SCIENTIFIC METHODS AND TECHNOLOGIES

UDC 004.8

Soldatkin P.I., Mezentseva E.M. Application of approaches to expanding the context of large language models in systems with high requirements for the accuracy of relational knowledge extraction

Применение подходов расширения контекста больших языковых моделей в системах с высокими требованиями к точности извлечения реляционных знаний

Soldatkin Pavel Ivanovich
Master's Student,
Samara National Research University named after Academician S.P. Korolev, Samara, Russia
Mezentseva Ekaterina Mikhailovna
Associate Professor, Associate Professor, PhD in Engineering, Samara National Research University named after Academician S.P. Korolev, Samara, Russia
Солдаткин Павел Иванович
Магистрант, Самарский национальный исследовательский университет имени академика С.П. Королева, Самара, Россия
Мезенцева Екатерина Михайловна
Доцент, доцент, к.т.н.
Самарский национальный исследовательский университет имени академика С.П. Королева, Самара, Россия

Abstract. *This paper considers the knowledge graph as a method for dynamically expanding the context of large language models (LLM) without their complete retraining. The aim of the work was to develop the architecture of a knowledge graph-based system and conduct a comparative evaluation of its effectiveness relative to hybrid search (RAG + BM25) on a real text corpus. We describe the architecture of a triplet extraction system based on two system prompts with a single closed dictionary of entity types and predicates, as well as mechanisms for entity normalization and triplet validation. Triplet extraction was performed by the GigaChat 2 Max model using fragment batch transmission and system prompt caching. The graph was stored in the Neo4j DBMS with the Cypher query language. A comparative experiment was conducted between graph search (Graph RAG) and hybrid search on a corpus of 300 fragments and 7 text questions, assessed using the RAGAS library metrics. It was shown that the integral metrics of both approaches are comparable. However, the graph approach demonstrates a consistent qualitative advantage for questions requiring multi-step reasoning along chains of relationships between named entities. When relevant context is missing, the graph clearly signals this, whereas hybrid search returns thematically related fragments that do not answer the question. The cost of constructing a graph index is approximately 85 times higher than that of vector indexing; the use of batching and system prompt caching reduces costs by 17%. It is concluded that the knowledge graph is a feasible structural complement to hybrid search in systems with high requirements for the accuracy of relational knowledge retrieval.*

Keywords: *knowledge graph, RAG, hybrid search, LLM, context extension, Neo4j, Cypher, RAGAS*

Аннотация. *В статье рассматривается граф знаний как метод динамического расширения контекста больших языковых моделей (LLM) без их полного переобучения. Целью работы являлась разработка архитектуры системы на основе графа знаний и проведение сравнительной оценки её эффективности относительно гибридного поиска (RAG+BM25) на реальном корпусе текстов. Описана архитектура системы извлечения триплетов на*

основе двух системных промптов с единым закрытым словарём типов сущностей и предикатов, а также механизмы нормализации сущностей и валидации триплетов. Извлечение триплетов осуществлялось моделью GigaChat 2 Max с применением пакетной передачи фрагментов и кэшированием системного промпта. Для хранения графа использована СУБД Neo4j с языком запросов Cypher. Проведён сравнительный эксперимент графового поиска (Graph RAG) и гибридного поиска на корпусе из 300 фрагментов и 7-ми текстовых вопросов с оценкой по метрикам библиотеки RAGAS. Показано, что интегральные метрики обоих подходов сопоставимы. Однако графовый подход демонстрирует устойчивое качественное преимущество на вопросах, требующих многоходовых рассуждений по цепочкам связей между именованными сущностями. При отсутствии релевантного контекста граф явно сигнализирует об этом, тогда как гибридный поиск возвращает тематически близкие, но не отвечающие на вопрос фрагменты. Стоимость построения графового индекса превышает стоимость векторного индексирования приблизительно в 85 раз; применение батчинга и кэширования системного промпта позволяет сократить затраты на 17%. Сделан вывод о целесообразности использования графа знаний как структурного дополнения к гибридному поиску в системах с высокими требованиями к точности извлечения реляционных знаний.

***Ключевые слова**: граф знаний, RAG, гибридный поиск, LLM, расширение контекста, Neo4j, Cypher, RAGAS*

Рецензент: Сагитов Рамиль Фаргатович - кандидат технических наук, доцент. Заместитель директора, главный научный сотрудник. ООО «Научно-исследовательский проектный институт «Промышленное и гражданское строительство»

Введение

Большие языковые модели (Large Language Models, LLM) очень часто применяются для решения прикладных задач. Несмотря на широту охватываемых знаний, подобные модели по своей природе не способны адаптироваться к узкоспециализированным предметным областям и не располагают актуальными сведениями о событиях, произошедших после даты завершения их обучения. Между тем окружающая информационная среда изменяется непрерывно – как на технологическом, так и на геополитическом уровне.

Для решения данной проблемы активно развиваются методы динамического расширения контекста модели без её полного переобучения. Среди них выделяют подход на основе поиска с дополнением (Retrieval-Augmented Generation, RAG), полнотекстовый поиск и их комбинацию – гибридный поиск. Принципиально иным механизмом расширения контекста является граф знаний (knowledge graph), который рассматривается в настоящей работе в качестве альтернативы RAG-ориентированным подходам.

Векторный поиск и подход RAG

В системе RAG поиск работает с использованием эмбеддингов, которые обеспечивают сжатое семантическое представление документа. Эмбеддинг выражается в виде вектора чисел. В процессе индексирования каждый документ разбивается на более мелкие фрагменты, которые преобразуются в эмбеддинг с помощью модели эмбеддинга. Затем исходный фрагмент и эмбеддинг индексируются в

базе данных. Если фрагменты слишком малы, на некоторые вопросы нельзя ответить, если фрагменты слишком длинны, то ответы будут содержать генерируемый шум [1].

Основные ограничения RAG заключаются в том, что он не учитывает связи между отдельными фрагментами документа, а также плохо справляется с вопросами требующих многоходовых рассуждений (Multi-hop Reasoning).

Полнотекстовый поиск

Полнотекстовый поиск или поиск, по ключевым словам, на основе алгоритма BM25 [2]. Запрос пользователя предварительно очищается от стоп-слов, после чего производится поиск по лексическому совпадению. Вместе с тем этот метод лишён семантического понимания. Также чувствителен к словоформам и опечаткам.

Гибридный поиск

Комбинирование векторного и полнотекстового поиска получило название гибридного поиска. Такой подход позволяет компенсировать недостатки каждого из методов в отдельности: семантическое сходство векторного поиска дополняется точностью лексического совпадения [3].

Тем не менее оба подхода объединяет общее структурное ограничение – отсутствие явного представления семантических связей между понятиями, которые содержатся в обрабатываемых текстах.

Граф знаний как метод расширения контекста

Граф знаний организует информацию в виде узлов (сущностей) и направленных рёбер (отношений между ними). Каждая сущность связана с соответствующими текстовыми фрагментами через их идентификаторы, хранящиеся в реляционной базе данных – это снижает избыточность и повышает гибкость [4].

Базовая единица – триплет «субъект-предикат-объект», где субъект (главная сущность с типом и атрибутами) соединён направленным предикатом с объектом. Триплеты извлекаются из фрагментов LLM по индексирующему промпту, а запросы разбираются поисковым промптом для графового поиска [5].

В отличие от векторного RAG, граф сохраняет явные семантические связи, обеспечивая более точный поиск для задач с отношениями между сущностями.

Правила формирования системных промптов

Система использует два промпта: индексирующий (извлечение триплетов из фрагментов) и поисковый (разбор запросов). Обязательное условие – идентичные словари типов сущностей и предикатов, адаптированные под конкретную предметную область (ML, юриспруденция, медицина). Несоответствие словаря приведет к отсутствию результатов поиска. Выход строго в JSON по шаблону (рис 1). Ускорение формирования графа через батчинг с сохранением chunk_id.

Рис. 1. Требуемый вывод от LLM

Словарь типов сущностей и предикатов

Для исследования использовался закрытый словарь типов сущностей и отношений, настроенный под предметную область машинного обучения, Data Science и компьютерных технологий. Словарь сущностей включал, например, MODEL, ALGORITHM, ARCHITECTURE, DATASET, FRAMEWORK, LIBRARY, PERSON, ORGANIZATION и CONCEPT. Словарь предикатов задавал направление связи и содержал такие отношения, как CREATED_BY, TRAINED_ON, BASED_ON, USES, COMPETES_WITH и PARTNERED_WITH. Такой подход обеспечивает единообразие извлечения триплетов и согласованность между индексированием и поиском [4].

Нормализация и валидация

Для повышения качества графа каждая сущность сохранялась не только в исходной форме, но и в нормализованном виде в поле normalized_name. Нормализация приводила имя к нижнему регистру, удаляла лишние пробелы и спецсимволы, что позволяло сводить разные варианты записи одной сущности к единому узлу. Такой подход уменьшает дублирование и повышает точность поиска [11].

Дополнительно выполнялась проверка извлечённых триплетов по закрытому словарю предикатов: связи, не входящие в эталонный набор, повторно проверялись на расширенном контексте. После записи в граф синонимичные отношения объединялись в единый тип, при этом сохранялся атрибут происхождения merged_from. Это позволяло поддерживать целостность структуры и отслеживать историю преобразований [6**Ошибка! Источник ссылки не найден.**].

Сохранение триплетов в Neo4j

Для хранения извлечённых триплетов использовалась графовая СУБД Neo4j, а взаимодействие с базой данных выполнялось через декларативный язык запросов Cypher [12]. В графе хранились узлы сущностей и идентификаторы фрагментов, а также рёбра, отражающие упоминания сущностей в текстовых фрагментах и семантические отношения между ними. Такой способ хранения позволяет быстро переходить от сущности или связи к исходному контексту, не дублируя текст внутри графа.

Для извлечения релевантных фрагментов использовались шаблонные запросы по сущности и по паре связанных сущностей [7]. Например, запрос по сущности находил все фрагменты, где она упоминается (рис. 2):

```
MATCH (e:Entity {normalized_name: $name})-[:MENTIONED_IN]->(c:Chunk)
RETURN c.id AS chunk_id
```

Рис. 2. Запрос сопоставления сущности по нормализованному имени с возвратом идентификатора связанных с ней текстовых фрагментов.

Для поиска по отношению между двумя сущностями использовался запрос вида (рис. 3):

```
MATCH (e1:Entity {normalized_name: $source})-[r]-(e2:Entity {normalized_name: $target})
MATCH (e1)-[:MENTIONED_IN]->(c:Chunk)
MATCH (e2)-[:MENTIONED_IN]->(c)
RETURN DISTINCT c.id AS chunk_id
```

Рис. 3. Запрос сопоставления двух сущностей из триплета с возвратом идентификатора связанных с ними текстовых фрагментов.

Такой запрос находит фрагменты, в которых совместно встречаются две связанные сущности, и особенно полезен для вопросов, ориентированных на связи между объектами предметной области.

Стоимость и время построения индексов

Для оценки вычислительных затрат были сопоставлены два способа построения индекса на выборке из 64 фрагментов, агрегированных из публичных Telegram-каналов. Построение векторного индекса с использованием модели Embeddings-2 (длина вектора 1024) потребовало 5962 токена и заняло 0,89 секунды при стоимости 0,08 рубля [13]. Построение графа знаний с использованием модели GigaChat 2 Max на том же объёме данных потребовало существенно больших ресурсов: от 27739 до 28024 реальных токенов в зависимости от режима запуска, время индексирования составило 94,03 секунды при использовании кэша и 96,24 секунды без него. При тарифе 0,65 рубля за тысячу токенов стоимость построения графа превышает стоимость векторного индексирования в 85 раз. Разница в числе тарифицируемых токенов между моделью создания векторов и LLM-моделью с кэшем составила 4471 токен (табл. 1), что обусловлено структурной разметкой фрагментов в индексирующем промпте.

Таблица 1

Сравнительная таблица по затраченным токенам для двух подходов

Метод	Токены (всего)	Токены (тарифици-руемые)	Стоимость, ₽	Время, с
Embeddings	5 962	5 962	0,08	0,89
Graph (без кэша)	28 024	12 879	8,37	96,24
Graph (с кэшем)	27 739	10 433	6,78	94,03

В рамках исследования были сопоставлены три стратегии построения графа (рис. 4): без батчинга и кэша системного промпта; с батчингом по 8 фрагментов без явного кэша; с батчингом по 8 фрагментов и явным управлением кэшем через идентификатор сессии [14]. Первая стратегия демонстрировала значительный разброс в расходе токенов, обусловленный стохастическим поведением кэша на стороне провайдера. На 40-ом шаге видно самоочистку, которая приводила к резкому скачку затрат. Вторая стратегия обеспечила более стабильный расход.

Комбинация батчинга и явного управления кэшем оказалась наиболее экономичной и стабильной: при 50 фрагментах экономия по токенам относительно базового подхода составила около 17%, а время обработки сократилось примерно на 20%.

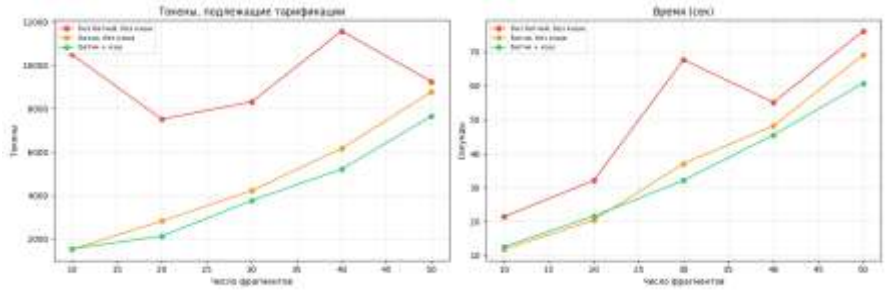

Рис. 4 – Три стратегии построения графа знаний

Сравнение качества ответов

Для оценки качества поиска использовалась библиотека RAGAS [15]. Граф знаний строился на корпусе из 300 фрагментов с использованием той же модели GigaChat 2 Max; для гибридного поиска применялась комбинация pgvector [8] (косинусное сходство, порог 0,7, топ-5) и полнотекстового поиска BM25 через механизм websearch_to_tsquery [9]. Оба метода работали на одной и той же выборке из 300 фрагментов. Для оценки было сформировано 7 вопросов (табл. 3).

В качестве поиска связей внутри графа использовались 5 типов запросов к Neo4j в зависимости от полноты триплета, возвращаемого поисковым промптом: 1) поиск фрагмента по одной сущности; 2) поиск фрагмента по двум сущностям; 3) поиск фрагмента по сущности и предикату в прямом направлении; 4) поиск фрагмента по сущности и предикату в обратном направлении; 5) поиск фрагмента по полному триплету [7]. Приоритет фрагментов после запросов: 5 > (4,3) > 2 > 1.

Различия между подходами по обеим метрикам незначительны (табл. 2), однако качественный анализ ответов выявляет характерные паттерны превосходства каждого из методов.

Таблица 2

Сравнительная таблица двух подходов по встроенным метрикам RAGAS

Метрика	Graph RAG	Гибридный поиск	Δ
Faithfulness (Достоверность)	0,908	0,905	0,003
Answer Relevancy (Релевантность ответа)	0,674	0,685	−0,011

Графовый поиск демонстрирует преимущество на вопросах, предполагающих явные связи между сущностями. На вопрос «С кем конкурирует DeepSeek?» граф возвращает точный контекст через ребро COMPETES_WITH и даёт однозначный ответ. На вопрос о партнёрах и создателях Palantir граф извлекает полный набор связей (CREATED_BY, PARTNERED_WITH). На вопрос об Agent Payments Protocol граф точно перечисляет всех партнёров и совместимые фреймворки через рёбра PARTNERED_WITH и COMPATIBLE_WITH. Показательно и поведение при отсутствии релевантного контекста: на вопрос о влиянии ИИ на занятость граф честно сообщает об отсутствии информации – в отличие от гибридного поиска, который возвращает тематически близкие, но не отвечающие на вопрос фрагменты.

Гибридный поиск превосходит графовый на вопросах широкого семантического охвата, не привязанных к конкретным именованным сущностям. На вопрос о влиянии генеративного ИИ на занятость гибридный поиск находит релевантный фрагмент с конкретными данными исследования (снижение доли джуниоров на 7,7%), тогда как граф не находит ничего – поскольку в графе отсутствует соответствующая связь между сущностями. На вопрос о связи математики с гуманитарными науками оба метода находят одинаковый базовый фрагмент, однако гибридный поиск дополнительно подтягивает контекст через BM25.

Отдельного внимания заслуживает вопрос о сотрудниках IBM: ни один из методов не даёт корректного ответа о публикациях, поскольку соответствующей информации нет в корпусе. Граф возвращает только связи WORKS_FOR, гибридный поиск – нерелевантные фрагменты. Это наглядно иллюстрирует общее ограничение обоих подходов: качество ответа не может превышать качество исходного корпуса.

Таблица 3

Сравнительная таблица вопросов и ответов для двух подходов

Вопрос	Graph RAG (Ответ)	Гибридный поиск (RAG+BM25) (Ответ)
С кем конкурирует DeepSeek?	DeepSeek конкурирует с ChatGPT.	Из предоставленного контекста нельзя однозначно определить...
Как генеративный ИИ влияет на занятость сотрудников?	В предоставленном контексте отсутствует информация...	...оказывает негативное влияние на занятость младших сотрудников... снижение на 7.7% спустя шесть кварталов.
Как математика связана с гуманитарными науками?	Развернутый ответ со списком связей (культура, дисциплина мышления, мост между науками).	Развернутый ответ с цитированием аргументов о психологическом развитии и семантике.
С кем сотрудничает Palantir и кто его создал?	Основан Александром Карпом и Питером Тилем... Сотрудничала с ЦРУ.	Создана Карпом и Тилем... Сотрудничает с ЦРУ... (примечание про Mastercard относится к другому проекту).
Кто из сотрудников IBM публиковал исследования и в каких изданиях?	Нет информации о публикациях.. (найдены только сотрудники).	Нет ответа в представленном контексте. (найдены только сотрудники).
Какие платежные системы партнёрятся с Agent Payments Protocol и с какими фреймворками он совместим?	Списки: Партнёры (Mastercard, PayPal), Фреймворки (MCP, A2A).	Списки: Партнёры (Mastercard, PayPal, Intuit, Salesforce), Фреймворки (MCP, A2A).
Какое китайское оборудование связано с передачей данных, и кто из известных личностей выступает против Китая?	Оборудование: Unitree G1... Личности: Замиска, Карп.	Оборудование: Unitree G1... Информация о личностях отсутствует.

Заключение

В настоящей работе граф знаний рассмотрен как способ динамического расширения контекста больших языковых моделей. Показано, что качество графовой базы определяется согласованностью словарей, нормализацией сущностей и валидацией извлечённых триплетов. Сравнительный эксперимент на корпусе из 300 фрагментов с использованием RAGAS показал, что графовый и гибридный поиск демонстрируют сопоставимые результаты по метрикам достоверности и релевантности, тогда как качественные различия проявляются в типе запросов: граф лучше справляется с вопросами на связи между сущностями, а гибридный поиск – с более общими семантическими запросами. При этом построение графа требует существенно больших вычислительных и токенных затрат, хотя батчинг и управление кэшем частично их снижают. В результате граф знаний целесообразно рассматривать не как замену

гибридному поиску, а как его структурное дополнение для предметных областей, где критичны отношения между сущностями.

References

1. Barnett S., Kurniawan S., Thudumu S., Brannelly Z., Abdelrazek M. Seven Failure Points When Engineering a Retrieval Augmented Generation System // arXiv preprint arXiv:2401.05856v1 [cs.SE]. – 2024. – URL: https://arxiv.org/html/2401.05856v1 (дата обращения: 18.03.2026).
2. Lù X.H. BM25S: Orders of magnitude faster lexical search via eager sparse scoring // arXiv preprint arXiv:2407.03618. – 2024. – URL: https://arxiv.org/pdf/2407.03618 (дата обращения: 18.03.2026).
3. Sultania D., Lu Z., Naik T. et al. Domain-specific Question Answering with Hybrid Search // arXiv preprint arXiv:2412.03736v2. – 2024. – URL: https://arxiv.org/html/2412.03736v2 (дата обращения: 19.03.2026).
4. Hogan A., Blomqvist E., Cochez M. et al. Knowledge Graphs // arXiv preprint arXiv:2003.02320. – 2021. – URL: https://arxiv.org/pdf/2003.02320 (дата обращения: 15.03.2026).
5. Wesslund D., Stenström V., Linde P., Holmberg A. LLM-based Triplet Extraction from Financial Reports // arXiv preprint arXiv:2602.11886v1. – 2026. – URL: https://arxiv.org/html/2602.11886v1 (дата обращения: 23.03.2026).
6. Richardeau G., Chali S., Le Merrer E., Penzo C., Tredan G. LLMs Prompted for Graphs: Hallucinations and Generative Capabilities // arXiv preprint arXiv:2409.00159v3. – 2025. – URL: https://arxiv.org/html/2409.00159v3 (дата обращения: 23.03.2026).
7. Tang L., Dou W., Zheng Y. et al. Proving Cypher Query Equivalence // arXiv preprint arXiv:2504.15742. – 2025. – URL: https://arxiv.org/pdf/2504.15742 (дата обращения: 24.03.2026).
8. pgvector: Open-source vector similarity search for Postgres [Электронный ресурс] // GitHub. – URL: https://github.com/pgvector/pgvector (дата обращения: 24.03.2026).
9. PostgreSQL Documentation: Parsing Queries [Электронный ресурс] // PostgreSQL. – URL: https://www.postgresql.org/docs/current/textsearch-controls.html (дата обращения: 24.03.2026).
10. Pan H., Zhang Q., Adamu M., Dragut E.C., Latecki L.J. Taxonomy-Driven Knowledge Graph Construction for Domain-Specific Scientific Applications // Findings of the

Association for Computational Linguistics: ACL 2025. – 2025. – URL: https://aclanthology.org/2025.findings-acl.223.pdf (дата обращения: 10.03.2026).

11. Zhang J. Entity Normalization in Knowledge Graphs: Solving the Fragmentation Problem [Электронный ресурс] // LinkedIn. – 2026. – URL: https://www.linkedin.com/posts/jzhang-ai_how-knowledge-graphs-are-really-built-8-activity-7420133178490662912-SuCz (дата обращения: 15.03.2026).

12. Cypher Manual v5.15: Introduction to Cypher [Электронный ресурс] // Neo4j Documentation. – URL: https://neo4j.com/docs/cypher-manual/current/introduction/cypher-neo4j/ (дата обращения: 18.03.2026).

13. Тарифы GigaChat [Электронный ресурс] // СберДевелоперс. – URL: https://developers.sber.ru/docs/ru/gigachat/tariffs/legal-tariffs (дата обращения: 20.03.2026).

14. Кеширование запросов [Электронный ресурс] // СберДевелоперс. – URL: https://developers.sber.ru/docs/ru/gigachat/guides/keeping-context (дата обращения: 25.03.2026).

15. RAGAS: Retrieval Augmented Generation Assessment [Электронный ресурс] // RAGAS Documentation. – URL: https://docs.ragas.io/en/stable/ (дата обращения: 25.03.2026).