МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ СИСТЕМ

ФАКУЛЬТЕТ КОМПЬЮТЕРНЫХ И ФИЗИКО-МАТЕМАТИЧЕСКИХ НАУК

КАФЕДРА ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

Допущен к защите

Заведующий кафедрой ПМИ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е. В. Разова

**АСПЕКТНЫЙ АНАЛИЗ ТОНАЛЬНОСТИ ОТЗЫВОВ О ФИЛЬМАХ**

Курсовой проект по дисциплине

«Теория алгоритмов»

Выполнил студент группы ФИб-3301-51-00 \_\_\_\_\_\_\_\_\_\_ М. С. Крупин

Руководитель,

канд. пед. наук, доцент кафедры ПМИ \_\_\_\_\_\_\_\_\_\_ Е. В. Разова

Работа защищена с оценкой \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_2022 г.

Члены комиссии: \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

Киров 2022

Оглавление

[Введение 4](#_Toc104314100)

[1 Обзор сайтов с отзывами о фильмах 6](#_Toc104314101)

[1.1 Кинопоиск 6](#_Toc104314102)

[1.2 Ivi 7](#_Toc104314103)

[1.3 Отзовик 7](#_Toc104314104)

[1.4 СпасибоВсем.ру 8](#_Toc104314105)

[1.5 Irecommend.ru 9](#_Toc104314106)

[1.6 Выводы по разделу 10](#_Toc104314107)

[2 Формирование набора отзывов с выбранного сайта 10](#_Toc104314108)

[2.1 Парсинг сайта otzovik.com 10](#_Toc104314109)

[2.2 Описание набора данных 12](#_Toc104314110)

[2.2 Выводы по разделу 17](#_Toc104314111)

[3 Обзор методов аспектного анализа тональности 18](#_Toc104314112)

[3.1 Извлечение и определение тональности аспектов 18](#_Toc104314113)

[3.2 Категоризация аспектов 24](#_Toc104314114)

[3.3 Выводы по разделу 26](#_Toc104314115)

[4 Разработка программы аспектного анализа тональности 26](#_Toc104314116)

[4.1 Выбор языка программирования и библиотек 26](#_Toc104314117)

[4.2 Обучение модели RNN 27](#_Toc104314118)

[4.3 Тестирование модели 33](#_Toc104314119)

[4.4 Описание разработанной программы 39](#_Toc104314120)

[4.4 Выводы по разделу 40](#_Toc104314121)

[Заключение 41](#_Toc104314122)

[Библиографический список 42](#_Toc104314123)

[Приложения 46](#_Toc104314124)

[Приложение А 46](#_Toc104314125)

[Приложение Б 48](#_Toc104314126)

[Приложение В 53](#_Toc104314127)

[Приложение Г 55](#_Toc104314128)

[Приложение Д 56](#_Toc104314129)

# Введение

Эпоха информационного общества характеризуется постоянно растущими объёмами данных, которыми делятся пользователи со всего мира. Пользователям Интернета постоянно предлагают поделиться своим мнением, оценкой по поводу какой-либо новости, качеству обслуживания, приобретённого товара. Желание людей узнать мнение о том товаре, который они собираются приобрести, у других людей, которые его уже приобрели, привело к росту количества отзывов практически ко всему. На веб-сайтах, предоставляющих какие-либо услуги или продающих товары, начали появляться специальные места, где пользователи могут оставить свой отзыв. Ведь мнение человека о чём-либо в большой части строится от мнений других людей. Также стали даже появляться сайты, предназначенные только для отзывов по продуктам какой-либо тематики, например: «Кинопоиск», «Отзовик», «IMDb».

С точки зрения производителя любой человек – потенциальный покупатель. Следовательно, знание их симпатий и антипатий может оказать большую помощь в разработке новых продуктов и улучшении существующих. Здесь и пригождается аспектных анализ тональности, который систематизирует отзывы, выделяет в каждом из них плюсы и минусы определённого аспекта товара.

Актуальность данной работы заключается в создании программы, которая будет проводить аспектный анализ отзывов о фильмах на русском языке, так как при изучении литературы и Интернета таковой найдено не было. Данная программа может применяться на сайтах, главное предназначение которых – просмотр фильмов, сериалов и написание отзывов о них. Она может быть полезна для пользователей в формировании общей оценки о фильме по всем отзывам, либо быстрому и ясному пониманию оценки фильма в конкретном отзыве. Также и администрация сайта может видеть предпочтения пользователей.

Исходя из актуальности, можно выделить цель данной работы: разработать программу для аспектного анализа тональности отзывов о фильмах.

Для достижения цели были поставлены следующие задачи:

1. Сделать обзор сайтов с отзывами о фильмах;
2. Написать парсер отзывов с выбранного сайта;
3. Сделать обзор методов аспектного анализа тональности;
4. Разработать программу аспектного анализа тональности фильмов;
5. Дать оценку качества разработанной программы.

Курсовой проект состоит из четырёх разделов, заключения, библиографического списка и приложений. Содержания разделов соответствуют поставленным задачам.

1 Обзор сайтов с отзывами о фильмах

Для аспектного анализа отзывов о фильмах необходимо сформировать базу этих отзывов. Желательно, чтобы у каждого отзыва уже была его оценка по различным аспектам, так как так будет легче проверять правильность работы программы аспектного анализа. Также было бы плюсом, если бы отзыв делился на составляющие, по которым будет проще его анализировать. Например: достоинства, недостатки, пожелания.

1.1 Кинопоиск

На данном сайте отзывы бывают трёх видов:

1. Положительные, помечаются зелёным фоном;
2. Отрицательные, помечаются красным фоном;
3. Нейтральные, помечаются серым фоном;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1.1 – Пример отзыва с сайта «Кинопоиск»

Информации о содержании отзыва достаточно мало. Для аспектного анализа недостаточно знать эмоциональную окраску отзыва. Сам отзыв не делится на составляющие, только если сам автор этого не сделает.

Для Кинопоиска существует API, с помощью которого можно получать всю информацию об отзыве, не прибегая к парсингу.

1.2 Ivi

На данном сайте об отзыве не предоставляется никакой информации. Однако есть общий аспектный анализ отзывов о фильмах, состоящий из режиссуры, зрелищности, игры актёров и сюжета. Каждый из аспектов оценивается по стобалльной шкале.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1.2 – Пример отзыва с сайта «Ivi»

1.3 Отзовик

Каждый отзыв сайта «Отзовик» делится на пункты: «Достоинства», «Недостатки» и основной раздел. В конце отзыва приводится аспектная оценка фильма, составленная пользователем, которая состоит из сюжета, зрелищности, игры актёров, оригинальности, музыки и звука. Также автор ставит оценку от 1 до 5 и рекомендует или не рекомендует этот фильм. По данным причинам этот сайт хорошо подходит для составления размеченного корпуса отзывов о фильмах.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1.3 – Пример отзыва с сайта «Отзовик»

1.4 СпасибоВсем.ру

Структура отзыва похожа на структуру отзыва с сайта «Отзовик», однако в оценке отсутствует шкала «Оригинальность». Сам сайт менее популярен по сравнению с вышеупомянутом с сайтом, поэтому отзывов на нём меньше, и собрать их достаточное количество будет труднее.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1.4 – Пример отзыва с сайта «СпасибоВсем.ру»

1.5 Irecommend.ru

На этом сайте пользователь ставит оценку фильму от 1 до 5, пишет отзыв и по желанию отмечает достоинства и недостатки. То, что достоинства и недостатки есть не в каждом отзыве, значительный минус.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1.5 – Пример отзыва с сайта «Irecommend.ru»

1.6 Выводы по разделу

В данном разделе был проведён обзор пяти сайтов с отзывами о фильмах. В каждом были выделены достоинства, которые бы лучше всего поспособствовали построению модели анализа аспектов фильма. В результате для дальнейшего использования и создания на его основе размеченного корпуса отзывов был выбран наилучший сайт – «Отзовик», на котором отзыв является структурированным (достоинства, недостатки, основной раздел), даётся оценка по пятибалльной шкале, общая рекомендация, а самое главное – оценка фильма по аспектам: «Сюжет», «Зрелищность», «Игра актёров», «Оригинальность», «Музыка и звук». В таблице 1.1 представлена информация о том, какая информация присутствует в отзывах на разных сайтах.

Таблица 1.1 – Пункты отзыва описания фильма по сайтам

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Пункт  Сайт | Плюсы | Минусы | Количество аспектов | Рекомендация | Оценка |
| Кинопоиск | – | – | 0 | – | + |
| Ivi | – | – | 0 | – | – |
| Отзовик | + | + | 5 | + | + |
| СпасибоВсем.ру | + | + | 4 | + | + |
| Irecommend.ru | ∓ | ∓ | 0 | – | + |

2 Формирование набора отзывов с выбранного сайта

2.1 Парсинг сайта otzovik.com

Для парсинга отзывов о фильмах был взят язык программирования Java по причине наличия опыта работы с библиотекой Jsoup, позволяющей осуществлять парсинг и предоставляющей широкий функционал, благодаря которому можно выбирать необходимую информацию в HTML-документе.

Для парсера задавались следующие параметры:

1. Количество фильмов;
2. Количество отзывов в каждом фильме;
3. Минимальное и максимальное количество отзывов в фильме;
4. Порядок фильмов (по убыванию или возрастанию рейтинга).

При парсинге параллельно считается статистика получаемых данных, которая записывается в файл (рис. 2.1). Также по причине того, что при большом количестве запросов к серверу сайт блокирует IP, то, когда возникает эта ошибка, парсер приостанавливается на одну минуту, оповещая об этом пользователя, и в это время необходимо поменять свой IP-адрес.

В результате парсинга отзывов программа создаёт файл с полученными данными, которые будут описаны в следующем подразделе.

Программная реализация парсера приведена в приложении Б.

Изображение выглядит как текст

Автоматически созданное описаниеИзображение выглядит как текст

Автоматически созданное описание

Рисунок 2.1 – Пример статистики собранных данных

2.2 Описание набора данных

В результате парсинга сайта otzovik.com был сформирован размеченный корпус, состоящий из 57361 отзыва о фильмах. Отзыв имеет следующую структуру:

1. Идентификатор отзыва;
2. Плюсы, выделенные ревьюером;
3. Минусы, выделенные ревьюером;
4. Основная часть отзыва;
5. Общий рейтинг фильма: количество звёзд от 1 до 5;
6. Рейтинг игры актёров: количество звёзд от 1 до 5;
7. Рейтинг музыки и звуков: количество звёзд от 1 до 5;
8. Рейтинг оригинальности: количество звёзд от 1 до 5;
9. Рейтинг сюжета: количество звёзд от 1 до 5;
10. Рейтинг зрелищности: количество звёзд от 1 до 5.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 2.2 – Пример отзыва

## 2.1.1 Описание оценок пользователей по аспектам фильмов

При парсинге отзывов было сделано так, чтобы количество отзывов на всех рейтингах было примерно одинаково (рис. 2.1), однако пользователи оставляют большее количество положительных отзывов, чем отрицательных, поэтому средний рейтинг (рис. 2.2) оказался на 0,2–0,4 балла больше, чем 3, что является средней оценкой. Медиана оценки (рис. 2.3) для всех отзывов составила 3 балла, что означает, что количество отзывов с оценкой, меньшей, чем 3, и оценкой, большей, чем 3, примерно одинаково. Мода, самая часто встречающаяся оценка, равна пяти (рис. 2.4).

Рисунок 2.3 – Количество отзывов каждой оценки каждого аспекта

Рисунок 2.4 – Средний рейтинг аспектов

Рисунок 2.5 – Медиана рейтинга аспектов

Рисунок 2.6 – Мода рейтинга аспектов

## 2.1.2 Описание общего рейтинга

Статистические оценки общего рейтинга фильма и рейтинга по аспектам совпадают, что говорит о корректности составления оценок пользователями.

Рисунок 2.7 – Количество отзывов для каждой оценки общего рейтинга

Средний рейтинг: 3,234.

Медиана рейтинга: 3.

Мода рейтинга: 5

## 2.1.3 Описание текста отзыва

Средняя длина плюсов и минусов, выделенных пользователем (табл. 2.1), составляет 32–35 символов, что равняется примерно шести-восьми словам. Поэтому из данной части можно выделить достаточно немного информации.

Средняя длина основной части отзыва составляет 1750 символов, что составляет примерно 290 слов. Из этой части и будет поступать основная информация об аспектах фильма. Однако плюсы и минусы тоже важны, потому что в них эмоциональная окраска определена заранее.

Таблица 2.1 – Характеристики длины (в символах) текстовых частей отзыва

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Среднее | Медиана | Минимум | Максимум |
| Плюсы | 31,059 | 24 | 2 | 255 |
| Минусы | 30,113 | 22 | 2 | 255 |
| Основная часть | 1423,16 | 1103 | 103 | 24133 |

Далее приведены диаграммы распределения количества отзывов по длине текстовых частей отзыва (по горизонтали – длина, по вертикали – количество отзывов).

Рисунок 2.8 – Распределение количества отзывов по длине (в символах) плюсов, выделенным пользователем

Рисунок 2.9 – Распределение количества отзывов по длине (в символах) минусов, выделенным пользователем

Рисунок 2.10 – Распределение количества отзывов по длине (тыс. символов) основной части отзыва

2.2 Выводы по разделу

В результате парсинга сайта с отзывами о фильмах otzovik.com был сформирован размеченный корпус отзывов со следующими аспектами: музыка, игра актёров, оригинальность, зрелищность, сюжет. Количество отзывов для каждой оценки аспекта примерно совпадает. Это означает, что при обучении модели определения аспектной тональности, она будет обучена с одинаковым качеством для всех оценок от 1 до 5.

3 Обзор методов аспектного анализа тональности

Рисунок 3.1 – Методы аспектного анализа тональности

3.1 Извлечение и определение тональности аспектов

Извлечение аспектов из отзывов для получения мнений пользователей впервые было изучено Hu, Liu и Street [10]. Они ввели термины явные и неявные аспекты. Явные аспекты – это существительные и их сочетания, которые есть в тексте. Например, «У этого телефона мощный аккумулятор, но экран маленький». Аккумулятор и экран – явные аспект, так как они прямо упомянуты в тексте. С другой стороны, аспект может быть упомянут не так очевидно [11], и он будет называться неявным. Например, «Стул очень мягкий». Здесь неявно присутствует аспект «сиденье стула», так как мягкость именно этого аспекта стула имелась в виду. В первом примере оценка общей тональности отзыва приведёт к нейтральному отношению, так как там есть один положительный и один отрицательный аспект. А во втором примере сформируется только положительное мнение, хотя информация есть только о сиденье стула. Поэтому важно при оценке отзыва выделять в нём аспекты, по которым в результате можно сформировать корректный анализ отзыва.

## 3.1.1 Синтаксический метод

Данный метод находит аспекты на основе синтаксических отношений, в которых они находятся [30]. Очень частым отношением является отношение существительного и прилагательного, например, словосочетание «вкусная еда». Здесь чаще всего поиск происходит по прилагательным, так как они почти всегда принадлежат к какому-то существительному.

Плюсы:

1. Находит малочастотные аспекты;
2. Эмоционально окрашенные слова известны или их легко найти.

Минусы:

1. Выделяется много неаспектов;
2. Необходимо ручное составление шаблонов, по которым находятся пары «аспект – отношение к нему».

## 3.1.2 Метод на основе правил

Методы, основанные на правилах их извлечения, были впервые изучены Poria и др. [13]. Авторы решают проблему извлечения аспектов на основе набора отзывов о продуктах, предлагая новый подход, который полагается на синтаксические связи в предложениях, деревья зависимостей предложений, лексиконе аспектов, лексиконе прилагательных, оценивающих аспекты, извлекая как явные, так и неявные аспекты. С тех пор, как данный метод стал популярен, появилось много его улучшений [14][15].

## 3.1.3 Методы обучения с учителем

В данных методах потребуется размеченный набор данных с заранее известными аспектами и их тональностями, так как на каждом этапе обучения с учителем полученный ответ сравнивается с правильным.

Наиболее распространённые методы обучения с учителем – деревья решений, метод опорных векторов [16], метод k ближайших соседей [17], наивный байесовский классификатор [18] и некоторые нейронные сети, например, рекуррентная нейронная сеть. Таже используются методы обучения и без учителя, чтобы не полагаться на размеченные данные [19], автоматически извлекать аспекты, используя синтаксические и контекстные шаблоны без необходимости использования размеченных отзывов [14][20].

## 3.1.4 Метод условных случайных полей

Это один из методов обучения с учителем, также использующийся в методах извлечения именованных сущностей. По этой причине множество исследователей используют метод условных случайных полей, поскольку он и метод извлечения именованных сущностей имеют аналогичные задачи. Shu и др. [21] использовали метод непрерывного обучения в сочетании с методом условных случайный полей для улучшения выделения аспектов за счёт сохранения информации, полученной из предыдущих стадий обучения, которые впоследствии использует метод условных полей.

## 3.1.5 Метод свёрточной нейронной сети

Методы, описанные выше, имеют свои ограничения. Условные случайные поля – линейная модель, поэтому ей нужно большое количество признаков, чтобы хорошо работать, языковые шаблоны необходимо составлять вручную, которые, к тому же, сильно зависят от грамматической правильности предложения. В последнее время методы, основанные на модели глубокого обучения, дают многообещающие результаты в любой задаче анализа настроений, включая извлечение аспектов. Чтобы преодолеть вышеуказанные ограничения, Poria, Cambria и Gelbukh [12] предложили свёрточную нейронную сеть, нелинейный классификатор с учителем. Они также представили определённые языковые шаблоны и объединили подход на основе этих шаблонов с подходом глубокого обучения для задач извлечения аспектов.

Свёрточные нейронные сети широко используются в исследованиях в области обработки изображений. Поскольку данный метод не требует сложных вычислений, Y. Kim [22] предложил использовать его для классификации текстов, что дало хороший результат. Благодаря этому свёрточные нейронные сети стали более популярными в области получения аспектов из отзывов.

Используя метод встраиваемого слоя, Shu и др. [29] изменили стандартную модель свёрточной нейронной сети и назвали её управляемой, идея которой заключается в асинхронном обновлении управляющих модулей и слоёв свёрточной нейронной сети, что повысило производительность задачи извлечения аспектов.

## 3.1.6 Метод рекуррентной нейронной сети

Данный метод был использован Д. С. Тарасовым [5] для аспектного анализа тональности отзывов о ресторанах. Рекуррентная нейронная сеть (англ. Recurrent Neural Network, RNN) – тип нейронной сети, имеющей рекуррентные связи, образующих направленную последовательность. Здесь нейроны могут обмениваться информацией между собой. Например, помимо входных данных нейрон получает некоторую информацию о предыдущем состоянии сети.

В сети Элмана [31] (рис. 3.2) активации *h(t)* скрытого слоя на временном шаге *t* вычисляются путём преобразования текущего входного слоя *x(t)* и предыдущего скрытого слоя *h(t-1)*. Выходные данные *y(t)* вычисляются из скрытого слоя *h(t)*. То есть, учитывая последовательность векторов *{x(t)}*, где *t=1…T*, рекуррентная нейронная сеть Элмана вычисляет выходные последовательности:

(3.1)

(3.2)

где *f* – нелинейная функция, такая как сигмоида и или функция гиперболического тангенса, а *g* – функция выхода. *W* и *V* – матрицы весов между входным и скрытым слоями. *U* – матрица выходных весов, *b* и *c* – векторы смещения, связанные со скрытыми и выходными состояниями. *h(0)* в уравнении 2.1 может быть константным значением, которое выбирается произвольно или обучается обратным распространением.

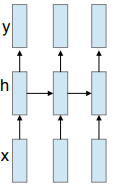


Рисунок 3.2

## 3.1.7 Долгая краткосрочная память

Это популярная архитектура RNN, которая была представлена Зеппом Хохрайтером и Юргеном Шмидхубером. В своей статье [32] они работают над решением проблемы долгосрочных зависимостей. То есть, если предыдущее состояние, влияющее на текущий прогноз, не было в недавнем прошлом, модель RNN не сможет точно предсказать текущее состояние. В качестве примера возьмём ситуацию, когда автор отзыва решил разбить свой отзыв на большие блоки: для каждого аспекта фильма по блоку. С проходом по блоку нейронная сеть забудет тему этого блока, которая была обозначена в начале, и не сможет соотносить оценку с аспектом. Для исправления этого в долгой краткосрочной памяти есть «ячейки памяти», которые имеют три типа фильтров: входные, выходные и фильтры забывания. Входные фильтры используются для сохранения информации, выходные – для чтения, фильтры забывания – для удаления информации.

Фильтры забывания определяют, сохранить информацию с предыдущего шага или забыть её.

(3.3)

где *σ* – сигмоидальная функция (изменяется от 0 до 1), *x(t)* – входное значение, *Wf* – матрица весов входных данных, *h(t-1)* – выходное значение скрытого слоя предыдущего шага, *Vf* – матрица весов скрытого слоя. Если *f(t)* = 0, то значение с предыдущего шага (*c(t-1)*) полностью отбрасывается, если *f(t)* = 1, то полностью запоминается.

Входной фильтр определяет, какая новая информация будет храниться в «ячейке памяти». Сначала высчитывается важность новой информации, получаемой из входных данных

(3.4)

где *i(t)* изменяется от 0 до 1. Далее высчитывается значение, которое будет добавляться в «ячейку памяти»

(3.5)

где *N(t)* будет изменяться от -1 до 1. Если значение будет отрицательным, то информация вычтется из «ячейки памяти», если положительным – то прибавится. Текущее состояние ячейки *c(t)* вычисляется следующим образом:

(3.6)

где *f(t)* – функция из формулы 3.3, определяющая.

В выходном фильтре сначала применяется сигмоидальная функция для определения, какую информацию мы будем выводить:

(3.7)

Далее считается текущее значение скрытого слоя, которое передаётся на следующий шаг:

(3.8)

## 3.1.8 Двунаправленная рекуррентная нейронная сеть

RNN данного типа [33] проводят обработку в прямом и обратном направлениях одновременно. Они используются, когда важно знать прошлый и будущий контекст. Здесь состояния нейрона разделяются на части, ответственные за прямое и обратное направления.

(3.9)

(3.10)

Например, мы хотим угадать недостающее слово по окружающему контексту в предложении «Уже сейчас искусственный интеллект \_\_\_ самые разные прикладные области». Зная только левую или только правую часть этого предложения, почти невозможно определить недостающее слово. Однако по обеим частям становится понятно, что здесь подходит слово «завоёвывает».

3.2 Категоризация аспектов

Вторая задача аспектного анализа – это отнесение мелких аспектов к более крупным, называемым аспектными категориями [23]. Например, в предложении «Должен сказать, у них одна из самых быстрых доставок в городе и вежливые курьеры». Здесь выделяются аспекты «время доставки» и «курьеры», что относится к более крупному аспекту «обслуживание». Ganu, Elhadad и Marian [25] разработали классификацию отзывов для конкретных категорий в обзорах ресторанов. Также они установили, что сам текст обзора является намного лучшим показателем, чем другая информация, такая как, например, звёздный рейтинг.

## 3.2.1 Модель тональностей с заданными аспектами

Mukherjee и Liu [23] решили проблему категоризации, когда пользователь предоставляет несколько ключевых слов для нескольких категорий аспектов, а модель одновременно извлекает и группирует аспекты в категории. Пользовательские параметры категоризации важны, поскольку это субъективная задача: для разных целей может понадобиться выделение разных категорий. Авторы предложили две статистические модели для решения этой задачи, направленные на то, чтобы выяснять, что именно хочет пользователь: модель тональностей с заданными аспектами (от англ. – seeded aspect sentiment model) и модель тональностей аспектов с максимальной энтропией (от англ. – maximum-entropy seeded aspect sentiment model).

## 3.2.2 Многозадачная нейронная сеть

Классификация аспектов и их извлечение из текста часто рассматриваются независимо, хотя они и тесно связаны. Полученные знания из одной задачи должны помогать в изучении другой. Xu и др. [24] предложили модель многозадачного обучения на основе нейронных сетей (MTNA) для решения обеих задач. Категоризация аспектов как контролируемая задача классификации, в которой предложение должно быть помечено в соответствии с множеством предопределённых меток аспектов, и выделение аспектов как задача последовательного помечания, в которых слова, относящиеся к одному и тому же аспекту, должны быть помечены в соответствии с предопределённой схемой маркировки. Авторы предложили объединять методы BiLSTM – для выделения аспектов и свёрточную нейронную сеть – для категоризации.

## 3.2.3 Объединение выделения и категоризации аспектов

Akhtar, Garg и Ekbal [26] предложили два подхода совместного изучения двух задач (извлечение аспектов и их классификация). Первый подход основан на структуре, в которой две задачи решаются последовательно. В ней используется нейронная сеть BiLSTM-CNN, которая отвечает за последовательное изучение слов в предложении, из которых впоследствии выделяются аспекты. Второй подход, напротив, объединяет две задачи в одну и решает как одну целостную задачу. В этом подходе аспекты делятся на классы: B-положительный, I-положительный, B-отрицательный, I-отрицательный, B-нейтральный, I-нейтральный, B-конфликтный, I-конфликтный и O.

3.3 Выводы по разделу

В данном разделе были рассмотрены методы аспектного анализа тональности текстов. Для дальнейшего использования, построения модели и разработки программы для аспектного анализа тональности отзывов к фильмам, была выбрана RNN (рекуррентная нейронная сеть), так как она хорошо зарекомендовала себя в задачах обработки естественного языка и используется при работе с последовательными данными, поскольку каждый следующий шаг зависит от предыдущего. Данная нейронная сеть использует слой краткосрочной памяти, благодаря которому она может более точно предсказать следующие данные или связать несколько слов между собой. Например, «фильм, который мы посмотрели, показался мне довольно интересным». В этом предложении аспект «фильм» и отзыв к нему «интересным» находятся в разных частях предложения, но благодаря краткосрочной памяти, они не пропускаются. Также можно применить двунаправленную RNN, чтобы проводить обработку текста не только в прямом, но и в обратном порядке.

4 Разработка программы аспектного анализа тональности

4.1 Выбор языка программирования и библиотек

При изучении программных библиотек для обучения нейронных сетей было обнаружено, что их подавляющее большинство предназначено для языка Python. С точки зрения широкого выбора инструментов для работы с машинным обучением, был выбран данный язык программирования. Для выбранной же модели (RNN) подходит библиотека «Keras», предоставляющая модели рекуррентной нейронной сети, долгой краткосрочной памяти и двунаправленной RNN. Для обработки обзоров фильмов была взята библиотека «Spacy», в которой есть поддержка русского языка.

4.2 Обучение модели RNN

## 4.2.1 Предобработка набора отзывов

Так как нейронная сеть принимает на вход числа, а не текст, то нам нужно каким-либо образом закодировать отзывы о фильмах.

Для начала каждый отзыв (структура отзыва описана в пункте 2.1) был разбит на слова с удалением знаков препинания при помощи библиотеки «Spacy». Все слова были приведены к начальной форме при помощи лемматизатора библиотеки «Spacy» на основе набора данных[[1]](#footnote-1) для лемматизации слов на русском языке. При помощи этой же библиотеки, используя морфологизатор Nerus[[2]](#footnote-2), каждому слову была приписана метка с частью речи. Далее в каждом отзыве были оставлены только те слова, длина которых больше 1 и которые являются существительным, глаголом или прилагательным.

К получившемуся набору отзывов было применено частотное кодирование. Каждое слово кодировалось числом в соответствии с его частотой вхождения во все отзывы. Самое часто встречающееся число кодировалось единицей, следующее по частое – двойкой, и т. д. Далее были удалены слова, которые встречаются не более двух раз. Все отзывы были обрезаны по длине 300, которая является средней длиной по всему датасету. В результате получилось 39000 уникальных слов.

Оценки аспектов кодировались вектором, состоящим из нулей и одной единицы, стоящей на месте с номером корректной оценки. Например, оценке 3 из 5 соответствует вектор (0, 0, 1, 0, 0).

В результате обработки получился датасет, состоящий из 57361 (количество отзывов в датасете) пар, левая часть которых – вектор чисел, представляющих отзыв, а правая – вектор, представляющий оценку какого-либо аспекта фильма.

Код предобработки находится в приложении В.

## 4.2.2 Создание модели на Python при помощи Keras

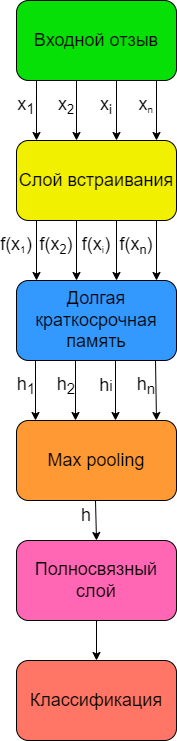


Рисунок 4.1 – Модель нейронной сети

1. Первый слой – входной. Он преобразует текстовый отзыв в числовой вектор (x1, x2, …, xn) так, как было описано в предыдущем пункте.
2. Слой встраивания преобразует числа входного вектора в числовые векторы, в результате чего на выход этого слоя поступает матрица. Так как нейронные сети работают на векторах целых чисел, то слова (которые были преобразованы до этого в числа) нужно представить в виде векторов. При наипростейшем кодировании каждому слову ставится в соответствие вектор размера количества уникальных слов в тексте, в котором на всех местах будут стоять нули, кроме одного, которому соответствует данное слово. Таким образом текст будет закодирован в матрицу с количеством строк, равным размеру текста, и с количеством столбцов, равным количеству уникальных слов в тексте. Допустим, в тексте 10000 слов, а размер его лексикона – 5000 слов. Тогда получится разреженная матрица размером 10000x5000, состоящая почти полностью из нулей. А при помощи слоя встраивания можно сделать это намного компактнее, задавая размер векторов для слов. На вход этого слоя поступает текст, размер его словаря и размер вектора для каждого слова. Например, входному вектору (1, 4, 2, 3) и размеру вектора 3 на выходе будет соответствовать матрица

В данном случае здесь выставлены случайные значения.

Слой встраивания обучаем, и в процессе обучения слова с похожими значениями будут иметь схожие векторы;

1. RNN: двунаправленная сеть долгой краткосрочной памяти, которая была описана в пунктах 3.1.7 и 3.1.8. На вход данного слоя поступает матрица из предыдущего пункта. На выходе – матрица с количеством строк, которые задаются в параметрах (выходная размерность), и столбцами – все выходные состояния *hi* этапов обучения.
2. В каждом векторе поступившей матрицы выбирается самое важное (наибольшее) значение. Таким образом формируется один вектор, который поступает на выход. Например, из матрицы

получится вектор (-0,0081; 0,045; 0,028; 0,036). При помощи добавления данного слоя в модель удалось увеличить показатели всех метрик на 10%–15%.

1. Полносвязный слой преобразует поступивший вектор в вектор, размерность которого – количество классов, к одному из которых может принадлежать входной отзыв. Элементом каждого вектора является вероятность принадлежности отзыва к классу с данным номером.
2. Входной отзыв относится к классу с номером, под которым стоит элемент выходного вектора полносвязного слоя с наибольшей вероятностью.

Программная реализация приведена в приложении Г.

## 4.2.3 Обучение при помощи кросс-валидации

Всего будет обучено 15 моделей: по 3 модели (двухбалльная, трёхбалльная и пятибалльная шкалы оценивания) для каждого из пяти аспектов. При обучении каждой модели датасет будет разделяться на пять равных частей, четыре из которых (45888 отзывов) будут браться для обучения, а одна (11473 отзыва) – для тестирования, и таким образом будут перебираться все 5 сочетаний тренировочных и тестовых наборов. В результате возьмётся средний результат по пяти итерациям обучения. Такой метод называется кросс-валидацией.

Каждый класс всех пяти частей разбиения на тренировочном и тестовом наборах имеет одинаковую долю относительно размера набора. То есть все тренировочные и тестовые датасеты одинаково сбалансированы. Процентные соотношения оценок для разных шкал оценивания приведены далее.

Рисунок 4.2 – Процентное соотношение оценок отзывов для сюжета

Рисунок 4.3 – Процентное соотношение оценок отзывов для актёрской игры

Рисунок 4.3 – Процентное соотношение оценок отзывов для музыки

Рисунок 4.4 – Процентное соотношение оценок отзывов для зрелищности

Рисунок 4.5 – Процентное соотношение оценок отзывов для оригинальности

Параметры нейронной сети при обучении:

1. Слой встраивания. Размер входного вектора (количество слов) – 250. Размер словаря (количество уникальных слов) – 39731. Выходной размер вектора для каждого слова – 100;
2. Долгая краткосрочная память. Выходная размерность матрицы – 100, процент прореживания сети – 20%;
3. Полносвязный слой. Функция активации – сигмоида;
4. Количество эпох – 20;
5. Batch size (размер партии) – 32 отзыва.

4.3 Тестирование модели

Для определения точности обученной модели будут использованы следующие метрики:

True positives (TP) – количество отзывов, которым был поставлен класс, к которому на самом деле относился отзыв.

False positives (FP) – количество отзывов, которым был поставлен класс, к которому на самом деле не относился отзыв.

False negatives (FN) – количество отзывов, которым не был поставлен класс, к которому на самом деле относился отзыв.

1. Точность для всех классов:

, (4.1)

где 0 ≤ *Accuracy* ≤ 1. *Accuracy* означает вероятность корректного определения класса отзыва.

1. Точность определения оценки отзывов для одного класса:

, (4.2)

где 0 ≤ *Precision* ≤ 1. TP+FP означает количество отзывов, которые классификатор пометил как отзывы, относящиеся к данному классу. Чем больше количество неверных определений класса, тем меньше *Precision*. При этом *Precision* не учитывает FN, то есть не учитывает процент обнаружения отзывов с данным классом.

1. Полнота, или способность обнаружения отзывов с данным классом:

, (4.3)

где 0 ≤ *Recall* ≤ 1. TP+FN означает общее количество отзывов, относящихся к данному классу. При этом *Recall* не учитывает FP, то есть точность обнаружения.

1. Среднее гармоническое между откликом и точностью:

, (4.4)

где 0 ≤ *мера* ≤ 1.

Результаты тестирования модели определения аспектного анализа тональности отзывов о фильмах (средние значения всех метрик приведены в Приложении А):

Таблица 4.1 – Средние значения F1-меры для каждой итерации кросс-валидации обучения модели сюжета

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Итерация  Шкала | 1 | 2 | 3 | 4 | 5 |
| Двухбалльная | 0,9356 | 0,9212 | 0,9235 | 0,9588 | 0,9206 |
| Трёхбалльная | 0,8676 | 0,8512 | 0,8632 | 0,8477 | 0,8635 |
| Пятибалльная | 0,655 | 0,6432 | 0,6211 | 0,6975 | 0,6383 |

Таблица 4.2 – Средние значения F1-меры для каждой итерации кросс-валидации обучения модели актёрской игры

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Итерация  Шкала | 1 | 2 | 3 | 4 | 5 |
| Двухбалльная | 0,928 | 0,9214 | 0,9378 | 0,9436 | 0,9133 |
| Трёхбалльная | 0,8312 | 0,8464 | 0,8427 | 0,866 | 0,8632 |
| Пятибалльная | 0,6415 | 0,64 | 0,645 | 0,6356 | 0,6423 |

Таблица 4.3 – Средние значения F1-меры для каждой итерации кросс-валидации обучения модели музыки

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Итерация  Шкала | 1 | 2 | 3 | 4 | 5 |
| Двухбалльная | 0,9287 | 0,9268 | 0,9102 | 0,9214 | 0,941 |
| Трёхбалльная | 0,8784 | 0,8649 | 0,8505 | 0,8477 | 0,8344 |
| Пятибалльная | 0,6587 | 0,6412 | 0,6479 | 0,6358 | 0,6338 |

Таблица 4.4 – Средние значения F1-меры для каждой итерации кросс-валидации обучения модели оригинальности

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Итерация  Шкала | 1 | 2 | 3 | 4 | 5 |
| Двухбалльная | 0,9263 | 0,9286 | 0,9314 | 0,9135 | 0,9252 |
| Трёхбалльная | 0,88 | 0,8749 | 0,863 | 0,8689 | 0,8234 |
| Пятибалльная | 0,6463 | 0,6774 | 0,6558 | 0,6342 | 0,6623 |

Таблица 4.5 – Средние значения F1-меры для каждой итерации кросс-валидации обучения модели зрелищности

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Итерация  Шкала | 1 | 2 | 3 | 4 | 5 |
| Двухбалльная | 0,9388 | 0,9455 | 0,9168 | 0,9278 | 0,9387 |
| Трёхбалльная | 0,8631 | 0,8734 | 0,8591 | 0,8514 | 0,879 |
| Пятибалльная | 0,6546 | 0,6595 | 0,6681 | 0,6512 | 0,6622 |

Средние значения по всем метрикам:

Рисунок 4.6 – Метрики классификации сюжета

Рисунок 4.7 – Метрики классификации музыки

Рисунок 4.8 – Метрики классификации актёрской игры

Рисунок 4.9 – Метрики классификации оригинальности

Рисунок 4.10 – Метрики классификации зрелищности

Метрика *Accuracy* не будет эффективна на несбалансированных датасетах, так как модель могла просто относить отзывы к наиболее часто встречающемуся классу и получать высокую оценку точности. Для такого случая также применяются метрики *Precision* и *Recall*. *Precision* показывает точность относительно одного класса, зависящую от количества отзывов, которые были верно и неверно помечены как отзывы, принадлежащие к данному классу. *Recall* же, наоборот, не учитывает количество неверных обнаружений, а берёт в расчёт количество необнаруженных отзывов с данным классом, то есть определяет способность обнаружения отзывов с данным классом. Для высчитывания общих метрик *Precision* и *Recall* используются *Macro-Average* и *Micro-Average* *Precision*, и *Recall*:

(4.5)

(4.6)

(4.7)

(4.8)

где *n* – количество классов. *Macro-Average* *Precision* просто считает среднее значение среди Precision всех классов, в то время как *Micro-Average* *Precision* объединяет метрики всех классов для вычисления одной средней. *Macro-Average* используется, когда нужно узнать, как модель работает в целом на всём наборе данных. *Micro-Average* используется для несбалансированных датасетов, когда количество образцов в одном классе больше или меньше, чем в другом. Например, возьмём следующий набор классов:

1. Класс A: 1 TP и 1 FP;
2. Класс B: 1 TP и 1 FP;
3. Класс C: 10 TP и 90 FP;
4. Класс D: 1 TP и 1 FP.

Тогда *PrecisionA = PrecisionB = PrecisionD = 0,5. PrecisionC = 0,1*

(4.9)

(4.10)

Как видно, *Macro-Average* *Precision* довольно неплохое, так как только один класс имеет низкую точность. А вот *Micro-Average* *Precision* уже имеет низкое значение, показывая, что имеет место несбалансированность.

Как видно по результатам тестирования на диаграммах выше и в таблицах приложения А, при пяти классах *Precision* довольно маленький (около 0,5), то есть много ложных обнаружений. При уменьшении количества классов (шкалы оценки) точность увеличивается и становится примерно равная 0,92 при двух классах. *Accuracy* ведёт себя так же, как и *Precision*: постепенно увеличивается с уменьшением шкалы оценивания. *Recall* же в любых случаях имеет высокое значение: от 0,91 до 0,95, то есть способность обнаружения отзывов с данным классом высокая. *F1 score*, которая объединяет в себе метрики *Precision* и *Recall* также увеличивается, поскольку *Recall* не изменяется, а *Precision* растёт со значения 0,65 до значения 0,93, что обозначает увеличение общей точности модели.

В итоге можно сказать, что наилучшее соотношение точности и величины шкалы оценивания достигается при трёхбалльной системе, так как все метрики в данном случае имеют достаточно высокую величину (в отличие от пятибалльной системы), и количество оценок даёт нам более понятное и полное описание аспектов отзыва. Можно сказать, что оценка «1» означает негативное отношение, оценка «2» – нейтральное, а оценка «3» – положительное.

4.4 Описание разработанной программы

На вход программы поступает путь к json-файлу со списком отзывов, оценку аспектов которых нужно дать, и шкала оценивания *S* (2, 3 или 5), которая определяет количество оценок каждого аспекта. Каждый входной отзыв имеет следующую структуру:

1. Идентификатор элемента списка (опционально);
2. Отзыв;
3. Плюсы, выделенные пользователем (опционально);
4. Минусы, выделенные пользователем (опционально);
5. Общий рейтинг фильма (опционально);
6. Рекомендация пользователя, советует ли он смотреть данный фильм («да» или «нет») (опционально).

Только одно поле является обязательным, остальные опциональны. Однако их наличие увеличивает точность предсказания аспектных оценок.

Результатом предсказания является json-файл, представляющий собой список из элементов следующей структуры:

1. Идентификатор (если был задан);
2. Оценка сюжета от 1 до *S*;
3. Оценка музыки от 1 до *S*;
4. Оценка игры актёров от 1 до *S*;
5. Оценка оригинальности от 1 до *S*;
6. Оценка зрелищности от 1 до *S*.

Изображение выглядит как текст

Автоматически созданное описаниеИзображение выглядит как текст

Автоматически созданное описание

Рисунок 4.11 – Пример входных и выходных данных

Код программы представлен в Приложении Д.

4.4 Выводы по разделу

В данном разделе были выполнены задачи разработки и оценки качества программы для аспектного анализа тональности отзывов о фильмах. Была сравнена точность модели для пяти-, трёх- и двухбалльной системы оценивания аспектов в отзыве, в результате чего было выяснено, что для всех аспектов отзыва значения всех метрик примерно одинаковы, и с уменьшением шкалы оценивания отзыва увеличивается точность определения оценки, в худшем случае имея *Accuracy* = 0,82, *Precision* = 0,48, *Recall* = 0,92, *F1* = 0,63, а в лучшем – *Accuracy* = 0,94, *Precision* = 0,93, *Recall* = 0,96, *F1* = 0,945. Также была описана программа, способ взаимодействия с ней и входные и выходные данные.

# Заключение

В ходе курсового проекта в целях формирования размеченного набора данных были рассмотрены сайты с отзывами о фильмах, выбран из них наиболее подходящий по критериям полноты информации: сайт otzovik.com, на котором есть оценки аспектов отзывов к фильмам, благодаря чему возможно более эффективно обучать модель, так как есть эталонные ответы, с которыми можно сравниваться.

После выбора сайта, с которого будут собираться отзывы для формирования датасета, был написан парсер, собирающий отзывы. В результате чего был получен 57361 отзыв и оценки аспектов (сюжет, музыка, игра актёров, оригинальность, зрелищность) для каждого из них, чего достаточно для обучения качественной модели.

Для выбора модели машинного обучения для дальнейшего написания программы был проведён обзор методов аспектного анализа тональности, в результате чего был выбран метод рекуррентной нейронной сети как наиболее подходящий под задачу обработки естественного языка и последовательных данных.

В результате поиска подходящих библиотек для написания программы для обучения нейронных сетей было решено использовать язык программирования Python, так он чаще всего используется для данных задач и, соответственно, для него есть большое количество библиотек для работы с машинным обучением.

В результате выбора средства программирования была написана программа для обучения модели рекуррентной нейронной сети и для её использования. Далее было оценено её качество при двух-, трёх- и пятибалльной системе оценивания и сделаны выводы.

Таким образом, все задачи, поставленные во введении, были выполнены, а цель достигнута.

# Библиографический список

1. Андрианов И. А. Современные методы аспектно-ориентированного анализа эмоциональной окраски / И. А. Андрианов, В. Д. Майоров, Д. Ю. Турдаков // Труды Института системного программирования РАН. – 2015. – Том 27, вып. 5. – сс. 5-21.

2. IEEE 2018 Fourteenth International Conference on Information Processing, Dec. 21st – Dec. 23rd, 2018: Aspect Based Sentiment Analysis on Product Reviews / A. P. Rodrigues, N. N. Chiplunkar. – Nitte, 2018 – 6 p.

3. Международная конференция «Диалог», 27.05.2015 – 30.05.2015: Извлечение аспектов, тональности и категорий аспектов на основании отзывов пользователей о ресторанах и автомобилях / В. В. Иванов, Е. В. Тутубалина, Н. Р. Мингазов, И. С. Алимова. – КФУ, 2015 – сс. 46-57.

4. Международная конференция «Диалог», 27.05.2015 – 30.05.2015: Высокоточный метод извлечения аспектных терминов для русского языка / В. Майоров, В. Аванесов, И. Андрианов, Н. Астраханцев. – Москва, 2015 – сс. 58-67.

5. Международная конференция «Диалог», 27.05.2015 – 30.05.2015: Sentirueval: Глубокие рекуррентные нейронные сети для аспектно-ориентированного анализа тональности отзывов пользователей на различных языках / Д. С. Тарасов. – Москва, 2015 – сс. 77-88.

6. Международная конференция «Диалог», 27.05.2015 – 30.05.2015: Sentirueval: Анализ тональности твитов о телекоммуникациях и банках на основе метода машинного обучения в рамках Sentirueval / Е. В. Тутубалина, М. А. Загулова, В. В. Иванов, В. А. Малых. – Москва, 2015 – сс. 89-99.

6. Byrne. S. Aspect Identification and Sentiment Analysis in Text-Based Reviews / S. Byrne // Lehigh Preserve Institutional Repository. – 2017. – Lehigh University – 90 p.

7. Chakraborty S. Aspect-based Sentiment Analysis of Scientific Reviews / S. Chakraborty, P. Goyal, A. Mukherjee – Electronic article. – 2020. – Access mode: https://arxiv.org/pdf/2006.03257.pdf (accessed 27 March 2022).

8. 7th International Conference on Advances in Computing & Communications, Aug. 22nd – Aug. 24th, 2017: Aspect based Sentiment Oriented Summarization of Hotel Reviews / A. P. Rodrigues, N. N. Chiplunkar. – Cochin, 2017 – 9 p.

9. Schouten K. Survey on Aspect-Level Sentiment Analysis / K. Schouten, F. Frasincar // IEEE Transactions on Knowledge, and Data Engineering. – 2016. – Vol. 28, Issue 3. – pp. 813-830.

10. Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Aug. 22nd – Aug. 25th, 2004: Mining and Summarizing Customer reviews / B. Lui, M. Hu. – Seattle, 2004 – pp. 168-177.

11. 54th Annual Meeting of the Association for Computational Linguistics, Aug. 7th – Aug. 12th, 2016: Implicit Polarity and Implicit Aspect Recognition in Opinion Mining / H. Y. Chen, H. H. Chen. – Berlin, 2016 – pp. 20-25.

12. Poria S. Aspect Extraction for Opinion Mining with a Deep Convolutional Neural Network / S. Poria, E. Cambria, A. Gelbukh // Knowledge-Based Systems, Elsevier. – 2016. – Vol. 108, Issue 15. – pp. 42-49.

13. Second Workshop on Natural Language Processing for Social Media, August, 2014: A Rule-Based Approach to Aspect Extraction from Product Reviews / S. Poria, E. Cambria, L. W. Ku, C. Gui, A. Delbukh. – Dublin, 2014 – pp. 28-37.

14. Liu Q. Automated Rule Selection for Opinion Target Extraction / Q. Liu, Z. Gao, B. Liu, Y. Zhang // Knowledge-Based Systems, Elsevier. – 2016. – Vol. 104, Issue 15. – pp. 74-88.

15. Rana T. A Two-Fold Rule-Based Model for Aspect Extraction / T. Rana, Y. Cheah // Expert Systems with Applications, Elsevier. – 2017. – Vol. 89, Issue 15. – pp. 273-285.

16. Manek A. Aspect Term Extraction for Sentiment Analysis in Large Movie Reviews Using Gini Index Feature Selection Method and SVM classifier / A. S. Manek, P. D. Shenoy, M. C. Mohan // World Wide Web. – 2017. – Vol. 20, Issue 2. – pp. 135-154.

17. Shah K. A Comparative Analysis of Logistic Regression, Random Forest and KNN Models for the Text Classifiaction / K. Shah, H. Patel, D. Sanghvi, M. Shah // Augmented Human Research. – 2020. – Vol. 5, Issue 1. – 16 p.

18. Kaur R. Naïve Bayes: A Text Classifier Based on Machine Learning / R. Kaur // International Journal of Research Publication and Reviews. – 2021. – Vol. 2, Issue 2. – pp. 260-266.

19. 55th Annual Meeting of the Association for Computational Linguistics, Jul. 30th – Aug. 4th, 2017: An Unsupervised Neural Attention Model for Aspect Extraction / R. He, W. S. Lee, H. T. Ng, D. Dahlmeier. – Vancouver, 2017 – pp. 388-397.

20. 24th International Joint Conference on Artificial Intelligence, Jul. 14th – Jul. 31st, 2015: Automated Rule Selection for Aspect Extraction in Opinion Mining / Q. Liu, Z. Gao, B. Liu, Y, Zhang. – Buenos Aires, 2016 – pp. 1291-1297.

21. Shu L. Lifelong Learning CRF for Supervised Aspect Extraction / L. Shu, H. Xu, B. Liu – Electronic article. – 2017. – Access mode: https://arxiv.org/pdf/1705.00251.pdf (accessed 04 April 2022).

22. Conference on Empirical Methods in Natural Language Processing, Oct. 25th – Oct. 29th, 2014: Convolutional Neural Network for Sentence Classification / Y. Kim. – Doha, 2014 – pp. 1746-1751.

23. 50th Annual Meeting of the Association for Computational Linguistics, Jul. 8th – Jul. 14th, 2012: Aspect Extraction through Semi-Supervised Modeling / A. Mukherjee, B. Liu. – Jeju island, 2012 – pp. 339-348.

24. 8th International Joint Conference on Natural Language Processing, Nov. 27th – Dec. 1st, 2017: MTNA: A Neural Multi-Task Model for Aspect Category Classification and Aspect Term Extraction on Restaurant Reviews / W. Xue, W. Zhou, T. Li, Q. Wang. – Taipei, 2017 – pp. 151-156.

25. Ganu G. Beyond the Stars: Improving Rating Predictions using Review Text Content / G. Ganu, N. Elhadad – Electronic article. – 2009. – Access mode: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.150.140&rep=rep1&type=pdf (accessed 04 April 2022).

26. Akhtar M. Multi-Task Learning for Aspect Term Extraction and Aspect Sentiment Classification / M. Akhtar, R. Kaur // Neurocomputing, Elsevier. – 2020. – Vol. 398, Issue 20. – pp. 247-256.

27. Tang D. Effective LSTMs for Target-Dependent Sentiment Classification / D. Tang, B. Qin, X. Feng, T. Liu – Electronic article. – 2016. – Access mode: https://arxiv.org/pdf/1512.01100.pdf (accessed 04 April 2022).

28. 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Jun. 19th – Jun. 24th, 2011: Target-Dependent Twitter Sentiment Classification / L. Jiang, M. Yu, M. Zhou, X. Liu, T. Zhao. – Portland, 2011 – pp. 151-160.

29. Xu H. Double Embeddings and CNN-Based Sequence Labeling for Aspect Extraction / H. Xu, B. Liu, L. Shu, P. S. Yu – Electronic article. – 2018. – Access mode: https://arxiv.org/pdf/1805.04601.pdf (accessed 04 April 2022).

30. 15th ACM International Conference on Information and Knowledge Management, Nov. 6th – Nov. 11th, 2006: Movie Review Mining and Summarization / L. Zhuang, F. Jing, X.-Y. Zhu. – Arlington, 2006 – pp. 43-50.

31. Elman J. Finding Structure in Time / J. Elman // Cognitive Science. – 1990. – Vol. 14, Issue 2. – pp. 179-211.

32. Hochreiter S. Long Short-Term Memory / S. Hochreiter, J. Schmidhuber // Neural Computation. – 1997 – Vol. 9, Issue 8. – pp. 1735-1780.

33. Schuster M. Bidirectional Recurrent Neural Networks / M. Schuster, K. Kuldip // IEEE Transactions on Signal Processing. – 1997 – Vol. 45, No. 11. – pp. 2673-2681.

# Приложения

Приложение А

Таблица А.1 – Метрика Accuracy

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Сюжет | Музыка | Актёрская игра | Оригинальность | Зрелищность |
| 5 баллов | 0,8248 | 0,8247 | 0,8353 | 0,8188 | 0,8143 |
| 3 балла | 0,8945 | 0,8934 | 0,8908 | 0,8868 | 0,8962 |
| 2 балла | 0,9413 | 0,92 | 0,9222 | 0,9283 | 0,9444 |

**Метрики аспекта «Сюжет»**

Таблица А.2 – Метрика Precision для шкал оценки

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Precision 1 | Precision 2 | Precision 3 | Precision 4 | Precision 5 | Macro Precision | Micro Precision |
| 5 баллов | 0,6353 | 0,4371 | 0,4682 | 0,4018 | 0,6641 | 0,5213 | 0,4914 |
| 3 балла | 0,8946 | 0,673 | 0,9089 |  |  | 0,8255 | 0,7943 |
| 2 балла | 0,9124 | 0,9459 |  |  |  | 0,9292 | 0,9354 |

Таблица А.3 – Метрика Recall для шкал оценки

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Recall 1 | Recall 2 | Recall 3 | Recall 4 | Recall 5 | Macro Recall | Micro Recall |
| 5 баллов | 0,9475 | 0,9499 | 0,9594 | 0,9596 | 0,9696 | 0,9572 | 0,9587 |
| 3 балла | 0,9456 | 0,9304 | 0,9586 |  |  | 0,9448 | 0,9345 |
| 2 балла | 0,8809 | 0,9518 |  |  |  | 0,9164 | 0,9285 |

Таблица А.4 – Метрика F1 для шкал оценки

|  |  |  |
| --- | --- | --- |
|  | Macro F1 | Micro F1 |
| 5 баллов | 0,675 | 0,65 |
| 3 балла | 0,8811 | 0,8587 |
| 2 балла | 0,9228 | 0,9319 |

**Метрики аспекта «Музыка»**

Таблица А.5 – Метрика Precision для шкал оценки

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Precision 1 | Precision 2 | Precision 3 | Precision 4 | Precision 5 | Macro Precision | Micro Precision |
| 5 баллов | 0,6174 | 0,4211 | 0,4648 | 0,4134 | 0,6818 | 0,5197 | 0,4864 |
| 3 балла | 0,874 | 0,6714 | 0,9282 |  |  | 0,8245 | 0,7863 |
| 2 балла | 0,8952 | 0,9584 |  |  |  | 0,9268 | 0,9291 |

Таблица А.6 – Метрика Recall для шкал оценки

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Recall 1 | Recall 2 | Recall 3 | Recall 4 | Recall 5 | Macro Recall | Micro Recall |
| 5 баллов | 0,9362 | 0,9371 | 0,9602 | 0,651 | 0,9789 | 0,8927 | 0,9506 |
| 3 балла | 0,9324 | 0,9316 | 0,9645 |  |  | 0,9428 | 0,9296 |
| 2 балла | 0,8712 | 0,961 |  |  |  | 0,9161 | 0,9222 |

Таблица А.7 – Метрика F1 для шкал оценки

|  |  |  |
| --- | --- | --- |
|  | Macro F1 | Micro F1 |
| 5 баллов | 0,6569 | 0,6435 |
| 3 балла | 0,8797 | 0,852 |
| 2 балла | 0,9214 | 0,9256 |

**Метрики аспекта «Актёрская игра»**

Таблица А.8 – Метрика Precision для шкал оценки

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Precision 1 | Precision 2 | Precision 3 | Precision 4 | Precision 5 | Macro Precision | Micro Precision |
| 5 баллов | 0,6199 | 0,4201 | 0,4623 | 0,4128 | 0,6881 | 0,5206 | 0,4812 |
| 3 балла | 0,8774 | 0,6694 | 0,9309 |  |  | 0,8259 | 0,7834 |
| 2 балла | 0,9003 | 0,9628 |  |  |  | 0,9316 | 0,9336 |

Таблица А.9 – Метрика Recall для шкал оценки

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Recall 1 | Recall 2 | Recall 3 | Recall 4 | Recall 5 | Macro Recall | Micro Recall |
| 5 баллов | 0,9387 | 0,9352 | 0,9602 | 0,6502 | 0,9834 | 0,8935 | 0,9592 |
| 3 балла | 0,9324 | 0,9298 | 0,9695 |  |  | 0,9439 | 0,9282 |
| 2 балла | 0,8765 | 0,9674 |  |  |  | 0,9219 | 0,924 |

Таблица А.10 – Метрика F1 для шкал оценки

|  |  |  |
| --- | --- | --- |
|  | Macro F1 | Micro F1 |
| 5 баллов | 0,6579 | 0,6409 |
| 3 балла | 0,881 | 0,85 |
| 2 балла | 0,9267 | 0,9288 |

**Метрики аспекта «Оригинальность»**

Таблица А.11 – Метрика Precision для шкал оценки

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Precision 1 | Precision 2 | Precision 3 | Precision 4 | Precision 5 | Macro Precision | Micro Precision |
| 5 баллов | 0,6402 | 0,4365 | 0,4681 | 0,4036 | 0,6653 | 0,5227 | 0,4976 |
| 3 балла | 0,8959 | 0,6722 | 0,9076 |  |  | 0,8252 | 0,7967 |
| 2 балла | 0,8893 | 0,9344 |  |  |  | 0,9118 | 0,9195 |

Таблица А.12 – Метрика Recall для шкал оценки

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Recall 1 | Recall 2 | Recall 3 | Recall 4 | Recall 5 | Macro Recall | Micro Recall |
| 5 баллов | 0,9479 | 0,9508 | 0,965 | 0,9618 | 0,9711 | 0,9593 | 0,959 |
| 3 балла | 0,9466 | 0,9308 | 0,9573 |  |  | 0,9449 | 0,9389 |
| 2 балла | 0,8828 | 0,9497 |  |  |  | 0,9163 | 0,9308 |

Таблица А.13 – Метрика F1 для шкал оценки

|  |  |  |
| --- | --- | --- |
|  | Macro F1 | Micro F1 |
| 5 баллов | 0,6767 | 0,6552 |
| 3 балла | 0,881 | 0,862 |
| 2 балла | 0,914 | 0,925 |

**Метрики аспекта «Зрелищность»**

Таблица А.14 – Метрика Precision для шкал оценки

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Precision 1 | Precision 2 | Precision 3 | Precision 4 | Precision 5 | Macro Precision | Micro Precision |
| 5 баллов | 0,6512 | 0,4316 | 0,471 | 0,4081 | 0,6689 | 0,5262 | 0,5014 |
| 3 балла | 0,8985 | 0,6749 | 0,9102 |  |  | 0,8279 | 0,8007 |
| 2 балла | 0,8916 | 0,935 |  |  |  | 0,9133 | 0,9268 |

Таблица А.15 – Метрика Recall для шкал оценки

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Recall 1 | Recall 2 | Recall 3 | Recall 4 | Recall 5 | Macro Recall | Micro Recall |
| 5 баллов | 0,9487 | 0,9516 | 0,9645 | 0,9637 | 0,9798 | 0,9617 | 0,9614 |
| 3 балла | 0,9474 | 0,9369 | 0,9597 |  |  | 0,948 | 0,9411 |
| 2 балла | 0,884 | 0,953 |  |  |  | 0,9185 | 0,9377 |

Таблица А.16 – Метрика F1 для шкал оценки

|  |  |  |
| --- | --- | --- |
|  | Macro F1 | Micro F1 |
| 5 баллов | 0,6802 | 0,6591 |
| 3 балла | 0,8839 | 0,8652 |
| 2 балла | 0,9159 | 0,9322 |

Приложение Б

**Парсер сайта «Otzovik.com»**

public class OtzovikParser {  
 private static boolean *objectClosed* = true;  
 private static final Logger *logger* = LoggerFactory.*getLogger*(OtzovikParser.class);  
 private static final String *url* = "https://otzovik.com/show\_filter.php?cat\_id=117&order=rate&f[r]=%d\_&page=%d";  
 private static Map<String,String> *additionalEntries*;  
  
 public static void parse(int moviesCount, final int reviewsPerMovie, final int minimumReviewsPerMovie,  
 final String outputPath, final String datasetStatisticsPath) throws IOException {  
 *additionalEntries* = Map.*of*("referer", String.*format*("https://otzovik.com/show\_filter.php?cat\_id=117&order=date\_desc&f[r]=%d\_&page=", minimumReviewsPerMovie));  
 final Statistics statistics = new Statistics();  
 Map<String, String> headers = *readHeaders*();  
  
 JsonWriter writer = null;  
 try {  
 writer = new JsonWriter(new FileWriter(outputPath));  
 writer.beginArray();  
 *logger*.debug(String.*format*("Парсинг стартовал. Примерное время ожидания %.2f - %.2f мин.",  
 0.75 \* moviesCount \* reviewsPerMovie / 60,  
 0.85 \* moviesCount \* reviewsPerMovie / 60)  
 );  
 for (int page = 1; moviesCount > 0; ++page) {  
 final String referer = headers.get("referer");  
 headers.put("referer", referer.substring(referer.lastIndexOf('=') + 1) + page);  
 Document doc;  
 try {  
 doc = Jsoup.*connect*(String.*format*(*url*, minimumReviewsPerMovie, page))  
 .timeout(5000)  
 .ignoreContentType(true)  
 .method(Connection.Method.*GET*)  
 .maxBodySize(Integer.*MAX\_VALUE*)  
 .headers(headers)  
 .get();  
 } catch (Exception e) {  
 ((Runnable) Toolkit.*getDefaultToolkit*().getDesktopProperty("win.sound.exclamation")).run();  
 *logger*.error("Ошибка при парсинге отзыва. Парсер уснул на 60 секунд. Смените прокси!");  
 if (!*objectClosed*) {  
 writer.endObject();  
 *objectClosed* = true;  
 }  
 ((Runnable) Toolkit.*getDefaultToolkit*().getDesktopProperty("win.sound.exclamation")).run();  
 Thread.*sleep*(60000);  
 headers = *readHeaders*();  
 continue;  
 }  
 Thread.*sleep*(600);  
  
 final Elements movies = doc.getElementsByClass("product-name");  
 final int size = Math.*min*(movies.size(), moviesCount);  
 if (size == 0) {  
 break;  
 }  
 for (int i = 0; i < size; ++i) {  
 try {  
 *parseReviewsPage*(movies  
 .get(i)  
 .attributes()  
 .get("href"),  
 headers,  
 reviewsPerMovie,  
 writer,  
 statistics);  
 } catch (Exception exc) {  
 ((Runnable) Toolkit.*getDefaultToolkit*().getDesktopProperty("win.sound.exclamation")).run();  
 *logger*.error("Ошибка при парсинге отзыва. Парсер уснул на 60 секунд. Смените прокси!");  
 if (!*objectClosed*) {  
 writer.endObject();  
 *objectClosed* = true;  
 }  
 ((Runnable) Toolkit.*getDefaultToolkit*().getDesktopProperty("win.sound.exclamation")).run();  
 Thread.*sleep*(60000);  
 headers = *readHeaders*();  
 }  
 }  
 moviesCount -= size;  
 }  
 } catch (Exception anyExc) {  
 *logger*.error("При парсинге возникла ошибка!", anyExc);  
 } finally {  
 *closeWriter*(writer);  
  
 *writeStatistics*(statistics, datasetStatisticsPath);  
  
 *logger*.debug("Парсинг закончен");  
 }  
 }  
  
 private static void parseReviewsPage(final String url, Map<String, String> headers, int reviewsPerMovie,  
 final JsonWriter writer, final Statistics statistics) throws IOException, InterruptedException {  
 final String refererBase = "https://otzovik.com" + url;  
 *logger*.debug(String.*format*("\tПарсинг страницы %s", refererBase));  
 headers.put("referer", refererBase + "1/");  
  
 for (int page = 2; reviewsPerMovie > 0; ++page) {  
 headers.put("referer", refererBase + page + "/");  
 final Document doc = Jsoup.*connect*(refererBase + page)  
 .timeout(5000)  
 .ignoreContentType(true)  
 .method(Connection.Method.*GET*)  
 .maxBodySize(Integer.*MAX\_VALUE*)  
 .headers(headers)  
 .get();  
  
 final Elements reviews = doc.getElementsByClass("review-btn review-read-link");  
 final int size = Math.*min*(reviews.size(), reviewsPerMovie);  
 if (size == 0) {  
 break;  
 }  
 reviewsPerMovie -= size;  
 for (int i = 0; i < size; ++i) {  
 ++statistics.moviesCount;  
  
 *logger*.debug(String.*format*("\t\tПарсинг отзыва https://otzovik.com%s (%d)",  
 url, statistics.moviesCount));  
 try {  
 *parseAndSaveReview*(reviews  
 .get(i)  
 .attributes()  
 .get("href"),  
 headers,  
 writer,  
 statistics);  
 } catch (Exception exc) {  
 --statistics.moviesCount;  
 ((Runnable) Toolkit.*getDefaultToolkit*().getDesktopProperty("win.sound.exclamation")).run();  
 *logger*.error("Ошибка при парсинге отзыва. Парсер уснул на 60 секунд. Смените прокси!");  
 if (!*objectClosed*) {  
 writer.endObject();  
 *objectClosed* = true;  
 }  
 ((Runnable) Toolkit.*getDefaultToolkit*().getDesktopProperty("win.sound.exclamation")).run();  
 Thread.*sleep*(60000);  
 headers = *readHeaders*();  
 }  
 }  
 }  
 }  
  
 private static void parseAndSaveReview(final String url, final Map<String, String> headers,  
 final JsonWriter jsonWriter, final Statistics statistics) throws IOException, InterruptedException {  
 final Document review = Jsoup.*connect*("https://otzovik.com" + url)  
 .timeout(5000)  
 .ignoreContentType(true)  
 .method(Connection.Method.*GET*)  
 .maxBodySize(Integer.*MAX\_VALUE*)  
 .headers(headers)  
 .get();  
 Thread.*sleep*(600);  
  
 jsonWriter.beginObject();  
 jsonWriter.name("id").value(url.substring(url.indexOf('\_') + 1, url.lastIndexOf('.')));  
  
 final String reviewPlus = review.getElementsByClass("review-plus").text();  
 jsonWriter.name("plus").value(reviewPlus);  
 statistics.pluses.lengths.add(reviewPlus.length());  
  
 final String reviewMinus = review.getElementsByClass("review-minus").text();  
 jsonWriter.name("minus").value(reviewMinus);  
 statistics.minuses.lengths.add(reviewMinus.length());  
  
 final String reviewBody = review.getElementsByClass("review-body description").text();  
 jsonWriter.name("body").value(reviewBody);  
 statistics.body.lengths.add(reviewBody.length());  
  
 jsonWriter.name("recommend").value(review  
 .select("td[class~=recommend-ratio]")  
 .text()  
 );  
  
 final Elements aspects = review.getElementsByClass("rating-item tooltip-top");  
 Integer plot = null, spectacularity = null, music = null, actors = null, originality = null;  
 int spaceIndex;  
 for (int i = 0; i < aspects.size(); ++i) {  
 final String aspectText = aspects.get(i)  
 .attributes()  
 .get("title");  
 spaceIndex = aspectText.indexOf(':') + 1;  
 int grade = Integer.*parseInt*(aspectText  
 .substring(spaceIndex + 1, spaceIndex + 2));  
 if (i == 0) {  
 plot = grade;  
 } else if (i == 1) {  
 spectacularity = grade;  
 } else if (i == 2) {  
 actors = grade;  
 } else if (i == 3) {  
 originality = grade;  
 } else if (i == 4) {  
 music = grade;  
 }  
 }  
 jsonWriter.name("plot").value(plot);  
 statistics.plot.incRating(plot);  
 jsonWriter.name("spectacularity").value(spectacularity);  
 statistics.spectacularity.incRating(spectacularity);  
 jsonWriter.name("music").value(music);  
 statistics.music.incRating(music);  
 jsonWriter.name("actors").value(actors);  
 statistics.actors.incRating(actors);  
 jsonWriter.name("originality").value(originality);  
 statistics.originality.incRating(originality);  
  
 final Integer rating = Integer.*parseInt*(  
 review.getElementsByClass("rating")  
 .get(0)  
 .attributes()  
 .get("title"));  
 jsonWriter.name("rating").value(rating);  
 statistics.rating.incRating(rating);  
  
 jsonWriter.endObject();  
 *objectClosed* = true;  
 }  
  
 private static void writeStatistics(final Statistics statistics, final String path) throws IOException {  
 try (JsonWriter statisticsWriter = new JsonWriter(new FileWriter(path))) {  
 statisticsWriter.beginObject();  
 statisticsWriter.name("moviesProcessed").value(statistics.moviesCount);  
 statisticsWriter.name("minuses");  
 statistics.minuses.writeToJson(statisticsWriter);  
 statisticsWriter.name("pluses");  
 statistics.pluses.writeToJson(statisticsWriter);  
 statisticsWriter.name("body");  
 statistics.body.writeToJson(statisticsWriter);  
 statisticsWriter.name("rating");  
 statistics.rating.writeToJson(statisticsWriter);  
 statisticsWriter.name("actors");  
 statistics.actors.writeToJson(statisticsWriter);  
 statisticsWriter.name("music");  
 statistics.music.writeToJson(statisticsWriter);  
 statisticsWriter.name("originality");  
 statistics.originality.writeToJson(statisticsWriter);  
 statisticsWriter.name("spectacularity");  
 statistics.spectacularity.writeToJson(statisticsWriter);  
 statisticsWriter.name("plot");  
 statistics.plot.writeToJson(statisticsWriter);  
 statisticsWriter.endObject();  
 }  
 }  
  
 private static void closeWriter(JsonWriter writer) {  
 if (writer != null) {  
 try {  
 writer.endArray();  
 writer.close();  
 } catch (Exception writerExc) {  
 *logger*.error("Не удалось корректно закончить файл", writerExc);  
 }  
 }  
 }  
  
 private static Map<String, String> readHeaders() throws IOException {  
 List<String> fileLines;  
 try (final BufferedReader reader = new BufferedReader(new FileReader("headers.txt"))) {  
 fileLines = reader.lines().collect(Collectors.*toList*());  
 }  
 HashMap<String, String> headers = new HashMap<>();  
 for (final String line : fileLines) {  
 if ("throwException".equals(line)) {  
 throw new IllegalArgumentException();  
 }  
 final String[] split = line.split("\\|");  
 headers.put(split[0], split[1]);  
 }  
 headers.putAll(*additionalEntries*);  
 return headers;  
 }  
}

Приложение В

**Преобразование текстового датасета в числовой**

nlp = spacy.load("ru\_core\_news\_sm")  
stemmer = SnowballStemmer("russian")

def generate\_dataset(path\_to\_text\_dataset, path\_to\_save\_numeric\_dataset, aspect, grade\_scale, threads\_num, review\_length=250):  
 with open(path\_to\_text\_dataset, "r", encoding="utf-8") as dataset:  
 processed\_reviews, y\_res = get\_list\_of\_reviews\_and\_y\_res(json.load(dataset), aspect, grade\_scale)  
 processed\_reviews = pos\_tag\_reviews(processed\_reviews, threads\_num)  
 processed\_reviews = clean\_reviews(processed\_reviews, threads\_num)  
 processed\_reviews = cut\_reviews(processed\_reviews, review\_length)  
 dictionary, X\_res = encode\_frequent(processed\_reviews, threads\_num)  
 print("Vocab size: " + str(len(dictionary)))  
  
 zipped = list(zip(X\_res, y\_res))  
 random.shuffle(zipped)  
 X\_res, y\_res = zip(\*zipped)  
  
 write\_to\_file(X\_res, y\_res, path\_to\_save\_numeric\_dataset)  
 save\_dictionary("dictionary.txt", dictionary)

def pos\_tag\_review(review):  
 tagged = []  
 tokens = nlp(review)  
 for token in tokens:  
 tagged.append([token.lemma\_, token.pos\_])  
 return tagged  
  
def pos\_tag\_reviews(reviews, threads\_num):  
 return Parallel(n\_jobs=threads\_num)(delayed(pos\_tag\_review)(review) for review in reviews)

def clean\_review(review):  
 cleaned\_review = []  
 for word in review:  
 if (len(word[0]) == 1 and word[0].isdigit()) or (len(word[0]) > 1 and word[1] in ['PROPN', 'ADJ', 'VERB', 'NOUN']):  
 cleaned\_review.append(stemmer.stem(word[0]))  
 return cleaned\_review  
  
def clean\_reviews(reviews, threads\_num):  
 return Parallel(n\_jobs=threads\_num)(delayed(clean\_review)(review) for review in reviews)

def cut\_reviews(reviews, length):  
 for i in range(len(reviews)):  
 if len(reviews[i]) > length:  
 reviews[i] = reviews[i][:length]  
 return reviews

def parallel\_encode(review, dictionary):  
 encoded = []  
 for word in review:  
 if word in dictionary:  
 encoded.append(dictionary[word])  
 return encoded  
  
  
def encode\_frequent(reviews, threads\_num):  
 dictionary = create\_sorted\_dictionary(reviews)  
 encoded\_reviews = Parallel(n\_jobs=threads\_num)(delayed(parallel\_encode)(review, dictionary) for review in reviews)  
 return dictionary, encoded\_reviews

def write\_to\_file(X\_res, y\_res, save\_path):  
 with open(save\_path, "w", encoding="utf-8") as save\_file:  
 for x in X\_res:  
 for el in x:  
 save\_file.write(str(el))  
 save\_file.write(" ")  
 save\_file.write("\n")  
 save\_file.write("\n")  
 for y in y\_res:  
 for star in y:  
 save\_file.write(str(float(star)))  
 save\_file.write(" ")  
 save\_file.write("\n")

def save\_dictionary(save\_path, dictionary):  
 with open(save\_path, "w", encoding="utf-8") as save\_file:  
 for item in dictionary.items():  
 save\_file.write(item[0] + " " + str(item[1]))  
 save\_file.write("\n")

Приложение Г

**Программа для обучения модели**

if \_\_name\_\_ == "\_\_main\_\_":  
 print("Time of start: " + datetime.now().strftime("%H:%M:%S"))  
  
 aspects = ["plot", "music", "actors", "originality", "spectacularity"]  
 scales = ["2", "3", "5"]  
  
 for aspect in aspects:  
 for scale in scales:  
 X\_dataset, y\_dataset = load\_dataset(f"grade\_{scale}/digit\_datasets/{aspect}.txt")  
  
 split = 1  
 results = []  
 kFold = StratifiedKFold(n\_splits=5, shuffle=True, random\_state=42)  
 for train, test in kFold.split([0] \* len(X\_dataset), [0] \* len(X\_dataset)):  
 X\_train = asarray(sublist(X\_dataset, train))  
 y\_train = asarray(sublist(y\_dataset, train))  
 X\_test = asarray(sublist(X\_dataset, test))  
 y\_test = asarray(sublist(y\_dataset, test))  
 get\_statistics(y\_train, y\_test, f"stats\_{aspect}\_{scale}\_{split}.json")  
 split += 1  
  
 words\_per\_review = 250  
  
 X\_train = pad\_sequences(X\_train, maxlen=words\_per\_review)  
 X\_test = pad\_sequences(X\_test, maxlen=words\_per\_review)  
 rnn = Sequential()  
 rnn.add(Input(shape=(250,)))  
 rnn.add(Embedding(input\_dim=39731, output\_dim=words\_per\_review, input\_length=words\_per\_review))  
 rnn.add(Bidirectional(LSTM(units=words\_per\_review // 2, return\_sequences=True, dropout=0.2, recurrent\_dropout=0.2)))  
 rnn.add(GlobalMaxPool1D())  
 rnn.add(Dense(units=int(scale), activation='sigmoid'))  
  
 metric = [metrics.CategoricalAccuracy(), metrics.Precision(), metrics.Recall()]  
 for i in range(int(scale)):  
 metric.append(metrics.Precision(class\_id=i))  
 metric.append(metrics.Recall(class\_id=i))  
  
 rnn.compile(optimizer='adam',  
 loss='categorical\_crossentropy',  
 metrics=metric)  
 rnn.summary()  
 chkpt = ModelCheckpoint(f'grade\_{scale}/models/{aspect}\_{split}\_best',  
 monitor='val\_categorical\_accuracy',  
 verbose=1,  
 save\_best\_only=True,  
 mode='auto')  
  
 rnn.fit(X\_train, y\_train, epochs=20, batch\_size=32, chkpt=[chkpt])  
 rnn.save(f'grade\_{scale}/models/{aspect}\_{split}')  
  
 results.append(rnn.evaluate(X\_test, y\_test))  
  
 print(results)  
  
 print("Time of end: " + datetime.now().strftime("%H:%M:%S"))

Приложение Д

**Программа для аспектного анализа отзывов**

import json  
from threading import Thread  
  
from keras import models  
from numpy import ndarray  
from numpy.typing import NDArray  
from reviews\_processor import read\_dictionary, index\_of\_max, load\_reviews\_dataset  
  
*# Переменные окружения*path\_to\_reviews = "reviews/test.json"  
path\_to\_dictionary = "dictionary.txt"  
threads\_num = 4  
grade\_scale = 3  
res\_path = "res.json"  
  
  
def predict\_aspect(reviews: NDArray[NDArray[int]], aspect: str, pred\_res: list):  
 pred\_res.append(models.load\_model(f"grade\_{grade\_scale}/models/{aspect}\_best")  
 .predict(reviews))  
  
  
*# Функция предсказания аспектов отзывов*def predict(reviews: NDArray[NDArray[int]]) -> list[dict[str, int]]:  
 plot\_predictions = []  
 music\_predictions = []  
 actors\_predictions = []  
 originality\_predictions = []  
 spectacularity\_predictions = []  
 plot\_thread = Thread(target=predict\_aspect, args=[reviews, "plot", plot\_predictions])  
 music\_thread = Thread(target=predict\_aspect, args=[reviews, "music", music\_predictions])  
 actors\_thread = Thread(target=predict\_aspect, args=[reviews, "actors", actors\_predictions])  
 originality\_thread = Thread(target=predict\_aspect, args=[reviews, "originality", originality\_predictions])  
 spectacularity\_thread = Thread(target=predict\_aspect, args=[reviews, "spectacularity", spectacularity\_predictions])  
 plot\_thread.start()  
 music\_thread.start()  
 actors\_thread.start()  
 originality\_thread.start()  
 spectacularity\_thread.start()  
 plot\_thread.join()  
 music\_thread.join()  
 actors\_thread.join()  
 originality\_thread.join()  
 spectacularity\_thread.join()  
  
 res = []  
 for i in range(len(plot\_predictions[0])):  
 res.append({  
 "plot": index\_of\_max(plot\_predictions[0][i]) + 1,  
 "music": index\_of\_max(music\_predictions[0][i]) + 1,  
 "actors": index\_of\_max(actors\_predictions[0][i]) + 1,  
 "originality": index\_of\_max(originality\_predictions[0][i]) + 1,  
 "spectacularity": index\_of\_max(spectacularity\_predictions[0][i]) + 1,  
 })  
  
 return res  
  
  
def main():  
 dictionary = read\_dictionary(path\_to\_dictionary)  
 dataset = load\_reviews\_dataset(path\_to\_reviews, dictionary, 250, threads\_num)  
 with open(res\_path, "w", encoding="utf-8") as file:  
 json.dump(predict(dataset), file)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

1. <https://github.com/michmech/lemmatization-lists/> [↑](#footnote-ref-1)
2. <https://github.com/natasha/nerus> [↑](#footnote-ref-2)