

РЕАЛИЗАЦИЯ GRAPHREADER-АГЕНТА ДЛЯ УЛУЧШЕНИЯ ТЕХНОЛОГИИ RAG С ИСПОЛЬЗОВАНИЕМ NEO4J, LANGCHAIN И LANGGRAPH

Кулевич А. О.

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Батура М. П. – д-р техн. наук, профессор

В статье описывается реализация и использование GraphReader-агента, который решает проблемы стандартных методов RAG, такие как извлечение нерелевантных данных и потеря контекста. Используя графовую структуру данных, GraphReader улучшает точность поиска и обеспечивает более глубокий контекст для сложных задач. Описание включает процесс разработки, внедрения и тестирования агента, который использует Neo4j для хранения данных и LangChain с LangGraph для логики работы.

Современные модели машинного обучения, такие как LLM (**Large Language Models**), могут обрабатывать большие объемы информации и генерировать осмысленные ответы, но качество поиска и актуальность данных остаются ключевыми проблемами. Технология **Retrieval-Augmented Generation (RAG)** [1] решает эту проблему, сочетая генерацию ответов с извлечением информации из внешних источников. Однако стандартные методы RAG могут извлекать нерелевантные данные, терять контекст и не учитывать семантические связи.

GraphReader-агент [2] решает эти проблемы, используя графовую структуру для представления данных. Графовые базы данных, такие как Neo4j, устанавливают связи между объектами, создавая более точную модель знаний с семантическими путями. GraphReader комбинирует графовые базы с языковыми моделями, что позволяет точно извлекать информацию и учитывать взаимосвязи между данными, обеспечивая более глубокий контекст для сложных задач. Этот граф отличается от классических сетевых и ближе к документному или лексическому.

В данной работе GraphReader агент реализован, используя Neo4j для хранения и структурирования данных и LangChain в сочетании с LangGraph для определения агента и его логики. Для демонстрации работы агента использовались статьи с сайта Medium [3].

Для начала данные были загружены в графовую БД Neo4j. Они представляются в виде документов, которые разбиваются на более мелкие фрагменты — *chunks*. Эти фрагменты затем анализируются для выделения атомарных фактов (*atomic facts*), которые являются ключевыми единицами знаний. Для улучшения поиска и контекстуального понимания выделяются ключевые элементы (*keyelements*), представляющие наиболее значимые концепции. Общее представление полученной базы данных представлено на рисунке 1.

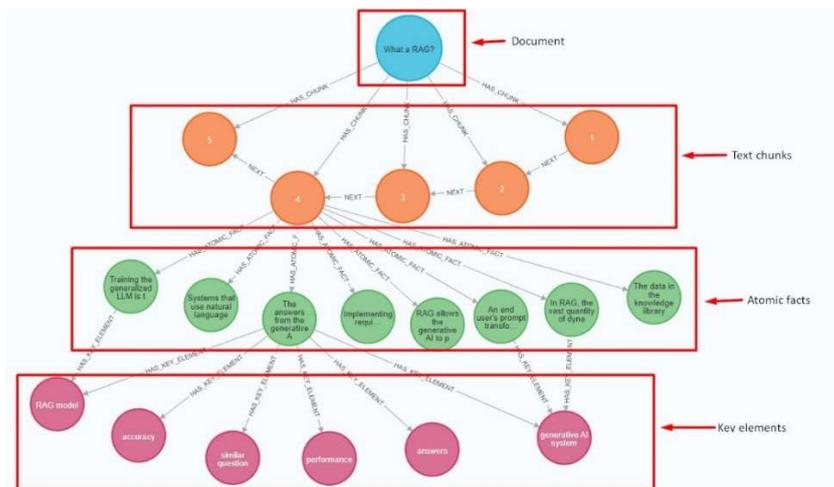


Рисунок 1 – Общее представление полученной графовой базы данных

Вместо линейной структуры хранения данных (как в традиционных RAG-подходах) создается семантический граф, в котором *chunks*, *atomic facts* и *keyelements* соединяются между собой. Это позволяет LLM находить взаимосвязи между частями информации. При запросе пользователя GraphReader-агент выполняет графовый поиск, анализируя связи между *keyelements*, документами и фактами, чтобы извлекать наиболее релевантные данные. Затем полученные результаты передаются модели, которая использует их для генерации более точного и контекстуально обоснованного ответа. Структура работы полученного агента представлена на рисунке 2.



Рисунок 2 – Структура работы GraphReader-агента

На рисунке 3 представлены результаты работы полученного GraphReader-агента в ответ на запрос, выполненный с использованием LangGraph. Запрос был сформулирован следующим образом: "What is RAG?". Процесс поиска и ответа можно разделить на несколько этапов, каждый из которых представлен в логах выполнения.

```

langgraph.invoke({"question": "What is RAG?"})
-----
Step: rational_plan
Rational plan: To answer the question "What is RAG?", we need to identify the definition and context in which RAG is used. This involves:

1. Determining if RAG is an acronym and what it stands for.
2. Understanding the context or field in which RAG is applied (e.g., technology, business, etc.).
3. Gathering any relevant details or examples that illustrate the use or significance of RAG.

Let's proceed with these steps to provide a comprehensive answer.
-----
Step: atomic_fact_check
Reading atomic facts about: ['Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks', 'Retrieval-Augmented Generation', 'unstructured data', 'enterprise data sources', 'advanced cloud-based AI services']
Rational for next action after atomic check: The atomic facts provide a good overview of what RAG is, its purpose, and its challenges. However, to ensure a comprehensive understanding, it would be beneficial to read the text chunks associated with these atomic facts. This will provide more detailed information and context about RAG, its applications, and its significance in the field of generative AI.
Chosen action: {'function_name': 'read_chunk', 'arguments': [['19734dd10ceb44f0d8863c8a9e6e6227', 'e8be25e339967d367b3b8eb6e61d36ab', 'd5a06ae41d630c30ab5c9040ecf5f0e3', '1c0b928f63172baccb5ed9608ad983e5', 'e3ec3ce5bbf478375edd25deaa992462']]}
-----
Step: read_chunk(e3ec3ce5bbf478375edd25deaa992462)
Rational for next action after reading chunks: The current chunk provides additional details about RAG, such as the types of data it can incorporate and its ability to cite references. This information complements the existing notes and provides a comprehensive understanding of RAG. Therefore, the information gathered is sufficient to answer the question comprehensively.
Chosen action: {'function_name': 'termination', 'arguments': []}
-----
Step: Answer Reasoning
{'answer': 'RAG stands for Retrieval-Augmented Generation, a technique in generative AI that enhances the output of a large language model by integrating specific information without altering the model itself, introduced in a 2020 paper by Patrick Lewis and others at Facebook AI Research.',
 'analysis': 'The notebook provides a comprehensive explanation of what RAG is. It stands for Retrieval-Augmented Generation, a technique in generative AI that enhances the output of a large language model by integrating specific information without altering the model itself. This method was introduced in a 2020 paper by Patrick Lewis and others at Facebook AI Research. RAG is designed to deliver timely, accurate, and contextually relevant responses by embedding enterprise data into a knowledge repository, converting it into vectors, and storing it in a vector database for retrieval. This approach can yield more precise results compared to using a generative LLM alone. However, it faces challenges in modeling both structured and unstructured data within the knowledge library and vector database. RAG can draw from various data sources, including relational databases, document repositories, internet streams, and more, and it can cite references for the data it retrieves, enabling error correction if necessary.'}
    
```

Рисунок 3 – Результаты выполнения запросов к полученному GraphReader-агенту

На первом шаге агент формирует рациональный план для ответа, определяя, что нужно понять, что такое RAG, что это за аббревиатура и в каком контексте она используется. Далее агент проверяет атомарные факты, связанные с RAG, такие как "Retrieval-Augmented Generation", "unstructured data", "enterprise data sources" и другие. Эти факты дают общее представление о технологии, но для глубокого понимания требуется дополнительное чтение текстовых фрагментов.

Агент выбирает несколько фрагментов данных для более детального анализа, извлекая дополнительные сведения и контекст для точного ответа. В конце агент формулирует ответ, объясняя, что RAG (Retrieval-Augmented Generation) — это технология, улучшая работу языковой модели, интегрируя информацию из внешних источников, без изменения самой модели.

В ходе работы был разработан GraphReader-агент, использующий Neo4j, что значительно улучшило качество Retrieval-Augmented Generation (RAG). Учет связей между сущностями повысил релевантность данных, а структурированное представление знаний обеспечило более точные ответы. Такой подход оптимизирует поиск и делает систему более гибкой, приближая ее к документному графу.

Список использованных источников:

1. *What is Retrieval-Augmented Generation?* [Electronic resource] // Amazon Web Services. – Mode of access: https://aws.amazon.com/what-is/retrieval-augmented-generation/?nc1=h_ls. – Date of access: 30.03.2025.
2. *GraphReader: Building Graph-based Agent to Enhance Long-Context Abilities of Large Language Models* [Electronic resource] // arXiv. – Mode of access: <https://arxiv.org/pdf/2406.14550v1>. – Date of access: 30.03.2025.
3. *Medium* [Electronic resource]. – Mode of access: <https://medium.com>. – Date of access: 30.03.2025.