МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра математической кибернетики и компьютерных наук

СРАВНИТЕЛЬНЫЙ АНАЛИЗ РЕЗУЛЬТАТОВ ОБУЧЕНИЯ NLP СЕТЕЙ НА РАЗЛИЧНЫХ ТИПАХ ДАТАСЕТОВ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

Студентки 4 курса 451 группы
направления 09.03.04 — Программная инженерия
факультета КНиИТ
Неловко Юлии Алексеевны

Научный руководитель
к. ф.-м. н., доцент _____ А. С. Иванов

Заведующий кафедрой
к. ф.-м. н., доцент _____ С. В. Миронов

Саратов 2023

ВВЕДЕНИЕ

Развитие искусственного интеллекта в настоящее время является одним из перспективных направлений научной и практической деятельности, так как оно позволяет автоматизировать такие сложные процессы, как подготовку отчетности, медицинскую и техническую диагностику, прогнозирование, планирование, управление, мониторинг, обучение, проектирование и многие другие.

Одним из шагов в этом направлении является создание нейронных сетей. Обученная нейронная сеть способна прогнозировать, предсказывать, классифицировать, распознавать и генерировать данные, не хуже, а зачастую даже лучше, чем человек — ведь нейронные сети исключают возможность человеческого фактора.

Целью бакалаврской работы является анализ результатов, полученных с использованием нейронной сети на основе нескольких датасетов. Один из них взят из открытого источника, а второй создан с помощью парсинга ленты новостей с равномерным распределением на рубрики. Проверка реализована на нейронных сетях разной архитектуры. По итогу определен лучший результат.

Были поставлены следующие задачи:

1. поиск готового датасета и составление собственного датасета путем парсинга новостей с Lenta.ru;
2. предобработка датасетов;
3. обучение нейронных сетей с разной архитектурой на выбранных датасетах;
4. создание пользовательского интерфейса для решения поставленной задачи;
5. сравнение качества работы реализованных нейронных сетей с помощью метрики точности (далее — accuracy).

Выпускная квалификационная работа состоит из введения, двух глав, заключения, списка из 20 источников и пяти приложений.

1 Теоретические аспекты нейронных сетей

1.1 Рекуррентные нейронные сети

В этом разделе представлен рассказ о рекуррентных нейронных сетях [1]. Этот вид нейронных сетей часто используют для:

- обработки естественного человеческого языка,
- анализа написанного текста,
- машинного перевода текста,
- генерации текста,
- генерации чисел,
- и др.

Рекуррентные сети подразделяют на 4 типа:

1. «Один к одному». Этот тип рекуррентной нейронной сети применяется если на вход подается единичная информация и на выходе получается единичная информация. Данный вид нейронных сетей часто используют при кодировании и декодировании информации [2].
2. «Один ко многим». Этот тип рекуррентной нейронной сети используют в случае поступления на входе единичной информации, а на выходе — последовательности. В качестве примера можно привести ситуацию, когда на вход поступает одно изображение, а на выходе получают текстовое описание изображенного на картинке.
3. «Многие ко одному». Этот тип рекуррентной нейронной сети применяется при поступлении на вход объемного количества любого вида информации, а на выходе получают единый результат. К примеру, на вход можно подать набор изображений, а на выходе получить информацию, что на всех изображениях находится какой-то конкретный объект.
4. «Многие ко многим». Этот тип рекуррентной нейронной сети используют в случае поступления на входе последовательности информации, а на выходе будет получена измененная последовательность информации. Так, в случае машинного перевода текста на вход поступает текст на одном языке, а на выходе получается текст на другом языке.

1.2 SimpleRNN

SimpleRNN — это реккурентная сеть Элмана. При использовании данного типа рекуррентных нейронных сетей существует проблема, связанная с пере-

полнением памяти, поэтому их используют в задачах, когда нужно не генерировать много значений, а лишь автоматически дополнять какую-то информацию.

В SimpleRNN структура одного из повторяющихся модулей очень проста, например, он может представлять собой один слой с функцией активации гиперболического тангенса. Далее в разделе описана ее структура.

1.3 LSTM

LSTM (long short-term memory, долгая краткосрочная память) является одним из типов нейронной сети. Этот вид нейронных сетей способен к обучению долговременным зависимостям. Но в отличие от первого вида, они потребляют много системных ресурсов. Сети LSTM были представлены Зеппом Хохрайтер и Юргеном Шмидхубером в 1997 году, а позднее усовершенствованы другими исследователями [3]. Эта архитектура была основной из применяемых рекуррентных сетей вплоть до 2017 года. И по сей день она остается весьма востребованной во многих прикладных задачах.

Данный вид рекуррентных сетей был создан для решения проблем долговременной зависимости. Структура LSTM напоминает цепочку, однако в отличии от SimpleRNN модули содержат по четыре слоя, которые взаимодействуют особенным образом.

1.4 GRU

GRU(Gated Recurrent Unit) — это управляемые рекуррентные сети, представленные в 2014-м году для решения часто встречающейся проблемы связанной с исчезанием градиента. Данный вид нейронных сетей представляет упрощенную версию ранее рассмотренного вида сети. По объему памяти ячейки совпадают с LSTM, но обработка информации идет слегка «обрезанным» по функциональности путем. С точки зрения эффективности блоки GRU сравнимы с LSTM во множестве практических задачах, к примеру, их используют для моделирования музыкальных и речевых сигналов, обработки текста и тому подобному. Существует несколько вариантов этого блока, в разделе будет рассмотрена классическая архитектура.

1.5 Класс Sequential

Нейронные сети в основном строят в формате последовательно расположенных слоев.

На самом деле, число скрытых слоев не является ограниченным. Для того чтобы построить подобные модели и применяется класс Sequential. Для работы с данным классом при создании модели пошагово добавляются слои.

Одним из основных слоев класса является слой Dense. Данный слой является полносвязным и отвечает за соединение нейронов между двумя соседними слоями. Он обрабатывает каждый элемент предыдущего слоя, выполняя матричное перемножение этих элементов со своими весами. После этого полученные данные отправляются на следующий слой.

За счет соединения нейронов полносвязного слоя со всеми нейронами предыдущего слоя, каждый из нейронов полносвязного слоя взаимодействует с любым нейроном предыдущего слоя. В этом заключается важность данного слоя для нейронных сетей, используемых для классификации данных. Слой Dense универсален и его можно использовать в любом виде нейронных сетей.

1.6 Оптимизатор Adam

Adam (ADAptive Momentum) — это алгоритм оптимизации, который совмещает принципы инерции MomentumSGD и адаптивного обновления параметров AdaGrad и его модификаций [4].

Для реализации алгоритма используется подход затухающего бегущего среднего для градиентов целевой функции и их квадратов.

1.7 Функции активации

Нейроны представляются в виде функции, называемой функцией активации. Функция активации нужна для определения выходного значения нейрона в зависимости от результата взвешенной суммы входов и порогового значения.

В бакалаврской работе в подразделах рассматриваются используемые в дальнейшем функции активации, а именно — Sigmoid, softmax, tanh, relu.

2 Разработка нейронных сетей

2.1 Сбор и подготовка данных

2.1.1 Редактирование датасета с kaggle

В выбранном датасете присутствуют 800975 строк и 5 столбцов. В разделах topic(далее, рубрики) и tags(далее, метки) есть пропуски. При обнаружении пропуска в столбце рубрик строка датасета при обработке будет удалена, а метки с пропущенными значениями заменены на пустую строку.

На этом этапе было принято решение оставить в датасете рубрики наиболее близкие по размеру, не имеющими общих значений(Россия, Мир) и являющиеся актуальными на данный момент(в отличии от ЧМ-2014). Таким образом осталось 6 рубрик.

2.1.2 Парсинг новостей по категориям с Lenta.ru

Для сбора данных был выбран сайт Lenta.ru. Это обосновано наличием удобного архива с категориями и сопоставимостью с датасетом, найденным на kaggle. Задача парсинга данных была решена с помощью библиотеки BeautifulSoup и lxml, которая является удобной библиотекой для обработки разметки XML и HTML.

Для начала было необходимо определить диапазон дат, за которые будут собраны новости. Так как в результате обработки датасета с kaggle последняя дата в нем — 15 декабря 2018 года, то для избежания пересечения новостей в качестве начала была выбрана дата 1 января 2019 года, а так как сбор проходил в феврале 2023 года, то в качестве даты конца было проставлено 7 февраля. Позже с помощью разработанной функции был получен массив из дат, который с помощью конкатенации строк был превращен в массив ссылок.

Сама функция, которая отвечает за создания массива ссылок была создана с помощью библиотеки datetime. С помощью timedelta был выставлен шаг в один день, с таким шагом перебираются все даты от начала заданного диапазона до его конца, после чего возвращается полученный список дат за этот промежуток времени.

Для того, чтобы обойти возможные проблемы с доступом к сайту, не допуская обвалы, был написан код, с помощью которого происходит пятисекундная пауза, в случае ошибки, связанной с соединением к сайту при get-запросе.

Далее, после получения данных со странички с помощью метода $\mathrm{find\_all}$

были собраны все элементы со значением класса archive-page__item _news, который отвечает за размещение новостей на странице, и собраны ссылки, ведущие на страницы самих новостей.

После чего можно было легко собирать новости, погружая их в датафрейм. Необходимо было собрать 16 тысяч каждого типа, для получения равномерности — теория такова, что в случае если на каждой категории новостей будет достаточное и одинаковое с точки зрения размерности количество, то точность будет выше. Также во избежание блокирования доступа к сайту стоит пятисекундная остановка на каждый день.

2.1.3 Лемматизация датасетов

Текст предварительно необходимо очистить. Для этого весь текст был приведен к нижнему регистру. С помощью регулярного выражения также были убраны пунктуационные символы и заменены ссылки на URL. Числа и цифры будут также заменены на NUM, лишние пробелы — удалены.

Для дальнейшей обработки был создан список для хранения преобразованных данных, загружены стоп-слова для русского языка(союзы, частицы, местоимения и так далее) и инициализирован лемматизатор.

После чего для каждой новости проводится очистка данных с помощью разработанной функциии, проводится токенизация, удаляются стоп-слова и проводится лемматизация с помощью pymorphy2.MorphAnalyzer(), после чего текст собирается в строку с разделителем-пробелом.

2.2 Создание интерфейса

Отрисовка пользовательского интерфейса была реализована с помощью библиотеки PySimpleGUI [5]. С использованием метода theme, была настроена тема типа DarkGreen2. После этого был также настроен заголовок формы, который соответствует классификации новостных статей.

Для удобства пользователей при построении формы помимо обычного набора текста была добавлена возможность выбирать файлы из проводника. Поле с результатом заблокировано от изменения пользователем, добавлены кнопки для получения результата и выхода из приложения.

В результате разработанный пользовательский интерфейс выглядит следующим образом (см. рис. 1):

Рисунок 1 – Внешний вид интерфейса

Далее для получения результатов, для кнопок была прописана их логика поведения. В случае кнопки отмены приложение закрывается, иначе данные подаются на функцию classRes, которая в результате вернет правильную рубрику. В случае, если полученный текст является путем к файлу, то считывается и передается в функцию классификации информация из файла, иначе — сам текст.

Для получения результата, как было отмечено выше, была разработана функция classRes. Сначала была проведена лемматизация полученных в окне данных (аналогичный код приведен в разделе выше), после чего текст был токенизирован в соответствии с уже обученным токенизатором. С помощью нейронной сети был получен массив соответствия каждой из рубрик, после чего была выявлена наибольшая схожесть с одной из рубрик и получен результат в словесной форме.

Для преобразование числового результата, полученного нейронной сетью, используется созданный словарь рубрик и функция для получения названия рубрики по значению-ключу, полученному в результате работы нейронной сети с дальнейшей предобработкой.

2.3 Построение, обучение и тестирование нейронных сетей

В представленных ниже нейронных сетях для борьбы с переобучением использовалась форма регуляризации ранняя остановка(EarlyStopping) [6]. При достижении количества периодов без улучшений равного двойке, обучение будет остановлено, а с помощью параметра restore_best_weights будут восстанавливаться веса модели и эпохи с лучшим значением отслеживаемой величины [7].

2.3.1 Нейронная сеть со слоем SimpleRNN

Для слоя SimpleRNN были проведены эксперименты с лучшей архитектурой и подбором параметра batch_size. В таблице (см. табл. 1) представлены

наиболее точные результаты подбора batch_size для каждой из созданных архитектур:

Таблица 1 – Сравнение результатов обучения нейронной сети со слоем SimpleRNN при разных параметрах

batch size	Мой датасет			kaggle датасет		
	loss	accuracy	остановка	loss	accuracy	остановка
120	0.2283	0.9390	6 эпоха	8.1170e-04	1	5 эпоха
20	0.4055	0.8764	7 эпоха	0.0667	0.9896	7 эпоха

2.3.2 Нейронная сеть со слоем LSTM

Для слоя LSTM были проведены эксперименты с лучшей архитектурой и подбором параметра batch_size. В таблице (см. табл. 2) представлены наиболее точные результаты подбора batch_size для каждой из созданных архитектур:

Таблица 2 – Сравнение результатов обучения нейронной сети со слоем LSTM при разных параметрах

batch size	Мой датасет			kaggle датасет		
	loss	accuracy	остановка	loss	accuracy	остановка
80	0.1495	0.9589	4 эпоха	0.0182	0.9956	3 эпоха
10	0.1518	0.9584	10 эпоха	0.0751	0.9896	10 эпоха

2.3.3 Нейронная сеть со слоем GRU

Для слоя GRU были проведены эксперименты с лучшей архитектурой и подбором параметра batch_size. В таблице (см. табл. 3) представлены наиболее точные результаты подбора batch_size для каждой из созданных архитектур:

Таблица 3 – Сравнение результатов обучения нейронной сети со слоем GRU при разных параметрах

batch size	Мой датасет			kaggle датасет		
	loss	accuracy	остановка	loss	accuracy	остановка
20	0.0941	0.9711	4 эпоха	0.0210	0.9949	5 эпоха
20	0.1699	0.9515	5 эпоха	0.0226	0.9948	5 эпоха
5	0.0934	0.9710	4 эпоха	6.2954e-04	1	4 эпоха

2.3.4 Сравнение результатов

В результате проведенных испытаний лучше всего показала себя нейронная сеть со слоем GRU. Показатели точности у датасета, взятого с kaggle лучше, несмотря на выдвинутую ранее теорию, что равномерное количество новостей в каждом из разделов будет выдавать более корректный результат. Ниже представлены модели с наилучшими показателями точности и функции потерь после проведения испытаний (см. табл 4):

Таблица 4 – Сравнение лучших результатов обучения нейронной сети с разными слоями

Слой	batch size	Мой датасет			kaggle датасет		
		loss	accuracy	остановка	loss	accuracy	остановка
LSTM	5	0.1518	0.9584	10 эпоха	0.0751	0.9896	10 эпоха
LSTM	80	0.1495	0.9589	4 эпоха	0.0182	0.9956	3 эпоха
GRU	5	0.0934	0.9710	4 эпоха	6.2954e-04	1	4 эпоха
GRU	20	0.0941	0.9711	4 эпоха	0.0210	0.9949	5 эпоха
SimpleRNN	120	0.2283	0.9390	6 эпоха	8.1170e-04	1	5 эпоха

Для сравнения лучших результатов была использована Метрика ROC-AUC, которая принимает значения от 0 до 1, где 0.5 соответствует случайному распределению классов, а 1 — идеальному разделению классов. Результаты представлены в таблице 5.

Таблица 5 – Сравнение лучших результатов обучения нейронной сети с разными слоями по метрике ROC-AUC

Слой	batch size	Мой датасет	kaggle датасет
GRU	20	0.9987536709165021	0.9999495812754516
GRU	5	0.9962962409654758	0.9993094326720087
LSTM	80	0.996194585469773	0.9998667422747061

ЗАКЛЮЧЕНИЕ

В рамках бакалаврской работы был с помощью библиотеки BeautifulSoup собран датасет русских новостей с сайта Lenta.ru на 96 тысяч строк, по 16 тысяч на каждый вид рубрики новостей. Теория о повышении точности результатов при равномерном распределении рубрик была опровергнута на основании проведенных экспериментов. Оказалось, что датасет такой же мощности, взятый из свободного доступа, дает лучшие результаты.

Было проведено 23 эксперимента над тремя видами рекуррентных нейронных сетей. Для каждой из них было подобрано по 2-3 варианта архитектуры, в рамках которых также проведено по 3-4 эксперимента с подбором гиперпараметров. Наивысшие показатели точности были у нейронной сети с управляемым рекуррентным нейроном(типа GRU). Данная нейронная сеть на собранных данных выдает результат примерно равный 0,97, на датасете, взятом с kaggle — 0.99.

Также был создан пользовательский интерфейс с помощью библиотеки PySimpleGUI. Данная библиотека удобнее для работы по сравнению с библиотекой Tkinter, с помощью нее можно задать цветовую гамму у окна за счет встроенных вариантов. Интерфейс позволяет как вводить текст вручную, так и воспользоваться проводником, что позволяет ускорить работу пользователя. В результате работы алгоритма нейронной сети в заблокированном для ввода пользователем поле выдается название рубрики новости.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 *Ростовцев, В. С.* Искусственные нейронные сети / В. С. Ростовцев. — Санкт-Петербург: Лань, 2021.

2 Neurohive [Электронный ресурс]. — URL: https://neurohive.io/ru/osnovy-data-science/rekurrentnye-nejronnye-seti/ (Дата обращения 03.05.2023). Загл. с экр. Яз. рус.

3 *Антонова, В. А.* Введение в анализ больших информационных массивов / В. А. Антонова, В. М. Антонова. — Москва: Издательство МГТУ им. Н.Э. Баумана (Москва), 2021.

4 Документация PuzzleLib [Электронный ресурс]. — URL: https://puzzlelib.org/ru/documentation/base/optimizers/Adam/ (Дата обращения 03.05.2023). Загл. с экр. Яз. рус.

5 *Векслер, В. А.* Использование графического интерфейса pysimplegui при решении учебных практикумов на python / В. А. Векслер // *Информационные технологии в образовании*. — 2022. — № 5. — С. 57–62.

6 *Мещерина, Е. В.* Системы искусственного интеллекта : учебно-методическое пособие / Е. В. Мещерина. — Оренбург: ОГУ, 2019.

7 *Вакуленко, С. А.* Нейронные сети : учебное пособие / С. А. Вакуленко. — Санкт-Петербург: Санкт-Петербургский государственный университет промышленных технологий и дизайна, 2019.