МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
Национальный исследовательский ядерный университет «МИФИ» (НИЯУ МИФИ)

Институт Интеллектуальных кибернетических систем

Кафедра кибернетики (№ 22)

Направление подготовки 01.03.02 Прикладная математика и информатика

Пояснительная записка

к ВКР бакалавра на тему:

Программная реализация алгоритмического обеспечения для решения задачи автоматического реферирования текстовой информации из открытых источников с помощью библиотеки DeepPavlov

Группа Б16-501

Студент _____ (подпись) Зайцева И.В. (ФИО)

Руководитель _____ (подпись) Киреев В.С. (ФИО)

Научный консультант _____ (подпись) _____ (ФИО)

Москва 2020

Реферат

Пояснительная записка содержит N_1 страниц, 25 рисунков, 9 таблиц, 2 приложений. Количество использованных источников - N_5.

Ключевые слова: автоматическое выделение ключевых слов, автоматическое аннотирование, TextRank, RAKE, логистическая регрессия, NLP,обработка естественного языка.

Целью данной работы является разработка web-приложения для автоматического определения ключевых слов и аннотирования текстовой информации из открытых источников для решения задачи информационного поиска.

В первом разделе проводится обзор основных методов и алгоритмов автоматического выделения ключевых слов в статьях и их автоматического аннотирования.

Во втором разделе описываются выбранные алгоритмы автоматического выделения ключевых слов в статьях и их автоматического аннотирования. Указываются этапы предобработки текста. Вводятся метрики для оценки качества работы алгоритмов.

В третьем разделе описано проектирование web-приложения, представлены макеты интерфейса.

В четвёртом разделе описывается процесс реализации web-приложения. Проводится тестирование разработанного web-приложения.

В конце пояснительной записки приводится заключение о выполненной работе.

В приложении 1 располагается полный набор скиншотов приложения.

В приложении 2 располагается полный набор тестов для приложения.

Содержание

Введение

В настоящий момент существует множество прикладных задач, решение которых связанно с анализом текстовых данных. К этим задачам относятся классификация, кластеризация, а также автоматическое извлечение ключевых слов, реферирование и другие. В частности, выделение ключевых слов в текстах может выступать в качестве первичной информации, представляя документ при решении задач информационного поиска и других задач.

Кроме этого, за последнее десятилетие социальные сети стали важным средством получения и распространения информации в различных областях, таких как бизнес, развлечения, наука и политика. Одной из причин популярности социальных сетей является возможность общения и обмена какой-либо информацией. Огромный рост использования социальных сетей привел к увеличению накопления данных, а также появлению в них большого количества тематических сообществ, в которых публикуются различные статьи и заметки. К этим текстам также можно применять алгоритмы выделения ключевых слов и реферирования, и полученные метаданные использовать в информационных системах.

Целью данной работы является разработка web-приложения для автоматического определения ключевых слов и аннотирования текстовой информации из открытых источников для решения задачи информационного поиска.

Теоретическая ценность данной работы заключается в исследовании нового алгоритма выделения ключевых слов текста.

Объектом исследования является автоматизация процесса выделения ключевых слов и аннотирование. Предмет исследования – способ обработки текстов для решения задач информационного поиска.

Актуальность данной работы обусловлена тем, что разработанное web-приложение позволит автоматизировать процесс аннотирования больших наборов текстов и определять ключевые слова в них, а также на основе полученных метаданных решать задачи информационного поиска.

В первом разделе данной работы проводится анализ предметной области, вводятся основные термины. Проводится обзор методов и основных алгоритмов для автоматического определения ключевых слов в статье и их аннотирования.

Во втором разделе приводиться описание выбранных алгоритмов определения ключевых слов в тексте и его автоматического аннотирования. Кроме этого, указываются метрики для оценки качества работы алгоритмов. Также описываются подходы, использующиеся для оценки качества автоматически сгенерированных аннотаций, проводиться сравнительный анализ алгоритмов.

В третьем разделе определяются функциональные и системные требования, предъявляемые к разрабатываемому web-приложению. Кроме этого, рассматривается вопрос проектирования web-приложение с учетом установленных требований. Также проектируется архитектура интерфейса взаимодействия web-приложения с пользователем.

В четвёртом разделе описывается процесс реализации web-приложения, который включает разработку интерфейса и реализацию внутренней составляющей, а именно процесса предобработки текста, применение выбранных алгоритмов для выделения ключевых слов и аннотирования подготовленных данных. Кроме этого, проводится тестирование приложения.

Раздел 1. Обзор существующих алгоритмов автоматического выделения ключевых слов и аннотирования

В данном разделе проводится анализ предметной области, вводятся основные термины, приводиться обзор различных программных разработках для решения задаси аннотирования и извлечения КС. Приводится классификация различных методов автоматического извлечения ключевых слов, дается краткое описание каждого из методов. Рассматриваются графовый и гибридный подходы, указываются их преимущества и недостатки. Кроме этого приводится классификация различных методов автоматического аннотирования текстов. Также рассматривается несколько алгоритмов, приводится их краткое описание, указываются их преимущества и недостатки. В конце раздела ставятся цели и задачи ВКР.

1.1. Анализ предметной области

Ключевое слово (КС) – последовательность из одного и более слов, которое в совокупности с другими КС дает сжатое представление о содержании текста. Набор КС обычно состоит из 5-15 слов.

Автоматизация процесса выделения ключевых слов может быть использована для расширения функциональности систем информационного поиска. В работе [1] описывается система Keyphind – это поисковая система для электронных библиотек, в основе работы которой лежит алгоритм KEA [2]. Автоматически излеченные ключевые фразы используются в качестве индексации и представления содержания. Content Analyzer лицензированный продукт предназначен для анализа содержания тематических Web-страниц в реальном времени на русском языке, используемый метод – TF-IDF [3]. Tesuck сервис автоматического реферирования и выделения ключевых слов, построенный на основе алгоритма TextRank. Представленный прикладной пакет позволяет осуществлять лингвистическую разметку неструктурированных текстов на русском языке с применением графовых моделей. Данная разметка выполняется путём построения графа слов с вычислением величины связи между ними и ранжированием вершин. Пакет реализован в виде масштабируемого Web-сервиса [4]. Также существует разработка Portal Min@s, которая включает в себя различные инструменты, в том числе и для выделения ключевых слов при помощи алгоритма LDA [5].

В таблице 1.1 приведены основные характеристики всех упомянутых выше разработок.

Также существуют различные аналоги данных систем. Недостатками всех этих систем является лицензированность, необходимость определенного программного

обеспечения, а также языкозависимость.

Таблица 1.1 – Характеристики систем для автоматического извлечения КС

Название	Метод	Язык	Платформа
Keyphind	KEA	английский	Java
Content Analyzer	TF-IDF	Русский	ПО Windows
Tesuck	TextRank	русский, английский	Веб-сервис
Portal Min@s	LDA	английский	Веб-сервис

Автоматическое аннотирование представляет собой процесс сокращения исходного текстового документа с использованием программного обеспечения с целью создания краткого содержания. Обычно это от 5 до 30% от исходного или же 5-10 предложений. Основной проблемой является сложность автоматического создания текстов на естественном языке, то есть построение грамотного и связного текста. Кроме того, большинство алгоритмов, построенных на машинном обучении и нейросетях, требуют больших вычислительных ресурсов.

В настоящее время текстовой информации в Интернете становиться все больше и больше и поэтому все более необходимы системы автоматического реферирования. Они позволяют получить краткое содержание, а также дать краткую характеристику первоисточника. Это позволяет уменьшить время на поиск необходимых статей.

Такая функция как “Автореферат” присутствует в программах Microsoft Word, Intelligent Text Miner, TextReferent, Oracle Context, Google News. Все эти разработки входят в самостоятельные лицензированные программы, поэтому их использование невозможно.

1.2. Методы автоматического выделения ключевых слов

Существуют различные методы извлечения КС:

- статистические;
- лингвистические;
- графовые;
- с использованием методов машинного обучения;
- нейросетевые;
- гибридные.

Статистические методы основываются на нелингвистических особенностях документов, такие как положение слова в тексте и относительных частотах встречаемости морфологических, лексических, синтаксических единиц и их комбинаций. Это делает создаваемые на их основе алгоритмы довольно простыми, но недостаточно точными. К ним относятся, например, TF-IDF(term frequency inverse document frequency) [6], word co-

occurrences[7] и PAT-tree [8].

В основе лингвистических методов лежит представление о тексте, как системе семантически и грамматически взаимосвязанных элементов-слов, каждые из которых обладают лингвистическими особенностями. Данный подход включает в себя лексический анализ, синтаксический анализ, учет регулярных синтаксических конструкций, содержащих на определенных позициях ключевые слова. В чистом виде такие методы слабо применимы к рассматриваемой задаче, но могут использоваться в сочетании с другими.

В графовых (граф-ориентированные) методах текст представляется множеством слов-вершин (или вершин-словосочетаний) и ребротношений между ними. Эти реброотношения отражают для каждой пары слов факт нахождение слов в окне некоторой ширины размера и семантическую близость. Для вершин полученного графа вычисляются меры центральности и по пороговому критерию отбираются ключевые слова. В различных методах отличаются способом определения значимости каждой вершины и вычисления отношений между ними.

Методы с использованием методов машинного обучения требует размеченных данных. Преимуществом таких методов является в том, что они хорошо работают на научных статьях [9].Один из наиболее известных алгоритмов в этом подходе является алгоритм извлечения (KEA).

Нейросетевые методы – это подход с использованием многослойных нейронных сетей к обобщению и выделению скрытых зависимостей между входными и выходными данными. Данный подход требует размеченных данных для обучения алгоритмов. Например, в работе [10] используется сверточная многослойная нейросеть (CNN) для выделения КС в текстах определенной тематики, а в работе [11] применяет рекуррентная нейросеть (RNN) для обработки твитов (записи в социальной сети Twitter).

В данной работе будут рассмотрены алгоритмы TextRank, RAKE и гибридный подход.

Алгоритм TextRank был разработан на основе известного алгоритма ранжирования веб-страниц PageRank[12]. Алгоритм заключается в следующем, происходит подсчет весов вершин (слово или словосочетание), две вершины смежны, если слова появляются внутри окна заданной ширины. Также перед определением слов-кандидатов может использоваться фильтрация. Далее веса вершин ранжируются и первые 10 - 15 слов или словосочетания становятся КС. Преимуществами данного алгоритма является языконезависимость, не требуется обучение, недостатком – высокие вычислительные затраты по сравнению с RAKE[13].

В алгоритме RAKE сначала формируется список потенциальных КС с помощью

заданного словаря разделителей фраз, а затем строится граф, вершинами которого являются отдельные слова. Значимость для слова определяется набором показателей: частота появления вершины, степень вершины, отношение степени к частоте. Значимость потенциальной ключевой фразы рассчитывается как сумма значимостей каждого входящего в него слова. В качестве КС для данного текста отбирается первая треть упорядоченного по убыванию значимости списка вершин. Преимуществами данного алгоритма являются языконезависимость, более быстрая работа по сравнению с TextRank, недостатком – составные ключевые слова получают больший вес, что может приводить к большим неточностям.

Также будет рассмотрен гибридный подход, в котором на первом этапе будет применен алгоритм, позволяющий классифицировать предложения на “содержащие КС” и “не содержащие КС”. Далее с помощью частотного анализа в предложениях, которые будут помечены как “содержащие КС” будет определен топ 10-15 наиболее часто встречающихся слов и словосочетаний. В качестве алгоритма классификации могут быть использованы методы машинного обучения и нейронные сети. Недостатком данного подхода является необходимость в обучающих данных, в предварительном обучении, а также алгоритм, обучаясь на одной выборке, вероятно, будет хуже работать на данных с другой структурой.

После проведения сравнительного анализа алгоритмов для выделения КС был выбран алгоритм Rake и гибридный подход.

1.3. Методы автоматического реферирования текстов

Существующие подходы автоматического аннотирования можно разделить на:

- экстрактивные и генерирующие методы;
- методы с обучением и без обучения;
- NLP и база знаний;
- глубокое обучение и алгоритмы.

Экстрактивные методы предполагают выделение из текста наиболее характерных предложений, которые в совокупности отражают его содержание. Для оценивания значимости фрагментов в работе [14] предлагается следующая модель линейных весов:

$$weight(F) = Location(F) + CuePrase(F) + StatTerm(F) + AddTerm(F) \quad (1.1)$$

$Location(F)$ определяется местонахождением фрагмента в тексте;

$CuePrase(F)$ определяется тем, встречаются ли в блоке лексические или фразовые резюмирующие конструкции;

$StatTerm(F)$ - статистическая важность;

$AddTerm(F)$ характеризует наличие в данном блоке терминов, которые также встречаются в заголовке, в колонтитуле, первом параграфе и тд.

При данном подходе фрагменты не обрабатываются, а извлекаются в том порядке и виде, в каком они приведены в тексте.

Более сложный подход – это генерирующие методы, или краткое изложение содержания, которые заключается в создании семантического представления исходного документа и последствии генерации нового текста. Такое преобразование в вычислительном отношении намного сложнее, чем извлечение, и включает как обработку естественного языка, так и зачастую глубокое понимание предметной области исходного текста в тех случаях, когда исходный документ относится к специальной области знаний.

Также методы автоматического реферирования можно разделить на методы с обучением и без обучения. Различия состоит в следующем, для методов с обучением необходимы предварительно размеченные данные для дальнейшего предварительного обучения системы, а для методов без обучения – нет. В работе [15] используется алгоритм без учителя, позволяющий генерировать аннотацию.

Natural language processing (NLP), или обработка естественного языка, заключается в анализе текста на основе грамматических правил, языковых моделей, деревьев разбора и других методов анализа и описания языковых структур. В методах с глубоким обучением используются знания в области NLP для корректного построения предложений, например, для объединения предложений в одно.

Подход к использованию базы знаний подразумевает под собой анализ предметной области и состоит из следующих основных компонентов: правил анализа полного документа, стоп-лексикона, информационно-концептуальной сети в виде корневого дерева и множества шаблонов для извлечения информации и правил генерации текста реферата[16].Правила анализа полного документа описывают порядок выполнения обработки документа. Стоп-лексикон используется для удаления из полного текста статьи нерелевантной для реферата информации с целью облегчения дальнейшего анализа. Информационно-концептуальная сеть используется для семантического анализа – выделения лексических маркеров в тексте документа.

Глубокое обучение подразумевает использование нейронных сетей для автоматического реферирования текстов. Используются модели на основе RNN, CNN, а также LSTM и другие. Алгоритмический подход предполагает использование различных статистических, графовых и других алгоритмов.

Рассмотрим алгоритмы экстрактивного подхода:

- базовый алгоритм;
- алгоритм Freq;
- TextRank.

В работе [17] рассмотрены два алгоритма, базовый и Freq.

Базовый алгоритм. В данном алгоритме выбираются фрагменты фиксированной длины с наибольшим весом. Требуемая длина выбирается из того условия, что суммарное количество символов в аннотации составляет около 300 штук. Вес фрагмента документа определяется следующим образом:

$$W_f = \sum w_i + K\frac{n}{L} \quad (1.2)$$

где $\sum w_i$– сумма весов слов запроса, вошедших во фрагмент;

K – некоторая константа;

n – число слов названия статьи, вошедших во фрагмент;

L – расстояние между первым и последним словом названия, встретившиеся во фрагменте.

Вес каждого слова зависит от распределения слова в коллекции и тем выше, чем более редкое это слово. Он рассчитывается по формуле:

$$w_i = \log_2 N_i / \log_2 N \quad (1.3)$$

где N_i– число документов коллекции, где встретилось данное слово;

N – общее число документов в коллекции.

Таким образом, формула (1.2) оценивает фрагменты, в которых слова названия располагаются более тесно, встречаются больше слов названия. Значительнее оцениваются те слова названия, которые реже встречаются во всей коллекции документов. То есть аннотация состоит из фрагментов текста с наибольшим весом. Если несколько фрагментов получают одинаковые веса, то выдается тот, который находится ближе к началу текста. Данный алгоритма имеет следующий недостаток - для оценки весов слов во фрагменте необходима коллекция документов, а также он не может показать высокого качества формирования аннотаций, если запрос содержит всего одно слово или часто встречающееся словосочетание.

Алгоритм Freq. Данный алгоритм учитывает частоту слов в окне длиной в n слов вокруг анализируемого фрагмента, то есть фрагмент находится в середине окна. Используется следующая формула для вычисления веса:

$$W_{freq} = W_b + \sum \log_2 F_k \quad (1.4)$$

где W_b – вес, вычисленный по базовому алгоритму;

F_k – количество раз, которое слово встречается в окне в n слов, включающий фрагмент.

Таким образом, наибольший вес получают те фрагменты, которые кроме того, что содержат и наибольшее число слов названия статьи, но и большее количество слов часто встречающихся в документе.

Недостатками данного алгоритма является то, что, во-первых, частота не несет информации о распределении слова в документе – распределено ли оно равномерно по всему документу или только в некоторых фрагментах, во-вторых, вычисление частот для фрагментов документов может требовать относительно много ресурсов.

Алгоритм TextRank. В основе данного алгоритма лежит использование алгоритма ссылочного ранжирования PageRank. Текст представляется в виде взвешенного ориентированного графа, вершинами которого являются предложения. Вес ребер между вершинами определяется степенью похожести предложений друг на друга. Данный алгоритм является языконезависимым, относительно прост в программной реализации и проводит аннотирование отдельной статьи, то есть нет необходимости в коллекции документов.

После проведения сравнительного анализа алгоритмов для выделения аннотации был выбран алгоритм TextRank.

1.4. Постановка задачи и вывод

Целью данной работы является разработка web-приложения для автоматического определения ключевых слов и аннотирования текстовой информации из открытых источников для решения задачи информационного поиска.

Для достижения поставленной цели необходимо выполнить следующие задачи:

1. провести анализ предметной области;
2. изучить алгоритмы автоматического выделения ключевых слов;
3. изучить алгоритмы автоматического аннотирования;
4. определить способы предобработки;
5. определить метрики оценка качества;
6. изучить алгоритм ближайшего соседа
7. разработать требования к приложению;
8. спроектировать web-приложения;
9. разработать интерфейс;
10. реализовать web-приложения;
11. провести тестирование web-приложения;
12. оценить качество работы приложения.

Выводы:

- Автоматическое извлечение ключевых слов является важным направлением исследований в области анализа текста, обработки естественного языка и поиска информации. Компактное представление документов может быть использовано для автоматического индексирования, суммирования, классификации, кластеризации и

фильтрации.

- Был проведен анализ предметной области, введены основные термины.
- Был произведен обзор основных методов и алгоритмов автоматического выделения ключевых слов и их автоматического аннотирования. В результате сравнения для выделения КС были выбраны алгоритмы RAKE и гибридный подход, для аннотирования – TextRank.

Раздел 2. Алгоритмы, используемые в работе

В данном разделе приведено описание выбранные алгоритмы выделения ключевых слов в тексте и его автоматического аннотирования. Вводятся основные понятия, указаны расчетные формулы. В описание приведены основные этапы этого алгоритма, указываются потенциальные сложности, которые могут возникнуть при реализации каждого этапа, для каждого алгоритма приводятся блок-схемы. Также указываются метрики для оценки качества работы алгоритмов. Приводятся расчетные формулы для метрик precision, recall, F-мера, отвечающие за оценку точности выделения КС. Также описывается приводиться матрика ROUGE для опрделения качества работы алгоритма для выдения аннотации. Описываются этапы предобработки текста: нормализация, стемминг, лемматизация, метод Bag of Words.

2.1. Описание выбранного алгоритма определения ключевых слов в тексте

Алгоритм RAKE основан на идее, что ключевые слова не содержат знаки препинания и стоп-слова, такие как предлоги, междометья и другие. Он применяется на одиночных документах. Первый этап заключается в выявление слов кандидатов. Текст разбивается на массив слов с помощью слов разделителей. Далее подсчитывается сочетания потенциальных ключевых слов и высчитывается частота слова и степень слова и отношение степени к частоте. Частота потенциального числа deg(w) это количество раз, сколько данное слово встречается как самостоятельное КС. Степень потенциального КС freq(w) определяется как отношение числа встречаемости этого слова как самостоятельного КС к общему числу встречаемости данного слова в составе потенциальных КС. Чтобы получить ранжированный список фраз проводится подсчет всех отношений deg(w)/freq(w), выбирается треть КС наиболее значимых.

Рассмотрим пример, пусть задана статья:

Compatibility of systems of linear constraints over the set of natural numbers Criteria of compatibility of a system of linear Diophantine equations, strict inequations, and nonstrict inequations are considered. Upper bounds for components of a minimal set of solutions and algorithms of construction of minimal generating sets of solutions for all types of systems are given. These criteria and the corresponding algorithms for constructing a minimal supporting set of solutions can be used in solving all the considered types of systems and systems of mixed types.

Далее определяем список потенциальных ключевых слов:

Compatibility – systems – linear constraints – set – natural numbers – Criteria – compatibility – system – linear Diophantine equations – strict inequations – nonstrict inequations – Upper bounds – components – minimal set – solutions – algorithms – minimal generating sets – solutions – systems – criteria – corresponding algorithms – constructing – minimal supporting set

– solving – systems – systems

Далее подсчитываем deg(w), freq(w) и отношение deg(w)/freq(w)

Таблица 2.1 – Результаты подсчетов для примера работы алгоритма RAKE

	algorithms	bounds	compatibility	components	constraints	Constructing	corresponding	criteria	dophanti	equation	generating	inequations	linear	minimal	naturel	nonnstrict	numbers	set	sets	solving	strict	supporting	system	systems	upper
deg(w)	3	2	2	1	2	1	2	2	3	3	3	4	5	8	2	2	2	6	3	1	2	3	1	4	2
freq(w)	2	1	2	1	1	1	1	2	1	1	1	2	2	3	1	1	1	3	1	1	1	1	1	4	1
$\frac{deg(w)}{freq(w)}$	$\frac{3}{2}$	2	1	1	2	1	2	1	3	3	3	2	2.5	2.7	2	2	2	2	3	1	2	3	1	1	2

minimal generating sets (8.7), linear diophantine equations (8.5), minimal supporting set (7.7), minimal set (4.7), linear constraints (4.5), natural numbers (4), strict inequations (4), nonstrict inequations (4), upper bounds (4), corresponding algorithms (3.5), set (2), algorithms (1.5), compatibility (1), systems (1), criteria (1), system (1), components (1),constructing (1), solving (1)

Если отобрать треть слов, тогда получим, что ключевые слова это *minimal generating sets (8.7), linear diophantine equations (8.5), minimal supporting set (7.7), minimal set (4.7), linear constraints (4.5), natural numbers (4).*

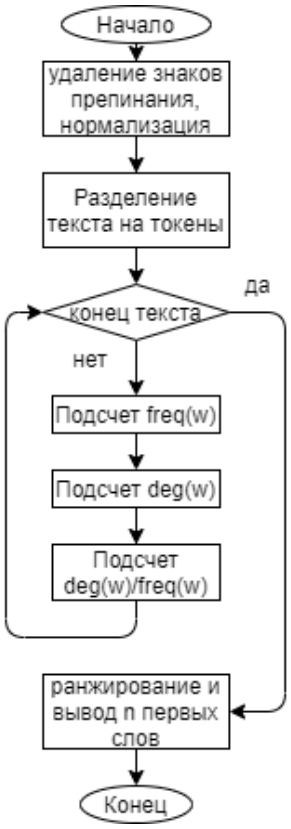

Рис. 2.1 – Блок-схема алгоритма RAKE

Второй подход - это гибридный метод, который совмещает в себе использование нейросетей или методов машинного обучения и частотный анализ. Первый этап заключается в предобработке текста, необходимо провести нормализацию и векторизацию обучающей и тестовой выборок, которые представляют совокупность предложений из статей и метку класса. В данном случае, нейросети или методы машинного обучения необходимы в качестве классификатора, который помечает каждое предложение как ”содержит КС” и “не содержит КС”. Далее проводиться частотный анализ, в предложениях помеченных как “содержат КС” подсчитывается частота встречаемости униграмм и словосочетаний, полученный список ранжируется и выбирается n первых ключевых фраз.

В данной работе проводиться сравнительный анализ различных классификаторов, с целью выявления наиболее точного, рассматривается логистическая регресиия и сверточная нейросеть.

Логистическая регрессия – это алгоритм машинного обучения, используемый для задач классификации и позволяющий оценивать апостериорные вероятности принадлежности объектам классам[18].

Пусть объекты описываются n числовыми признаками $f_j: X \to \mathbb{R},\ j = \overline{1,n}$. Тогда пространство признаковых описаний объектов есть $X = \mathbb{R}^n$. Пусть Y – конечное множество номеров (имен, меток) классов.

Пусть задана обучающая выборка пар “объект, ответ” $X^m = \{(x_1, y_1), \dots, (x_m, y_m)\}$.Для решения поставленной задачи рассмотрим бинарную классификацию.

Положим $Y = \{0,1\}$, где 0 – предложение не содержит КС, 1 – содержит. В логистической регрессии строится линейный алгоритм классификации $\alpha = X \to Y$ вида

$$\alpha(x, \omega) = sign\left(\sum_{j=1}^{n} \omega_j f_j(x) - \omega_0\right) = sign\langle x, \omega\rangle, \qquad (2.1)$$

где ω_j- вес j-го признака, ω_0- порог принятия решения, $\omega = (\omega_0, \omega_1, \dots, \omega_n)$-вектор весов, $\langle x, \omega\rangle$ – скалярное произведение признакового описания объекта на вектор весов.

Задача обучения линейного классификатора заключается в том, чтобы по выборке X^m определить вектор весов ω. Для этого решаеся задача минимизации эмпирического риска с функцией потерь вида:

$$Loss\ function: F(\omega) = \sum_{i=1}^{m} \log\left(1 + \exp\left(-y_i\right)\langle x, \omega\rangle\right) \to min_{\omega} \qquad (2.2)$$

Регуляризация — это метод добавления некоторых дополнительных ограничений к условию с целью решить некорректно поставленную задачу или предотвратить переобучение. В данном случае используется L2 регуляризация

$$J(X, \vec{y}, \vec{\omega}) = F(\omega) + \lambda|\vec{\omega}|^2 \qquad (2.3)$$

Базовый алгоритм оптимизации весов – это метод градиентного спуска. Это

алгоритм итерационной оптимизации для нахождения локального минимум дифференцируемой функции с помощью движения вдоль градиента. Для нахождения локального минимума функции потерь значение весов пересчитываются по следующей формуле:

$$\vec{\omega} = \vec{\omega} - \eta \nabla_{\vec{\omega}} \mathrm{L}(\vec{\omega}) \quad (2.4)$$

где $\boldsymbol{\eta}$ – скорость обучения, $\nabla_{\vec{\omega}} \mathrm{L}(\vec{\omega})$ – градиент функции потерь по $\vec{\omega}$.

Важным предположением в задачах МО является то, что градиент является аддитивной функцией относительно обучающих примеров:

$$\nabla_{\vec{\omega}} L(\vec{\omega}) = \sum_{i=1}^{n} \nabla_{\vec{\omega}} L_i(\vec{\omega}) \quad (2.5)$$

Тогда получаем, что решением задачи оптимизации будет

$$\hat{\omega} = \underset{\vec{\omega}}{\operatorname{argmin}} J(X, \vec{y}, \vec{\omega}) = \underset{\vec{\omega}}{\operatorname{argmin}} (C \sum_{i=1}^{m} \log(1 + \exp(-y_i) \langle x, \omega \rangle) + |\vec{\omega}|^2)$$

где C = $\frac{1}{\lambda}$ - обратный коэффициент регуляризации

Далее возможно определить принадлежность к классу, а также оценить апостериорные вероятности принадлежности к классам:

$$P(y|x) = \sigma(y\langle x, \omega \rangle), \quad y \epsilon Y, \quad (2.6)$$

где $\sigma(z) = \frac{1}{1+e^{-z}}$ – сигмоидная функция, монотонно возрастающая всюду дифференцируемая S-образная нелинейная функция. Выбор именно этой функции $f(x)$ можно обосновать, рассматривая логистическую регрессию как обобщенную линейную модель в предположении, что зависимая переменная y распределена по закону Бернулли. Одним из свойств является то, что для любого входного значения она сопоставляет его с выходным значением от 0 до 1.

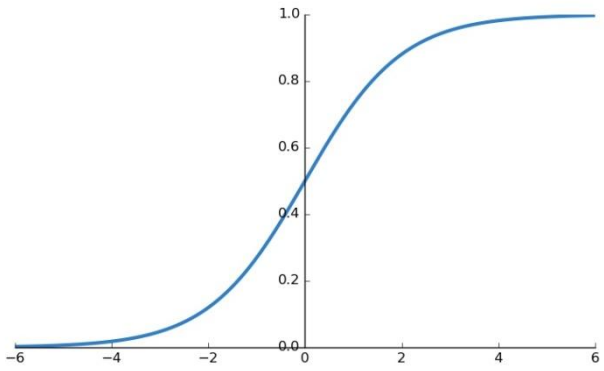

Рис. 2.2– Сигмоидная функция

Нейронные сети (NN) состоят из искусственных нейронов. Каждый нейрон имеет один или несколько входов и соответствующие входам весам. Веса - это числовые

характеристики получаемые во время обучения. Эти веса показывают, насколько сильно один нейрон воздействует на другие нейроны в данном слое. Входное значение и веса проходят через функцию активации и выдают результаты. Функция активации - это нелинейная функция, которая устанавливает порог для определения важности каждого нейрона.

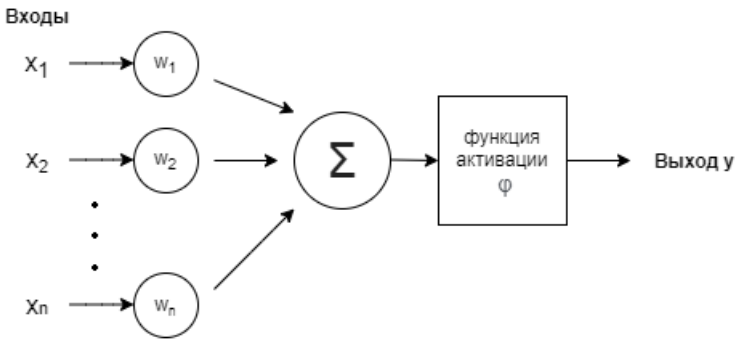

Рис. 2.3 –Однослойная нейросеть

Архитектура нейронных сетей содержит 3 различных слоя: входной слой, скрытый слой и выходной слой[19]. Входной слой представляет собой совокупность нескольких искусственных нейронов, которые принимают входные значения. Скрытые слои обрабатывают входные значения, обнаруживая отношения между ними, в одной нейросети может быть несколько скрытых слоев.

Сверточная нейронная сеть (CNN) – специальная архитектура искусственных нейронных сетей, отличительной чертой которой, является наличие операции свертки, умножение фрагментов векторизованного представления текста на матрицу свертки (ядро свертки) и суммирование полученного результата и его запись на ту же позицию. Весовые коэффициенты ядра свёртки неизвестны и устанавливаются в процессе обучения.

На Рис. 2.4 показан пример сверточного слоя с ядром свертки размера 3 × 3.

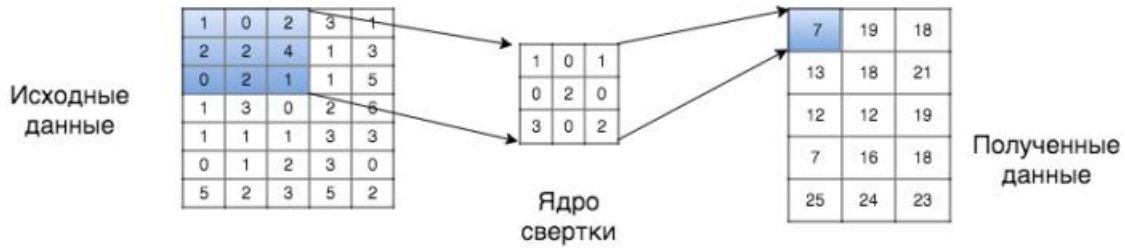

Рис. 2.4 –Сверточный слой

Субдискретизирующий слой – это слой позволяющий уменьшать размерность. Существуют различные способы, один из них метод выбора максимального элемента (max-pooling) — вся карта признаков разделяется на ячейки , из которых выбираются максимальные по значению. На рис.2.5 показан пример данного слоя с методом max-

pooling.

Рис. 2.5– Субдискретизирующий слой

Dropout слой (dropout регуляризация) необходим для предотвращения переобучением в нейронных сетях, обучение которых обычно производят стохастическим градиентным спуском, случайно выбирая некоторые объекты из выборки. Dropout регуляризация заключается в изменение структуры сети: каждый нейрон выбрасывается с некоторой вероятностью p. По такой прореженной сети производится обучение, для оставшихся весов делается градиентный шаг, после чего все выброшенные нейроны возвращаются в нейросеть. Таким образом, на каждом шаге стохастического градиента мы настраиваем одну из возможных 2^N архитектур сети, где под архитектурой понимается структура связей между нейронами, а через N обозначаем суммарное число нейронов. При тестировании нейросети нейроны уже не выбрасываются, но выход каждого нейрона умножается на $(1 - p)$ — благодаря этому на выходе нейрона мы будем получать математическое ожидание его ответа по всем 2^N архитектурам. Таким образом, обученную с помощью dropout-регуляризации нейросеть можно рассматривать как результат усреднения 2^N сетей.

Далее из предложений классифицированных как “содержат КС”, выделяются словосочетания и проводиться частотный анализ по всему тексту. И таким образом, получаем, что топ первых n словосочетаний являются ключевыми словами.

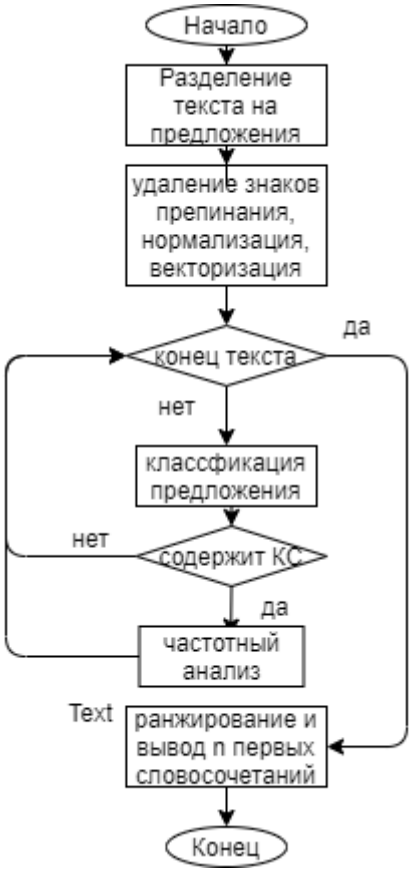

Рис. 2.6 –Блок-схема гибридного алгоритма

2.2. Описание выбранного алгоритма автоматического реферирования текстов

Выбранный алгоритм аннотирования заключается в следующем, сначала текст разбивается на предложения. Все предложения представляются в виде вектора. Далее строится граф, где вершины графа - это предложения в тексте. “Важность” вершины определяется с помощью алгоритма ссылочного ранжирования TextRank и зависит от количества ребер, которые идут из других вершин и “рейтинга “ этих ребер[20]. Также в алгоритме TextRank необходимо учитывать вес ребер, который определяется связью между двумя предложениями. Таким образом, составляется взвешенный ориентированный граф $G = (V, E)$, где V-вершины графа, E-ребра графа и $E = V \times V$. Тогда важность вершины V_iопределяется следующим образом:

$$S(V_i) = \sum_{Vj \in In(V_i)} \frac{\omega_{ji}}{\sum_{Vk \in Out(V_i)} \omega_{jk}} S(V_j) \qquad (2.7)$$

Существуют различные метрики, позволяющие оценить связь между двумя вершинами V_i и V_j, то есть вес ребра между ними ω_{ij}:

- коэффициентом Сёренсена - отношение количества одинаковых слов в предложениях к суммарной длине предложений.

$$Similarity(S_i, S_j) = \frac{|\{A_{k_{A_k \in S_i}} \& B_{k_{B_k \in S_j}}\}|}{|S_i| + |S_j|} \qquad (2.8)$$

- косинусное расстояние

$$Similarity(S_i, S_j) = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}} \quad (2.9)$$

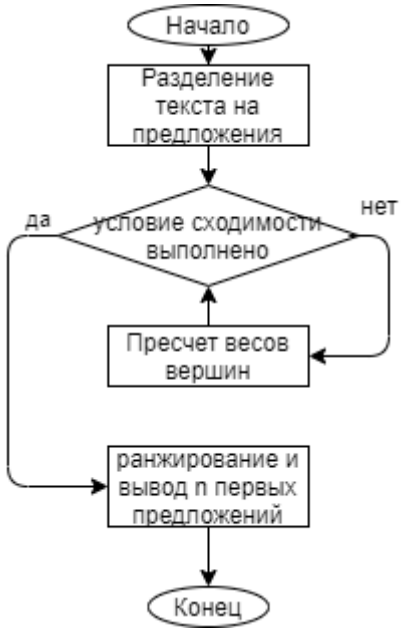

Рис. 2.7 –Блок-схема алгоритма TextRank

2.3. Метрики для оценки качества алгоритмов

Для оценки качества извлечения ключевых слов из текста используются следующие метрики: точность(precision), полнота(recall), F-мера.

Точность – это доля всех автоматически выделенных ключевых слов, являющихся ключевыми, относительно всех автоматически выделенных ключевых слов. Полнота системы – это доля выделенных из статьи слов, определенных как ключевые относительно всех слов, действительно характеризующих данную статью. F-мера представляет собой гармоническое среднее между точностью и полнотой. Она стремится к нулю, если точность или полнота стремится к нулю[21].

$$Precision = \frac{TP}{TP + FP} \quad (2.10)$$

$$Recall = \frac{TP}{TP + FN} \quad (2.11)$$

$$Precision = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (2.12)$$

где TP – истинно-положительное решение;

TN – истинно-отрицательно решение;

FP – ложно-положительное решение;

TN – ложно-отрицательно решение.

Формула 2.12 придает одинаковый вес точности и полноте, поэтому F-мера будет уменьшаться одинаково при уменьшении и точности и полноты.

Таблица 2.2 – Таблица контингентности для i-класса

Категория i		Экспертная оценка	
		Положительная	Отрицательная
Оценка системы	Положительная	TP	FP
	Отрицательная	FN	TN

Для оценки качества аннотации, составленной автоматически, применяется ROUGE(Recall-Oriented Understudy for Gisting Evaluation)[22].

Формула имеет следующий общий вид:

$$ROUGE - N_{precision} = \frac{\sum_{S \in S_H} \sum_{g_n \in S} Count\_m(g_n)}{\sum_{S \in S_H} \sum_{g_n \in S} Count(g_n)} \quad (2.13)$$

$$ROUGE - N_{recall} = \frac{\sum_{S \in S_H} \sum_{g_n \in S} Count\_m(g_n)}{\sum_{S \in S_H} \sum_{g_n \in S} Count(g_n)} \quad (2.14)$$

$$ROUGE - N_{F-measure} = 2 * \frac{ROUGE-N_{precision} * ROUGE-N_{reacll}}{ROUGE-N_{precision} + ROUGE-N_{reacll}} \quad (2.15)$$

где g_n-N-gram;

S_H-набор аннотаций, составленных вручную (в данной работе это одна аннотация, составленная автором статьи);

$Count\_m(g_n)$- число n-gram, содержащихся одновременно в аннотации, созданной вручную, и автоматически созданной аннотацией;

$Count(g_n)$- число n-gram, содержащихся в автоматически созданной аннотации.

В данной работе используются n-gram длиной 2(биграммы). Биграмма - это последовательность из двух слов.

2.4. Алгоритм ближайшего соседа

Для поиска похожих статей по ключевым словам будет использован алгоритм kNN – это алгоритм классификации, основанный на оценивании сходства векторов. Для решения поставленной задачи алгоритм адаптирован так, чтобы для некоторого набора ключевых слов из заданного выборки, состоящей также из ключевых слов некоторых статей, подбирались наиболее схожие объекты. Заранее необходимо каждый набор ключевых слов нормализовать и перевести в вектор методом “bag of words”.

Пусть $X \epsilon \mathbb{R}^n$ – множество векторов, над множеством объектов задается евклидова функция расстояния $\rho(x_1, x_2)$:

$$\rho(x_1, x_2) = \sum_{i=1}^{n} (x_1 - x_2)^2 \quad (2.16)$$

Для произвольного объекта x расположим объекты исходной выборки $\{x\}_{i=1}^{m}$ в порядке возрастания расстояния до x.

$$\rho(x, x_{1,x}) \leq \rho(x, x_{2,x}) \leq \cdots \leq \rho(x, x_{m,x}) \quad (2.17)$$

Где через $x_{i,x}$ обозначается тот объект исходной выборки, который является i-м соседом объекта x.

На вход алгоритму необходимо подать исходную выборку наборов ключевых слов, представленных в виде векторов, а также желаемое количество похожих наборов.

К достоинствам относиться простота реализации, к недостаткам неэффективный расход памяти и даны алгоритм предполагает сравнение некоторого объекта со всеми объектами выборки, что требует линейного по длине выборки числа операций.

2.5. Этапы обработки текстов

Для работы с текстом необходимо провести нормализацию. Она включает в себя следующие этапы:

- удаление всех символов, кроме букв верхнего и нижнего регистра, а также цифр;
- преобразование слова к нижнему регистру;
- токенизация;
- удаление стоп слов;
- лемматизация/стемминг.

Токенизация – это разбиение текста на более мелкие части, токены.

Лемматизация – процесс приведения слова к начальной форме.

Стемминг – процесс приведения слова к основе (отрезание окончания и формообразующего суффикса)[23].

Предобработка текста для алгоритма автоматического аннотирования включает в себя деление текст на токены, в данном случае это предложения. Для оценки близости двух токенов необходимо провести предобработку предложений, используя все вышеперечисленные этапы.

Для работы алгоритма RAKE в список стоп-слов помещаются все слова, не являющиеся существительными и прилагательными. Кроме этого, используется “bag of words”. Это модель текста, в которой текст представляет собой неупорядоченный набор слов, входящих в обрабатываемый текст. Таким образом, каждый документ представляет собой некоторый вектор, где значение координаты является частота встречаемости некоторого слова. Нормализация способствует увеличению полноты поиска и одновременно снижению его точности. Когда слова приводятся к единой форме, системе проще распознать их сходство, благодаря чему находится больше кандидатов и полнота возрастает. Однако это способствует и тому, что при упрощении слово может изменить изначально заложенный смысл, из-за чего снижается показатель точности информационного поиска.

Для гибридного подхода необходимо разделить текст на предложения и провести нормализцию. Далее предобработанный текст представляется в виде вектора. Каждая координата вектора отвечает за конкретное слово и рассчитывается по TF-IDF.

TF-IDF — статистическая мера, используемая для оценки важности слова в контексте документа, являющегося частью коллекции документов или корпуса.

TF — отношение числа вхождений некоторого слова к общему числу слов документа. Таким образом, оценивается важность слова в пределах отдельного документа.

$$tf(t,d) = \frac{n_t}{\sum_k n_k} \qquad (2.18)$$

где n_t есть число вхождений слова t в документ, а в знаменателе – общее число слов в данном документе.

IDF — инверсия частоты, с которой некоторое слово встречается в документах коллекции. Учёт IDF уменьшает вес широкоупотребительных слов. Для каждого уникального слова в пределах конкретной коллекции документов существует только одно значение IDF.

$$idf(t,D) = log\frac{|D|}{|\{d_i \in D \mid t \in d_i\}|} \qquad (2.19)$$

Где |D| - число документов в коллекции,

$|\{d_i \in D \mid t \in d_i\}|$ - число документов из коллекции D, в которых встречается t (когда $n_t \neq 0$).

Выбор основания логарифма в формуле не имеет значения, поскольку изменение основания приводит к изменению веса каждого слова на постоянный множитель, что не влияет на соотношение весов.

Таким образом, мера TF-IDF является произведением двух сомножителей:

$$id - idf(t,d,D) = tf(t,d) * idf(t,D) \qquad (2.20)$$

2.6. Выводы

- Приведено описание выбранных алгоритмов выделения КС и приведены метрики precision, recall, F-мера для оценки качества работы алгоритма.
- Описан метод для аннотирования текста, приведена метрика ROUGE для оценки качества работы алгоритма.
- Рассмотрен алгоритм поиска похожих текстов.
- Описаны этапы предобработки текстов.

Раздел 3. Проектирование web-приложения

В данном разделе определяются функциональные и системные требования, предъявляемые к разрабатываемому web-приложению. Данные требования касаются различных аспектов, таких как web-дизайн, функционал, структура разрабатываемого приложения. Кроме этого рассматривается вопрос проектирования web-приложение с учетом установленных требований, приводятся физическая схема данных, диаграмма компонентов, диаграмма последовательности и диаграмма активности. Также проектируется архитектура интерфейса взаимодействия web-приложения с пользователем, с целью создания полностью ориентированного на пользователя приложения, разрабатываются макеты интерфейса. Спроектированный интерфейс должен обеспечить выполнение поставленной задачи. Результаты проектирования оформлены с помощью языка UML.

3.1. Функциональные и системные требования к приложению

К разрабатываемому приложению предъявляются функциональные и системные требования.

Системные требования — это описание характеристик, которым должен соответствовать компьютер для того, чтобы на нём могло использоваться данное ПО.

Таблица 3.1 – Системные требования

Языка разработки	Python 3.7
ОС	Windows 8 Ubuntu 16.04 MacOS Cataline
Браузер	Chrome 29+ Safari 9+ Firefox 28+

Функциональные требования описывают предполагаемое поведение системы, определяя действия, которые система способна выполнять.

1. Система должна предоставлять возможность авторизации и работы с личным кабинетом
2. Система должна осуществлять поиск по словам в режимах по “ключевым словам”, ” по названию” и “по автору”
3. Система должна производить поиск в общей и личной библиотеке
4. Система должна предоставлять возможность просмотра личной библиотеки
5. Система должна предоставлять возможность загрузки статей в личную

библиотеку путем загрузки файла формата csv

6. Система должна поддерживать функцию аннотирования и выделения ключевых слов одиночной статьи
7. Система должна поддерживать пакетный режим обработки, подразумевающий од собой загрузку файла формата csv и аннотирования и выделения ключевых слов, содержащихся в данном файле текстов и сохранять ее на ПК
8. Система должна предоставлять возможность загрузки в личную библиотеку записей из групп социальной сети “Вконтакте”
9. Система должна предоставлять возможность просмотра списка добавленных групп, их обновления и удаления с подтверждением действия
10. Система должна поддерживать поиск в личной библиотеке записей общий поиск и поиск в конкретной группе
11. Система должна предоставлять возможность ознакомления с полным функционалом с помощью информации на главной странице

3.2. Проектирование web-приложения

В данной работе для хранения данных используется Google BigQuery. Google BigQuery – это реляционная СУБД и полностью управляемое, безсерверное хранилище данных, которое обеспечивает масштабируемый, экономически эффективный и быстрый анализ данных. Ниже на Рис. 3.1 представлена физическая схема данных.

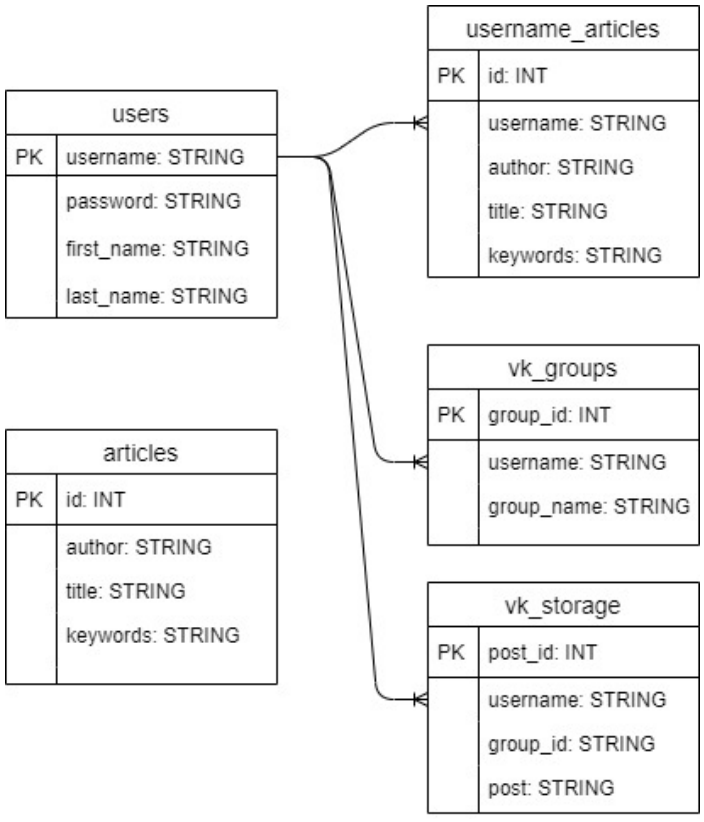

Рис. 3.1 – Физическая схема данных

Таблица users отвечает за хранение данных об авторизированных пользователях, их логины, пароли и имена. Таблица articles содержит набор метаданных о статьях с сайта rsl.ru, авторы, названия и ключевые слова. Таблица username_artciles содержит метаданные о статьях, загруженных пользователем. Таблица vk_groups содержит информацию о добавленных пользователями группах. Таблица vk_storage содержит скаченные посты групп пользователей.

Для моделирования взаимодействия объектов в языке UML используются соответствующие диаграммы взаимодействия. Взаимодействия объектов можно рассматривать во времени, и тогда для представления временных особенностей передачи и приема сообщений между объектами используется диаграмма последовательности. На Рис.3.2 представлена диаграмма последовательности, на которой более детально описана логика сценария использования разработанного web-приложения пользователем. Клиент представляет собой некоторый браузер, сервер – это программа на языке Python обрабатывающая поступающие запросы, БД – сервер облачной базы данных BugQuery.

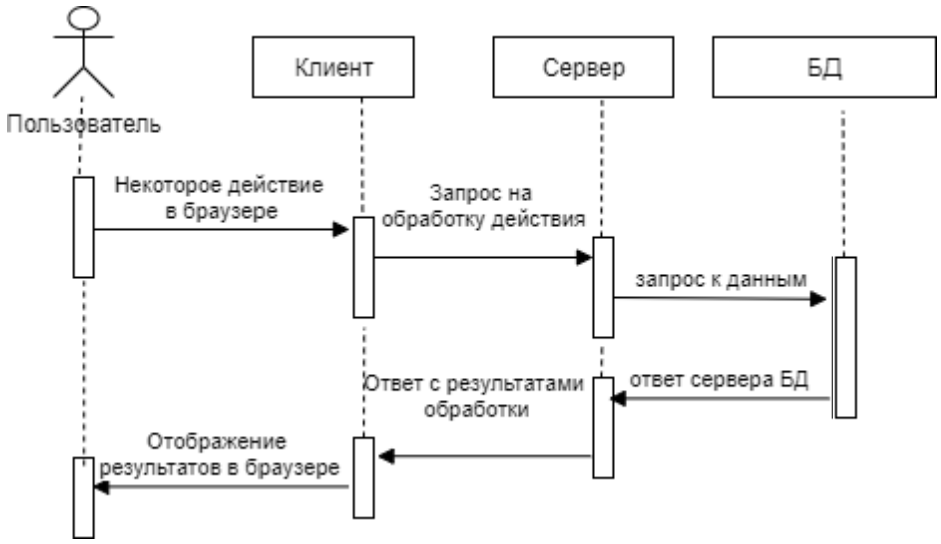

Рис. 3.2 – Диаграмма последовательности

Архитектура приложения делится на нижнеуровневую и верхнеуровневую архитектуру. На Рис.3.3 представлена модель взаимодействия *клиент-сервер*.

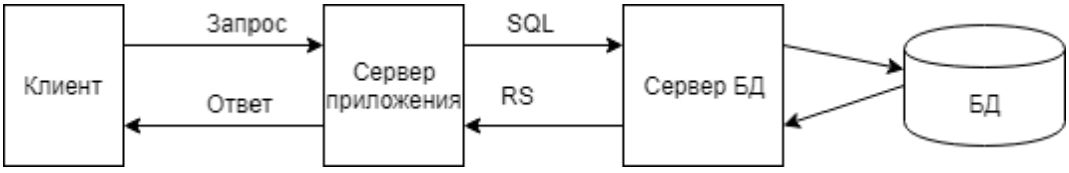

Рис. 3.3 – Архитектура «клиент-сервер»

Диаграмма компонентов, в отличие от ранее рассмотренных диаграмм, описывает особенности физического представления системы. Диаграмма компонентов позволяет определить архитектуру разрабатываемой системы, установив зависимости между

программными компонентами, в роли которых может выступать исполняемый код.

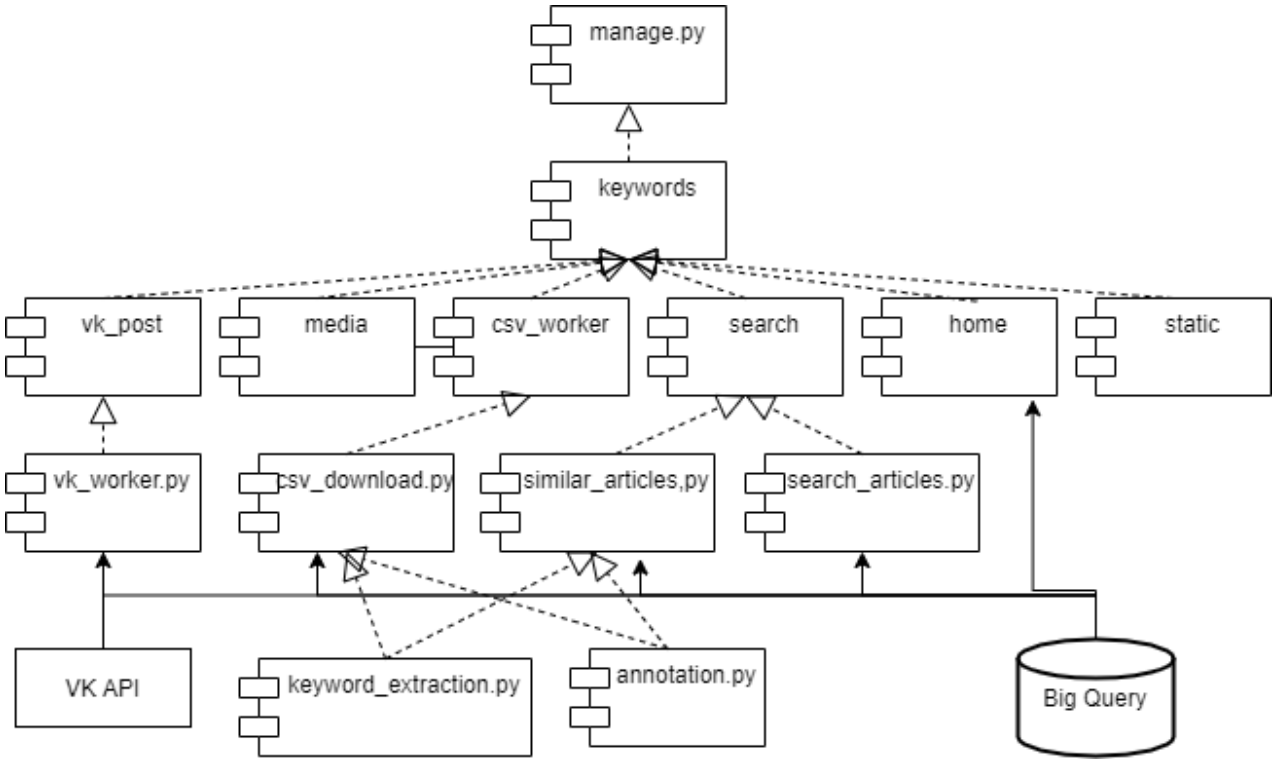

Рис. 3.4 – Диаграмма компонентов

На рис.3.4 представлена диаграмма компонентов данной системы. Компонент manage.py отвечает за инициализацию всего web-приложения, keywords отвечает за настройки web-приложения, vk_post – за логику обработку запросов пользователя и выдачу ему ответов разделов, связанных с работой соцсети, csv_worker – за обработку запросов пользователя и выдачу ему ответов разделов “Загрузка” и “Пакетная обработка”, search – за обработку запросов пользователя и выдачу ему ответов разделов “Найти похожие” и “Поиск по статьям”, home – за авторизацию, регистрацию, раздела “Личная библиотека” и отображение главной страницы, static – это компонент, в котором находиться css-файлы, media – компонент, в котором находятся статические файлы, в данном случае csv-файлы, vk_worker.py в зависимости от запроса пользователя обращается к VK API или выполняет другие задачи связанный с “ВКонтаке”, csv_download.py отвечает за работу с csv-файлами, similar_articles.py – выполняет задачу поиска похожих статей, search_articles.py отвечает за различные виды поиска, keyword_extraction.py – за выделение ключевых слов, annotation.py – за аннотирование текста, BigQuery – облачное хранилище данных.

Для моделирования процесса выполнения операций в языке UML используются так называемые диаграммы деятельности (Рис. 3.5).

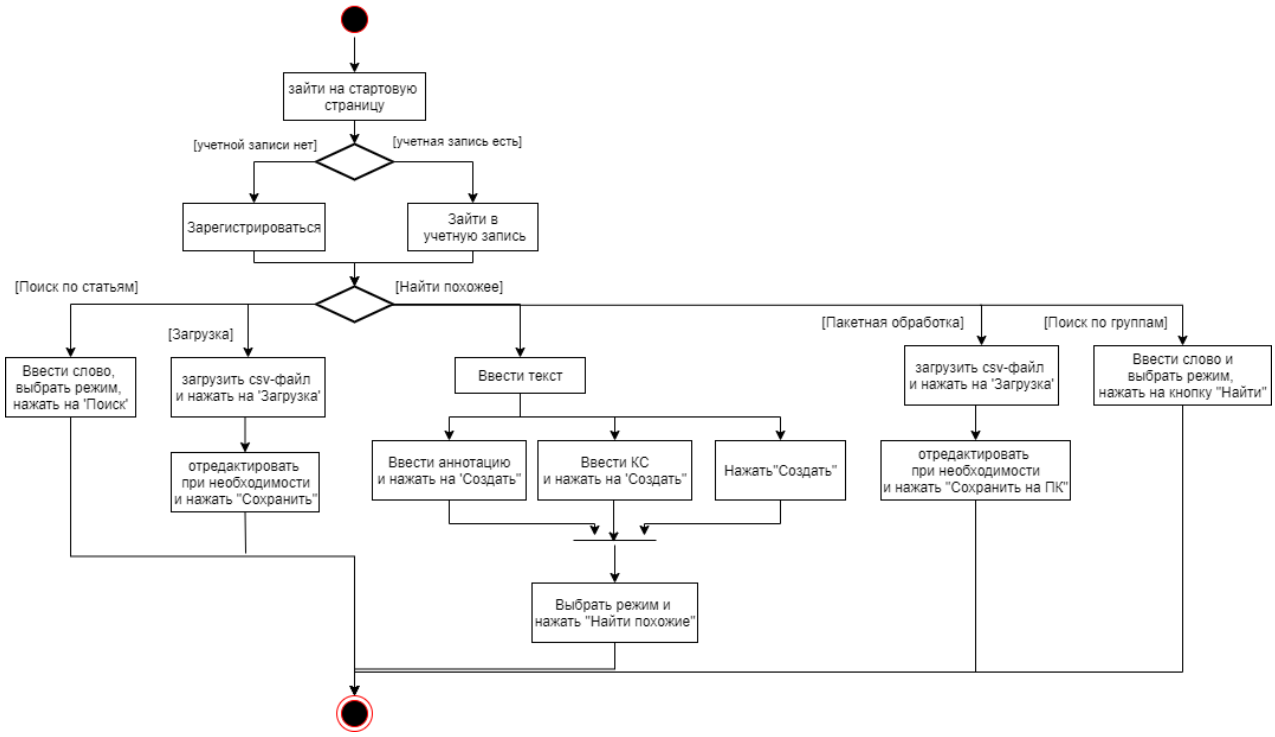

Рис. 3.5 – Диаграмма деятельности

3.3. Интерфейс web-приложения

Интерфейс – это набор элементов, которые в комплексе обеспечивают взаимодействие с web-приложением. Спроектированный интерфейс должен обеспечить выполнение всех функциональных требований.

На первом этапе работы с приложением необходимо зарегистрироваться, либо зайти в уже существующую учетную запись.

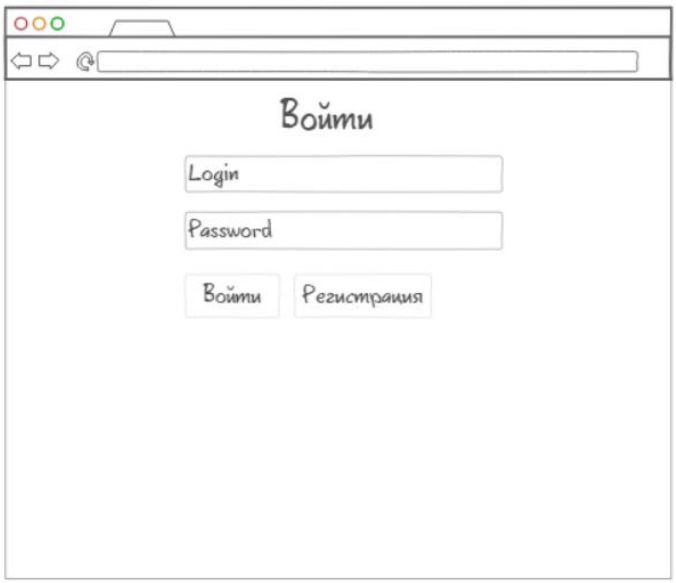

Рис. 3.6 –Макет стартовой страницы

Далее пользователь попадает на главную страницу приложения, на которой указана инструкция по работе с ним.

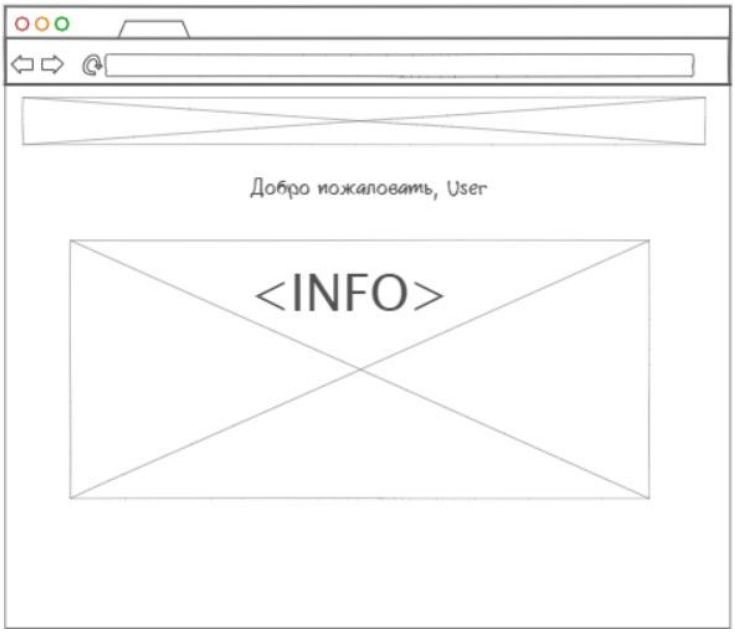

Рис. 3.7 – Макет главной страницы

Страницы, предназначенные для поиска по статьям и по постам из “ВКонтакте”, содержат форму для ввода слов и также различные режимы поиска.

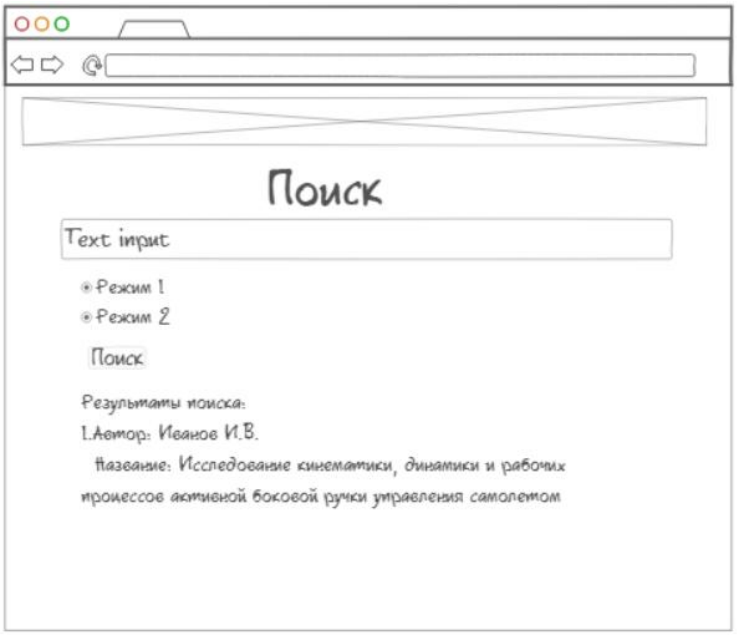

Рис. 3.8 – Макет страницы для поиска

Для выделения ключевых слов и аннотирования текстов, а также поиска похожих текстов разработан макет страницы, представленный на рис.3.9.

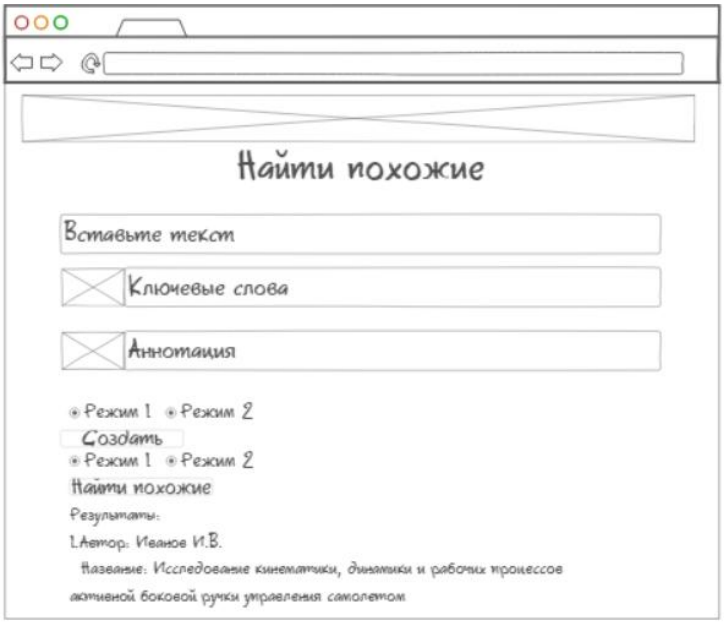

Рис. 3.9 – Макет страницы "Найти похожие"

Для пакетной обработки для этапа редактирования загруженного csv-файла разработан макет, представленный на рис.3.10.

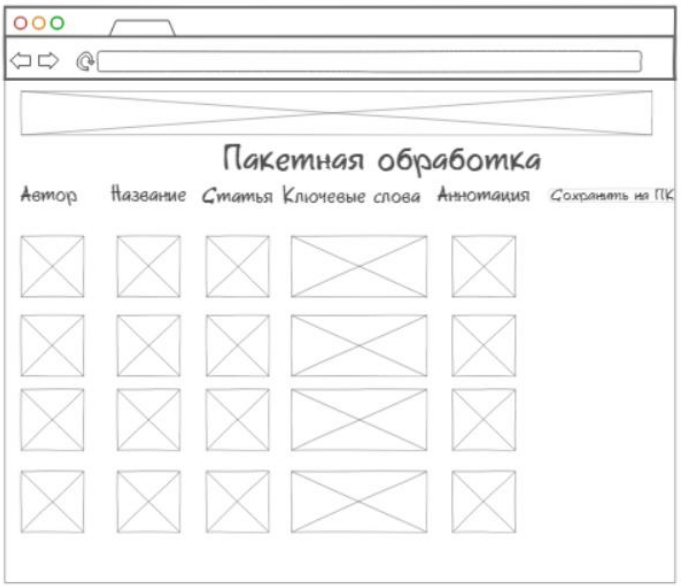

Рис. 3.10 –Макет страницы "Пакетная обработка"

3.4. Выводы

- Рассмотрен вопрос проектирования web-приложения с учетом функциональных и системных требований.
- Спроектирована архитектура интерфейса взаимодействия web-приложения с пользователем, были приведены примеры макетов страниц.
- Были построены физическая схема данных, диаграмма компонентов, диаграмма последовательности, а также приводится диаграмма активности.

Раздел 4. Программная реализация и тестирование приложения

В данном разделе описывается процесс реализации web-приложения, который включает разработку интерфейса, а также реализацию внутренней составляющей, а именно процесс предобработки текста, работу с базой данных, работу с csv-файлами, применение выбранного алгоритма для выделения ключевых слов и аннотирования научных статей обработки подготовленных данных, поиск похожих статей, возможность авторизации, а также процесс создания web-приложения с использованием библиотеки Django. Приводиться список используемых инструментов и их назначение. Описывают потенциальные ошибки, которые могут возникнуть во время работы и способы их устранения. Кроме этого проводится функциональное тестирование приложения, для этого составляется чек-лист. Проведена оценка качества работы алгоритмов.

4.1. Реализация web-приложения

Для реализации приложения были использованы следующие инструменты:

1) Python – высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис языка Python минималистичен. В то же время стандартная библиотека включает большой объем полезных функций [24].

2) PyCharm – интегрированная среда разработки для языка программирования Python. Она представляет собой средства для анализа кода и графический отладчик. PyCharm работает под операционными системами Windows, Mac OS X и Linux. На Рис. 4.1 приведен скрин данной среды.

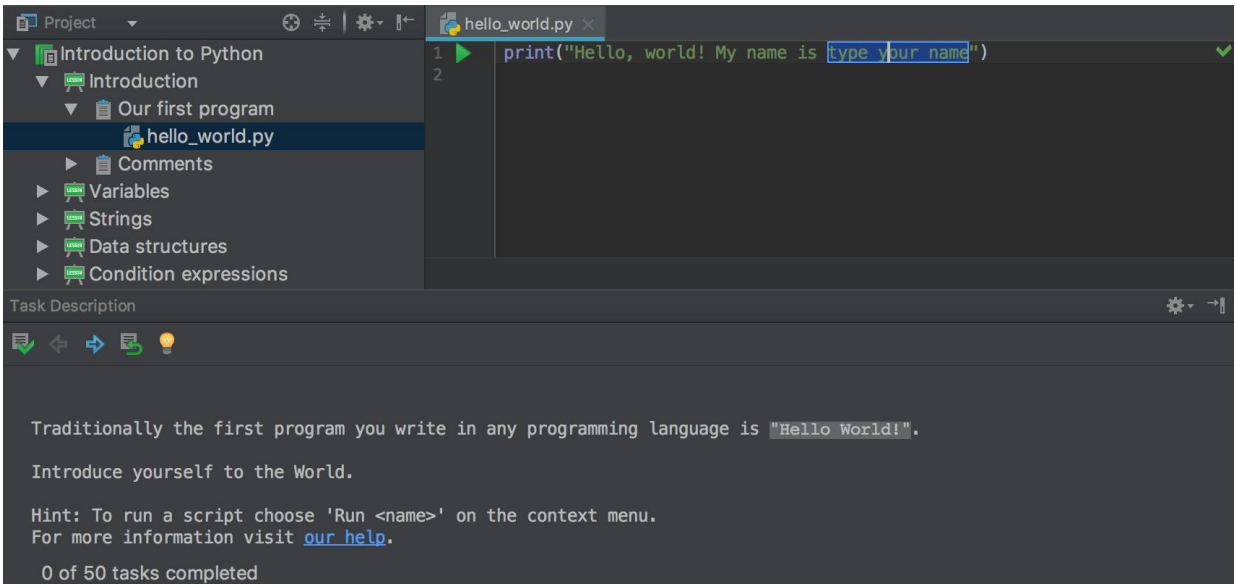

Рис. 4.1 – Среда разработки PyCharm

3) Django – фреймворк для веб-разработки на Python.

4) DeepPavlov – это библиотека с открытым исходным кодом для анализа текста. Она включает в себя различные модели нейронных сетей, набор инструментов для интеграции приложений со смежной инфраструктурой (мессенджеры, программное обеспечение службы поддержки и т. д.). Также она содержит удобный API для подключения к Scikit-learn, бесплатная библиотека машинного обучения, и Keras, открытая нейросетевая библиотека.

5) nltk – пакет библиотек и программ для символьной и статистической обработки естественного языка, написанных на языке программирования Python.

6) pymorphy2 — морфологический анализатор, разработанный на языке программирования Python. Он выполняет лемматизацию и анализ слов и способен осуществлять склонение по заданным грамматическим характеристикам слов. Анализатор работает со словарём OpenCorpora, а для незнакомых слов строит гипотезы. Поддерживаются русский и украинский языки.

7) re - встроенная библиотека для работы с регулярными выражениями. В данной работе она используется для удаления из текста всех символов, кроме букв русского алфавита и цифр.

8) pandas – это библиотека, которая предоставляет очень удобные с точки зрения использования инструменты для хранения данных и работе с ними.

9) requests — библиотека Python для выполнения HTTP-запросов. В данной работе она используется для скачивания научных статей, которые используются для оценки качества работы алгоритма.

10) csv – библиотека для работы с файлами формата csv.

11) networkx – библиотека предназначена для создания, манипуляции и изучения структуры, динамики и функционирования сложных сетевых систем. В данной работе используется для построения графа..

12) JSON (JavaScript Object Notation) – простой формат обмена данными, основанный на подмножестве синтаксиса JavaScript. Модуль json позволяет кодировать и декодировать данные в удобном формате.

13) OAuth2 – представляет собой фреймворк для авторизации, позволяющий приложениям осуществлять ограниченный доступ к пользовательским аккаунтам на HTTP сервисах.

14) pandas_gbq – модуль, который предоставляет оболочку для веб-службы аналитики BigQuery от Google, чтобы упростить извлечение результатов из таблиц BigQuery с помощью SQL-подобных запросов.

15) pickle – модуль , который используется для сериализации и десериализации объектов.

16) sys – данный модуль обеспечивает высокоуровневое взаимодействие с операционной системой.

17) Модуль os из стандартной библиотеки языка программирования Python используется для работы с файловой системой и ОС.

В результате реализации было получено web-приложение, удовлетворяющее требованиям из раздела 3.1.

Социальные сети, такие как Facebook, Twitter, ВКонтакте, стали неотъемлемой частью жизни многих людей. В данной работе в качестве одного из источников данных была выбрана социальная сеть «Вконтакте». Пользователям «ВКонтакте» доступен характерный для многих социальных сетей набор возможностей: создание профилей с информацией о себе, производство и распространение контента. Также пользователи «ВКонтакте» могут организовывать сообщества двух видов — группы (писать записи в ленте может любой участник) и публичные страницы (редактирование и публикация сообщений доступны лишь администрации). Для парсинга будет использоваться API ВКонтакте.

API ВКонтакте — это интерфейс, который позволяет получать информацию из базы данных vk.com с помощью http-запросов к специальному серверу [25]. Синтаксис запросов и тип возвращаемых ими данных строго определены на стороне самого сервиса. В ответ сервер вернет JSON-объект с запрошенными данными (или сообщение об ошибке, если что-то пошло не так). JSON — это формат записи данных в виде пар «имя свойства»: «значение».

Для работа с BigQuery необходимо иметь аккаунт в почте Google, также на сайте BigQuery нужно создать проект и получить json-ключ, позволяющий взаимодействовать с данными на сервере. Манипуляции с данными можно производить как через и интерфейс на сервере, так и с помощью программного кода и SQL запросов.

В языке Python все является объектами, в том числе и обученные модели. Исходя из этого, чтобы каждый раз при использовании обученных моделей не было необходимости при каждом запуске производить повторное обучение, отдельно от основных модулей, модели были подготовлены и сохранены с использование библиотеки pickle. Таким образом, это дает возможность не хранить обучающий набор данных и сокращает время работы программы.

Некоторый функционал разрабатываемого web-приложения предполагает под собой работу с csv-файлами. CSV-файлы - текстовый формат, предназначенный для

представления табличных данных. Строка таблицы соответствует строке текста, которая содержит одно или несколько полей, разделенных знаком-разделителем, в данной работе знак разделитель – точка с запятой.

В данной работе используется библиотека DeepPavlov, так как она включает в себя различные модели нейронных сетей, набор инструментов для интеграции приложений со смежной инфраструктурой (мессенджеры, программное обеспечение службы поддержки и т. д.). Также она содержит удобный API для подключения к Scikit-learn, бесплатная библиотека машинного обучения, и Keras, открытая нейросетевая библиотека.

Django представляет собой современный фреймворк, написанный на языке Python. Большим преимуществом Django является то, что многие решения предоставляет фреймворк, например, роутинг, ORM, авторизация, административная часть сайта и т.д. Архитектурой Django является MVC (Model-View-Controller), также её именуют MTV(Model-Template-View), потому что логика отображения данных находится в Templates, а обработка ответа пользователю во Views.

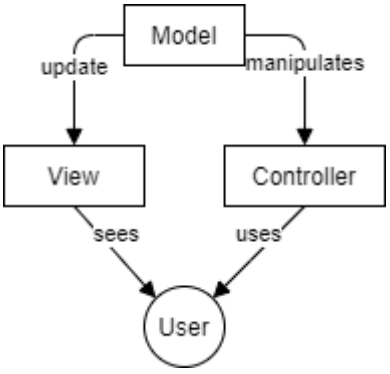

Рис. 4.2 – Model-View-Controller

Сайт на Django состоит из нескольких приложений, которые независимы друг от друга и их можно переиспользовать в других проектах. Каждое приложение состоит из нескольких файлов, в которых описывается логика приложения. В models.py описываются модели приложения. Описание модели представляет собой класс, в котором указываются все поля таблицы базы данных. В admin.py указывается, то как будут отображаться модели приложения в административной части ресурса, например, поля моделей, которые будут отображены, по каким полям сортировать данные и т.д. В forms.py описываются формы, которые будут использованы в приложении. Такие формы позволяют валидировать данные, которые ввел пользователь. В urls.py указывается основной роутинг приложения. Во views.py описывается логика приложения, а именно то, как приложение будет отображать данные и возвращать их в зависимости от действий пользователя.

На рис. 4.3 представлена страница для входа в систему, предварительно необходимо авторизоваться.

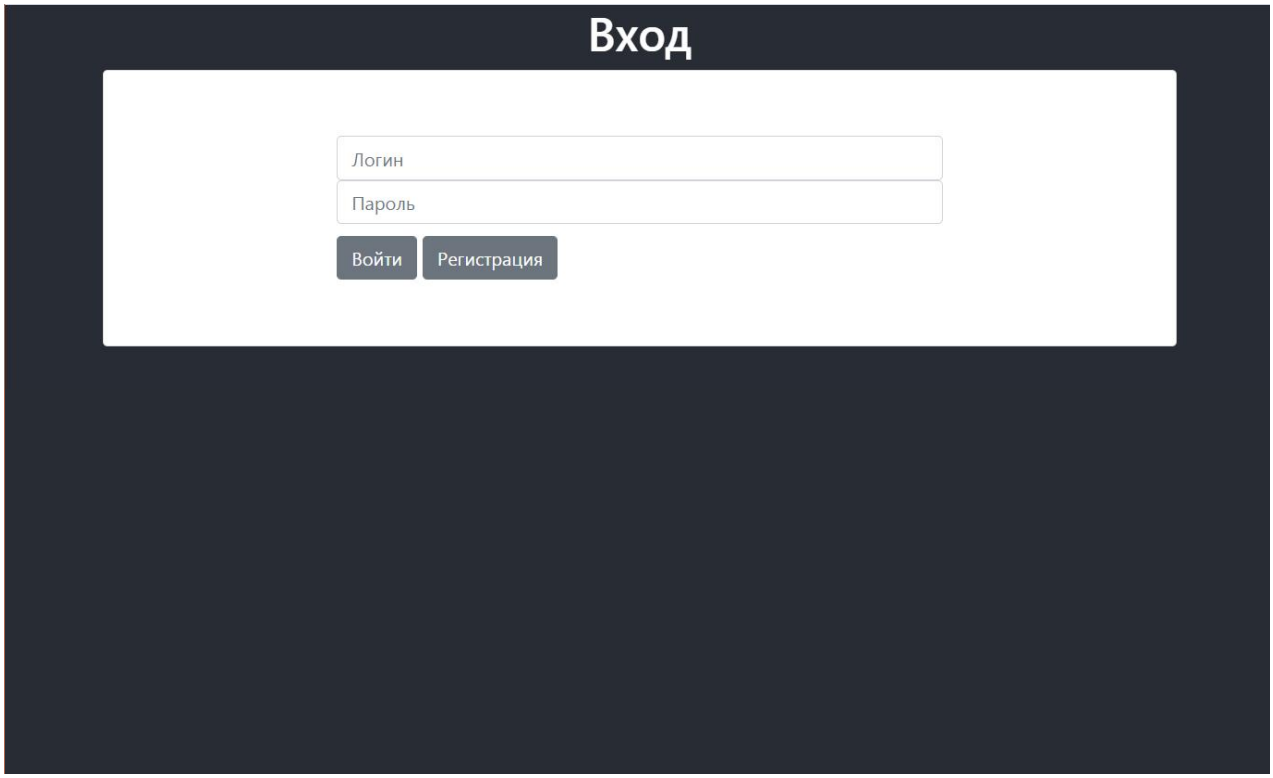

Рис. 4.3 – Страница для входа в систему

На рис. 4.4 представлена страница по поиску в различных источниках и на рис.4.5 результаты поиска по запросу “цифровые сигналы”.

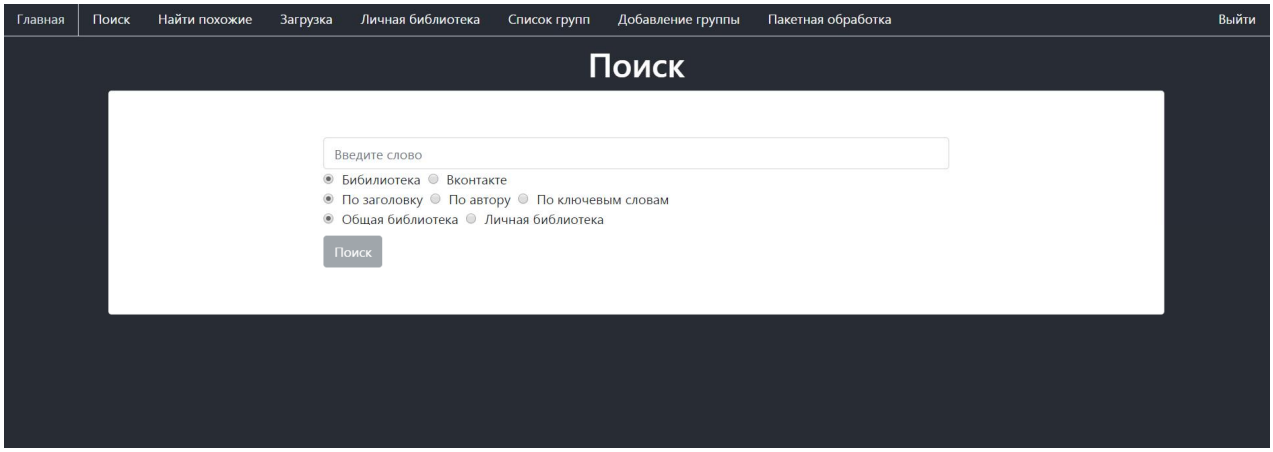

Рис. 4.4 – Страница для входа в систему

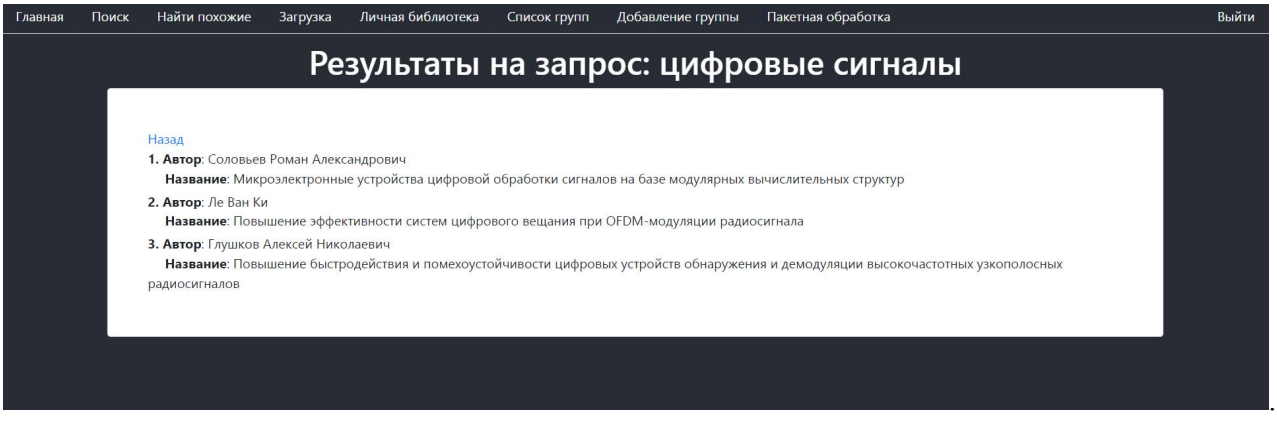

Рис. 4.5 – Страница поиска с результатом запроса

На рис. 4.6 представлена страница для пакетной обработки после загрузки csv-файла.

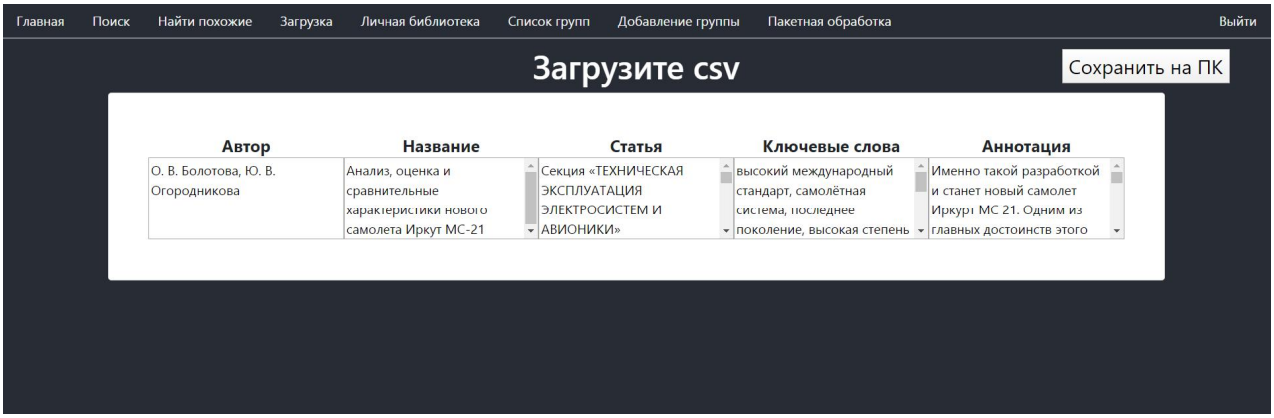

Рис. 4.6 – Страница "Пакетная обработка"

На рис. 4.7, 4.8 представлена страница для аннотирования и выделения КС одиночной статьи, а также для поиска похожих статей по ключевым словам.

Рис. 4.7 – Страница "Найти похожие"

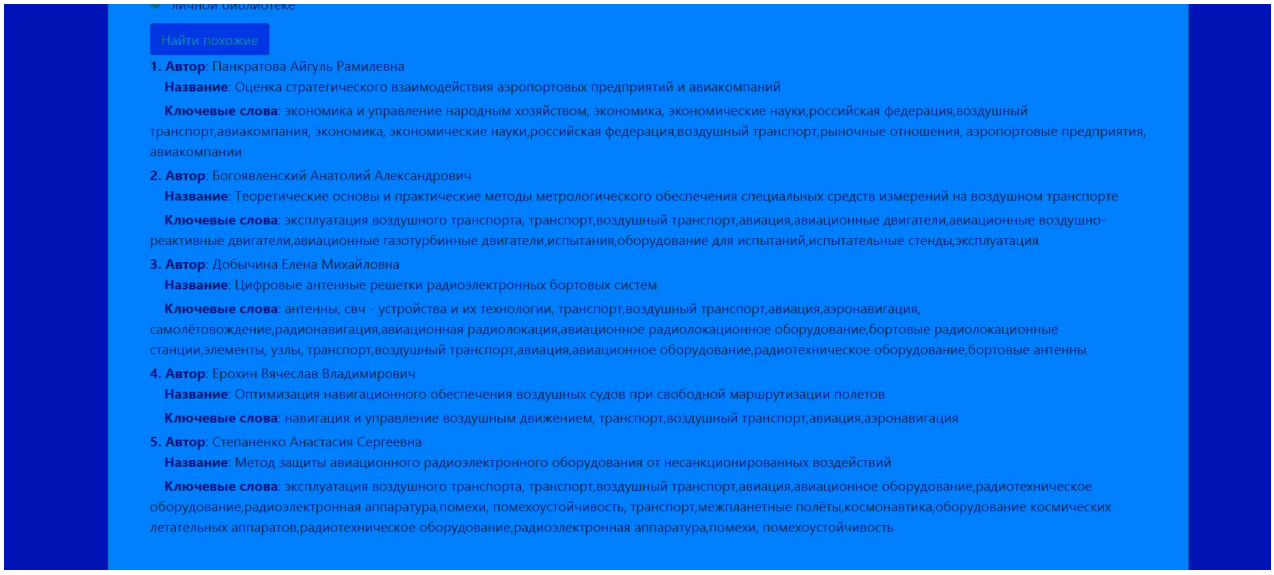

Рис. 4.8 – Страница "Найти похожие"(продолжение)

4.2. Оценка качества работы алгоритмов

Для оценки качества работы алгоритмов была вручную сформирована тестовая выборка из 40 статей различной тематики. Для оценки качества работы алгоритма выделения ключевых слов необходимо сначала сравнить качество работы классификатора при гибридном подходе.

Таблица 4.1 – Оценка качества работы классификатора

Классификатор	precision “0”	recall “0”	F-мера “0”	precision “1”	recall “1”	F-мера “1”
Логистическая ререссия	0.85	0.87	0.86	0.84	0.82	0.83
CNN	0.82	0.43	0.56	0.57	0.89	0.7

В таблице 4.2 сравниваются два алгоритма выделения КС.

Таблица 4.2 – Оценка качества работы алгоритмов выделения ключевых слов

	precision	recall	F-мера
Гибридный подход	0.41	0.69	0.51
RAKE	0.92	0.14	10.6

Таким образом, гибридный подход показывает лучшее качество по сравнению с алгоритмом RAKE.

В таблице 4.3 приводятся результаты оценки качества работы алгоритма для автоматического аннотирования.

Таблица 4.3 – Оценка качества работы алгоритма аннотирования

Метрики	коэффициентом Сёренсена	косинусное расстояние
Precision	80.52	82.34
Recall	36.02	36.42
F-мера	49.78	49.97

4.3. Тестирование web-приложения

Для тестирования работы web-приложения была сформирована Таблица 4.4, которая представляет из себя чек-лист. Чек-лист – это документ, который включает в себя некоторое действий, которые необходимо проверить на корректность выполнения и результата выполнения. Если данный функционал отрабатывает без ошибок, статус «+/-», если замечены незначительные ошибки, и статус «-», если результат не достигнут. Полный список тестов указан в приложении 2.

Таблица 4.4 – Функциональное тестирование

№	Действие пользователя	Ожидаемый результат	Успешность прохождения
1	Регистрация в приложении	Регистрация успешна	+
2	Ввести в поле ввода словосочетание, выбрать режим, нажать на кнопку “Поиск”	Появление результатов поиска	+
3	В разделе “Пакетная обработка” выбрать файл, загрузить	Появление таблицы, с возможностью редактирования	+
4	В разделе “Найти похожие”, вставить текст, нажать “создать”, далее нажать “найти”	Появление в соответствующих полях ключевых слов и аннотации, появления списка похожих статей	+

4.4. Выводы

- Указаны инструменты для разработки, описаны различные особенности реализации.
- Описан процесс реализации web-приложения: разработку интерфейса, реализацию процесса предобработки текста, работу с базой данных, работу с csv-файлами, поиск похожих статей и общий поиск, возможность авторизации и другой функционал.
- Проведена оценка качества работы алгоритмов.
- Составлен чек-лист, по которому необходимо проводить тестирование приложения.

Заключение

В рамках данной учебно-исследовательской работы достигнута цель: разработано web-приложения для автоматического определения ключевых слов и аннотирования текстовой информации из открытых источников для решения задачи информационного поиска.

Для достижения поставленной цели были решены следующие задачи:

1. проведен анализ предметной области;
2. изучены алгоритмы автоматического выделения ключевых слов;
3. изучены алгоритмы автоматического аннотирования;
4. определены способы предобработки;
5. определены метрики оценка качества;
6. разработаны требования к приложению;
7. изучен алгоритм ближайшего соседа;
8. спроектировано web-приложения;
9. разработан интерфейс;
10. реализовано web-приложения;
11. проведено тестирование web-приложения;
12. оценено качество работы приложения.

Список литературы

1. Gutwin C., Paynter G., Witten I. Improving browsing in digital libraries with keyphrase indexes//Decision Support Systems 27(1-2), 81-104(2003)
2. Witten I., Paynte G., Frank E., Gutwin C. KEA: Practical Automatic Keyphrase Extraction URL:https://www.researchgate.net/publication/221347768_KEA_Practical_Automatic_Keyphrase_Extraction (дата обращения 01.04.2020)
3. CleverStatSitecontentanalyzer [Элекстронный ресурс] URL: http://cleverstat.com/ru/sca-website-analysis-software-index.htm (дата обращения 01.04.2020)
4. Tesuck. Веб-сервис Tesuck [Электронный ресурс] URL: https://tesuck.eveel.ru (дата обращения 01.04.2020)
5. Arnaldo C., Thiago L., Portal Min@s: Uma Ferramenta de Apoio ao Processamento de Córpus de Propósito Geral//Proceedings of the 10th Brazilian Symposium in Information and Human Language Technology, 2015, pp. 69–73
6. J. Ramos, “Using tf-idf to determine word relevance in document queries,” // Proceedings of the first instructional conference on machine learning, 2003, pp. 1-4.
7. Y. Matsuo, M. Ishizuka, “Keyword extraction from a single document using word co-occurrence statistical information”// International Journal on Artificial Intelligence Tools, vol. 13 (01), 2004, pp. 157-169
8. Lee-Feng Chien, “Pat-tree-based keyword extraction for Chinese information retrieval,” in: ACM SIGIR Forum, Vol. 31, ACM, 1997, pp. 50-58
9. Kovacevic A. Automatic extraction of metadata from scientific publications for CRIS systems//Program electronic library and information systems 45(4):376-396 · September 2011
10. Wang J. Using Convolutional Neural Networks to Extract Keywords and Keyphrases About Foodborne Illnesses// 2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA) ,December 2019
11. Qi Zhang Keyphrase Extraction using deep recurrent neural networks on Twitter //Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, January 2016
12. Rada Mihalcea and Paul Tarau, ‘TextRank: Bringing Order into Texts’, in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004), Barcelona, Spain, July 2004
13. Stuart Rose, Nick Cramer Automatic Keyword Extraction from Individual Documents // Text Mining: Applications and Theory, pp.1 – 20, 2010
14. Ефремова М.И. Автоматический разбор и аннотирование статей// Фундаментальные

исследования. – 2015. - №2 (часть 22) – С. 4866-4870
15. Yousefi-Azar M. Semi-supervised convolutional extreme learning machine Conference Paper (PDF Available) //International Joint Conference on Neural Networks (IJCNN) May 2017
16. Осминин П.Г. Модель автоматического реферирования на основе базы знаний, ориентированный на автоматический перевод //Вестник Южно-Уральского государственного университета. Серия: Лингвистика2014 год, №2 С.65-69
17. Губин М.В., Меркулов А.И. Эффективный алгоритм формирования контекстно-зависимых аннотаций // Компьютерная лингвистика и интеллектуальные технологии. Труды международной конференции «Диалог'2005» (Звенигород, 1-6 июня 2005 г.). – М.: Наука, 2005. – С. 116–120
18. Логистическая регрессия [Электронный ресурс]. URL: machinelearning.ru/wiki/index.php?title=Логистическая регрессия (дата обращения 15.04.2020)
19. Yann LeCun, Yoshua Bengio & Geoffrey Hinton. Deep learning. Nature 521, 436–444 (28 May 2015)
20. Rada Mihalcea and Paul Tarau, 'TextRank: Bringing Order into Texts', in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004), Barcelona, Spain, July 2004
21. Yang Y. An Evaluation of Statistical Approaches to Text Categorization. / Journal of Information Retrieval, 1999 — V.1 — pp. 67-88
22. Chin-Yew Lin. ROUGE: A Package for Automatic Evaluation of Summaries /Proceedings of the Challenges Workshop, pp. 98-103, 2004
23. Автоматическая обработка текстов на естественном языке и анализ данных : учеб. пособие / Большакова Е.И., Воронцов К.В., Ефремова Н.Э., Клышинский Э.С., Лукашевич Н.В., Сапин А.С. — М.: Изд-во НИУ ВШЭ, 2017. — 269 с.
24. Lutz M. Learning Python: Powerful Object-Oriented Programming. – « O'Reilly Media, Inc.», 2013.
25. Знакомство с API ВКонтакте [Электронный ресурс]. URL: vk.com/dev/first_quide (дата обращения 23.04.2020)
26.

Приложения