

Hands-On Agentic RAG Development with Qdrant

By Mohammed Arbi Nsibi

13/06/2025





MOHAMED ARBI NSIBI

- ML engineer
- Qdrant Star
- Former GDSC Lead 23/24



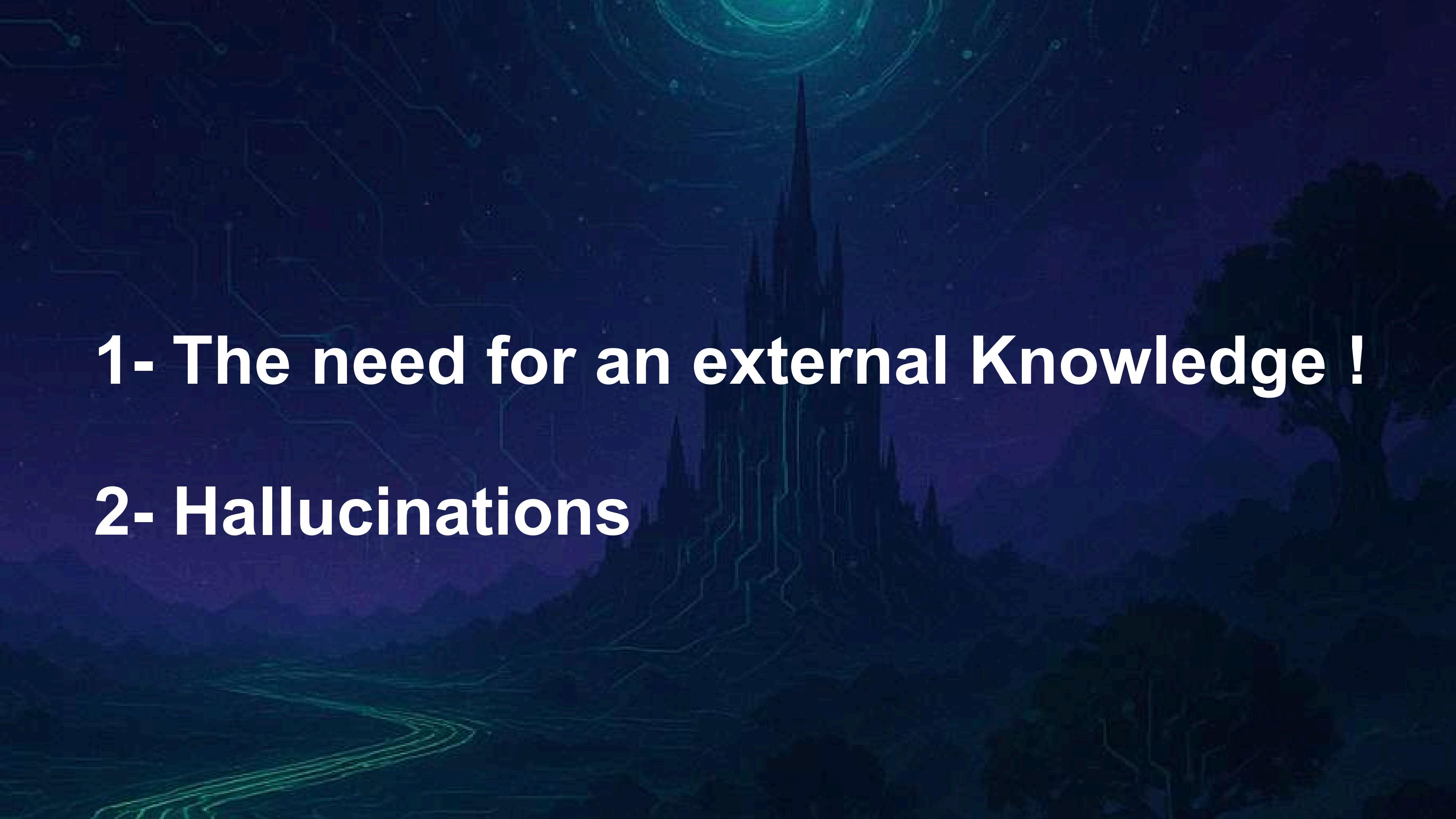
<https://huggingface.co/Goodnight7>



<https://www.linkedin.com/in/mohammed-arbi-nsibi>

Content

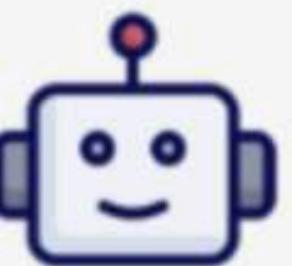
- Motivation
- RAG components
- Vector stores deep dive
- Building basic RAG pipeline
- What makes an AI an Agent ?
- Some Advanced RAG Techniques

- 
- 1- The need for an external Knowledge !
 - 2- Hallucinations

Hallucinations



Is 9677 a prime number?



No, 9677 is not a prime number.

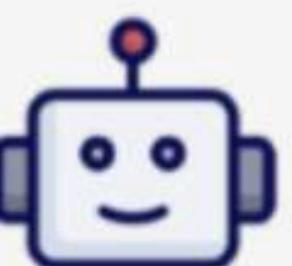
It can be factored into 13 and 745, as $9677 = 13 \times 745$.

} incorrect assertion

} snowballed hallucination



Is 9677 divisible by 13?



No

in a separate session,
GPT-4 recognizes its
claim as incorrect!



Hallucinations

- The model is not trained on enough data.
- The model is trained on noisy or dirty data.
- The model is not given enough context .
- The model is not given enough constraints (rules, guidelines, or limitations)

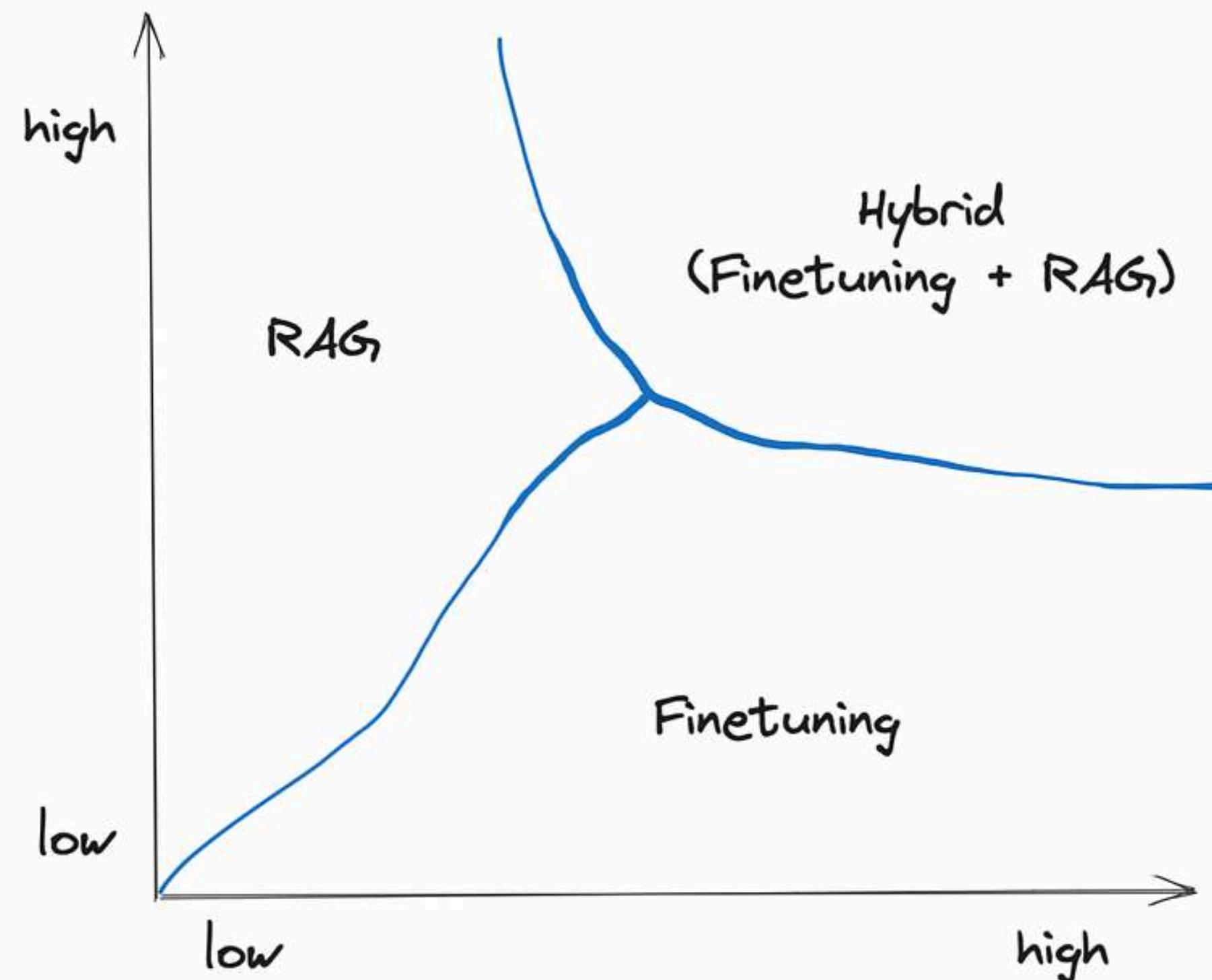
LLM AFTER TRAINING
ON 90% OF THE INTERNET...



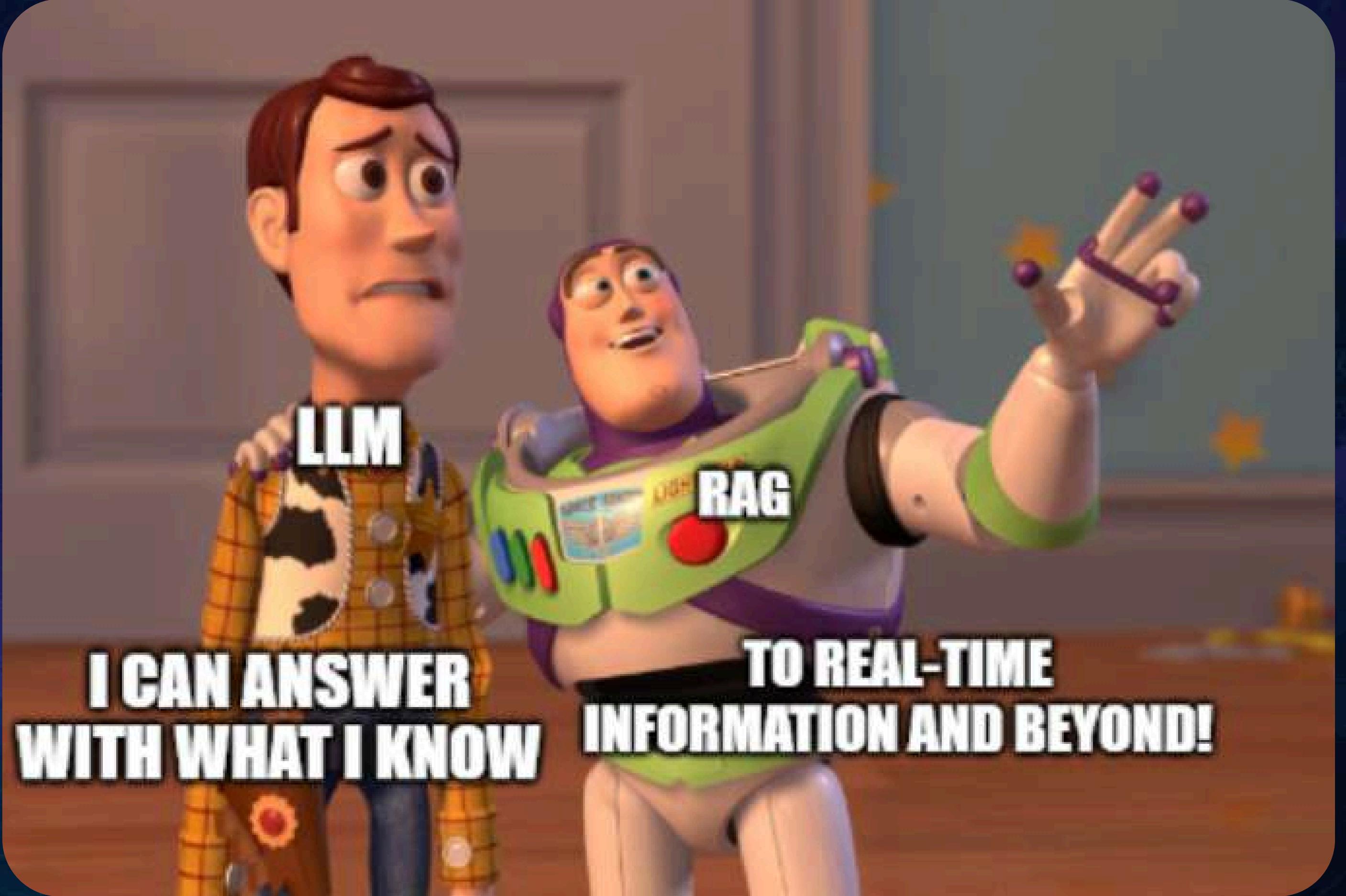
TIME TO HALLUCINATE SOME FACTS!

RAG / Fine-tuning

external knowledge
required

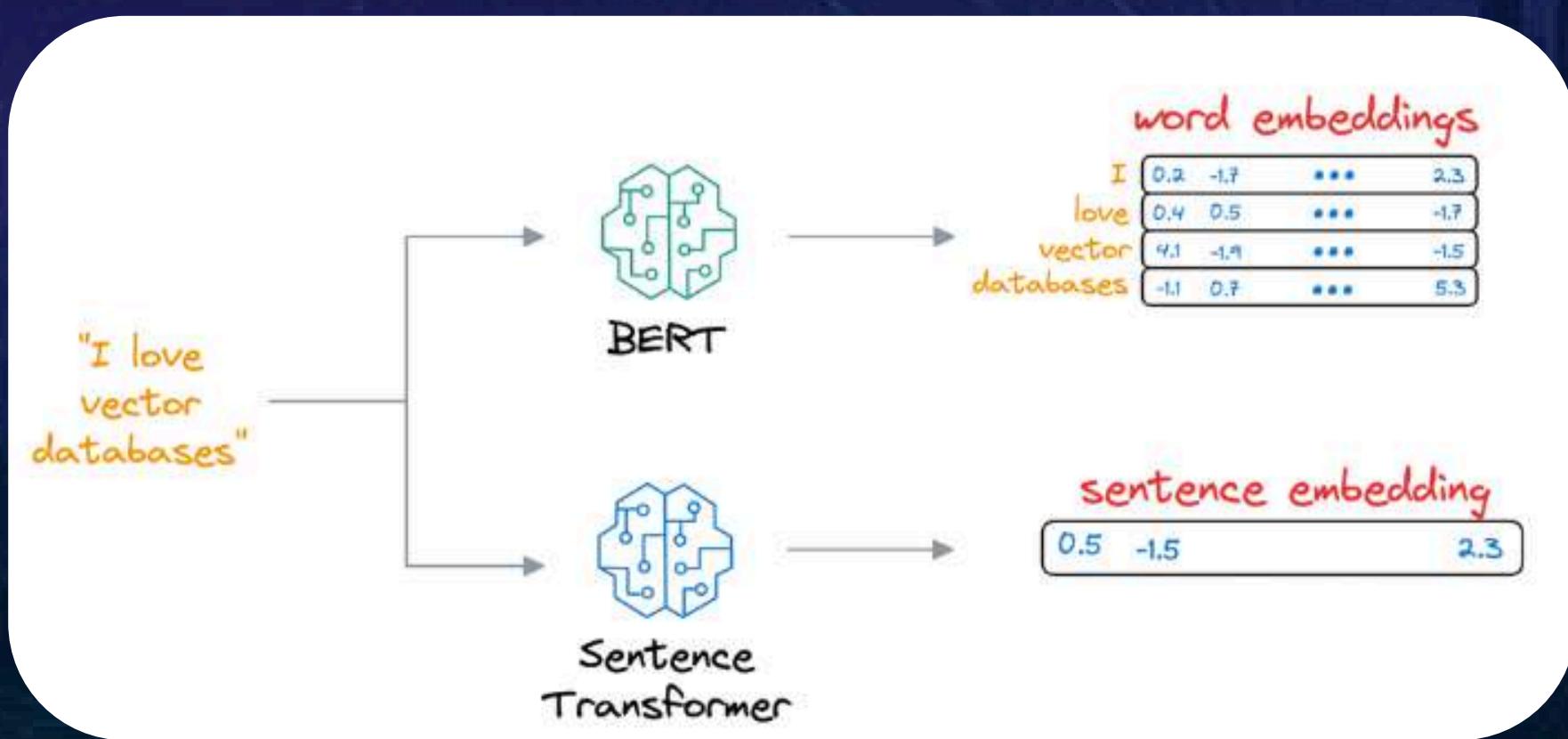


model adaptation required
(e.g. behaviour/
writing style/
vocabulary)

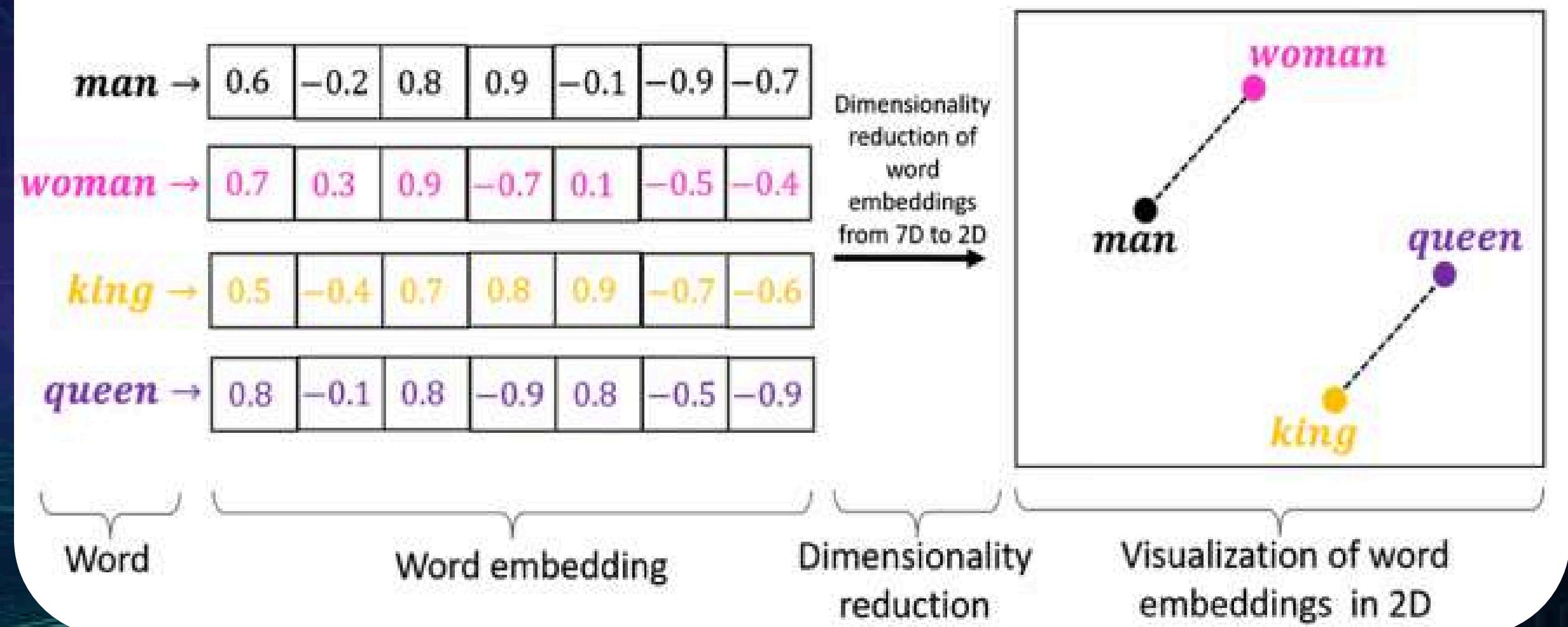
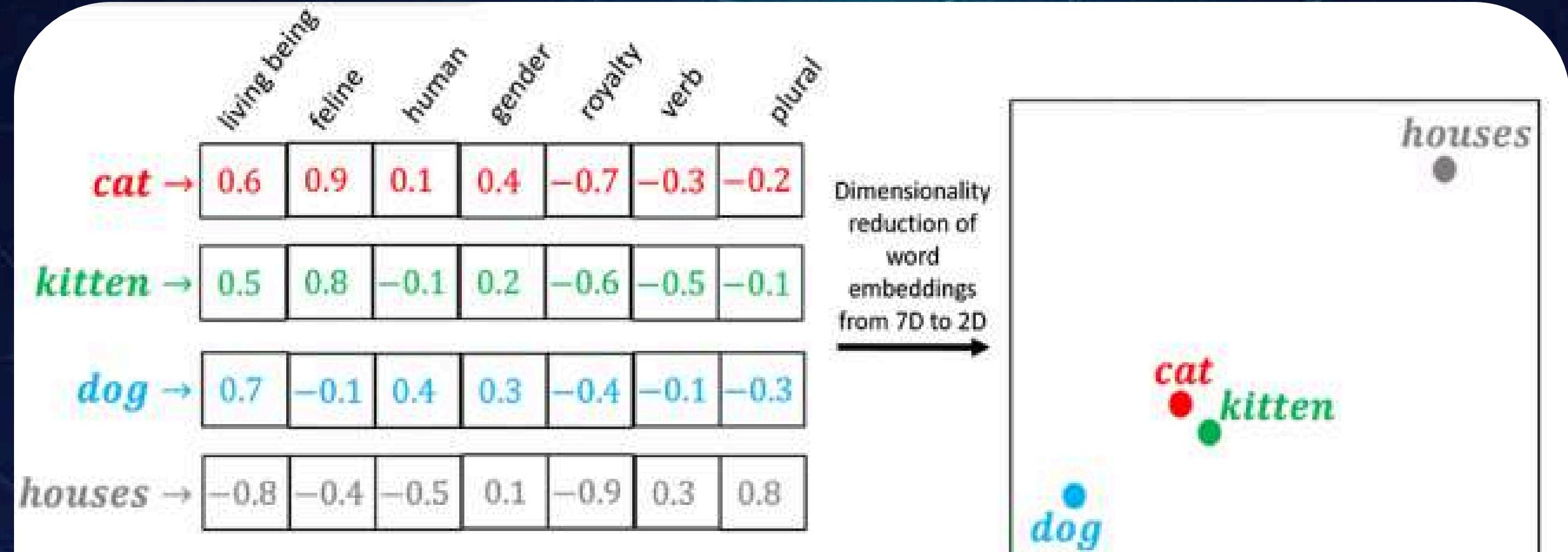


Embedding model

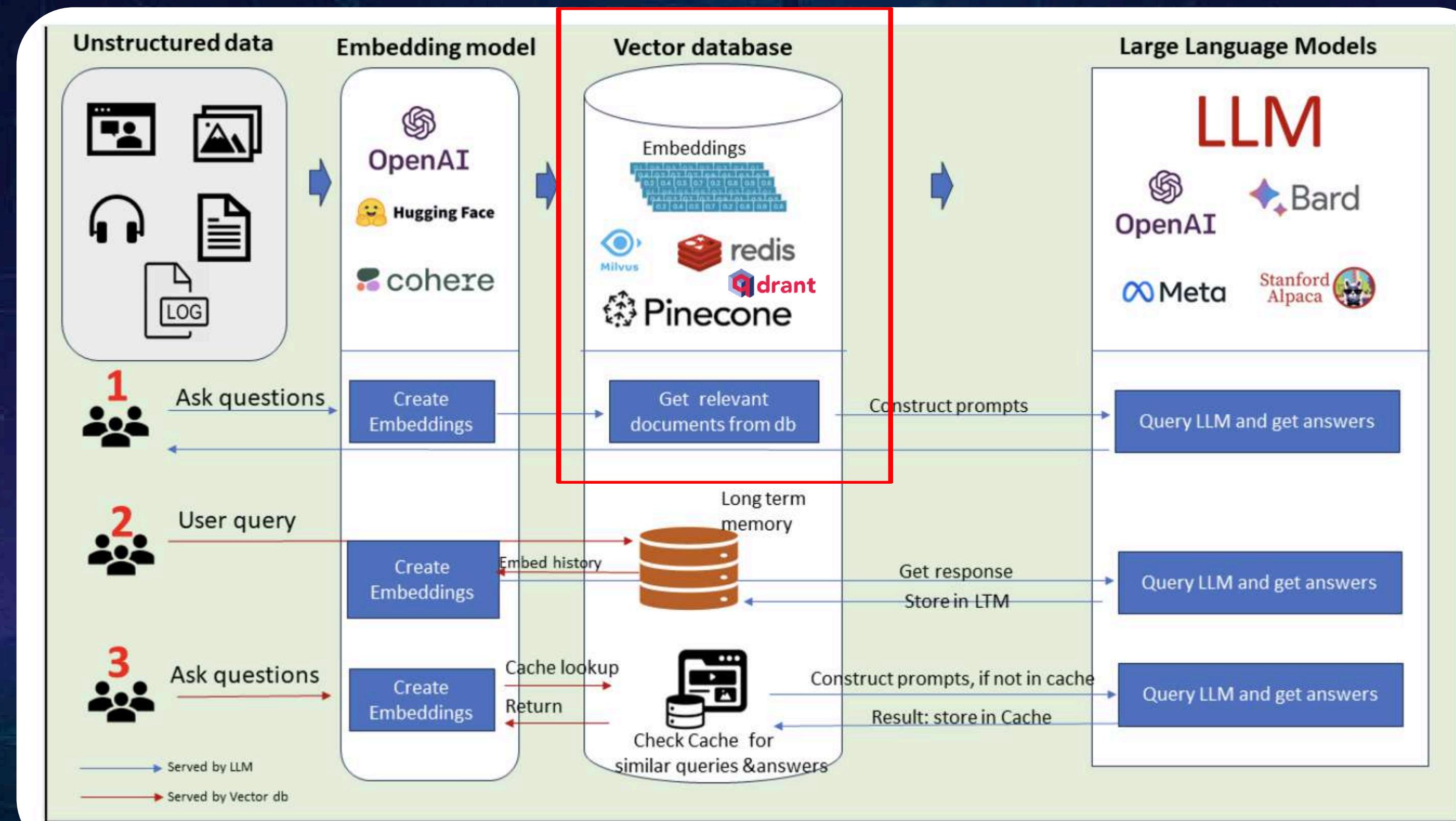
- These vectors live in a high-dimensional space where the proximity between vectors reflects the **relatedness** of the original items.



- Embedding model trained along LLM and learn to produce representation (vectors) based on context in word appear.



Vector Database





Fully Open-source
Self-hosting : run on ur own infra
Super fast: Latency ~0.024s (~24ms)
Hybrid Search
UI support
Free Tier ~1M(vectors) 768-dim vectors

Used by

[NLWeb link](#)

HubSpot

This repository uses GPT's visual capabilities to solve Wordle, indicating an AI-driven approach likely involving natural language processing and pattern recognition algorithms.

jennifermarsman/DeepReinforcementLearning

This repository implements the AlphaZero methodology, a deep reinforcement learning algorithm, which is a type of algorithm that can be adapted for solving complex problems like Wordle. Although it is not specifically designed for Wordle, the approach is relevant because it involves learning optimal strategies through self-play and reinforcement learning.

Do I have any code repositories that use the Phi model?

jennifermarsman/PhiRecycling

This repository uses the Phi vision model for sorting trash and recycling at scale, indicating direct use of the Phi model in its codebase.

jennifermarsman/Phi-3CookBook

This repository is directly related to the Phi model, specifically Phi-3, which is a family of open AI models developed by Microsoft. It serves as a practical resource for getting started with Phi-3, making it highly pertinent for anyone looking to work with or understand the Phi model.

What are some code repositories that use the Phi model?

Andre Zayarni (Author)

Co-founder & CEO, Qdrant | Open-Source Vector Search

Kevin Scott, Microsoft's CTO, used Qdrant as the vector engine for the #NLWeb demo at the opening keynote of the annual Build Conference.

#opensource rules! 🎉 ❤️ 🎉

You and 907 others

53 comments • 23 reposts

Love Comment Repost Send

Tell them what you loved... 😊 📸

Most relevant

Andre Zayarni (Author)

Co-founder & CEO, Qdrant | Open-Source Vector Search

Recording <https://youtu.be/ceV3RsG946s?si=Xh6NkXCJH1jhKHeV&t=4879>

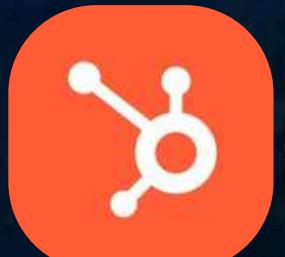
Like 4 Reply

Philippe Bourcier (2nd)

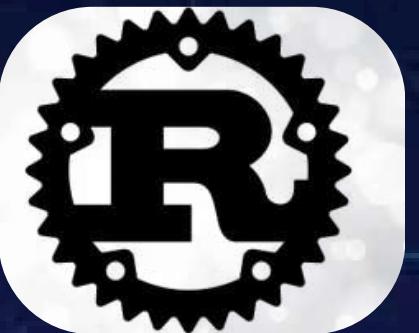
CTO #GenAI / Business Angel / Maker

Soon a Windows build for Qdrant ?

Like Reply 2 replies



Why Qdrant is super fast ?

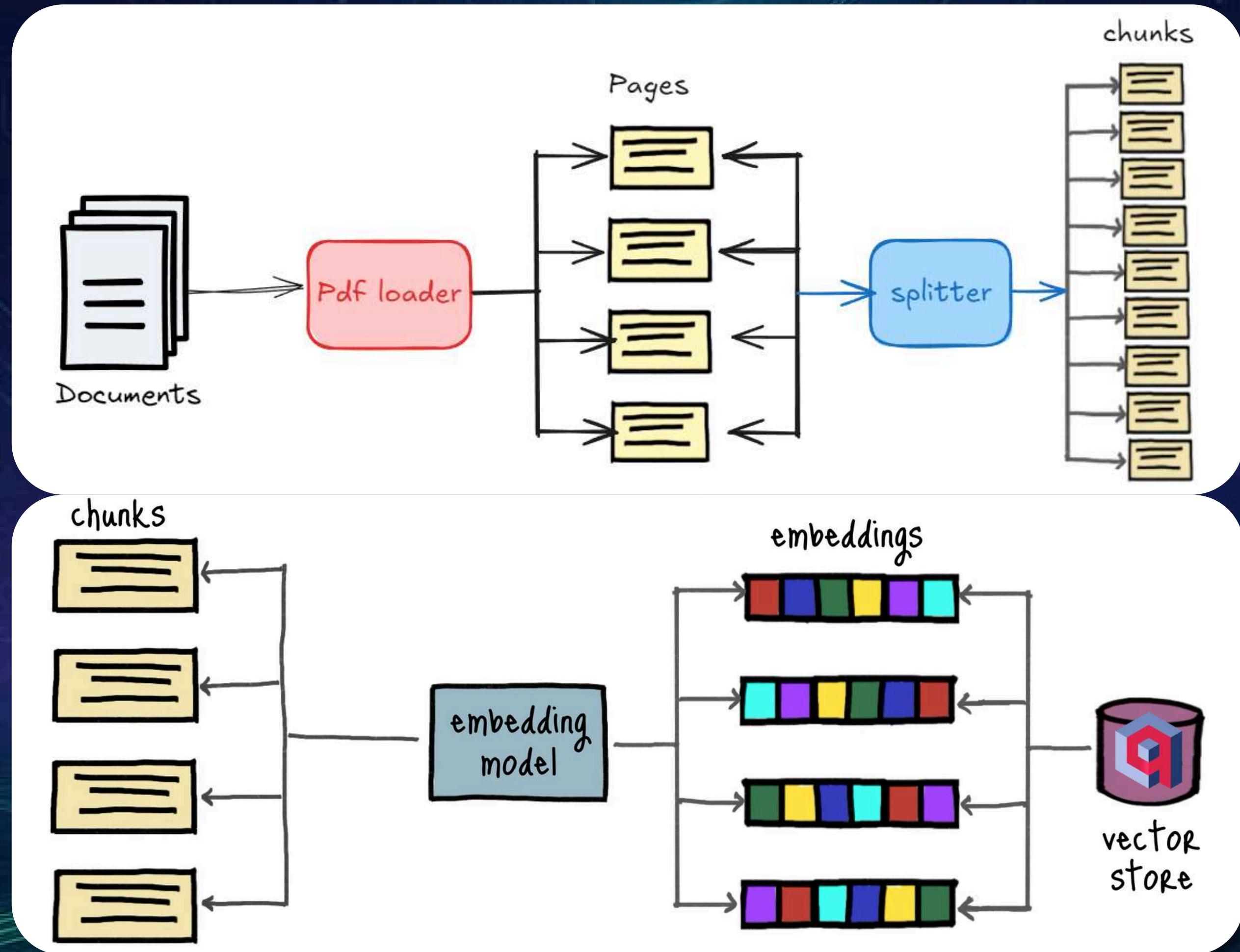


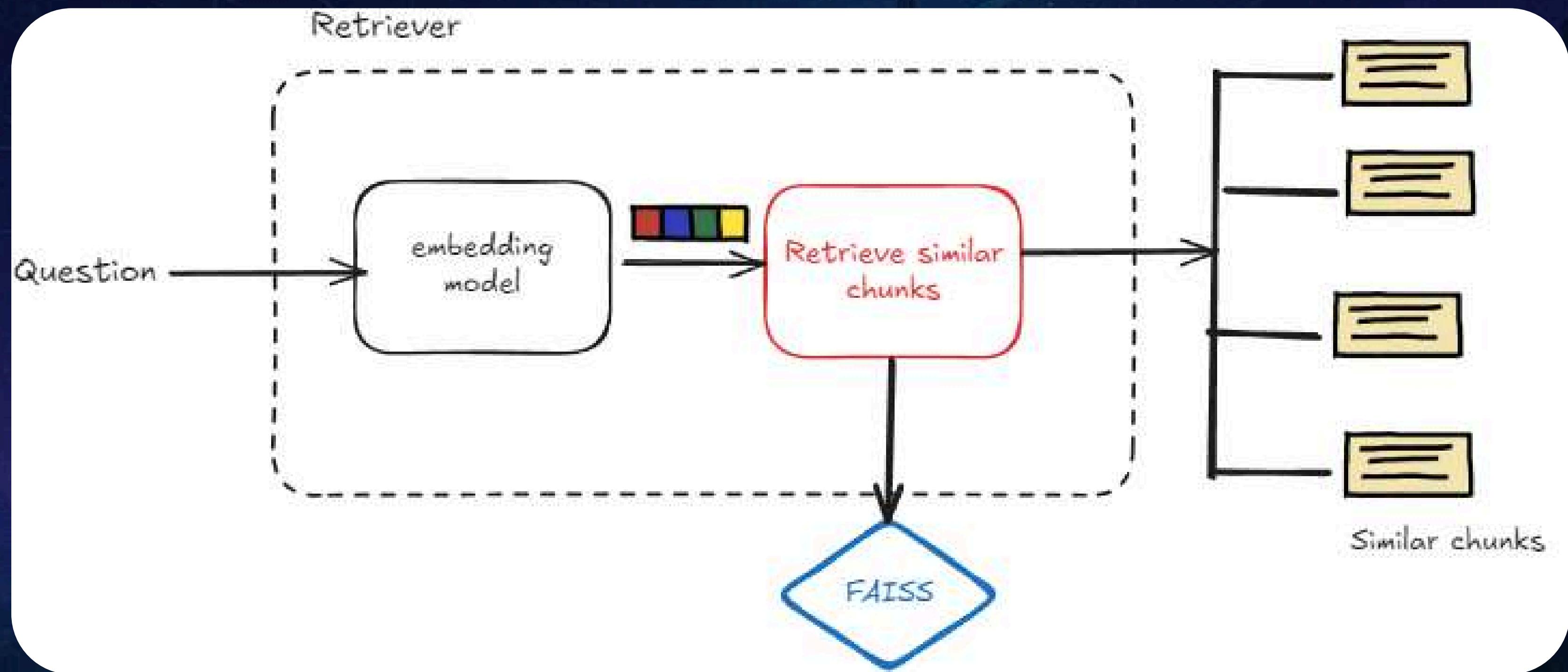
- Rust-Based Engine
- HNSW (Hierarchical Navigable Small World) Indexing : fast ANN search (on the best matches without scanning everything.)
- Vector Quantization (useful for large-scale datasets) : Saves RAM (up to 16x)
- Batch & Parallel Processing

QUANTIZATION EXAMPLE

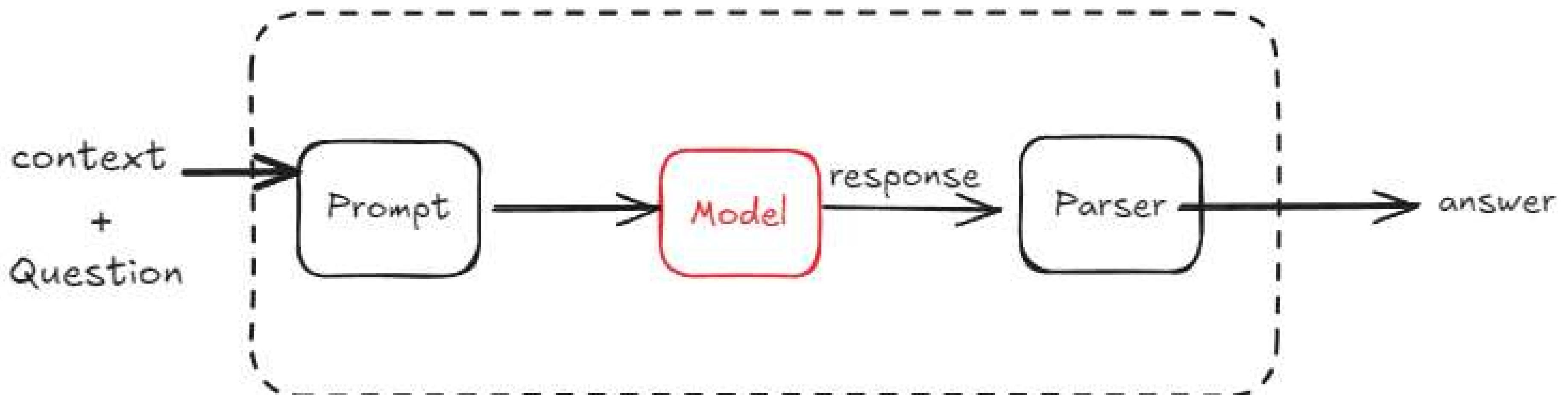
```
client.recreate_collection(  
    collection_name=collection_name,  
    vectors_config=models.VectorParams(  
        size=512, # dimension of your vectors #PQ works best with float32 vectors and dimensions divisible by 4 or 8.  
        distance=models.Distance.COSINE,  
        quantization_config=models.QuantizationConfig(  
            quantization=models.Quantization(  
                quantization=models.ProductQuantization( # using PQ method  
                    enabled=True,  
                    compression="x8", # or "x4", "x16" # x8 means 8x compression  
                    always_ram=False, # keep quantized vectors on disk  
                )  
            )  
        )  
    )  
)
```

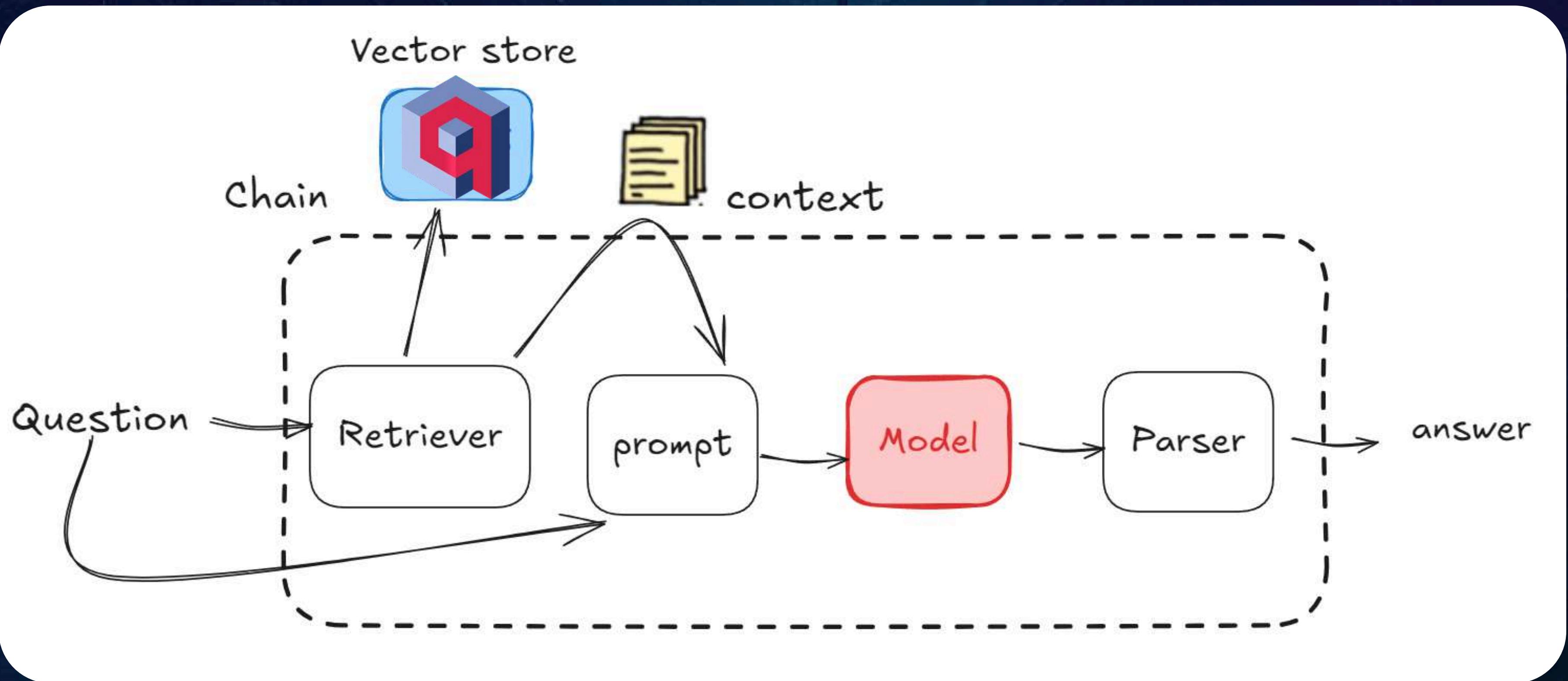
RAG architecture





Chain





What makes an AI an Agent

```
1  {- if tools %}
2    {{- '<|im_start|>system\n' }}
3    {- if messages[0].role == 'system' %}
4      {{- messages[0].content + '\n\n' }}
5    {- endif %}
6    {{- "# Tools\n\nYou may call one or more functions to assist with the user quer
y.\n\nYou are provided with function signatures within <tools></tools> XML tags:\n<t
ools>" }}
7    {- for tool in tools %}
8      {{- "\n" }}
9      {{- tool | toJSON }}
10     {- endfor %}
11     {{- "\n</tools>\n\nFor each function call, return a json object with function na
me and arguments within <tool_call></tool_call> XML tags:\n<tool_call>\n{"name": <
function-name>, "arguments": <args-json-object>}</tool_call><|im_end|>\n" }}
12   {- else %}
13     {- if messages[0].role == 'system' %}
14       {{- '<|im_start|>system\n' + messages[0].content + '<|im_end|>\n' }}
15     {- endif %}
16   {- endif %}
17   {- for message in messages %}
18     {- if message.content is string %}
19       {{- set content = message.content %}}
20     {- else %}
21       {{- set content = '' %}}
22     {- endif %}
23     {- if (message.role == "user") or (message.role == "system" and not loop.first)
%}
24       {{- '<|im_start|>' + message.role + '\n' + content + '<|im_end|>' + '\n' }}
25     {- elif message.role == "assistant" %}
26       {{- '<|im_start|>' + message.role + '\n' + content %}}
27     {- if message.tool_calls %}
28       {{- for tool_call in message.tool_calls %}}
29         {- if (loop.first and content) or (not loop.first) %}
30           {{- '\n' }}
31         {- endif %}
32         {- if tool_call.function %}
33           {{- set tool_call = tool_call.function %}}
34         {- endif %}
35         {{- '<tool_call>\n{"name": "' %}}
36         {{- tool_call.name %}}
37         {{- "', "arguments": ' %}}
38         {- if tool_call.arguments is string %}
39           {{- tool_call.arguments %}}
        {- else %}
```

- If a model can use tools or functions, you can classify it as an agentic AI model. A simple way to confirm this is to look at its [chat template](#).

What makes an AI an Agent

```
class AdditionTool(Tool):
    """
    A class-based tool for adding two numbers.

    name = "add_numbers"
    description = "Adds two numbers (integers or floats) together and returns the result."
    inputs = {
        'a': {
            'type': "integer",
            'description': "The first number to add."
        },
        'b': {
            'type': "integer",
            'description': "The second number to add."
        },
    }
    output_type = "integer"

    def forward(self, a: int, b: int) -> int:
        """
        The core logic of the tool. This method is executed when the tool is called.
        """
        return a + b

addition_tool = AdditionTool()
```

creating a subclass of Tool.

- Explicit control over schema.
- Good when you want strict typing, validation, or more complex tools.

```
@tool
def add_numbers(a: int, b: int) -> int:
    """
    Adds two numbers (integers or floats) together and returns the result.
    """

    Args:
        a: The first number to add.
        b: The second number to add.

    Returns:
        The sum of the two input numbers.
    """
    return a + b
```

decorator based

- Short.
- Simple.
- Feels natural when the tool is just a single operation.
- Ideal for quick tools

How to get started ?

LangChain

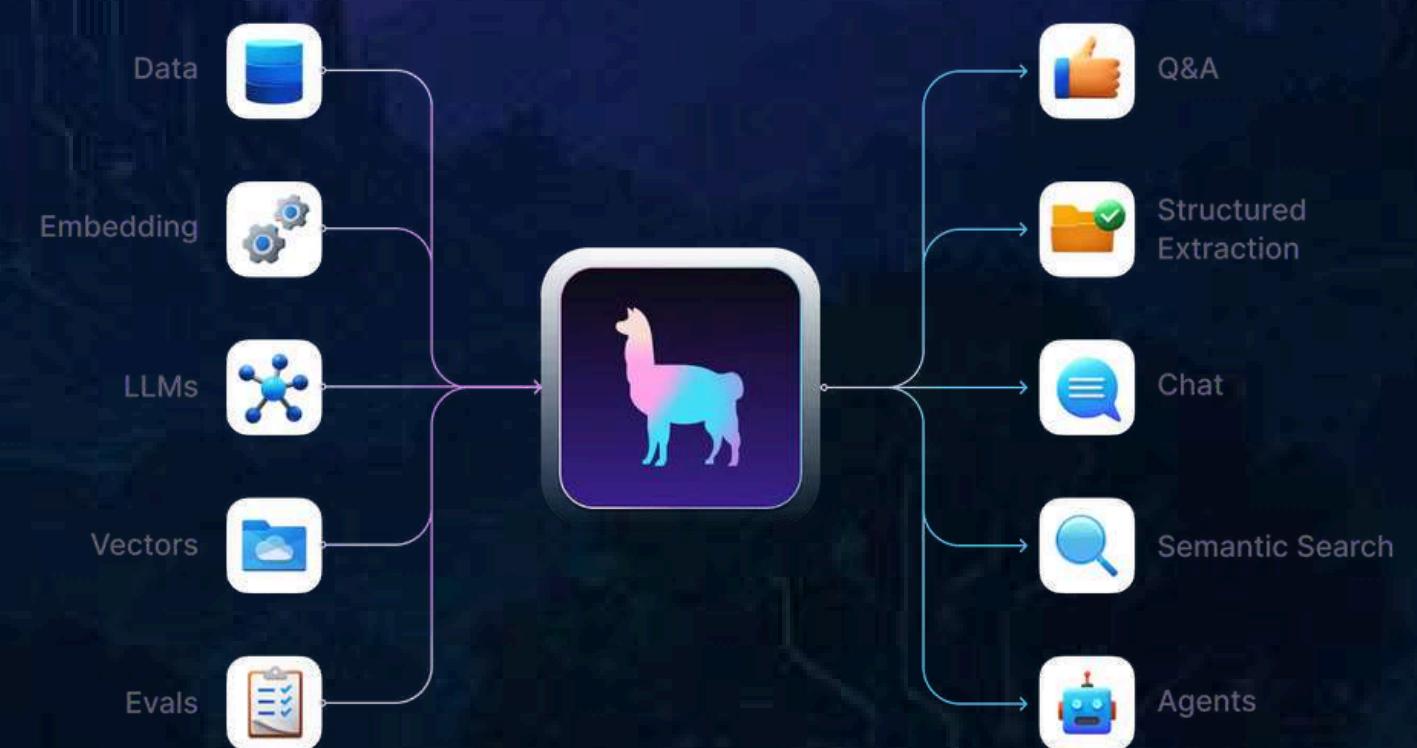
- LangChain is a framework designed to simplify the creation of applications using large language models.



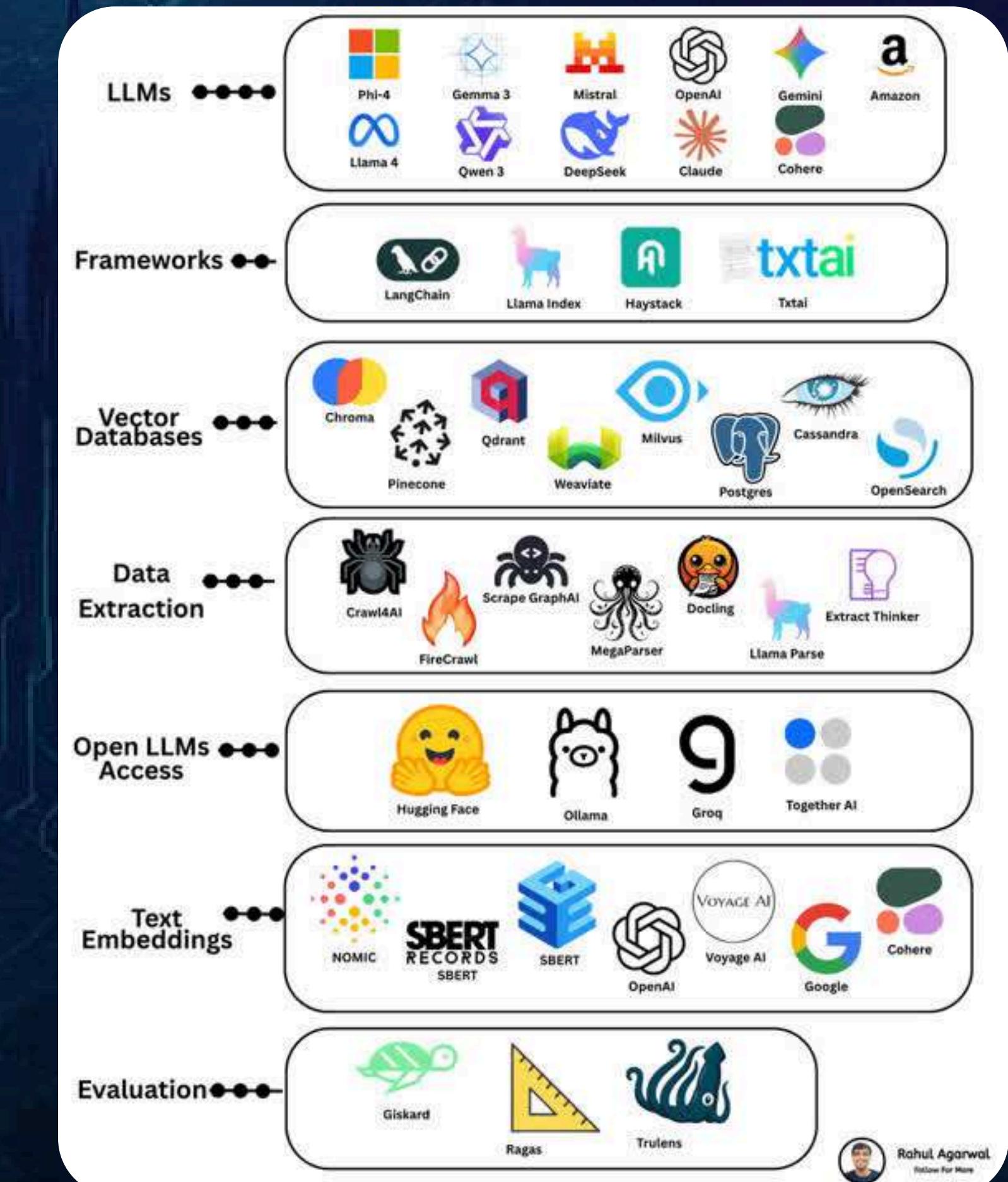
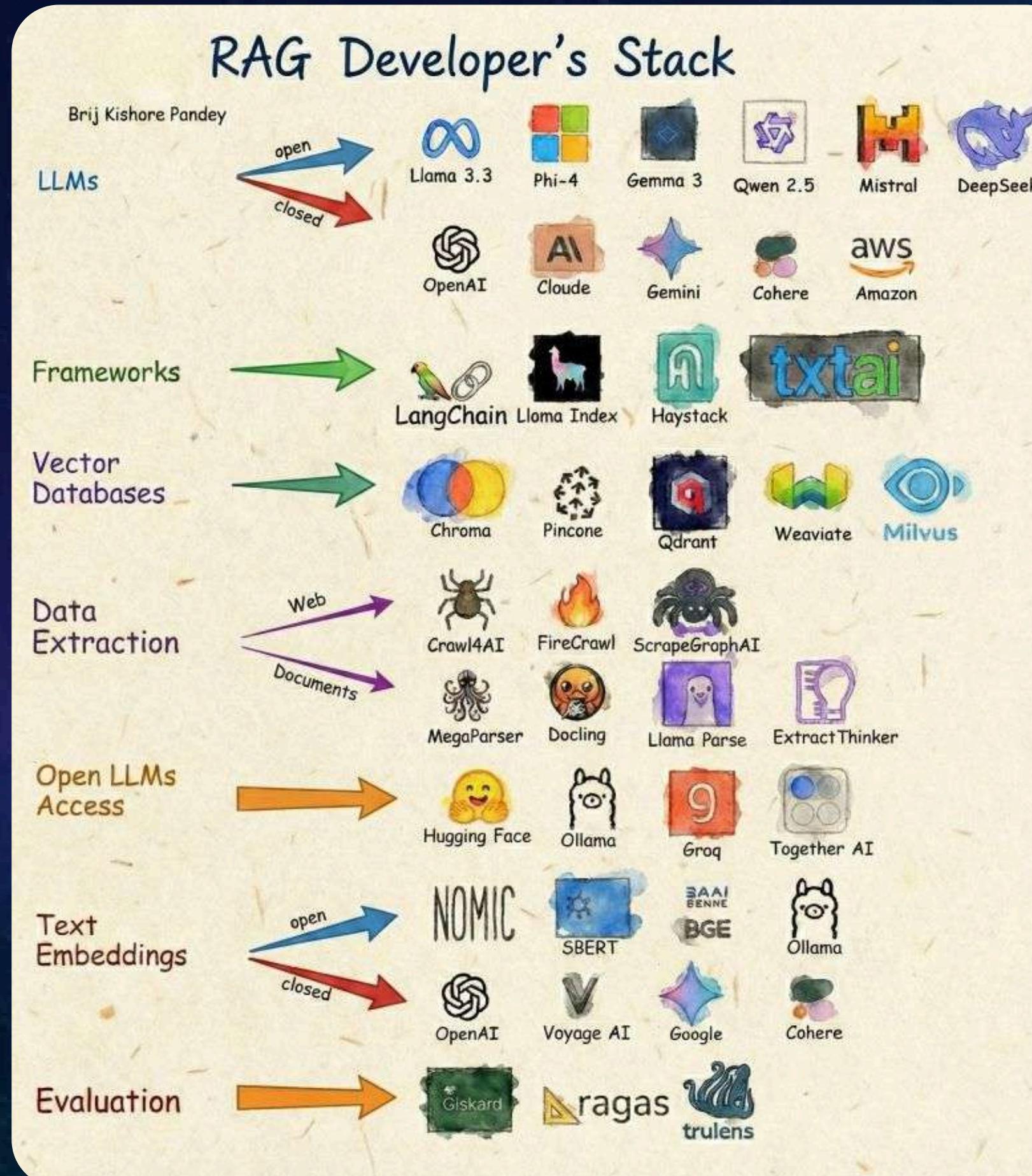
LlamaIndex



- LlamaIndex is a handy tool that acts as a bridge between your custom data and large language models (LLMs) which are powerful models capable of understanding human-like text.



How to get started ?





smolagents

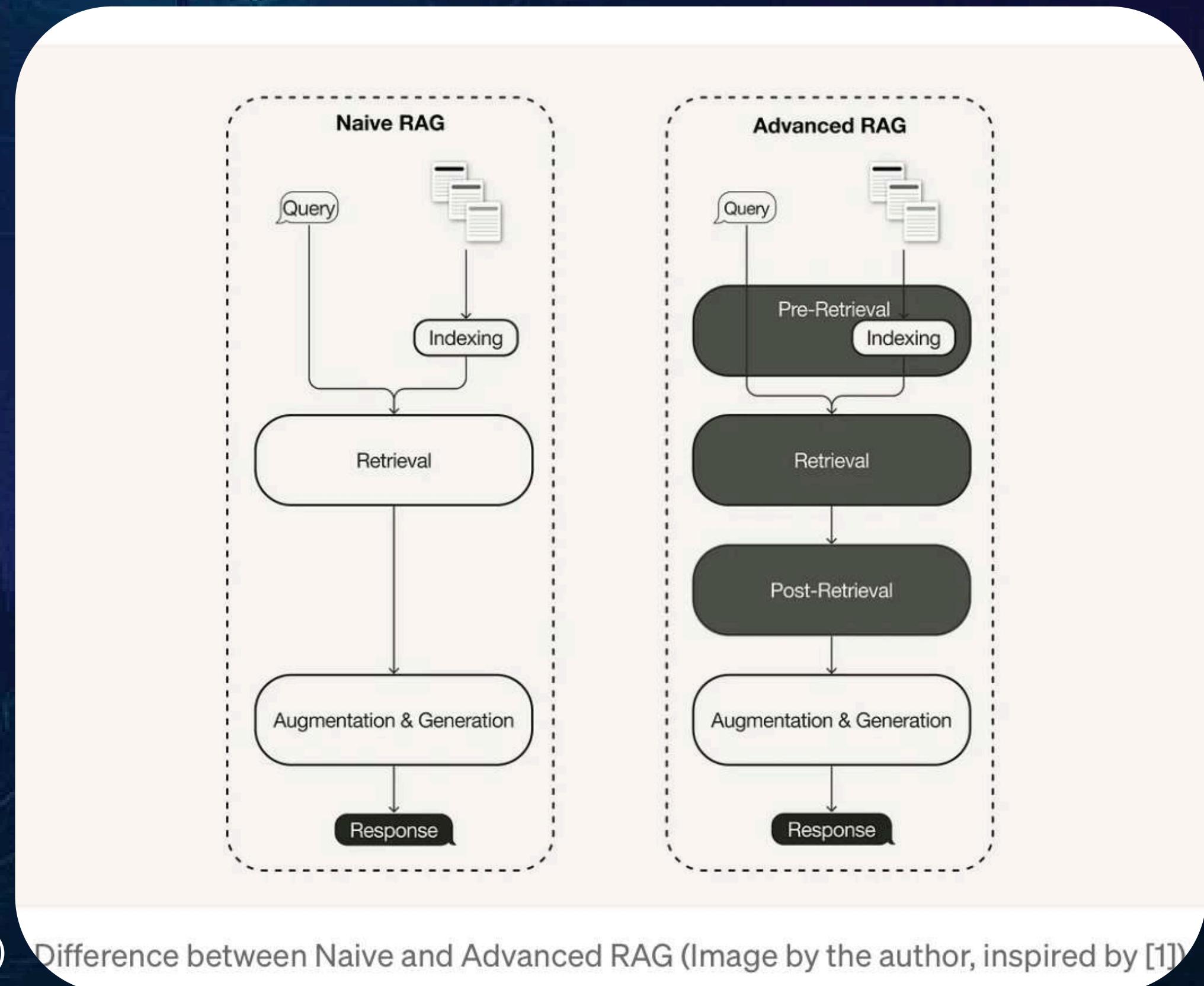
DEMC



smollagents

Naive RAG vs Advanced RAG

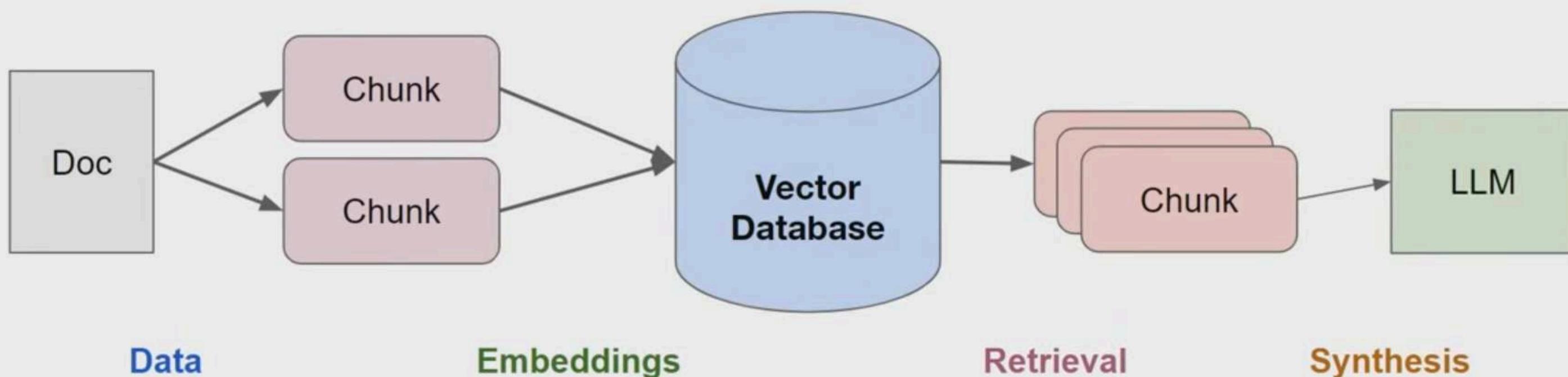
- There are many implementation to further improve performance of Naive RAG.
- Advanced RAG has evolved as a new paradigm with targeted enhancements to address some of the limitations of the naive RAG paradigm.
 - Advanced RAG techniques can be categorized into
 - pre-retrieval optimization,
 - retrieval optimization, and
 - post-retrieval optimization
 - some examples :
 - Feedback loops (re-ranking, similarity score thresholds)
 - Hybrid Search (dense + keyword)
 - Contextual compression (summarize before feeding to LLM)
 - Multi-vector per chunk (dense embeddings per aspect)

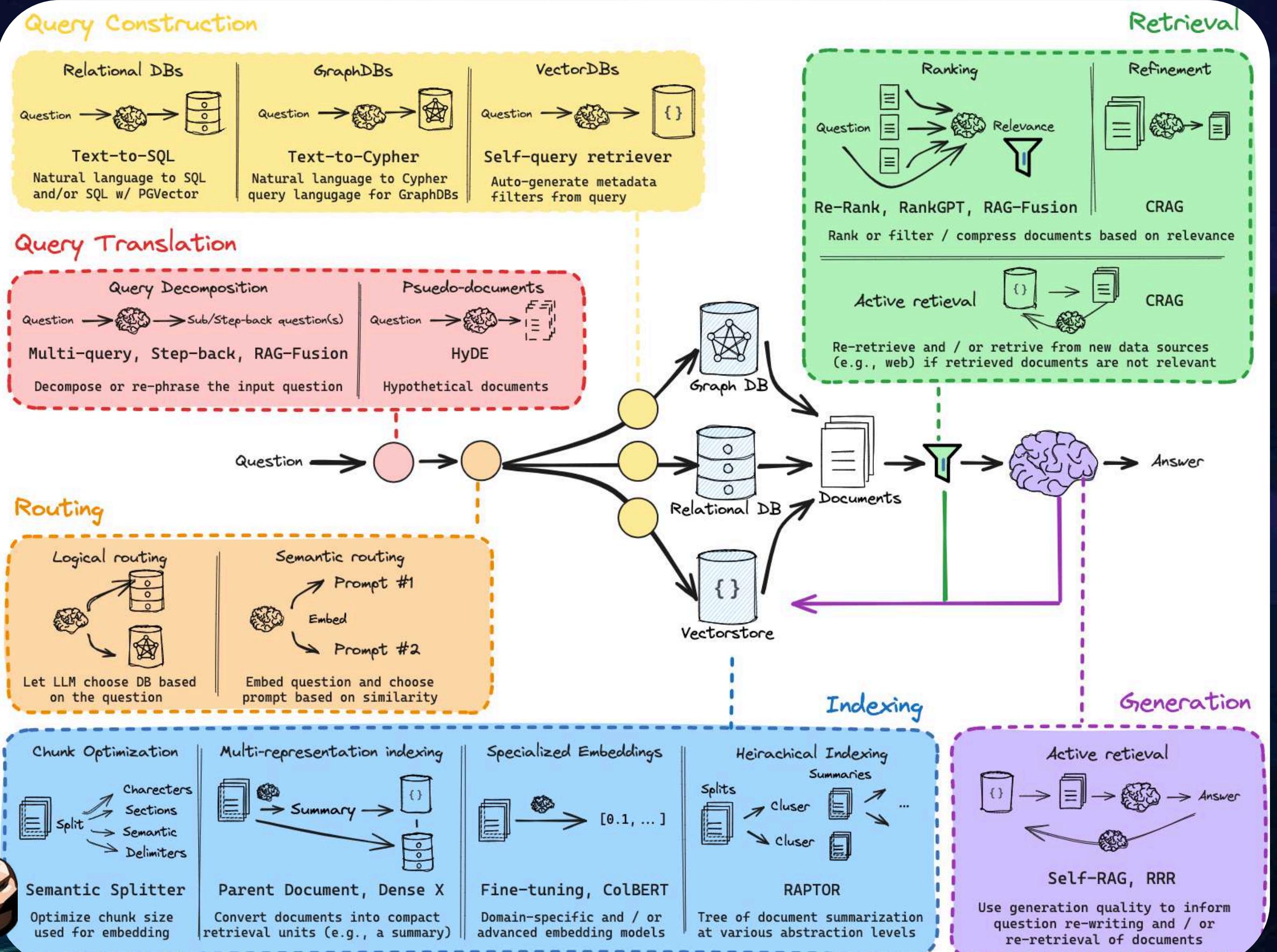


Naive RAG vs Advanced RAG

What do we do?

- **Data:** Can we store additional information beyond raw text chunks?
- **Embeddings:** Can we optimize our embedding representations?
- **Retrieval:** Can we do better than top-k embedding lookup?
- **Synthesis:** Can we use LLMs for more than generation? ✓





Resources

- [Qdrant + DataTalks.club free course](#)
- [Just-RAG Github Repo](#)
- [How to get started with Qdrant](#)
- [Similarity search HNSW](#)
- [Building neural search service with ST and Qdrant](#)
- [Your RAG powered by Google Search Technology](#)
- [Embedding models leaderboard](#)
- [Let's talk about LlamaIndex and LangChain](#)
- [Retrieval-Augmented Generation \(RAG\) framework in Generative AI](#)
- [\(RAG\): From Theory to LangChain Implementation](#)
- [Free Perplexity Pro for 3 months](#)

THANK YOU FOR YOUR ATTENTION!!

Where to find me?



Linkedin Profile



Goodnight



MedArbiNsibi

THANK YOU FOR YOUR ATTENTION!!

 Khushal Kumar  Following
Software Engineer – GenAI @Wingify | Masters in AI/ML | Follow me to Learn AI Engineering in Quick Si...
1w • 4

...

I once took an interview where a candidate really stood out, all because of how he handled RAG.

Most candidates presented the same basic Retrieval-Augmented Generation setup. You know, plug in a vector DB, chunk text, retrieve, generate... nothing unusual.

But this candidate went deeper.

He didn't just talk about how RAG works. He showed how tables and other unstructured data could be extracted and indexed from documents. He thought about real-world use cases, not just the standard pipeline.

The tools weren't what impressed me, it was the thinking. The attention to detail. The willingness to go beyond what everyone else was doing.

It's been months, and I still remember that interview.

In a world where everyone knows the basics, it's the "depth" that makes you unforgettable.

Curious, what's the most memorable interview experience and why?

And by the way, if you're trying to level up your GenAI interview or assignment game, I've created something new to help with exactly that.

 Link's in the comments.

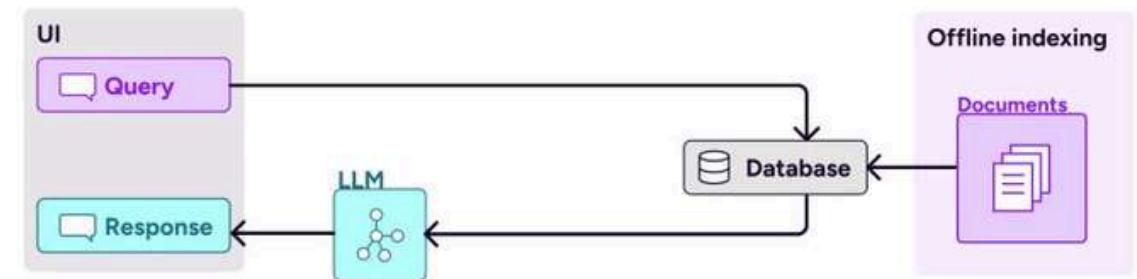
#coding #codingInterview #RAG #python

 Ahmed SIDI AHMED and 102 others

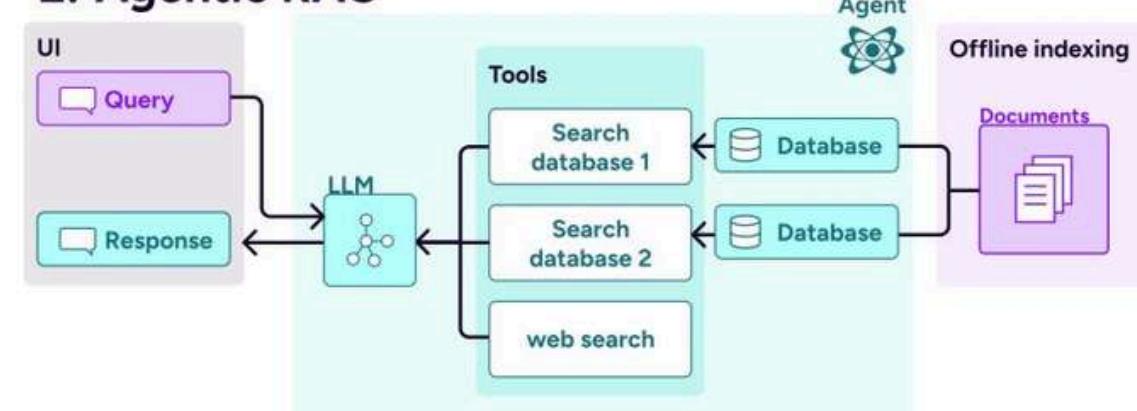
4 comments

The Evolution from RAG to Agentic RAG to Agent Memory

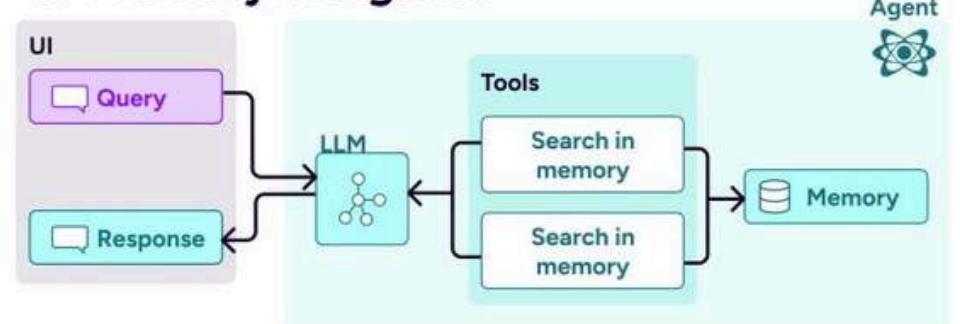
1. Naive RAG



2. Agentic RAG



3. Memory in Agents



visual inspired by *The Evolution from RAG to Agentic RAG to Agent Memory*, Leonie Monigatti