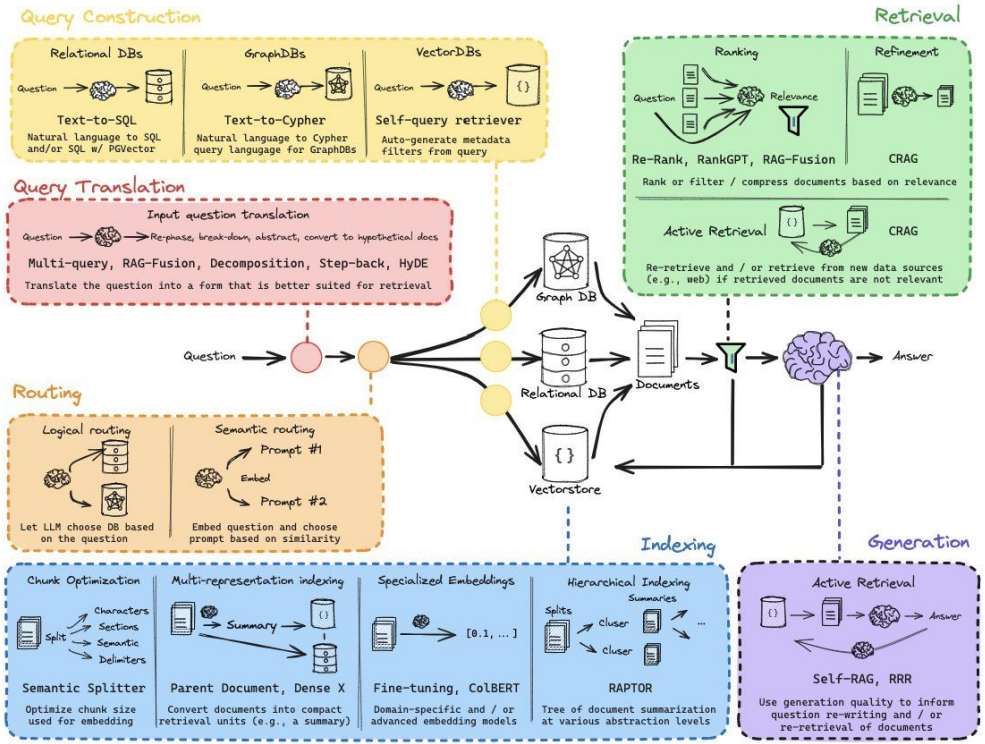
LangChain

Unifying RAG and long context LLMs

Lance Martin
Software Engineer, LangChain
@RLanceMartin

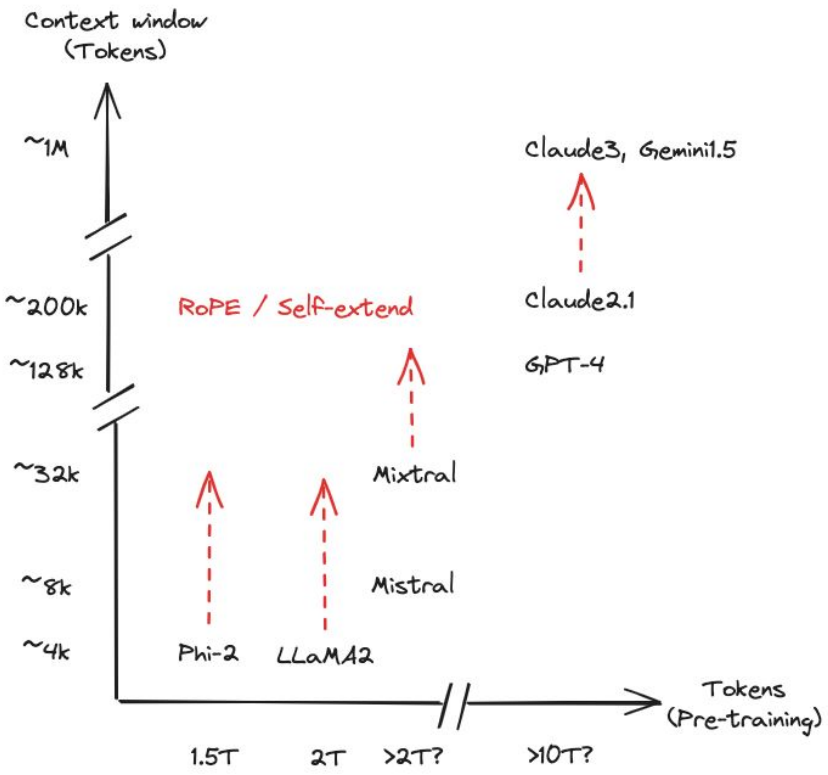
LangChain

Preface: RAG Course



https://github.com/langchain-ai/rag-from-scratch

Context windows are getting larger



Do we need RAG anymore?

Shaun.AGI
@agishaun

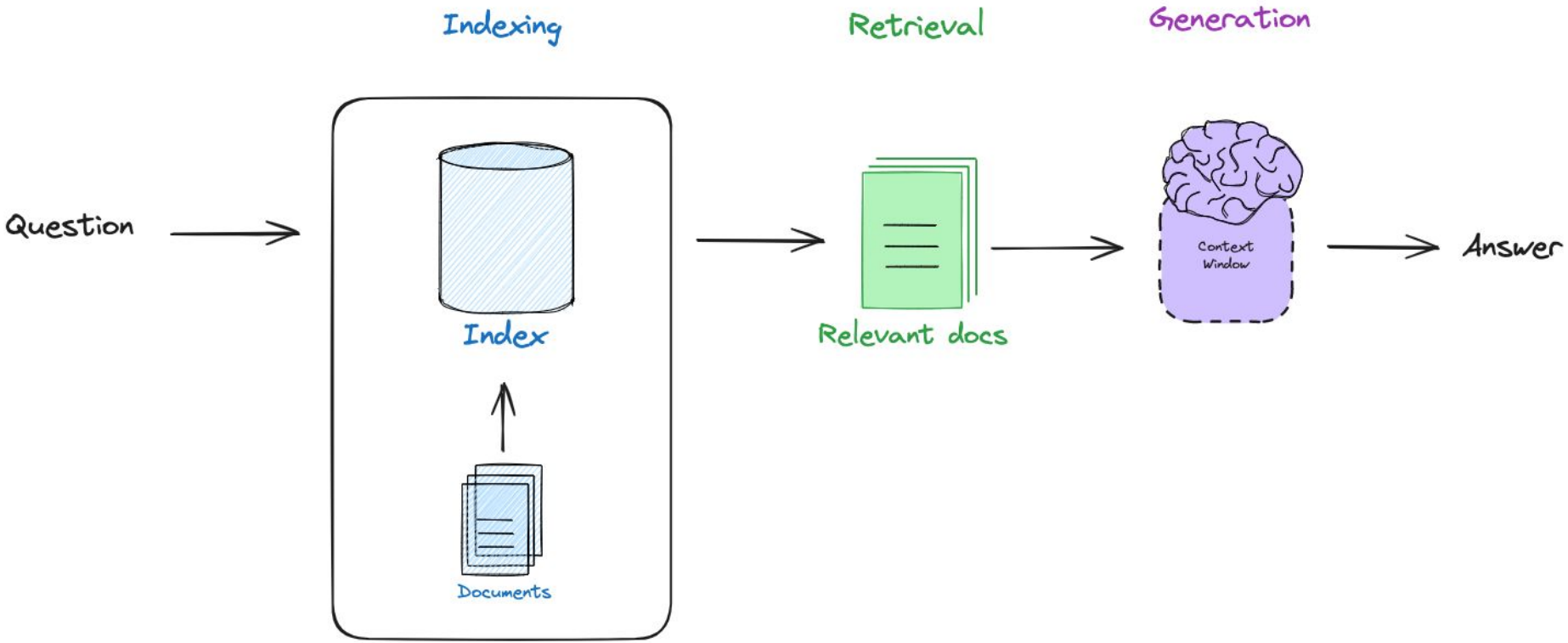
⚠️ RAG might be dead, after reading 58 pages of Genimi 1.5 Pro tech report. Here's my thoughts as AI founder,

1. Simple RAG system like similarity search with vector db will be dead. But more customized RAG will still live. The goal of RAG is mostly on retrieval relevant information. After reading the report, I am convinced LLM can do retrieval really really well.
2. RAG itself may not be dead totally, but 90% of people won't need it anymore. Most dataset can fit in 1M tokens. Just like OpenAI's assistant API, once Gemini API can handle large files, the only thing matters is the cost. However based on the report, 1.5 Pro's training cost and inference cost is much much lower than Gemini 1.0 📉

https://twitter.com/agishaun/status/1758561862764122191

LangChain

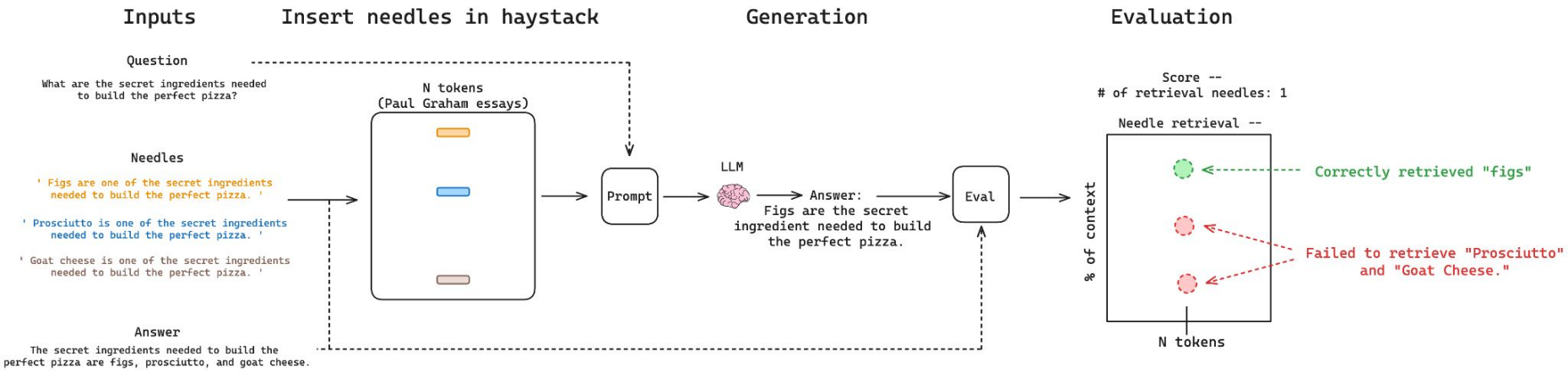
RAG: reasoning & retrieval on multiple chunks of information



https://www.youtube.com/watch?v=wd7TZ4w1mSw

LangChain

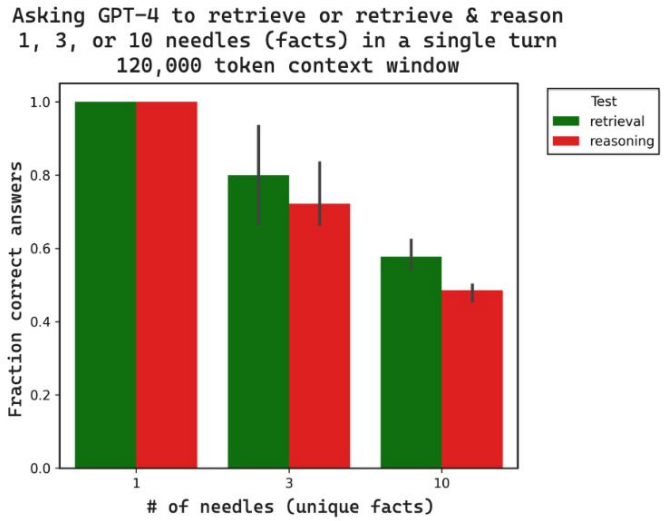
Needle In A Haystack: test reasoning & retrieval in long context LLMs

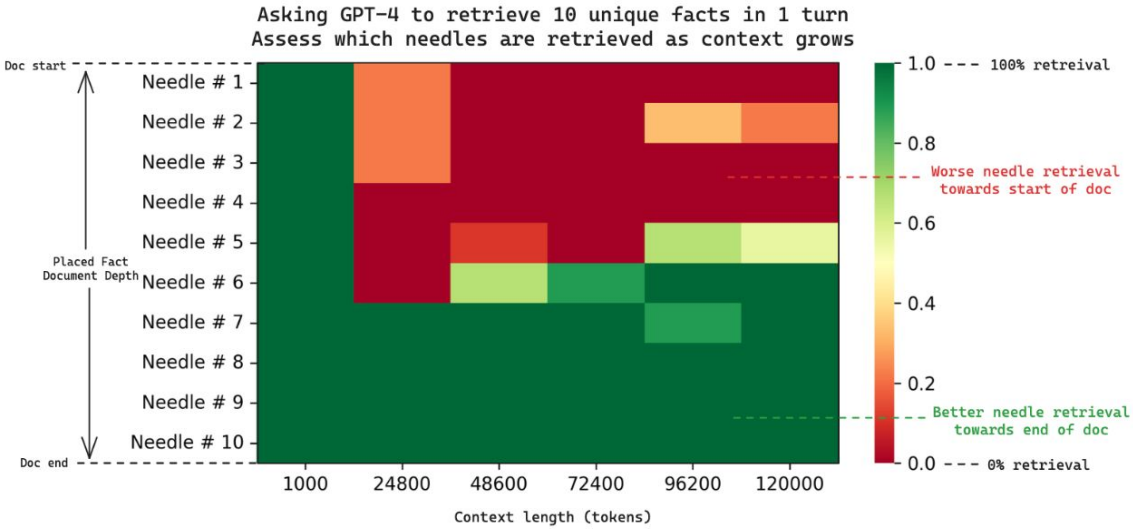


https://github.com/gkamradt/LLMTest_NeedleInAHaystack

LangChain

Retrieval is not guaranteed, reasoning harder than retrieval





https://youtu.be/UlmyyYQGhzc
https://blog.langchain.dev/multi-needle-in-a-haystack/

Challenge may be recency bias in LLMs

A likely culprit for this phenomenon is a mismatch between the task LLMs are trained on and context-augmented generation tasks. Among the documents typically used to pre-train LLMs such as web pages, books, articles and code, the most informative tokens for predicting a particular token are typically the most recent ones. During pre-training, this induces a learned bias to attend to recent tokens. In addition, the rotary positional embedding (RoPE) scheme used in the open source models we investigate has an inductive bias towards reduced attention at long distances [27] that may make it even easier for these models to learn to attend preferentially to recent tokens. Extreme recency bias is not a good prior for context augmented generation tasks where far away tokens may, in fact, contain very relevant information.

https://arxiv.org/pdf/2310.01427.pdf

(1) Be wary of context stuffing. No retrieval guarantees.

(2) Single needle may be misleadingly easy (no reasoning, only 1 fact)

Thomas Ahle
@thomasahle

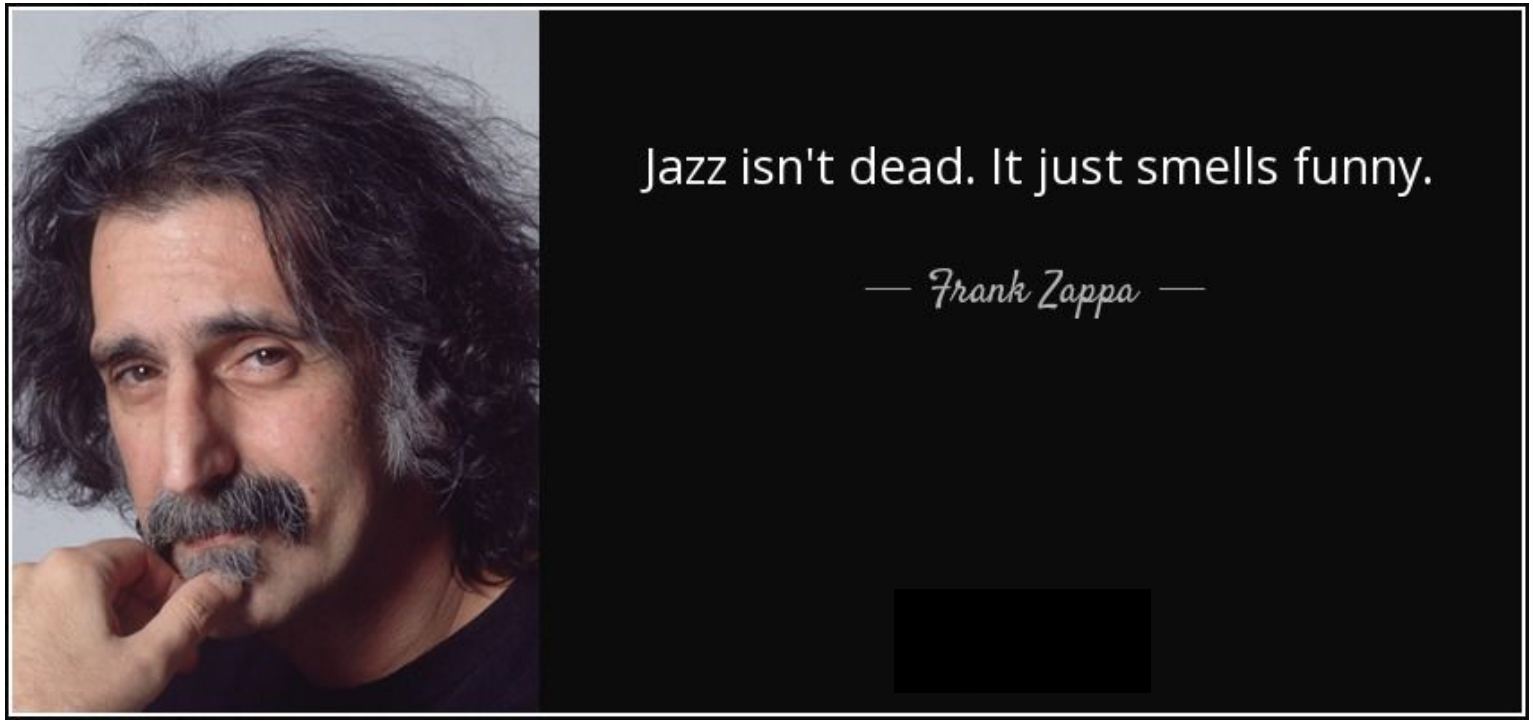
Needle In A Haystack tests are flawed.

Did you know that the Long-Context Attention in Gemini and GPT-4 is based on inserting the sentence *"The best thing to do in San Francisco is eat a sandwich and sit in Dolores Park on a sunny day"* at a random location in a text?

We've seen Claude even **anticipating** that it was being tested because of the awkwardness of the test: twitter.com/alexalbert__/s.... The models know to keep track of the sentence before they are even prompted to retrieve it!

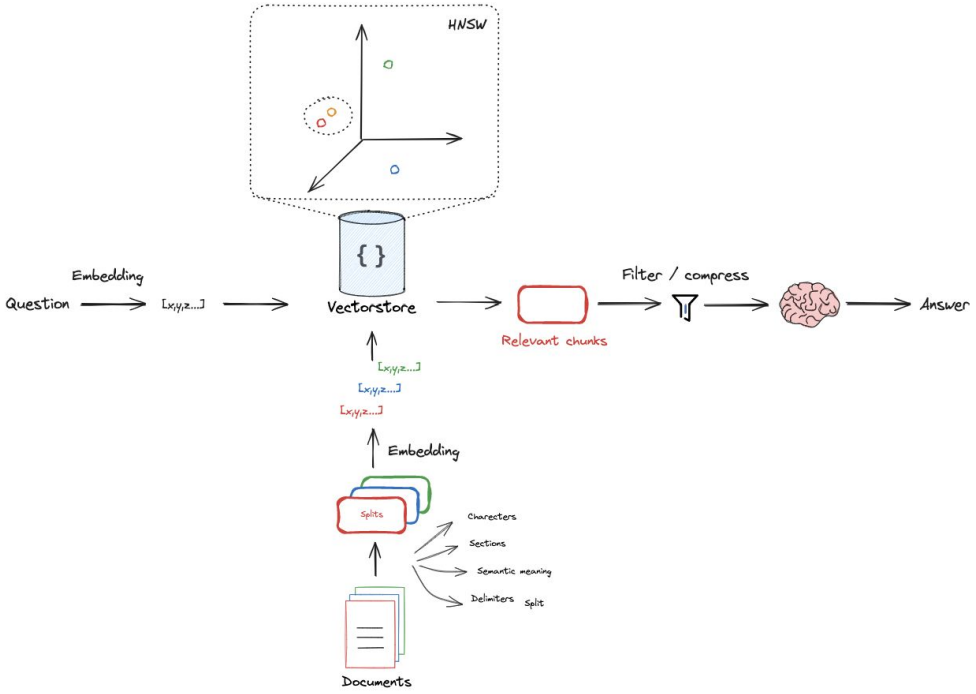
New tests from @NormalComputing demonstrate that models do much worse **if the inserted sentence is subtle**. Like changing the name of one historical person to another.

This will obviously get better and RAG will change



LangChain

RAG today focused on **precise retrieval** of relevant doc **chunks**



https://github.com/langchain-ai/rag-from-scratch

LangChain

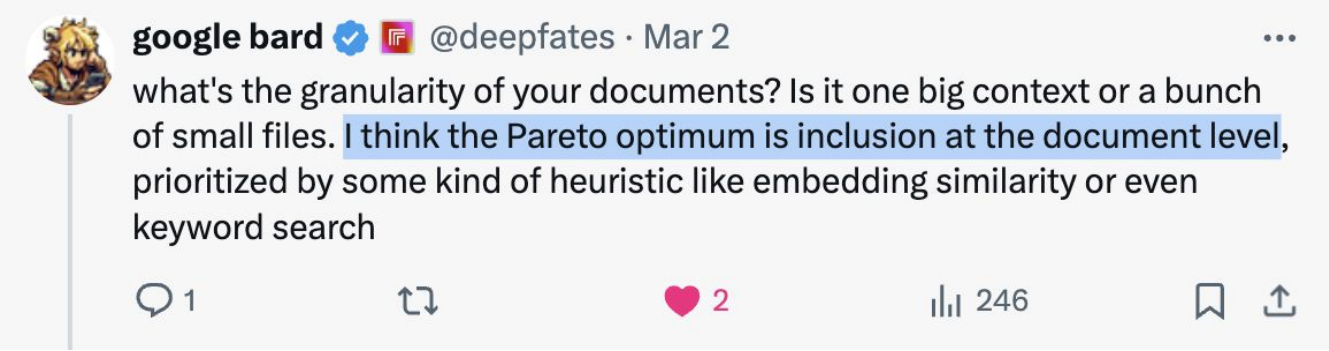
Need to balance **system complexity** vs **latency & token usage**

You need the exact relevant chunk
- risk of over-engineering
- higher complexity
- can suffer lower recall
- sensitivity to chunk size, k, etc

Pareto Optimal

Just throw all your docs into context
- higher latecy
- higher token usage
- can't audit retrieval
- security / auth

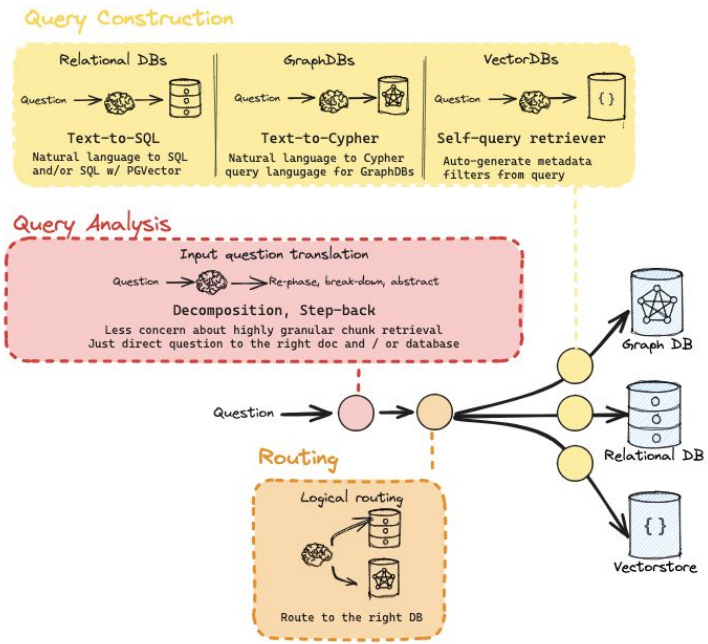
Some ideas …



https://x.com/RLanceMartin/status/1764018419618484325?s=20

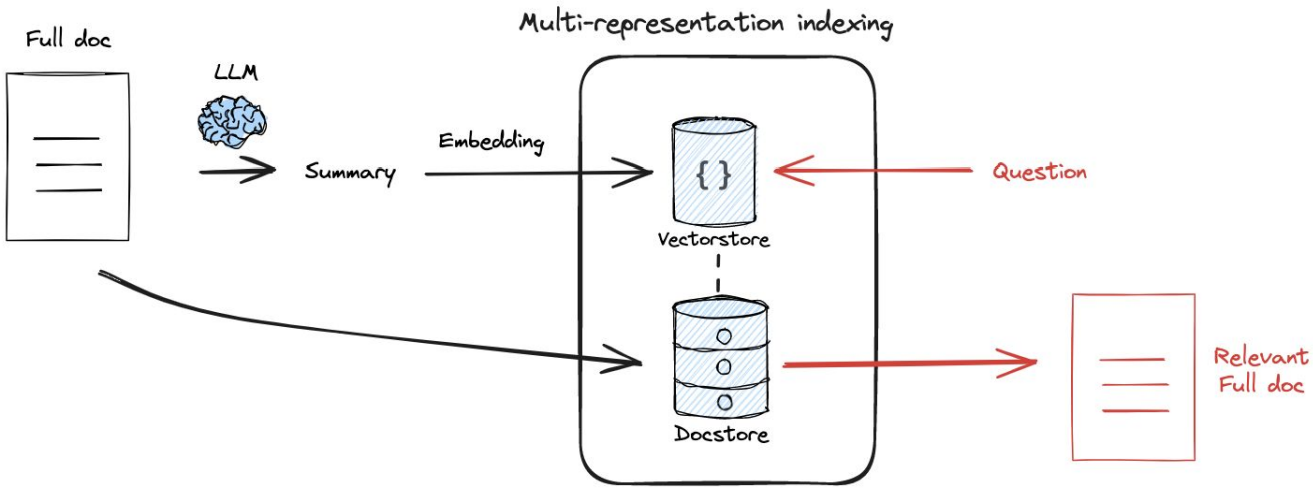
LangChain

Query analysis: Connect questions to the right document



https://python.langchain.com/docs/use_cases/query_analysis/

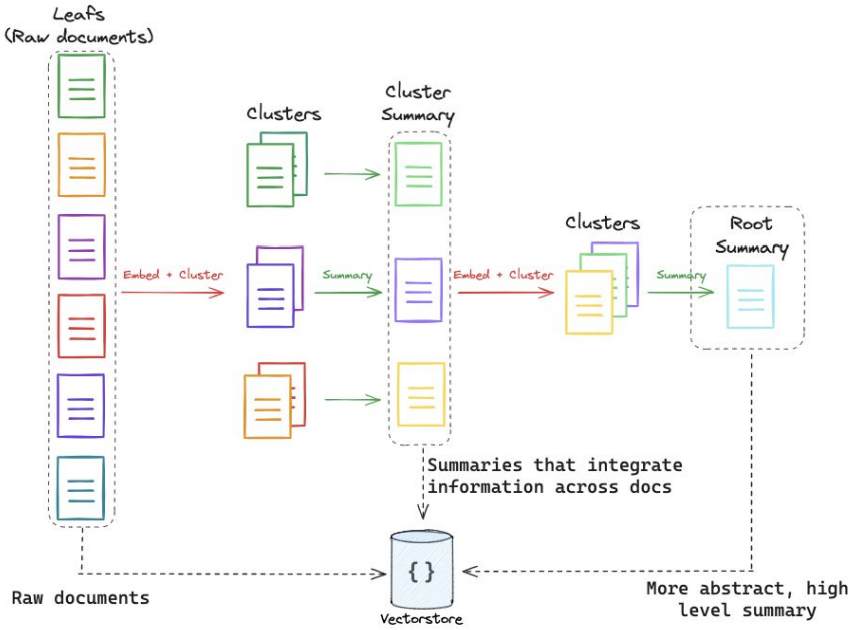
Indexing: Use representations to simplify document retrieval



https://arxiv.org/pdf/2312.06648.pdf
https://blog.langchain.dev/semi-structured-multi-modal-rag/
https://python.langchain.com/docs/modules/data_connection/retrievers/parent_document_retriever

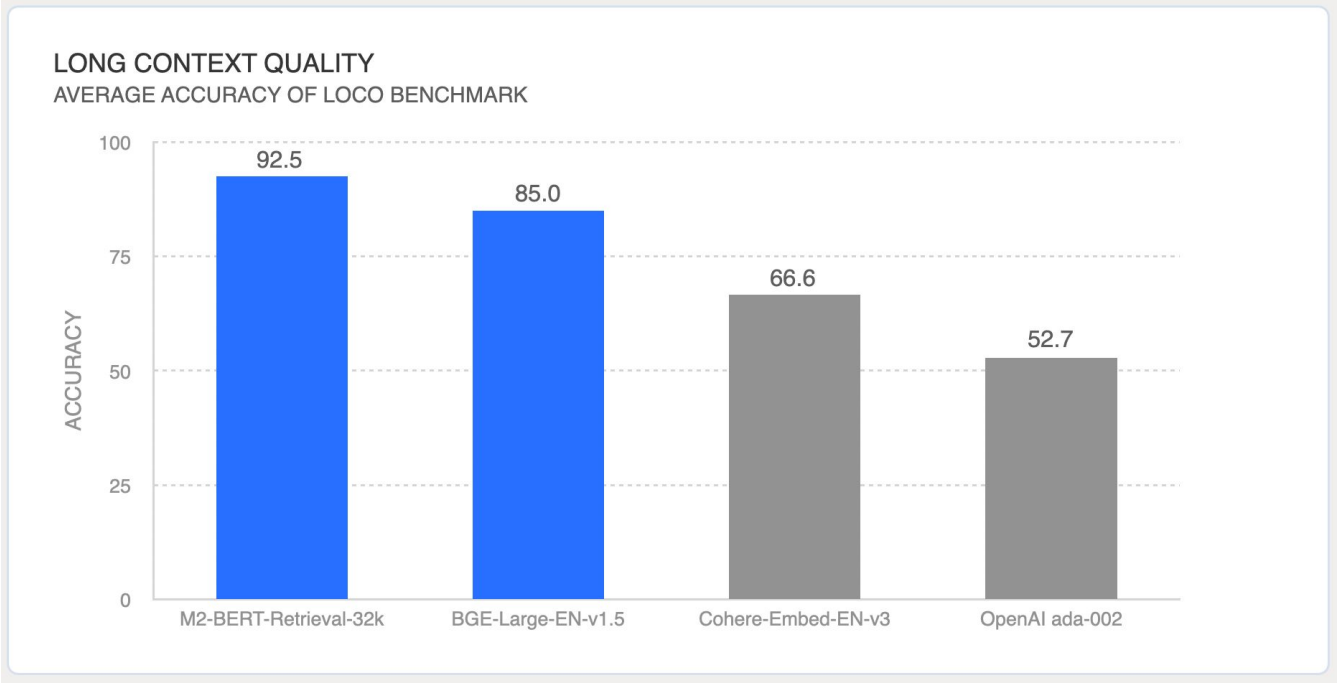
LangChain

Indexing: Use trees to consolidate info across many documents



https://arxiv.org/abs/2401.18059
https://www.youtube.com/watch?v=jbGchdTL7d0

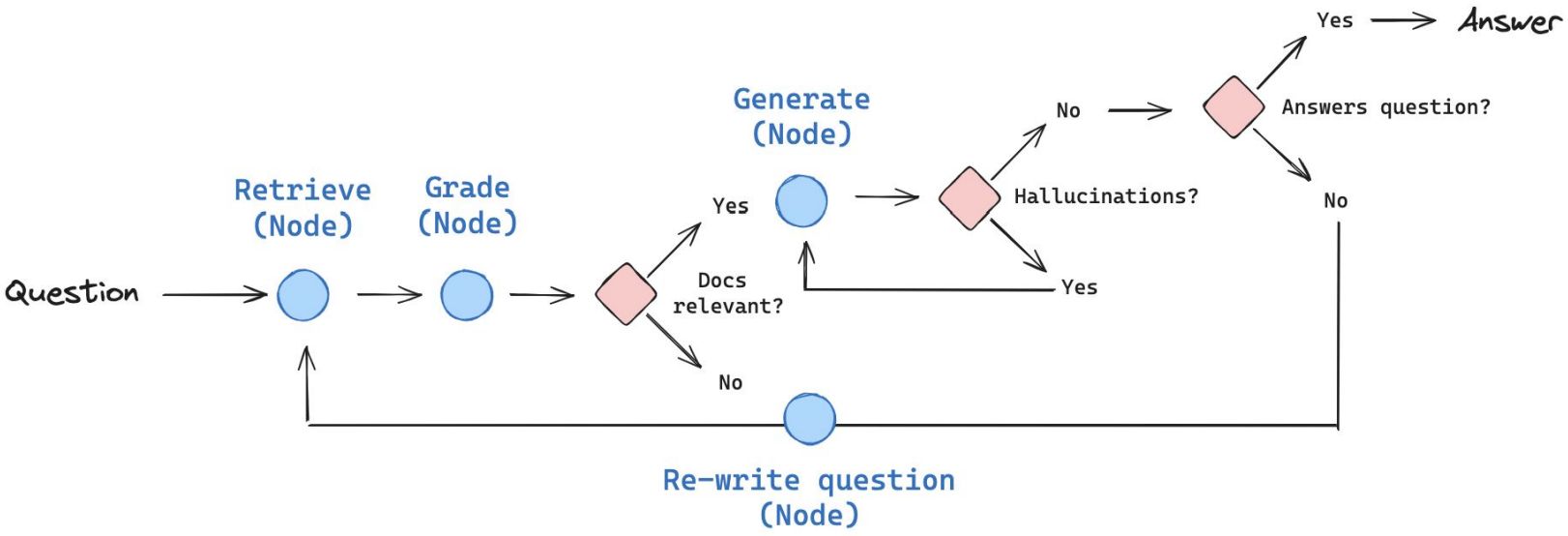
Indexing: Use long context embeddings



https://www.together.ai/blog/embeddings-endpoint-release
https://hazyresearch.stanford.edu/blog/2024-01-11-m2-bert-retrieval

LangChain

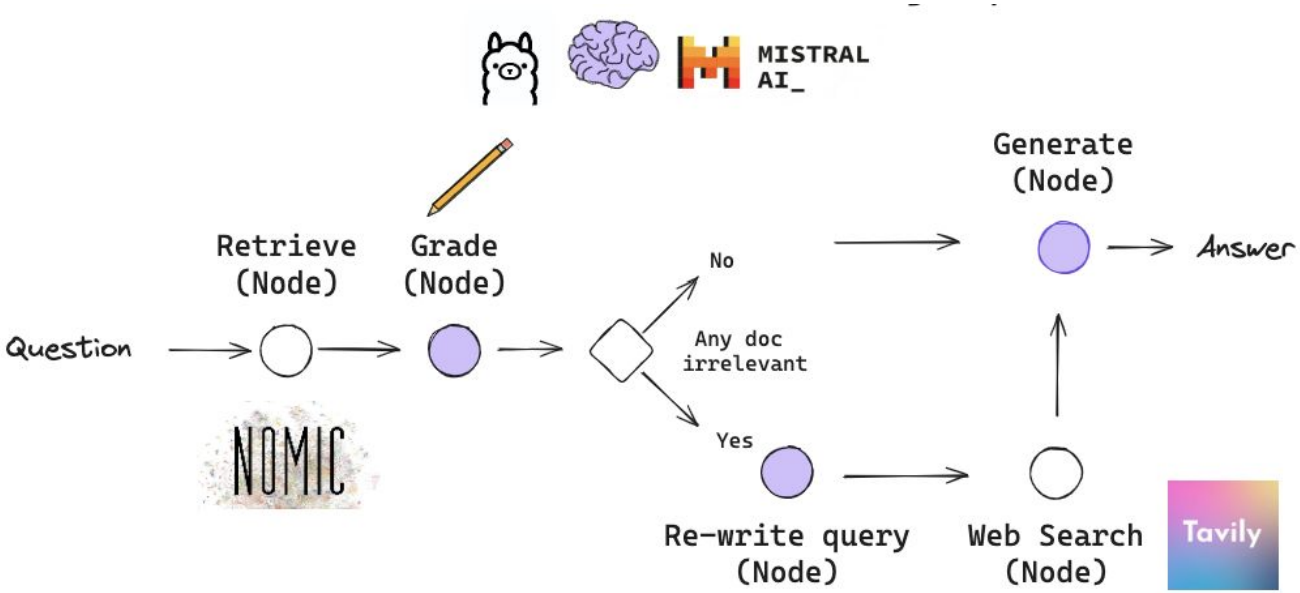
Reasoning: Use reasoning / self-reflection around RAG



https://arxiv.org/abs/2310.11511
https://www.youtube.com/watch?v=E2shqsYwxck
https://github.com/langchain-ai/langgraph/blob/main/examples/rag/langgraph_self_rag.ipynb

LangChain

Reasoning: Use reasoning / self-reflection around RAG



https://arxiv.org/abs/2401.15884
https://www.youtube.com/watch?v=E2shqsYwxck
https://github.com/langchain-ai/langgraph/blob/main/examples/rag/langgraph_crag.ipynb

LangChain

Overall picture: Doc-centric, no splits or compression, reason pre/post retrieval