E

УДК 004.912 **DOI** 10.24412/2413-7383-2025-4-39-36-48

В. О. Елисеев[1], В. И. Бондаренко[2], А. Ю. Максимова[1]
[1]Федеральное государственное бюджетное научное учреждение
«Институт прикладной механики и математики»
283048, Донецкая Народная Республика, г. Донецк, ул. Розы Люксембург, 74
[2]Федеральное государственное бюджетное образовательное учреждение
высшего образования «Донецкий государственный университет»
283001, Донецкая Народная Республика, г. Донецк, ул. Университетская, 24

РАЗРАБОТКА ИНТЕЛЛЕКТУАЛЬНОГО АССИСТЕНТА ДЛЯ ОБРАЗОВАТЕЛЬНОЙ ОРГАНИЗАЦИИ*

V. O. Eliseev[1], V. I. Bondarenko[2], A. Y. Maksimova[1]
[1]Federal State Budgetary Scientific Institution "Institute of Applied Mathematics and Mechanics"
283048, Donetsk People's Republic, Donetsk, Rosa Luxemburg st, 74
[2]Federal State Budgetary Educational Institution of Higher Education "Donetsk State University"
283001, Donetsk People's Republic, Donetsk, University st, 24

DEVELOPMENT OF AN INTELLIGENT ASSISTANT FOR AN EDUCATIONAL INSTITUTION

В статье предложен метод построения интеллектуальных ассистентов для образовательных организаций на основе больших языковых моделей (LLM). Это позволит пользователям получать справочные ответы об организации в течение минуты вместо часов/суток. Метод был реализован для построения ассистента Донецкого Государственного университета (ДонГУ). Выполнен сбор и предобработка корпоративных данных, обоснован выбор способа их интеграции в LLM. Итоговая система представляет собой классический Retrieval Augmented Generation (RAG) с дообученной LLaMA-3.1-8b (генератор) и Giga-Embeddings-instruct (ретривер). Для оценки качества работы модели-генератора использовались метрики совпадения на уровне токенов и оценка семантической близости ответов ассистента к целевым ответам для ряда заготовленных сценариев. Исходный код разработанных скриптов и финальная версия ассистента для ДонГУ находятся в открытом доступе, предложенный подход применим к адаптации для других образовательных организаций.

Ключевые слова: большая языковая модель, обработка естественного языка, вопросно-ответный поиск, генерация, дополненная поиском, LLAMA, тонкая настройка.

The article proposes the method for development intelligent assistants for educational institutions based on Large Language Models (LLMs). This enables users to receive informational responses about an institution within a minute, as opposed to the hours or days typically required. The method was implemented to create an assistant for Donetsk State University (DonSU). The study involved selecting a method for integrating corporate data into an LLM, followed by the direct collection and preprocessing of this data. The final system is a standard Retrieval-Augmented Generation (RAG) pipeline, which utilizes a fine-tuned LLaMA-3.1-8b model as the generator and Giga-Embeddings-instruct as the retriever. To evaluate the performance of the generator model, we used token-level similarity metrics and assessed the semantic similarity between the assistant's responses and the target answers for a set of predefined scenarios. The source code for all implemented steps and the final version of the DonSU assistant are open-source what makes the proposed approach applicable for other educational institutions.

Key words: large language model, natural language processing, question answering, Retrieval-Augmented Generation, LLAMA, fine-tuning.

* Работа выполнена при финансовой поддержке Минобрнауки России в рамках научной темы "Разработка и совершенствование интеллектуальных методов классификации и прогнозирования для задач распознавания образов и моделирования информационных процессов" FREM-2024-0001 (Регистрационный номер 1023111000141-9-1.2.1)

Введение

Развитие современных методов NLP и появление больших языковых моделей (LLM) предоставляет возможность автоматизации работы агентов поддержки путем построения интеллектуальных ассистентов, способных отвечать на вопросы пользователей в режиме реального времени. Внедрение ассистентов в вузы сократит время ожидания ответа для заинтересованных пользователей и освободит время сотрудников. Существует явная необходимость в разработке методологии и прикладных инструментов для создания подобных цифровых помощников.

Существующие LLM не могут быть использованы напрямую в образовательных организациях, таких как Донецкий Государственный университет (ДонГУ), поскольку данные большинства организаций не были использованы для их обучения и по умолчанию модели не способны давать специфические ответы на вопросы, касающиеся их деятельности. Для решения данной проблемы необходимо определить источники корпоративных данных и методы их интеграции с LLM. Также нужно разработать механизм актуализации информации. Настройка такого взаимодействия позволит создать прикладной сервис в виде чат-бота, где можно задавать вопросы и получать ответы, основанные на специфике вуза.

В данной статье предложена методика создания интеллектуальных ассистентов для образовательных организаций и описаны проделанные шаги – от выбора LLM и способа интеграции корпоративных данных до реализации прикладного сервиса.

Цель исследования – разработать методику построения ассистентов для учебных заведений и реализовать ассистента для ДонГУ. Методика должна содержать универсальные шаги для вузов РФ и учитывать ограниченность ресурсов (в ДонГУ использовали 2 ПК с NVIDIA RTX 4080 SUPER и RTX A4000).

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

1) выбрать LLM для генерации ответов;
2) разработать интеграцию корпоративных данных;
3) выполнить предобработку и фильтрацию данных;
4) реализовать сервис для взаимодействия с ассистентом

Выбор LLM и механизма интеграции данных

Большинство общедоступных LLM не обладает знаниями о внутренних процессах конкретных организаций (исключением могут быть общеизвестные корпорации, однако и в этом случае знания будут носить лишь общий характер). В связи с этим, для построения интеллектуального ассистента, работающего на основе LLM, необходимо найти способ внедрить корпоративную информацию в модель.

Существует 3 способа интеграции корпоративных данных в LLM: обучить собственную модель с нуля, как это было в случае BloombergGPT [1]; провести тонкую настройку уже предобученной LLM [2], либо же использовать механизм генерации, дополненной поиском (Retrieval Augmented Generation, RAG) [3].

Обучение собственной модели с нуля не подходит для решаемой задачи, поскольку данный процесс является крайне ресурсоемким (необходимо иметь в распоряжении кластеры GPU) и требует наличия текстовых корпусов большого объема (сотни миллиардов токенов) и высокого качества.

Тонкая настройка предобученной LLM снижает требования к вычислительным ресурсам и размеру и качеству текстового корпуса. Поскольку модель не учится моделировать язык с нуля, ей достаточно «показать» текстовый корпус с корпоративными

данными конкретной организации, и она будет способна отвечать на связанные с ней вопросы. В данном случае достаточно иметь корпус объемом от сотен тысяч до миллионов токенов, а процесс осуществим при ограниченных ресурсах, при использовании метода адаптации матрицами низкого ранга (LoRA) [4]. Так как процессы в образовательных организациях подвержены частым изменениям, важно поддерживать знания модели в актуальном состоянии. Для их актуализации необходимо часто переобучать модель, что ресурсозатратно, поэтому дообучение в чистом виде не подходит для решаемой задачи.

Наименее ресурсоемким способом внедрить корпоративные знания в LLM является построение Retrieval Augmented Generation (RAG) систем. Данный подход не требует дообучения LLM и позволяет использовать как локальные модели из открытого доступа, так и коммерческие по API. Корпоративные данные сначала векторизуются с помощью модели-ретривера – предобученной encoder-only модели, после чего помещаются в векторную базу. Во время работы системы ретривер получает пользовательский запрос, векторизует его, после чего ищет в векторной базе N наиболее семантически близких документов, в которых потенциально может находиться ответ. В качестве метрики близости [5] обычно используется косинусное сходство или L2-расстояние. Найденные документы передаются вместе с запросом пользователя модели-генератору, которая должна сформировать ответ на запрос на основе собственных данных и найденной ретривером информации. Недостатком такой системы является зависимость от качества модели-ретривера [6], так как в случае ошибок в его работе модель-генератор не получит необходимой корпоративной информации и не сможет сгенерировать качественный ответ.

В данной работе мы объединили дообучение LLM и RAG, использовав дообученную специальным образом LLM в качестве генератора. Однако мы не просто дообучили модель отвечать на вопросы, связанные с деятельностью ДонГУ, передав в нее соответствующие корпоративные данные в виде инструкционного датасета [7], а натренировали ее извлекать релевантные ответы из предложенных ретривером документов в соответствии с пользовательским запросом, как того требует RAG. Для этого был разработан контекстно-инструкционный набор данных [8], состоящий из потенциального пользовательского запроса, трех наиболее релевантных относительно запроса фрагментов документов из векторной базы и целевого ответа модели. Процедура сбора данных для данной работы будет описана в следующем разделе. Подобный подход был успешно применен при дообучении моделей-генераторов для систем RAG, решающих общие задачи используя мультидоменные знания [9], [10]. В нашем случае набор данных был собран под задачу вопросно-ответного поиска по внутренним процессам ДонГУ.

В качестве дообучаемой модели была выбрана квантизованная LLaMA-3.1-8b, что призвано провести процесс дообучения с помощью LoRA на ограниченных ресурсах и сохранить приемлемый уровень качества ответов.

Сбор и подготовка контекстно-инструкционного набора данных

Для дообучения генератора собирается обучающий корпус и данные для заполнения базы знаний. Источником выбран официальный сайт ДонГУ, так как его содержимое структурированно, машиночитаемо и обновляемо, что регламентировано нормативами [11]. Это делает предложенный способ применимым для других вузов.

Сбор и подготовка данных включают:

1) выгрузку текстов с сайта,
2) фильтрацию нерелевантных материалов,
3) построение датасета и векторной базы,
4) добавление релевантных фрагментов как context.

Данные собираются скриптами из открытого доступа [12]. Текстовое содержимое страниц доступно через уникальный в рамках сайта CSS-селектор, что позволяет использовать скрипты для других вузов. В результате обхода получено 2912 файлов для обучения модели и наполнения базы, и 3546 документов для инференса.

Для фильтрации нерелевантного содержимого использовались три LLM (Gemma2-9b-it [13], LLaMA-3.1-70B-Instruct, LlaMA-3.3-70B-Instruct), которые оценивали файлы по пятибалльной шкале. Распределение оценок показано на рисунке 1.

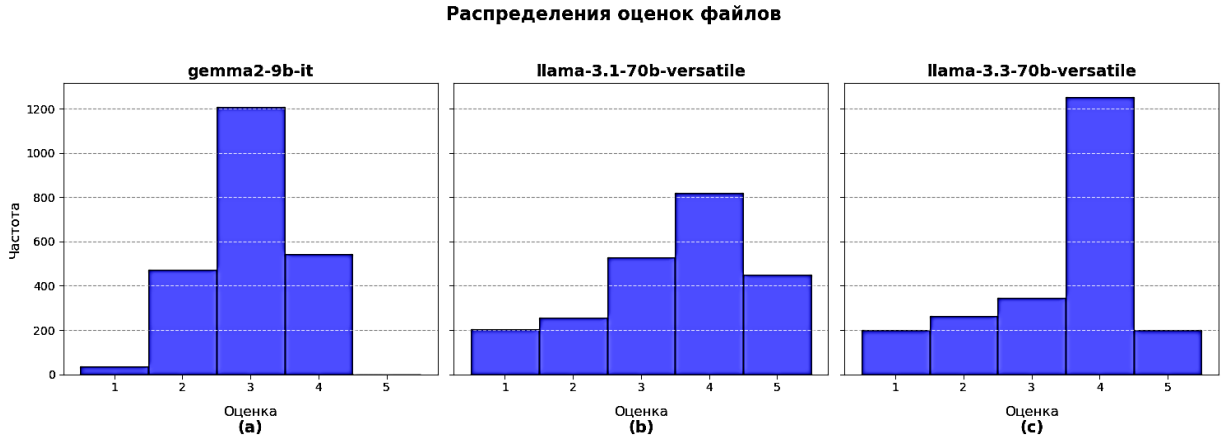

Рисунок 1 – Распределения оценок файлов, полученные от выбранных LLM

Полученные распределения оценок смещены вправо, а мода совпадает с предыдущим максимальным значением. Анализ распределений показал, что модели LLaMA более эффективно анализируют текстовые данные и лучше подходят для синтетической генерации инструкционных пар.

Файлы со средней оценкой 2 балла и ниже были исключены, что свидетельствует о нерелевантности данных текстов. После фильтрации из 2912 файлов осталось 2337, этого объема достаточно для формирования качественного инструкционного датасета. Исходный код модуля фильтрации и использованный промпт доступны в открытом доступе [14].

Для построения классического инструкционного датасета на основе отобранных текстовых файлов было решено также использовать модель LlaMA-3.3-70B-Instruct как лучшую в процессе исключения нерелевантных текстов. Был создан промпт, включающий в себя название файла, его содержимое и метку последнего обновления. и запрос модели сгенерировать максимально возможное число инструкций по заданной информации о странице, либо, если поданный текст не является релевантным – отказаться от генерации, выдав сообщение об ошибке.

После первичного запуска процесса генерации инструкций и анализа распределения количеств инструкций по файлам было выявлено, что для ряда информативных файлов сгенерировано малое количество инструкций. Повторный промптинг моделей для файлов, получивших 3 и менее инструкции, с пониженными значениями температуры (с 1.0 до 0.7) и top-p (с 1.0 до 0.8) позволил получить более детерминированные ответы модели [15]. В результате получено 806 новых инструкций. Итого

датасет включил в себя 25633 инструкции, что достаточно для дообучения модели, работающей с единственным доменом. Исходный код модуля генерации инструкционного датасета также находится в открытом доступе [16]. Для расширения построенного инструкционного датасета полем с контекстом была построена векторная база на основе релевантных текстовых файлов. В качестве ретривера выбрана модель Giga-Embeddings-instruct [17], разработанная командой Сбера. Несмотря на относительно небольшой размер (2.5 миллиарда параметров), контекст модели 4096 токенов, а размерность получаемых эмбеддингов 2048, что делает их более репрезентативными и позволяет лучше описать семантику входной последовательности [18]. Выбранная модель была обучена на русском языке, что позволяет описать больший объем текста меньшим числом токенов.

Документы были фрагментированы по 500 токенов с перекрытием 50 по абзацам, что позволило уместить их в контекстное окно модели LLaMA-3.1-8b и избежать разрывов контекста внутри предложений. После удаления дубликатов и фрагментации сформировано 9935 уникальных фрагментов для векторной базы.

Контекстно-инструкционный набор данных для дообучения LLM в системе RAG был построен следующим образом: для каждого input инструкционного датасета выбраны три наиболее релевантных фрагмента из векторной базы на основе косинусного сходства, их содержимое объединено в поле context результирующего датасета. Исходный код модулей проекта доступен в открытом доступе [16].

Дообучение и тестирование модели-генератора

Для обучения seq2seq моделей требуется объединение текстовых полей датасета, что реализовано с помощью форматированного промпта. Такой подход показал себя лучше всего для систем RAG [19], так как имитирует процесс инференса.

Важную роль играет способ разметки входных и выходных данных. Исследованы два варианта: первый — input содержит форматированный промпт с контекстом и запросом, output — целевой ответ; второй — input и output совпадают, но в output все токены до метки ответа («[ANSWER]») замаскированы значением -100. Это позволяет функции потерь учитывать только токены ответа, что фокусирует модель на генерации финальной реплики [20].

В таблице 1 приведены дескриптивные характеристики полного датасета, обучающей и тестовой подвыборок в токенизированном виде (разделение 90/10).

Таблица 1 – Описательные статистики токенизированного датасета

Статистика (в токенах, округлено до целых)	Полный датасет	Train	Test
количество	40 156 778	36 137 150	4 019 628
минимальный пример	396	396	436
максимальный пример	3282	3282	2288
средняя длина примера	1567	1567	1568
медианная длина примера	1673	1673	1674

Модель-генератор была обучена более чем на 36 миллионах токенов. На рисунке 2 представлено распределение длин экземпляров в полном датасете.

Распределение смещено вправо, есть выбросы, размером около 3000 токенов, но основная часть датасета состоит из примеров длиной в 1500-2000 токенов. Так как в нашем случае ресурсы ограничены, было решено не использовать мини-пакеты напрямую, во избежание необходимости паддинга и выравнивания датасета по длине.

Дообучение проходило на машине с видеоадаптером NVIDIA GeForce RTX 4080 SUPER (16 Гб видеопамяти), дообучаемая модель была квантизована в формате NF4 [21]. Квантизация проходила в 2 этапа: сначала веса переводились в 8-битный формат, а после в 4-битный, снижая ошибку квантизации без увеличения размера модели [21].

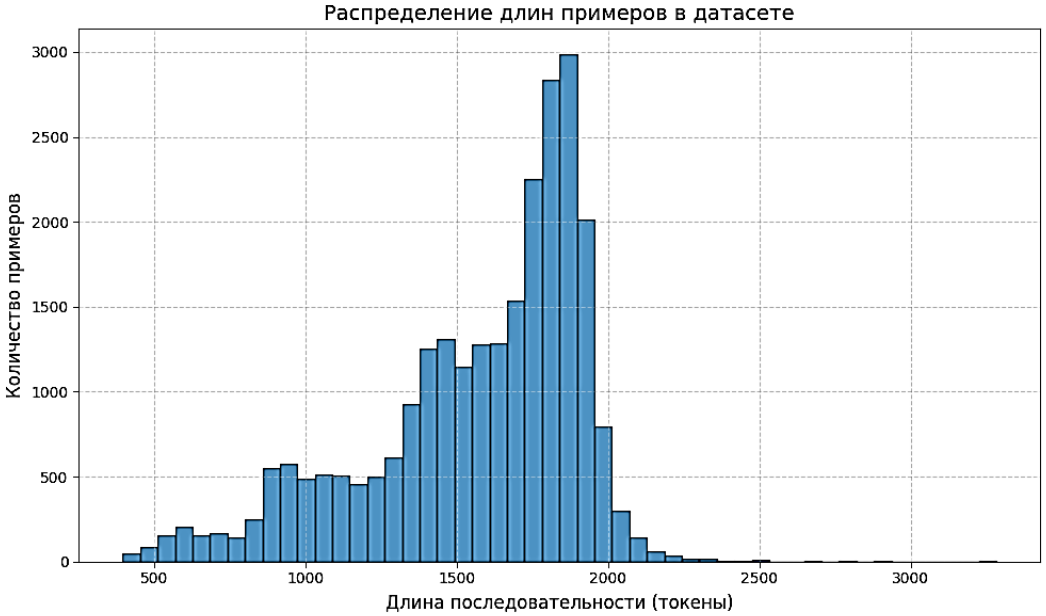

Рисунок 2 – Распределение длин экземпляров в датасете (в токенах)

В качестве метода дообучения модели выбрана LoRA [6] с параметрами:

1) $r = 8$ (ранг матриц A и B, аппроксимирующих матрицу градиентов, чем ниже, тем меньше параметров необходимо обучить);

2) *alpha* = 32 (коэффициент масштабирования, показывающий, насколько новые знания повлияют на имеющиеся знания – полученная LoRA-вставка усиливается в *alpha/r* раз перед слиянием с моделью);

4) *dropout* = 0.05;

5) дообучаемые модули: *q_proj*, *k_proj*, *v_proj*, *o_proj* – модули блоков внимания и *gate_proj*, *up_proj*, *down_proj* – модули MLP.

Использование квантизации и механизма LoRA позволит существенно сократить ресурсные затраты на процесс обучения. Так, после квантизации модель LLaMA-3.1-8b имеет 1 050 939 392 обучаемых параметра. При включении LoRA-адаптеров при указанных выше параметрах модель содержит только 20 971 520 обучаемых параметров (поскольку в данном случае исходные параметры модели замораживаются), тем самым мы уменьшили число обучаемых параметров более чем в 50 раз относительно квантизованной версии и в 400 относительно исходной.

В качестве главных параметров обучения были выбраны следующие:

1) *learning_rate* = 2e-4;

2) *weight_decay* = 1e-3;

3) *warmup_ratio* = 0.1 (10% от общего числа шагов обучения будут использоваться для постепенного увеличения learning_rate от нуля до 2e-4);

4) *num_epoch* = 25;

5) *gradient_accumulation_steps* = 8 (градиентов матриц A и B обновляются каждые 8 шагов, имитируя batch_size равный 8, без увеличения расхода видеопамяти);

6) *batch_size* = 1 (фактически на используемой GPU в один момент будет обрабатываться только 1 экземпляр данных).

Процесс обучения для каждого способа передачи данных в модель на одной GPU NVIDIA GeForce RTX 4080 SUPER занял около 200 часов. На рисунке 3 показаны графики изменения функции потерь (кросс-энтропия) и средней точности предсказания токенов. Поскольку для разных способов графики практически идентичны, приведем только второй.

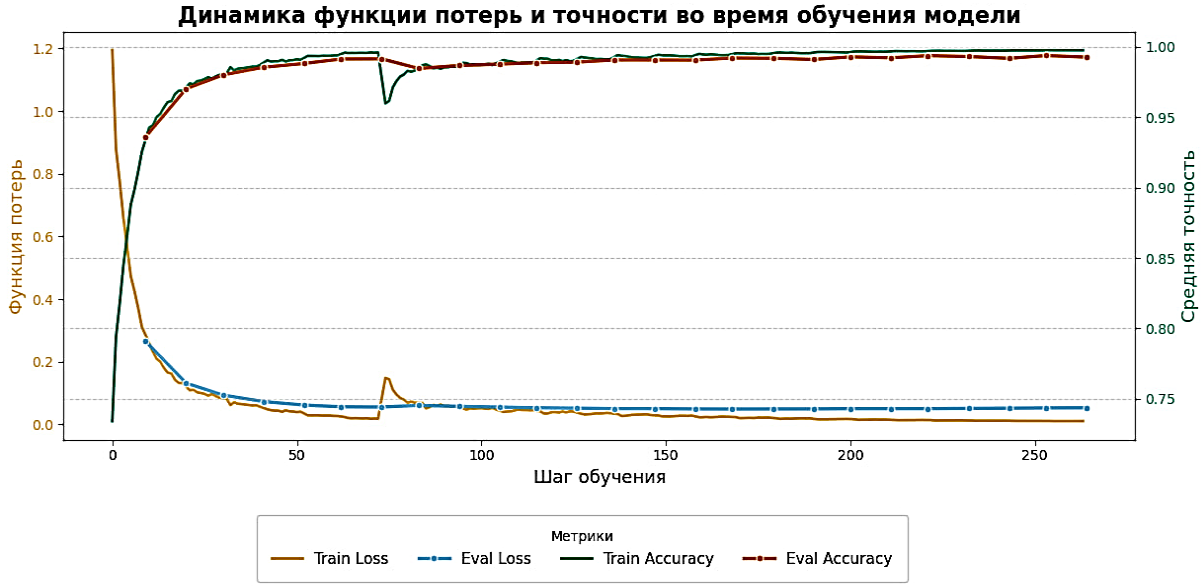

Рисунок 3 – График изменения функции потерь и средней точности предсказания токенов

На финальной, 25-й эпохе средняя точность модели на тестовой выборке при первом способе дообучения составила 99.35%, а при втором – 99.26%. Наилучшую точность на тестовой выборке при первом способе обучения модель показала после 24-й эпохи (99.45%). При втором способе обучения лучшую точность модель также показала после 24-й эпохи, и она составила 99.38%. Значение функции потерь на финальной эпохе при первом способе обучения составило примерно 0.04, лучший результат наблюдался на 16-й, составив примерно 0.042. При втором способе дообучения значение функции потерь на финальной эпохе составило примерно 0.05, лучший результат, равный 0.049, модель показала на 16-й. Также исходя из графиков можно заметить отсутствие значительного разрыва между качеством работы модели на тренировочных и на тестовых данных, что дает возможность предположить отсутствие факта переобучения модели.

Перед тестированием дообученной модели требуется интеграция LoRA-адаптеров в исходные веса. Если модель была квантизована, а адаптеры — bfloat16, то для слияния сначала проводится деквантизация либо загрузка в формате bfloat16 и объединение весов, что требует большой видеопамяти и работает не с любым оборудованием. Для проверки была реализована интеграция обоими способами: при ограниченной

памяти модель выгружалась на CPU, делилась на части до 5 Гб и объединялась пофрагментно на GPU, позволяя интегрировать адаптеры без дополнительной квантизации.

Для сравнения качества моделей использовали 30% примеров из контекстно-инструкционного датасета. Ответы на вопросы сначала генерировала базовая, затем дообученные версии LLaMA-3.1-8b. Для оценки применяли BLEU [22], ROUGE [23] и косинусное сходство (Giga-Embeddings-instruct), затем усредняли значения по тестовой подвыборке (7690 примеров). В таблице 2 приведены результаты для бейзлайна и дообученных моделей. Суффиксы «sep» — для моделей с раздельной подачей context в inputs и labels; «un» — для моделей с полным промптом с маскировкой; приставка «nod» — для моделей слияния весов без квантизации.

Таблица 2 – Усредненные метрики сходства эталонных ответов и ответов, сгенерированных классической LLaMA-3.1-8b и ее дообученными версиями

Метрика	baseline-llama	tuned-uni-llama	nod-tuned-uni-llama	tuned-sep-llama	nod-tuned-sep-llama
BLEU	0.177851	0.301470	**0.311701**	0.114690	0.130832
ROUGE-1	**0.152560**	0.138591	0.133058	0.112703	0.117614
ROUGE-2	0.065618	**0.074913**	0.072057	0.043937	0.046375
ROUGE-L	**0.151232**	0.137470	0.132025	0.110703	0.115748
cosine-similarity	0.716595	0.755788	**0.761709**	0.675485	0.685384

Обучение моделей с раздельной подачей промпта и ответа дало худший результат, чем базовая LLaMA-3.1-8b, то есть этот метод обучения неэффективен. Зато обучение с одинаковыми inputs и labels (с маскированием части запроса у labels) дало двукратный прирост BLEU, небольшой прирост ROUGE-2, незначительное отставание по ROUGE-1, ROUGE-L, а также преимущество по семантической близости генераций к целевым ответам. Модель уверенно генерирует биграммы, но немного уступает по отдельным словам и предложениям при ROUGE-сравнении, зато превосходит по смысловой близости.

Преимущество моделей с интегрированными без квантизации весами минимально и проявляется на третьих-четвертых знаках после запятой, то есть деквантизация весов после квантизации негативно не влияет при отсутствии возможностей для слияния весов в исходном виде. Все дообученные варианты моделей доступны на HuggingFace [24].

Реализация сервиса интеллектуального ассистента

После дообучения модели можно строить прикладной сервис, включающий RAG пайплайн и пользовательский интерфейс взаимодействия с системой. На рисунке 5 представлена структура реализованной системы интеллектуального ассистента.

Векторная база знаний предварительно наполнялась релевантными документами, использованными для инструкционного датасета. В базе документы делились на фрагменты по 2000 токенов с перекрытием 200 (Giga-Embeddings-instruct). Для обновления базы создан инструмент, не требующий изменений в системе.

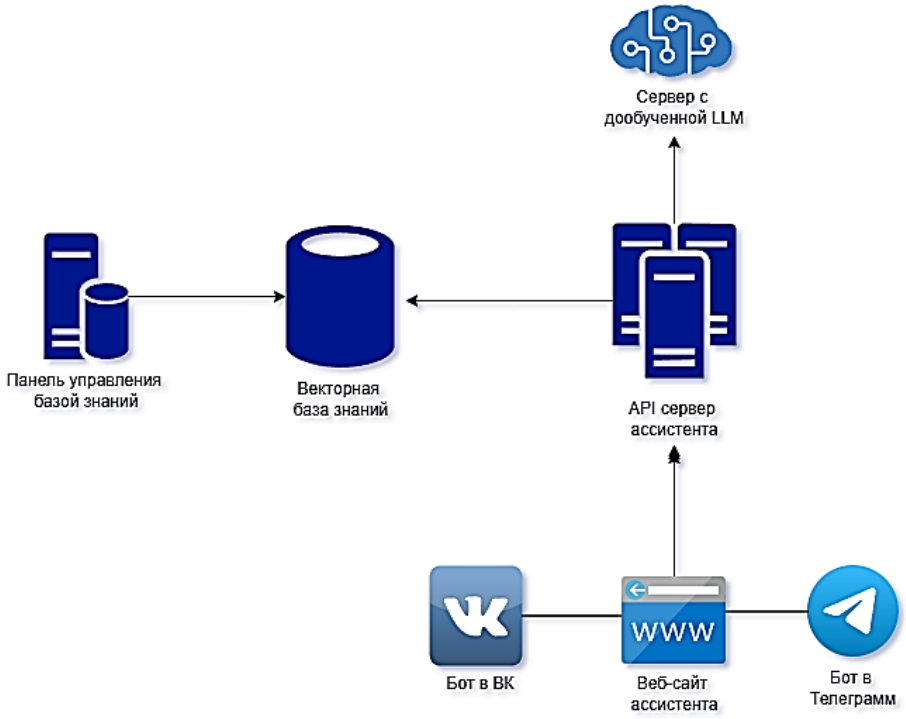

Рисунок 5 – Структура системы интеллектуального ассистента

Дообученная модель размещена на сервере с RTX 4080 SUPER, работая через LMStudio (OpenAI API совместимость). Использована 4-битная квантизация Q4_K_M (K-means), которая сокращает размер при сохранении производительности [25], и формат GGUF, поддерживающий запуск на CPU [26].

API ассистента реализован на FastAPI и обслуживает клиентские приложения, хранит историю чатов (20 последних сообщений), автоматически перефразирует запросы с учётом истории для лучшего понимания и разрешения местоименной анафоры, извлекает 3 ближайших документа из базы и формирует промпт для LLM.

В качестве клиентов доступны веб-чат, боты в ВК и Telegram, а также API для сторонних сервисов. Весь исходный код находится в открытом доступе [27].

Выводы

Разработана методика создания ассистентов на основе LLM для образовательных организаций, реализован ассистент для ДонГУ. Методика использует дообученную на контекстно-инструкционном датасете LLM как генератор для RAG, что преодолевает ограничения раздельного применения дообучения и RAG.

Для ассистента ДонГУ была выбрана модель LLaMa-3.1-8b, дообученная QLoRA на датасете, собранном с сайта университета. Модель показала результаты выше базовой на 13% (BLEU) и 4.5% (векторная близость). После сравнения лучшая версия интегрирована в систему RAG с обновляемой базой знаний и реализован сервис с клиентскими приложениями.

Весь исходный код находится в открытом доступе, что упрощает использование методики для других учебных заведений.

Список литературы

1. Wu S. et al. Bloomberggpt: A large language model for finance //arXiv preprint arXiv:2303.17564. – 2023.
2. Suchin Gururangan, Ana Marasovic, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. arXiv preprint arXiv:2004.10964, 2020.
3. Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen-tau Yih, Tim Rockt ¨ aschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. Advances in Neural Information Processing Systems, 33:9459–9474, 2020.
4. Hu, E., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L. & Chen, W. Lora: Low-rank adaptation of large language models, International Conference OnLearning Representations. (2021).
5. Андриевская, Н. К. Гибридная интеллектуальная мера оценки семантической близости / Н. К. Андриевская // Проблемы искусственного интеллекта. – 2021. – № 1(20). – С. 4-17. – EDN ZDZKGK.
6. Scott Barnett, Stefanus Kurniawan, Srikanth Thudumu, Zach Brannelly, and Mohamed Abdelrazek. Seven failure points when engineering a retrieval augmented generation system. In Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI, pp. 194–199, 2024.
7. Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A. & Others Training language models to follow instructions with human feedback, Advances In Neural Information Processing Systems. 35 pp. 27730-27744 (2022).
8. Eliseev, V.; Maksimova, A.; Bondarenko, V. Transforming raw corporate texts into instruction dataset for fine-tuning generator within a RAG system. System Informatics 2025, 27, 77-92.
9. Zhang T. et al. Raft: Adapting language model to domain specific rag //First Conference on Language Modeling. 2024.
10. Liu W. et al. RAG-Instruct: Boosting LLMs with Diverse Retrieval-Augmented Instructions //arXiv preprint arXiv:2501.00353. 2024.
11. Методические рекомендации «Методические рекомендации представления информации об образовательной организации высшего образования в открытых источниках с учетом соблюдения требований законодательства в сфере образования» (вступают в силу с 01.09.2024) // Федеральная служба по надзору в сфере образования и науки. – 2024
12. Eliseev Vadim. Website Crawler [Электронный ресурс]. URL: https://github.com/EliseevVadim/WebsiteCrawler.
13. Team G. et al. Gemma 2: Improving open language models at a practical size //arXiv preprint arXiv:2408.00118. 2024.
14. Eliseev Vadim. Texts Estimator [Электронный ресурс]. URL: https://github.com/EliseevVadim/TextsEstimator.
15. Lee M. H. J. Examining the Robustness of Homogeneity Bias to Hyperparameter Adjustments in GPT-4 //arXiv preprint arXiv:2501.02211. – 2025.
16. Eliseev Vadim. Instructions Generator [Электронный ресурс]. URL: https://github.com/EliseevVadim/InstructionsGenerator.
17. Giga-Embeddings-instruct | HuggingFace [Электронный ресурс]. URL: https://huggingface.co/ai-sage/Giga-Embeddings-instruct.
18. Kaplan J. et al. Scaling laws for neural language models //arXiv preprint arXiv:2001.08361. – 2020.
19. Rodrigues J., Branco A. Meta-prompting Optimized Retrieval-Augmented Generation //EPIA Conference on Artificial Intelligence. – Cham: Springer Nature Switzerland, 2024. С. 203-214.
20. Huerta-Enochian M., Ko S. Y. Instruction Fine-Tuning: Does Prompt Loss Matter? //arXiv preprint arXiv:2401.13586. – 2024.
21. Dettmers T. et al. Qlora: Efficient finetuning of quantized llms //Advances in neural information processing systems. 2023. Т. 36. С. 10088-10115.
22. Reiter E. A structured review of the validity of BLEU //Computational Linguistics. 2018. Т. 44. №. 3. С. 393-401.
23. Lin C. Y. Rouge: A package for automatic evaluation of summaries //Text summarization branches out. 2004. С. 74-81.
24. Eliseev Vadim. DonSU LLaMa Family [Электронный ресурс]. URL: https://huggingface.co/collections/insidious316/donsu-llama-family.
25. Zhao E. et al. Quantitative Analysis of Performance Drop in DeepSeek Model Quantization //arXiv preprint arXiv:2505.02390. – 2025.
26. Kavathekar I. et al. Small Models, Big Tasks: An Exploratory Empirical Study on Small Language Models for Function Calling //arXiv preprint arXiv:2504.19277. 2025.
27. Eliseev Vadim. DonSU Intelligent Assistant [Электронный ресурс]. URL: https://github.com/EliseevVadim/DonSU-Intelligent-Assistant.

References

1. Wu S. et al. Bloomberggpt: A large language model for finance //arXiv preprint arXiv:2303.17564. – 2023.
2. Suchin Gururangan, Ana Marasovic, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. arXiv preprint arXiv:2004.10964, 2020.
3. Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen-tau Yih, Tim Rockt ¨ aschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. Advances in Neural Information Processing Systems, 33:9459–9474, 2020.
4. Hu, E., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L. & Chen, W. Lora: Low-rank adaptation of large language models, International Conference On Learning Representations. (2021).
5. Andrievskaya N. K. Hybrid Intelligent Measure for Assessing Semantic Similarity // Problems of Artificial Intelligence. – 2021. – No. 1(20). – Pp. 4-17.
6. Scott Barnett, Stefanus Kurniawan, Srikanth Thudumu, Zach Brannelly, and Mohamed Abdelrazek. Seven failure points when engineering a retrieval augmented generation system. In Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI, pp. 194–199, 2024.
7. Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A. & Others Training language models to follow instructions with human feedback, Advances In Neural Information Processing Systems. 35 pp. 27730-27744 (2022).
8. Eliseev, V.; Maksimova, A.; Bondarenko, V. Transforming raw corporate texts into instruction dataset for fine-tuning generator within a RAG system. System Informatics 2025, 27, 77-92.
9. Zhang T. et al. Raft: Adapting language model to domain specific rag //First Conference on Language Modeling. – 2024.
10. Liu W. et al. RAG-Instruct: Boosting LLMs with Diverse Retrieval-Augmented Instructions //arXiv preprint arXiv:2501.00353. – 2024.
11. Methodical Guidelines: Methodical Guidelines for the Presentation of Information about a Higher-Education Institution in Open Sources, Taking into Account Compliance with Education-Legislation Requirements (effective 01.09.2024) // Federal Service for Supervision in Education and Science, 2024.
12. Eliseev Vadim. Website Crawler [Электронный ресурс]. – URL: https://github.com/EliseevVadim/WebsiteCrawler.
13. Team G. et al. Gemma 2: Improving open language models at a practical size //arXiv preprint arXiv:2408.00118. – 2024.
14. Eliseev Vadim. Texts Estimator [Electronic resource]. – URL: https://github.com/EliseevVadim/TextsEstimator.
15. Lee M. H. J. Examining the Robustness of Homogeneity Bias to Hyperparameter Adjustments in GPT-4 //arXiv preprint arXiv:2501.02211. – 2025.
16. Eliseev Vadim. Instructions Generator [Electronic resource]. – URL: https://github.com/EliseevVadim/InstructionsGenerator.
17. Giga-Embeddings-instruct | HuggingFace [Электронный ресурс]. – URL: https://huggingface.co/ai-sage/Giga-Embeddings-instruct.
18. Kaplan J. et al. Scaling laws for neural language models //arXiv preprint arXiv:2001.08361. – 2020.
19. Rodrigues J., Branco A. Meta-prompting Optimized Retrieval-Augmented Generation //EPIA Conference on Artificial Intelligence. – Cham: Springer Nature Switzerland, 2024. – С. 203-214.
20. Huerta-Enochian M., Ko S. Y. Instruction Fine-Tuning: Does Prompt Loss Matter? //arXiv preprint arXiv:2401.13586. – 2024.
21. Dettmers T. et al. Qlora: Efficient finetuning of quantized llms //Advances in neural information processing systems. – 2023. – Т. 36. – С. 10088-10115.
22. Reiter E. A structured review of the validity of BLEU //Computational Linguistics. 2018. Т. 44. №. 3. – С. 393-401.
23. Lin C. Y. Rouge: A package for automatic evaluation of summaries //Text summarization branches out. – 2004. – С. 74-81.
24. Eliseev Vadim. DonSU LLaMa Family [Electronic resource]. – URL: https://huggingface.co/collections/insidious316/donsu-llama-family.
25. Zhao E. et al. Quantitative Analysis of Performance Drop in DeepSeek Model Quantization //arXiv preprint arXiv:2505.02390. – 2025.
26. Kavathekar I. et al. Small Models, Big Tasks: An Exploratory Empirical Study on Small Language Models for Function Calling //arXiv preprint arXiv:2504.19277. – 2025.
27. Eliseev Vadim. DonSU Intelligent Assistant [Electronic resource]. URL: https://github.com/EliseevVadim/DonSU-Intelligent-Assistant.

RESUME

V. O. Eliseev, V. I. Bondarenko, A. Y. Maksimova

Development of an Intelligent Assistant for an Educational Institution

Background: timely provision of reference information is crucial in both education and business. Previously, this problem was addressed through human support staff, what caused big delays. Advances in Natural Language Processing (NLP) and the emergence of Large Language Models (LLM) enabled the automation of this process by developing intelligent assistants. The main goal of this study is creating the method that enables building of such assistants for educational institutions and which is applicable for majority of Russian universities with minimal effort. Also, the intelligent assistant of Donetsk State University (DonSU) was built using this method.

Materials and methods: various methods of integrating of corporate data into LLMs were analyzed among with their disadvantages. The combined approach of LLMs fine-tuning and building of Retrieval Augmented Generation (RAG) systems was proposed. It involves fine-tuning an LLM using the context based instructional dataset that consists of input-context-output triples and using it as a generator in RAG system. As part of applying the method for DonSU, a dataset of the proposed format was built based on raw textual data from the university's website. The LLaMA-3.1-8B model was then fine-tuned on this dataset to serve as a generator. The Giga-Embeddings-Instruct model was used as the retriever for dataset construction and in the system in general.

Results: the LLaMA-3.1-8B model fine-tuned on the built dataset demonstrated a 13% improvement over the base model in BLEU score and a 4.5% increase in cosine similarity between generated and target responses across several interaction scenarios. The resulting best-performing model was integrated into the RAG-based intelligent assistant system of DonSU, along with a vector database populated with content from the university's website and an implemented mechanism for its updating. Additionally, an applied service was developed, enabling interaction with the assistant via a web interface and bots in Telegram and VK.

Conclusion: the proposed method for building intelligent assistants proved effective in the development of the DonSU intelligent assistant. The final system produces high-quality responses to user queries while requiring minimal resources during development. The source code for the scripts used to collect and prepare the context based instructional dataset, as well as the code for the DonSU intelligent assistant system, is open source, making it easy to implement the approach for other educational institutions.

РЕЗЮМЕ

В. О. Елисеев, В. И. Бондаренко, А. Ю. Максимова

Разработка интеллектуального ассистента для образовательной организации

Справочная информация: своевременное предоставление справочной информации востребовано как в образовании, так и в бизнесе. Ранее задача решалась вручную сотрудниками поддержки, что приводило к большим задержкам. Развитие технологий обработки естественного языка (NLP) и появление больших языковых моделей (LLM) позволили автоматизировать этот процесс за счет создания интеллектуальных ассистентов. Целью этой работы является разработка методики, позволяющей создавать интеллектуальных ассистентов для образовательных организаций, и применимой к различным вузам РФ с минимальными затратами, а также ее реализация для построения интеллектуального ассистента Донецкого Государственного университета (ДонГУ).

Материалы и методы: были рассмотрены существующие способы интеграции корпоративных данных в LLM, затронуты их недостатки. Предложен подход, комбинирующий дообучение LLM и построение систем генерации, дополненной поиском, путем дообучения модели на контекстно-инструкционном наборе данных (тройки запрос-контекст-ответ) и ее использования в качестве генератора в системе RAG. В рамках применения метода для ДонГУ был собран набор данных предложенного формата на основе сырых текстовых данных с сайта вуза, после чего на нем была дообучена LLaMA-3.1-8b для роли генератора. В качестве ретривера для формирования набора данных выступала модель Giga-Embeddings-instruct.

Результаты: дообученная на собранном датасете LLaMA-3.1-8b показала прирост по сравнению с базовой моделью на 13% по метрике BLEU и на 4.5% по косинусному сходству генераций к целевым ответам для ряда сценариев взаимодействия. Полученная лучшая модель была интегрирована в систему RAG интеллектуального ассистента ДонГУ вместе с векторной базой данных, наполненной содержимым сайта университета и реализованным механизмом ее актуализации. Также был реализован прикладной сервис, позволяющий взаимодействовать с ассистентом с помощью веб-интерфейса и ботов в Telegram и VK.

Вывод: предложенная методика построения интеллектуальных ассистентов успешно продемонстрировала себя при разработке интеллектуального ассистента для ДонГУ, полученная система генерирует качественные ответы на запросы пользователей при использовании минимальных ресурсов при ее разработке. Исходный код скриптов по сбору и подготовке контекстно-инструкционного датасета, а также код системы интеллектуального ассистента ДонГУ находятся в открытом доступе, что позволяет легко имплементировать подход для других образовательных организаций.

Елисеев Вадим Олегович – стажер-исследователь лаборатории интеллектуальных систем Федерального государственного бюджетного научного учреждения Институт прикладной математики и механики.
Область научных интересов: искусственный интеллект, машинное обучение, нейронные сети, обработка естественного языка, генеративные и большие языковые модели.

Бондаренко Виталий Иванович – кандидат технических наук, доцент кафедры компьютерных технологий физико-технического факультета, Федерального государственного бюджетного образовательного учреждения высшего образования "Донецкого Государственного Университета".
Область научных интересов: искусственный интеллект, интеллектуальный анализ данных, машинное обучение, математическое моделирование гидро- и теплофизических процессов, разработка пользовательских интерфейсов для прикладных программ моделирования.

Максимова Александра Юрьевна – кандидат технических наук, заведующий лабораторией интеллектуальных систем, Федерального государственного бюджетного научного учреждения Институт прикладной математики и механики.
Область научных интересов: машинное обучение, анализ данных, распознавание образов, миварный логический вывод, нейросетевые модели.

Статья поступила в редакцию 15.10.2025.