Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное
учреждение высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Институт естественных и точных наук
Факультет математики, механики и компьютерных технологий
Кафедра прикладной математики и программирования
Направление подготовки: 09.03.04 Программная инженерия

РАБОТА ПРОВЕРЕНА
Рецензент,

_____
«____»_____2019г.

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой, д.ф.-м.н., доцент
_____А.А. Замышляева
«____»_____2019 г.

Веб-приложение для сбора и анализа тональности сообщений пользователей в Twitter по заданной теме

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ–09.03.04.2019.96.ПЗ ВКР

Руководитель работы, доцент, к.т.н.
_____/Т.Ю. Оленчикова
«____»_____2019 г.
Автор работы
Студент группы ЕТ-414
_____/Л.С. Елисеев
«____»_____2019 г.
Нормоконтролер, ассистент
_____/Н.С. Мидоночева
«____»_____2019 г.

Челябинск
2019

АННОТАЦИЯ

Елисеев Л.С. Веб-приложение для сбора и анализа тональности сообщений пользователей в Twitter по заданной теме. – Челябинск: ЮУрГУ, ЕТ-414, 71с., 18 ил., 5 табл., библиогр. список – 14 наим., 2 прил.

Выпускная квалификационная работа посвящена разработке веб-приложения, выполняющего анализ тональности сообщений пользователей в сети.

В рамках выполнения работы был выполнен обзор существующих приложений подобного типа, наиболее востребованных средств разработки приложений и систем управления базами данных. Была изучена предметная область и разработана концепция, на основе которой спроектировано и реализовано приложение.

Спроектирована и разработана архитектура системы, включающая в себя диаграмму вариантов использования, диаграмму классов и диаграмму активности. Выполнен анализ предметной области и спроектирована база данных для хранения сообщений пользователей. Выполнена программная реализация системы, ее отладка и тестирование.

Программа реализована на языке программирования JavaScript с использованием СУБД MongoDB. В приложении приведен текст программы.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность темы. За последнее время значительно выросло пользование различными онлайн-ресурсами, в частности, социальными сетями и блогами, такими как Twitter. Большое количество компаний и организаций определяют эти ресурсы как важные для маркетинговых исследований спроса. Чтобы получить обратную связь и понимание того, как покупатели относятся к их продукции, компании проводят интервью, анкетирования и опросы. Это дорогие и длительные методы; кроме того, они не всегда дают нужный результат.

Ежедневно в сеть загружается огромное количество данных, содержащих потребительское мнение. Подобные данные являются, в основном, неструктурированным текстом, из которого компьютеру сложно извлечь мнение потребителя. В прошлом было невозможно обрабатывать такой большой объём неструктурированных данных, но сегодня это не составляет большого труда. Таким образом, обработка естественного языка и анализ тональности играют важнейшую роль в принятии обоснованных решений о маркетинговых стратегиях и дают полезную обратную связь о продуктах и услугах.

Целью выпускной квалификационной работы является разработка интерактивного приложения, направленного на поиск и анализ тональности сообщений пользователей в сети Twitter, с использованием алгоритма нейронной сети. Для достижения поставленной цели необходимо решить следующие задачи:

– выполнить анализ требований к программному обеспечению;

– провести обзор существующих решений для оценки тональности текста, осуществить сравнительный анализ рассмотренных средств и разрабатываемого приложения;

– выбрать платформу, средства и инструменты для создания программного обеспечения;

– спроектировать архитектуру и интерфейс приложения;

– описать основные алгоритмы работы программы;

– разработать ряд тестов для отладки и тестирования системы.

Первая глава посвящена обзору теории, аналогичных решений и формулированию требований к системе. На основе анализа существующих приложений принято решение о построении архитектуры приложения с использованием нейронной сети.

Во второй главе приведена разработка архитектуры системы в целом и базы данных, приведена диаграмма прецедентов.

Третья глава посвящена разработке модулей системы и их описанию.

И, наконец, четвертая глава включает в себя отладку и тестирование приложения с помощью тестовых запросов, приведен пример работающей программы.

1 МЕТОДЫ ОПРЕДЕЛЕНИЯ ТОНАЛЬНОСТИ СООБЩЕНИЙ. АНАЛИЗ ТРЕБОВАНИЙ К ПРИЛОЖЕНИЮ

1.1 Постановка задачи

Поставлена задача разработать программное обеспечение (ПО) для поиска и анализа тональности сообщений пользователей в сети Twitter с использованием клиент-серверной технологии. ПО предназначено для анализа и построения визуального представления процентного соотношения положительных и отрицательных комментариев.

Главной особенностью разрабатываемого ПО является использование алгоритма оценки тональности поста с помощью нейронной сети.

Целями создания ПО являются:

− поиск сообщений в Twitter по тегам или ключевым словам;

− анализ тональности сообщений;

− построение графика пропорциональности.

В качестве пользователей данного ПО могут выступать:

− любые пользователи социальных сетей для поиска всех постов на выбранную тему;

− рекламные агентства для анализа отзывов пользователей о товарах или акциях;

− социологи с целью оценки реакции пользователей на те или иные события в мире;

− SEO-администраторы для повышения эффективности работы механизма продвижения продукта.

1.2 Описание предметной области

1.2.1 Понятие эмоциональной окраски текста

Анализ тональности текста – это автоматическое извлечение его особенностей (эмоционально окрашенной лексики и отношения авторов по

отношению к тому, что сказано в тексте). Анализ тональности можно рассматривать, как метод количественного описания качественных данных, реализуемый путем присваивания некоторых оценок настроения. Хотя тональность в общем случае субъективна, анализ настроений находит много полезных применений. Например, компании получают возможность лучше понять реакцию потребителей на товар, а также могут выявлять негативные комментарии.

Анализ полярности и субъективности являются основными задачами анализа эмоциональной окраски текста. Первый предполагает наличие словаря «хороших» и «плохих» слов. В анализируемом тексте каждому слову присваивается оценка: обычно +1 в случае позитивной тональности, и -1 – в случае негативной, то есть система определяет «позитивный» текст или «негативный». Данный подход имеет ограничения, такие как, пренебрежение контекстом и близлежащими словами. Второй анализ определяет, субъективен текст или нет. Стоит отметить, что под «субъективным» мы подразумеваем тот текст, в котором содержится личное мнение автора, а объективный текст содержит факты. Именно тот текст, в котором выражается мнение автора и подлежит анализу.

При анализе эмоциональной окраски полагают, что текстовая информация в сети Интернет и, в частности, в социальных сетях подразделяется на два класса: мнения и факты. Наиболее значимым понятием является определение мнения. Мнения делятся на два типа: простое мнение и сравнение.

1.2.1.1 Мнение

Простое мнение содержит высказывание автора об одном объекте. Оно может быть высказано прямо: «Меня приятно удивило качество сборки мебели», или неявно: «После курса терапии мое здоровье укрепилось». В обоих случаях простое мнение обычно имеет эмоциональную окраску – положительную или отрицательную.

В анализе тональности текста для мнения первого типа дается формальное определение: простым мнением называется кортеж из пяти элементов (entity, feature, sentiment value, holder, time), где entity – объект, об аспекте (feature) которого автор (holder) высказал мнение в момент времени (time). Выделяются 3 вида эмоций (sentiment_value): позитивная, негативная и нейтральная. Под нейтральной подразумевается, что текст не содержит эмоциональной составляющей.

Объектом является человек, организация, событие, товар или тема обсуждения. Поэтому в различных публикациях объект так же называется object или topic. Часто объект можно представить в виде иерархического дерева компонент и подкомпонент (рисунок 1.1). С каждой компонентой связан набор атрибутов. В приведенном выше определении мнения аспект подразумевает под собой и компоненты, и их атрибуты. Частным случаем аспекта является сам объект.

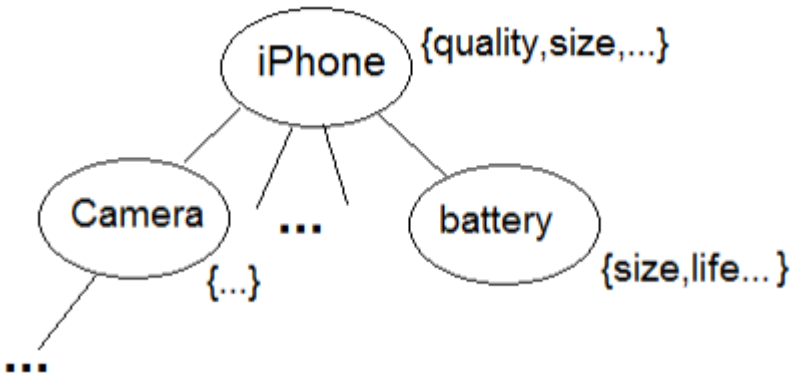


Рисунок 1.1 – Объект обладает компонентами и атрибутами, связанными с ними

Аспект объекта имеет несколько названий: feature, aspect, facet. Вместо «автора мнения» (opinion holder) часто используется термин «источник мнения» (opinion source). Эти различия в терминологии существуют из-за специфичности текстов анализируемых в работах исследователей и не меняют сути определения мнения.

Рисунок 1.2 – Пример эмоционально окрашенного сообщения, содержащего мнение первого типа

Например, в сообщении (рисунок 1.2): «Twitter for iPhone» – объект, «Новый интерфейс» – аспект объекта, «@mgerasimov» – обладатель мнения, «11 Dec» – дата отправки сообщения, «Откровенно разочаровал», «☹» – части сообщения, из которых следует, что мнение об аспекте объекта негативно.

Второй тип мнений – сравнение – можно разделить на три вида.

1. Сравнение аспектов объектов в пользу одного (Non-equal Gradable).

2. Приравнивание аспектов разных объектов (Equative).

3. Превосходство одного объекта над другими (Superlative).

Сравнения первого типа имеют вид «аспект объекта 1 превосходит в чем-то аспект объекта 2», например: «Экран Galaxy Tab сделан более качественно, чем у конкурентов». Второй тип выражает похожесть аспектов разных объектов, например: «И Android, и iOS одинаково удобны для разработки приложений под них». Примером третьего типа может послужить предложение «Аппарат от Canon оказался самым дешевым в том магазине».

Мнение второго типа определяется как кортеж (E1, E2, A, po, holder, time). Где E1 и E2 – множества сравниваемых объектов по аспекту A, po – множество объектов, которые автор (holder) предпочел. Time - момент времени в который мнение было опубликовано.

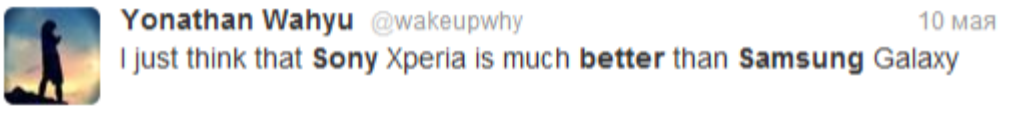


Рисунок 1.3 – Пример сообщения, содержащего мнение второго типа

Например, для сообщения (рисунок 1.3) кортеж мнения выглядит так: ({Sony Xperia}, {Samsung Galaxy}, {общий}, {Sony Xperia}, @wakeupwhy, 10

мая). В отличие от кортежа, определяющего мнение первого типа, кортеж мнения второго типа не содержит оценку эмоций автора.

1.2.1.2 Субъективность

В анализе тональности текста часто встречается термин, связанный с понятием мнения – субъективность.

Объективное предложение выражает фактическую информацию о чем-либо, тогда как субъективное предложение выражает чьи-то личные чувства и предположения.

Предложения, содержащие мнение, обычно являются субъективными, поэтому анализ текста на наличие субъективной информации часто является подзадачей определения полярности текста.

1.2.2 Определение полярности текста

Задача определения полярности текста формулируется следующим образом: «определить, эмоциональная окраска текста положительна или отрицательна?» Определение полярности текста обычно рассматривается на двух уровнях.

1. На уровне документа.
2. На уровне предложения.

1.2.2.1 Определение полярности документа

Задача состоит в определении полярности документа в целом. Причем, текст документа может одновременно содержать предложения, как с негативной, так и с позитивной эмоциональной окраской.

На уровне документа делается предположение о том, что документ содержит мнение об одном объекте. Обзоры товаров и услуг на специализированных ресурсах сети Интернет удовлетворяют этому предположению.

Задаче определения полярности документа посвящено много ранних работ по анализу тональности текста. В большинстве современных систем

анализа мнений, эта задача не рассматривается, т.к. результат – полярность документа в среднем – считается неинформативной оценкой мнения, выраженного в документе.

1.2.2.2 Определение полярности предложений

Задача определения полярности предложений решается как в качестве подзадачи анализа тональности документа, так и в качестве самостоятельной задачи, например, при анализе коротких сообщений и комментариев в социальных сетях.

Часто определение полярности предложения происходит в два этапа. Сначала проводится анализ предложения на субъективность. Если предложение содержит информацию субъективного характера, то, скорее всего, в нем выражено мнение. Далее у субъективного предложения определяется полярность. В противном случае предложение содержит фактическую информацию и далее не рассматривается.

Методы определения полярности отдельных предложений предназначены для более точного анализа мнения в тексте: совместно с методами извлечения аспектов можно провести детальный анализ мнения автора по всем аспектам товара, затронутым в тексте.

1.2.3 Виды классификации полярности

В современных системах автоматического определения эмоциональной оценки текста чаще всего используется одномерное эмотивное пространство: позитив или негатив (хорошо или плохо). Однако могут использоваться и многомерные пространства.

Основной задачей в анализе тональности является классификация полярности данного документа, то есть определение, является ли выраженное мнение в документе или предложении позитивным, негативным или нейтральным. Более развёрнуто, классификация тональности

выражается, например, такими эмоциональными состояниями, как «злой», «грустный» и «счастливый».

Определить полярность текста, значит проанализировать его эмоциональную окраску и определить, положительна она или негативна. В случае если не удается причислить текст или сообщение к одному из типов полярности, его считают нейтрально окрашенным. Полярность текста определяется как на уровне предложения, так и в общем на уровне документа, текста, сообщения и т. д. Для определения полярности текста необходимо решить задачу определения полярности отдельных предложений. Наиболее часто полярность определяется в два этапа: проверка на субъективность и непосредственно определение полярности. Объективное предложение выражает фактическую информацию о каком-либо объекте и не рассматривается с точки зрения определения полярности. Субъективное, напротив, чаще всего содержит мнение автора об объекте, поэтому оно наиболее пригодно для анализа эмоциональной окраски предложения и текста.

Текст, в отличие от предложения, часто может содержать части как с негативной, так и с позитивной тональностью. Поэтому усредненное значение полярности документа в современных системах анализа мнений считается неинформативно оценкой, не позволяющей достоверно утверждать, как именно эмоционально окрашен текст.

1.2.3.1 Классификация по бинарной шкале

Полярность документа можно определять по бинарной в шкале. В этом случае для определения полярности документа используется два класса оценок: позитивная или негативная. Одним из минусов данного подхода является то, что эмоциональную составляющую документа не всегда можно однозначно определить, т.е. документ может содержать признаки как позитивной, так и негативной оценки. Ранние работы в этой области включают в себя труды Терни и Панга, которые применяют различные

методы распознавания полярности обзоров товара и отзывов о фильмах соответственно.

1.2.3.2 Классификация по многополосной шкале

Можно классифицировать полярность документа по многополосной шкале, что было предпринято Пангом и Снайдером (среди прочих). Ими была расширена основная задача классификации киноотзывов от оценки «положительный или отрицательный» в сторону прогнозирования рейтинга по 3-х или 4-балльной шкале. В то же время Снайдер провёл углублённый анализ обзоров ресторанов, предсказывая рейтинги их различных свойств, таких как еда и атмосфера (по 5-балльной шкале).

1.2.3.3 Системы шкалирования

Другим методом определения тональности является использование систем шкалирования, посредством чего словам, обычно связанным с отрицательными, нейтральными или позитивными тональностями, ставятся в соответствии числа по шкале от −10 до 10 (от самого отрицательного к самому положительному). Вначале фрагмент неструктурированного текста исследуется с помощью инструментов и алгоритмов обработки естественного языка, а затем выделенные из этого текста объекты и термины анализируются с целью понимания значения этих слов.

1.2.3.4 Субъективность/объективность

Другое исследовательское направление – это идентификация субъективности/объективности. Эта задача обычно определяется как отнесение данного текста в один из двух классов: субъективный или объективный. Эта проблема иногда может быть более сложной, чем классификация полярности: субъективность слов и фраз может зависеть от их контекста, а объективный документ может содержать в себе субъективные предложения (например, новостная статья, цитирующая мнения людей).

Более того, результаты в большей степени зависят от определения субъективности, употребляющейся в рамках аннотации текстов. Как бы то ни было, Панг показал, что удаление объективных предложений из документа перед классификацией полярности помогло повысить точность результатов.

Модель более подробного анализа называется анализом на основе функции/аспекта. Эта модель ссылается на определение мнений или настроений, выраженных различными функциями или аспектами сущностей, например, у сотового телефона, цифровой камеры или банка. Свойство/аспект – это атрибут или компонент сущности, исследуемой на тональность, например, экран сотового телефона или же качество съёмки камеры. Эта проблема требует решения ряда задач, например, идентификации актуальных сущностей, извлечения их функций/аспектов и определения, является ли мнение, высказанное по каждой функции/аспекту, положительным, отрицательным или нейтральным.

1.2.4 Методы классификации тональности

1.2.4.1 Методы, основанные на правилах и словарях

Этот метод основан на поиске *эмотивной лексики* (лексической тональности) в тексте по заранее составленным тональным словарям и правилам с применением лингвистического анализа. По совокупности найденной эмотивной лексики текст может быть оценен по шкале, содержащей количество негативной и позитивной лексики. Данный метод может использовать как списки правил, подставляемые в регулярные выражения, так и специальные правила соединения тональной лексики внутри предложения. Чтобы проанализировать текст, можно воспользоваться следующим алгоритмом: сначала каждому слову в тексте присвоить его значение тональности из словаря (если оно присутствует в словаре), а затем вычислить общую тональность всего текста путём суммирования значения тональностей каждого отдельного предложения.

Основной проблемой методов, основанных на словарях и правилах, считается трудоёмкость процесса составления словаря. Для того, чтобы получить метод, классифицирующий документ с высокой точностью, термины словаря должны иметь вес, адекватный предметной области документа. Например, слово «огромный» по отношению к объему памяти жёсткого диска является положительной характеристикой, но отрицательной по отношению к размеру мобильного телефона. Поэтому данный метод требует значительных трудозатрат, так как для хорошей работы системы необходимо составить большое количество правил. Существует ряд подходов, позволяющих автоматизировать составление словарей для конкретной предметной области (например, тематика ресторанов или тематика мобильных телефонов).

1.2.4.2 Машинное обучение с учителем

В наше время наиболее часто используемыми в исследованиях методами являются методы на основе машинного обучения с учителем. Сутью таких методов является то, что на первом этапе обучается машинный классификатор (например, байесовский) на заранее размеченных текстах, а затем используют полученную модель при анализе новых документов. Опишем краткий алгоритм.

1. Вначале собирается коллекция документов, на основе которой обучается машинный классификатор.
2. Каждый документ раскладывается в виде вектора признаков (аспектов), по которым он будет исследоваться.
3. Указывается правильный тип тональности для каждого документа.
4. Производится выбор алгоритма классификации и метод для обучения классификатора.
5. Полученную модель используем для определения тональности документов новой коллекции.

1.2.4.3 Машинное обучение без учителя

В основе этого подхода лежит идея, что термины, которые чаще встречаются в этом тексте и в то же время присутствуют в небольшом количестве текстов во всей коллекции, имеют наибольший вес в тексте. Выделив данные термины, а затем определив их тональность, можно сделать вывод о тональности всего текста.

1.2.4.4 Метод, основанный на теоретико-графовых моделях

В основе этого метода используется предположение о том, что не все слова в текстовом корпусе документа равнозначны. Какие-то слова имеют больший вес и сильнее влияют на тональность текста. При использовании этого метода анализ тональности разбивается на несколько этапов.

1. Построение графа на основе исследуемого текста.
2. Ранжирование его вершин.
3. Классификация найденных слов.
4. Вычисление результата.

Для классификации слов используется тональный словарь, в котором каждому слову соотносится оценка, например «положительная», «отрицательная» или «нейтральная». Для получения конечного результата нужно вычислить значения двух оценок: положительной составляющей текста и отрицательной. Для того, чтобы найти положительную составляющую текста необходимо найти сумму тональностей всех положительных терминов текста с учетом их веса. Значение отрицательной составляющей текста находится аналогичным образом. Для итоговой оценки тональности всего текста нужно вычислить отношение этих составляющих по формуле: T = P / N, где T – итоговая оценка тональности, P – оценка положительной составляющей текста и N – негативная составляющая текста. Текст, в котором значение T близко к единице, будет считаться нейтральным, если немного превосходит 1 – положитель-

ным. Если сильно превосходит 1, то сильно положительным. Обратное верно и для текстов отрицательной тональности.

1.2.5 Проблемы в определении тональности сообщения

Текстам в социальных сетях более характерен разговорный стиль речи, нежели литературный. Как следствие, это вызывает серию существенных трудностей при автоматической обработке, так как в разговорном стиле чаще встречаются сленг, фразеологизмы, авторская пунктуация, опечатки и ошибки, а также другие стилистические особенности, которые сложно обрабатывать в автоматическом режиме.

Первой задачей анализа тональности является классификация субъективности. При этом происходит обработка отдельных структурных единиц текста – предложений. Каждое предложение проверяется на наличие в нем субъективного суждения, и в соответствии с результатом, ему присваивается метка наличия или отсутствия субъективности. Как правило, предложения с объективной меткой далее не анализируются, так как они содержат сообщения исключительно информационного характера, то есть описание фактов и событий.

Пример:

– Цукерберг пожертвует 99% акций Facebook на благотворительность.

– Чудесное утро, вкусный кофе, - день обещает быть прекрасным!

Первое предложение будет классифицировано как объективное, так как оно содержит лишь описание события, а второе предложение как субъективное, так как оно отражает эмоциональную оценку начала дня.

Для решения задачи ошибок в тексте и авторских знаков существуют несколько способов решения.

1. В зависимости от анализируемой социальной сети и её норм публикации сообщений, можно принять допущение, что сообщение пользователя состоит из одного предложения, на протяжении которого развивается лишь одна мысль. В таком случае нет необходимости определять

границы предложения, так как весь текст считается одним предложением. Это особенно актуально для сервисов, которые ограничивают длину сообщения (ограничение 140 символов в Twitter).

2. Если принять допущение о том, что реальные пользователи социальной сети публикуют преимущественно субъективное мнение относительно событий и фактов, классификация субъективности может быть пропущена. Следуя данной логике, аккаунты, который публикуют в основном сообщения информационного характера, как правило, являются аккаунтами организаций, либо аккаунтами СМИ и не подлежат анализу.

При наличии субъективного суждения осуществляется анализ эмоциональной окраски. В большинстве случаев прибегают к бинарной классификации сообщений на положительные и отрицательные, но в некоторых случаях может быть использовано более подробное ранжирование. Русский язык богат на речевые средства выразительности, которые непосредственно влияют на эмоциональную окраску передаваемого сообщения.

Одним из наиболее ярких примеров является прямое и переносное значение слова или фразы. Под прямым значением подразумевается первоначальное, исходное значение слова. Как правило, прямое значение является основным наименованием определенного действия, предмета или признака. В свою очередь, переносное значение - это вторичное, дополнительное значение, возникшее на основе прямого по сходству или по смежности. В результате, эмоциональная окраска может меняться на противоположную относительного прямого смысла сообщения. Задачи идентификации вышеперечисленных средств выразительности остаётся крайне актуальной и до конца не решенной задачей. Периодически даже человеку сложно понять, что обозначает определенное сообщение - стоит ли его воспринимать прямо, или в нем есть скрытый смысл. В связи с этим часто принимается допущение, что анализируемое сообщение не содержит переносного смысла.

Отдельное внимание при анализе сообщений из социальных сетей следует уделать эмотиконам - пиктограммам, изображающим эмоцию. Допущение, что тональность сообщения соответствует общей тональности эмотиконов, используемых в нем, может значительно облегчить задачу классификации и увеличить точность. При этом следует учитывать, что проявление и восприятие эмоций в разных частях мира существенно отличается, точно так же, как язык эмотиконов и место их использования. Хэштеги, популярные в сервисах микроблоггинга, так же могут служить идентификатором эмоциональной окраски. Если принять допущение, что тональность твита соответствует тональности хэштегов в нем, то задача классификации заметно упрощается.

В зависимости от специфики анализа, иногда бывает необходимо определить автора высказывания либо, если автор не один, сопоставить каждому высказыванию своего автора. В отношении социальных сетей можно принять допущение, что автором сообщения является владелец аккаунта, от которого было опубликовано сообщение. Однако помимо публикации собственных сообщений, в интернете развита практика цитирования записей другого пользователя. Наиболее распространенным в данном случае решением является принятие допущения, что если пользователь намеренно цитирует другого пользователя, то мнение первого полностью совпадает с мнением второго, следовательно, автором высказывания можно считать процитировавшего пользователя. Некоторые сервисы имеют особые правила упоминания других пользователей в сообщениях, что существенно упрощает их идентификацию. К примеру, в Twitter обращение к другому пользователю начинается со знака «@». Таким образом, задача идентификация сводиться к задаче нахождения слова, начинающегося с «@».

Более глубоким анализом тональности является аспектный анализ тональности, то есть определение отношения к набору характеристик. При

этом сначала в высказывании выделяются объекты, о которых идет речь, а потом определяются их субъективные характеристики.

Пример:

– Планшет лёгкий, по бокам рамки небольшие, а сверху и снизу большие - удобно держать в планшетной ориентации.

– Очки слишком легкие, падают от сильного ветра.

В данном примере объектом высказывания является «планшет», а его тональность складывается из набора характеристик, которые имеют различную полярность (легкий, небольшие рамки, удобно держать). Таким образом, задача сводится к выявлению аспектов объекта речи и выявлении их субъективной оценки. Но стоит учитывать, что одна и та же характеристика может иметь различную эмоциональную окраску для объектов разных типов. К примеру, в сообщениях выше одна и та же характеристика «легкий» имеет позитивный окрас в первом случае и негативный во втором, так как маленький вес для планшета - это хорошо, когда для очков эта же характеристика вызывает дискомфорт в использовании. Хэштеги в социальных сетях сильно упрощают задачу выделение аспектов и их характеристик из предложения, так как в большинстве случаев являются искомыми характеристиками.

1.2.6 Алгоритм поиска в сети “Twitter”

Существует множество способов использования поиска в Твиттере. Можно искать свои твиты, твиты друзей, местных компаний и кого угодно, от знаменитых артистов до политических лидеров мирового уровня. При помощи поиска по ключевым словам или хэштегам можно читать переписки о важных новостях или личных интересах.

Режим безопасного поиска позволяет управлять представлением результатов поиска. С помощью фильтров можно исключить из результатов поиска контент, который может носить деликатный характер, а также учетные записи, которые вы внесли в список игнорируемых или в черный

список. Данную настройку можно отключать и включать в любое время (инструкции приведены ниже).

Если вы выполнили вход в свою учетную запись на веб-сайте, использование поиска немного отличается от поиска без авторизации. Пользователи без авторизации не имеют доступа к просмотру защищенных твитов.

Алгоритм поиска Twitter автоматически фильтрует содержимое поиска и повышает релевантность сообщений, а также исключает спам.

Пользователям, выполнившим авторизацию, доступен так называемый “расширенный поиск”. Он позволяет фильтровать результаты поиска по датам, людям и т. д. С его помощью можно найти определенные твиты.

При помощи расширенного поиска можно ограничить поисковую выдачу при помощи любого сочетания указанных ниже полей.

Слова:

– твиты, содержащие все слова в любом месте («Твиттер» и «поиск») ;

– твиты, содержащие точные фразы («поиск в Твиттере»);

– твиты, содержащие хотя бы одно из слов («Твиттер» или «поиск»);

– твиты, не содержащие определенных слов («Твиттер», но не «поиск»);

– твиты с определенным хэштегом (#Твиттер);

– твиты на определенном языке (твиты на русском).

Люди:

– твиты определенной учетной записи (отправлено @TwitterComms);

– твиты, отправленные в ответ определенной учетной записи (в ответ @TwitterComms);

– твиты, упоминающие определенную учетную запись (твиты, содержащие @TwitterComms).

Места:

– твиты, отправленные из определенного географического местоположения, например, определенного города, штата, страны;

– для выбора географического местоположения используйте раскрывающийся список.

Даты:

– твиты, отправленные до определенной даты, после определенной даты или в указанном диапазоне дат;

– для выбора даты начала, даты окончания или обеих дат используйте раскрывающийся календарь;

– поиск твитов за любую дату с момента публикации первого общедоступного твита.

Комбинации полей расширенного поиска позволяют эффективно управлять поисковой выдачей. Например, можно выполнить поиск твитов, которые содержат слова «Новый год», но не содержат слова «разрешение», за период с 30 декабря 2013 года по 2 января 2014 года, или поиск твитов на русском языке с хэштегом #ЧМ2014, отправленных из Бразилии в июле 2014 года.

С помощью ключевых слов, в Twitter можно выполнять поиск по местоположению, расстоянию и почтовому индексу, фильтрации твитов, которые содержат ссылки, а также отображения постов, написанных с использованием определенного клиента.

1.2.7 Особенности текстов в социальных сетях

1.2.7.1 Смайлы

Для выражения эмоций в тексте пользователи ставят смайлы. Смайл – это набор символов, условно иллюстрирующий выражение лица автора, а точнее его настроение. Все смайлы можно поделить на восточные и западные по географии их использования, последние приведены в таблице 4 с метками, соответствующими их эмоциональной окраске. В случае с короткими

текстами нет более простого способа отметить своё отношение к теме, чем поставить смайл, но не все пользователи так делают, поэтому размечать сообщения с их помощью в общем случае не получится. Есть и более сложные конструкции из скобок, двоеточий и других символов, но они используются не так часто и обычно означают уже не просто отношение, а какие-то действия или объекты, то есть эмоциональной окраски не несут.

Таблица 1.1 – Эмоциональная окраска смайлов

Смайл	Метка	Смайл	Метка	Смайл	Метка	Смайл	Метка	Смайл	Метка
:-)	+	:)	+	:o)	+	:]	+	:3	+
:c)	+	:>	+	=]	+	8)	+	=)	+
:}	+	:^)	+	:>)	+	:-D	+	:D	+
8-D	+	8D	+	x-D	+	xD	+	X-D	+
XD	+	=-D	+	=D	+	=-3	+	=3	+
B^D	+	:-))	+	>:[	-	:-(	-	:(	-
:-c	-	:c	-	:-<	-	:>C	-	:<	-
:-[	-	:[	-	:{	-	;(	-	:-‖	-
:@	-	>:(	-	:'-(	-	:'(	-	:'-)	+
:')	+	D:<	-	D:	-	D8	-	D;	-
D=	-	DX	-	v.v	-	D-‘:	-	:*	+
:^*	+	(	+	}{	+	)	+	;-)	+
;)	+	*-)	+	*)	+	;-]	+	;]	+
;D	+	;^)	+	:-,	+	>:P	+	:-P	+
:P	+	X-P	+	x-p	+	xp	+	XP	+
:-p	+	:p	+	=p	+	:-Þ	+	:Þ	+
:þ	+	:-þ	+	:-b	+	:b	+	d:	+
>:\	-	>:/	-	:-/	-	:-.	-	:/	-
:\	-	=/	-	=\	-	:L	-	=L	-
:S	-	>.<	-	:|	-	:-|	-	:$	-
O:-)	+	0:-3	+	0:3	+	0:-)	+	0:)	+
0;^)	+	O_O	-	ø/	+	<3	+	</3	-

Кроме ASCII смайлов есть ещё и графические – это картинки, которые вставляются в текст. В современных веб-сервисах и мобильных приложениях используется графический язык Emoji для записи слов, эмоций и действий. На рисунке 4 изображены некоторые известные графические смайлы, которые используются в социальной сети Facebook. Обычно для каждого из них есть ASCII аналог, причём не один. Набирая сообщение на клавиатуре компьютера или ноутбука, удобнее поставить двоеточие со скобкой, но смартфоны и планшеты предоставляют все удобства для вставки улыбчивых

картинок: наряду с русской и английской клавиатурой, например, на них можно подключить и клавиатуру графического языка Emoji.

Так как смайлы являются своего рода разметкой сообщений самими пользователями, их необходимо использовать при анализе эмоциональной окраски. В этой работе будет рассмотрено применение символьных и графических улыбок для сбора корпуса твитов и для предобработки данных непосредственно перед классификацией.

Рисунок 1.4 – Некоторые используемые в тексте графические смайлы

1.2.7.2 Хештеги

Ещё одна особенность общения в микроблогах – хештеги. Пользователь помечает в своём сообщении слово, ставя перед ним «#», тем самым показывая связь объекта, обозначаемого этим словом, и всего твита. Платформы для микроблогов предлагают возможность искать по хештегам, выбирать из них популярные и следить за потоками актуальной информации. Многие хештеги используются в течение короткого периода времени, но затем именно по ним можно найти информацию, которая когда-то была актуальной и понадобилась через несколько месяцев. Например, организаторы мероприятий стараются придумывать уникальный хештег, размещать его на информационных стендах, чтобы участники следили за

твитами друг друга и распространяли информацию по всему Интернету. Можно сказать, что это повествовательная функция хештегов, точнее, тех из них, которые указывают на объект, – они могут помочь осуществлять поиск сообщений на определённую тему.

Другую функцию этих специальных слов-ассоциаций можно назвать описательной. Именно такие хештеги можно использовать в определении эмоциональной окраски текстов. Для наглядности в таблице 5 приведены первые пять для каждой эмоции.

Таблица 1.2 – Самые популярные хештеги для пяти чувств: привязянности, ярости, страха, наслаждения и грусти.

Привязанность	Ярость	Страх	Наслаждение	Грусть
#youthebest	#godie	#hatespiders	#thankinggod	#catlady
#yourthebest	#donttalktome	#freakedout	#thankyoulord	#buttrue
#hyc	#fuckyourself	#creepedout	#thankful	#singleprobs
#yourethebest	#getoutofmylife	#sinister	#superexcited	#singleproblems
#alwaysandforever	#irritated	#wimp	#tripleblessed	#lonelytweet

Использование хештегов непосредственно для оценки эмоциональной окраски можно считать примерно таким же, как и у смайлов, но лишь тогда, когда слово однозначно относится либо к положительным, либо к отрицательным. В противном случае они либо становятся обычными словами: без символа «#» они участвуют в классификации наравне с другими, либо уточняют вероятность сообщения попасть в тот или иной класс при помощи подсчёта условных вероятностей, где условием и является хештег.

1.2.7.3 Сокращения, пролонгирования и пунктуация

Тексты в микроблогах содержат не только уточняющую информацию, но и отчасти мешающую. Её нужно научиться использовать, так как специфика сообщений не позволяет хоть что-то выкидывать.

Ограничение в 140 символов заставляет людей сокращать слова, причём как при помощи общеизвестных аббревиатур, например, «СПбГУ» – это Санкт-Петербургский Государственный Университет, так и при помощи

жаргонных конструкций: «h8» – это на самом деле hate. На примере последнего видно, что избавиться от этого слова было бы расточительно, но вряд ли «h8» внесло бы вклад в вероятность сообщения попасть в класс отрицательных такой же, как и слово «hate». Получается, сокращения нужно уметь переводить.

Когда пользователям кажется, что длина сообщения не такая уж и маленькая, они используют пролонгирования гласных – ещё один способ выражать обеспокоенность темой твита. Автор преумножает гласную в слове, изображая её продолжительное звучание, то есть, например, крик. Так «nooooooo» будет, скорее всего, означать категорическое несогласие, а «so cuuuute» – умиление. Таким образом, каждое такое слово что-то значит, но классификатор может об этом не знать, значит, нужно рассказывать классификатору какими-то другими способами, что это важное слово и какое из известных является его менее эмоциональным аналогом.

Авторская пунктуация может рассказать об эмоциональной окраске сообщения не меньше, чем смайлы. Например, в нейтральных твитах крайне редко встречаются восклицательные знаки. Впрочем, однозначно классифицирующих особенностей пунктуации не так и много: наличие восклицательных знаков указывает на наличие эмоциональной окраски, при этом нельзя без дополнительного анализа сказать, какой именно; сочетание «?!», скорее всего, будет означать недоумение, то есть классифицируется как отрицательное; многоточия обычно говорят о нейтральности.

1.2.8 Использование особенностей текстов для предобработки

Смайлы, хештеги, сокращения, пролонгирования и пунктуация – это то, про что классификатор уже не знает, то есть перед подачей ему сообщения необходимо преобразовать это сообщение так, чтобы все перечисленные особенности не выбивались и превратились в обычные слова.

Смайлы, перечисленные в таблице 4, заменяются в тексте на соответствующую им метку. Это делается для того, чтобы в обучающей

выборке слово «+» встретилось больше раз среди положительных твитов, тем самым, в вероятность попасть в класс положительных «+» даст больший вклад, чем просто «:)». Так же, заменой на «+» и «−», обрабатывается пунктуация.

К смайлам, заменяемым на метки, добавляется замена некоторых однозначно классифицирующихся хештегов. Происходит это по той же причине, что и со смайлами. Если замена хештега не произошла, то считается, что он должен стать обычным словом, то есть «#» из начала пропадает, и дальше работа происходит уже без учёта того, что это хештег.

Если в сообщении встречается неизвестное слово, его стоит проверить на наличие в словаре сокращений. Запрос к словарю происходит в онлайн-режиме, и для обработки сокращений нужно подключение к Интернету.

Повторения гласных убирать совсем не нужно: достаточно сократить количество повторяющихся гласных до двух, то есть «nooooooo» заменится на «noo». В этом случае слово «noo» может встретиться в обучающей выборке, в отличие от «nooooooo», где именно семь, а не восемь или девять букв «o». Таким образом слово «no» уже не то же самое, что «noo», но все, сколько угодно длинные продолжения гласной «o» сведутся к одному и тому же эмоциональному «noo», которое даст каждому из таких слов с продолжениями одинаковый повод попасть в класс «−».

1.3. Анализ требований к ПО.

Поставлена задача разработать ПО для поиска и анализа сообщений пользователей в сети Twitter. Целями создания ПО являются:

− гибкий поиск сообщений в социальной сети с указанием параметров;

− анализ тональности сообщений;

− построение графического изображения анализа сообщений и их процентное соотношение.

1.4. Выбор среды разработки

В качестве программной платформы для построения серверной части приложения была выбрана Node.js, основанная на транслирующем код JavaScript движке V8. С использованием этой платформы доступна обширная библиотека API, реализуемая в сторонних подключаемых модулях. В частности, в разрабатываемом ПО используется модуль twitter - асинхронная клиентская библиотека для REST и Streaming API сети Twitter.

1.4.1 REST-архитектура приложения

REST (Representational state transfer) – это стиль архитектуры программного обеспечения для распределенных систем, таких как World Wide Web, который, как правило, используется для построения веб-служб. Термин REST был введен в 2000 году Роем Филдингом, одним из авторов HTTP-протокола. Системы, поддерживающие REST, называются RESTful-системами.

В общем случае REST является очень простым интерфейсом управления информацией без использования каких-то дополнительных внутренних прослоек. Каждая единица информации однозначно определяется глобальным идентификатором, таким как URL. Каждая URL в свою очередь имеет строго заданный формат.

Свойства архитектуры, которые зависят от ограничений, наложенных на REST-системы:

– производительность – взаимодействие компонентов системы может являться доминирующим фактором производительности и эффективности сети с точки зрения пользователя;

– масштабируемость для обеспечения большого числа компонентов и взаимодействий компонентов.

Рой Филдинг – один из главных авторов спецификации протокола HTTP, описывает влияние архитектуры REST на масштабируемость следующим образом:

– простота унифицированного интерфейса;

– открытость компонентов к возможным изменениям для удовлетворения изменяющихся потребностей (даже при работающем приложении);

– прозрачность связей между компонентами системы для сервисных служб;

– переносимость компонентов системы путем перемещения программного кода вместе с данными;

– надёжность, выражающаяся в устойчивости к отказам на уровне системы при наличии отказов отдельных компонентов, соединений или данных.

Преимущества:

– надёжность (за счёт отсутствия необходимости сохранять информацию о состоянии клиента, которая может быть утеряна);

– производительность (за счёт использования кэша);

– масштабируемость;

– прозрачность системы взаимодействия (особенно необходимая для приложений обслуживания сети);

– простота интерфейсов;

– портативность компонентов;

– лёгкость внесения изменений;

– способность эволюционировать, приспосабливаясь к новым требованиям (на примере Всемирной паутины).

1.4.2 Архитектура «клиент-сервер»

Клиент-сервер (англ. *Client-server*) – вычислительная или сетевая архитектура, в которой задания или сетевая нагрузка распределены между поставщиками услуг (сервисов), называемыми серверами, и заказчиками услуг, называемыми клиентами. Нередко клиенты и серверы взаимодействуют через компьютерную сеть и могут быть как различными физическими устройствами, так и программным обеспечением.

Двухзвенная архитектура

В любой сети (даже одноранговой), построенной на современных сетевых технологиях, присутствуют элементы клиент-серверного взаимодействия, чаще всего на основе двухзвенной архитектуры. Двухзвенной (two-tier, 2-tier) она называется из-за необходимости распределения трех базовых компонентов между двумя узлами (клиентом и сервером).

Двухзвенная архитектура используется в клиент-серверных системах, где сервер отвечает на клиентские запросы напрямую и в полном объеме, при этом используя только собственные ресурсы. Т.е. сервер не вызывает сторонние сетевые приложения и не обращается к сторонним ресурсам для выполнения какой-либо части запроса.

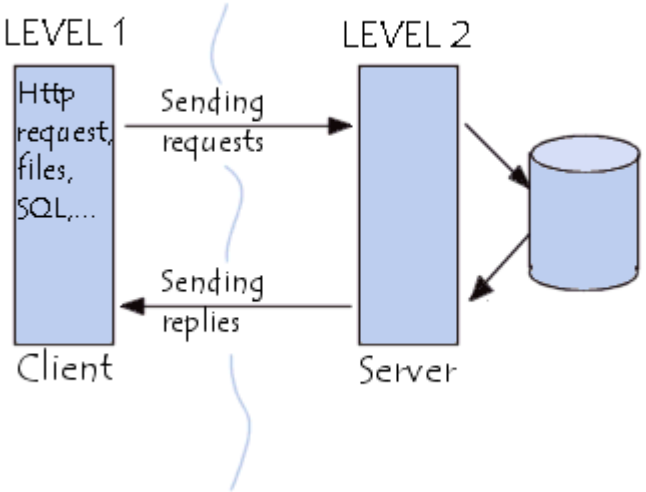


Рисунок 1.5 – Двухзвенная архитектура “клиент-сервер”

1.4.3 Многоуровневая архитектура клиент-сервер

Многоуровневая архитектура клиент-сервер – разновидность архитектуры клиент-сервер, в которой функция обработки данных вынесена на один или несколько отдельных серверов. Это позволяет разделить функции хранения, обработки и представления данных для более эффективного использования возможностей серверов и клиентов.

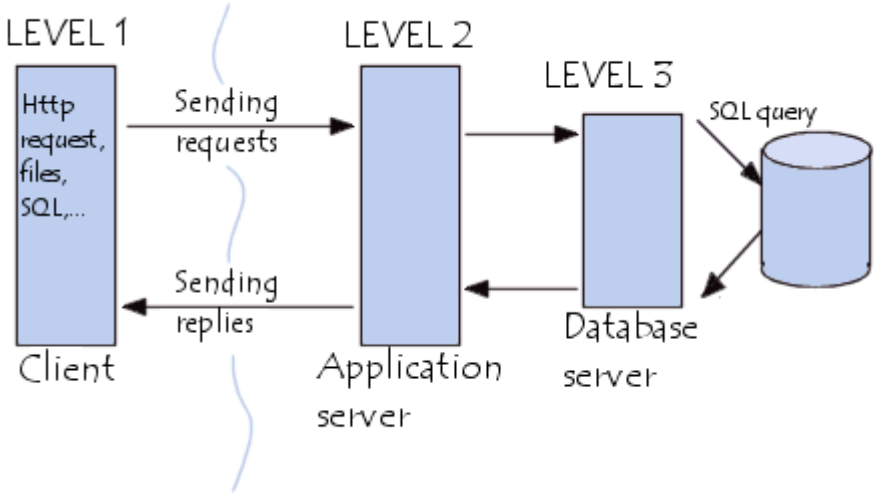


Рисунок 1.6 – Многоуровневая архитектура “клиент-сервер”

1.4.4 Сравнение архитектур

Двухзвенная архитектура проще, так как все запросы обслуживаются одним сервером, но именно из-за этого она менее надежна и предъявляет повышенные требования к производительности сервера.

Трехзвенная архитектура сложнее, но благодаря тому, что функции распределены между серверами второго и третьего уровня. Описание архитектуры приведено ниже.

1. Высокая степень гибкости и масштабируемости.

2. Высокая безопасность (т.к. защиту можно определить для каждого сервиса или уровня).

3. Высокая производительность (т.к. задачи распределены между серверами).

Преимущества клиент-серверной архитектуры:

– отсутствие дублирования кода программы-сервера программами-клиентами;

– так как все вычисления выполняются на сервере, то требования к компьютерам, на которых установлен клиент, снижаются;

– все данные хранятся на сервере, который, как правило, защищён гораздо лучше большинства клиентов. На сервере проще обеспечить контроль полномочий, чтобы разрешать доступ к данным только клиентам соответствующими правами доступа;

– позволяет объединить различные клиенты. Использовать ресурсы одного сервера часто могут клиенты с разными аппаратными платформами, операционными системами и т. п.;

– позволяет разгрузить сети за счёт того, что между сервером и клиентом передаются небольшие порции данных.

Недостатки:

– неработоспособность сервера может сделать неработоспособной всю вычислительную сеть. Неработоспособным сервером следует считать сервер, производительности которого не хватает на обслуживание всех клиентов, а также сервер, находящийся на ремонте, профилактике и т. п.;

– поддержка работы данной системы требует отдельного специалиста – системного администратора;

– высокая стоимость оборудования.

Инструментарий разработки

Node.js

Node.js – программная платформа, основанная на движке V8 (транслирующем JavaScript в машинный код), превращающая JavaScript из узкоспециализированного языка в язык общего назначения. Node.js добавляет возможность JavaScript взаимодействовать с устройствами ввода-вывода через свой API (написанный на C++), подключать другие внешние библиотеки, написанные на разных языках, обеспечивая вызовы к ним из

JavaScript-кода. Node.js применяется преимущественно на сервере, выполняя роль веб-сервера, но есть возможность разрабатывать на Node.js и десктопные оконные приложения (при помощи NW.js, AppJS или Electron для Linux, Windows и macOS) и даже программировать микроконтроллеры (например, tessel и espruino). В основе Node.js лежит событийно-ориентированное и асинхронное (или реактивное) программирование с неблокирующим вводом/выводом.

Преимуществом использования Node.js в проекте является обширная библиотека дополнительных модулей сторонних разработчиков.

Twitter API

Свободно распространяемая стороняя библиотека с возможностью выполнения GET и POST запросов и поддержкой потокового обмена данными.

1.5. Анализ существующих программных решений.

Azure Stream Analytics

Косвенным аналогом является сервис Azure Stream Analytics - модуль обработки событий, который позволяет проверять и анализировать большие потоки данных. С его помощью приложение может оценивать тональность сообщений в режиме реального времени, однако для этого требуется подписка Azure и авторизация в сети Twitter.

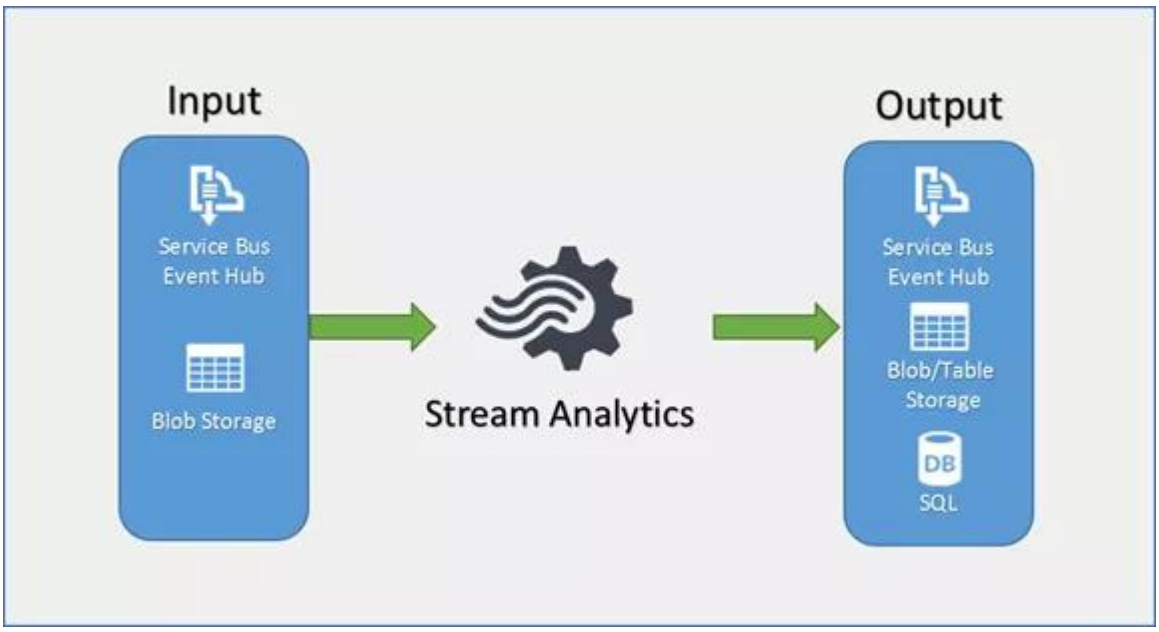


Рисунок 1.7 – Схема работы алгоритма Azure Stream Analytics

YouScan

Решение для мониторинга социальных медиа. Позволяет мониторить блоги, форумы, все соцсети, «отзывники», онлайн-СМИ. В YouScan есть уникальная для СНГ функция – возможность отслеживания упоминаний по логотипу бренда (Visual Insights) и по ситуации потребления продуктов бренда. Поддержка, база знаний, продвинутая фильтрация, отслеживание спама.

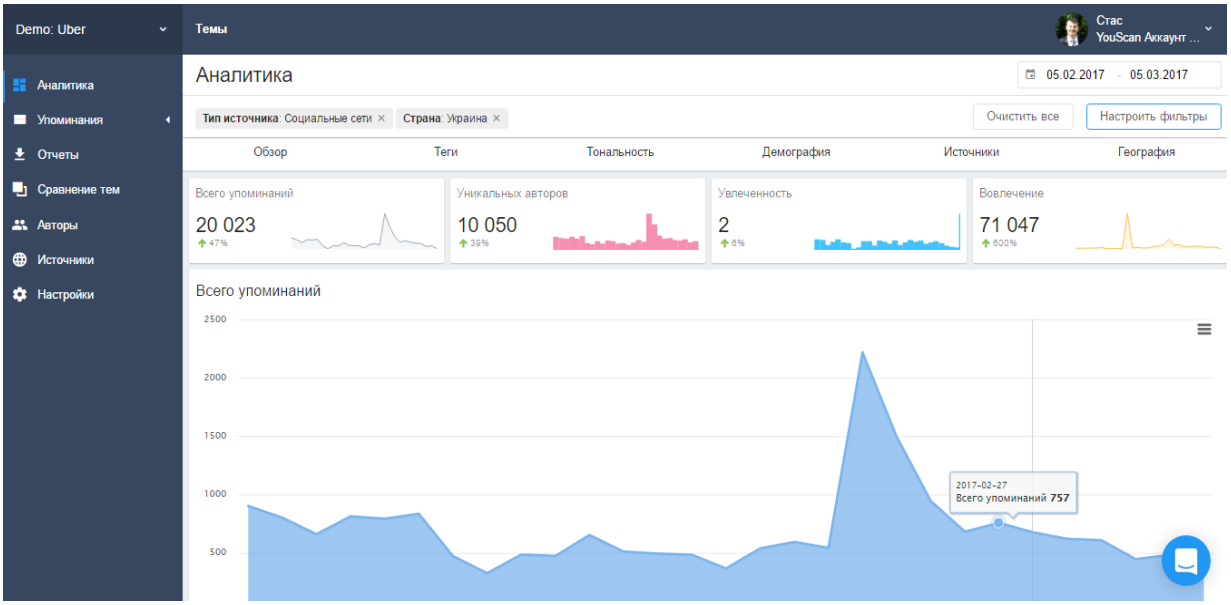


Рисунок 1.8 – Сервис YouScan

Функционал сервиса:

– мультимониторинг – любая аналитика в разрезе любого фильтра;

– автоматическое выявление горячих трендов;

– аналитика по авторам поможет найти лидеров мнений;

– находите важные площадки, благодаря аналитике по источникам.

Преимущества:

– поддержка всех популярных социальных сетей;

– современные облачные технологии;

– аналитические инструменты;

– возможность распределения заданий, например, ответов на негативные отзывы.

Недостатки:

– высокая стоимость;

– большая загруженность сторонними данными.

IQBuzz

Сканер социальных сетей. Обрабатывает информацию из Facebook, Twitter, ВКонтакте, LiveJournal, LiveInternet, Youtube и множества других источников. Он также имеет функции для коллективной работы, умеет автоматически определять позитивные и негативные сообщения, контролировать дубликаты сообщений (это полезно, т.к. часто встречаются перепосты и ретвиты), предоставляет мощный поиск по истории сообщений.

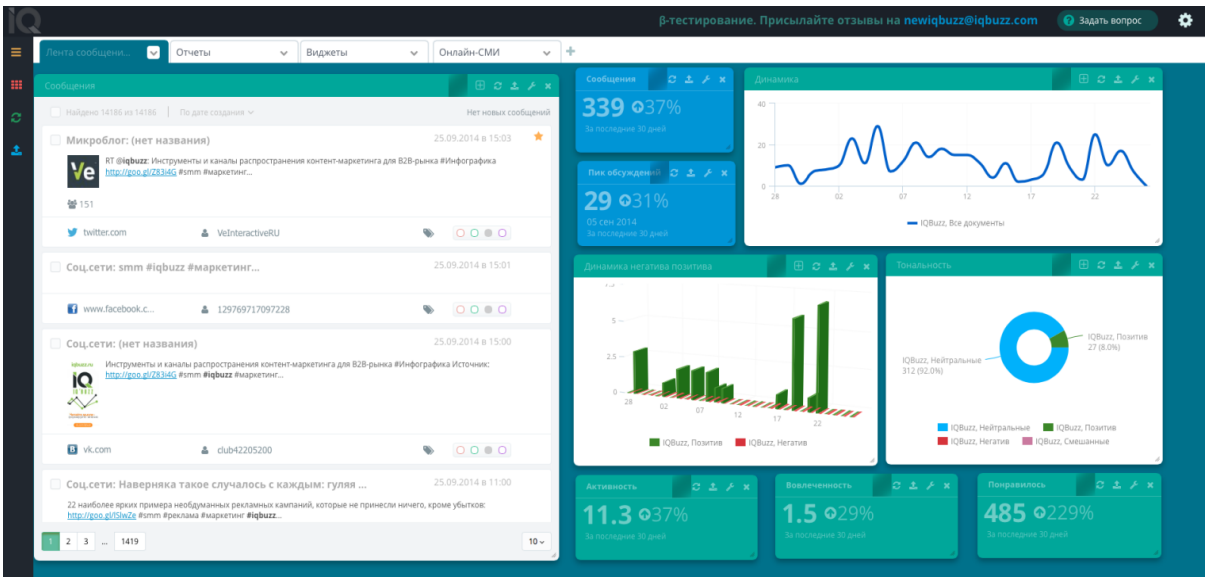


Рисунок 1.9 – Сервис IQBuzz

Преимущества:

– аналитический центр исследований;

– выгрузка отчетов из системы в текстовые редакторы.

Недостатки:

– ограничение по количеству тем для поиска.

Brand Analytics

Система анализа бренда в социальных медиа. Отслеживает упоминания в социальных сетях, блогах, форумах, сайтах отзывов, месенджерах, а также онлайн СМИ. Высокая степень автоматизации – определение тональности с точностью 90%, автоматическое тегирование, фильтрация спама и нерелевантных сообщений, оповещение об угрозе для репутации, персонализированные отчеты.

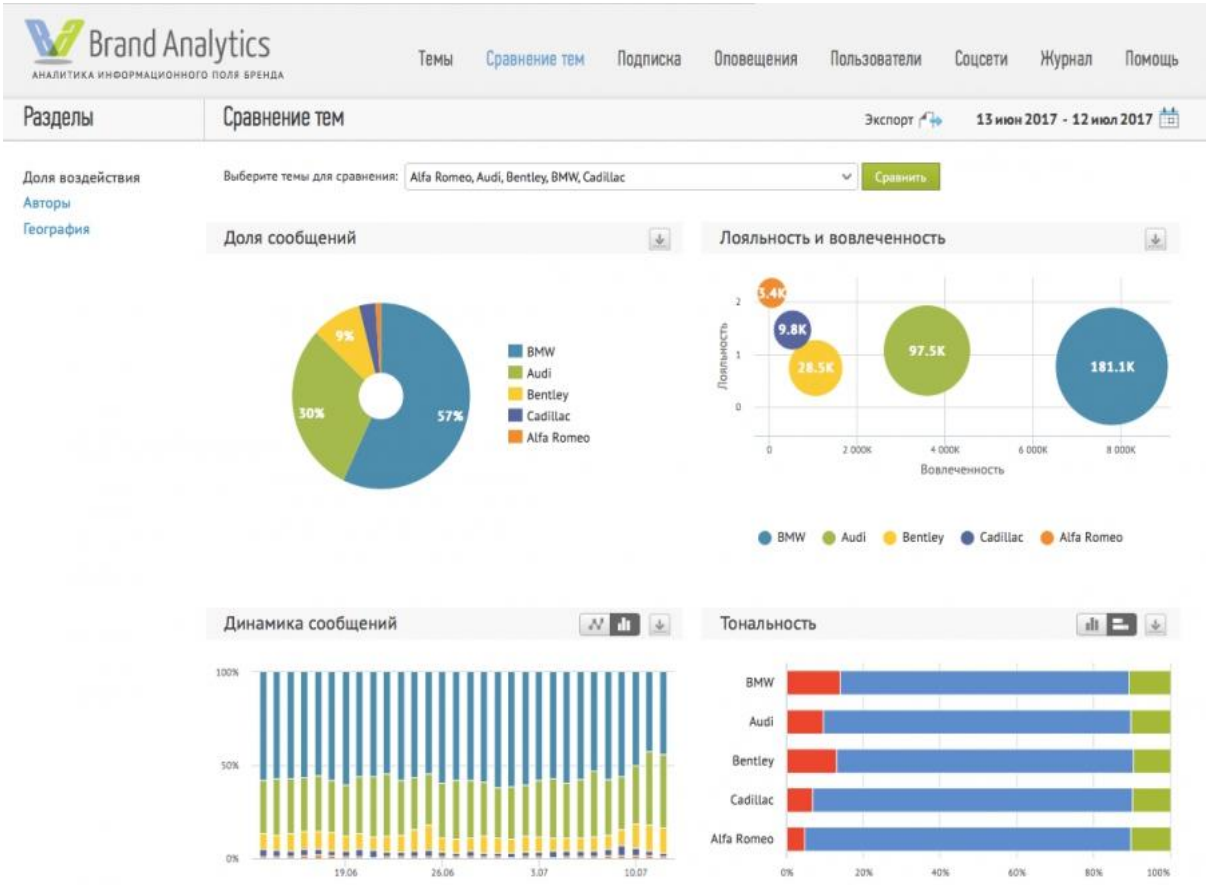


Рисунок 1.10 – Сервис Brand Analytics

Преимущества:

– самообучаемость модуля;

– обработка сленга, используемого в социальных сетях;

– собственный поисковый механизм.

Недостатки:

– высокая стоимость;

– ограниченный набор источников для анализа данных.

1.6 Постановка задачи

Цель работы: разработать приложение для поиска и анализа сообщений пользователей в Twitter по заданной теме.

Отличием от существующих аналогов является использование алгоритма самообучающейся нейронной сети.

Для достижения поставленной задачи были выполнены шаги, приведенные ниже.

1.Разработать клиент-серверную архитектуру приложения

2.Разработать алгоритм обмена информацией с базой данных.

3.Разработать архитектуру сервера базы данных.

4.Реализовать разработанные алгоритмы, клиент-серверную часть приложения.

5.Реализовать базу данных.

6.Развертка приложения на сервере.

7.Отладка и тестирование.

1.7 Выводы по разделу

На данном этапе работы проведён анализ предметной области, определена значимость анализа сообщений пользователей в социальных сетях, а также существующие методы определения тональности. Выполнен обзор аналогов и выделены положительные стороны каждой разработки, что позволило выстроить общий вектор дальнейшего развития проекта.

2 РАЗРАБОТКА АРХИТЕКТУРЫ СИСТЕМЫ

2.1 Диаграмма прецедентов

Для построения диаграмм архитектуры приложения использовался язык графического описания UML (Unified Model Language). Данный язык очень удобен для представления общих понятий, таких как, класс, обобщение, поведение.

На рисунке 2.1 приведена диаграмма вариантов использования (или диаграмма прецедентов) системы в целом, раскрывающая, какие взаимодействия и типы пользователей предусмотрены в системе.

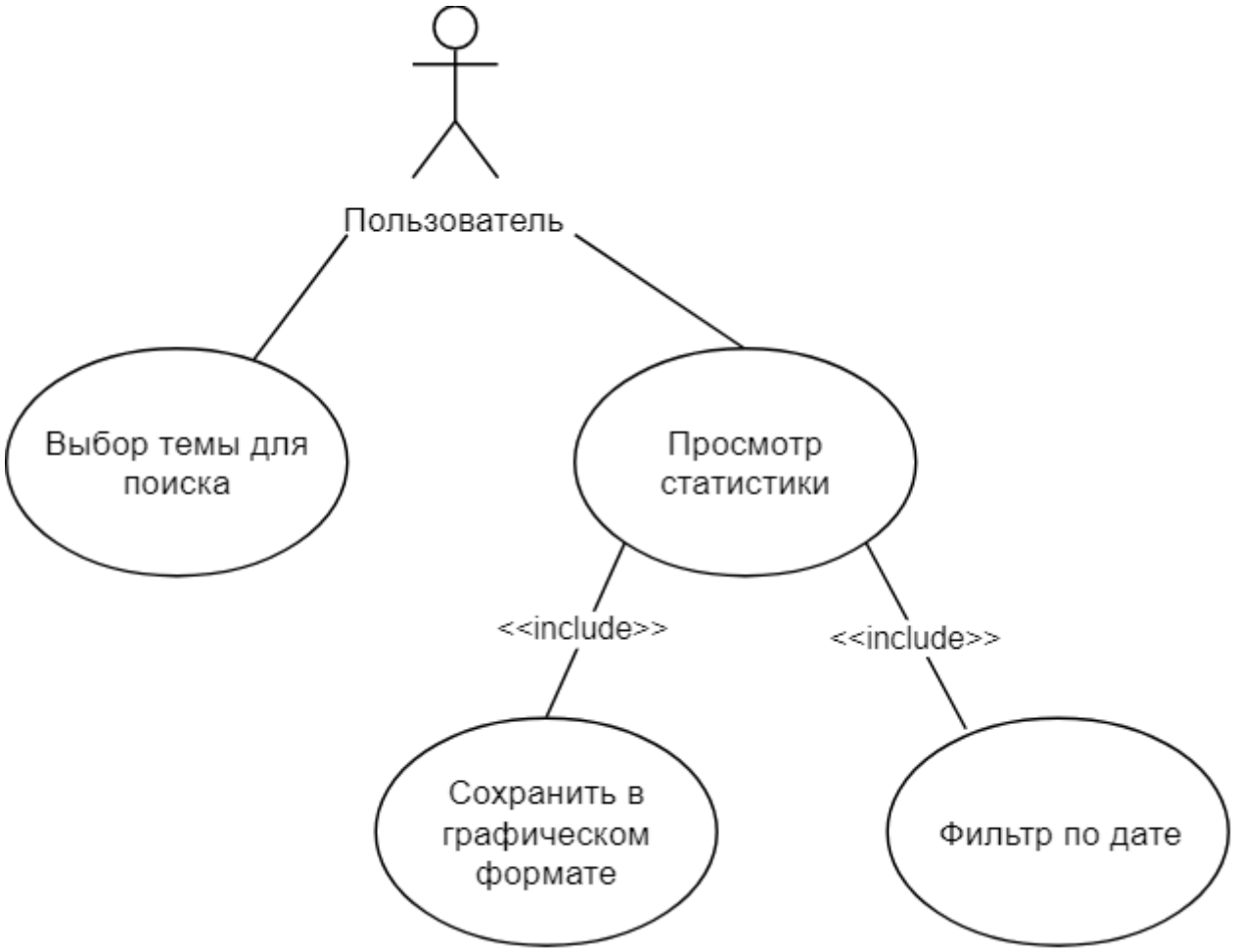


Рисунок 2.1 – Диаграмма прецедентов системы в целом

Далее следует описание всех возможных событий для каждого варианта использования.

2.1.1 Вариант использования: «Выбор темы для поиска»

Для поиска сообщений в сети Twitter с возможностью анализа тональности и построения графика пользователю необходимо ввести тему в соответствующее поле.

Основной вариант использования.

Пользователь вводит название темы для поиска в форме на странице и подтверждает ввод. Система выполняет поиск и анализ и выводит статистику.

Альтернативный вариант использования.

Пустое поле запроса. Система выдает сообщение о том, что необходимо ввести тему для поиска, и предлагает повторить ввод.

2.1.2 Вариант использования: «Просмотр статистики»

Пользователь может просматривать статистику, которую приложение собирает, анализируя тональность сообщений по заданной пользователем теме, и выводит в виде графического представления.

2.1.3 Вариант использования: «Фильтр по дате»

Пользователь может фильтровать найденные сообщения по определенной дате, вводя начальную и конечную дату в поле.

Основной вариант использования.

Пользователь вводит начальную и конечную дату в форме на странице и подтверждает ввод. Система выполняет фильтр сообщений по указанному промежутку и обновляет статистику.

Альтернативный вариант использования.

Пользователь оставляет поля пустыми и нажимает ввод. Система отменяет фильтр, если он есть, и обновляет статистику.

2.1.4 Вариант использования: «Сохранить в графическом формате»

Пользователь может сохранить график анализа тональности в одном из предложенных форматов.

Основной вариант использования.

Пользователь выбирает формат изображения для сохранения и нажимает ввод. Система сохраняет изображение на жесткий диск пользователя.

2.2 Разработка базы данных

На рисунке 2.2 приведена схема базы данных для разрабатываемой системы. База данных предназначена для хранения всей долговременной информации системы: сообщения пользователей, оценки их тональности, коэффициенты для используемой нейронной сети.

Описание назначения и свойств полей базы данных приведено в таблицах 2.1-2.2.

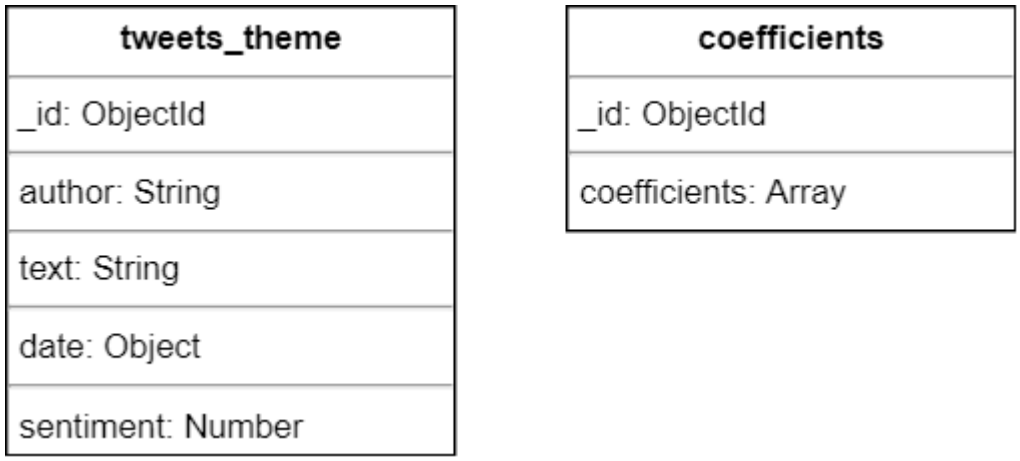


Рисунок 2.2 – Схема базы данных

Таблица 2.1 – Сообщения пользователей (Tweets-theme)

	Тип данных	Значение по умолчанию	Обязательность	Первичный ключ	Внешний ключ	Ограничения
_id	ObjectId	-	+	+	-	уникальный
author	String	-	+	-	-	-
text	String	-	+	-	-	-
date	Object	-	+	-	-	-
sentiment	Number	-	+	-	-	-

В данной таблице хранятся сообщения пользователей по заданной теме, которая является именем таблицы. Каждой записи присваивается уникальный ключ (_id).

Запись состоит из имени автора (author), текста сообщения (text), даты сообщения (date) и оценки тональности (sentiment).

Таблица 2.2 – Коэффициенты для нейронной сети (Coefficients)

	Тип данных	Значение по умолчанию	Обязательность	Первичный ключ	Внешний ключ	Ограничения
_id	ObjectId	-	+	+	-	уникальный
coefficients	Array	-	+	-	-	-

В данной таблице хранятся коэффициенты для используемой нейронной сети в виде массива данных. Также записи присвоен обязательный уникальный ключ (_id).

2.3 Выводы по разделу

В данном разделе была разработана архитектура веб-приложения с помощью диаграмм вариантов.

Диаграмма прецедентов включает в себя все возможные варианты использования системы.

Диаграммы были описаны с помощью языка UML. При разработке системы не было использовано каких-либо шаблонов проектирования.

Также была разработана база данных для системы. Был осуществлен переход к даталогическому проектированию – построена схема базы данных, приведено описание всех таблиц и атрибутов, находящихся в ней.

3 РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ

3.1 Архитектура модулей разработки

Анализируя изложенные требования к веб-приложению, разделим их на подсистемы (рисунок 3.1), которые обеспечивают заданный функционал разрабатываемого веб-приложения:

– модуль поиска;

– модуль фильтра по дате;

– модуль вывода сообщений;

– модуль построения графика;

– модуль сохранения результата.

Перечисленные функциональные модули связаны между собой, некоторые зависят друг от друга.

Рисунок 3.1 – Модульная структура сайта

3.1.1 Модуль поиска

Главный модуль веб-приложения, осуществляющий его работу. Данный модуль выполняет запрос к API поиска и отдает результат нейронной сети для анализа.

3.1.2 Модуль фильтра по дате

В веб-приложении имеется возможность ограничения результата по временному промежутку. Данный модуль устанавливает фильтр для поиска сообщений в сети.

3.1.3 Модуль вывода сообщений

Данный модуль необходим для вывода проанализированных сообщений пользователей. Текст имеет фон, соответствующий его тональности после анализа.

3.1.4 Модуль построения графика

Главный модуль отображения результата анализа. Он выполняет построение графической зависимости тональности сообщений в виде диаграммы с соответствующими полями.

3.1.5 Модуль сохранения результата

Данный модуль отвечает за экспорт статистики анализа с возможностью сохранения пользователем. Формат экспортируемого результата – графический формат PNG.

3.2 Разработка модуля поиска

Модуль поиска принимает на вход введенную текстовую строку, по которой выполняется дальнейший поиск данных для работы приложения. В результате выполнения поиска, осуществляется запрос к поисковому API. Полученные данные передаются нейронной сети для их анализа, и

возвращаются для построения результата в графическом виде. Также в случае использования модуля фильтра по дате, данные ограничиваются по заданному пользователем временному промежутку.

3.3 Разработка модуля построения графика

Модуль принимает данные от нейронной сети после выполнения анализа и выполняет построение диаграммы, отображающей результат в виде его количественной характеристики. Эмоциональная окраска сообщений пользователей отображается в виде соответствующего цвета: красный для негативных, зеленый для позитивных и белый для нейтральных текстов. Также при наведении курсором на поле диаграммы показывается количество сообщений, соответствующих выбранной тональности.

3.4 Выводы по разделу

В данном разделе были разработаны и архитектурно спроектированы алгоритмы поиска, построения графика, приведено их описание. А также описаны некоторые другие модули, необходимые для правильной работы веб-приложения.

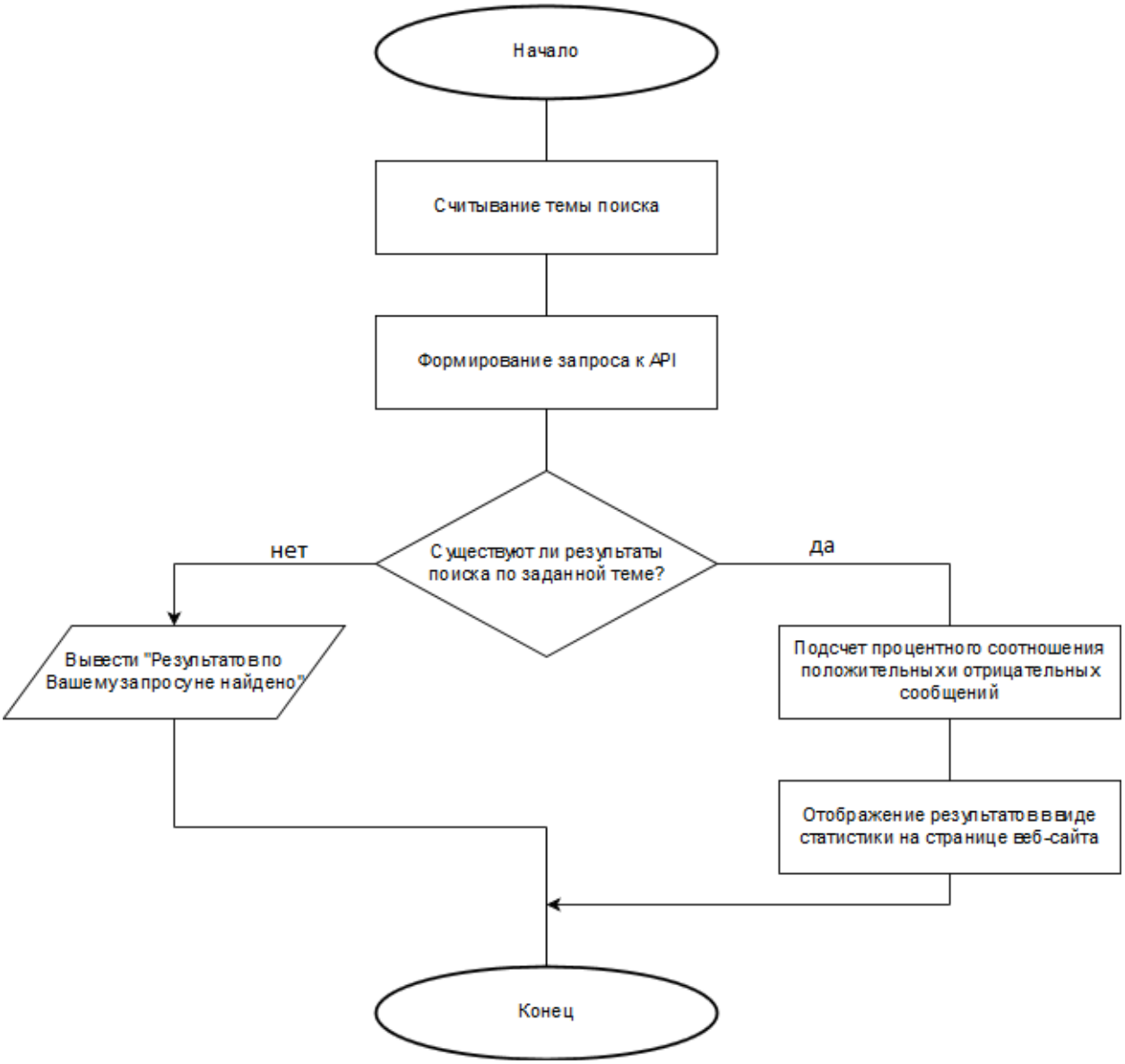
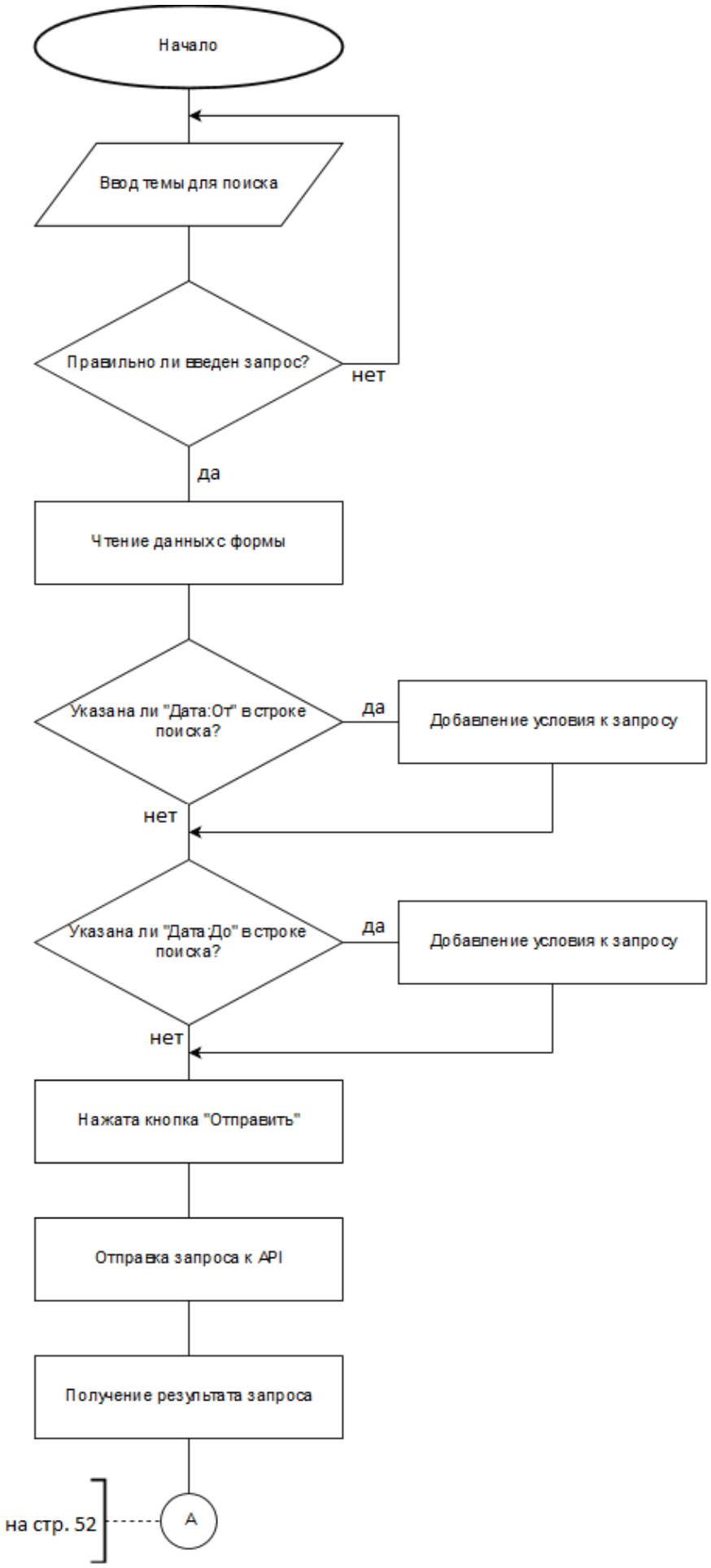


Рисунок 3.2 – Схема алгоритма поиска сообщений


Начало
Ввод темы для поиска
Правильно ли введен запрос?
нет
да
Чтение данных с формы
Указана ли "Дата:От" в строке поиска?
да
Добавление условия к запросу
нет
Указана ли "Дата:До" в строке поиска?
да
Добавление условия к запросу
нет
Нажата кнопка "Отправить"
Отправка запроса к API
Получение результата запроса
на стр. 52
A

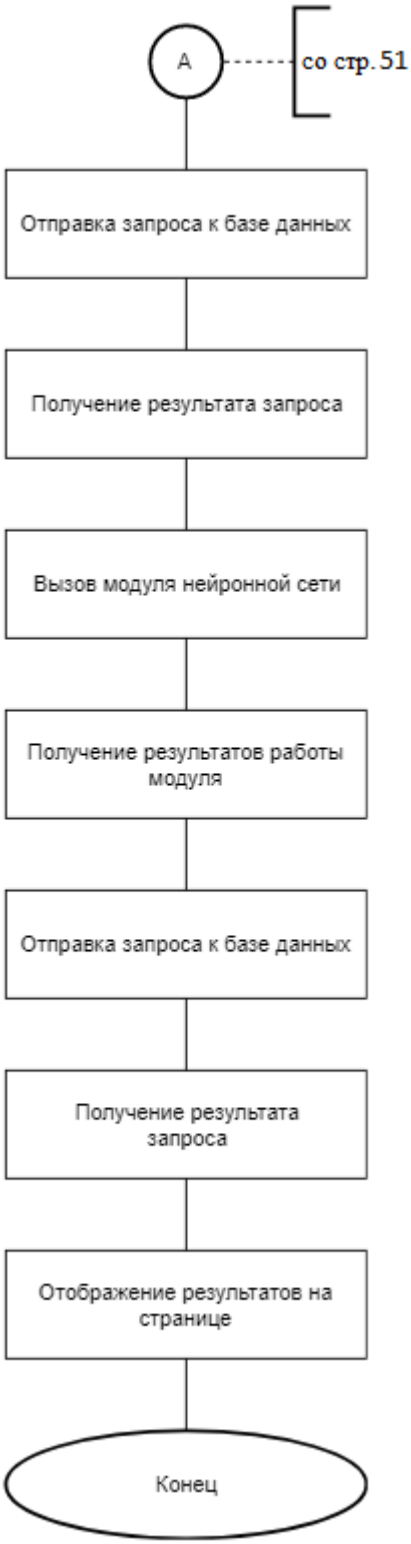


Рисунок 3.3 – Схема алгоритма системы анализа сообщений

4 ПРОВЕРКА РАБОТОСПОСОБНОСТИ

4.1 Описание порядка работы с веб-приложением

Работа с разработанным веб-приложением выполняется посредством использования главной страницы (рисунок 4.1), которая включает в себя следующие возможности:

– поиск сообщений пользователей в социальной сети;

– анализ тональности найденных сообщений;

– графическое отображение статистики в виде диаграммы;

– переход на главную страницу сайта.

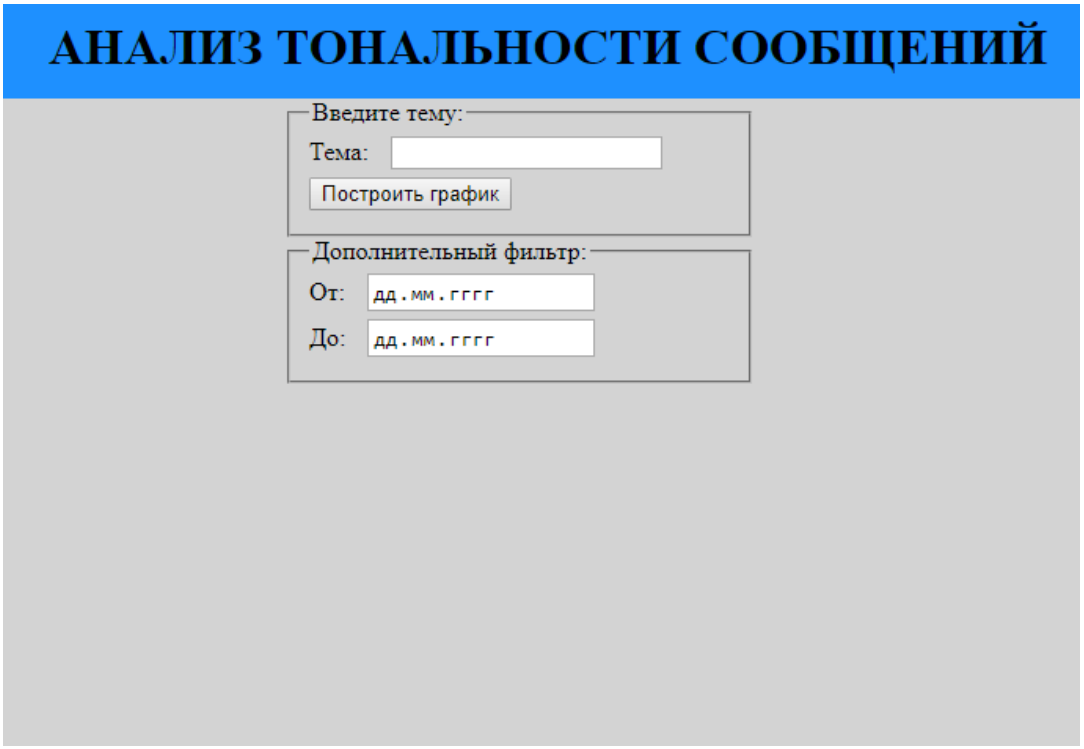


Рисунок 4.1 – Главная страница сайта

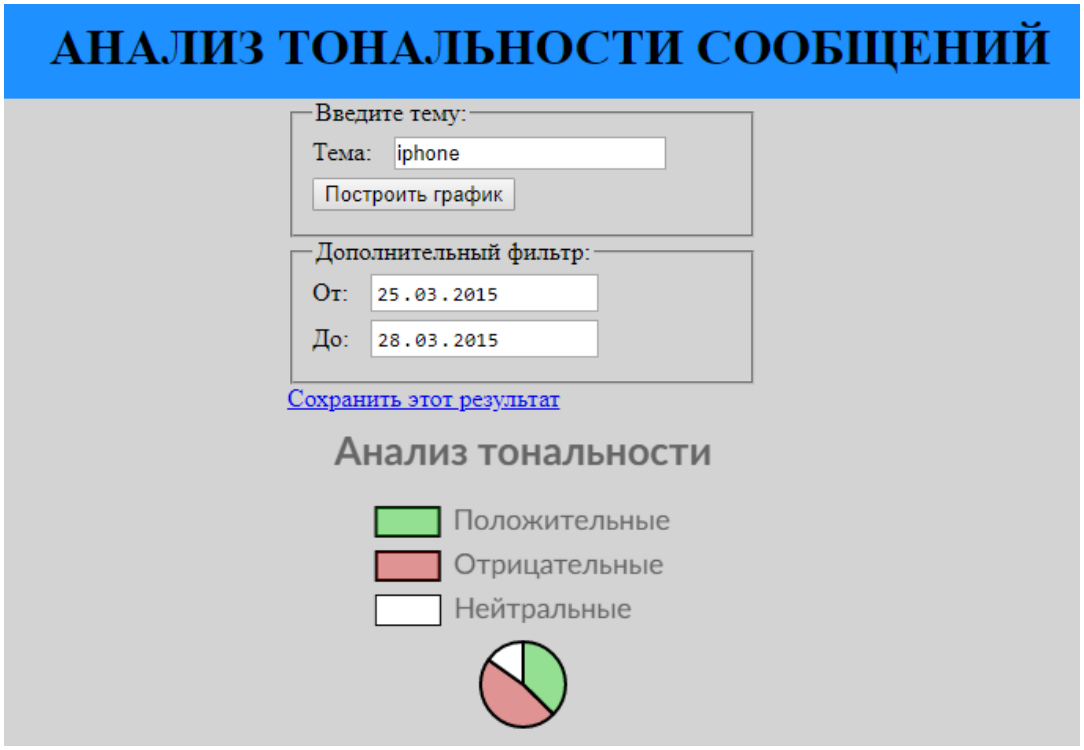


Рисунок 4.2 – Страница выбранного запроса

При введении пользователем темы в соответствующее поле на форме и нажатии на кнопку «Построить график», система выполняет поиск сообщений в социальной сети Twitter в соответствии с заданными критериями и выполняет анализ их тональности. После этого, результат отображается на главной странице веб-приложения в виде диаграммы с 3 полями: положительные, отрицательные и нейтральные сообщения (рисунок 4.2). При наведении курсором на поле в диаграмме пользователю выводится соответсвенная количественная статистика (рисунок 4.3). Также пользователь может сохранить результат в формате PNG.

Также пользователю доступен сам текст комментариев, выделенный соответсвующим их тональности фоном, который выводится ниже диаграммы анализа (рисунок 4.4). Комментарий содержит имя автора, его никнейм в социальной сети и сам текст.

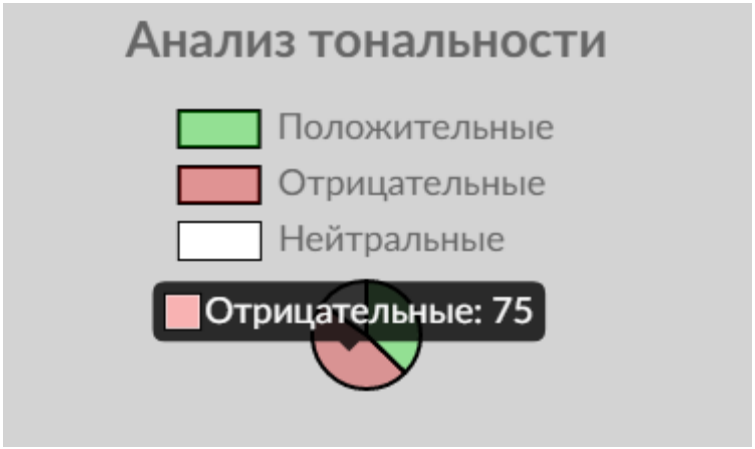


Рисунок 4.3 – Статистика количества на диаграмме

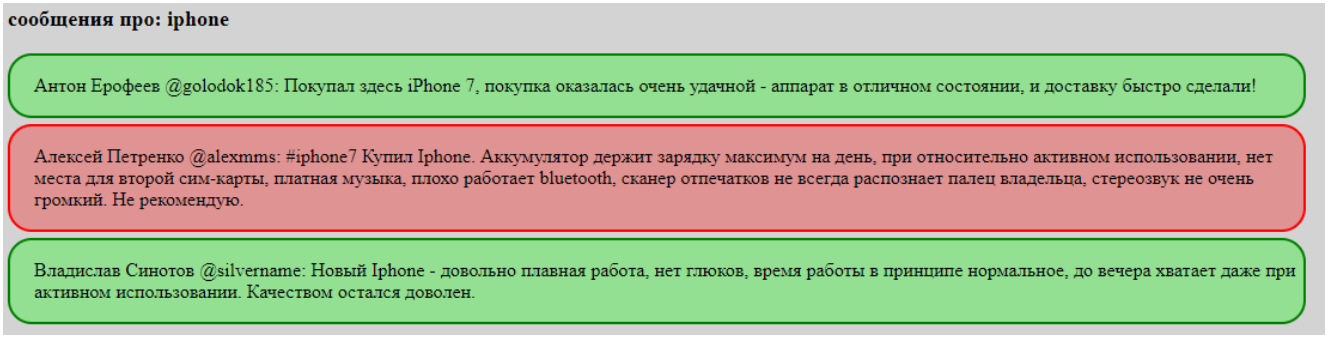


Рисунок 4.4 – Текст сообщений пользователей

4.2. Выводы по разделу

В данном разделе проведена проверка корректности работы веб-приложения. Описан весь функционал, проверена его работоспособность. По результатам тестирования можно сделать вывод о правильной работоспособности заявленного функционала.

ЗАКЛЮЧЕНИЕ

Данная работа посвящена реализации веб-приложения с использованием нейронной сети, выполняющей анализ сообщений.

Разработанное веб-приложение позволяет пользователю осуществлять поиск сообщений в социальной сети, фильтровать сообщения по указанному временному промежутку, просматривать статистику анализа тональности текста и сами сообщения пользователей.

В результате работы были решены следующие задачи:

– рассмотрены различные методы анализа тональности текста в социальных сетях;

– выбран язык программирования, подходящий для реализации рекомендательной системы, и самого веб-приложения в целом;

– спроектирована база данных, удовлетворяющая условиям поставленной нами задачи;

– разработаны алгоритм поиска сообщений пользователей;

– реализован вывод статистики в виде диаграммы;

– выполнена реализация веб-приложения, проверена его работоспособность.

В дальнейшем можно дополнить функционал веб-приложения для большего комфорта пользователей такими возможностями, как:

– добавление большего количества фильтров для более точного поиска;

– расширение количества социальных сетей, в которых может выполнятся поиск;

– сравнение разных наборов результатов.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 Флэнаган, Д. JavaScript. Подробное руководство / Д. Флэнаган; пер. с англ. А. Киселев. – изд. «Символ-Плюс», 2008. – 992 с.

2 Стригулин, К.А. Анализ тональности высказываний в Twitter / Л.В. Журавлева; «Молодой учёный», 2016. № 12. – с.185-189.

3 Cambria, E. SenticNet 2: A semantic and affective resource for opinion mining and sentiment analysis / Proceedings of AAAI FLAIRS: конференция, 2012. – с. 202–207.

4 Стефанов, С. JavaScript. Шаблоны / пер. с англ. А. Киселев. – изд. «Символ-Плюс», 2011. – 272 с.

5 Kobayshi, N. Opinion Mining on the Web by Extracting Subject-Aspect-Evaluation Relations / R. Iida, K. Inui, Y. Matsumoto; Nara Institute of Science and Technology, Takayama, Ikoma, Nara 630-0192, Japan: конференция, 2006. – с. 1-6.

6 Макконнелл, С. Совершенный код / изд. «Русская Редакция», «Microsoft Press», 2017. – 1255 с.

7 Liu, B. Sentiment Analysis and Subjectivity / Handbook of Natural Language Processing, под ред. N. Indurkhya и F. J. Damerau.) , 2010. – 38 с.

8 Крокфорд, Д. JavaScript. Сильные стороны / пер. с англ. А. Лузган. – изд. «Питер», 2013. – 176 с.

9 Pang, B. A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts / L. Lee; Proceedings of the Association for Computational Linguistics (ACL): журнал, 2004. – с. 271–278.

10 Goldberg, A. Seeing stars when there aren’t many stars:Graph-based semi-supervised learning for sentiment categorization / X. Zhu; Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing,Computer Sciences Department University of Wisconsin-Madison: конференция, 2006. – с. 45-52.

11 Закас, Н. ECMAScript 6 для разработчиков / пер. с англ. А. Киселев; изд. «Питер», 2017. – 352 с.

12 Thelwall, M. Sentiment strength detection in short informal text / K. Buckley, G. Paltoglou, Cai Di,A. Kappas; Journal of the American Society for Information Science and Technology : журнал, 2010. – с. 2544–2558.

13 Клековкина, М. Метод автоматической классификации текстов по тональности, основанный на словаре эмоциональной лексики / Е. Котельников; RCDL-2012, Переславль-Залесский, Россия: конференция, 2012.

14 Пазельская, А. Метод определения эмоций в текстах на русском языке / А. Соловьев; The international conference on computational linguistics and intellectual technologies “Dialogue 2011”: конференция. – Москва, 2011. – с. 510 - 522.

ПРИЛОЖЕНИЕ 1

ОПИСАНИЕ ПРОГРАММЫ

Веб-приложение для анализа тональности сообщений

2019

ОГЛАВЛЕНИЕ

П1.1 Общие сведения

Продукт «Веб-приложение для анализа тональности сообщений» представляет собой сайт для людей, желающих проанализировать отзывы пользоваталей по интересующей их теме.

П1.2 Функциональное назначение

Данное веб-приложение использует нейронную сеть для выполнения анализа тональности текстов.

Приложение обладает следующими возможностями:

− поиск сообщений пользователей в социальной сети;

− использование фильтра по временному промежутку;

− анализ сообщений пользователей;

− вывод результата в виде диаграммы;

− вывод проанализированных сообщений пользователей.

П1.3 Описание логической структуры

Серверная часть состоит из модулей, обеспечивающих работу с базой данных и нейронной сетью. Клиентская часть визуализирует данные на стороне пользователя.

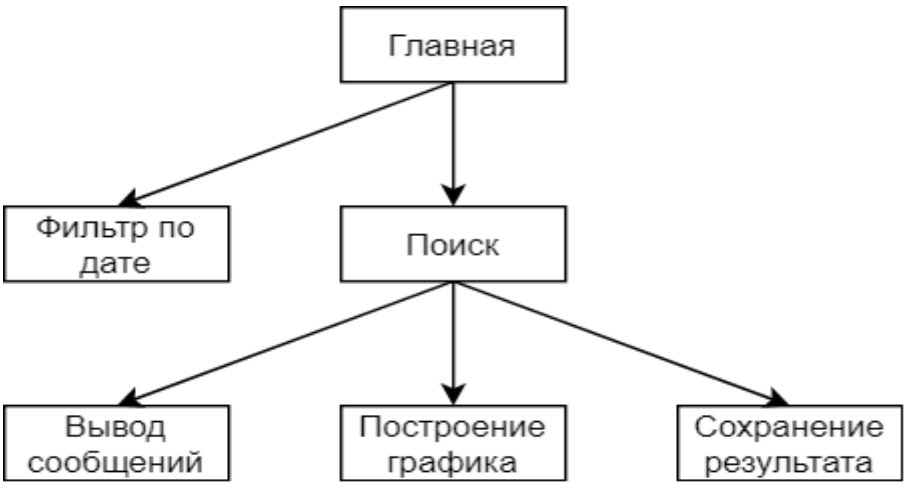


Рисунок П1.1 – Модульная структура сайта

П1.4 Используемые технические средства

Для обеспечения функционирования среды требуютсяследующие технические средства:

– IBM PC-совместимый компьютер;

– цветной монитор с диагональю не менее 15’’;

– клавиатура;

– мышь.

Необходимые программно-аппаратные ресурсы для запуска локального сервера представлены в таблице 1.1.

Таблица П1.1 – Программно-аппаратная поддержка

Составляющие	Требования
Операционная система	Microsoft Windows 7 и выше
Оперативная память, Гб	2
Пространство на жестком диске, Мб	10
Рабочая частота ЦПУ, МГц	1000

П1.5 Входные и выходные данные

Входными данными для приложения являются текстовая информация, вводимая пользователем в предназначенные для этого поля и информация о выборе предлагаемых опций, элементов меню, нажатии кнопок при управлении программой при помощи мыши.

На выходе программа формирует HTML документ, представляющий собой готовую страницу веб-приложения.

ПРИЛОЖЕНИЕ 2

ТЕКСТ ПРОГРАММЫ

2019

ОГЛАВЛЕНИЕ

server.js

```
const express = require('express');
const bodyParser = require('body-parser');
const MongoClient = require('mongodb').MongoClient;
const path = require('path');
const request = require('request');
const cheerio = require('cheerio');
require('dotenv').config();

const app = express();
const port = process.env.PORT;
const dbUrl = process.env.DB_URL;
const dbName = process.env.DB_NAME;
const mongoClient = new MongoClient(dbUrl, { useNewUrlParser: true });

app.use(bodyParser.urlencoded({ extended: true }));
app.use(bodyParser.json());
app.use(express.static('public'));

mongoClient.connect((err, client) => {
    if (err) throw err;
    console.log('Connected successfully to server');

    const db = client.db(dbName);

    require('./routes/routes')(app, path, db, request, cheerio);
});

app.listen(port);
```

index.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
        <title>twitterapi</title>
        <link rel="stylesheet" href="/styles/style.css">
        <script
src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.8.0/Chart.min.js"></script>
        <script defer src="/scripts/script.js"></script>
```

```
    </head>

    <body>
        <div class="divtitle">
            <h1>АНАЛИЗ ТОНАЛЬНОСТИ СООБЩЕНИЙ</h1>
        </div>
        <div class='divcenter'>
            <div id='tweetsDiv'>
                <form id="tweetsForm">
                    <fieldset>
                        <legend>Введите тему:</legend>
                        <label for="topicNameID">Тема:</label>
                        <input type="text" name="topicName" id="topicNameID"><br>
                        <input type="submit" value="Отправить" id="tweetsFormBtn">
                        <input type="submit" value="Построить график"
id="chartFormBtn">
                    </fieldset>
                </form>
            </div>

            <div id="fDiv">
                <form action="">
                    <fieldset>
                            <legend>Дополнительный фильтр:</legend>
                            <label for="topicNameID">От:</label>
                            <input type="date"><br>
                            <label for="topicNameID">До:</label>
                            <input type="date">
                    </fieldset>
                </form>
                <a href="#">Сохранить этот результат</a>
            </div>
        </div>

        <div id="canvasDiv">
            <canvas id="tweetChart" width="600" height="400"></canvas>
        </div>

        <div id="textsDiv"></div>
    </body>
</html>
```

style.css

```
body {
    color: black;
    background-color: lightgray;
}
.divtitle {
    background-color: dodgerblue;
    height: 60px;
}
h1 {
    display: inline-block;
    margin: 0 auto;
}
.divcenter {
    margin: 0 auto;
    width: 80%;
}
div#tweetsDiv,
#fDiv,
#canvasDiv {
    width: 300px;
    margin: 0 auto;
}

label {
    margin-right: 10px;
}

input {
    margin-bottom: 5px;
}

.tweetsDiv {
    margin-bottom: 20px;
}

.tweetsTxt {
    color: black;
    width: 67%;
    border: 2px solid black;
    border-radius: 20px;
```

```
    padding-left: 20px;
    margin-bottom: 5px;
}

#textsDiv {
    width: 100%;
    color: black;
}

#textsDiv > p {
    font-weight: bold;
    font-size: 18px;
}

.good {
    border-color: green;
    background-color: rgba(0,255,0,0.3);
}

.bad {
    border-color: red;
    background-color: rgba(255,0,0,0.3);
}
```

script.js

```
document.getElementById('tweetsFormBtn').addEventListener('click', (event) => {
    event.preventDefault();

    let tweetsForm = document.forms['tweetsForm'];
    let topic = tweetsForm.elements['topicName'].value;

    let tweet = JSON.stringify({topic: topic});
    let request = new XMLHttpRequest();

    request.open('POST', '/searchTweet', true);
    request.setRequestHeader('Content-Type', 'application/json');
    request.send(tweet);

    tweetsForm.reset();
});
```

```
let tweetCanvas = document.getElementById("tweetChart").getContext('2d');

Chart.defaults.global.defaultFontFamily = "Lato";
Chart.defaults.global.defaultFontSize = 18;

let barChart = new Chart(tweetCanvas, {
    type: 'pie',
    data: {
        labels: ["Положительные", "Отрицательные", "Нейтральные"],
        datasets: [{
            label: "Анализ тональности",
            data: [59, 75, 24],
            backgroundColor: ['rgba(0,255,0,0.3)', 'rgba(255,0,0,0.3)', '#FFF'],
            borderWidth: 2,
            borderColor: '#000'
        }]
    },
    options: {
        title: {
            display: true,
            text: 'Анализ тональности',
            fontSize: 25
        },
        legend: {
            display: true
        }
    }
});

document.getElementById('chartFormBtn').addEventListener('click', (event) => {
    event.preventDefault();

    let chartForm = document.forms['chartForm'];
    let text = chartForm.elements['chartTopicName'].value;

    let getData = JSON.stringify({text: text});
    let request = new XMLHttpRequest();

    request.open('POST', '/getData', true);
    request.setRequestHeader('Content-Type', 'application/json');
    request.addEventListener('load', () => {
```

```
        let receivedData = JSON.parse(request.response);
        console.log(receivedData.text);
    });
    request.send(getData);
});
```

routes.js

```
module.exports = function(app, path, db, request, cheerio) {
    app.get('/', (req, res) => {
        res.sendFile(path.join(__dirname, '../client', 'index.html'));
    });

    app.post('/searchTweet', (req, res) => {
        if(!req.body) return res.sendStatus(400);
        let insertObject = req.body;
        collection = db.collection(req.body.topic);
        let queryString = 'https://twitter.com/search?q=' + insertObject.topic +
'&src=typd&lang=ru';

        request(queryString, (err, res, body) => {
            if (err) throw err;

            let $ = cheerio.load(body);

            let authors = [];
            let tweets = [];

            $('div.content>div.stream-item-header>a').each(function(i) {
                authors[i] = $(this).text();
            });
            $('div.content>div.js-tweet-text-container>p').each(function(i) {
                tweets[i] = $(this).text();
            });
            console.log(authors);
            console.log(tweets);

            /*for (let i = 0; i < authors.length; i++) {
                insertObject.author = authors[i];
                insertObject.text = tweets[i];
                insertObject.sentiment = '0';
                collection.insertOne(insertObject, (err, result) => {
```

```
                    if (err) throw err;
                });
                console.log(insertObject);
            }*/

        });

        res.json(req.body);
    });

    app.post('/getData', (req, res) => {
        let collection = db.collection(req.body.text);
        let count = collection.countDocuments({});

        console.log(collection.find().count());
        res.json(req.body);
    });
};
```