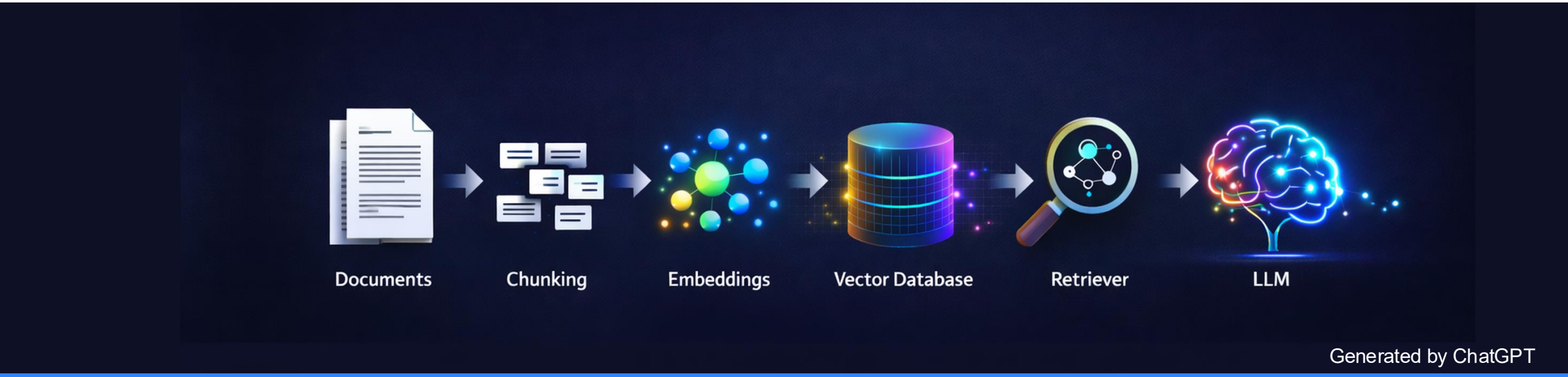


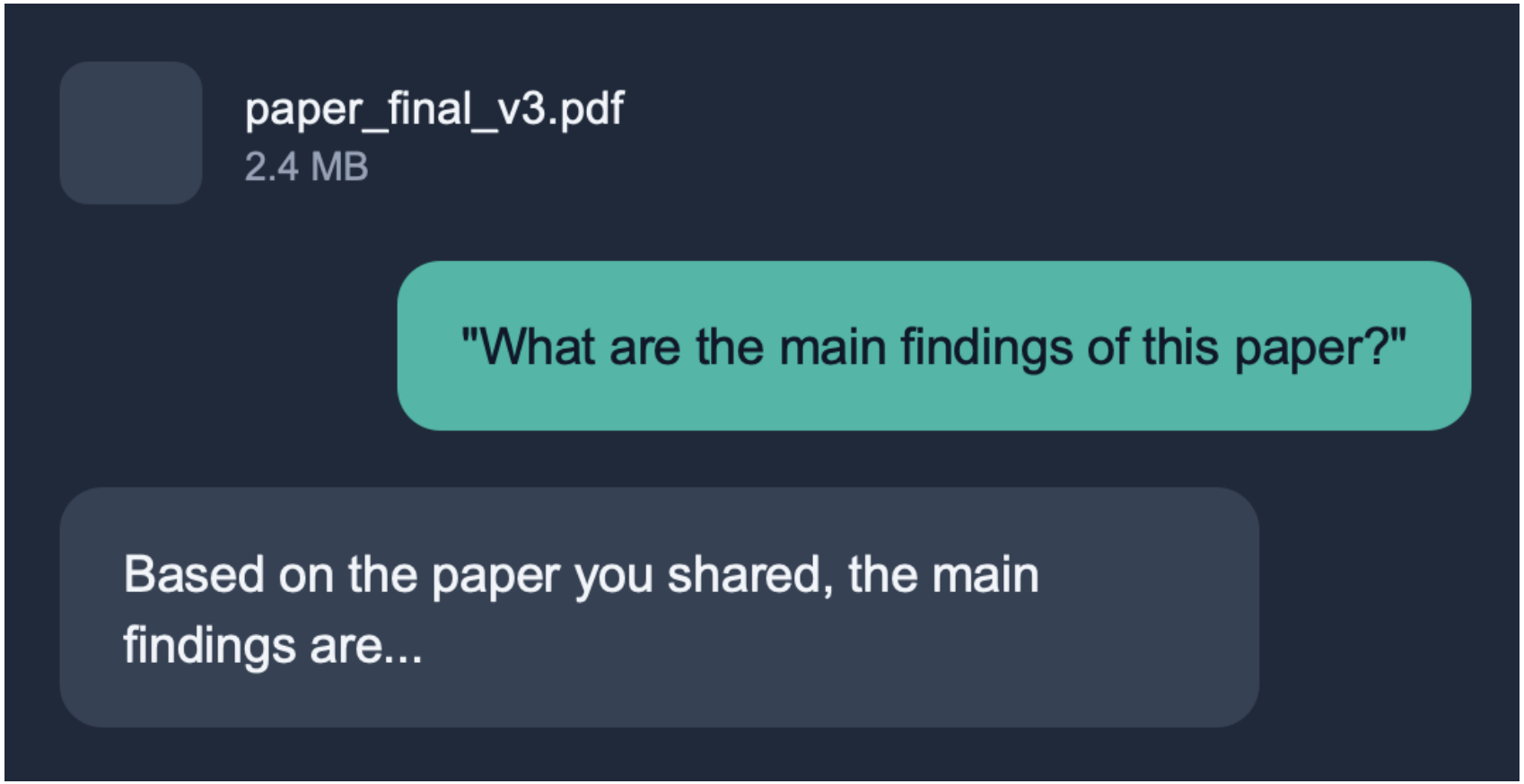
Generated by ChatGPT

Tutorial: Retrieval Augmented Generation

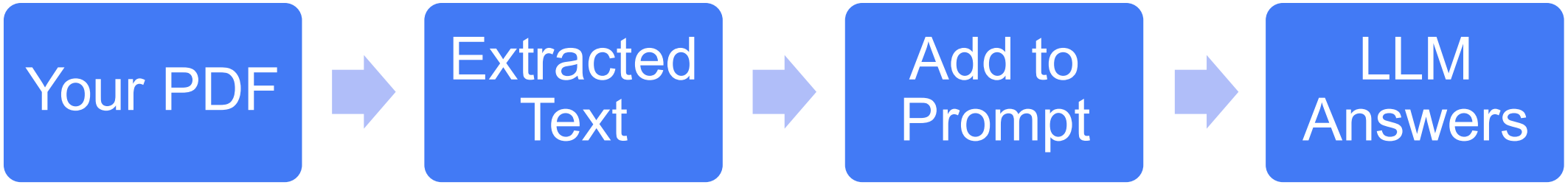
From dropping PDFs into ChatGPT to building your own system

Jesse Grootjen

Have you ever done this?



What's happening behind the scenes?

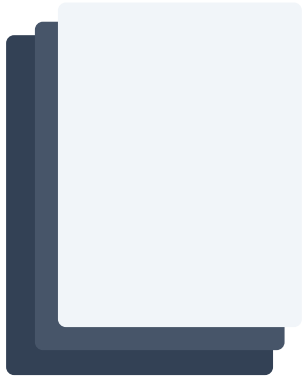


- The entire document is stuffed into the **context window** along with your question.
- The LLM reads everything and generates an answer based on what it found.

Simple and effective... but does it scale?

The problems with this approach?

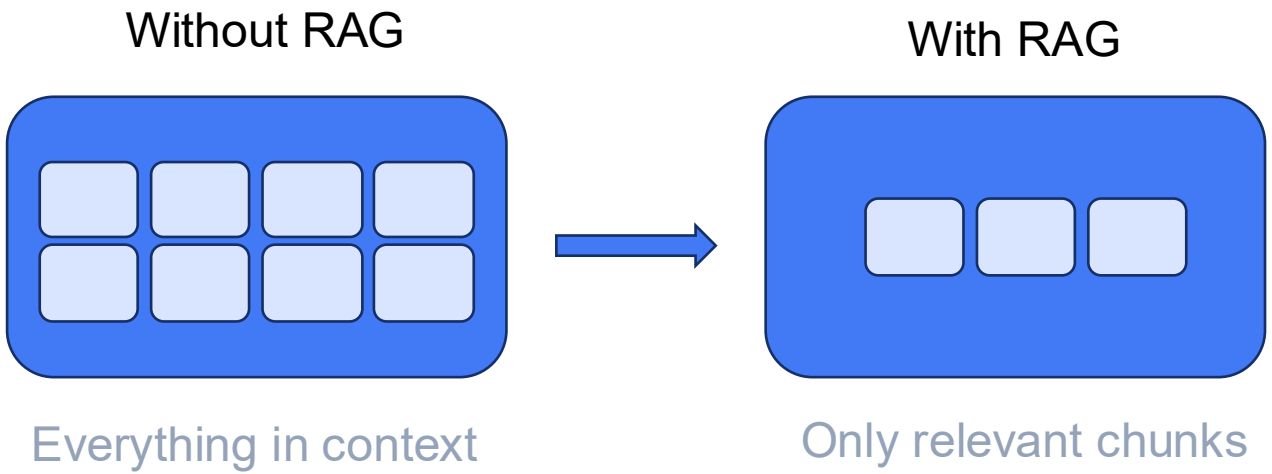
- Context Window Limits
- Cost & Latency
- Lost in the Middle
- “Hallucination”



Your document library doesn't fit
in the context window

The solution:
Retrieve first, then generate

- Instead of stuffing everything into the context, **find the relevant pieces first.**



Retrieval

Find relevant information in your documents

Augmented

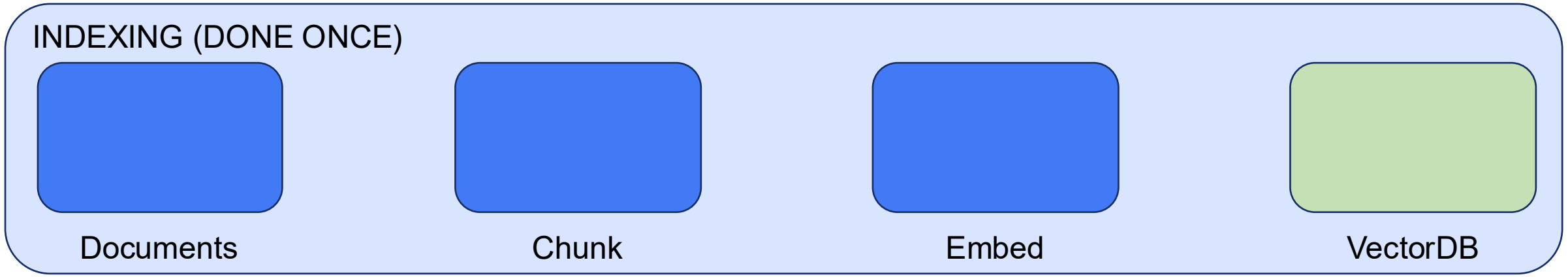
Add that information to the LLMs prompt

Generation

Generate an answer grounded in the context

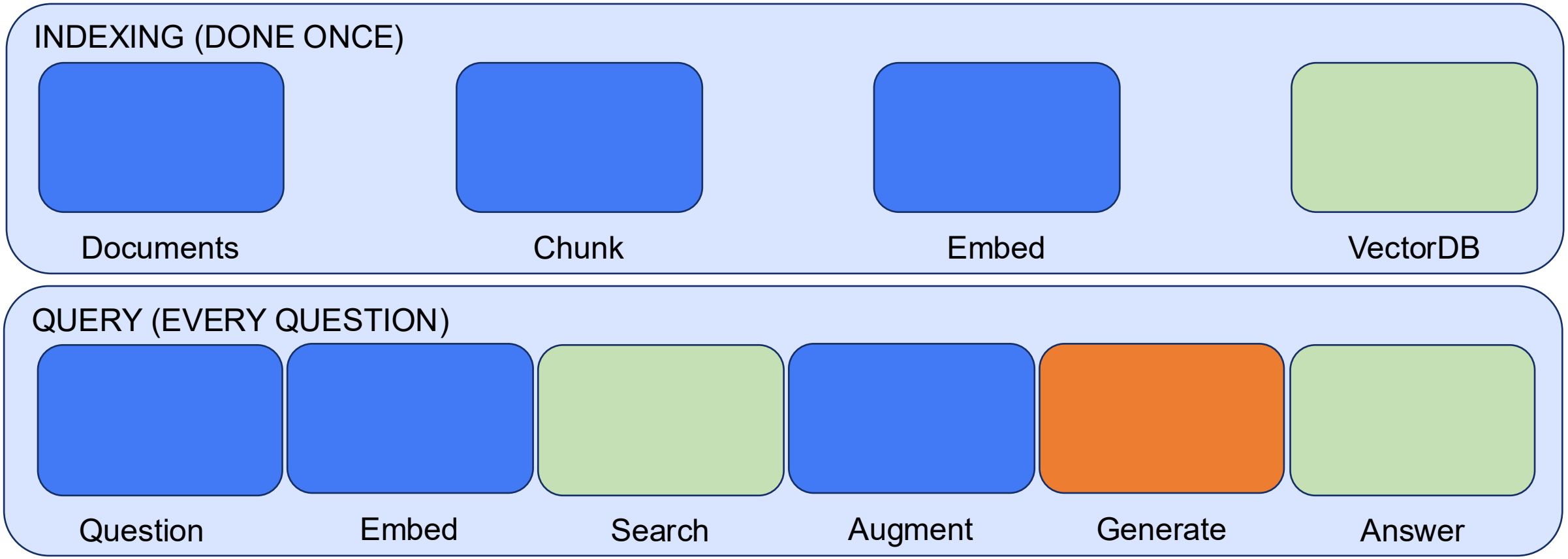
A technique that combines **information retrieval** with **language model generation** to produce answers grounded in specific documents.

The RAG Pipeline



Storage/retrieval components

The RAG Pipeline



Step 1: Chunking

- Documents are split into smaller pieces, typically **200-1000 characters** each.

Why Chunks?

- Smaller units = more precise retrieval
- Each chunk can be independently matched
- Only relevant pieces enter the context

Trade-off

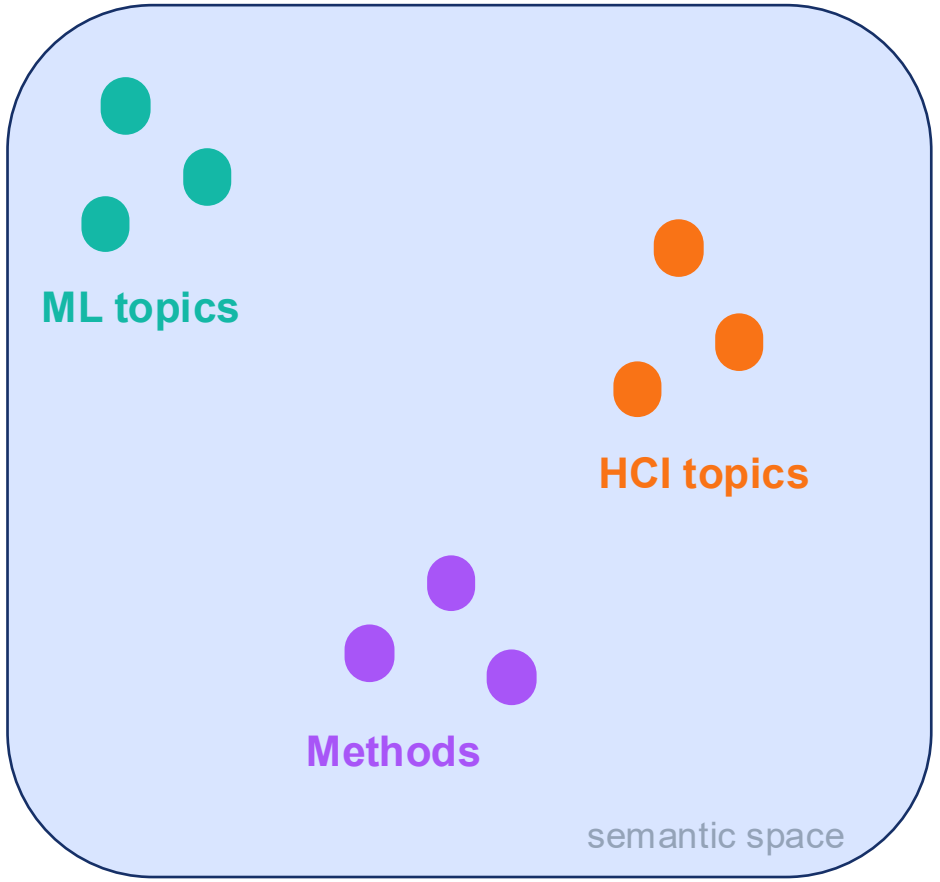
- Too small = loses context.
- Too big = noisy retrieval.

One document → Many chunks

Step 2: Embedding

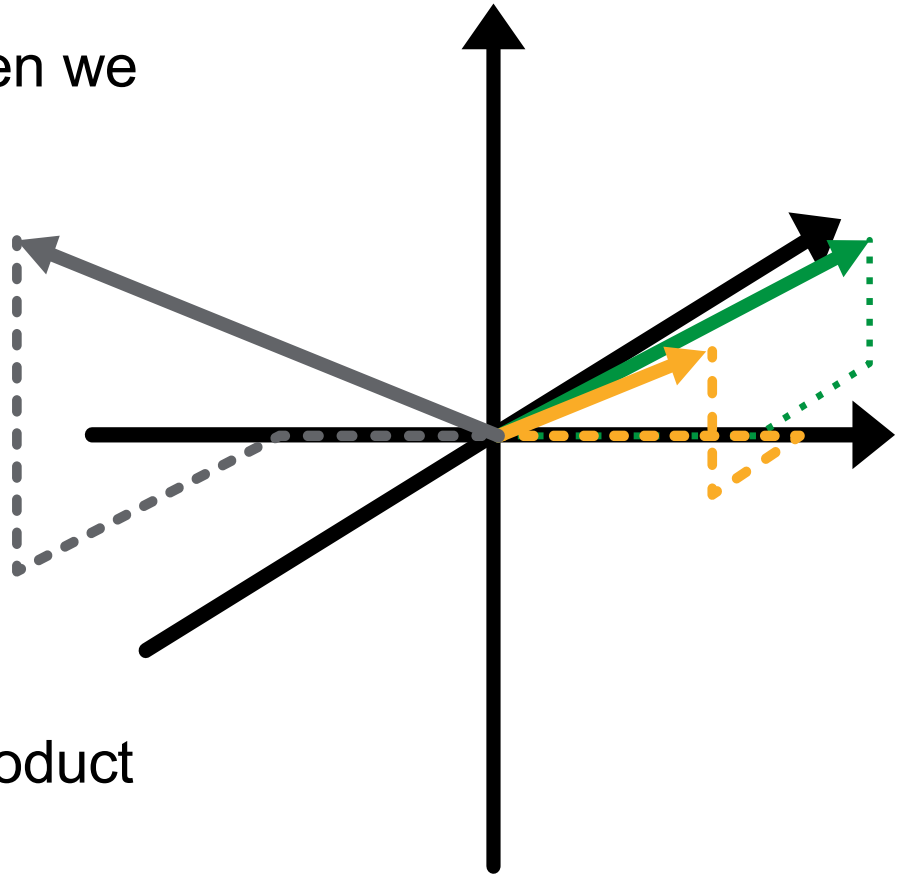
- An **embedding model** converts text into a list of numbers (a vector).

- These vectors capture **semantic meaning**. Similar concepts end up with similar vectors.



Step 3: Retrieval

- When you ask a question, it gets embedded too. Then we find the **nearest neighbors** in vector space.
  - Embed the user's question
  - Compare to all stored chunk vectors
  - Return top-k most similar chunks



- **Similarity metric:** Usually cosine similarity or dot product between vectors.

Source: https://iui-lecture.org

Step 4: Augment & Generate

- Combine the retrieved chunks with the question, then send to the LLM.

- **PROMPT TO LLM:**
  - Context from your documents:
  - [Chunk 1] "The study employed a mixed-methods approach combining eye-tracking data with..."
  - [Chunk 2] "Results showed significant differences in fixation patterns between groups..."
  - [Chunk 3] "The methodology section describes the calibration procedure for..."
- Question: "What methodology did this study use?"

Grounded
LLM answers based on your actual documents

Traceable
You can see which chunks were used

Efficient
Only relevant content in context

Why does this matter for Researchers?

Literature Review
Query hundreds of papers at once. Find related work, compare methods, identify gaps.

Lab Documentation
Build a searchable knowledge base from protocols, notes, and past experiments.

Grant Writing
Quickly find supporting evidence and citations from your collected papers.

✓ **Cite your sources**

✓ **Reduce hallucinations**

✓ **Keep it private**

When RAG shines vs. struggles

✓ RAG works well for

- Factual questions with clear answers in the docs
- Lookup queries — "What does paper X say about Y?"
- Domain-specific knowledge not in LLM training
- Questions where sources matter

✗ RAG struggles with

- Synthesis across many docs — "Compare all 50 papers"
- Vague queries — "Tell me something interesting"
- Multi-hop reasoning — connecting distant facts
- Tables, figures, and images (without special handling)

Let’s Build

Our Stack

- **Ollama** (Local LLM runtime (Llama 3.1))
- **HuggingFace Embeddings** (sentence-transformers for vectors)
- **ChromaDB** (Local vector database)
- **LlamaIndex** (RAG orchestration framework)

Your Task

- Add PDFs to the documents folder
- Run the notebook to index them
- Experiment with different queries
- Observe retrieval quality & failures

  - What made the difference between good and bad RAG responses?
  - When did the system fail to use the context properly?
  - How might you evaluate RAG quality systematically?
  - What would you change about the chunking strategy?
  - When might RAG NOT be the right approach?
  - What happens when you have contradicting documents in your folder?

License