

**МИНОБРНАУКИ РОССИИ**  
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**  
**Кафедра ВТ**

**ПРАКТИЧЕСКАЯ ЗАДАНИЕ**  
**по дисциплине «Анализ потоковых данных»**  
**Тема: Потоковая обработка тональности комментариев пользователей**  
**нейронной сетью BERT**

Студент гр.7307

\_\_\_\_\_

Державин Д.П.

Преподаватель

\_\_\_\_\_

Бекенева Я.А.

Санкт-Петербург

2022

## **АННОТАЦИЯ**

В данной работе приводится описание разработанной системы потоковой обработки тональности комментариев пользователей нейронной сетью BERT. При разработке системы были использованы следующие технологии: Python 3.8, PyTorch, FastAPI, SQLAlchemy, Alembic, RabbitMQ, Celery и PostgreSQL.

## **SUMMARY**

This paper describes the developed system for streaming processing of user comment tones by a BERT neural network. The following technologies were used to develop the system: Python 3.8, PyTorch, FastAPI, SQLAlchemy, Alembic, RabbitMQ, Celery, and PostgreSQL.

## СОДЕРЖАНИЕ

<b>ВВЕДЕНИЕ .....</b>	<b>5</b>
<b>1. ПОСТАНОВКА ЗАДАЧИ.....</b>	<b>6</b>
<b>2. ОПИСАНИЕ МЕТОДА РЕШЕНИЯ ЗАДАЧИ.....</b>	<b>7</b>
2.1 Нейронная сеть BERT .....	7
2.2 Датасет IMDB .....	8
2.3 Тренировка модели нейронной сети .....	8
2.4 RabbitMQ .....	8
2.5 Celery.....	9
2.6 Веб-приложение потоковой обработки комментариев .....	10
<b>3. АНАЛИЗ МЕТОДА РЕШЕНИЯ ЗАДАЧИ .....</b>	<b>11</b>
3.1 Рабочее окружение тестирования решения .....	11
3.2 Анализ точности модели нейронной сети .....	11
3.3 Анализ производительности системы .....	11
<b>ЗАКЛЮЧЕНИЕ.....</b>	<b>13</b>

## ВВЕДЕНИЕ

Анализ тональности текста – задача компьютерной лингвистики, которая заключается в определении эмоциональной окраски текста и выявлении эмоциональной оценки авторов по отношению к объектам, описываемым в тексте. Одним из приложений анализа тональности текста является масштабная обработка мнений пользователей социальных сетей, результаты которой используются для агрегации отзывов, мониторинга СМИ и предсказания результатов социальных и экономических явлений.

Масштабная обработка комментариев пользователей социальных сетей подразумевает условия, при которых данные в систему, например, веб-приложение, поступают непрерывно и в большом объёме. В подобных обстоятельствах требуются методы и решения, позволяющие правильно распределить нагрузку на вычислительную систему. Направление в области информационных технологий, в рамках которого занимаются решением данных задач называется потоковой обработкой данных.

**Целью** данной работы является разработка системы потоковой обработки тональности комментариев пользователей нейронной сетью BERT.

Для достижения поставленной цели необходимо решить следующие **задачи**:

- изучить технологию BERT;
- выбрать данные для обучения нейронной сети;
- обучить нейронную сеть BERT с оптимальными параметрами;
- выбрать технологии для взаимодействия с очередью обработки данных;
- проанализировать производительность нейронной сети при потоковой обработке данных.

**Объектом исследования** является нейронная сеть BERT.

**Предмет исследования** – производительность обработки комментариев нейронной сетью в системе потоковой обработки данных.

## **1. ПОСТАНОВКА ЗАДАЧИ**

Модель на основе нейронной сети BERT должна принимать на вход текст, содержанием которого является комментарий пользователя социальных сетей, и на выходе возвращать результат оценки тональности комментария: положительный или отрицательный. Естественный язык произвольный. Комментарии пользователей должны обрабатываться непрерывно в режиме реального времени или в условиях, приближённых к условиям реального времени.

## **2. ОПИСАНИЕ МЕТОДА РЕШЕНИЯ ЗАДАЧИ**

### **2.1 Нейронная сеть BERT**

BERT – нейронная сеть, разработанная корпорацией Google в 2018 году. BERT основана на архитектуре нейронной сети Transformer и предназначена для предобучения языковых представлений с целью их последующего применения в широком спектре задач NLP (Natural Language Processing – обработка естественного языка), например, ответы на вопросы, генерация аннотаций к текстам, генерация диалогов, классификация тональности и тематики текста и др. BERT со значительным отрывом превзошла существовавшие на тот момент методы NLP и стала решением State-of-the-Art в данной области. На сегодняшний день существуют различные модификации BERT: RoBERTa, SiBERT, MoEBERT. Тем не менее, классическая архитектура BERT до сих пор является актуальной.

BERT использует Transfer learning, что позволяет использовать её основную языковую модель, дополнительно обучив под свои конкретные задачи. Чтобы применять BERT для решения задач NLP, на первом этапе BERT обучают (pre-training) на очень большом корпусе конкретного языка. Например, авторы BERT в качестве корпуса для обучения языковой модели английского языка использовали BookCorpus (800 миллионов слов) и английскую версию Википедии (2,5 миллиарда слов). Данный процесс требует огромных вычислительных ресурсов. Так для создания корпуса языковой модели английского языка корпорация Google обучала BERT на 16 Cloud TPU в течение 4-ёх дней. На втором, заключительном этом, в зависимости от конкретной задачи NLP архитектуру BERT дополняют последующими слоями сетей прямого распространения и дообучают (fine-tuning). Этап дообучения не требует огромных вычислительных ресурсов. Например, модель BERT SQuAD была обучена на одном Cloud TPU в течение 30 минут до значения точности F1-метрики равной 91%.

## **2.2 Датасет IMDB**

В качестве датасета для тренировки нейронной сети BERT и оценки её точности был выбран датасет IMDB «Large Movie Review Dataset». Датасет состоит из 50 тысяч комментариев, по 25 тысяч комментариев на тренировку и тестирование нейронной сети. У каждого комментария есть метка, значение которой может быть только «positive» (положительный) или «negative» (отрицательный).

## **2.3 Тренировка модели нейронной сети**

Нейронная сеть была реализована с использованием фреймворка PyTorch. Полная архитектура нейронной сети представляет собой структуру из двух блоков: блок BERT и слой GRU. Параметры GRU: 128 входных нейронов, 1 выходной нейрон, сеть двунаправленная, дропаут равен 0,25.

Рабочее окружение обучения нейронной сети совпадает с окружением её тестирования, которое описано в пункте 3.1. Количество комментариев в тренировочной выборке составило 17500, в проверочной – 7500. Количество эпох – 4, размер батча – 32, время обучения – 3,5 часа.

## **2.4 RabbitMQ**

В большинстве случаев системы потоковой обработки данных не обходятся без шины обмена данными между компонентами системы. Для обеспечения переносимости шины обмена данными между операционными системами и вычислительными устройствами были разработаны различные протоколы для её реализации. Одним из таких протоколов является AMQP.

AMQP – открытый протокол прикладного уровня TCP/IP для передачи сообщений между компонентами системы. Основная идея состоит в том, что отдельные подсистемы (или независимые приложения) могут обмениваться произвольным образом сообщениями через AMQP-брокер, который осуществляет маршрутизацию, возможно гарантирует доставку, распределение потоков данных, подписку на нужные типы сообщений.



AMQP основан на трёх понятиях: message (сообщение), queue (очередь), exchange (точка обмена). Сообщение – это единица передаваемых данных. Очередь – коллекция сообщений. Сообщения хранятся в очереди до тех пор, пока они не будут забраны клиентами. Точка обмена – место, в которое первоначально поступают сообщения от клиентов брокеру. Точка обмена отвечает за распределение сообщений в одну или несколько очередей. При этом сообщения в точке обмена не хранятся. Точки обмена бывают трёх типов: fanout, direct и topic. Fanout распространяет сообщения по всем очередям, direct – в очередь с именем, совпадающим с ключом маршрутизации, topic – в очереди, для которых совпадает маска на ключ маршрутизации.

RabbitMQ – это программный брокер сообщений с открытым исходным кодом на основе стандарта AMQP, написанный на языке программирования Erlang. RabbitMQ маршрутизирует сообщения по всем базовым принципам протокола AMQP, описанным в спецификации. Отправитель передает сообщение брокеру, а брокер доставляет его получателю. RabbitMQ реализует и дополняет протокол AMQP. В состав RabbitMQ входит сервер, библиотеки поддержки протоколов HTTP, XMPP и STOMP, клиентские библиотеки и различные плагины, например, для мониторинга и управления по протоколу HTTP. RabbitMQ поддерживает горизонтальное масштабирование для построения кластерных решений.

## **2.5 Celery**

Celery – это распределённая асинхронная очередь задач с открытым исходным кодом или очередь задач, основанная на распределенной передаче сообщений. Celery поддерживает планирование задач, но основное внимание уделяется операциям в реальном времени. Библиотека Celery написана на языке программирования Python, и программный вызов задач доступен только на Python. Тем не менее, задача, реализованная на другом языке программирования, может быть вызвана с помощью HTTP запросов GET и POST. Рекомендуемыми брокерами сообщений для Celery являются RabbitMQ и Redis.

## **2.6 Веб-приложение потоковой обработки комментариев**

Реализованная в данной курсовой работе система потоковой обработки комментариев является веб-приложением и представляет собой веб-сервер, написанный с применением стека технологий FastAPI, SQLAlchemy и Alembic, брокер сообщений RabbitMQ, очередь фоновых задач, основанную на Celery, фоновую задачу оценки тональности комментариев и базу данных PostgreSQL. FastAPI – это веб-фреймворк для создания WEB API. SQLAlchemy – это ORM-фреймворк для взаимодействия с различными источниками данных, в основном с базами данных. Alembic – это инструмент для миграции базы данных, используемый SQLAlchemy. Задача оценки тональности комментариев – это задача Celery, которая обрабатывает комментарии батчами с помощью нейронной сети BERT.

Пользователями системы выступают HTTP-клиенты. Ниже перечислены основные функции системы:

- принимать запрос на публикацию новости и сохранять её в базу данных;
- принимать запрос на получение списка новостей;
- принимать запрос HTTP-клиента на публикацию комментария к новости и сохранять его в базу данных;
- принимать запрос на получение списка комментариев к новостям;
- оценивать тональность комментариев пользователей и сохранять результат в базу данных.

Система предназначена для обработки комментариев только на английском языке.

### 3. АНАЛИЗ МЕТОДА РЕШЕНИЯ ЗАДАЧИ

#### 3.1 Рабочее окружение тестирования решения

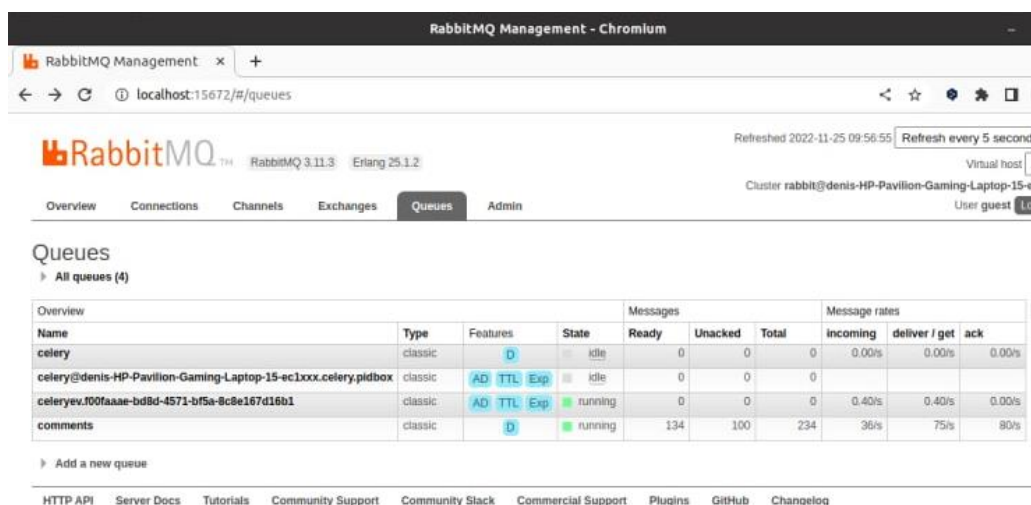
Обученная модель нейронной сети BERT была протестирована на ноутбуке с операционной системой UBUNTU 20.04. Технические характеристики ноутбука: CPU – AMD Ryzen 5 4600H with Radeon Graphics 3.00 GHz, GPU – NVIDIA GeForce 1650.

#### 3.2 Анализ точности модели нейронной сети

Тестирование точности проводилось на тестовой выборке из датасета IMDB «Large Movie Review Dataset», размер которой составил 25000 комментариев. Точность нейронной сети составила 91%.

#### 3.3 Анализ производительности системы

Для тестирования производительности была написана программа на языке Python, которая в 6 процессов отправляла комментарии к одной новости. Сервер был настроен для работы в 4 процесса. Настройки Postgres по умолчанию были изменены следующим образом: `shared_buffers = 2048` Мб, `temp_buffers` и `work_mem` равные 16 Мб. На рис. 3.1 и рис. 3.2 приведён мониторинг RabbitMQ с помощью клиента RabbitMQ Management Interface. На рис. 3.1 приведена максимальная зафиксированная нагрузка на очередь. На рис. 3.2 приведена минимальная зафиксированная нагрузка на очередь.



RabbitMQ Management - Chromium										
RabbitMQ Management x +										
localhost:15672/#/queues										
Refreshed 2022-11-25 09:56:55 Refresh every 5 second										
Virtual host										
Cluster rabbit@denis-HP-Pavilion-Gaming-Laptop-15-e										
User guest										
Overview Connections Channels Exchanges Queues Admin										
Queues										
All queues (4)										
Overview										
Name	Type	Features	State	Ready	Unacked	Total	incoming	deliver	get	ack
celery	classic	D	idle	0	0	0	0.00/s	0.00/s	0.00/s	0.00/s
celery@denis-HP-Pavilion-Gaming-Laptop-15-ec1xxx.celery.pidbox	classic	AD TTL Exp	idle	0	0	0				
celeryev.f00faaa-bd8d-4571-bf5a-8c8e167d16b1	classic	AD TTL Exp	running	0	0	0	0.40/s	0.40/s	0.00/s	0.00/s
comments	classic	D	running	134	100	234	36/s	75/s	80/s	80/s

Рисунок 3.1 – Максимальная зафиксированная нагрузка на RabbitMQ

Overview				Messages			Message rates		
Name	Type	Features	State	Ready	Unacked	Total	incoming	deliver / get	ack
celery	classic	D	idle	0	0	0			
celery@denis-HP-Pavilion-Gaming-Laptop-15-ec1xxx.celery.pidbox	classic	AD TTL Exp	idle	0	0	0			
celeryev.83be573f-a6c9-4d07-b3f8-051d5724172f	classic	AD TTL Exp	running	0	0	0	0.40/s	0.40/s	0.00/s
comments	classic	D	running	0	46	46	14/s	14/s	14/s

Рисунок 3.2 – Минимальная зафиксированная нагрузка на RabbitMQ

Уменьшение количества приходящих сообщений (incoming messages) и количества принятых на обработку (ack) связано с тем, что вставка и обработка комментариев происходит следующим образом: сначала вставляется в базу данных комментарий со значением тональности NULL, затем отправляется сообщение с комментарием в очередь задач, после оценки тональности комментария значение комментария с соответствующим идентификатором в базе данных обновляется на вычисленное. Дополнительные проверки показали, что лимитирующим звеном данной системы является процесс вставки комментария в базу данных PostgreSQL: чем больше комментариев в базе данных, тем дольше будет происходить процесс вставки новых комментариев.

## **ЗАКЛЮЧЕНИЕ**

В ходе выполнения курсовой работы была изучена нейронная сеть BERT, были исследованы точность и производительность вычислений нейронной сети BERT при оценке тональности комментариев пользователей. Данными для исследования послужил датасет IMDB, который содержит рецензии к фильмам. Была разработана система обработки комментариев пользователей на английском языке нейронной сетью BERT в условиях, приближённых к условиям реального времени.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Документация Pytorch // URL: <https://pytorch.org/docs/stable/index.html> (дата обращения: 20.11.2022).
2. Датасет IMDB // URL: (дата обращения 20.11.2022).
3. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding // URL: <https://arxiv.org/abs/1810.04805> (дата обращения 20.11.2022).
4. Attention Is All You Need // URL: <https://arxiv.org/abs/1706.03762> (дата обращения 20.11.2022).
5. Официальный репозиторий BERT // URL: <https://github.com/google-research/bert> (дата обращения 20.11.2022).
6. Официальный сайт RabbitMQ // URL: <https://www.rabbitmq.com/> (дата обращения 20.11.2022).
7. Официальный сайт Celery // URL: <https://docs.celeryq.dev/en/stable/> (дата обращения 20.11.2022).
8. Официальный сайт FastAPI // URL: <https://fastapi.tiangolo.com/> (дата обращения 20.11.2022).