Graph Retrieval Augmented Generation

Yogesh Haribhau Kulkarni

YHK

About Me

YHK

Yogesh Haribhau Kulkarni

Bio:

- 20+ years in CAD/Engineering software development
- Got Bachelors, Masters and Doctoral degrees in Mechanical Engineering (specialization: Geometric Modeling Algorithms).
- Currently doing Coaching in fields such as Data Science, Artificial Intelligence Machine-Deep Learning (ML/DL) and Natural Language Processing (NLP).
- Feel free to follow me at:
  - Github (github.com/yogeshhk)
  - LinkedIn (www.linkedin.com/in/yogeshkulkarni/)
  - Medium (yogeshharibhaukulkarni.medium.com)
  - Send email to yogeshkulkarni at yahoo dot com

Office Hours: Saturdays, 2 to 5pm (IST); Free-Open to all; email for appointment.

YHK

Introduction to Graph RAG

YHK

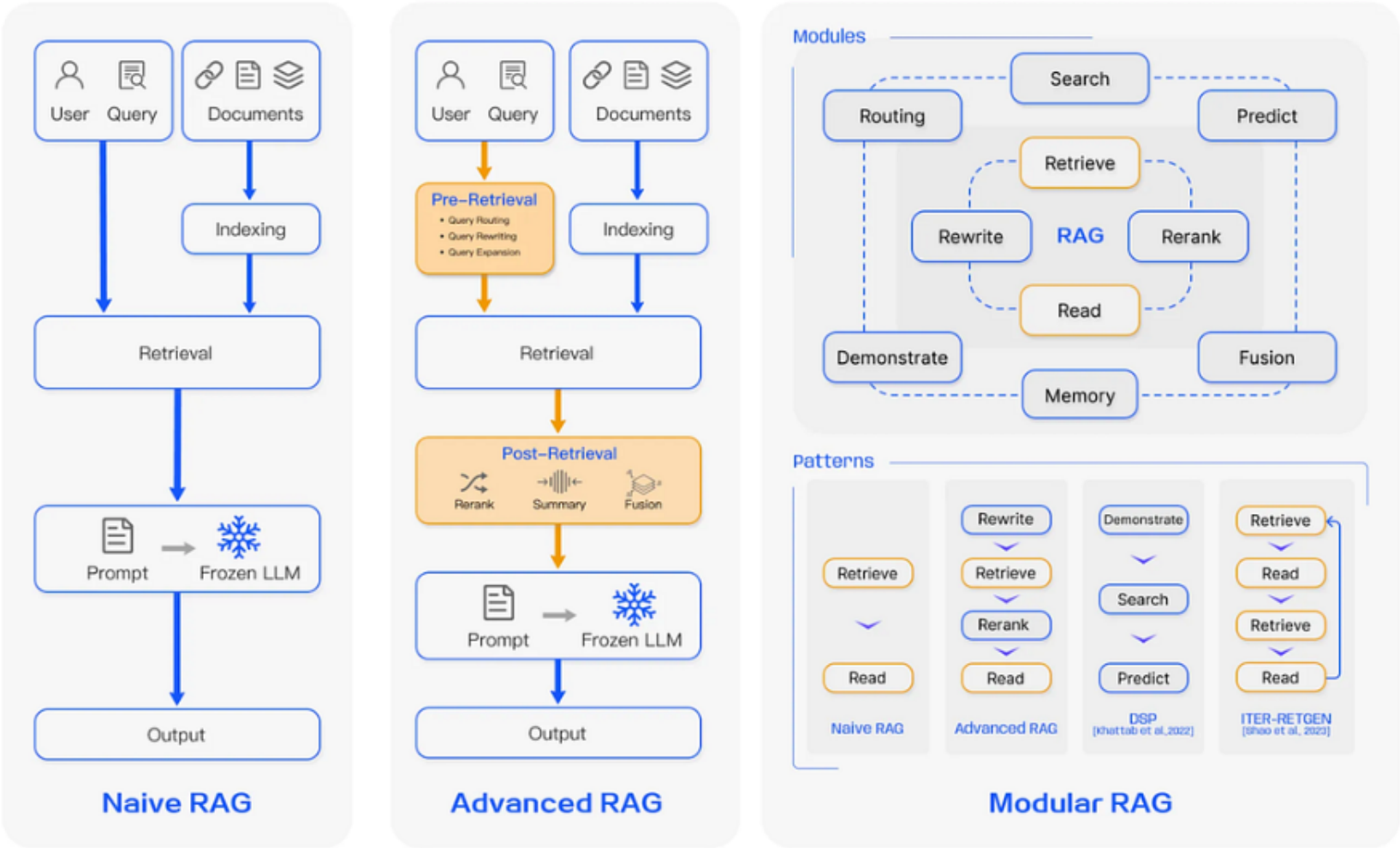
Challenges with LLMs

- Learns random sentences from random people
- Talks like a person but doesn't really understand what it's saying
- Occasionally speaks absolute non sense
- Sensitive to question phrasing
- Limited to public "knowledge"

(Ref: The GenAI Stack - Andreas Kollegger - Neo4j)

YHK

Thematic RAG Classification



(Ref: https://graphrag.com/concepts/intro-to-graphrag/))

The phases of an advanced RAG system

- Pre-retrieval-Query rewriting, query entity extraction, query expansion, etc.
- Retrieval of relevant context
- Post-retrieval: Reranking, pruning, etc.
- Answer generation

(Ref: https://graphrag.com/concepts/intro-to-graphrag/))

YHK

Why Graph RAG?

- Language models struggle with factual accuracy and real-world knowledge.
- Retrieval-Augmented Generation (RAG) improves accuracy using external text data.
- Traditional RAG has limitations in context understanding and scalability.
- GraphRAG leverages knowledge graphs for better retrieval and response generation.

YHK

Limitations of Traditional RAG

- **Flat Retrieval:** Documents are treated as isolated entities.
- **Contextual Shortcomings:** Lacks deep semantic understanding.
- **Scalability Issues:** Slower retrieval with increasing data volume.
- Struggles with scalability due to flat data representation.
- Lacks global context over the entire data corpus.
- Inefficient for complex reasoning across multiple documents.
- **Lack of Explainability:** Hard to trace the source of retrieved information.
- **Local Window:** Limited chunk-level context leads to incomplete responses.
- **Loss of Structural Relationships:** Ignores hierarchical relationships in data.

YHK

Challenges with Microsoft's GraphRAG

- Too expensive and impractical for large-scale industrial use.
- Most companies prefer standard vector databases.
- Lack of widespread production adoption due to cost and complexity.

YHK

Text-To-Cypher/SPARQL Alternative

- Effective alternative to Microsoft's GraphRAG.
- Requires costly LLM calls for query generation.
- Adds uncertainty due to prompt effectiveness and model performance.
- Increases response time and implementation complexity.

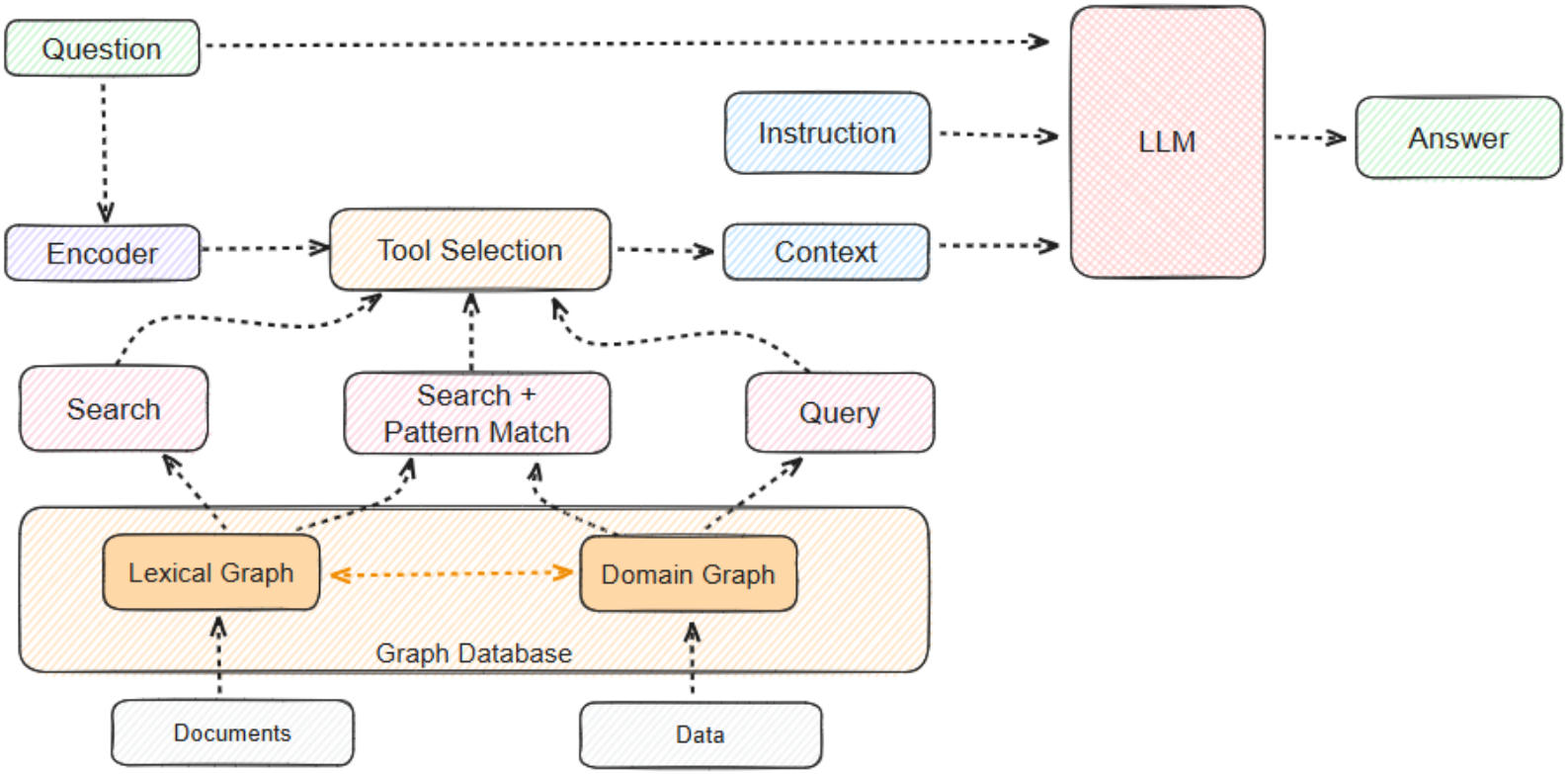
YHK

What is GraphRAG?

- No universally accepted definition yet.
- Some associate it with Microsoft's graph-based search approach.
- Others define it as querying LPG or RDF graphs using LLM-generated queries (Cypher, SPARQL).
- Uses knowledge graphs instead of unstructured text.
- Captures entities, relationships, and hierarchical structures.
- Enables accurate, context-aware retrieval and response generation.
- Supports complex query handling with enhanced explainability.
- Combines structured Knowledge Graphs (KGs) with semantic vectors.
- Enables LLMs to reason over multi-hop connections.
- Provides a holistic perspective by connecting different data sources.

YHK

Basics of GraphRAG



(Ref: https://graphrag.com/concepts/intro-to-graphrag/))

YHK

Why RAG?

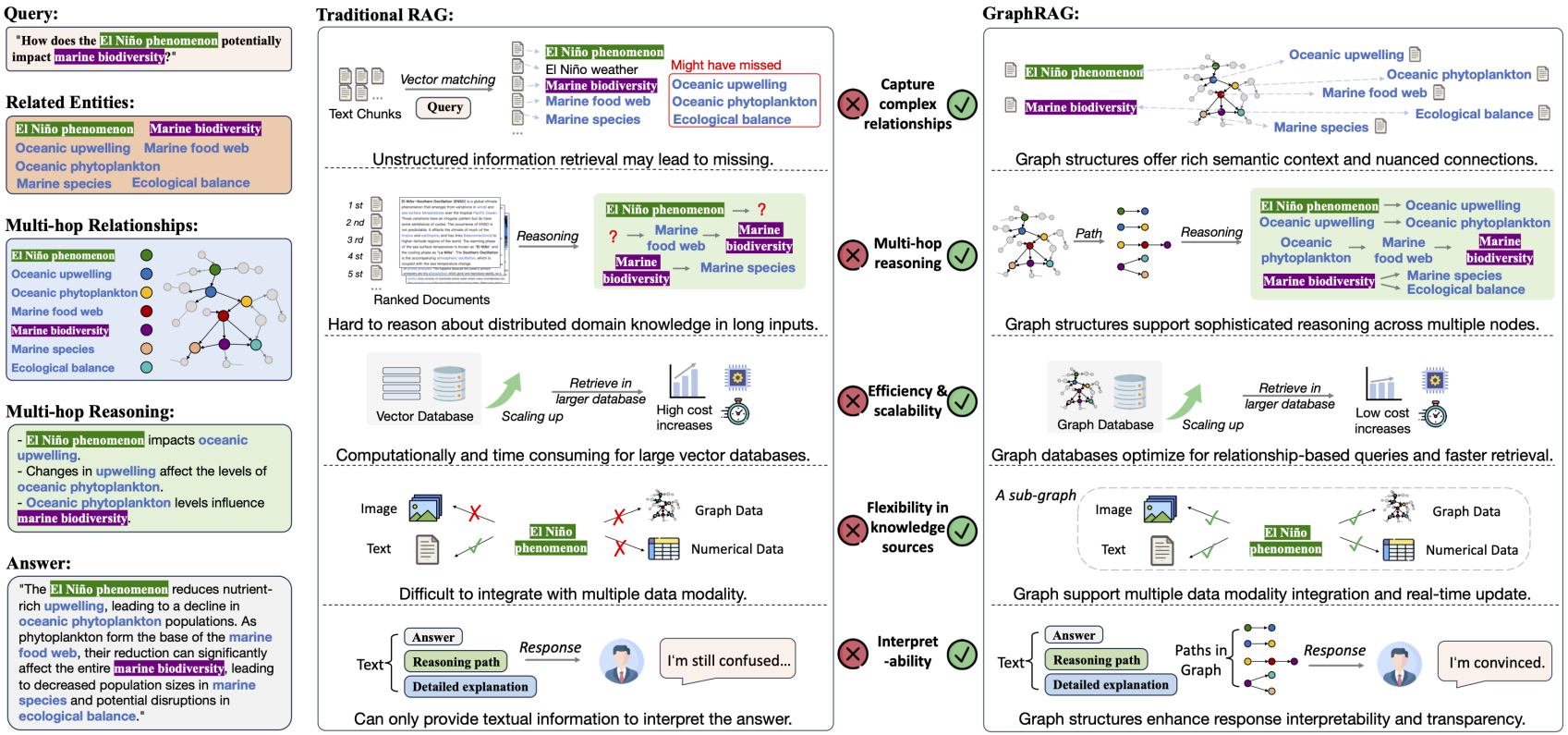
- RAG is widely used for real-world enterprise applications.
- Augments external knowledge sources to query private corpora.
- Traditional vector-based RAG relies only on semantic similarity.

YHK

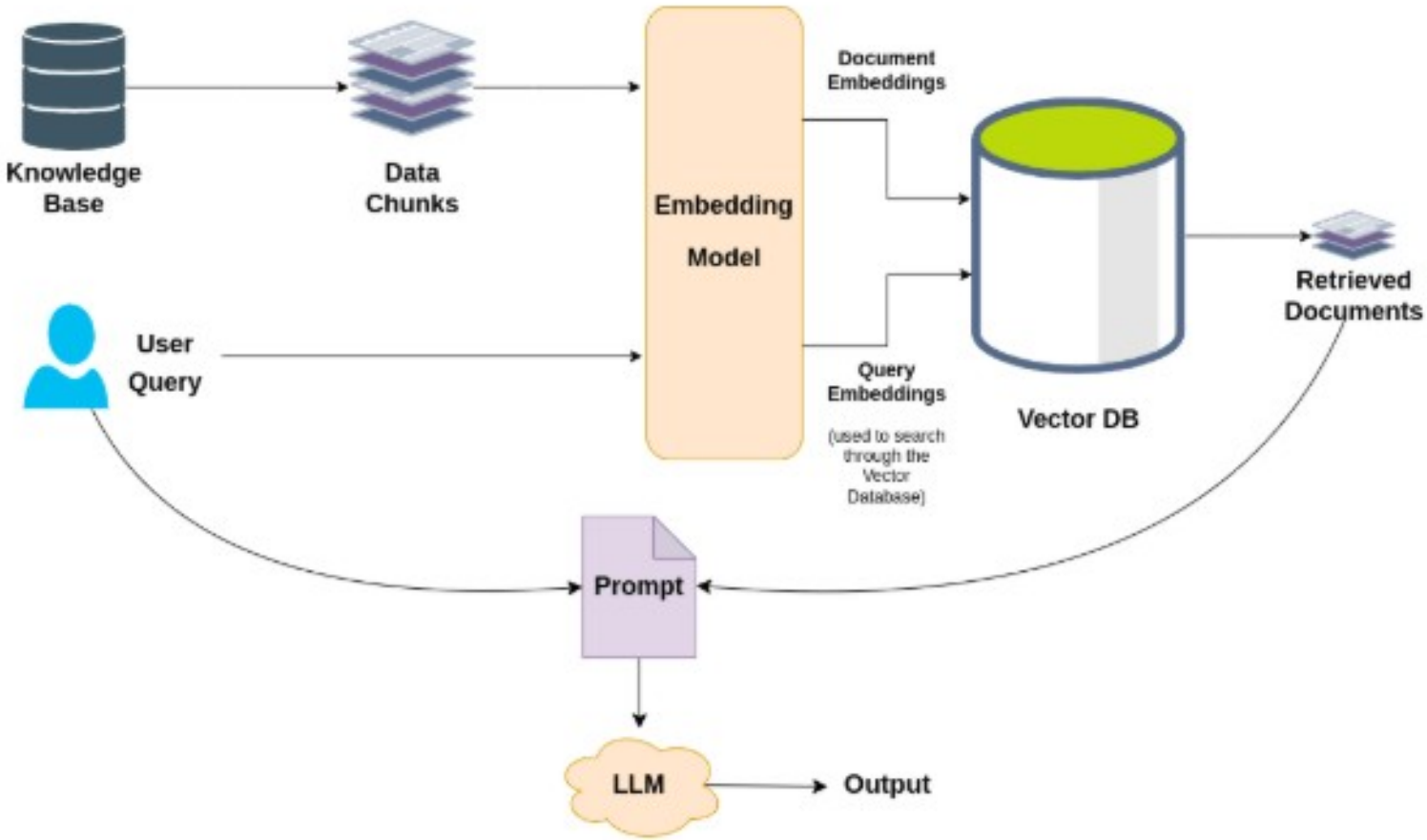
Key Features of GraphRAG

- **Structured Representation:** Uses knowledge graphs.
- **Contextual Retrieval:** Understands semantic relationships.
- **Efficient Processing:** Reduces computational cost.
- **Multi-Faceted Queries:** Synthesizes data from multiple sources.
- **Explainability:** More transparent than black-box models.
- **Continuous Learning:** Expands knowledge over time.

YHK


Query:
"How does the El Niño phenomenon potentially impact marine biodiversity?"
Related Entities:
El Niño phenomenon
Marine biodiversity
Oceanic upwelling
Marine food web
Oceanic phytoplankton
Marine species
Ecological balance
Multi-hop Relationships:
El Niño phenomenon
Oceanic upwelling
Oceanic phytoplankton
Marine food web
Marine biodiversity
Marine species
Ecological balance
Multi-hop Reasoning:
- El Niño phenomenon impacts oceanic upwelling.
- Changes in upwelling affect the levels of oceanic phytoplankton.
- Oceanic phytoplankton levels influence marine biodiversity.
Answer:
"The El Niño phenomenon reduces nutrient-rich upwelling, leading to a decline in oceanic phytoplankton populations. As phytoplankton form the base of the marine food web, their reduction can significantly affect the entire marine biodiversity, leading to decreased population sizes in marine species and potential disruptions in ecological balance."
Traditional RAG:
Vector matching
Text Chunks
Query
El Niño phenomenon
El Niño weather
Marine biodiversity
Marine food web
Marine species
Might have missed
Oceanic upwelling
Oceanic phytoplankton
Ecological balance
Unstructured information retrieval may lead to missing.
Reasoning
Ranked Documents
Hard to reason about distributed domain knowledge in long inputs.
Vector Database
Scaling up
Retrieve in larger database
High cost increases
Computationally and time consuming for large vector databases.
Image
Text
El Niño phenomenon
Graph Data
Numerical Data
Difficult to integrate with multiple data modality.
Answer
Reasoning path
Detailed explanation
Response
I'm still confused...
Can only provide textual information to interpret the answer.
Capture complex relationships
Multi-hop reasoning
Efficiency & scalability
Flexibility in knowledge sources
Interpret -ability
GraphRAG:
Oceanic upwelling
Oceanic phytoplankton
Marine food web
Ecological balance
Marine species
Graph structures offer rich semantic context and nuanced connections.
Path
Reasoning
Graph structures support sophisticated reasoning across multiple nodes.
Graph Database
Scaling up
Retrieve in larger database
Low cost increases
Graph databases optimize for relationship-based queries and faster retrieval.
A sub-graph
Graph support multiple data modality integration and real-time update.
Paths in Graph
I'm convinced.
Graph structures enhance response interpretability and transparency.

Limitations of Traditional RAG



(Ref: GraphRAG: The Practical Guide for Cost-Effective Document Analysis with Knowledge Graphs -Jaykumaran)

YHK

Challenges in Standard RAG

- **Lack of Explainability:** Hard to trace the source of retrieved information.
- **Local Window:** Limited chunk-level context leads to incomplete responses.
- **Scalability Issues:** Struggles with large-scale medical and legal corpora.
- **Loss of Structural Relationships:** Ignores hierarchical relationships in data.

YHK

GraphRAG: Evolution of Knowledge Graphs

- Earlier, KGs were built using statistical and NLP techniques.
- GraphRAG scales efficiently by using LLMs for entity extraction.
- Entities in KG are nodes, linked by edges encoding relationships.

YHK

Advantages of Knowledge Graphs

- Connects information from multiple sources for deeper insights.
- Boosts retrieval accuracy and enables multi-hop reasoning.
- Enhances LLM responses by integrating structured relationships.

YHK

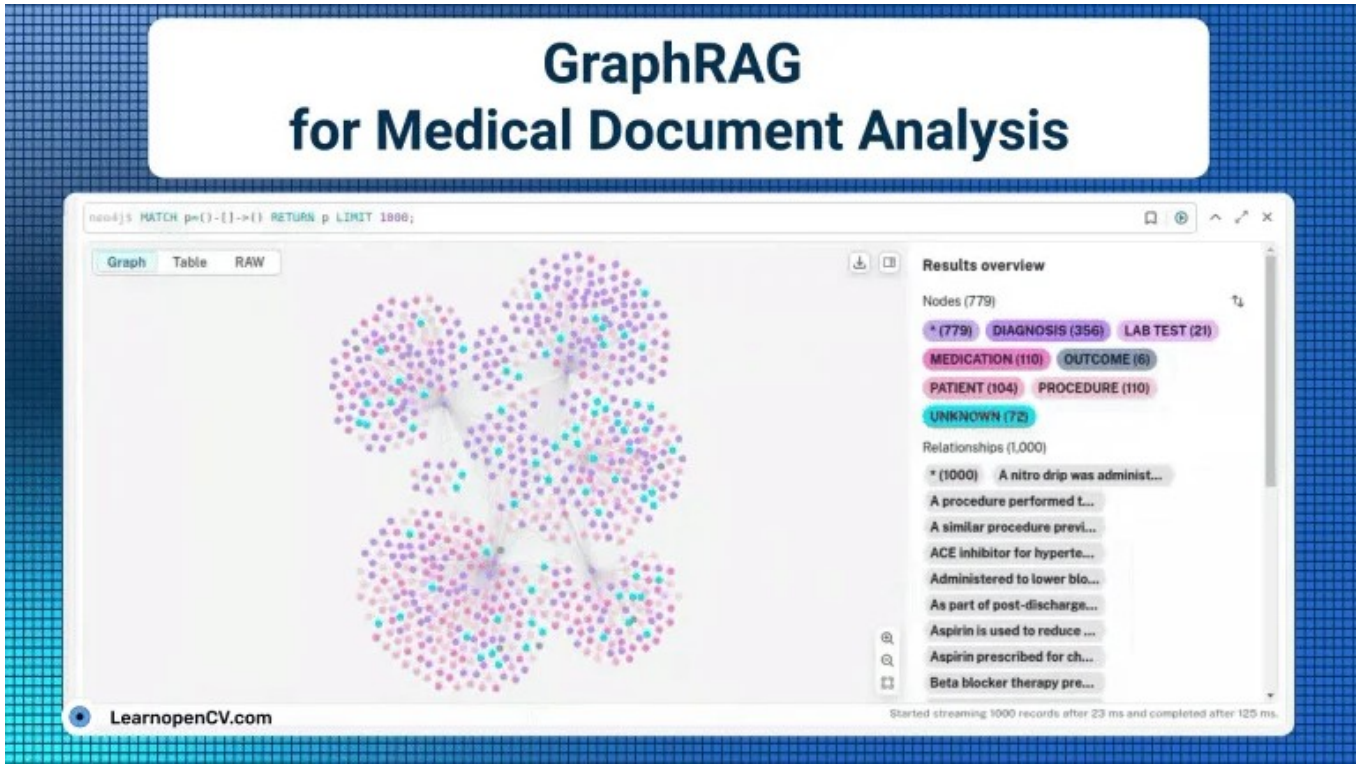
Key Features of GraphRAG

- Maintains hierarchical communities preserving local and global insights.
- Ensures source traceability down to the node level for citations.
- Aggregates information from multiple sources, reducing bias.
- Focuses on meaningful nodes, filtering out irrelevant information.

YHK

Example: Medical Query Challenge

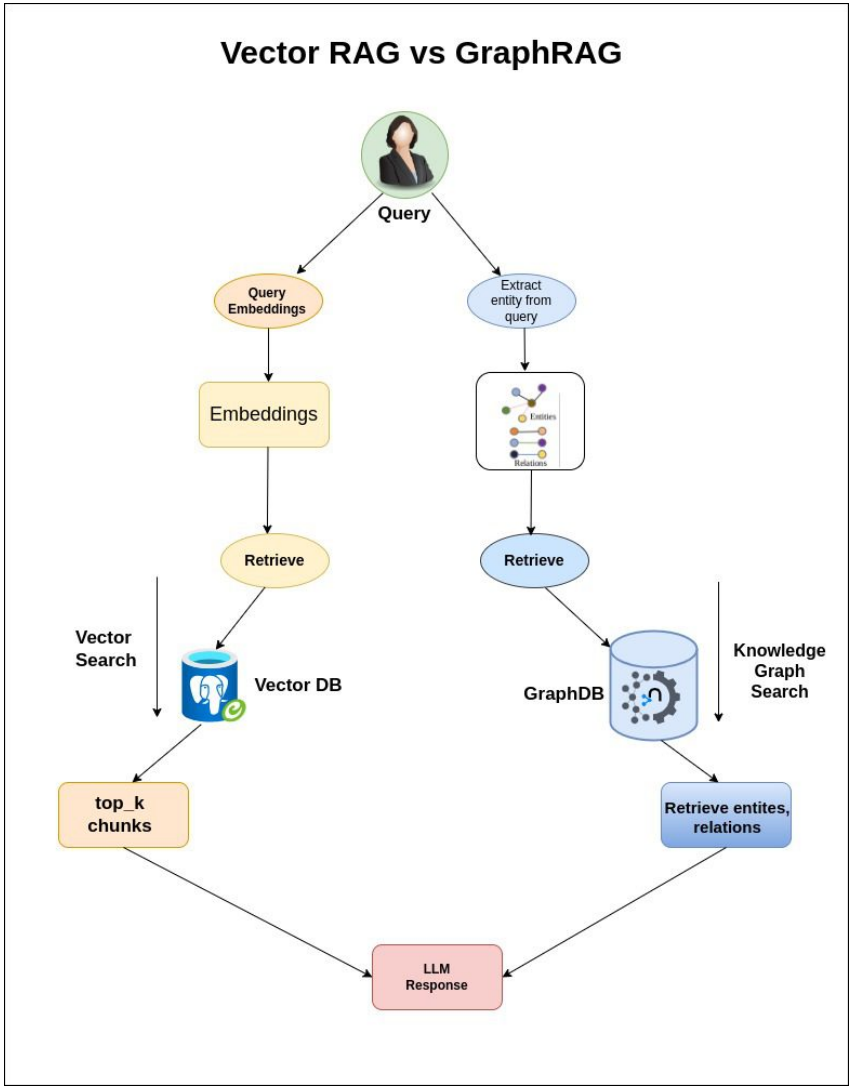
- Query: How does Medication A in Patient Record 1 affect Condition B in Patient Record 2?
- LLM needs to infer relationships across multiple records.
- Standard RAG struggles with such complex dependencies.
- Scaling this to millions of patient records is infeasible.



(Ref: GraphRAG: The Practical Guide for Cost-Effective Document Analysis with Knowledge Graphs -Jaykumaran)

YHK

Vector RAG vs GraphRAG



(Ref: GraphRAG: The Practical Guide for Cost-Effective Document Analysis with Knowledge Graphs -Jaykumaran)

Traditional RAG vs GraphRAG

Feature	**Traditional RAG**	**GraphRAG**
Data Representation	Flat Vectors	Knowledge Graph
Query Scope	Local context	Global Reasoning
Scalability	Low	High
Citation Transparency	Low	High (Traceable sources)
Response Coherence	Fragmented	Relevant and Context-Rich

YHK

How GraphRAG Solves the Problem

- GraphRAG combines graph structures with vector search.
- Traverses multi-hop connections to infer relationships.
- Offers more reliable responses than vector-only RAG.

Graph-based RAG helps agentic AI make human-like decisions. – May Habib, CEO of Writer.com

YHK

Applications of GraphRAG

- **Healthcare:** Assists in diagnoses and treatment decisions.
- **Banking:** Detects fraudulent transactions using knowledge graphs.
- **Customer Service :** quickly answer customer questions from thousands of pages of policy documentation
- **Recommendations:** understand customer behavior and preferences better, to provide personalized services.
- **Supply Chain:** product recall and associated quality control checking, internal documentation search

YHK

How GraphRAG Works?

- **Knowledge Graph Construction:** Extracts entities and relationships.
- **Knowledge Graph Summarization:** Generates hierarchical summaries.
- **Retrieval-Augmented Generation:** Uses local and global searches for queries.

YHK

Example of GraphRAG Representation

```
# Entities
Type 2 Diabetes (Condition)
High Blood Sugar Levels (Symptom)
Nerve Damage (Complication)
Kidney Disease (Complication)
Cardiovascular Problems (Complication)

# Relationships
Type 2 Diabetes -> has_symptom -> High Blood Sugar Levels
Type 2 Diabetes -> can_lead_to -> Nerve Damage
Type 2 Diabetes -> can_lead_to -> Kidney Disease
Type 2 Diabetes -> can_lead_to -> Cardiovascular Problems

```

YHK

Advantages of GraphRAG

- Structured knowledge representation.
- Context-aware and efficient retrieval.
- Handles complex queries effectively.
- Provides explainability and transparency.

YHK

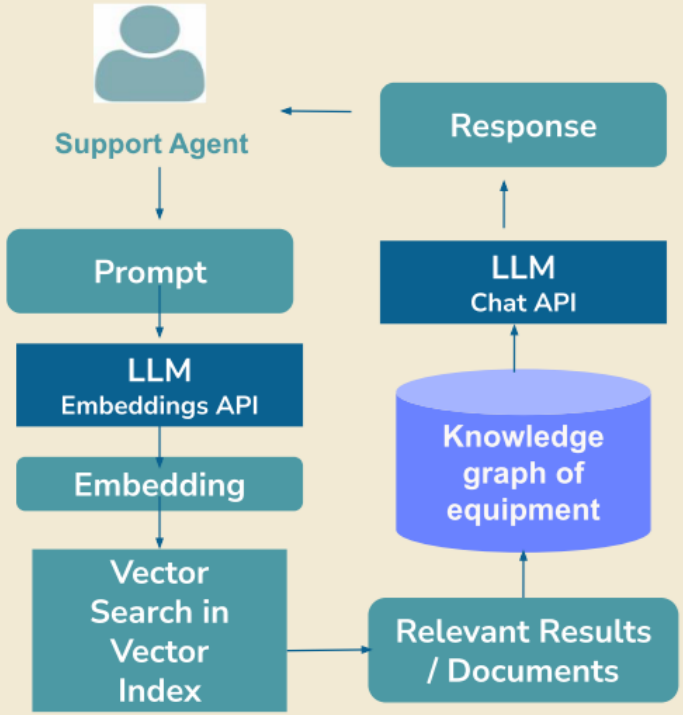
Improve Answers: Pure text w/ Knowledge Graph

Global Equipment Manufacturer:

Challenge: Support agents want to lower their mean time to repair (MTTR) by quickly providing a resolution, but the best answer is hidden among thousands of field docs, engineering summaries, documentation, and case histories.

Solution: Use vector embeddings to look for the most relevant information, while the Neo4j knowledge graph adds the exact match for critical components.

Desired Outcome: Reduce MTTR, and provide the best option for the customer quickly and with confidence.



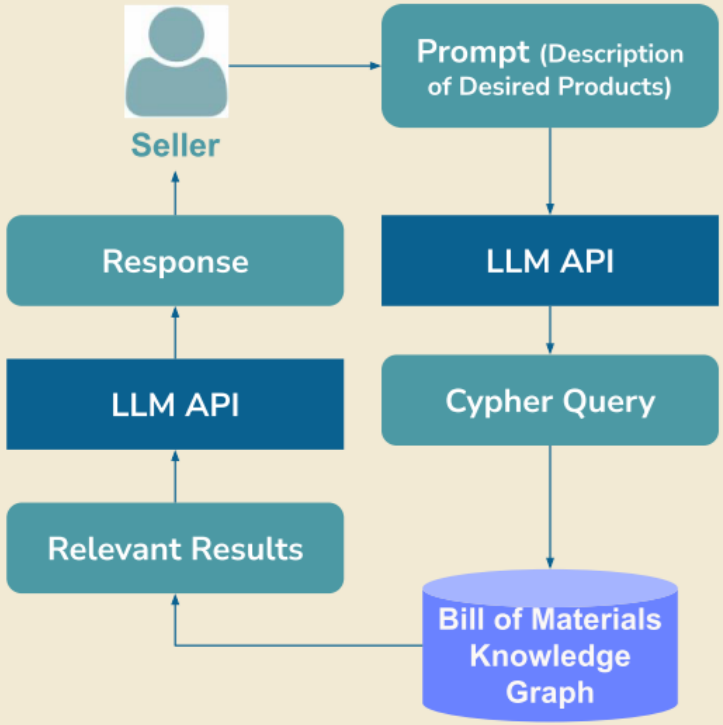
Knowledge Access: Pure data w/ Text to Query

Global Electronics Manufacturer:

Challenge: Sellers want to quote the right product combination and price for every opportunity across five unique types of external stakeholders. Information is stored across millions of bills of materials and product combinations.

Solution: Bring bills of materials into a knowledge graph that can be queried by an LLM interface by sellers.

Desired Outcome: Quote the best option for each customer quickly and with confidence.



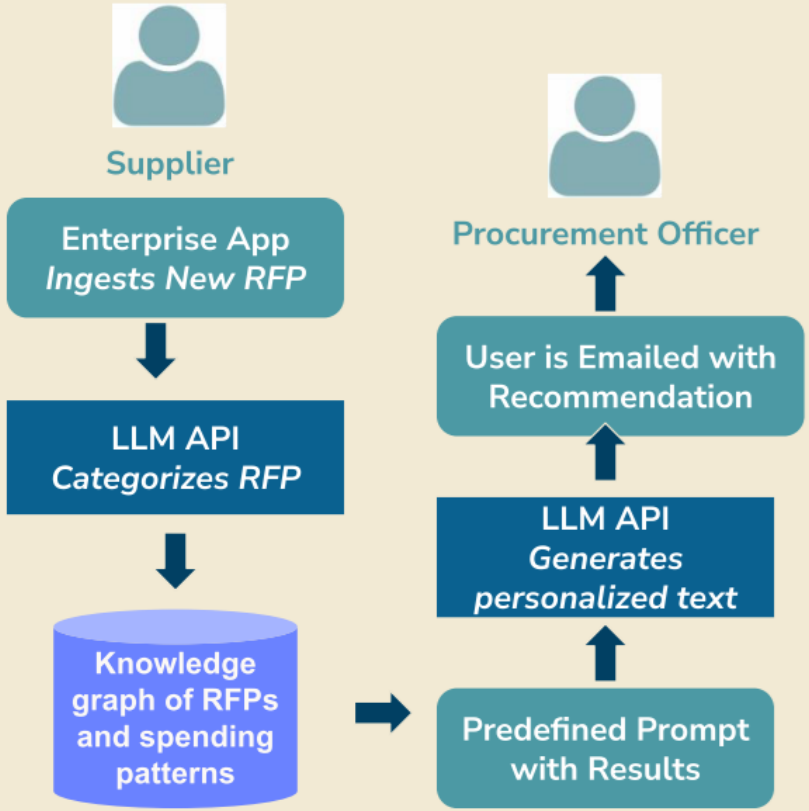
Complex Processes: Mixed Text + Data

Government Procurement Entity:

Challenge: Government entities can overspend or acquire goods and services redundantly because the volume of RFPs makes reviewing each of them resource prohibitive.

Solution: Use LLM to read the nature of RFP and classify it accordingly. Compare the new RFP against the knowledge graph of active and historic RFPs and associated spend. Recommend opportunities for consolidation or terms negotiation with suppliers.

Desired Outcome: Save tax dollars for other important projects

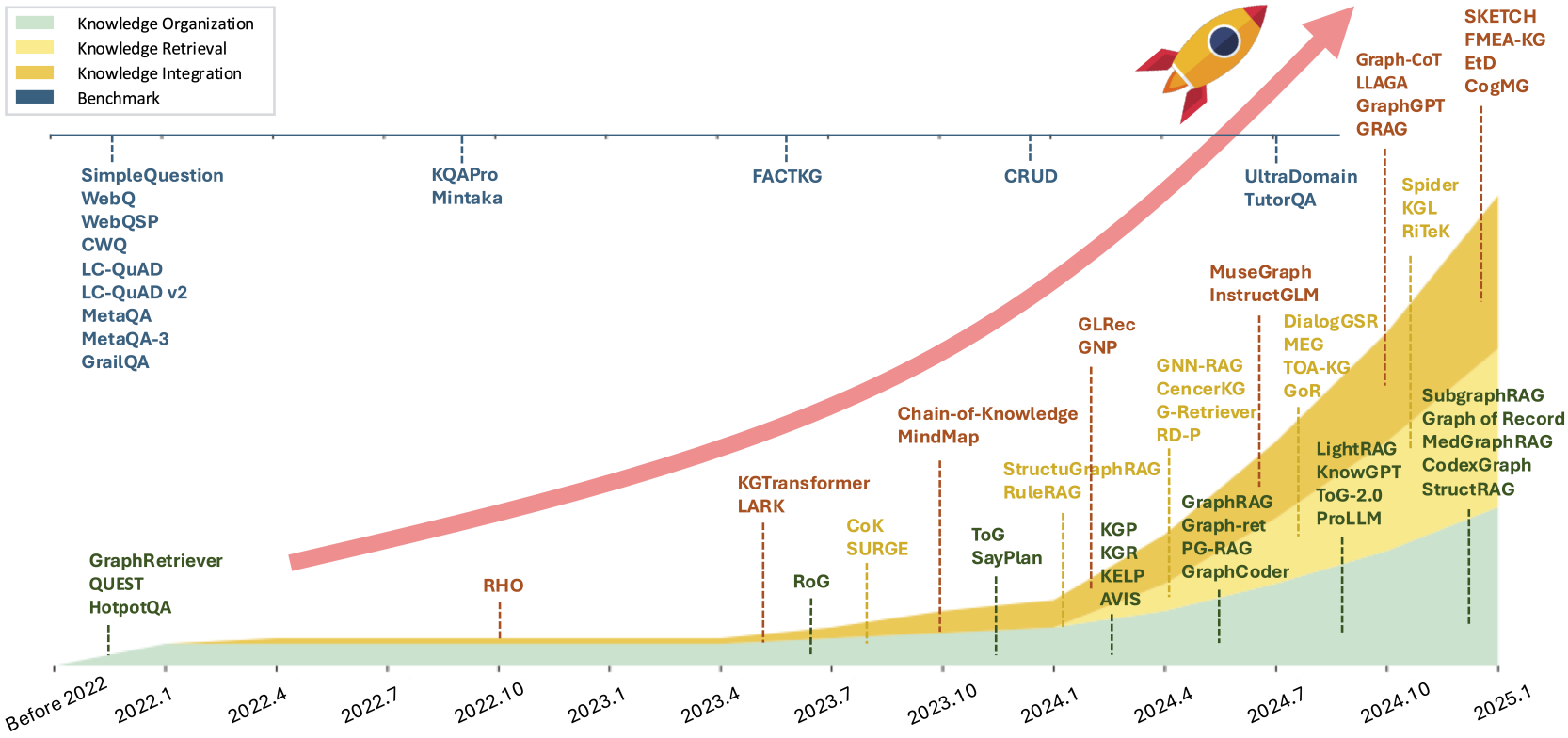


Challenges of GraphRAG

- **Complex Knowledge Graph Construction:** Requires sophisticated NLP techniques.
- **Data Dependency:** Performance relies on input data quality.
- **Scalability Issues:** Large graphs require significant computational resources.

YHK

Trend of GraphRAG Research



(Ref: Awesome-GraphRAG (GraphRAG Survey))

YHK

Conclusion

- GraphRAG enhances traditional RAG models using structured knowledge.
- Improves accuracy, context-awareness, and efficiency.
- Useful in various domains like healthcare and banking.
- A promising approach for future AI-powered knowledge retrieval.

YHK

Concepts

Cost-Efficient GraphRAG Approach

- Leverages graphs for RAG without high costs.
- Minimizes reliance on LLMs or uses smaller on-premise models.
- Structured as a layered graph:
  - Ontology Layer – Defines domain structure (fixed or nearly fixed).
  - Document Layer – Contains chunked documents, similar to a vector DB.
  - Entity Layer (Optional) – Extracted entities enhance search.

YHK

Challenges in Ontology-Based Graphs

- Not all datasets belong to a well-defined domain.
- Subject Matter Experts (SMEs) may not be available.
- Eliminating fixed ontology layer could improve flexibility.

YHK

NLP-Powered Graph Approach

- Drops the ontology layer for cost-efficiency.
- Graph consists of:
  - Document Layer – Contains document chunks.
  - Tokens Layer – Extracted tokens improve search.
- NLP reduces dependency on LLMs.

YHK

Data Preprocessing Pipeline

- Chunking – Splitting documents into segments.
- Embedding – Using Hugging Face model for embeddings.
- Graph Construction – Built using NetworkX or Neo4j.
- Token Extraction – Generates token, bigram, and trigram nodes.

YHK

Indexing and Interconnectivity

- Tokens are shared across documents, interlinking content.
- Need to connect entities using context, logic, and semantics.
- Avoid reliance on massive models due to cost constraints.

YHK

Triple Extraction for Graph Optimization

- Triple extraction improves retrieval efficiency.
- Uses a smaller transformer model fine-tuned for this task.
- Triplets mapped to token nodes enhance query performance.
- Queries traverse graph using triplet relationships.

YHK

Optimizing Graph-Based Retrieval

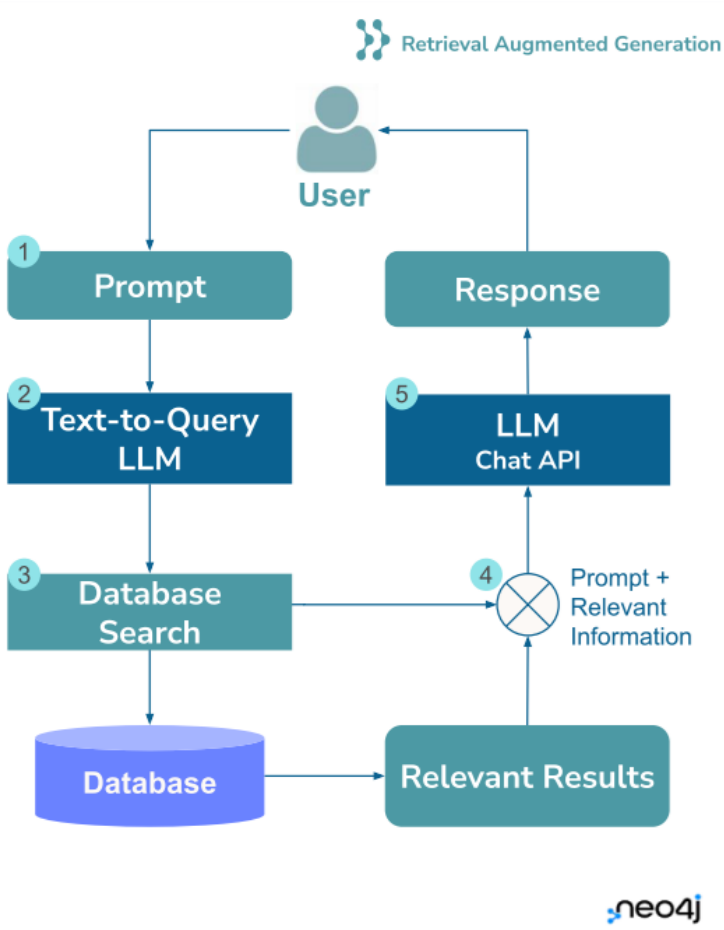
- Combines standard RAG with triplet-enhanced retrieval.
- Retrieves relevant text chunks and connected triplets.
- Improves search relevance and reduces retrieval complexity.

YHK

Approaches

Pure Data Retrieval

1. accept prompt from user
2. pass prompt to a fine-tuned code generation model
3. run query against database
4. combine user prompt with query results
5. generate natural language results with LLM



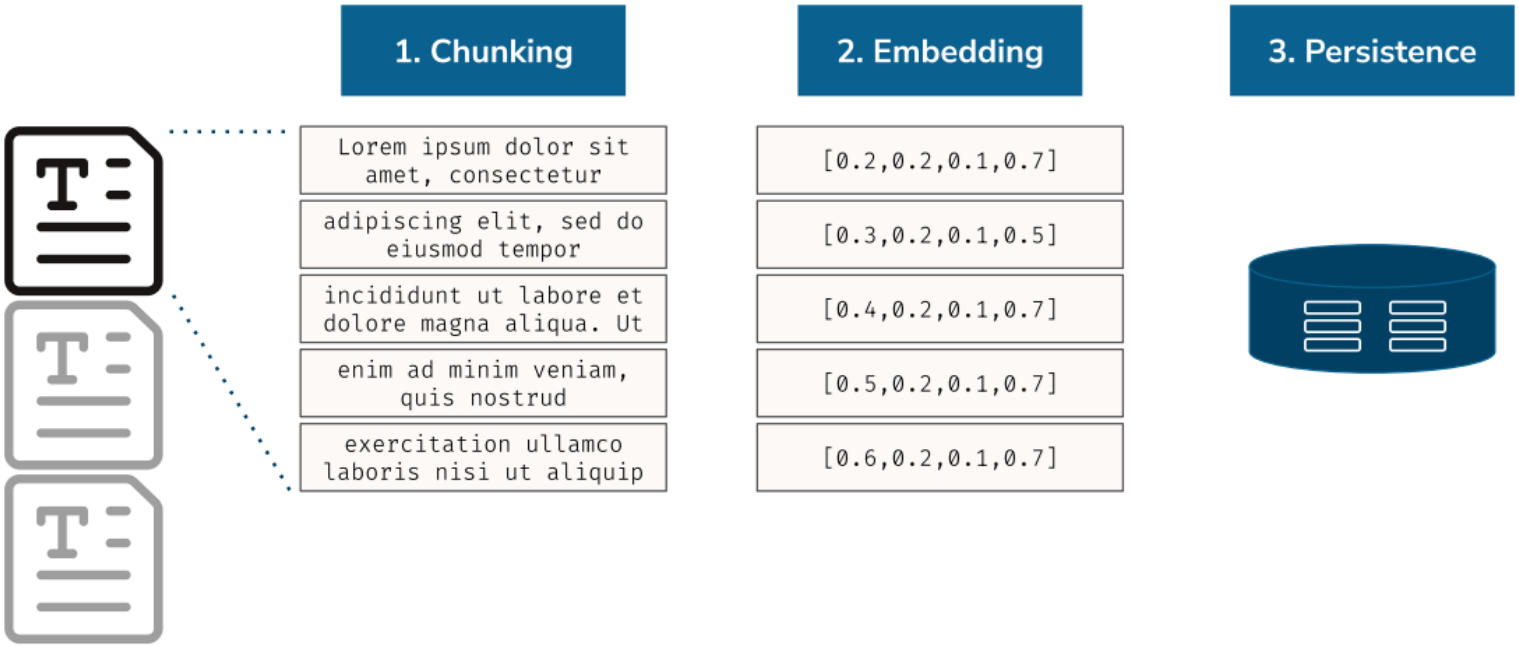
Challenges

- Getting it to work at all – generating syntactically correct queries
- Getting it to do the right thing – producing meaningful results
- Avoiding accidents – mistaken deletion
- Preventing malicious intent – SQL injection gone wild

(Ref: The GenAI Stack - Andreas Kollegger - Neo4j)

YHK

Pure Text Preparation



Challenges

- How?
  - pick a chunk method & size
  - each chunk is a record
  - store chunk with metadata
  - connect each chunk to original document
  - connect previous/next chunk
- Challenges:
  - what makes a good chunk?
  - potential chunk duplication
  - how to re-assemble chunk context?
  - what about cross-document chunks?
  - explaining the relevance

(Ref: The GenAI Stack - Andreas Kollegger - Neo4j)

YHK

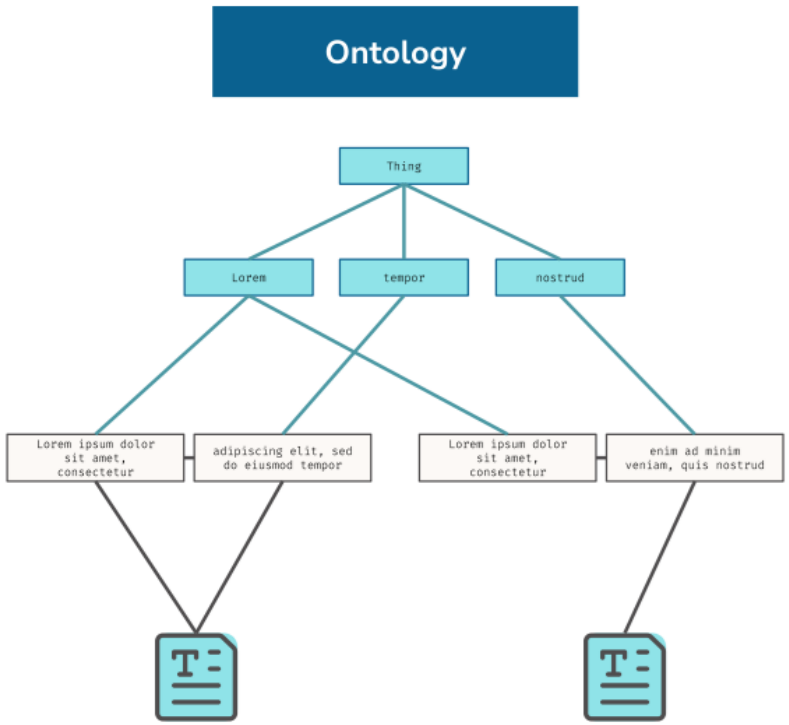
Explicit Similarity

How?

-
- named entity recognition
- metadata extraction
- document cross-linking, page ranking

Challenges:

- does chunk similarity provide the most relevant answer?
- what about the person asking the question?



YHK

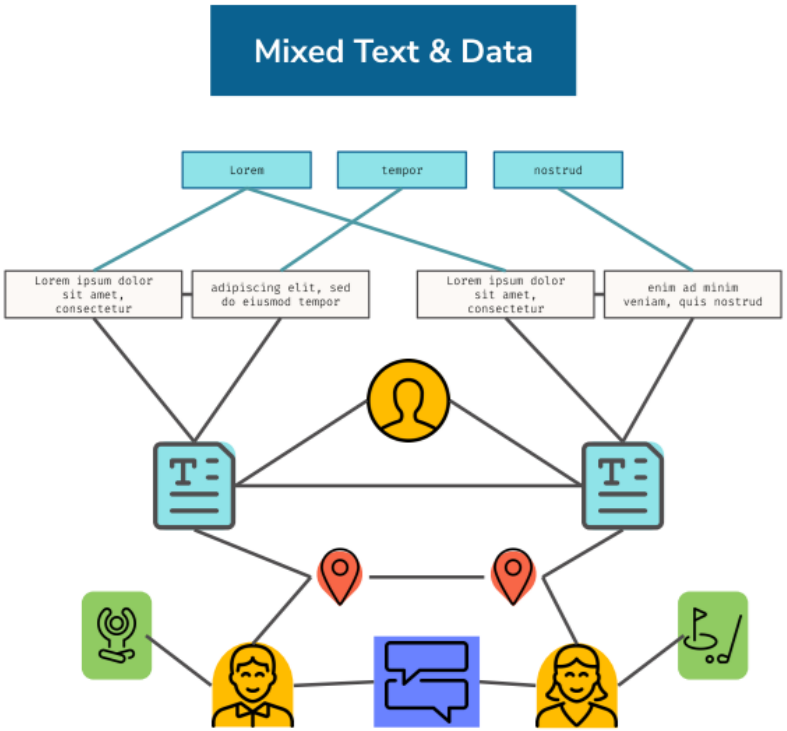
Mixed Text & Data

How?

- extracted information
- application data
- user data
- explicit semantic connections

Challenges:

- what is most relevant, to the user?



Knowledge Graphs

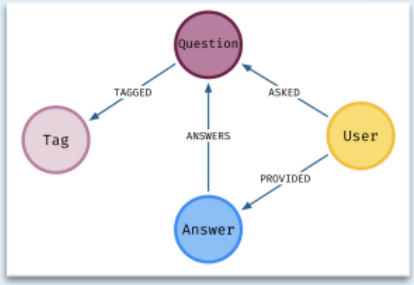
Definition

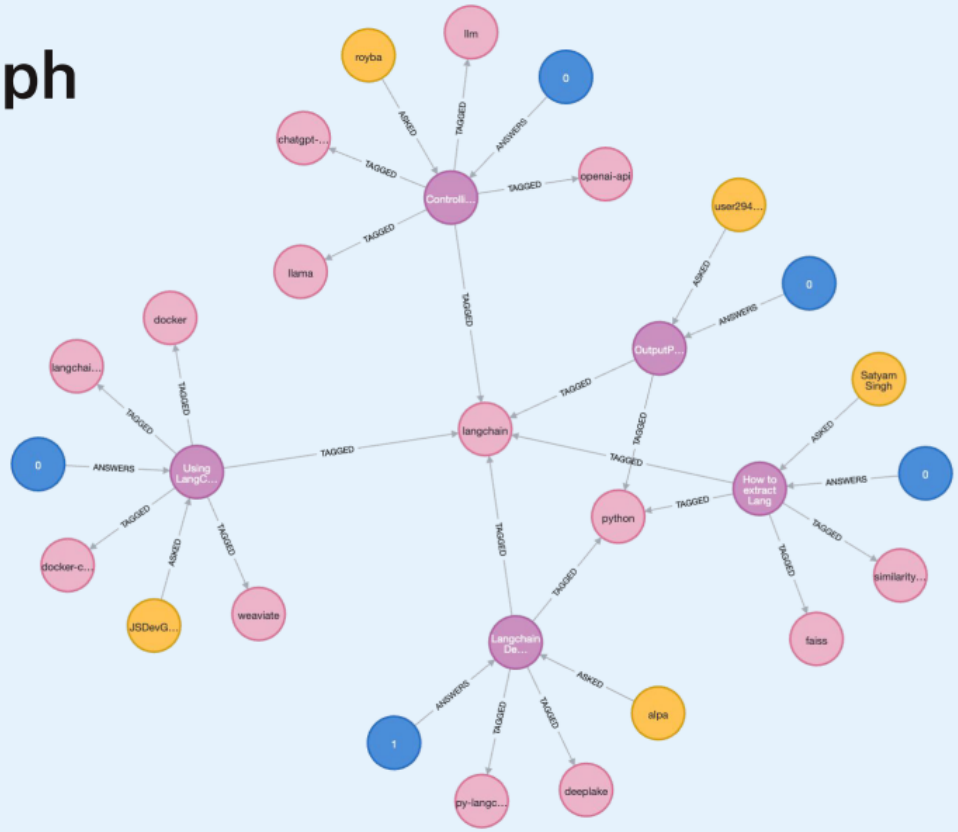
A Knowledge Graph is a structured way of representing information, typically using nodes and edges to depict relationships between entities (e.g., people, places, things, concepts). These entities and their interconnections form a graph-like structure, which can be used to model complex sets of data and the relationships within that data.

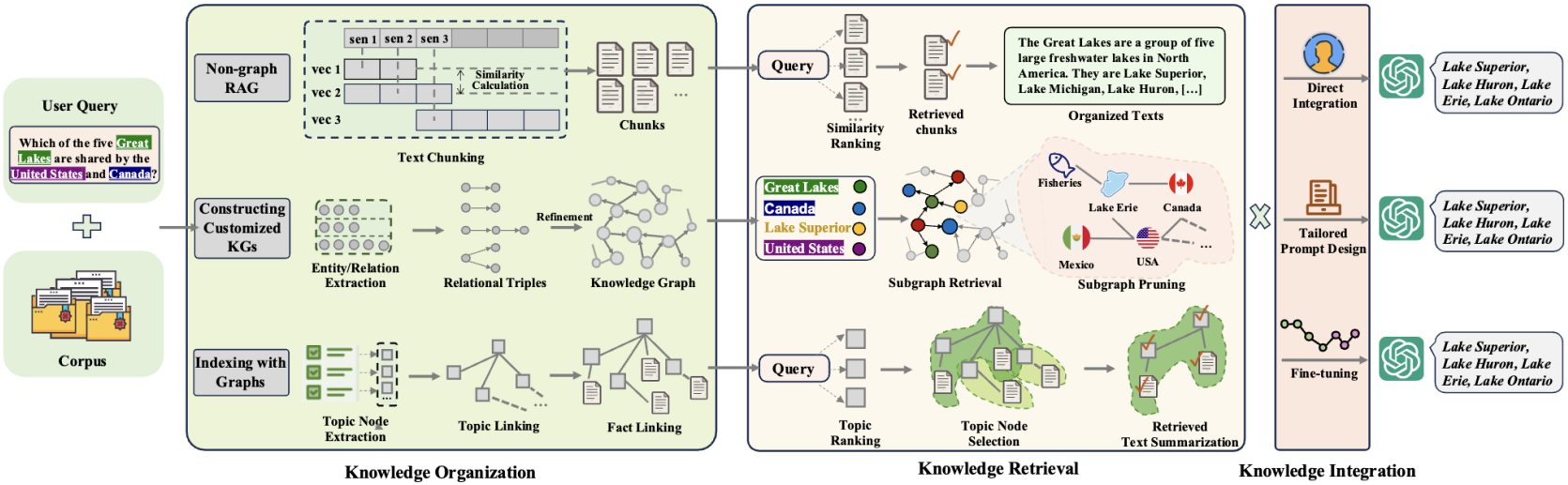
(Ref: The GenAI Stack - Andreas Kollegger - Neo4j)

YHK

StackOverflow Knowledge Graph






User Query
Which of the five Great Lakes are shared by the United States and Canada?
Corpus
Non-graph RAG
sen 1 sen 2 sen 3
vec 1
vec 2
vec 3
Similarity Calculation
Text Chunking
Chunks
Constructing Customized KGs
Entity/Relation Extraction
Relational Triples
Refinement
Knowledge Graph
Indexing with Graphs
Topic Node Extraction
Topic Linking
Fact Linking
Knowledge Organization
Query
Similarity Ranking
Retrieved chunks
The Great Lakes are a group of five large freshwater lakes in North America. They are Lake Superior, Lake Michigan, Lake Huron, [...]
Organized Texts
Great Lakes
Canada
Lake Superior
United States
Subgraph Retrieval
Fisheries
Lake Erie
Canada
Mexico
USA
Subgraph Pruning
Query
Topic Ranking
Topic Node Selection
Retrieved Text Summarization
Knowledge Retrieval
Direct Integration
Tailored Prompt Design
Fine-tuning
Lake Superior, Lake Huron, Lake Erie, Lake Ontario
Lake Superior, Lake Huron, Lake Erie, Lake Ontario
Lake Superior, Lake Huron, Lake Erie, Lake Ontario
Knowledge Integration

Overview

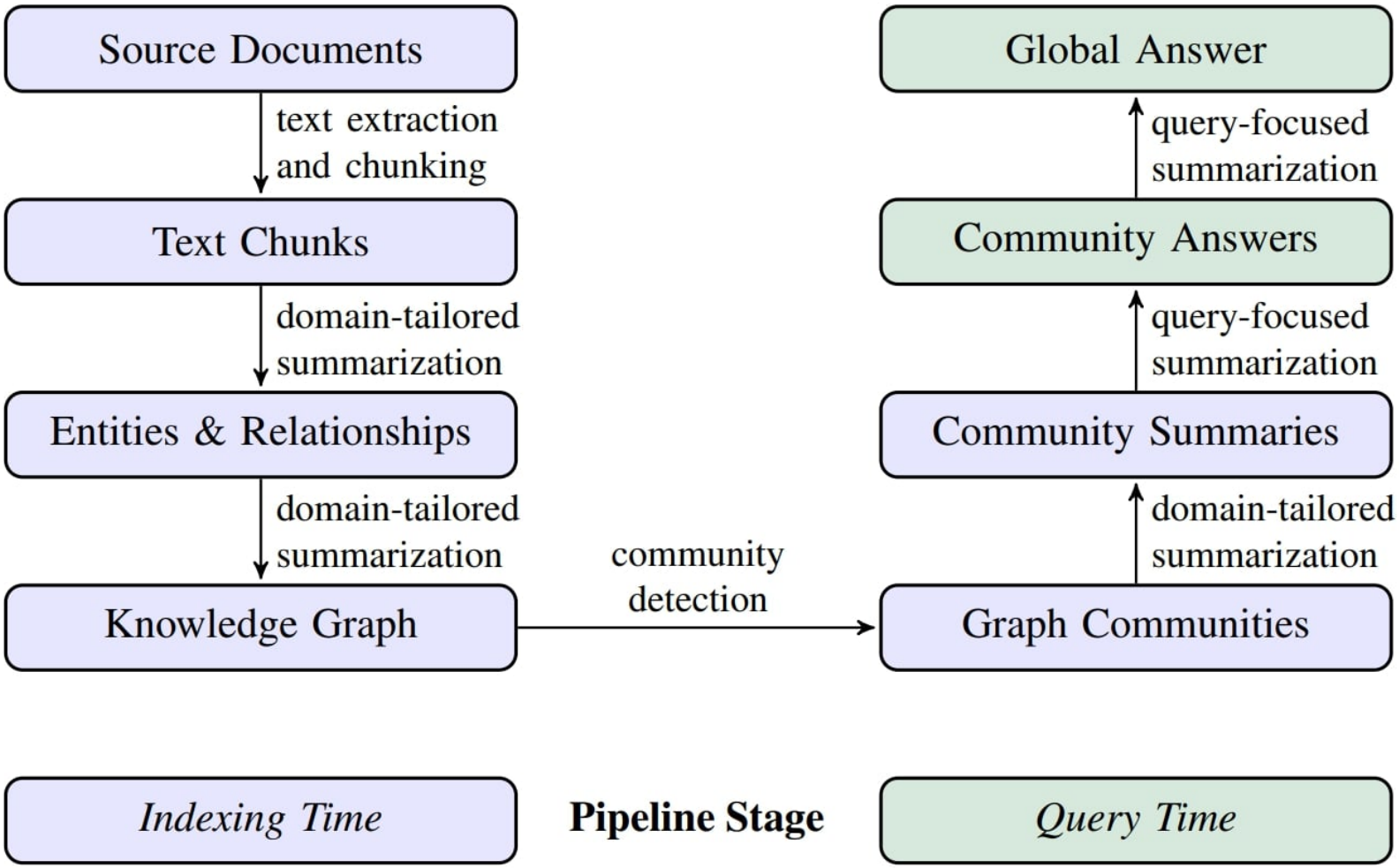
Overview of traditional RAG and two typical GraphRAG workflows.

- Non-graph RAG organizes the corpus into chunks, ranks them by similarity, and retrieves the most relevant text for generating responses.
- Knowledge-based GraphRAG extracts detailed knowledge graphs from the corpus using entity recognition and relation extraction, offering fine-grained, domain-specific information.
- Index-based GraphRAG summarizes the corpus into high-level topic nodes, which are linked to form an index graph, while the fact linking maps topics to text.

(Ref: Awesome-GraphRAG (GraphRAG Survey))

YHK

Microsoft GraphRAG Workflow



(Ref: GraphRAG: The Practical Guide for Cost-Effective Document Analysis with Knowledge Graphs -Jaykumaran)

YHK

Indexing Phase Overview

- Two-stage process: Generate Knowledge Graph (KG) and Community Clustering.
- Extract entities and relationships, construct KG.
- Partition KG into semantically related clusters.
- Generate community summaries for optimized retrieval.

YHK

Step 1: Segment Documents into Chunks

- Large text split into smaller chunks for LLM processing.
- Small chunks retain details but increase API cost.
- Larger chunks reduce cost but may omit key data.
- Self-reflection in prompts improves extraction accuracy.

YHK

Step 2: Extract Entities and Relationships

- LLM extracts entities, relationships, and descriptions.
- Entities assigned unique IDs for traceability.
- Pronouns and ambiguities resolved.

```
{
  "nodes": [
    {"id": "n0", "entity_type": "PATIENT", "name": "71-YEAR-OLD
GENTLEMEN"},
    {"id": "n1", "entity_type": "DIAGNOSIS", "name": "PERIPHERAL VASCULAR
DISEASE"}
  ],
  "edges": [
    {"id": "e0", "source": "n0", "target": "n1", "description": "Patient
has history of this condition"}
  ]
}

```

YHK

Step 2: Extract Entities and Relationships

```
**-Goal-**
Given a text document that is potentially relevant to this activity and a list
    of entity types, identify all entities of those types from the text and
    all relationships among the identified entities.

-Steps-
1. Identify all entities. For each identified entity, extract the following
    information:
- entity_name: Name of the entity, capitalized
- entity_type: One of the following types: [{entity_types}]
- entity_description: Comprehensive description of the entity's

2. From the entities identified in step 1, identify all pairs of
    (source_entity, target_entity) that are *clearly related to each other.

For each pair of related entities, extract the following information:
- source_entity: name of the source entity, as identified in step 1
- target_entity: name of the target entity, as identified in step 1
- relationship_description: explanation as to why you think the source entity
    and the target entity are related to each other
- relationship_strength: a numeric score indicating strength of the
    relationship between the source entity and target entity
```

YHK

Step 2: Extract Entities and Relationships

```
# Entities:
{
  "nodes": [
    {
      "id": "n0",
      "entity_type": "PATIENT",
      "description": "A male patient aged 71 with several medical conditions",
      "source_id": "n0",
      "name": "71-YEAR-OLD GENTLEMEN"
    },
    {
      "id": "n1",
      "entity_type": "DIAGNOSIS",
      "description": "Condition treated with right leg angioplasty\nPeripheral
     vascular disease (PVD) is a circulatory condition characterized by
     narrowed or blocked arteries outside of the heart and brain, leading to
     reduced blood flow, particularly to limbs. Patients with PVD may have
     undergone procedures like angioplasty in affected areas to improve
     circulation. Treatment generally involves medications such as Plavix and
     aspirin, as well as surgical interventions based on individual
     assessments.",
      "source_id": "n1",
      "name": "PERIPHERAL VASCULAR DISEASE"
    },

```

YHK

Step 2: Extract Entities and Relationships

```
# Relations
 "edges": [
    {
      "id": e0,
      "source": "n0",
      "target": "n1",
      "description": "The patient has a medical history of this condition",
    },
    {
      "id": e1,
      "source": "n0",
      "target": "n2",
      "description": "Chronic Obstructive Pulmonary Disease is part of his
     health profile",
    },

```

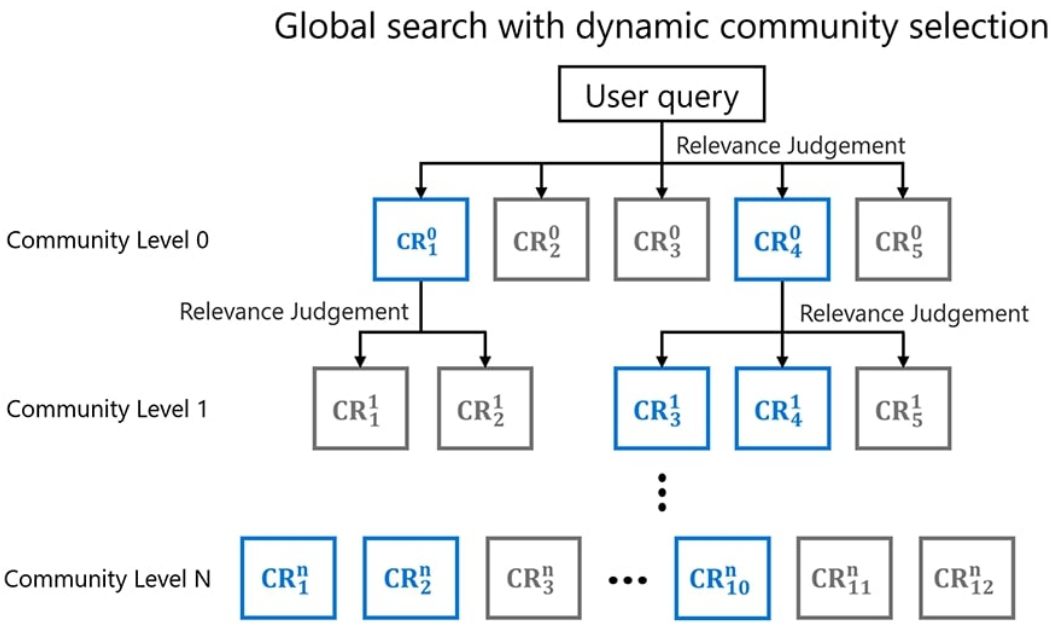
Step 3: Construct Knowledge Graph

- Entities and relationships aggregated into a unified KG.
- Duplicates merged for efficiency.

YHK

Step 4: Graph Partitioning into Communities

- Leiden algorithm partitions KG into clusters.
- Hierarchical approach ensures fine granularity.
- Nodes belong to only one community.
- Dynamic community selection at multiple levels.



Step 1: Recursive Community Selection
Start at root level (level 0), using an LLM to rate the relevance of each community report (CR) against the user query.

If the community report is not relevant (grey boxes), then we move on to the next report.

If the community report is relevant, then we repeat the same operation to its child nodes. We keep the report (blue boxes) if none of its child communities is relevant or if it is a leaf node.

Step 2: Map-reduce Search
Perform map-reduce search on the selected community reports to generate the final response.

(Ref: GraphRAG: The Practical Guide for Cost-Effective Document Analysis with Knowledge Graphs -Jaykumaran)

YHK

Step 5: Community Summarization

```
# **Goal**
Write a comprehensive report of a community, given a list of
    entities that belong to the community as well as their
    relationships and optional associated claims. The report
    will be used to inform decision-makers about information
    associated with the community and their potential impact.
    The content of this report includes an overview of the
    community's key entities, their legal compliance,
    technical capabilities, reputation, and noteworthy claims.


```

YHK

Step 5: Community Summarization

```
# Report Structure
The report should include the following sections:

- TITLE: community's name that represents its key entities
- SUMMARY: An executive summary of the community's overall
    structure,
- IMPACT SEVERITY RATING: a float score between 0-10 that
    represents the severity of IMPACT posed by entities
    within the community.
- RATING EXPLANATION: Give a single sentence explanation of
    the IMPACT severity rating.
- DETAILED FINDINGS: A list of 5-10 key insights about the
    community.

```

YHK

Querying Phase Overview

- Identifies key entities and relationships in queries.
- Retrieves relevant community summaries instead of entire KG.

YHK

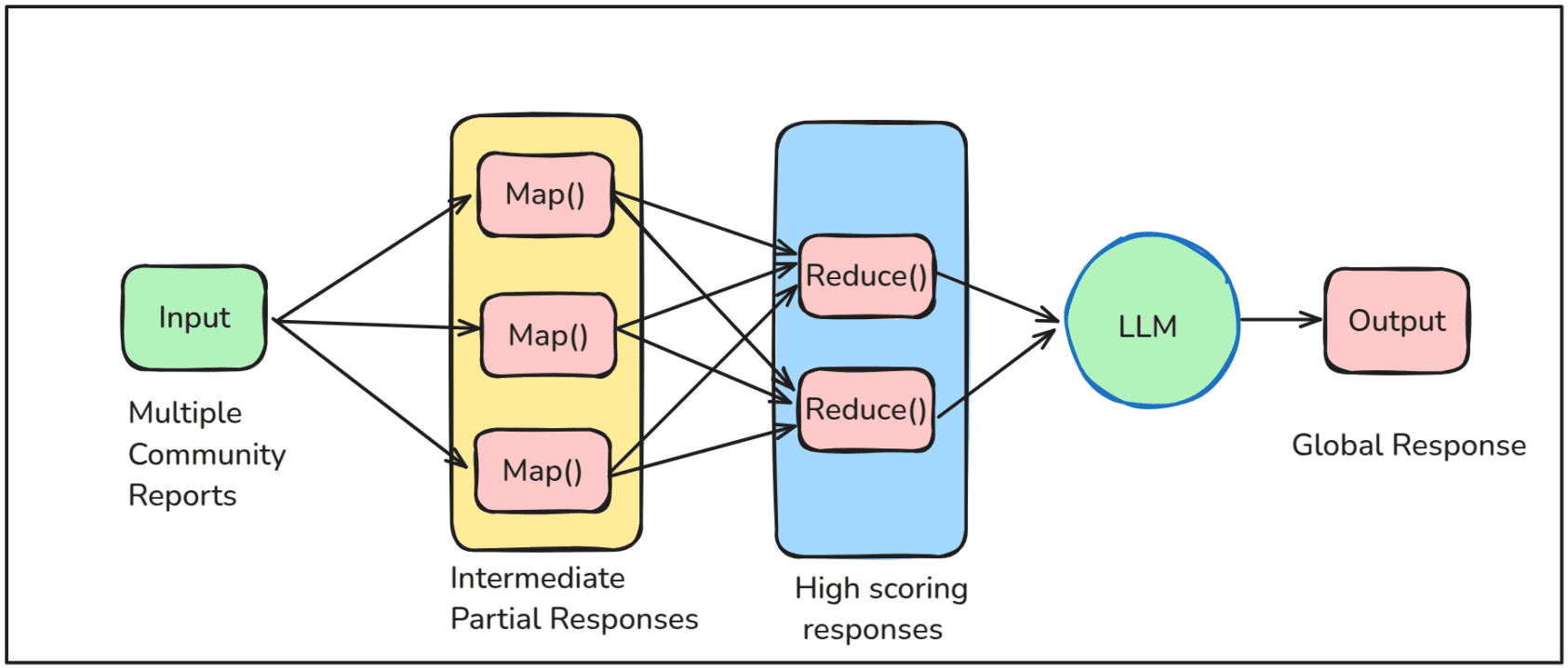
Query Focused Summarization (QFS)

- Identifies patterns and correlations from queries.
- Matches against community summaries.
- Optimized retrieval ensures efficient responses.

YHK

Map Reduce Approach

- **Map Phase**: Generates multiple partial responses with importance scores.
- **Reduce Phase**: Sorts and merges top-scoring responses into a global answer.
- Efficiently filters search space while maintaining accuracy.



(Ref: GraphRAG: The Practical Guide for Cost-Effective Document Analysis with Knowledge Graphs -Jaykumaran)

Benchmarks and Evaluation Metrics

- Traditional RAGAS metrics are inadequate for sensemaking queries.
- LLM judges responses based on:
  - **Comprehensiveness**: Covers all aspects of the query.
  - **Diversity**: Provides varied perspectives and insights.
  - **Empowerment**: Aids decision-making without false assumptions.
  - **Directness**: Clear, concise, and to the point.
- GraphRAG excels in comprehensiveness and diversity.
- Naive RAG is better at generating direct answers.

YHK

Shortcomings of Microsoft GraphRAG

- Multiple LLM API calls make it slow, inefficient, and costly.
- No de-duplication step, resulting in a noisy graph index.
- Upserting new data requires reconstructing the entire KG, making it impractical for production.

YHK

GraphRAG Alternatives and Improvements

- Graph-based RAG is not new; earlier versions by NebulaGraph, Langchain, LlamaIndex, Neo4j.
- Microsoft's approach became widely accepted but has limitations.
- Led to new variants focusing on efficiency and accuracy.

YHK

GraphRAG Variants

- **Nano-GraphRAG**: Introduced lighter, faster versions.
- **Top-k Selection**: Unlike map-reduce, it selects only top-k communities for efficiency.
- Derived versions: LightRAG, MedGraphRAG, FastGraphRAG.
- MedGraphRAG is challenging to adapt to OpenAI endpoints.

YHK

FastGraphRAG Performance

- 27x faster and 40% more accurate retrieval.
- Benchmarks (2wikimultihopQA, 101 queries):

Method	Accuracy (All)	Accuracy (Multihop)	Insertion Time (min)
VectorDB	0.42	0.23	∼0.3
LightRAG	0.45	0.28	∼25
GraphRAG	0.73	0.64	∼40
FastGraphRAG	0.93	0.90	∼1.5

(Ref: GraphRAG: The Practical Guide for Cost-Effective Document Analysis with Knowledge Graphs -Jaykumaran)

YHK

Graph Construction Economy Principle

- Complex graphs require significant computational resources.
- Trade-off: Performance gains must justify resource investment.
- Aim: Maximize performance-to-resource ratio.

YHK

FastGraphRAG at Inference Time

- Uses a PageRank-like algorithm (similar to Google).
- Determines importance of elements in the KG.
- Retrieves only the most relevant entities and relationships.
- Produces high-quality, precise responses.

YHK

Evaluating FastGraphRAG for Your Use Case

- Consider cost-to-performance ratio for your specific dataset.
- Assess viability based on document complexity and enterprise needs.
- FastGraphRAG offers a scalable approach with efficiency gains.

YHK

Conclusion

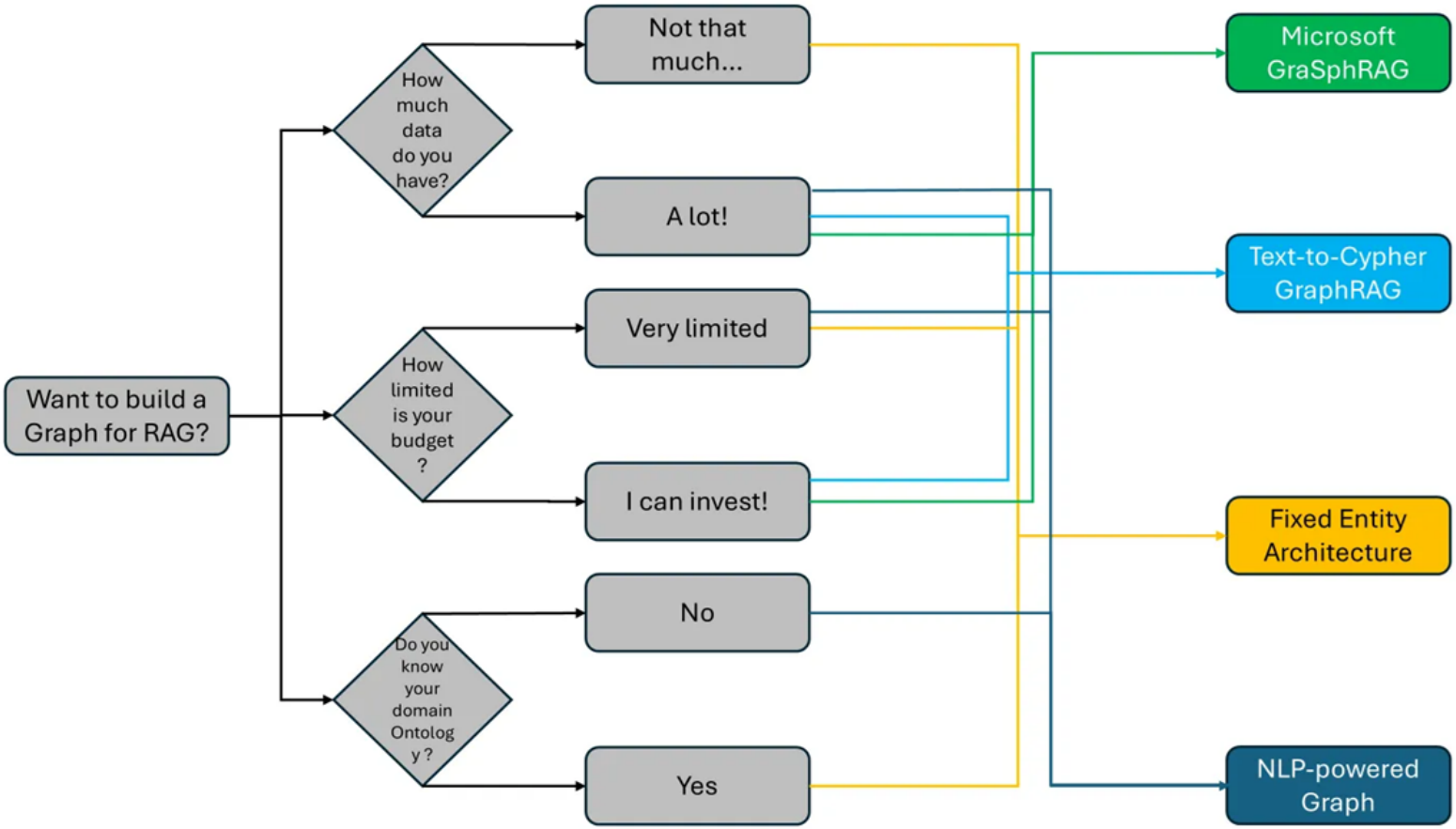
- GraphRAG scales from local to global knowledge representation.
- Community-based summaries improve response quality.
- Enables optimized retrieval with hierarchical partitioning.

YHK

Implementation

YHK

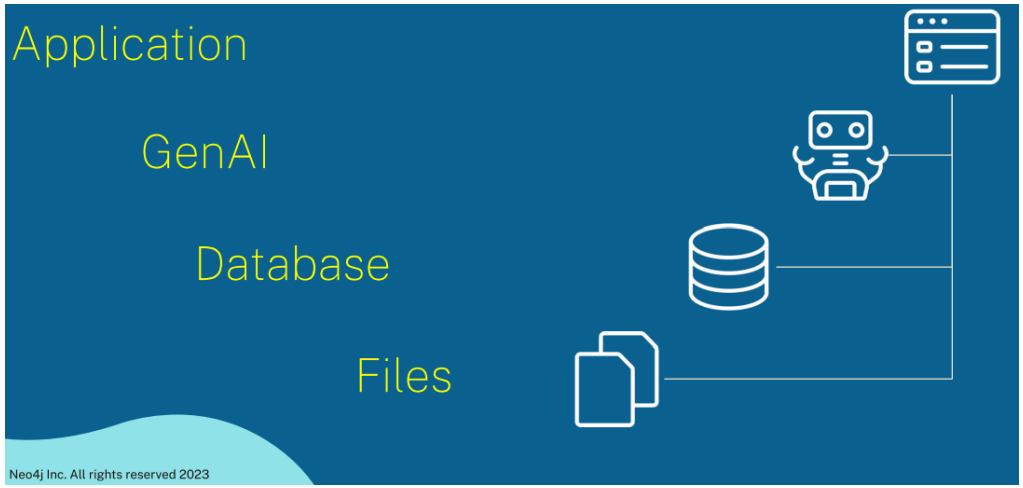
How to build GraphRAG?



(Ref: Build your hybrid-Graph for RAG & GraphRAG applications using the power of NL - Irina Adamchic)

YHK

Neo4j Ollama GraphRAG



(Ref: The GenAI Stack - Andreas Kollegger - Neo4j)

- Application: LangChain: an "orchestration framework" for integrating with LLMs
- Gen AI : Ollama : locally managed language models
- Database: Neo4j: knowledge graph to augment the LLM
- Files: import: sample data sources for constructing a knowledge graph

YHK

Code Walkthrough: Medical Doc Analysis with Fast GraphRAG

(Ref: GraphRAG: The Practical Guide for Cost-Effective Document Analysis with Knowledge Graphs -Jaykumaran)

YHK

FastGraphRAG Overview

- All functions are asynchronous, enabling concurrent API calls.
- Works well with providers without RPM limits (e.g., OpenAI).

YHK

Installation

```
!git clone https://github.com/circlemind-ai/fast-graphrag.git
%cd fast_graphrag
%poetry install

# or
!pip install fast-graphrag
```

Importing Dependencies

```
    import instructor  # Structured outputs for LLMs
2   import os
    from fast_graphrag import GraphRAG, QueryParam
4   from fast_graphrag._llm import OpenAIEmbeddingService, OpenAILLMService
```

YHK

Dataset: MIMIC-IV-ICU

- Public dataset of de-identified ICU patient EHRs.
- Using first 430 samples (650 words per file).

```
mimic_ex_500/
|- report_0.txt
|- report_1.txt
|-  ...
|- report_499.txt
|- report_500.txt
```

YHK

Defining Domain and Entity Types

```
DOMAIN = "Analyze clinical records for key medical entities."
EXAMPLE_QUERIES = [
  "What are the common risk factors for sepsis in ICU patients?",
  "How do trends in lab results correlate with patient outcomes in cases of acute kidney injury?",
  "Describe the sequence of interventions for patients undergoing major cardiac surgery.",
  "How do patient demographics and comorbidities influence treatment decisions in the ICU?",
  "What patterns of medication usage are observed among patients with chronic obstructive pulmonary disease (COPD)?"]
  ENTITY_TYPES = ["Patient", "Diagnosis", "Procedure", "Lab Test", "Medication", "Outcome", "Unknown"]
  working_dir = "./WORKING_DIR/mimic_ex500/"

```

YHK

Configuring the FastGraphRAG Pipeline

```
grag = GraphRAG(
    working_dir=working_dir,
    n_checkpoints=2, # to save backups (recommended)
    domain=DOMAIN,
    example_queries="\n".join(EXAMPLE_QUERIES),
    entity_types=ENTITY_TYPES,
    config=GraphRAG.Config(
        llm_service=OpenAILLMService(
            model="Phi4_6k",  # gemini-2.0-flash-exp
            # or https://generativelanguage.googleapis.com/v1beta/openai/
            base_url="http://localhost:11434/v1",  # Ollama
            api_key="ollama",  # or GEMINI_API_KEY
            mode=instructor.Mode.JSON,
            client="openai"),
        embedding_service=OpenAIEmbeddingService(
            model="mxbai-embed-large",  # mxbai-embed-large
            base_url="http://localhost:11434/v1",
            api_key="ollama",
            embedding_dim=1024,  # mxbai-embed-large - 1024 ; nomic-embed - 768
            client="openai"),
    ),
)
directory_path = "mimic_ex_430" # input dir to the dataset
```

YHK

Graph Indexing

```python
def graph_index(directory_path):
    file_count = 0 # Keep track of processed files.
    for filename in os.listdir(directory_path):
        if filename.endswith('.txt'):
            file_path = os.path.join(directory_path, filename)
            with open(file_path, 'r') as file:
                content = file.read()
                if isinstance(content, list):
                    content = "\n".join(map(str, content))
                if isinstance(content, dict):
                    key_to_use = next(iter(content.keys()), None)
                    content = content[key_to_use] if key_to_use else
     str(content)
                else: content = str(content)
                grag.insert(content)
            file_count += 1
            total_files = sum(1 for f in os.listdir(directory_path) if
     f.endswith(".txt"))
            print("******************** $$$$$$ ******************")
            print(f"Total Files Processed: -> {file_count} / {total_files}")
            print("******************** $$$$$$ ******************")
    return None
graph_index(directory_path)
```

YHK

Indexing Performance

- Indexed 430 items in 510 minutes (8.5 hours).
- Processed locally with a 14B model.

YHK

Querying Example 1: Targeted at Global

```
    grag.query("Describe the sequence of interventions for
    cardiac surgery?").response

**********
The sequence of interventions for the clinical scenario in
    post-aortic valve repair surgery with complications like
    respiratory issues, neurological impairment, and renal
    failure includes:
**Pre-surgery Preparation:**
- Evaluate and stabilize the patient.
**Intraoperative Procedures:**
- Conduct aortic valve replacement or repair.
**Immediate Postoperative Care:**
- Continue invasive mechanical ventilation, especially in
    patients with respiratory issues or neurological
    impairments preoperatively.

```

YHK

Querying Example 1: Targeted at Global

```
**Rehabilitation and Follow-up:**
- Perform swallow evaluation tests to ensure readiness for
    oral intake, followed by a diet progression to soft or
    clear liquids if necessary.

**Commonly Used Medications:**
- Anticoagulants (e.g., heparin) to prevent blood clots.

 A multidisciplinary team including cardiologists,
    pulmonologists, and other specialists is often involved
    to address any comorbid conditions.
**********************************************

# o3-mini-high rated this response : 8/10.
```

YHK

Querying Example 2: Targeted at Global

```
    grag.query("How do demographics and comorbidities
    influence ICU decisions?").response

**********************************************
Patient demographics and comorbidities significantly
    influence treatment decisions in the ICU, as evidenced by
    detailed hospital courses described in various scenarios.

1. **Demographics**: Specific ages or life stages, such as
    "on day of life 45" for a patient with unique medical
    conditions (metabolic issues), indicate tailored care
    plans sensitive to developmental stages.

2. **Comorbidities**:
   - A history like short gut syndrome and colectomy
    influences decisions about when and how to restart total
    parenteral nutrition (TPN). \ldots .

```

YHK

Querying Example 2: Targeted at Global

```
3. **Interactions**: Decisions like replacing a PICC line
    with a Hickman catheter after infection shows proactive
    steps in avoiding repeat complications due to prior
    infections from central lines.
4. **Treatment Customization**: The management decisions,
    such as adjusting insulin administration for endocrine
    challenges even when insulin levels are normal,
    demonstrate personalized ICU therapy adapting for nuanced
    metabolic control in conditions like hypoglycemia of
    unknown origin.

These scenarios highlight that ICU treatment is deeply
    influenced by a patient's demographic specifics \ldots
    and chronic health issues effectively.
**********************************************
# o3-mini-high rated this response : 8/10.

```

YHK

Querying Example 3: Targeted at Global

```
    grag.query("How can lung function be improved in COPD +
    Heart Failure?").response

**********************************************
To optimize the treatment for patients with both COPD and
    heart failure, it is crucial to manage fluids carefully
    using diuretics without causing respiratory issues. This
    necessitates precise adjustments in medication dosages,
    closely monitoring fluid balance, and ensuring proper
    oxygenation levels. Right heart catheterization can
    assist in measuring cardiac function and tailoring
    treatments for these patients. The goal is to alleviate
    symptoms of both conditions while preventing
    exacerbations, which often involves using supplemental
    oxygen or bronchodilators as required.
**********************************************
# o3-mini-high rated this response : 8/10.
```

YHK

Querying Example 4: Targeted at Local

```
    grag.query("Discuss Hypocapnic and Hypoxemic Respiratory
    Failure").response

Hypocapnic Respiratory Failure and Hypoxemic Respiratory
    Failure are two distinct categories of respiratory
    failure, each characterized by different clinical
    features. Conversely,\ldots Accurate assessment and
    differentiation between these conditions are crucial for
    appropriate management and treatment.

In medical practice, both types may coexist or require
    different clinical interventions based on the specific
    underlying pathology.
**********************************************
# o3-mini-high rated this response : 6/10.
```

Limitations of FastGraphRAG

- Asynchronous and Pydantic-validated, challenging for beginners.
- 7B-class models struggle with structured responses.
- Limited documentation and community support.
- LightRAG offers better support for local LLMs.
- Faster than GraphRAG/LightRAG but slower than vector-only RAG.
- Best for large datasets and high-stakes domains.
- For simple use cases, vector-only RAG is recommended.

YHK

Trivia: PDF Indexing Challenges

- Faced issues with PDF indexing using pdf-plumber.
- Pydantic validation errors encountered.
- Works fine with .txt files converted from PDFs.
- Did not extensively debug the issue.

YHK

Conclusion

- GraphRAG enables tracking patient behavioral patterns.
- Supports complex, multi-hop queries with accurate responses.
- Ideal for high-stakes domains requiring precision.
- Knowledge graphs improve reasoning and interpretability.
- Hybrid approaches (KG + vectors) outperform naive RAG.

YHK

References

Slides primarily borrowed from . . .

- End-to-End Implementation of GraphRAG(Knowledge Graphs + Retrieval Augmented Generation): Unlocking LLM Discovery on Narrative Private Data - Vinod Kumar G R
- GraphRAG tutorials by Data Science in your pocket
- GraphRAG: Enhancing LLMs with Knowledge Graphs - Vijayakumar Ramdoss
- Awesome-GraphRAG (GraphRAG Survey) https://github.com/DEEP-PolyU/Awesome-GraphRAG
- Build your hybrid-Graph for RAG & GraphRAG applications using the power of NL - Irina Adamchic
- The GenAI Stack - Andreas Kollegger - Neo4j
- GraphRAG: The Practical Guide for Cost-Effective Document Analysis with Knowledge Graphs JaykumaranJaykumaran

YHK

Thanks . . .

- Search **"Yogesh Haribhau Kulkarni"** on Google and follow me on LinkedIn and Medium
- Office Hours: Saturdays, 2 to 5pm (IST); Free-Open to all; email for appointment.
- Email: yogeshkulkarni at yahoo dot com

(https://www.linkedin.com/in/yogeshkulkarni/, QR by Hugging Face QR-code-AI-art-generator, with prompt as "Follow me")

YHK