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

РАЗРАБОТКА И ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ВОПРОСНО-ОТВЕТНОЙ СИСТЕМЫ ДЛЯ ОБУЧЕНИЯ PYTHON

АВТОРЕФЕРАТ МАГИСТЕРСКОЙ РАБОТЫ

Студентки 2 курса 273 группы
направления 02.04.03 — Математическое обеспечение и администрирование
информационных систем
факультета КНиИТ
Огневой Татьяны Алексеевны

Научный руководитель
ст. преп. _____ Е. Е. Лапшева

Заведующий кафедрой
к. ф.-м. н., доцент _____ М. В. Огнева

Саратов 2025

ВВЕДЕНИЕ

Актуальность темы. Большие языковые модели (Large Language Models, LLM), имитирующие человеческое общение, в последнее время получили широкую популярность благодаря тому, что их можно использовать при решении различных задач. LLM, такие как ChatGPT от OpenAI, показали хорошие результаты на академических и профессиональных экзаменах по различным дисциплинам, что позволяет предположить, что они могут помочь людям в профессиональных, промышленных и академических областях.

Однако универсальные модели из-за широкого охвата областей могут показывать плохие результаты при углублении в какие-то из этих областей. Таким образом, возникает потребность в создании моделей, более специфичных для конкретной области. В частности, исследования показывают, что наиболее популярные крупные языковые модели плохо справляются с различными манипуляциями с программным кодом.

Цель магистерской работы — разработка и программная реализация вопросно-ответной системы для обучения Python.

Поставленная цель определила **следующие задачи:**

1. Провести обзор существующих языковых моделей.
2. Рассмотреть основные подходы к обучению языковых моделей.
3. Рассмотреть основные архитектуры больших языковых моделей.
4. Рассмотреть RAG модели.
5. Провести исследовательский анализ и предобработку готового датасета.
6. Применить методы кластеризации и сравнения схожести строк для формирования ответов на вопросы.
7. Провести обзор инструментов для сбора данных.
8. Отладить процесс сбора данных с сайта StackOverflow.
9. Собрать датасет с сайта StackOverflow.
10. Провести исследовательский анализ и предобработку датасета.
11. Дообучить модель Gemma с использованием собранного датасета.
12. Рассмотреть архитектуру RAG моделей.
13. Реализовать различные RAG модели.
14. Провести сравнительный анализ реализованных RAG моделей.
15. Провести подбор гиперпараметров.
16. Создать чат-бот на основе лучшей модели.

Практическая значимость магистерской работы заключается в том, что разработанная система может использоваться для обучения языку Python школьников средней и старшей школы, студентов технических направлений, а также студентов дополнительных курсов и программ переподготовки и повышения квалификации. Она создает ответы на вопросы, основываясь на актуальной информации. Время создания ответов не превышает 10 секунд.

Структура и объём работы. Магистерская работа состоит из введения, 7 разделов, заключения, списка использованных источников и 12 приложений. Общий объем работы — 92 страницы, из них 60 страниц основное содержание, включая 19 рисунков и 11 таблиц, список использованных источников информации — 34 наименований.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Первый раздел «Обзор существующих языковых моделей» посвящен обзору наиболее популярных существующих языковых моделей, исследованию их ответов на различные вопросы.

В одном из исследований оценивались 15 вопросов из различных дисциплин, включая социологию, бизнес, математику, историю, астрономию и историю искусств, имеющих отношение к высшему образованию. В результате оценки этих вопросов выяснилось, что лучше всего справился с задачей GPT-4, за ним следует GPT-3.5, Bing Chat и Bard. Другое сравнительное исследование, в котором также участвовал Claude, оценивало LLM по 1002 вопросам, охватывающим 27 категорий. Это исследование продемонстрировало коэффициент успешности 84,1% для GPT-4, 78,3% для GPT-3.5, 64,5% для Claude и 62,4% для Bard, где Claude оказался лучшим в категории «факты». В целом чат-боты продемонстрировали мастерство в понимании языка (орфография, грамматика, перевод, словарный запас и т. д.) и фактов, но столкнулись с трудностями в математике, кодировании, тестах IQ (стандартизированный показатель человеческого интеллекта) и рассуждениях.

Второй раздел «Обзор существующих подходов к обучению языковых моделей» посвящен обзору существующих подходов к обучению и оптимизаторов моделей.

Подходы к обучению языковых моделей:

1. Fine-Tuning (FT) — это подход, который заключается в обновлении весов предварительно обученной модели путем обучения на контролируемом наборе данных, специфичном для конкретной задачи.
2. Few-Shot (FS) — это подход, который заключается в том, что модели дается несколько демонстраций задачи во время вывода, но обновления весов не допускаются.
3. One-Shot (1S) — это то же самое, что и Few-Shot, за исключением того, что допускается только одна демонстрация в дополнение к описанию задания на естественном языке.
4. Zero-Shot (0S) — это то же самое, что и One-Shot, за исключением того, что демонстрации не допускаются, а модели дается только инструкция на естественном языке, описывающая задание.

Оптимизаторы, используемые в языковых моделях:

— Adam — метод эффективной стохастической оптимизации, требующий только градиентов первого порядка и не требующий большого объема памяти.
— Lion — EvoLved Sign Momentum, отслеживает только импульс и использует операцию знака для вычисления обновлений, что приводит к меньшим затратам памяти и равномерной величине обновлений по всем измерениям.
— Adan — Adaptive Nesterov Momentum Algorithm, разрабатывает метод оценки Nesterov momentum для стабильной и точной оценки моментов градиента первого и второго порядка в адаптивных градиентных алгоритмах для ускорения сходимости.

В третьем разделе «Архитектура больших языковых моделей» рассмотрены основные архитектуры больших языковых моделей.

В декабре 2017 года была опубликована основополагающая статья, написанная сотрудниками Google Brain и Google Research. В этой статье первый раз описали модель трансформер. Трансформер превзошел все существовавшие модели для решения задач обработки естественного языка. Трансформер обучался быстрее, чем предшествовавшие архитектуры, и получал более высокие результаты оценки. Трансформеры стали ключевым компонентом сферы обработки естественного языка.

Разрозненные экспертные модели обычно заменяют слой нейронной сети набором экспертов. В обработке естественного языка используется MoE слой, который принимает на вход представление лексем x и направляет его к наиболее подходящим экспертам top k, выбранным из набора $E_i(x)_{i=1}^{N}$ из N экспертов.

В четвертом разделе «RAG модели» рассматривается технология RAG улучшения LLM моделей. Большие языковые модели сталкиваются со значительными ограничениями в задачах, связанных с конкретным доменом или требующих больших знаний , в частности, они создают «галлюцинации» при обработке запросов выходящих за рамки обучающих данных или требующих текущей информации. Чтобы преодолеть трудности, технология Retrieval-Augmented Generation (RAG) улучшает LLM, извлекая соответствующие фрагменты документов из внешней базы знаний путем вычисления семантического сходства. Обращаясь к внешним знаниям, RAG эффективно уменьшает проблему генерации фактологически неверного контента.

RAG объединяет сильные стороны генеративного ИИ и поисковых технологий для повышения качества и релевантности генерируемого текста. В отличие от традиционных генеративных моделей, которые полагаются исключительно на свои внутренние знания, RAG включает дополнительный этап, на котором он извлекает информацию из внешних источников, таких как базы данных, документы или Интернет, прежде чем генерировать ответ. Такая интеграция механизмов поиска позволяет RAG получать доступ к актуальной информации и контексту, что делает его особенно ценным для приложений, где важна точная и актуальная информация.

Пятый раздел «Сбор данных» посвящен видам представления данных и веб-скрейпингу.

Существуют различные формы представления данных: табличные данные или данные в виде электронных таблиц; многомерные массивы; несколько таблиц данных, связанных между собой ключевыми столбцами; равномерно или неравномерно распределенные временные ряды; графы.

Доступ к данным — необходимый первый шаг для использования большинства инструментов, описанных в этой книге. Ввод и вывод данных обычно делятся на несколько основных категорий: чтение текстовых файлов и других более эффективных форматов на диске, загрузка данных из баз данных и взаимодействие с сетевыми с источниками данных, такими как веб-интерфейсы.

Веб-скрейпинг - это практика сбора данных любыми средствами, кроме программы, взаимодействующей с API (или, очевидно, через человека, использующего веб-браузер).

Чаще всего это достигается путем написания автоматизированной программы, которая обрвщается к серверу, запрашивает данные (обычно в виде HTML и других файлов, из которых состоят веб-страницы), а затем анализирует эти данные для извлечения необходимой информации. На практике вебскреппинг включает в себя широкий спектр методов программирования и технологий, таких как анализ данных и информационная безопасность.

Шестой раздел «Инструменты для сбора и обработки датасета» посвящен библиотекам для парсинга и анализа датасета. Библиотека requests — одна из самых распространенных библиотек в Python для выполнения HTTP-запросов. Она позволяет легко загружать файлы из Сети, не беспокоясь о таких сложных проблемах, как сетевые ошибки, проблемы с подключением и сжатием

данных.

Наиболее распространенными типами форматирования ответов являются eXtensible Markup Language (XML) и JavaScript Object Notation (JSON).

Библиотека разбора JSON является частью Python Core. В отличие от многих языков, которые могут разбирать JSON в специальный JSON-объект или JSON-узел, Python использует более гибкий подход и превращает JSON-объекты в словари, массивы JSON в списки, строки JSON в строки и так далее. Таким образом, упрощается доступ к значениям, хранящимся в JSON, и манипулирование ими.

Библиотека pandas предоставляет высокоуровневые структуры данных и функции, разработанные для того, чтобы упростить работу со структурированными или табличными данными. Основные объекты pandas — это DataFrame, табличная, ориентированная на столбцы структура данных с метками строк и столбцов, и Series, одномерный объект массива с метками.

Polars - это библиотека с открытым исходным кодом для работы с данными, известная как одно из самых быстрых решений для обработки данных на одной машине. Она имеет хорошо структурированный, типизированный API, который прост в использовании. Polars может выполнять обычные операции примерно в 5-10 раз быстрее, чем pandas. Кроме того, для выполнения операций Polars требуется значительно меньше памяти, чем pandas: для выполнения операций pandas требуется в 5-10 раз больше оперативной памяти, чем размер набора данных, по сравнению с 2-4 разами, необходимыми для Polars.

В седьмом разделе «Реализация вопросно-ответной системы» рассматривается процесс сбора, анализа и предобработки данных, а также создание и сравнительный анализ различных вопросно-ответных систем.

Для вопросно-ответной системы для обучения языку Python использовались данные с сайта Stack Overflow.

На платформе kaggle представлен датасет ”Python Questions from Stack Overflow”. ”Python Questions from Stack Overflow” — это набор данных, который содержит вопросы и ответы на них по языку программирования Python (помеченных тегом ”Python”), заданные в период со 2 августа 2008 года по 19 октября 2016 года.

Для эффективной работы с данными нужно понимать их устройство и адаптировать под конкретную задачу. Для этого были проведены исследова-

тельский анализ данных и предобработка.

Для вопросно-ответной системы исследовалась кластеризация.

Подходы:

— Кластеризация с помощью методов подсчета схожести строк. Обработанный датасет кластеризовался с помощью метода DBSCAN с использованием методов подсчета схожести строк: расстояние Левенштейна и метод выравниваний.
— Представление записей обработанного датасета в виде вектора и кластеризация с помощью метода DBSCAN с использованием косинусной меры близости. Векторизация проводилась двумя способами: по методу подсчета количества вхождений и по методу TF-IDF.

Первый подход оказался не очень эффективным. В самом деле, методы подсчета схожести строк работают, не учитывая смысла предложений. Второй подход показал лучшие результаты. По результатам исследования для вопросно-ответной системы выбран способ векторизации TF-IDF. Он помогает наиболее точно определить схожие по смыслу вопросы.

В наборе данных на Kaggle содержатся все вопросы, заданные в период со 2 августа 2008 года по 19 октября 2016 года. Программирование постоянно развивается, поэтому данные, собранные 8 и более лет назад, утрачивают актуальность. Были собраны более "свежие".

Для таблиц проводилось исследование данных: исследование информации о датасете в целом, выявление пропусков и дубликатов, исследование отдельно взятых столбцов и взаимосвязей между ними.

Для дообучения использовалась модель Gemma 2B, язык английский. Оптимизатор AdamW, loss-функция SparseCategoricalCrossentropy.

Дообучение модели даже на небольшом датасете требует больших затрат времени и ресурсов. В условиях быстро меняющихся данных будет сложно поддерживать их актуальность.

Были созданы 9 RAG моделей и проведен их сравнительный анализ.

Для валидации данных перед обучением моделей была проведена сортировка по рейтингу ответов и фильтрация по столбцам и по строкам.

Для обработки текстов использовалась библиотека polars и ноутбук с процессором AMD Ryzen 5 3500U with Radeon Vega Mobile Gfx, 4 ядра процессора и 16 Гб оперативной памяти.

Для создания эмбеддингов использовались модели ”all-MiniLM-L6-v2”, ”paraphrase-TinyBERT-L6-v2” и ”multi-qa-MiniLM-L6-cos-v1”.

Для создания faiss-индексов использовался индекс IndexFlatL2 из библиотеки faiss.

Исследование RAG моделей проводилось на виртуальной машине Yandex Cloud c1.8, 8 ядер процессора и 64 Гб оперативной памяти. Были исследованы модели: ”bart-large”, ”bart-large-cnn” и ”flan-t5-base”.

Для исследования был подобран список из 5 вопросов, проведены замеры времени создания ответов.

Для сравнительного анализа использовались метрики Recall@k, MRR, BERTScore F1 и ROUGE метрики.

В соответствии со значениями метрик самая лучшая модель — комбинация токенизатора ”multi-qa-MiniLM-L6-cos-v1” и генератора ”bart-large-cnn”.

При аналитическом сравнении ответов на вопросы в качестве лучшей модели также была выбрана комбинация токенизатора ”multi-qa-MiniLM-L6-cos-v1” и генератора ”bart-large-cnn”.

С учетом времени, затраченного моделью на ответы на вопросы, модель признана лучшей из исследуемых.

Для улучшения качества ответов и времени работы модели были исследованы комбинации различных значений гиперпараметров top_k и do_sample. Сильных преимуществ различные комбинации гиперпараметров не дали, хотя можно заметить, что результаты улучшались с увеличением количества документов. Принято решение оставить значения по умолчанию.

Для комфортного взаимодействия пользователя с системой был создан чат-бот в Telegram.

ЗАКЛЮЧЕНИЕ

В данной работе был проведен обзор существующих языковых моделей (ChatGPT-3.5, ChatGPT-4, Google Bard, Copilot, Claude-2), обоснована актуальность создания специфичных моделей для работы с кодом.

Рассмотрены основные подходы к обучению языковых моделей, такие как Fine-Tuning, Few-Shot, One-Shot, Zero-Shot; разобраны основные алгоритмы оптимизаторов, применяемые в языковых моделях (Adam, AdamW, Lion, Adan), их преимущества и недостатки; рассмотрены основные архитектуры (Transformer и MoE).

Рассмотрены создание, обучение и преимущества RAG моделей.

В данной работе был проведен обзор инструментов для сбора и анализа данных (форматы представления данных, веб-скрейпинг, библиотеки requests, json, pandas и polars).

Реализована кластеризация методом DBSCAN с помощью методов подсчета схожести строк (расстояние Левенштейна и метод выравниваний) и с помощью предварительной векторизации (мешок слов и TF-IDF) и использованием косинусной меры близости.

Отлажен процесс сбора данных с сайта StackOverflow. Были устранены недостатки, выявленные при тестировании.

Был собран датасет с сайта StackOverflow. Проведен исследовательский анализ и предобработка датасета. Проведено дообучение модели Gemma 2B.

Были исследованы 9 RAG моделей (попарные комбинации токенизаторов ”all-MiniLM-L6-v2”, ”paraphrase-TinyBERT-L6-v2”, ”multi-qa-MiniLM-L6-cos-v1” и генераторов ”bart-large”, ”bart-large-cnn”, ”flan-t5-base”).

Проведен сравнительный анализ RAG моделей математическим и аналитическим способами. Выбрана лучшая модель: сочетание токенизатора ”multi-qa-MiniLM-L6-cos-v1” и генератора ”bart-large-cnn”.

Проведен подбор и сравнительный анализ комбинаций гиперпараметров.

Создан чат-бот с использованием лучшей модели для комфортного взаимодействия с пользователем.

Отдельные части магистерской работы были опубликованы / представлены на конференциях:

1. Студенческая научная конференция факультета КНиИТ (22.04.2022, фКНиИТ, СГУ; доклад на тему «Анализ и предобработка данных для вопросно-

ответной системы»).

2. Международная конференция ”Recent Scientific Investigation” (02.06.2025, по итогу участия статья ”Creating a question answering system for teaching Python” была опубликована в сборнике трудов конференции).

Основные источники информации:

1. Ray Partha Pratim. ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope // Internet of Things and Cyber-Physical Systems. — 2023. — Vol. 3. — P. 121–154.
2. Rudolph Jurgen, Tan Shannon, Tan Samson. War of the chatbots: Bard, Bing Chat, ChatGPT, Ernie and beyond. The new AI gold rush and its impact on higher education // Journal of Applied Learning & Teaching. — 2023. — Vol. 6, no. 1. — P. 364–389.
3. Hochmair Hartwig H., Juhasz Levente, Kemp Takoda. Correctness Comparison of ChatGPT-4, Bard, Claude-2, and Copilot for Spatial Tasks. — 2024. — 2401.02404.
4. Будума Нихиль, Локашо Николас. Основы глубокого обучения. Создание алгоритмов для искусственного интеллекта следующего поколения. — Москва : Манн, Иванов и Фербер, 2020.
5. McKinney, Wes. Python for Data Analysis. Data Wrangling with Pandas, NumPy, and IPython / Wes McKinney. — Springfield : O’REILLY, 2017.
6. Mitchell, Ryan. Web Scraping with Python / Ryan Mitchell. — Springfield : O’REILLY, 2015.
7. Polars-DataFrames for the new era [Электронный ресурс] https://pola.rs/ (дата обращения: 20.03.2025).
8. Stack Exchange API v2.3 [Электронный ресурс] https://api.stackexchange.com/docs (дата обращения: 25.12.2024)