МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование кафедры)

09.03.03 Прикладная информатика
(код и наименование направления подготовки)

Бизнес-информатика
(направленность (профиль) / специализация)

# ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка информационной системы для анализа поисковых запросов в социальных сетях»

Обучающийся Д.Э. Семенов
(Инициалы Фамилия) (личная подпись)

Руководитель к.т.н, В.С. Климов
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2022

## Аннотация

Тема бакалаврской работы – «Разработка информационной системы для анализа поисковых запросов в социальных сетях».

Анализ поисковых запросов позволяет определить ключевые темы, интересные для каждого отдельно взятого человека. Технологии анализа запросов используются в системах рекомендации товаров и контента в социальных сетях. Использование этих технологий позволяет сформировать представления о предпочтениях человека. Данная бакалаврская работа направлена на совершенствование технологий анализа поисковых запросов.

Актуальность работы заключается в разработке системы для анализа поисковых запросов и визуализации полученных результатов.

Объектом исследования бакалаврской работы является технологии анализа текстовых данных (text mining).

Предметом исследования бакалаврской работы является система для анализа и визуализации текстовых запросов.

Цель выпускной квалификационной работы – разработка информационной системы анализа поисковых запросов в социальных сетях для определения для определения интересов пользователя.

Методы исследования – технологии текстовых данных (text mining), технологии проектирования информационных систем, технологии программирования.

Данная работа состоит из введения, трех глав, заключения и списка используемой литературы.

В первой главе работы описываются особенности анализа текстовых данных, во второй главе приводятся методы анализа поисковых запросов и визуализации результатов, в третьей главе описана практическая реализации системы для анализа поисковых запросов в социальных сетях.

Бакалаврская работа состоит из 41 страницы текста, 26 рисунков и 20 источников.

## Оглавление

# Введение

Развитие технологий text mining связано с необходимостью анализа и понимания естественных языков. В настоящее время text mining применяется при создании чат-бот, инкассировании текстовых документов, анализе эмоциональной составляющей текста, определения ключевой темы текста и при решении других задач [1], [5].

В данной бакалаврской работе разрабатывается система определения интересующих пользователя тем на основе анализа его текстовых запросов в социальных сетях с использованием технологий text mining [8].

Актуальность выбранной темы исследования обусловлена в первую очередь ростом популярности социальных сетей. Так по официальным данным социальной сети Вконтакте ее ежедневно посещают около 52% российских интернет-пользователей [4]. Одновременно с этим доход социальных сетей, связанный с рекламой, зависит в количества времени, проведенном в них пользователями. Для удержания внимания пользователей социальной сети необходимо понимать интересы каждого пользователя и рекомендовать для просмотра тот контент, который ему будет интересен.

Источником информации об интересах пользователя может являться как данные, указанные им в анкете (город проживания, увлечения и т.д.), так и статистическая информация о просмотренных им страницах в социальных сетях (тематические публики и группы).

В рамках данного исследования предложено расширить список источников информации об интересах пользователя и добавить туда данные, полученные в ходе анализа поисковых запросов пользователя.

Проблема использования поисковых запросов для определения интересов пользователя заключается в том, что обычно задаются в произвольной форме в текстовом виде. Для извлечения требуемых данных из текста необходимо применение технологий text mining [7].

Цель выпускной квалификационной работы – разработка информационной системы анализа поисковых запросов в социальных сетях для определения для определения интересов пользователя.

Для достижения данной цели необходимо выполнить следующие задачи:

– моделирование процесса определения интересов пользователя;

– проектирование системы для анализа поисковых запросов в социальных сетях;

– реализация и тестирование системы поисковых запросов.

Методы исследования – технологии текстовых данных (text mining), технологии проектирования информационных систем, технологии программирования.

Практическая значимость бакалаврской работы заключается в разработке программного продукта для оценки интересов пользователя на основе анализа его поисковых запросов, представленных в текстовом виде.

Данная работа состоит из введения, трех глав, заключения, списка используемой литературы и приложений.

В первой главе приводится описание исследований в области text mining, раскрывается проблема удержания внимания пользователей социальных сетей, а также моделируется процесс определения интересов пользователей в нотации IDEF0.

Вторая глава посвящена проектированию системы для анализа поисковых запросов в социальных сетях. В этой главе раскрывается схема функционирования разрабатываемого программного обеспечения и описывается алгоритм анализа текстовых данных.

В третьей главе представлен процесс разработки программного обеспечения. Также здесь приведены результаты тестирования программного модуля на собственных поисковых запросах.

В заключении описываются результаты выполнения выпускной

квалификационной работы.

В ходе выполнения бакалаврской работы на языке программирования python разработано приложение, реализующее следующий функционал по анализу текстовых поисковых запросов: загрузка и обзор данных о поисковых запросах, предварительная обработка текстовых запросов, очистка текстовой информации от стоп-слов, трансформация слов в основную форму, частотный анализ слов и визуализация наиболее часто используемых слов в запросах в виде облака.

Разработанное программное обеспечение протестировано на реальных текстовых поисковых запросах.

Бакалаврская работа состоит из 41 страниц текста, 26 рисунков, и 20 источников.

# Глава 1 Обзор технологий text mining

## 1.1 Области применения технологий анализа текстовых данных

Сферы применения технологий искусственного интеллекта в последние годы значительно расширилась. Аналитические обзоры в различных научных статьях показывают, что он при применяется: в робототехнике, при анализе изображений, при перевыполнении автоматического перевода текста, в бизнес-аналитике, при распознавании зрительных образов, в задачах распознавания текстов, при извлечении информации, в экспертных системах, в интеллектуальных системах информационной безопасности, при распознавании речи, при анализе текстов на естественном языке. Сферы применения искусственного интеллекта показаны на рисунке 1.

Расширение сфер искусственного интеллекта связано с возможностью повышения степени оптимизации решения практических задач, а следовательно, и с повышением производительности труда.

Отдельным направлением, которое в настоящее время активно развивается является анализ активностей пользователей в социальных сетях. Действия пользователей социальных сетей анализируется для решения различных задач [9]:

- удаление фейковых аккаунтов пользователей;
- пресечение неправомерных действий в социальных сетях;
- увеличение времени удержания пользователя на площадке социальной сети;
- продвижение товаров, связанных с интересами пользователя;
- организации рекомендательной системы контента в зависимости от интереса пользователей;
- косвенная оценка реакции пользователей на различные события.


Рисунок 1 – Сферы применения искусственного интеллекта

Так как деятельность компаний, владеющих сервисами социальных сетей направлена в первую очередь на извлечение прибыли, то важнейшей

целью является удержания пользователя на своей площадке как можно большее время. Для этого требуется понимать интересы каждого отдельного пользователя, что бы рекомендовать ему тот контент для просмотра, который ему будет интересен.

Источником информации об интересах пользователя может являться как данные, указанные им в анкете (город проживания, увлечения и т.д.), так и статистическая информация о просмотренных им страницах в социальных сетях (тематические публики и группы).

В рамках данного исследования предложено расширить список источников информации об интересах пользователя и добавить туда данные, полученные в ходе анализа поисковых запросов пользователя.

Проблема использования поисковых запросов для определения интересов пользователя заключается в том, что обычно задаются в произвольной форме в текстовом виде. Для извлечения требуемых данных из текста необходимо применение технологий text mining.

Анализ литературных источников показал, что text mining происходит по схеме, показанной на рисунке 2. Сначала осуществляется сбор данных, в качестве источников данных могут быть использованы текстовые документы, веб-страницы и комментарии пользователей. Затем осуществляется парсинг текста, который заключается в извлечении текстовой информации из собранных данных [16], [17]. На этапе фильтрации текста отбрасываются не нужная часть текста, при этом в качестве ориентиров используются старт и стоп слова, а также списки слов, подлежащие фильтрации. Если text mining’у подвергаются большие объемы текстовых данных, то производится преобразование пространства признаков, когда каждый текстовый документ описывается уже с помощью вектора токенов. И на последнем этапе осуществляется анализ извлечённых из текста признаков для решения поставленной задачи. В ходе контроля результатов text mining может потребоваться корректирование алгоритма анализа данных [18], [19], [20].


Рисунок 2 – Схема анализа текстовых данных (text mining)

В различных источниках литературы указывается, что решаемые с помощью text mining задачи можно условно разделить на 4 группы: задачи классификации, задачи кластерного анализа, задачи регрессионного анализа и задачи аффинитивного анализа [10], [12], [13]. Примеры задач из каждой группы по казаны на рисунке 3.

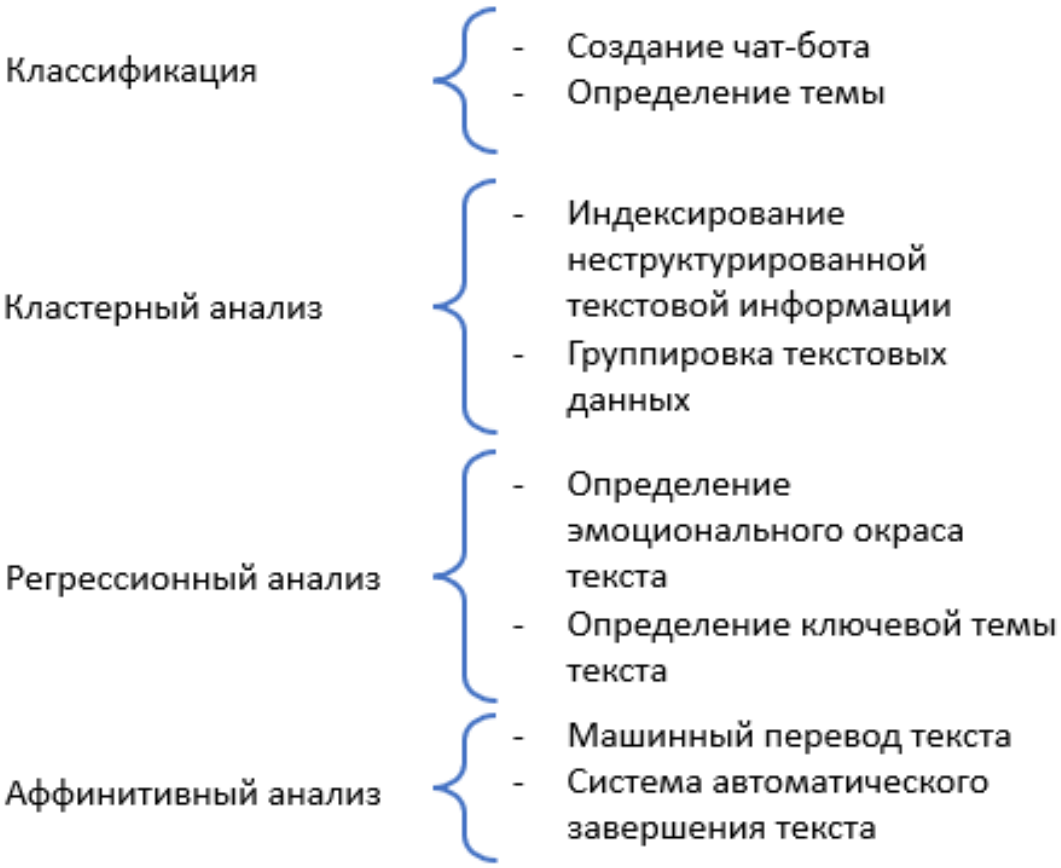



Рисунок 3 – Различные типы задач при анализе текстовых данных

В нашем случае text mining будет использоваться для определения списка интересных пользователю тем на основе анализа поисковых запросов в социальной сети. Завершим описание теоретической части text mining и перейдем к моделирования процесса определения интересов пользователя.

## 1.2 Моделирование процесса определение интересов пользователя

Моделирование процесса определения интересов пользователя будем производить с использование программного обеспечения ERwin Process Modeler [6], [14].

Входными параметрами процесса определения интересов пользователя являются:

- информация об активности пользователя в социальной сети;
- данные пользователя.

К информации об активности пользователя относятся совершаемые им действия в социальной сети, включающие в себя: время в хода в свой профиль; устройство, с которого осуществлён вход; комментарии, оставленные пользователем; активность, связанная маркировкой контента (например, использованием кнопки “Like”); поисковые запросы пользователя с привязкой ко времени. К данным пользователя относится информация из его анкеты, а также индивидуальный уникальный идентификатор.

Процесс определения интересов пользователя регулируется законами РФ, а также внутренними правилами пользования социальной сети, прописанным в соглашении пользователя.

Выполнение процесса определения интересов пользователя, осуществляется с использованием персонала, различных материально-технических средств, а также с использованием разрабатываемой в рамках бакалаврской работы информационной системы.

В результате выполнения процесса определение интересов пользователя мы получаем список интересующих пользователя тем, а также графики с визуализацией интересов пользователя.

Построенная модель TO-BE анализируемого процесса показана на рисунке 4.

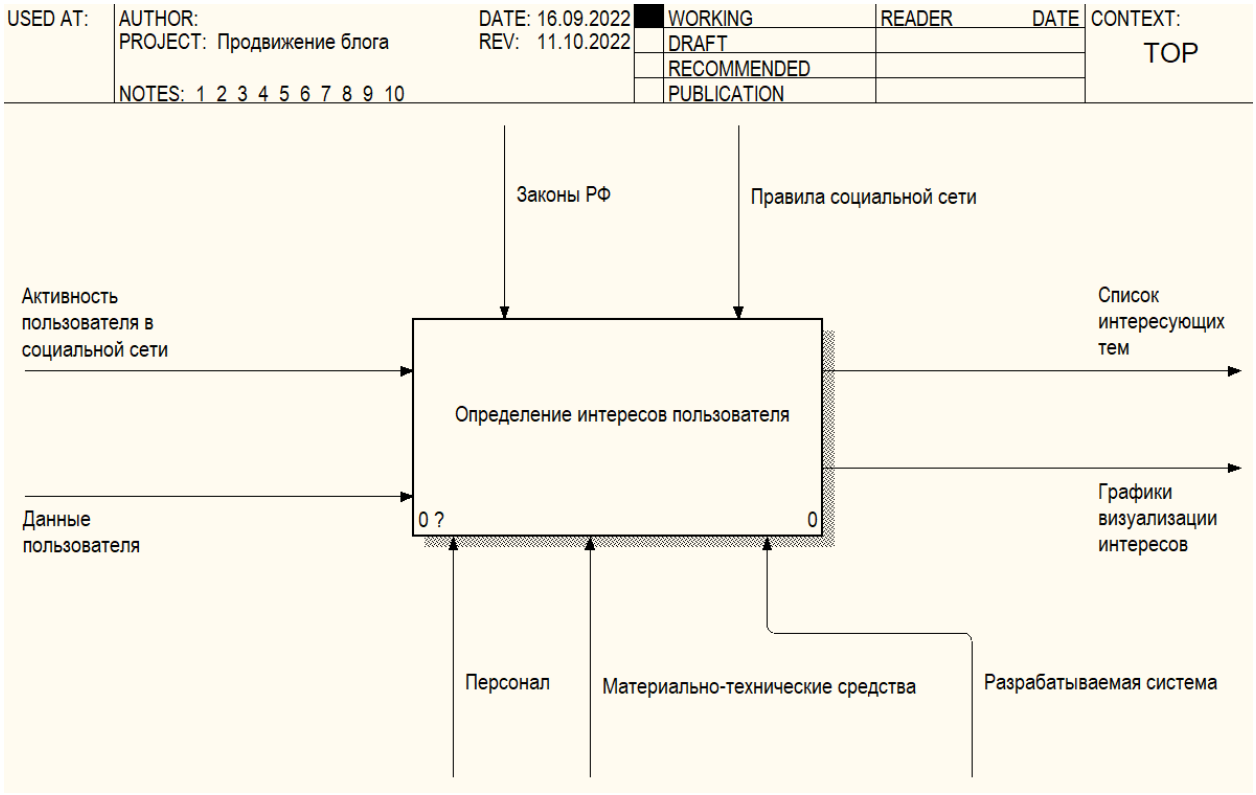



Рисунок 4 – Модель TO-BE, уровень A-0

Проведем декомпозицию процесса «Определение интересов пользователя». Выполнение данного процесса связано со следующими основными действиями:

– сбор поисковых запросов;
– подготовка данных для анализа;
– анализ данных.

Сбор поисковых запросов включает в себя – просмотр активностей пользователя и формирование набора использованных поисковых запросов в текстовом виде. Далее собранный набор поисковых запросов пользователя проходит этап подготовки данных для анализа. На данном шаге, в соответствии с концепцией text mining производится преобразование текста в набор токенов (отдельные слова, приведенные в базовую форму). Выполнение

этого этапа осуществляется с использованием разрабатываемой информационной системы. На следующем шаге полученный список токенов подвергается анализу, в результате которого:

– определяется список интересующих пользователя тем;

– формируются графики, визуализирующие интересы пользователя.

С учетом выше сказанного модель TO-BE процесса «определение интересов пользователя» будет выглядеть так, как это показано на рисунке 5.

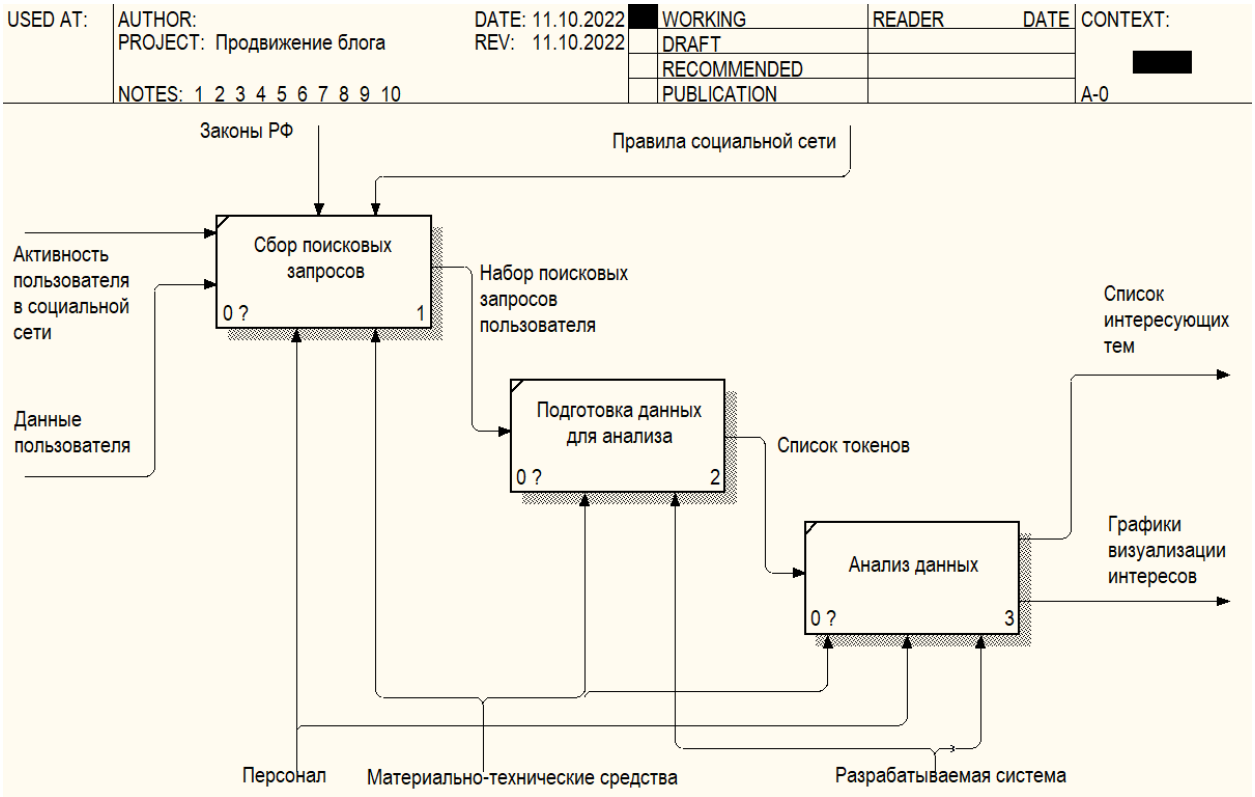



Рисунок 5 – Модель TO-BE, декомпозиция

Таким образом, проведено моделирование процесса «Определение интересов пользователя» и определена роль разрабатываемой системы в данном процессе.

## Выводы по главе 1

Приведем выводы по первой главе бакалаврской работы:

– в ходе анализа литературных данных установлено, что одной важных задач для компаний, владеющих сервисами социальных сетей, является удержание пользователей на своих площадках;

– для удержания внимания пользователей сервисам необходимо уметь определять интересы пользователей с целью предоставления интересующего их контента;

– в рамках бакалаврской работы предложено производить определение интересов пользователя на основе их текстовых поисковых запросов, которые предлагается анализировать с помощью технологий text mining;

– анализ литературных источников показал, что основными этапами text mining является: сбор данных, парсинг текста, фильтрация текста, преобразование пространства признаков и анализ данных.

– с использованием методологии IDEF0 проведено функциональное моделирование процесса «Определение интересов пользователя» в ходе которого определена составные элементы процесс и роль разрабатываемой информационной системы.

# Глава 2 Проектирование системы для анализа поисковых запросов в социальных сетях

## 2.1 Алгоритм анализа поисковых запросов

При проведении исследований в рамках бакалаврской работы предложена следующая схема функционирования программного модуля (рисунок 6). Пользователь осуществляет взаимодействие с социальной сетью и ищет в ней контент на интересующую в данный момент времени тему. Поисковые запросы пользователя постепенно накапливаются или в текстовом файле, или в базе данных. Когда требуется получить и аналитику по вопросу какими темами интересуется пользователь все собранные поисковые запросы отправляются для анализа в разрабатываемый программный модуль.

Если требуется определить список интересующих пользователя тем за определенный период, например за последний месяц, то на анализ в программный модуль передаются не все поисковые запросы, а только за необходимый период.

С помощью программного модуля можно определять список интересующих тем не только для отдельно взятого пользователя, но и для группы пользователей, объединённых общим признаком, например, городом проживания. Для этого на анализ отправляются объединенный список поисковых запросов группы пользователей.

Также если есть доступ к поисковым запросам пользователя из нескольких разных социальных сетей, то их также можно объединить в единый список для более емкого анализа интересующих пользователя тем.

По результатам анализа программный модуль определяет список интересующих пользователя тем с использованием частотного анализа.

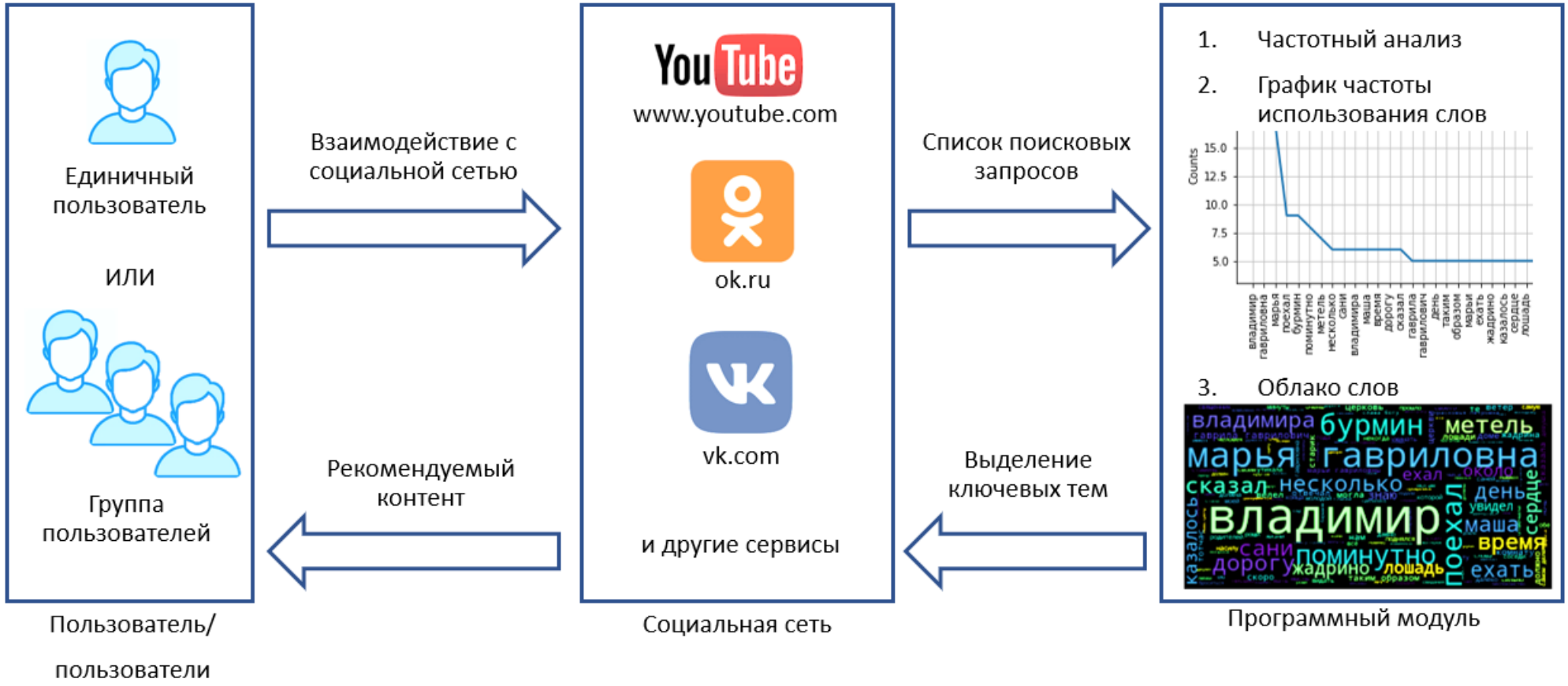



Рисунок 6 – Схема работы программного обеспечения для анализа поисковых запросов пользователей в социальных сетях

Базовым сценарием использования выделенных ключевых тем является передача полученных данных в сервис социальной сети. На основе полученных данных социальная сеть корректирует рекомендуемый контент для просмотра пользователю. Если, например пользователь интересуется программированием, то ему в рекомендуемом будут демонстрироваться группы по программированию, в также видео, посвящённые программированию и т.д.

Для удобства человеческого восприятия результатов программный модуль по результатам частотного анализа строит график популярности слов, а также облако слов.

Для дальнейшего машинного анализа данных программный модель на выход передает массив данных (вектор слов), состоящий из пар слово и частота его использования в поисковых запросах (рисунок 7). При необходимости эти данные можно подвергнуть анализу, например, с помощью алгоритмов машинного обучения, чтобы найти в них скрытые закономерности. Однако этот вопрос выходит за рамки данной бакалаврской работы.

Для реализации описанного выше функционала в программном модуле используется готовые компоненты, такие как:

– библиотека для анализа текстовых данных NLTK;
– библиотека для построения графиков Matplotlib;
– библиотека для построения облака слов WordCloud.

Использование готовых компонентов существенно облегчает разработку программного обеспечения, так большая часть требуемого функционала уже реализована в предоставляемых ими методах.

В качестве платформы работы программного модуля используется сервис облачных вычислений Google Colab, однако при необходимости разработанный программный модуль можно запускать локально на выбранном сервере с использованием платформы Anaconda.

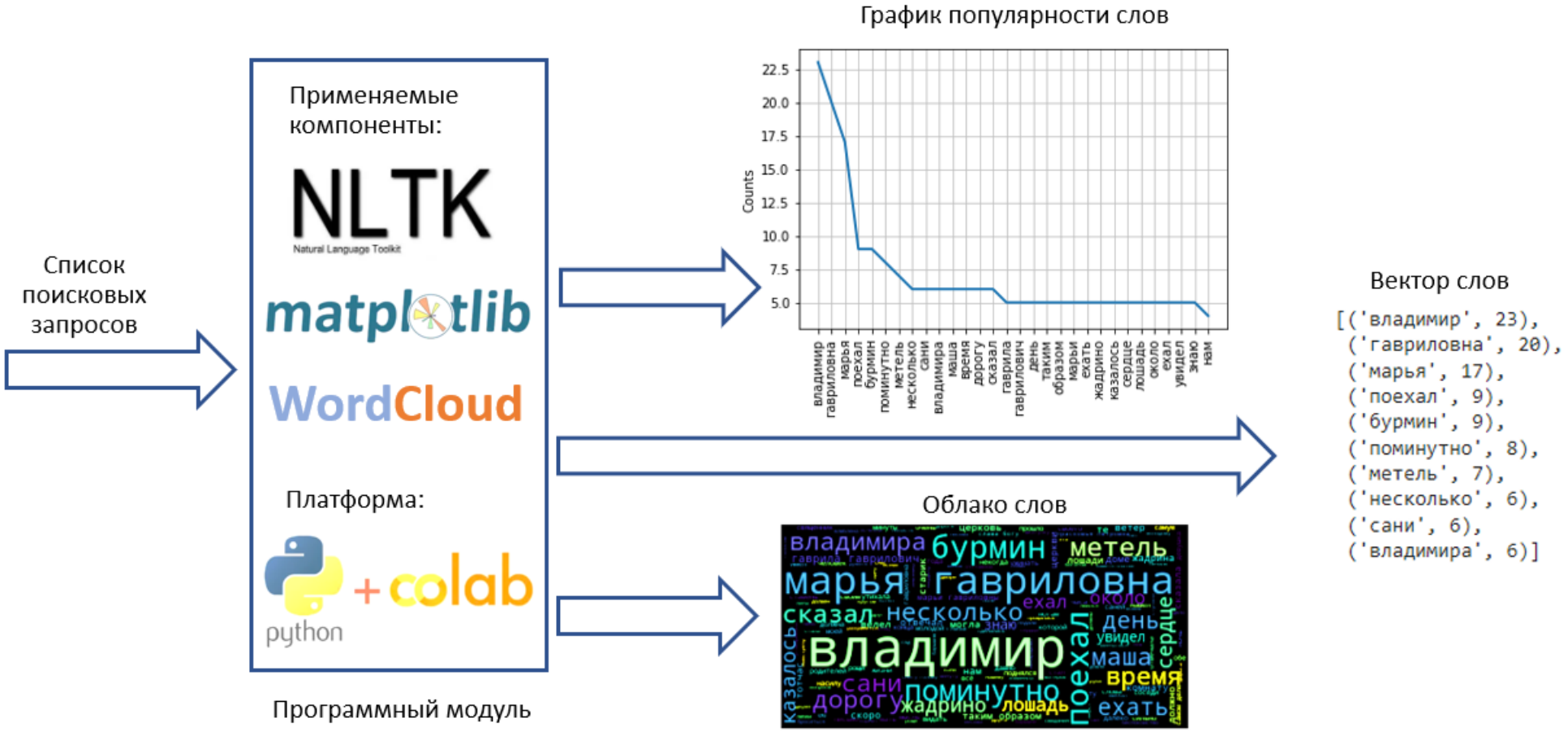



Рисунок 7 – Схема работы программного обеспечения для анализа поисковых запросов пользователей в социальных сетях

Теперь, когда мы выделили функциональную часть программного модуля детально опишем алгоритм анализа текстовых данных.

## 2.2 Детализация алгоритма анализа текстовых данных поисковых запросов

Анализ текстовых данных поисковых запросов осуществляется с использованием следующей последовательности (рисунок 8). Сначала производится загрузка списка поисковых запросов из текстового файла. Результаты загрузки исходных текстовых данных сохраняется в переменную строкового типа.

Затем проводится тонизация данных из строковой переменной. Токенизация – это процесс разделения текстовых данных на составные элементы (токены). В нашем случае в роли токенов выступают отдельные слова в базовой форме. Про токенизации текста производится удаление служебных символов, таких, например, как символ переноса строки, а также знаков препинания. На выходе мы получаем список (вектор), состоящий из отдельных слов в базовой форме.

На следующем этапе производится очистка списка токенов от предлогов, союзов и других вспомогательных слов, которые называются стоп-словами. Стоп-слава встречаются часто, но по отдельности они несут в себе какого-либо смысла.

Затем список токенов, очищенный от стоп-слов подвергается частотному анализу. Для каждого уникального слова подсачивается кое количество раз оно встречалось в поисковых запросах пользователя. В результате мы получаем массив значений, состоящий из пар «токен-частота его появления в запросах пользователя».

Полученный массив используется для построения графика популярности слов, а также для построения облака слов.

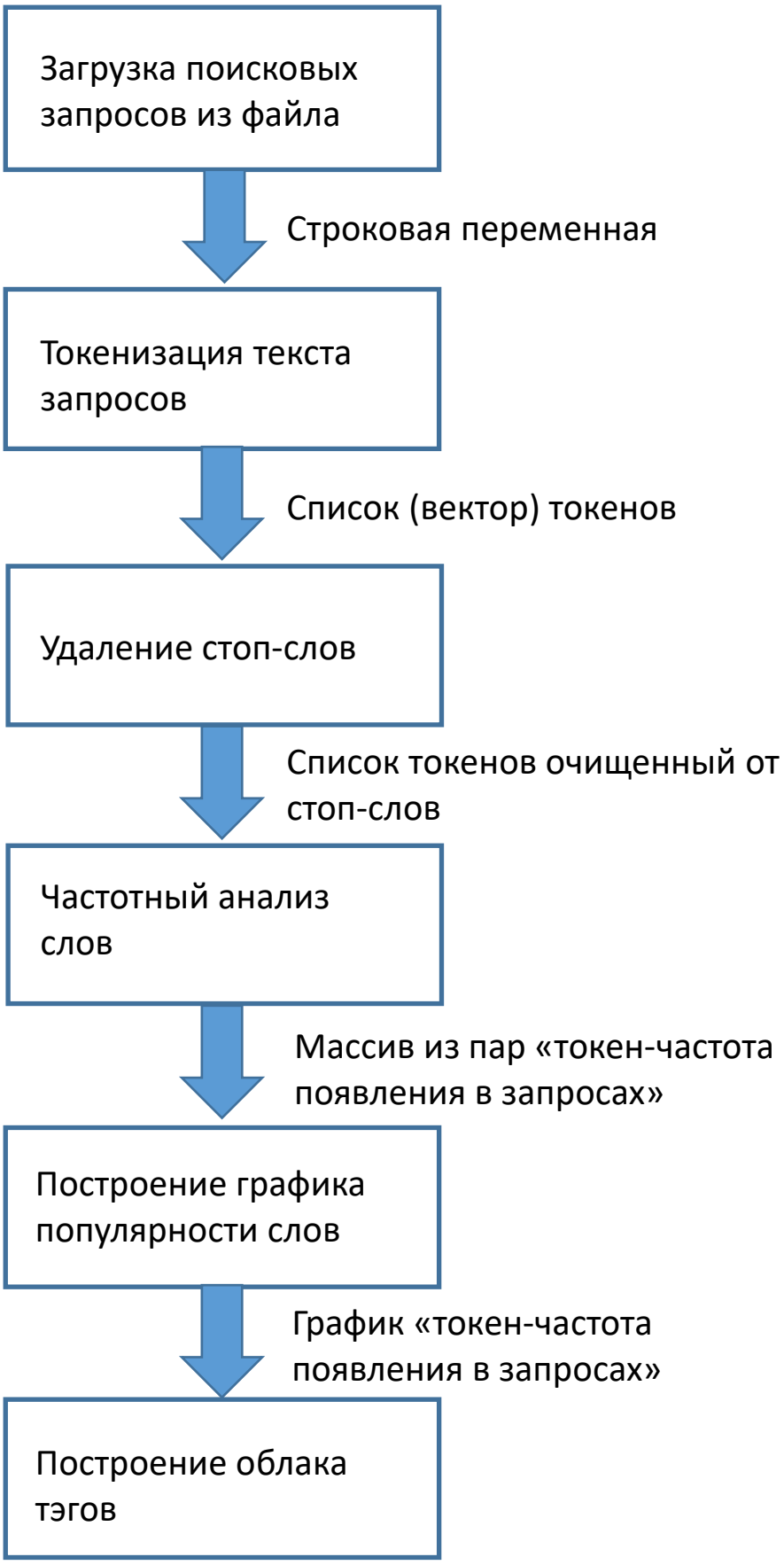



Рисунок 8 – Алгоритм анализа данных

График строится следующем образом. По оси X располагаются токены в порядке снижения частоты их упоминания, а по оси Y откладывается количество упоминания соответствующего токена.

Облако слов строится следующим образом. В прямоугольной области отображаются все встречаемые в запросах пользователя токены, однако, чем чаще встречается токен, тем больший размер шрифта используется для его отображения.

Теперь, когда проведена детализация алгоритма анализа текстовых данных поисковых запросов перейдем разработке программной части информационной системы.

## Выводы по главе 2

Приведем выводы по второй главе бакалаврской работы:

− предложена схема работы программного обеспечения, которая включает в себя: информации об активности пользователя в социальной сети, формирование списка поисковых запросов пользователя, определение интересующих пользователя тем на основе частотного анализа текста запросов;

− предложен алгоритм анализа текстовых данных поисковых запросов для определения, интересующих пользователя, тем, который включает в себя: загрузку поисковых запросов из текстового файла, токенизацию текста запросов, удаление стоп-стоп слов, частотный анализ токенов, построение графика популярности слов, построение облака слов.

# Глава 3 Разработка приложения для анализа поисковых запросов

## 3.1 Особенности реализации программного модуля

Для разработки программного обеспечения был выбран язык программирование Python на основе следующих причин:

- наличие бесплатной библиотеки NLTK реализующей методы по анализу тактовых данных [2];
- возможность разработки кроссплатформенных приложений без необходимости существенного изменения программного кода [3];
- поддержка интерактивных веб-инструментов для разработки скриптов, таких как среда Google colab;
- наличие свободно распространяемой библиотеки wordcloud для визуализации данных.

Разработанное программное обеспечение реализует следующие функциональные возможности:

- загрузка данных поисковых запросов из файла и их обзор;
- предварительный анализ текстовых данных;
- удаление стоп-слов из текста запросов;
- перевод всех слов в основную форму;
- частотный анализ слов;
- визуализация результатов частотного анализа в виде облака.

Рассмотрим особенности реализации программного обеспечения.

Для загрузки данных с запросами пользователя используется стандартный метод open(). В качестве параметров задается: атрибуты открытия файла (“r” – только чтение), кодировка файла и имя файла. Результата чтения файла сохраняется в переменной text (рисунок 9).

```
[ ]  f = open('text.data', "r", encoding="utf-8")
     text = f.read()
```

Рисунок 9 – Программный код для загрузки поисковых запросов пользователя из файла

На следующем шаге проверяется тип загруженных данных из файла. Это необходимо так как в некоторых случаях тип хранящихся данных может при конвертации может распознаться не верно. Проверка типа данных осуществляется с использованием метода type().

Для проверки объема загруженных данных используется метод len(). Для строковых переменных данный метод выводит количество символов в тексте (рисунок 10).

```
[ ]  type(text)

     str

[ ]  len(text)

     22968
```

Рисунок 10 – Программный код для проверки корректности загруженных данных

Для удобства пользователя при обеспечении контроля за загруженными данными на экран выводится первые 300 символов загруженных данных. Это осуществляется с использованием операции среза, как это показано на рисунке 11. Благодаря такому выводу пользователь может понять правильные ли данные загружены, а также оценить наличие специальных символов в текстовой информации.

```
[ ]  text[:300]

     'Метель \n\n\nКони мчатся по буграм, \n\nТопчут снег глубокой
     клоками; \n\nЧерный вран, свистя крылом, \n\nВьется над санями
     гривы\n\n\xa0\xa0\xa0\xa0Жу'
```

Рисунок 11 – Программный код вывода первых 300 символов текста

На следующем шаге осуществляется предварительный анализ текстовых данных. Данный шаг включает в себя очистку текста от знаков пунктуации, специальных символов (таких, например, как символ переноса строки) и лишних пробелов.

Пред очисткой текста осуществляется приведение всех символов текста к нижнему регистру. Этот шаг необходим, для обеспечения правильных результатов частотного анализа, так как одно и тоже слова написанное с использованием символов разного регистра воспринимается как разные строковые значения. Данный шаг выполняется с помощью с помощью встроенного в язык программирования метода lower(). Программный код для приведения всех символов к нижнему регистру показан на рисунке 12.

```
[ ]  # перевод в единый регистр
     text = text.lower()
```

Рисунок 12 – Программный код для приведения всех символов к нижнему регистру

Теперь необходимо очистить текст от знаков пунктуации. Для этого можно воспользоваться стандартным набором символов, хранящегося в строке punctuation. Для получения доступа к данной строке необходимо подключить библиотеку string. Для удобства пользователя на экран выводится информация, о хранящихся в этой строке символах (рисунок 13).

```
[ ]  import string
     string.punctuation

     '!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~'
```

Рисунок 13 – Импортированный набор символов для очистки текстовых данных

Как видно из рисунка 14 хранящаяся в переменной punctuation строка не содержит в себе специальные символы и некоторые часто встречающиеся знаки пунктуации. По этой причине строка punctuation была дополнена с использованием программного кода, приведенного на рисунке 14.

```
[ ]  spec_chars = string.punctuation + '\n\xa0«»\t—…'

[ ]  %%time
     text = "".join([ch for ch in text if ch not in spec_chars])
```

Рисунок 14 – Задание символов для очистки текста

Программный код для очистки исходных данных был обернуть в функцию remove_chars_from_text, где в качестве параметров предаются исходный текст text и символы chars предназначенные для удаления. Так оценки быстродействия программного кода была добавлена инструкция %%time которая измеряет время выполнения программного кода и выводит результат замера в миллисекундах.

Пример вызова функции с использованием переменных text и spec_char показан на рисунке 15.

```
[ ]  def remove_chars_from_text(text, chars):
         return "".join([ch for ch in text if ch not in chars])
```

```
[ ]  %%time
     text = remove_chars_from_text(text, spec_chars)

     Wall time: 4.02 ms
```

Рисунок 15 – Вызов функции для очистки текста

При необходимости можно также использовать очистку текста от цифр. Для этого можно использовать функцию remove_chars_from_text, предав в нее в качестве параметра значение встроенной переменной string.digits. Программный код показан на рисунке 16.

```
%%time
text = remove_chars_from_text(text, string.digits)

Wall time: 2.99 ms
```

Рисунок 16 – Очистка текста от лишних символов

Приведенный программный код для очистки текста можно считать оптимизированным, так как тесты показывают, что выполнение данного этапа на тексте длинной 23000 символов осуществляется за несколько миллисекунд.

На следующем этапе выполняется токенизация текста. Это процедура разделения текста на составные элементы, которыми могут являться, например, слова. Для выполнения процедуры токенизации текста в приложении используется метод word_tokenize(), который храниться в библиотеке nltk.

Результат тонизации текста представляет из себя список, который сохранен в переменную text_tokens. Просмотр содержимого переменной text_tokens осуществляется с использованием операции среза так, как это

показано на рисунке 17.

```
[ ]  from nltk import word_tokenize
     text_tokens = word_tokenize(text)
```

```
[ ]  print(type(text_tokens), len(text_tokens))
     text_tokens[:10]
```

```
<class 'list'> 3402
['метель',
 'кони',
 'мчатся',
 'по',
 'буграм',
 'топчут',
 'снег',
 'глубокой',
 'вот',
 'в']
```

Рисунок 17 – Программный код для токенизации текста

На следующем этапе проводится частотный анализа упоминания токенов в исходных поисковых запросах. Частотный анализ осуществляется подсчетом количества раз упоминания токенов в запросах. При выполнении частотного анализа используется метод FreqDist(), реализованный в библиотеке nltk. Результат подсчета частоты упоминания токенов сохраняется в виде словаря в переменной fdist.

С помощью метода most_common осуществляется вывод наиболее часто встречающихся в поисковых запросах токенов. В качестве входного параметра метода most_common задается количество токенов для которых необходимом вывести результат. Программный код для выполнения частотного анализа представлен на рисунке 18.

```
[ ]  %%time
     from nltk.probability import FreqDist
     fdist = FreqDist(text)
     fdist

     Wall time: 6.98 ms
     FreqDist({'и': 146, 'в': 101, 'не': 69, 'что': 54, 'с': 44, 'он': 42,

[ ]  fdist.most_common(5)

     [('и', 146), ('в', 101), ('не', 69), ('что', 54), ('с', 44)]
```

Рисунок 18 – Программный код для частотного анализа

Для визуализации результатов частотного анализа применяется метод plot, который позволяет в виде графика вывести информацию о наиболее часто встречающихся тегов в поисковых запросах.

У метода plot два входных параметра:

– количество наиболее часто встречающихся токенов, отображаемых на графике;

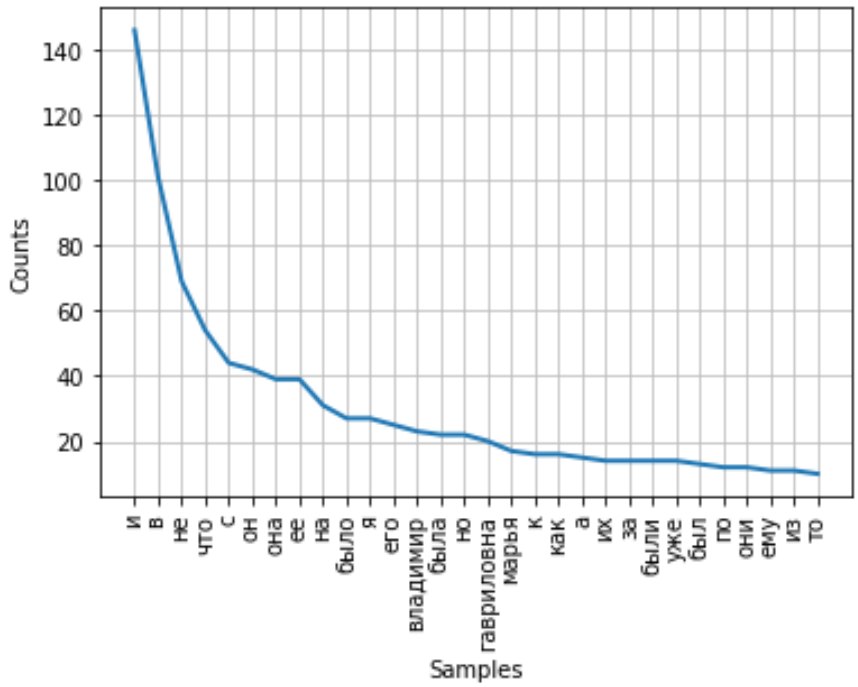
– метод подсчёта частоты - значение cumulative=false означает, что частота каждого токена будет подсчитываться отдельно от других токенов.

График строится следующем образом. По оси X располагаются токены в порядке снижения частоты их упоминания. А по оси Y, именуемой Counts откладывается количество упоминания соответствующего токена. Отложенные значения соединены прямыми линиями.

Программный код и пример визуализации данных частотного анализа показаны на рисунке 19.

Как видно из рисунка, наибольшей частотой обладают союзы и предлоги, которые по отдельности не несут в себе значимости при определении предпочтений пользователя. Поэтому данные слова необходимо отфильтровать. Фильтрация таких малозначимых союзов и предлогов называется удалением стоп-слов.

```
[ ]  fdist.plot(30,cumulative=False)
```




```
<matplotlib.axes._subplots.AxesSubplot at 0x196c076c488>
```

Рисунок 19 – Программный код визуализации результатов частотного анализа

Для удаления стоп-слов необходимо сначала сформировать словарь с этими словами. Стандартный словарь стоп-слов на русском языке храниться в библиотеке nltk.corpus. Воспользовавшись функцией words() можно поместить желаемый список слов в выбранную переменную, например, russian_stopwords. Такой подход позволяет при необходимости расширять список, воспользовавшись методом extend().

Для проверки количества стоп-слов в словаре применяется функция len, которая возвращает длину списка в виде целого числа.

По умолчанию длина списка стоп слов на русском языке, хранящегося в библиотеке nltk.corpus составляет 153 элемента. Програмный код по формированию списка стоп слов показан на рисунке 20.

```
[ ]  from nltk.corpus import stopwords
     russian_stopwords = stopwords.words("russian")
     russian_stopwords.extend(['это', 'нею'])
```

```
[ ]  print(len(russian_stopwords))
     # russian_stopwords
```

```
153
```

Рисунок 20 – Программный код для формирования списка стоп-слов

Для очистки списка токенов от стоп-слов используется программная конструкция, представленная на рисунке 21. В среднем скорость выполнения данного шага занимает не более 10 миллисекунд. Для контроля результатов удаления стоп-слов производится подсчет количества оставшихся токенов. При этом используется функция len, которая определяет длину списка text_tokens. Программный код для выполнения данного шага показан на рисунке 21.

```
[ ]  %%time
     text_tokens = [token.strip() for token in text_tokens if token not in russian_stopwords]
```

```
Wall time: 6.98 ms
```

```
[ ]  print(len(text_tokens))
```

```
2158
```

Рисунок 21 – Программный код для удаления стоп-слов

После очистки списка токенов от стоп-слов, проводится частотный анализ по алгоритму описанному выше. При этом используется метод FreqDist(), реализованный в библиотеке nltk. Результат подсчета частоты упоминания токенов сохраняется в виде словаря в переменной fdist_sw. А с

помощью метода most_common осуществляется вывод наиболее часто встречающихся в поисковых запросах токенов. Программный код и результат частотного анализа после фильтрации стоп-слов показан на рисунке 22.

```
fdist_sw = FreqDist(text)
fdist_sw.most_common(10)
```

```
[('владимир', 23),
 ('гавриловна', 20),
 ('марья', 17),
 ('поехал', 9),
 ('бурмин', 9),
 ('поминутно', 8),
 ('метель', 7),
 ('несколько', 6),
 ('сани', 6),
 ('владимира', 6)]
```

Рисунок 22 – Программный код для частотного анализа

На следующем этапе осуществляется визуализация результатов частотного анализа. При этом будет построено облако слов, где размер каждого слова будет зависеть от частоты его появления в поисковых запросах. Чем чаще встречается слово, тем более крупным шрифтом оно будет написано. Такое представление данных позволяет наглядно определить какими темами интересуется пользователь на основе связанных с ним список поисковых запросов.

Для построения облака тегов применяется библиотека worldcloud, а для вывода сгенерированного изображения облака слов на экран используется библиотека matplotlib.

Для построения облака слов производится предварительная предобработка текста с использованием метода join(). Для генерирование облака слов осуществляется с использованием метода generate(). Программный код по созданию облака слов показан на рисунке 23.

```
[ ] from wordcloud import WordCloud
    import matplotlib.pyplot as plt
    %matplotlib inline
```

```
[ ] text_raw = " ".join(text)
```

```
[ ] wordcloud = WordCloud().generate(text_raw)
```

Рисунок 23 – Программный код для визуализации результатов частотного анализа

Программный код и пример сгенерированного облака слов показаны на рисунке 24.

```
[ ] plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis("off")
    plt.show()
```


Рисунок 24 – Построение облака слов на основе результатов частотного анализа

Теперь проведем тестирование работы приложения на реальных поисковых запросах, сделанных в социальных сетях.

### 3.2. Результаты тестирования приложения

Тестирование приложения производилось на данных своих поисковых запросов, сделанных в течение дня в социальных сетях. Общая длинна собранных поисковых забросов составляет 535 символов.

Примеры поисковых запросов: «Россия новости сегодня», «мобилизация как избежать», «программирование курсы» и т.д.

На этапе токенизации было выделено 60 уникальных токенов. Был проведен частотный анализ токенов до удаления стоп слов-слов. Визуализация частотного анализа представлена на рисунке 25.

После удаления стоп слов количество уникальных токенов снизилось до 50 штук. Визуализация данных частотного анализа после удаления стоп-слов показана на рисунке 26.

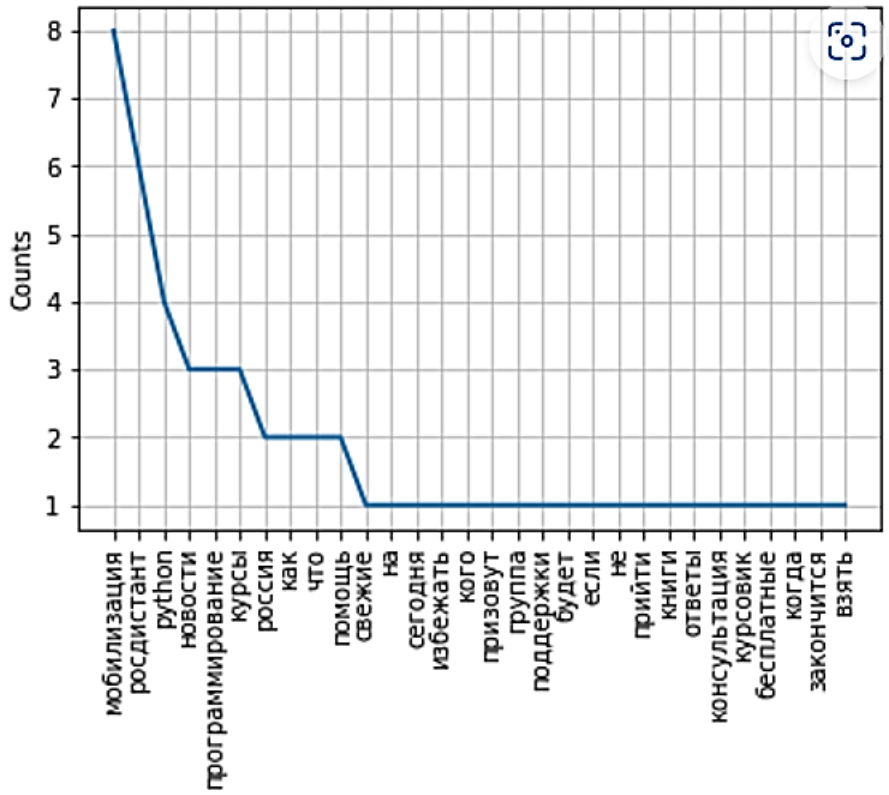



Рисунок 25 – Частотный анализ до удаления стоп-слов

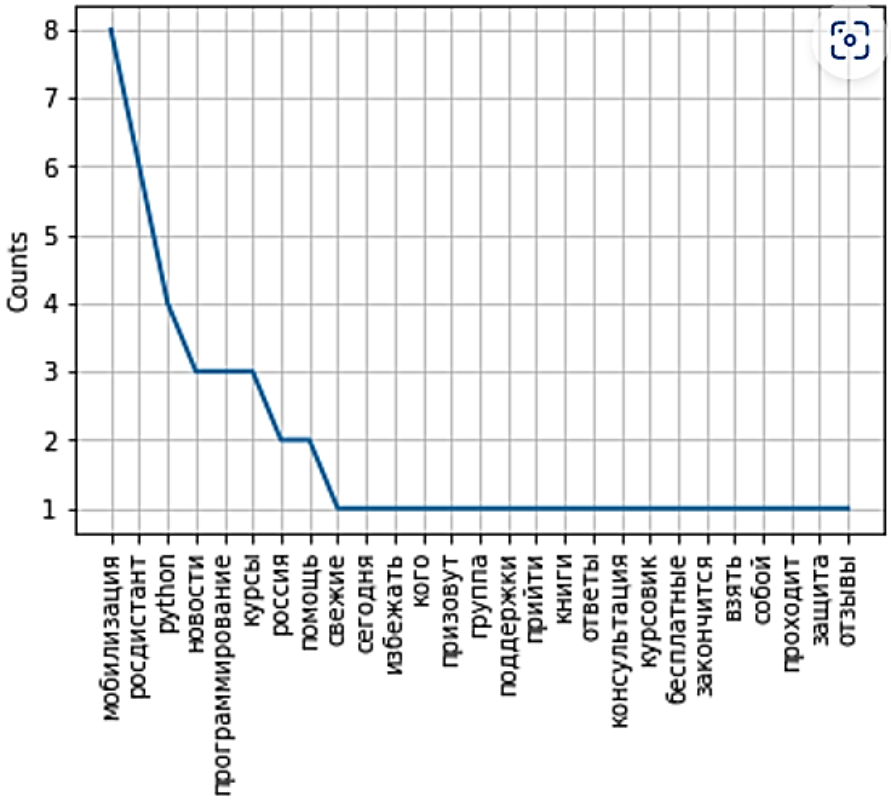



Рисунок 26 – Частотный анализ после удаления стоп-слов

На основе данных частотного анализа приложение было построено облако слов, представленное на рисунке 27.


Рисунок 26 – Визуализация результатов частотного анализа

На основе обзора рисунка 26 можно увидеть, что наиболее важные темами для человека, которому принадлежать поисковые запросы, является мобилизация, Росдистант и программирование.

На основе результатов, полученных в ходе тестирования приложения можно сделать вывод, что оно работает корректно и позволяет получать адекватные результаты анализа поисковых запросов.

## Выводы по главе 3

Приведем выводы по третьей главе бакалаврской работы:

− на языке программирования python разработано приложение, реализующее следующий функционал по анализу текстовых поисковых запросов: загрузка и обзор данных о поисковых запросах, предварительная обработка текстовых запросов, очистка текстовой информации от стоп-слов, трансформация слов в основную форму, частотный анализ слов и визуализация наиболее часто используемых слов в запросах в виде облака.

− разработанное программное обеспечение протестировано на текстовых поисковых запросах, полученный результаты подтверждают работоспособность созданного приложения.

# Заключение

В качестве заключения приведем результаты выполнения бакалаврской работы:

- в ходе анализа литературных данных установлено, что одной важных задач для компаний, владеющих сервисами социальных сетей, является удержание пользователей на своих площадках;
- для удержания внимания пользователей сервисам необходимо уметь определять интересы пользователей с целью предоставления интересующего их контента;
- в рамках бакалаврской работы предложено производить определение интересов пользователя на основе их текстовых поисковых запросов, которые предлагается анализировать с помощью технологий text mining;
- анализ литературных источников показал, что основными этапами text mining является: сбор данных, парсинг текста, фильтрация текста, преобразование пространства признаков и анализ данных.
- с использованием методологии IDEF0 проведено функциональное моделирование процесса «Определение интересов пользователя» в ходе которого определена составные элементы процесс и роль разрабатываемой информационной системы;
- предложена схема работы программного обеспечения, которая включает в себя: информации об активности пользователя в социальной сети, формирование списка поисковых запросов пользователя, определение интересующих пользователя тем на основе частотного анализа текста запросов;
- предложен алгоритм анализа текстовых данных поисковых запросов для определения, интересующих пользователя, тем, который включает в себя: загрузку поисковых запросов из текстового файла,

токенизацию текста запросов, удаление стоп-стоп слов, частотный анализ токенов, построение графика популярности слов, построение облака слов.

− на языке программирования python разработано приложение, реализующее следующий функционал по анализу текстовых поисковых запросов: загрузка и обзор данных о поисковых запросах, предварительная обработка текстовых запросов, очистка текстовой информации от стоп-слов, трансформация слов в основную форму, частотный анализ слов и визуализация наиболее часто используемых слов в запросах в виде облака.

− разработанное программное обеспечение протестировано на текстовых поисковых запросах, полученный результаты подтверждают работоспособность созданного приложения.

## Список используемой литературы и используемых источников


1. Агеев М. С. Автоматическая рубрикация текстов: методы и проблемы / М.С. Агеев, Б.В. Доброе, Н.В. Лукашевич // Ученые записки казанского государственного университета, 2008. – №4. – с. 25-41

2. Григорьев Е.А. Разведочный анализ данных с помощью Python / Григорьев Е.А., Климов Н.С. // E-Scio. 2020. №2 (41). URL: https://cyberleninka.ru/article/n/razvedochnyy-analiz-dannyh-s-pomoschyu-python (дата обращения: 22.09.2022).

3. Гришков, Д.Ю. Язык высокого уровня программирования Python / Гришков Данила Юрьевич, Аусилова Назерке Мырзабековна // НИР/S&R. 2022. №1 (9). URL: https://cyberleninka.ru/article/n/yazyk-vysokogo-urovnya-programmirovaniya-python (дата обращения: 22.09.2022).

4. Ершов, В.Е. Тенденции развития рекламной деятельности в социальных сетях / Ершов Вадим Евгеньевич // Вестник евразийской науки. 2015. №5 (30). URL: https://cyberleninka.ru/article/n/tendentsii-razvitiya-reklamnoy-deyatelnosti-v-sotsialnyh-setyah (дата обращения: 22.09.2022).

5. Корелов, С.В. Предобработка текстов электронных писем в задаче обнаружения спама / С.В. Корелов, А.М. Петров, Л.Ю. Ротков, А.А. Горбунов // Труды учебных заведений связи, 2020. – №4. – с. 80-91

6. Леоненков А. В. Объектно-ориентированный анализ и проектирование с использованием UML и IBM Rational Rose [Электронный ресурс] : учебное пособие. М. : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2020. 317 с. [Электронный ресурс]. URL: https://www.iprbookshop.ru/97554.html (дата обращения: 06.09.2021).

7. Маннинг, К.Д. Введение в информационный поиск / Г Кристофер Д. Маннинг, Прабхакар Рагхаван, Хайнрих Шютце. : Пер. с англ. - М. : ООО “И.Д. Вильямс”, 2014 - 528с.

8. Мкртычев С.В., Гущина О.М., Очеповский А.В. Прикладная информатика. Бакалаврская работа [Электронный ресурс] : электрон. учеб-метод. пособие. Тольятти. ТГУ: Изд-во ТГУ, 2019. 1 оптический диск.

9. Тарасова А.Н. Сентиментальный анализ постов в социальных сетях посредством Python / Тарасова А.Н., Иванов К.О. // Символ науки. 2022. №3-1. URL: https://cyberleninka.ru/article/n/sentimentalnyy-analiz-postov-v-sotsialnyh-setyah-posredstvom-python (дата обращения: 22.09.2022).

10. Чибирова, М.Э. Анализ данных и регрессионное моделирование с применением языков программирования Python и R / Чибирова Марина Эльбрусовна // Научные записки молодых исследователей. 2019. №2. URL: https://cyberleninka.ru/article/n/analiz-dannyh-i-regressionnoe-modelirovanie-s-primeneniem-yazykov-programmirovaniya-python-i-r (дата обращения: 22.09.2022).

11. Amasaki, S. The Effects of Vectorization Methods on Non-Functional Requirements Classification / Sousuke Amasaki, Pattara Leelaprute // 2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 2018. – IEEE, Prague, Czech Republic, 2018. – pp.55-78.

12. Bird, S. Natural Language Processing with Python / Steven Bird, Ewan Klein, Edward Loper. – Published by O'Reilly Media, Inc., 2009. – 502p.

13. Bugueno, M. Learning to combine classifiers outputs with the transformer for text classification / Margarita Bugueno, Marcelo Mendoza // Intelligent Data Analysis, 2020 – № 24. – pp. 15-41

14. Business Process Model and Notation [Электронный ресурс]. URL: https://www.omg.org/spec/BPMN/2.0/About-BPMN/ (дата обращения: 22.08.2021).

15. Gao, G. Research on Routing Selection Algorithm Based on Genetic Algorithm / Guohong Gao, Baojian Zhang, Xueyong Li, Jinna Lv // International Conference on Intelligent Computing and Information Science – International Conference, ICICIS 2011, Chongqing, China, January 8-9, 2011. Proceedings, Part

II: Intelligent Computing and Information Science. – Springer-Verlag Berlin Heidelberg 2011. – pp. 353-358

16. Higuchi, T. Special Section on Nonparametric Approach to Time Series Analysis / Tomoyuki Higuchi, Genshiro Kitagawa // Annals of the Institute of Statistical Mathematics, 2002. - №54 (169). - Springer Nature Switzerland AG 2002. - pp.101-112

17. Jurafsky, D. Speech and Languge Processing: An Introduction to Natural Language Proccerring, Computation Linguistics and Speech Recognition / Jurafsky, Daniel; H. James, Martin. – Stanford University, 2021. – 613 p.

18. Kowsari, K. Text Classification Algorithms: A Survey / Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, Donald Brown // Machine Learning on Scientific Data and Information. – Cornell University, 2019. – pp. 100-168.

19. Srividhya, V. Evaluating Preprocessing Techniques in Text Categorization / V. Srividhya, R. Anitha // International Journal of Computer Science and Application Issue 2010. – pp. 49-51.

20. Sun, C. How to Fine-Tune BERT for Text Classification? / Chi Sun, Xipeng Qiu, Yige Xu, Xuanjing Huang // Computation and Language, 2020. – Cornell University, 2020. – pp. 23-45.