МЕТОД ПОСТРОЕНИЯ КОНТЕКСТНО-ИНСТРУКЦИОННОГО НАБОРА ДАННЫХ ДЛЯ ДООБУЧЕНИЯ ГЕНЕРАТОРА В СИСТЕМЕ RAG

В.И. Бондаренко[1], **В.О. Елисеев**[2], А.Ю. Максимова[2],

[1]ДонГУ, г. Донецк
[2]ИПММ, г. Донецк

mail@vitbond.ru, eliseevv02@mail.ru, maximova.alexandra@mail.ru

С ростом количества и качества современных больших языковых моделей (LLM) появилась возможность создания интеллектуальных ассистентов, выполняющих консультационную и справочную роль в бизнес-процессах конкретных организаций. Однако мы не можем использовать LLM в чистом виде, так как в данных, на которых они обучались, нет достаточной информации об организациях, для которых строятся ассистенты.

Наиболее распространенные решения этой проблемы – дообучение LLM [1] на корпоративных данных и построение систем Retrieval Augmented Generation (RAG) [2]. Эти подходы позволяют эффективно использовать LLM для работы с корпоративными данными, однако обладают рядом недостатков.

Дообучение LLM требует значительных ресурсов и большого количества качественно размеченных данных. Главным недостатком подхода является необходимость постоянного повторного обучения модели на обновляемых данных. Даже использование самого экономного подхода к дообучению – Low-Rank Adaptation (LoRA) [3] не подходит для частого повторного обучения модели в условиях ограниченности вычислительных мощностей.

Использование RAG позволяет нам гибко взаимодействовать с корпоративными знаниями, так как в данном случае мы не обучаем модель, а передаем ей факты из базы знаний вместе с запросом. Сами факты выбираются за счет схожести с запросом на основе косинусного сходства либо L2 расстояния. Векторизация текста [4], позволяющая искать схожесть, происходит с помощью модели-ретривера, чаще всего LLM, состоящей только из энкодера [5]. Однако, RAG зависим от качества ретривера, ошибки в его работе могут нарушить работу системы. Кроме того, мы не можем напрямую управлять стилем ответов модели.

В данной работе мы предлагаем объединить эти подходы, дообучив LLM на наборе данных со специфичным форматом в качестве генератора в системе RAG. Вместо обычного инструкционного набора данных [6], состоящего из пар input-output, мы предлагаем использовать набор, с добавленным полем context, заполняемым top-n текстовыми фрагментами из векторной базы данных, построенной на корпоративных данных. Метод позволит модели научиться извлекать наиболее релевантные ответы из найденных ретривером фактов, в зависимости от запроса пользователя, вместо простого запоминания фактов.

Схожий метод был проанализирован в работах [7, 8], однако работы были направлены на решение общей задачи при использовании мультидоменных

знаний. В нашем случае мы построим контекстно-инструкционный набор данных для обучения модели-генератора RAG системы интеллектуального ассистента Донецкого Государственного Университета (ДонГУ).

В качестве источника данных был использован сайт ДонГУ [9], поскольку он содержит структурированный и машинночитаемый контент. Для парсинга был разработан набор скриптов [10], позволяющий указать тег с текстовым содержимым на страницах сайта, необходимость загружать файлы текстового формата (.pdf, .docx, .pptx и т.д.), и глубину обхода. После полного обхода сайта было загружено 2912 текстовых файлов (полученных на основе HTML-страниц), 3546 файлов различных форматов, суммарно было выгружено 14.3 Гб данных.

При анализе текстовых файлов, полученных при парсинге HTML страниц сайта, было обнаружено, что некоторые из них не содержат релевантной для разрабатываемого ассистента информации. Они были получены из промежуточных страниц, содержащих только ссылки на скачиваемые файлы либо на другие страницы. Для удаления непригодных к генерации инструкций файлов использовались в качестве судей три большие языковые модели: Gemma2-9b-it [11], LLaMA-3.1-70B-Instruct и LLaMA-3.3-70B-Instruct. Был создан запрос, включающий название и содержимое файла, и задание оценить пригодность этих данных для генерации инструкций по пятибальной шкале. После оценки было принято решение удалить файлы, у которых средняя оценка меньше двух. Проект с выбором релевантных текстовых файлов, вместе с запросами на русском и английском языках находится в открытом доступе [12].

Далее была проведена генерация инструкционного набора данных на основе 2337 оставшихся после предыдущего шага текстовых файлов. Был построен запрос, содержащий название и содержимое файла, и задание сгенерировать максимальное число пар инструкция-ответ. Он был отправлен для каждого релевантного файла модели LLaMA-3.3-70B-Instruct. Для файлов, с менее чем тремя полученными инструкциями, была проведена повторная генерация с пониженным значением температуры. Это добавило более 600 новых инструкций, всего мы получили 25633 инструкционные пары, что достаточно для обучения модели на корпоративном домене.

Построенный классический инструкционный набор данных был расширен контекстом, в соответствии с input. На основе релевантных текстовых файлов была построена векторная база с помощью модели Giga-Embeddings-instruct [13]. Размер фрагмента составил 500 токенов модели LLaMA-3.1-8B с перекрытием в 50, что позволит как получить достаточно описательные текстовые фрагменты, так и избежать сильной ресурсной нагрузки во время дообучения. Из 2337 релевантных текстовых файлов было выделено 9935 фрагментов.

Для каждого input из полученного ранее набора данных выбраны три наиболее близких фрагмента из векторной базы (по косинусному сходству) и добавлены в поле context, сформировав набор данных в формате input-context-output. Проект с созданием итогового набора данных также находится в открытом доступе [14].

В дальнейшем планируется дообучить модель LLaMA-3.1-8B на построенном в работе наборе данных и использовать ее в качестве генератора в системе RAG интеллектуального ассистента ДонГУ. В данной системе будет использована векторная база данных с текстовыми фрагментами большего размера – 2000 токенов модели Giga-Embeddings-instruct с перекрытием в 200, включающая не только рассмотренные в данной работе релевантные текстовые файлы, но и файлы различных форматов, выгруженные с сайта ДонГУ.

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).

Список литературы:

1. Gururangan S. [et al.]. Don't stop pretraining: Adapt language models to domains and tasks // arXiv:2004.10964. 2020.
2. Lewis P. [et al.]. Retrieval-augmented generation for knowledge-intensive nlp tasks // Adv Neural Inf Process Syst. 2020. Vol. 33. Pp. 9459–9474.
3. Hu E.J. [et al.]. Lora: Low-rank adaptation of large language models // ICLR. 2022. Vol. 1. Issue 2. P. 3.
4. Mikolov T. [et al.]. Distributed representations of words and phrases and their compositionality // Adv Neural Inf Process Syst. 2013. Vol. 26.
5. Karpukhin V. [et al.]. Dense Passage Retrieval for Open-Domain Question Answering // EMNLP. 2020. Pp. 6769–6781.
6. Nayak N.V. [et al.]. Learning to generate instruction tuning datasets for zero-shot task adaptation // arXiv:2402.18334. 2024.
7. Zhang T. [et al.]. Raft: Adapting language model to domain specific rag // First Conference on Language Modeling. 2024.
8. Liu W. [et al.]. RAG-Instruct: Boosting LLMs with Diverse Retrieval-Augmented Instructions // arXiv:2501.00353. 2024.
9. Официальный сайт ДонГУ. URL: https://donnu.ru/.
10. Website Crawler. URL: https://github.com/EliseevVadim/WebsiteCrawler
11. Team G. et al. Gemma 2: Improving open language models at a practical size // arXiv:2408.00118. 2024.
12. TextsEstimator. URL: https://github.com/EliseevVadim/TextsEstimator
13. Giga-Embeddings-instruct. Карточка модели на HuggingFace URL: https://huggingface.co/ai-sage/Giga-Embeddings-instruct.
14. InstructionsGenerator. URL: https://github.com/EliseevVadim/InstructionsGenerator