УДК 004.891.2

РАЗРАБОТКА И ОПТИМИЗАЦИЯ LLM-СЕРВИСА ДЛЯ КОНСУЛЬТИРОВАНИЯ ПОЛЬЗОВАТЕЛЕЙ ПО ЭКСПЛУАТАЦИИ ПЛАТФОРМЫ — СРАВНЕНИЕ ПОДХОДОВ ПРОЕКТИРОВАНИЯ ВОПРОСНО-ОТВЕТНЫХ СИСТЕМ

Хисаметдинова Д.Н. (Университет ИТМО), **Леманов А.А**. (Университет ИТМО)
Научный руководитель – кандидат технических наук Ходненко И. В.
(Университет ИТМО)

Введение. QA-системы позволяют быстро ориентироваться в объёмной базе знаний, снижая нагрузку пользователей. Применение больших языковых моделей (LLM) и Retrieval-Augmented Generation (RAG) позволяет генерировать точные ответы, опираясь на релевантный контекст документации. Однако практическая реализация подобных систем сопряжена с рядом проблем: embedding представления вопросов и ответов могут оказаться слишком разобщенными, а терминологические несоответствия приводят к недостатку совпадений при поиске.

Основная часть. LLM, такие как GPT, BERT и их аналоги, демонстрируют высокую способность к генерации текста и пониманию контекста, что делает их привлекательными для создания QA-систем. Однако их ограничения связаны с необходимостью тонкой настройки (fine-tuning) для специфических задач и высокими вычислительными затратами.

RAG сочетает методы извлечения информации из внешних источников и генерации ответов на основе извлеченных данных. Этот подход позволяет улучшить точность ответов за счет использования релевантной информации из документации, но требует эффективных алгоритмов поиска и интеграции данных.

Были проведены следующие этапы проектирования:

1. Исходная база представлена JSON-файлом, содержащим контекст документации, разбитый на чанки по тематикам, списки вопросов и соответствующих ответов. Данные последовательно извлекаются и агрегируются в пары «вопрос–ответ» с сохранением контекста. Генерация 5-10 синонимичных вопросов и их добавление к имеющимся данным перед векторизацией уменьшает семантическое расстояние между запросом и данными базы.
2. Было проведено написание сервисов с двумя LLM (у одной 32 миллиарда параметров, у другой 8, вследствие чего дала четырехкратный прирост в скорости, но она дообучалась на данных на русском языке). При добавлении контекста в пайплайн, косинусное сходство между парами векторов первой оказалось на 10 процентных пункта больше, чем второй.
3. Автоматическое извлечение ключевых слов для каждого вопроса в базе NLP-методами дало косинусное сходство менее 0.4. На этапе генерации статического .json скорость не имеет значения, в отличие от релевантности терминов, поэтому для этой задачи лучше подходит LLM. Для каждого вопроса были вычислены эмбеддинги с использованием Sentence-BERT, показавшего наилучшее сочетание скорости и точности среди протестированных моделей. Полученные векторы сохранялись и индексировались в FAISS, что обеспечило ускорение поиска на 32% по сравнению с последовательным сравнением векторов. Параллельно реализовывался BM25+, который предварительно отбирал кандидатов, сокращая размер выборки на 78% перед векторным сравнением.
4. Запрос пользователя уточняется LLM, которая повысила точность поиска на 14% за счет приведения к терминологии документации и генерации синонимичных вариантов. Это уменьшило среднее семантическое расстояние между запросом и данными на 18%, что позволило находить более релевантные ответы, хотя и в

среднем увеличило скорость генерации на 2-15 секунд в зависимости от модели.

5. Были проанализированы традиционные частотные методы ранжирования и нейронные подходы на основе эмбеддингов. Оптимальная комбинация BM25+ для предварительного отбора, FAISS для векторного поиска и BERT для уточнения запроса обеспечила баланс точности и скорости. Внедрение фильтрации нерелевантных тем и оптимизации параметров модели позволило достичь среднего косинусного сходства между векторами пользовательских запросов и векторами найденных наиболее релевантных вопросов более 0.87 и ускорить поиск на 32% по сравнению с полным перебором кандидатов.
6. По найденному релевантному вопросу определяется индекс в базе, и на его основе извлекается соответствующий ответ. Если не найдено точного совпадения, LLM генерирует ответ на основе найденного контекста, что делает систему адаптивной.
7. Для демонстрации работы был развёрнут сервис с балансировкой нагрузки на FastAPI и Angular.

Выводы. При разработке QA-систем необходимо балансировать между временем ожидания ответа и его релевантностью. Использование двух различных LLM (моделей с 32B и 8B параметрами) показало, что более компактная модель обеспечила четырехкратное ускорение работы, но снижение метрики косинусного сходства на 10%. Расширение выборки за счет предварительной генерации вопросов, синонимичных с теми, что есть в исходной БД является быстрым методом улучшения качества ответа. Гибридный поиск, где уточненный запрос сначала проходит BM25-фильтрацию, а затем обрабатывается FAISS, сочетает текстовый и векторный поиски, дал прирост скорости на 32% по сравнению с полным перебором всех кандидатов. Реализация сервиса демонстрирует практическую применимость данного подхода, однако дальнейшая оптимизация (например, кэширование обращений к LLM) остается перспективным направлением для повышения скорости и стабильности работы системы.

Список использованных источников:

1. Lewis, P., et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. arXiv preprint arXiv:2005.11401.
2. Brown, T., et al. (2020). Language Models are Few-Shot Learners. arXiv preprint arXiv:2005.14165.
3. Devlin, J., et al. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv preprint arXiv:1810.04805.
4. Raffel, C., et al. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. arXiv preprint arXiv:1910.10683.
5. Носков Д. В. Классификация текстов при помощи алгоритмов машинного обучения // Вестник науки и образования. 2018. №4 (40).