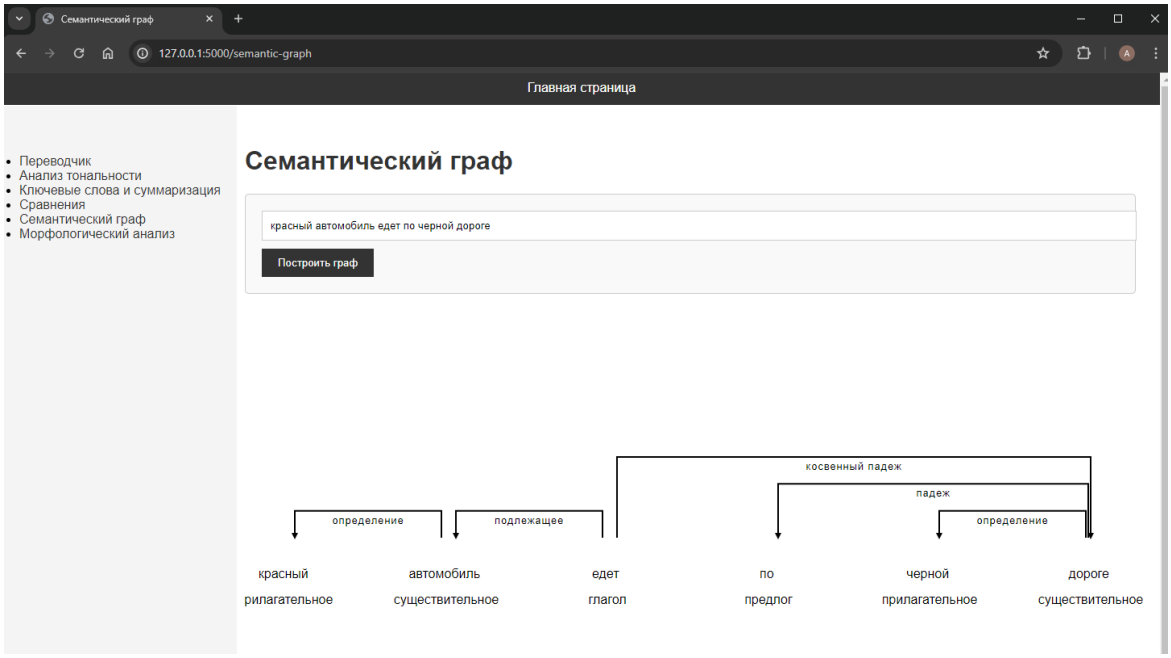

Рисунок 10 – Демонстрация работы страницы «Семантический граф»

2.2.6 Морфологический анализ

Страница морфологического анализа предлагает обширные статистические критерии оценки текста, анализ морфологии слов, извлечение именованных сущностей и исправление опечаток.

К статистическим функциям относится: количество слов, количество именных частей речи, частота встречаемости по словам и частям речи.

Именованные сущности (Named Entity Recognition, NER)- слова или словосочетания, которые выделяют определенный предмет или явление из ряда однотипных предметов и явлений. К ним относятся: ФИО, город, название компаний и организаций, адреса, названия книг, фильмов и другие.

Извлечение именованных сущностей нужно для большого числа задач, например:

- анонимизация текста;
- автоматическое создание метаданных;
- создание геоинформационной системы (GIS) для дальнейшего анализа;
- Обнаружение упоминаний и мониторинг новостей.

В данном проекте извлечение сущностей работает с помощью библиотеки pymorphy2, проверяются теги (наборы граммем, характеризующих данное слово) при морфологическом анализе и, если слово имеет тег Name (имя), Surn (фамилия), Patr (отчество), Geox(топоним), Orgn (организация), Trad (торговая марка), то данное слово добавляется в список и возвращается пользователю. Демонстрация работы на рисунке 11.

Исправление опечаток работает с помощью обращения к API Яндекс.Спеллер, возвращается наиболее вероятное слово. По результатам тестирования можно сделать вывод, что работает хорошо, так как наиболее частые ошибки при компьютерном вводе - односимвольные (пропуск буквы, замена на букву, стоящую на клавиатуре рядом или повторное нажатие при залипании). Пример работы на рисунке 11.

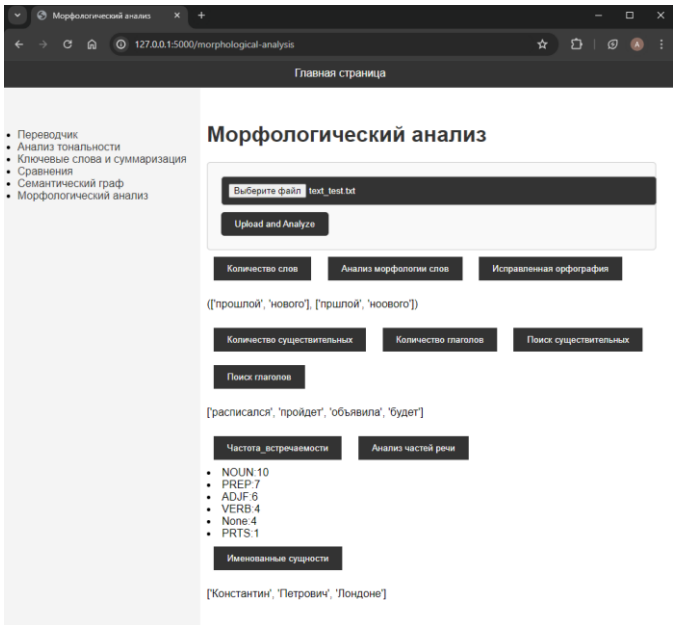

Рисунок 11 – Демонстрация работы исправления орфографии и NER

ЗАКЛЮЧЕНИЕ

В рамках ВКР была поставлена задача создать веб-приложение, которое осуществляет анализ текстовой информации с применением современных методов обработки естественного языка и современных технологий.

Поставленная задача решена в полном объеме. Созданное приложение имеет дружественный интерфейс и богатый функционал, который позволяет пользователю осуществлять анализ текстовой информации по различным параметрам.

В ходе работы были изучены методы анализа текста, необходимые библиотеки, алгоритмы, модели и языки программирования. Также были проанализированы конкуренты. После этого была был создан веб-ресурс, решающий поставленные задачи.

В дальнейшем планируется добавить дополнительную функциональность, чтение разных форматов документов, обеспечить безопасность сайта от внешних атак и загрузить программу на удаленный сервер для беспрерывной работы.

Можно заключить, что данный проект внес вклад в развитие инструментов для анализа текстовых данных и он может быть использован в научной и профессиональной деятельности.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ

1. Korobov M.: Morphological Analyzer and Generator for Russian and Ukrainian Languages // Analysis of Images, Social Networks and Texts, pp 320-332 (2015).
2. Погорелов Д. А. Сравнительный анализ алгоритмов редакционного расстояния Левенштейна и Дамерау-Левенштейна / Д. А. Погорелов, А. М. Таразанов, Л. Л. Волкова // Синергия Наук. – 2019. – № 31. – С. 803-811.
3. Митрофанова О. А. Эксперименты по автоматическому выделению ключевых выражений в стилистически разнородных корпусах русскоязычных текстов / О. А. Митрофанова, Д. А. Гаврилик // Terra Linguistica. – 2022. – Т. 13, №4. – С. 22-40.
4. Smetanin S. The Applications of Sentiment Analysis for Russian Language Texts: Current Challenges and Future Perspectives / S. Smetanin // IEEE Access – 2020. – vol. 8, pp. 110693-110719.
5. Бизли Д. Python. Книга рецептов / Д. Бизли, Б. К. Джонс / пер. с анг. Б. В. Уварова. – М.: ДМК Пресс, 2019. – 648 с.: ил.
6. Сервис для сокращения текста онлайн | Retext.AI – URL: https://retext.ai/ru/summarization (дата обращения: 20.05.2024).
7. Морфологический анализатор pymorphy2 – URL: https://pymorphy2.readthedocs.io/en/stable/ (дата обращения: 20.05.2024)
8. Автоматическая обработка текста – URL: http://www.aot.ru (дата обращения: 20.05.2024).

ПРИЛОЖЕНИЕ А

Исходный код ресурса

Исходный код веб-ресурса расположен на github (https://github.com/Alexksksss/VKR-NLP). Его можно посмотреть, воспользовавшись приведенном на рисунке А.1 QR-кодом.

Рисунок А.1 – QR-код для ознакомления с кодом веб-ресурса

АНТИПЛАГИАТ
ОБНАРУЖЕНИЕ ЗАИМСТВОВАНИЙ

СПРАВКА

Томский Государственный Университет

о результатах проверки текстового документа
на наличие заимствований

ПРОВЕРКА ВЫПОЛНЕНА В СИСТЕМЕ АНТИПЛАГИАТ.ВУЗ

Автор работы: Литвак Александра Ильинична
Самоцитирование рассчитано для: Литвак Александра Ильинична
Название работы: Литвак А.И., ВКР
Тип работы: Выпускная квалификационная работа
Подразделение: НИ ТГУ, Институт прикладной математики и компьютерных наук

РЕЗУЛЬТАТЫ

■ **ОТЧЕТ О ПРОВЕРКЕ КОРРЕКТИРОВАЛСЯ:** НИЖЕ ПРЕДСТАВЛЕНЫ РЕЗУЛЬТАТЫ ПРОВЕРКИ ДО КОРРЕКТИРОВКИ

СОВПАДЕНИЯ	14.17%	СОВПАДЕНИЯ	14.17%
ОРИГИНАЛЬНОСТЬ	85.83%	ОРИГИНАЛЬНОСТЬ	85.83%
ЦИТИРОВАНИЯ	0%	ЦИТИРОВАНИЯ	0%
САМОЦИТИРОВАНИЯ	0%	САМОЦИТИРОВАНИЯ	0%

ДАТА ПОСЛЕДНЕЙ ПРОВЕРКИ: 30.05.2024

ДАТА И ВРЕМЯ КОРРЕКТИРОВКИ: 30.05.2024 18:41

Структура документа: Проверенные разделы: основная часть с.4-29

Модули поиска: Переводные заимствования IEEE; Переводные заимствования издательства Wiley; Перефразирования по СПС ГАРАНТ: аналитика; Перефразирования по коллекции IEEE; Переводные заимствования (RuEn); Интернет Плюс*; Публикации eLIBRARY (переводы и перефразирования); СПС ГАРАНТ: нормативно-правовая документация; Переводные заимствования по коллекции Гарант: аналитика; Переводные заимствования по коллекции Интернет в русском сегменте; Перефразирования по Интернету; Перефразированные заимствования по коллекции Интернет в английском сегменте; Переводные заимствования по коллекции Интернет в английском сегменте; ИПС Адилет; Кольцо вузов; Перефразирования по коллекции издательства Wiley; Шаблонные фразы; Переводные заимствования по Интернету

С результатами ознакомлена
— [signature] —

Работу проверил: Пахомова Елена Григорьевна
ФИО проверяющего

Дата подписи: 30.05.2024 — [signature] —
Подпись проверяющего

Чтобы убедиться в подлинности справки, используйте QR-код, который содержит ссылку на отчет.

Ответ на вопрос, является ли обнаруженное заимствование корректным, система оставляет на усмотрение проверяющего. Предоставленная информация не подлежит использованию в коммерческих целях.