УДК 004.89

RETRIEVAL-AUGMENTED GENERATION: АРХИТЕКТУРА, МЕТОДЫ И НАПРАВЛЕНИЯ РАЗВИТИЯ

Вершинин М.В., магистрант гр. ПИм-241, II курс
Научный руководитель: Пимонов А.Г., д.т.н., профессор
Кузбасский государственный технический университет имени Т.Ф. Горбачёва
г. Кемерово

1. Введение

Большие языковые модели (Large Language Models, LLM) продемонстрировали впечатляющие результаты в задачах генерации текста, ответов на вопросы, суммаризации и рассуждения. Вместе с тем они обладают рядом существенных ограничений: склонностью к генерации фактически некорректных ответов (так называемые «галлюцинации»), невозможностью обновления знаний без повторного обучения, а также отсутствием доступа к закрытым или специализированным источникам информации [1].

Одним из наиболее перспективных подходов к решению данных проблем является Retrieval-Augmented Generation (RAG) – архитектура, предложенная Lewis et al. в 2020 году [2]. RAG дополняет генеративную модель компонентом информационного поиска, позволяя извлекать релевантные фрагменты из внешних источников и использовать их в качестве контекста при генерации ответа. За последние годы RAG превратилась из исследовательской концепции в стандартный архитектурный паттерн, широко применяемый в промышленных системах. Целью настоящей работы является систематизация существующих подходов к построению RAG-систем, анализ их компонентной архитектуры и выявление ключевых направлений развития.

2. Архитектура RAG-систем

Общая схема работы. Классическая архитектура RAG включает три основных этапа: индексацию, извлечение и генерацию [2, 3]. На этапе индексации исходные документы разбиваются на фрагменты (chunks), для каждого из которых вычисляется векторное представление (embedding) с помощью модели-энкодера. Полученные векторы сохраняются в специализированном хранилище, обеспечивающем эффективный поиск по семантическому сходству.

На этапе извлечения пользовательский запрос также преобразуется в вектор, после чего выполняется поиск наиболее релевантных фрагментов по метрике сходства (как правило, косинусное расстояние). Найденные фрагменты формируют контекст, который передаётся генеративной модели вместе с исходным запросом.

На этапе генерации языковая модель формирует ответ, опираясь как на собственные параметрические знания, так и на предоставленный контекст.

Это позволяет модели оперировать актуальной и верифицируемой информацией, снижая вероятность галлюцинаций.

Компоненты системы. Принято выделять три основных составляющих RAG-системы:

1. Модель-энкодер. Она отвечает за преобразование текстовых фрагментов и запросов в векторные представления. Широко используются модели семейства Sentence-BERT, E5, BGE, а также коммерческие решения [4]. Качество энкодера напрямую определяет точность поиска: если семантически близкие тексты получают далёкие векторные представления, релевантные документы не будут найдены.

2. Векторное хранилище. Оно обеспечивает хранение и быстрый поиск по векторным представлениям. Существуют как специализированные решения, так и расширения традиционных СУБД, поддерживающие векторный поиск. Выбор хранилища определяется требованиями к масштабируемости, скорости поиска и функциональности фильтрации.

3. Генеративная модель. Именно она формирует итоговый ответ на основе контекста и запроса. В качестве генератора могут использоваться как открытые модели различного масштаба, так и коммерческие API. Ключевым требованием является способность модели корректно интерпретировать предоставленный контекст и следовать ему при генерации [1, 3].

3. Стратегии индексации и извлечения

Разбиение документов (Chunking). Выбор стратегии разбиения документов на фрагменты существенно влияет на качество поиска. Основные подходы включают: разбиение по фиксированному размеру с перекрытием, разбиение по семантическим границам (предложения, абзацы, разделы), а также рекурсивное разбиение, при котором документ итеративно делится на всё более мелкие единицы [3].

Размер фрагмента определяет компромисс между точностью и полнотой контекста. Слишком мелкие фрагменты могут терять контекст, слишком крупные – вносить шум и снижать релевантность. Практика показывает, что оптимальный размер зависит от предметной области и типа задач.

Методы улучшения поиска. В RAG-системах можно выделить несколько методов улучшения поиска.

1. Гибридный поиск. Такой поиск предполагает комбинирование семантического (векторного) поиска с лексическим (по ключевым словам, BM25). Гибридный подход позволяет компенсировать слабые стороны каждого метода: семантический поиск лучше обрабатывает парафразы и синонимы, тогда как лексический - точные термины и идентификаторы [5].

2. Переранжирование (Reranking). Это двухэтапная стратегия, при которой первоначальный набор кандидатов, полученный быстрым поиском, повторно оценивается более точной, но ресурсоёмкой моделью-ранкером. Это позволяет значительно повысить качество выдачи без пропорционального увеличения вычислительных затрат [3, 6].

3. Переформулирование запроса (Query Rewriting). Это преобразование пользовательского запроса перед поиском для улучшения соответствия между формулировкой вопроса и формулировками в базе знаний. Оно может осуществляться как с помощью языковой модели, так и с использованием методов расширения запроса [3].

4. Многошаговый поиск. Итеративный процесс, при котором система выполняет несколько последовательных запросов к базе знаний, уточняя контекст на каждом шаге. Такой поиск особенно эффективен для сложных вопросов, требующих синтеза информации из нескольких источников [3, 7].

4. Сравнительный анализ подходов к адаптации LLM

RAG является одним из трёх основных подходов к адаптации больших языковых моделей для работы с предметно-ориентированными данными. Два других – дообучение (fine-tuning) и промпт-инжиниринг (in-context learning). Результаты сравнительного анализа этих подходов представлены в таблице 1.

Таблица 1 – Сравнение подходов к адаптации LLM

Критерий	RAG	Дообучение (Fine-tuning)	Промпт-инжиниринг
Актуальность данных	Высокая (обновление базы без переобучения)	Низкая (требуется переобучение)	Низкая (ограничена обучением)
Вычислительные затраты	Средние (инференс + поиск)	Высокие (обучение GPU)	Низкие (только инференс)
Прозрачность	Высокая (источники можно проверить)	Низкая (знания в весах)	Низкая
Галлюцинации	Снижены (привязка к источникам)	Умеренные	Высокие
Адаптация к домену	Через базу знаний	Через обучающую выборку	Через примеры в промпте

Как видно из табл. 1, RAG обладает уникальным сочетанием преимуществ: высокая актуальность данных (база знаний обновляется без переобучения модели), прозрачность (ответы можно проверить по источникам) и умеренные вычислительные затраты. При этом подходы не являются взаимоисключающими: на практике часто используются гибридные схемы, например, дообученная модель с RAG-компонентом [3, 8].

5. Продвинутые архитектурные паттерны

Naive RAG. Базовый паттерн, включающий простое разбиение документов, векторный поиск по одному запросу и однократную генерацию. Прост в реализации, но ограничен в качестве для сложных задач [3].

Advanced RAG. Расширяет базовый паттерн за счёт предварительной и постобработки: переформулирование запросов, гибридный поиск, переранжирование результатов, фильтрация контекста. Обеспечивает существенно более высокое качество ответов при умеренном увеличении сложности [3].

Modular RAG. Архитектура, в которой каждый компонент (поиск, ранжирование, генерация, валидация) реализован как независимый модуль с возможностью замены и конфигурирования. Позволяет гибко адаптировать систему под конкретные задачи и итеративно улучшать отдельные компоненты [3].

Agentic RAG. Наиболее продвинутый паттерн, в котором языковая модель выступает в роли агента, самостоятельно принимающего решения о необходимости поиска, выборе источников, формулировке запросов и оценке достаточности полученной информации. Такой подход позволяет решать многошаговые задачи, требующие сложных цепочек рассуждений [7].

6. Ограничения и открытые вопросы

Несмотря на значительный прогресс, RAG-системы сталкиваются с рядом нерешённых проблем. Одной из ключевых является проблема «lost in the middle»: языковые модели склонны уделять больше внимания информации в начале и конце контекстного окна, игнорируя релевантные фрагменты в середине [9].

Другой важной проблемой является оценка качества RAG-систем. В отличие от классических задач информационного поиска, где существуют устоявшиеся метрики, оценка RAG требует учёта как качества поиска, так и качества генерации, а также их взаимодействия. Развиваются фреймворки для комплексной оценки, предлагающие метрики верности ответа контексту, релевантности контекста запросу и полноты ответа [10].

К практическим ограничениям относятся: зависимость качества ответов от качества исходных документов, сложность обработки мультимодального контента (таблицы, изображения, графики), а также вопросы масштабирования при работе с объёмами в миллионы документов.

7. Заключение

Retrieval-Augmented Generation представляет собой зрелый и активно развивающийся подход к расширению возможностей больших языковых моделей. За четыре года с момента публикации оригинальной работы Lewis et al. архитектура RAG эволюционировала от простой схемы «поиск + генерация» до сложных модульных и агентных систем.

Ключевые преимущества RAG – актуальность данных, прозрачность и верифицируемость ответов, умеренные вычислительные затраты – делают его предпочтительным подходом для широкого круга прикладных задач: от корпоративных систем поддержки принятия решений до специализированных ассистентов в медицине, юриспруденции и инженерии.

Перспективные направления развития включают: совершенствование мультимодальных RAG-систем, способных работать с текстом, изображениями и структурированными данными; развитие агентных архитектур с автономным планированием поисковых стратегий; а также создание надёжных методов оценки качества, учитывающих специфику конкретных предметных областей.

Список литературы:

1. Zhao W.X. et al. A Survey of Large Language Models // arXiv preprint arXiv:2303.18223. – 2023.

2. Lewis P., Perez E., Piktus A. et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks // Advances in Neural Information Processing Systems. – 2020. – Vol. 33. – P. 9459-9474.

3. Gao Y. et al. Retrieval-Augmented Generation for Large Language Models: A Survey // arXiv preprint arXiv:2312.10997. – 2023.

4. Wang L. et al. Text Embeddings by Weakly-Supervised Contrastive Pre-training // arXiv preprint arXiv:2212.03533. – 2022.

5. Ma X. et al. Fine-Tuning LLaMA for Multi-Stage Text Retrieval // arXiv preprint arXiv:2310.08319. – 2023.

6. Nogueira R., Cho K. Passage Re-ranking with BERT // arXiv preprint arXiv:1901.04085. – 2019.

7. Yao S. et al. ReAct: Synergizing Reasoning and Acting in Language Models // International Conference on Learning Representations. – 2023.

8. Ovadia O. et al. Fine-Tuning or Retrieval? Comparing Knowledge Injection in LLMs // arXiv preprint arXiv:2312.05934. – 2023.

9. Liu N.F. et al. Lost in the Middle: How Language Models Use Long Contexts // Transactions of the ACL. – 2024. – Vol. 12. – P. 157-173.

10. Es S. et al. RAGAs: Automated Evaluation of Retrieval Augmented Generation // arXiv preprint arXiv:2309.15217. – 2023.