Harnessing GraphRAG with LLM: Comparative Analysis and Applications in Statistical Programming

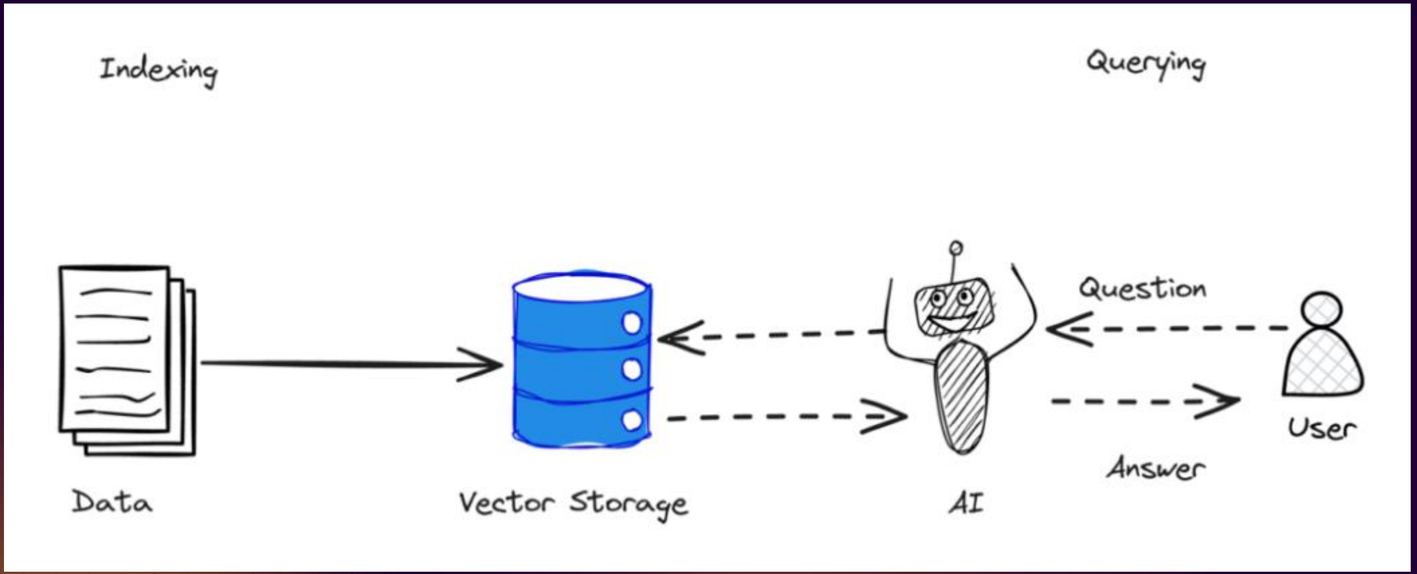
PhUSE SDE Beijing, Yi (Eason) Yang, Hengrui

Disclaimer

This presentation is intended for informational purposes only. The content herein does not constitute professional advice, nor does it represent an official position of the organization. Any views or opinions expressed are those of the presenter and do not necessarily reflect those of the company. Certain statements may be forward-looking and are subject to risks and uncertainties. No part of this presentation may be reproduced or distributed without prior written permission.

Retrieval Augmented Generation (RAG)

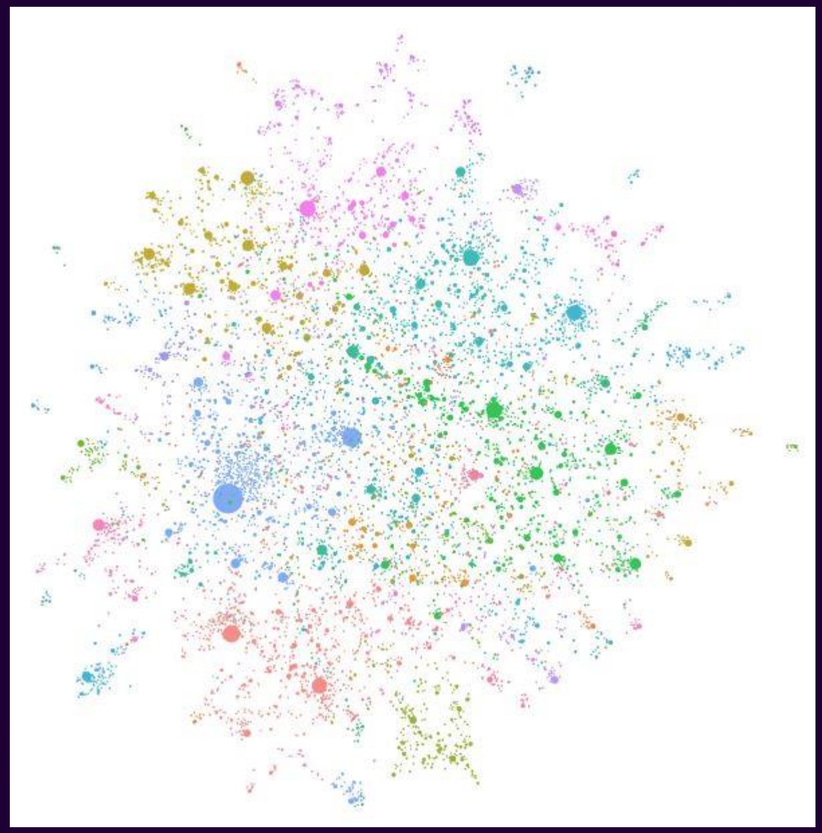
Retrieval-Augmented Generation (RAG) is a technique that enhances the accuracy and reliability of generative AI models by incorporating information retrieved from specific and relevant data sources. It essentially bridges the gap between what a large language model (LLM) knows and what it needs to know, making it more accurate and reliable.



GraphRAG

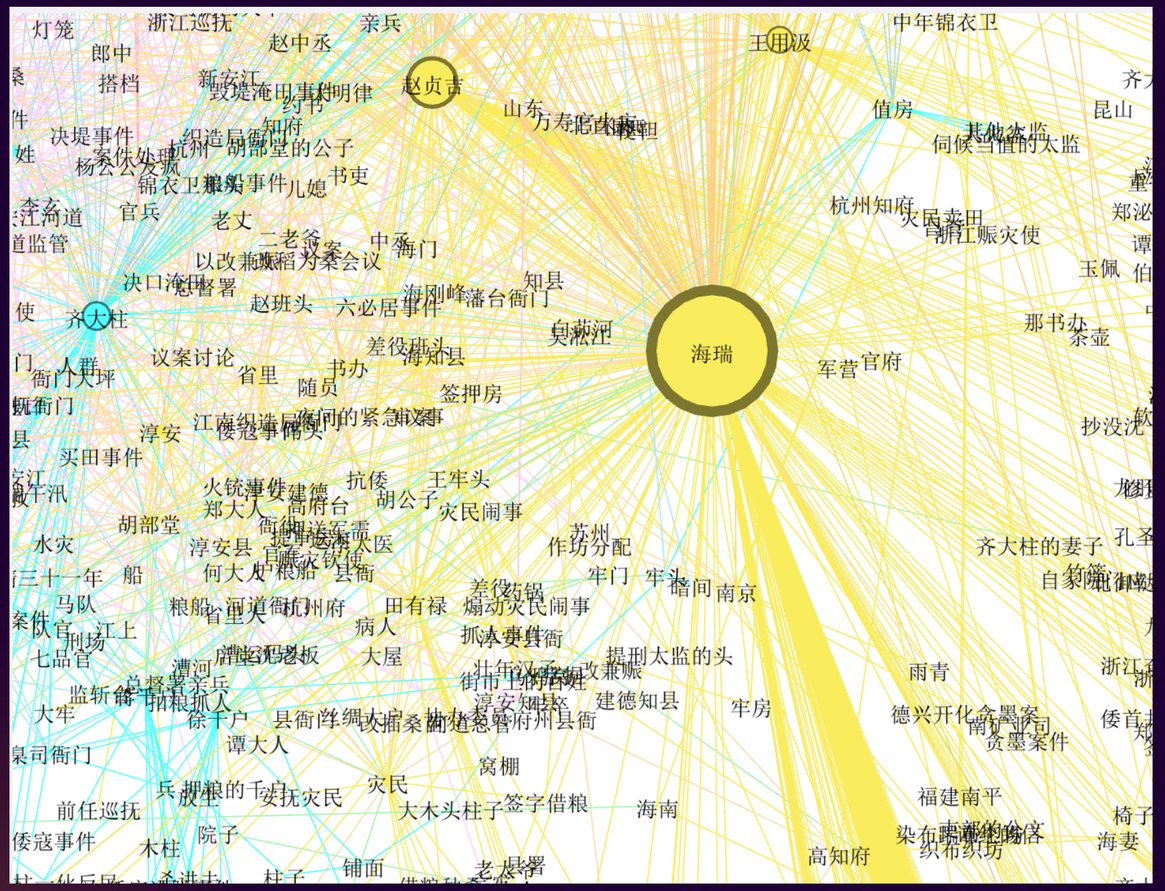
GraphRAG is a structured, hierarchical approach to **Retrieval Augmented Generation (RAG)**, as opposed to naive semantic-search approaches using plain text snippets.

The GraphRAG process involves extracting a **knowledge graph** out of raw text, building a community hierarchy, generating summaries for these communities, and then leveraging these structures when perform RAG-based tasks.

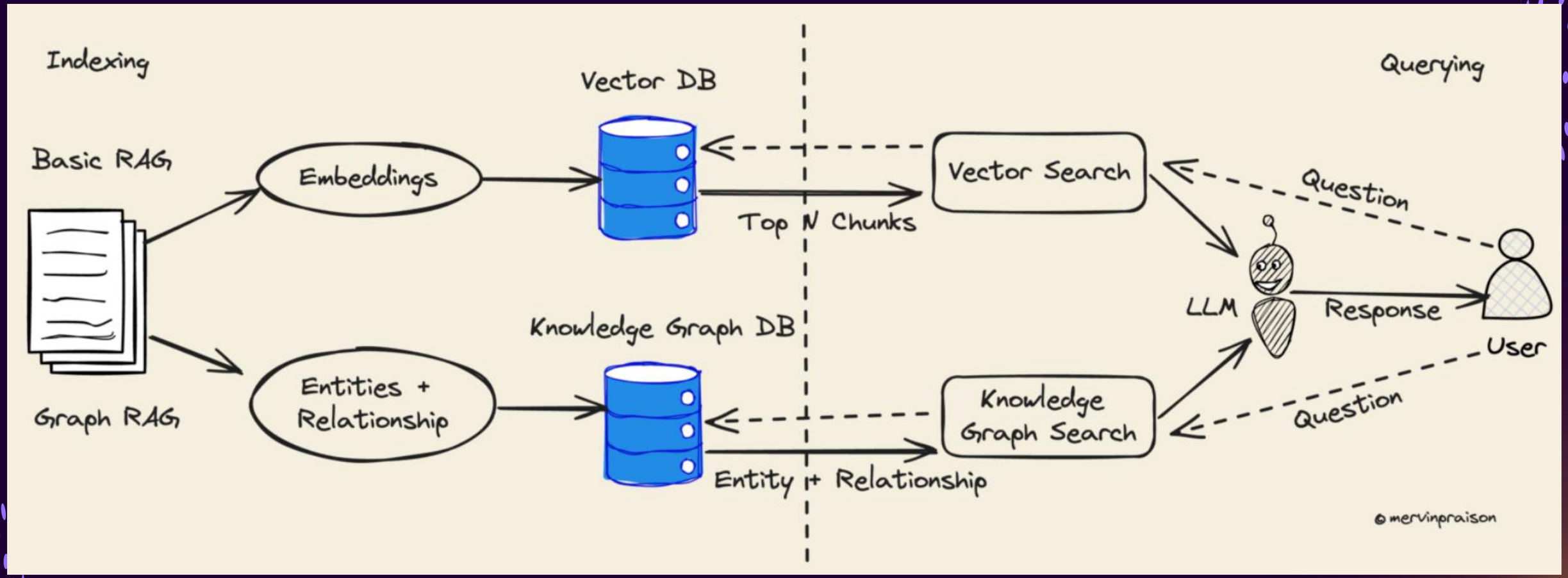


Knowledge Graph

A knowledge graph is a structured representation of data that emphasizes relationships between entities (like objects, concepts, or events)



GraphRAG vs Basic RAG



Demo

Demo

	GraphRAG Response	RAG Response
Pros	- Comprehensive statistical methods (Kaplan-Meier, log-rank, Brookmeyer-Crowley) - Accurate 90.4% power - Clear, focused structure - Intent-to-treat principles	- Includes mRECIST v1.1, 6-week assessments- Clear PFS definition- Regulatory context - Strong conclusion
Cons	- Omits mRECIST v1.1, 6-week assessments- Vague PFS definition- Event numbers lack context	- Incorrect 80% power- Omits key statistical methods - Includes irrelevant details- Less statistical detail
Overall Assessment	- Superior statistical rigor, accuracy, and focus for technical audience. - Best for technical focus. - Improve with mRECIST and PFS definition.	- Strong but less robust: Accurate endpoint details but weaker stats and inaccuracies. - Good for clinical context but needs statistical depth.

Commands

Getting Started

Install GraphRAG

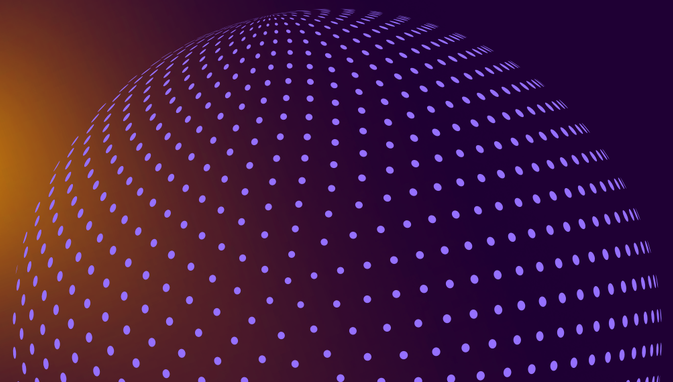
pip install graphrag -i https://pypi.tuna.tsinghua.edu.cn/simple

Set Up Your Workspace Variables

graphrag init --root ./ragtest

Running the Indexing pipeline

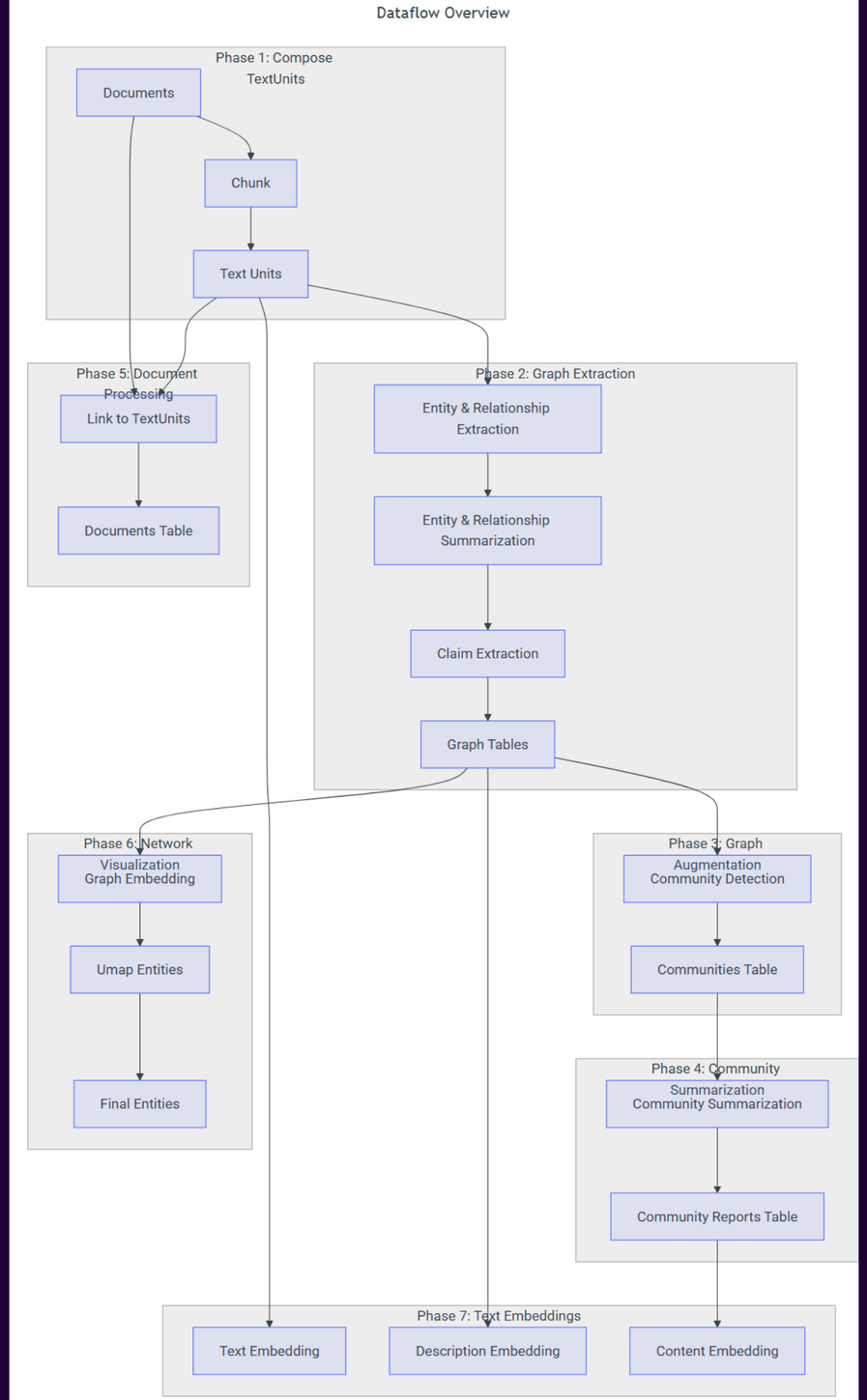
graphrag index --root ./ragtest

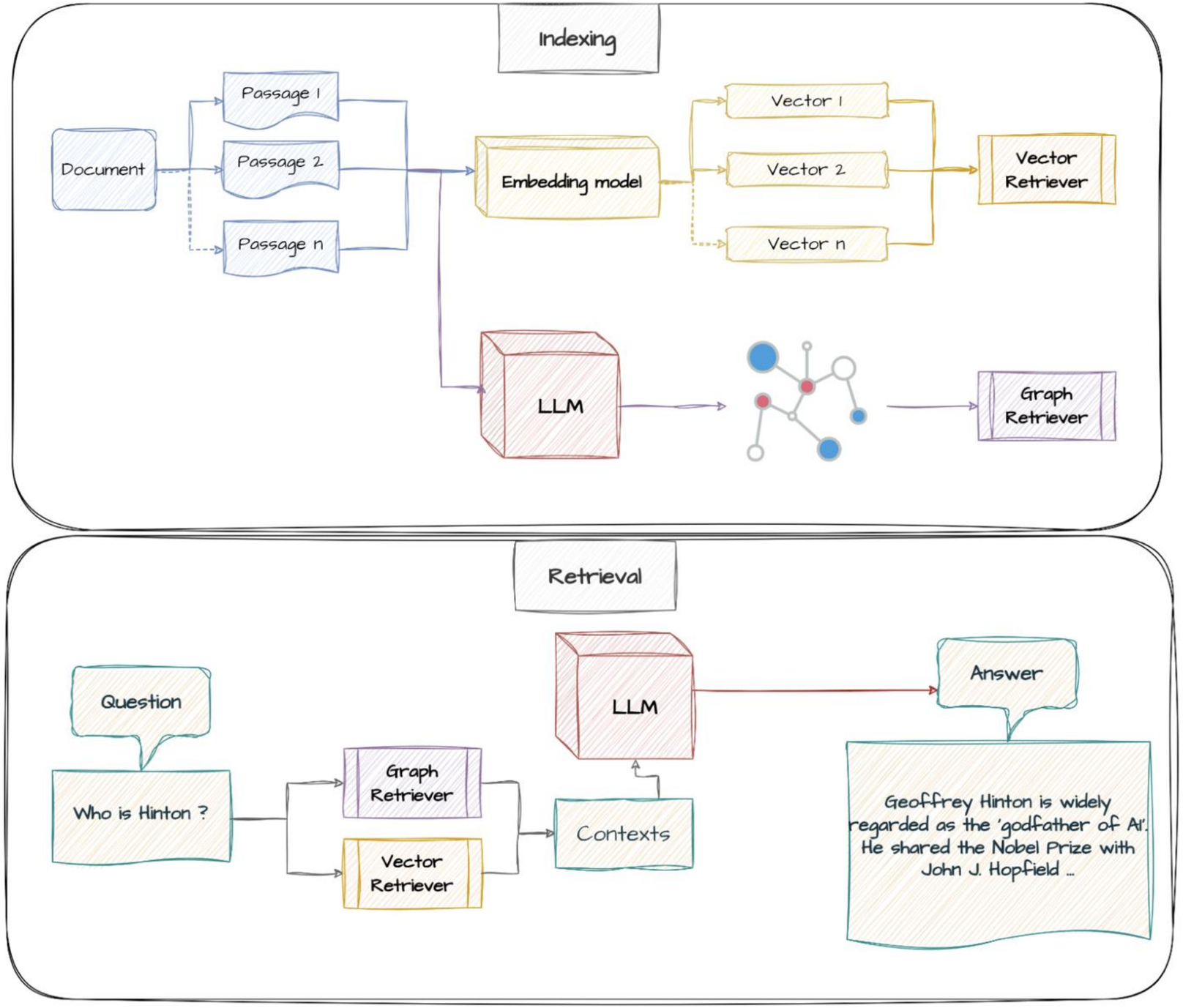


Indexing

GraphRAG Indexer

```
├── Loading Input (InputFileType.text)      # Reads the input text file
├── create_base_text_units                  # Splits text into smaller units for entity extraction and context tracking
├── create_final_documents                  # Organizes text units into structured documents for consistent input
├── extract_graph                           # Extracts entities and relationships to build a basic knowledge graph
├── compute_communities                     # Groups related entities into communities for efficient querying
├── create_final_entities                   # Optimizes and organizes entities for the knowledge graph
├── create_final_relationships              # Optimizes relationships for consistency in the graph
├── create_final_nodes                      # Creates graph nodes from entities, supporting graph traversal and queries
├── create_final_communities                # Organizes community data for better search efficiency
├── create_final_text_units                 # Integrates entities, relationships, and communities with text units
├── create_final_community_reports          # Generates summary reports for each community
└── generate_text_embeddings                # Creates vector embeddings for text and entities to improve search accuracy.
```


Dataflow Overview
Phase 1: Compose TextUnits
Documents
Chunk
Text Units
Phase 5: Document Processing
Link to TextUnits
Documents Table
Phase 2: Graph Extraction
Entity & Relationship Extraction
Entity & Relationship Summarization
Claim Extraction
Graph Tables
Phase 6: Network Visualization
Graph Embedding
Umap Entities
Final Entities
Phase 3: Graph Augmentation
Community Detection
Communities Table
Phase 4: Community Summarization
Community Summarization
Community Reports Table
Phase 7: Text Embeddings
Text Embedding
Description Embedding
Content Embedding


Indexing
Document
Passage 1
Passage 2
Passage n
Embedding model
Vector 1
Vector 2
Vector n
Vector Retriever
LLM
Graph Retriever
Retrieval
Question
Who is Hinton ?
Graph Retriever
Vector Retriever
Contexts
LLM
Answer
Geoffrey Hinton is widely regarded as the 'godfather of AI'. He shared the Nobel Prize with John J. Hopfield ...

Dataflow

Entity

An entity extracted from a text unit. These represent people, places, events, or some other entity-model that you provide.

Relationship

A relationship between two entities.

Community

Once the graph of entities and relationships is built, we perform hierarchical community detection on them to create a clustering structure.

Read Parquet file

Use Pandas to read a Parquet file and view its contents

Parquet is a columnar storage file format commonly used in big data processing and analysis

create_final_communities.parquet

create_final_community_reports.parquet

create_final_documents.parquet

create_final_entities.parquet

create_final_nodes.parquet

create_final_relationships.parquet

create_final_text_units.parquet

Query

Auto prompt-tuning (optional)

graphrag prompt-tune --root ./ragtest --config ./ragtest/settings.yaml --discover-entity-types --output ./ragtest/prompts

Loading Input (InputFileType.text).
INFO: Generating domain...`
INFO: Generated domain:
INFO: Detecting language...
INFO: Generating persona...
INFO: Generating community report ranking description...
INFO: Generating entity types...
INFO: Generating entity relationship examples...
INFO: Generating entity extraction prompt...
INFO: Generating entity summarization prompt...
INFO: Generating community reporter role...
INFO: Generating community summarization prompt...
INFO: Writing prompts to

Query

Global Search
(questions that require an understanding of the dataset as a whole)

graphrag query --root ./ragtest --method global –query " What is the top theme of this document?"

Local Search
(questions that require an understanding of specific entities mentioned in the documents)

graphrag query --root ./ragtest --method local –query "What is CDISC and what is its usage in submissions?"

Configuration

using YAML

```
llm:
    api_key: ${GRAPHRAG_CHAT_API_KEY}                    # Call the LLM API Key
    model: ${GRAPHRAG_CHAT_MODEL}
    model_supports_json: true                            # It is recommended to enable if the model supports JSON
    api_base: ${GRAPHRAG_CHAT_API_BASE}
embeddings:
    async_mode: threaded
    vector_store:
    type: lancedb
llm:
    api_key: ${GRAPHRAG_EMBEDDING_API_KEY}               # Call the embedding model API Key
    model: ${GRAPHRAG_EMBEDDING_MODEL}
    api_base: ${GRAPHRAG_EMBEDDING_API_BASE}
snapshots:
    graphml: true                                        # Set to true or false to determine whether to enable the graphml snapshot
```

Configuration

using Env Vars

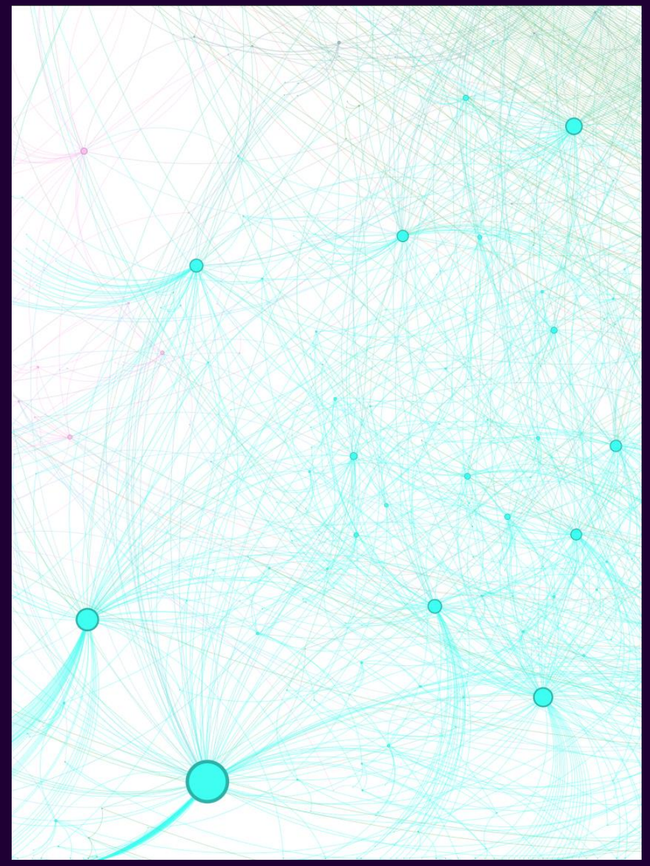
```
# Embedding Model
GRAPHRAG_EMBEDDING_API_BASE    = http://localhost:11434/v1
GRAPHRAG_EMBEDDING_API_KEY     = ollama
GRAPHRAG_EMBEDDING_MODEL       = bge-m3:latest

# Chat Model
GRAPHRAG_CHAT_API_BASE         = https://dashscope.aliyuncs.com/compatible-mode/v1
GRAPHRAG_CHAT_API_KEY          = XXXXXXXXXXXXXXXXXXXXXXXX
GRAPHRAG_CHAT_MODEL            = qwen-max-latest
```

Visualization

Visualizing the Graph

Open the Graph in Gephi

https://microsoft.github.io/graphrag/visualization_guide/



Summary

Advantages

Better Interpretability

Uses knowledge graphs, making answers easier to verify

Reduced Hallucinations

Minimizes inaccurate information, improving reliability

Higher Context Relevance

Handles complex queries better, improving answer quality

Evidence Sources

Provides sources for generated information, enhancing trustworthiness

Challenges

Complex Construction

Building knowledge graphs is resource-heavy

High Computational Cost

Requires significant computing power.

Scalability Issues

Large graphs are harder to manage efficiently

Costly Updates

Regular updates are expensive, especially in fast-changing fields

Thank You