

CSE/DSC 234

Data-Centric AI and AI Engineering

Arun Kumar

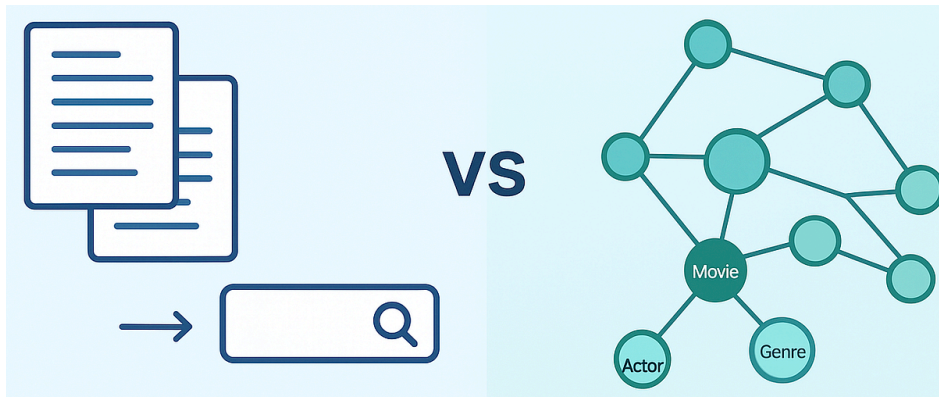
Topic 2: Retrieval and Context Engineering

Outline

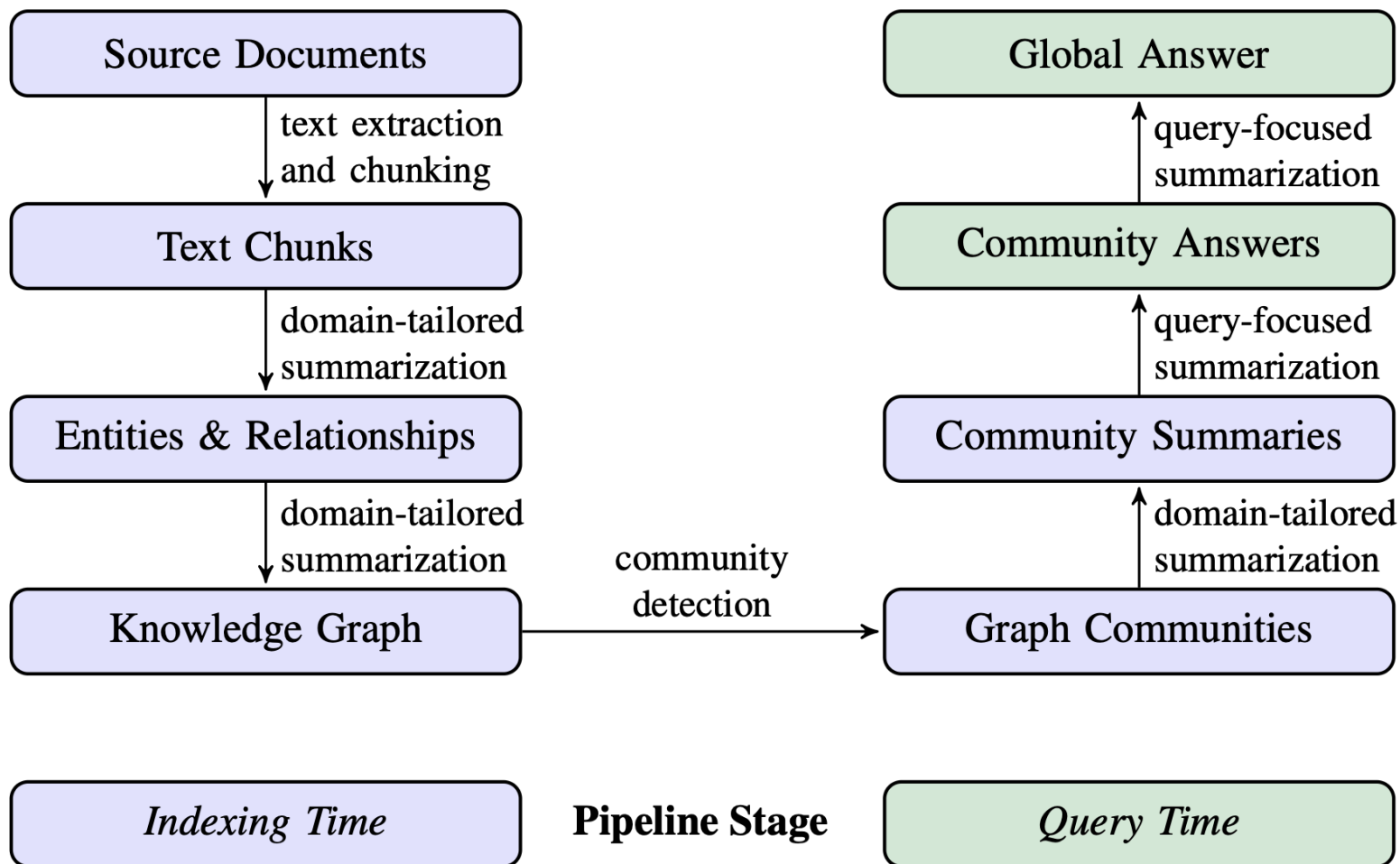
- ❖ Introduction
- ❖ Prompt Engineering
- ❖ Retrieval and Context Engineering
 - ❖ Retrieval vs. Long Context
 - ❖ VectorRAG
 -  ❖ GraphRAG
 - ❖ Other Forms of RAG

Motivation for GraphRAG

- ❖ VectorRAG retrieves top-K passages by semantic similarity
 - ❖ Works when the answer lives in a few specific chunks but fails when no single chunk contains the answer!
- ❖ Two main failure modes:
 - ❖ **Global Sensemaking**, e.g., “What are the main themes?”
 - ❖ **Multi-hop Reasoning**, e.g., “How does X connect to Y via Z?”
- ❖ Retrieval type queries insufficient; need summarization type too



The GraphRAG Approach



Example Queries for GraphRAG

❖ Corpus: RapidFire AI documentation as in Project 1

Local

"What does the `setup_retriever()` function do?"

Answer lives in one docstring chunk

Global

"What design principles guide the RapidFire AI API?"

No single chunk states the principles

Needs theme
aggregation

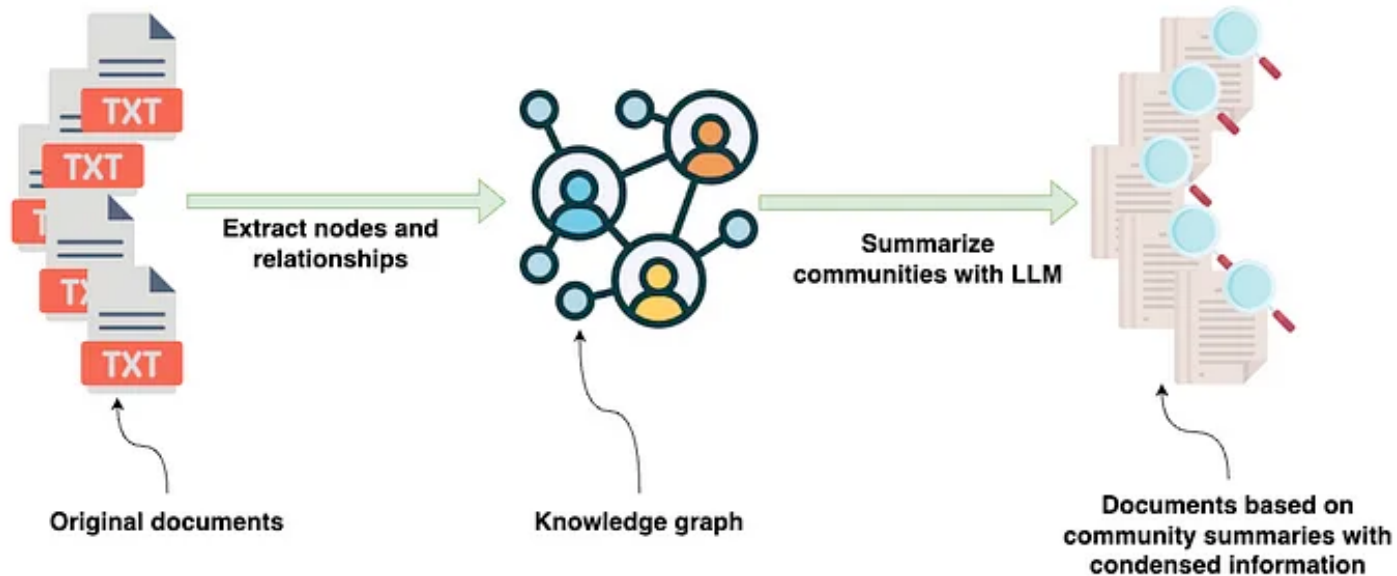
Multi-hop

"Which config flags affect the scheduler,
and which examples demonstrate them?"

flag → scheduler module → example notebooks

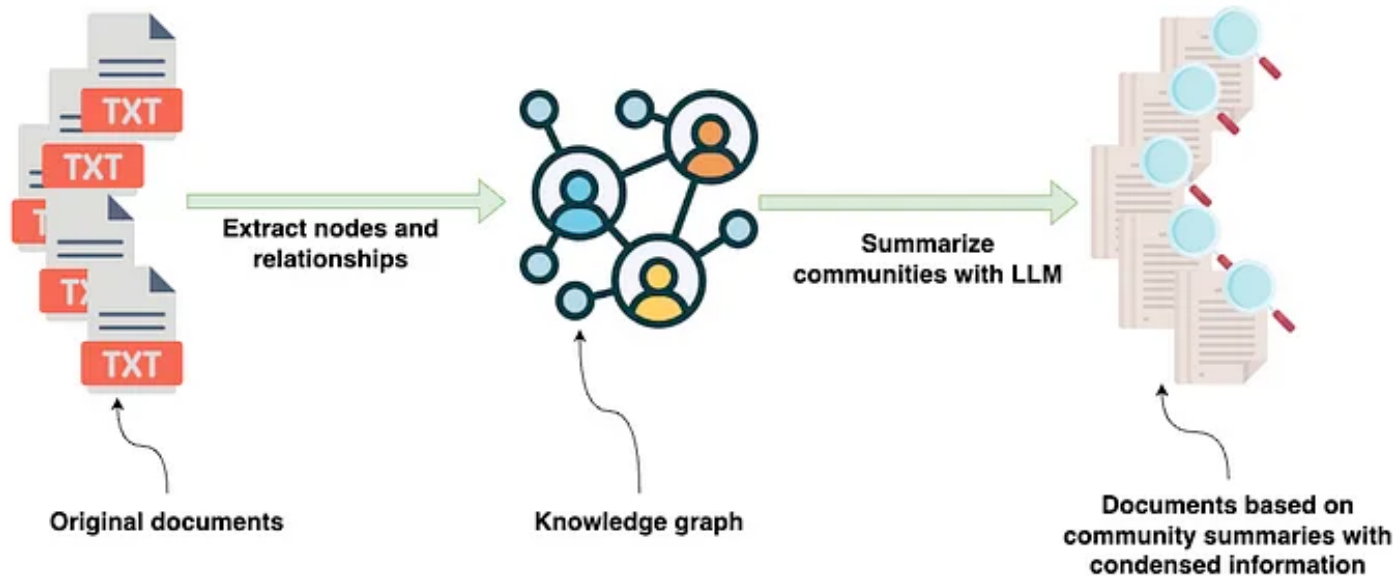
Needs entity
traversal

Graph Construction Stage



- ❖ Knowledge Graph (KG) / Knowledge Base (KB) construction is well-studied in NLP+DB intersection
 - ❖ Need notions of **entities** and **relationships** (an ontology)
 - ❖ SOTA is to use LLMs to convert corpus into KG/KB
- ❖ Expensive but one-time “indexing” stage; stored in Graph DBMS

Summarization Stage



- ❖ Leiden algorithm for community detection (semantic clustering)
 - ❖ Graph-oriented and preserves semantic coherence
 - ❖ LLM produces one summary per community at each level
- ❖ At query time need a “MapReduce” computation:
 - ❖ MAP: Each summary answers query + emits usefulness score
 - ❖ REDUCE: Merge answers, weighted by usefulness

GraphRAG in Practice

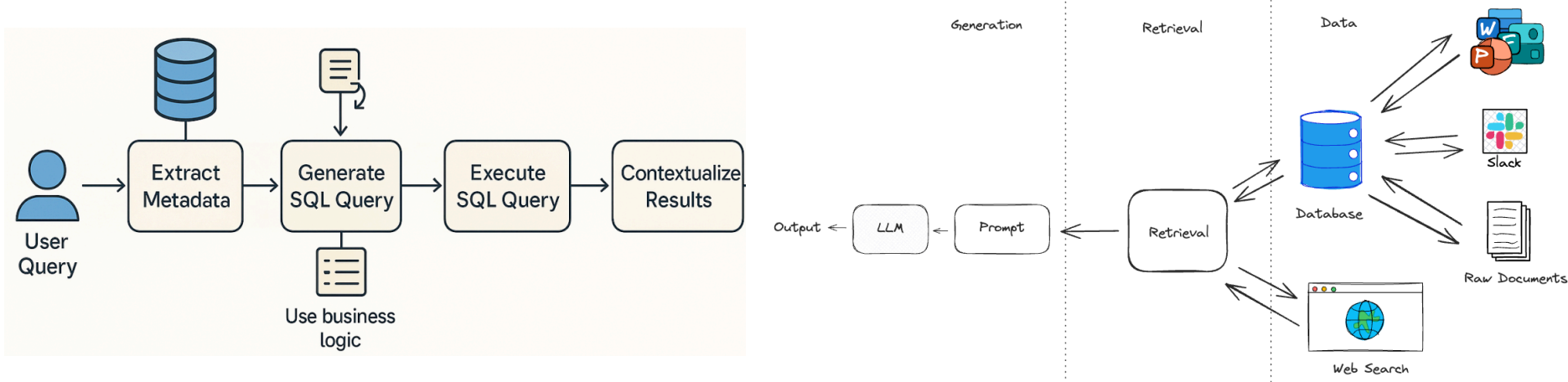
- ❖ GraphRAG is not only for global queries; also has **Local Mode**:
 - ❖ Seed graph search with entities in query
 - ❖ Expand N hops on edges, gather neighbor text + relationships
 - ❖ Useful for multi-hop and entity-centric queries
 - ❖ **Example:** *“Which config flags affect the scheduler?”*
 - ❖ Seed is “scheduler”
 - ❖ 1-hop expansion reaches per-flag neighbor node
- ❖ In production, systems often route queries across multiple retrieval modes rather than picking just one: VectorRAG, GraphRAG global mode, GraphRAG local mode, BM25, etc.

Outline

- ❖ Introduction
- ❖ Prompt Engineering
- ❖ Retrieval and Context Engineering
 - ❖ Retrieval vs. Long Context
 - ❖ VectorRAG
 - ❖ GraphRAG
- ➔ ❖ Other Forms of RAG

Other Forms of RAG

- ❖ “Retrieval” can be via any mechanism, not just vectors/graphs
- ❖ Other popular forms in use today:
 - ❖ Structured Retrieval: SQL, HTML/Wiki Navigation
 - ❖ Web Search
 - ❖ Tool API Calls, Model Context Protocol (more in Topic 3)



Peer Instruction Activity

(Switch to Google Form)