RAG vs Fine-tuning: which one is better for your AI project?

By Shao Hang HE

A bit about myself

I'm Shao Hang He

I'm a senior developer with 11 years of experience.

I launched MailMagic.ai (3 000+ users).

I launched Athena AI (30 000+ users).

I launched Athena Whiteboard (500 000+ users).

I'm an entrepreneur running my own Development and LLM consulting firm.

Prompt engineering

The most basic way to use OpenAI's GPT models is to ask simple questions.

Prompt: Explain what is a dev conference in 2 sentences.

Output: A developer conference is an event where software developers, programmers, and tech enthusiasts gather to learn about the latest technologies, programming languages, and industry trends through workshops, talks, and networking opportunities. These conferences serve as platforms for sharing knowledge, fostering collaboration, and exploring innovations in the realm of software development.

Few-shot learning

Since LLMs are pretrained, we only need to guide it with a few examples in most cases.

Prompt: Here is an email I received today: “It was nice meeting you at Confoo. Can you tell me more about your company?”. Write a reply to this message.
Here are some examples of my previous reply:

1. It was nice to meet you at CollisionConf, my company provide dev services to small and medium enterprises.
2. Great meeting you at the Web Summit! I am doing AI consulting now! Let’s discuss in detail next week?

Advanced LLM techniques:

When you have too much data for the LLM prompt context, you would need other ways to extend the LLM’s knowledge:

- Fine tuning: Supervised learning with well formatted training set
- Retrieval Augmented Generation (RAG): Search algorithm on existing unstructured data such as PDFs or structured data such as SQL or JSON

Fine tuning

- Only useful for getting more consistent output
- Reduce the chance of hallucinations
- Shorter prompts for a well fine tuned model

Challenge with Fine tuning

- Time consuming to gather data.
- Required large amount of data to be effective
- Data privacy issues if trained with real user data
- Does NOT improve the strength of the model
- Only the weaker models are available for fine-tuning
- Not ideal for small businesses

Retrieval Augmented Generation (RAG)

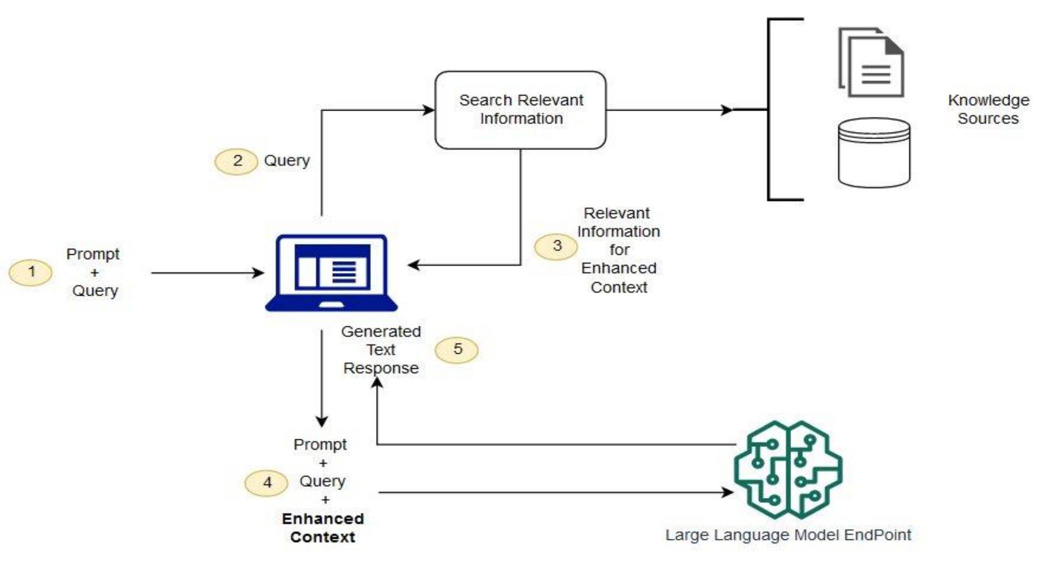
- Augment your prompt with external data
- Work with existing unstructured data such as PDF files and web pages.
- No need to create a training set.
- Work with other LLMs

Vector Embeddings

- Vector embeddings are a mathematical representation of the text data.
- Use models to generate embeddings
- Turn your unstructured text data into vector embeddings
- Turn the user query into a vector embedding.

Basic RAG pipeline

Find the vector with the closest distance to the query vector. We can use the search algorithm from OpenAI.


Search Relevant Information
Knowledge Sources
2 Query
Relevant Information for Enhanced Context
3
1 Prompt + Query
Generated Text Response
5
Prompt + Query + Enhanced Context
4
Large Language Model EndPoint

Vector Databases

In a real application, you would need to store the vector embeddings in a vector database. This way, we don't need to turn the documents into embeddings every time.

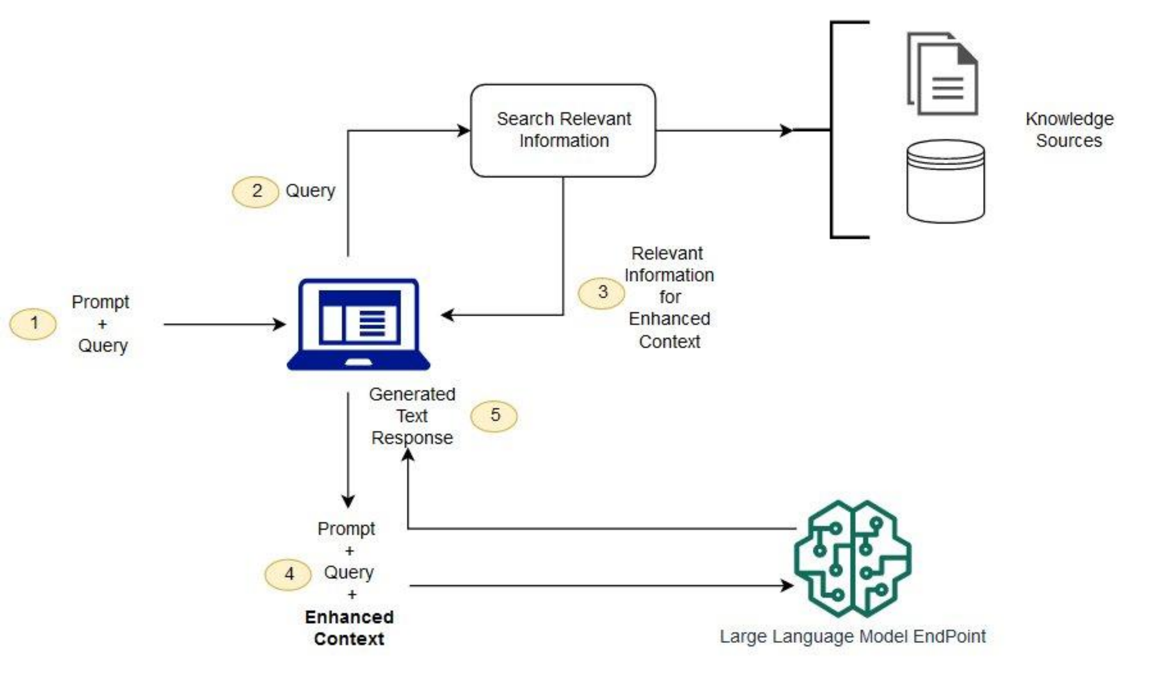
You can you a classic database that support vector embeddings such as PostgreSQL and Redis.

Dedicated Vector Databases:

- Pinecone
- Chroma
- Weaviate

Prompt engineering with RAG

Once you get the matched embeddings, you can use it to feed them into your prompt.


Search Relevant Information
Knowledge Sources
2 Query
1 Prompt + Query
3 Relevant Information for Enhanced Context
Generated Text Response 5
4 Prompt + Query + **Enhanced Context**
Large Language Model EndPoint

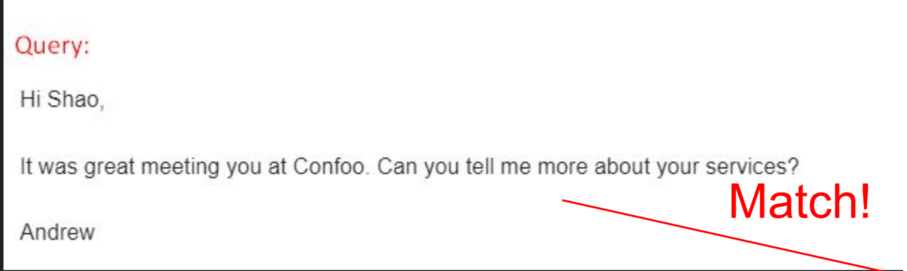
Example application Mail Magic

Mail Magic uses RAG to generate draft email replies for the user. It can write email draft while you are sleeping! Please check https://mailmagic.ai/

- Read your past emails and turn them into vector embeddings.
- Turn the incoming email (query) into a vector embedding.
- Run search algorithm to extract the matched vectors.
- Generate enhanced prompt with the query and matched vectors.
- Give the prompt to the LLM (GPT or Llama)

Mail Magic RAG

A past message in sent inbox contain the reply I sent and the original message in quote. We can use the new message as query and RAG will find the messages that have a close match with the original message.



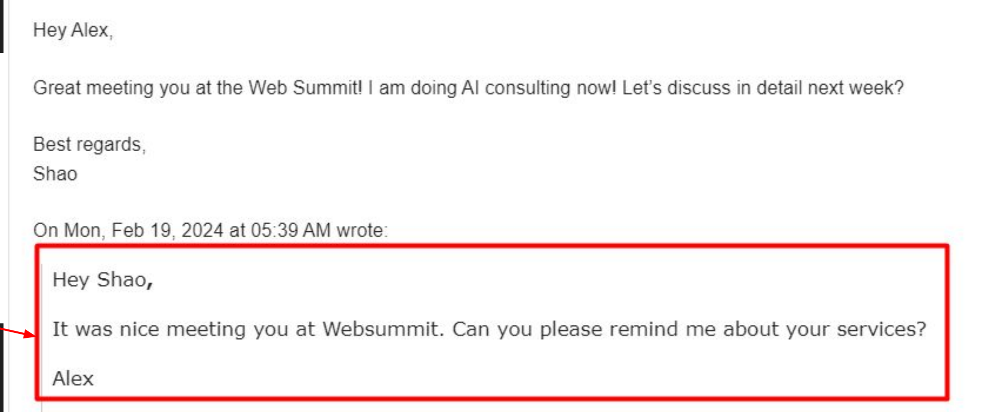
Query:

Hi Shao,

It was great meeting you at Confoo. Can you tell me more about your services?

Andrew

Match!



Hey Alex,

Great meeting you at the Web Summit! I am doing AI consulting now! Let's discuss in detail next week?

Best regards,
Shao

On Mon, Feb 19, 2024 at 05:39 AM wrote:

> Hey Shao,
>
> It was nice meeting you at Websummit. Can you please remind me about your services?
>
> Alex

Mail Magic Fine-tuned tones

RAG often gets the correct information but it’s overly polite. It always starts with “Thank you for your message” or “Thanks for your clarification”.

Fine tune the answer with thousands of my example replies and get my specific tone: “Please let me get back to you shortly”.

Retrieval with structured data

Pro:

- more accurate retrieval
- no need to generate and maintain embeddings
- work with your existing database

Cons:

- not always do-able
- need filterable data and well structured schema

Example: Chess puzzles recommender

Chess puzzles recommender in https://athenachat.bot

- Database with 4 million chess puzzles
- Filterable structured data: rating, puzzle theme, opening tags
- Chess is a popular game where LLMs have a lot of information.

Example: Chess puzzles recommender

System prompt:

- Retrieve all possible themes and opening tags
- Describe the database schema with details
- Ask the AI to return a db query based on user’s prompt

User Prompt:

I play the Queen’s Gambit. I need a difficult puzzle on middle game King side attack.

Example: Chess puzzles recommender

Use another prompt to filter the result.

System prompt: “Give the data about these chess puzzles: {insert data}, return the most relevant puzzle that matches user’s request.”

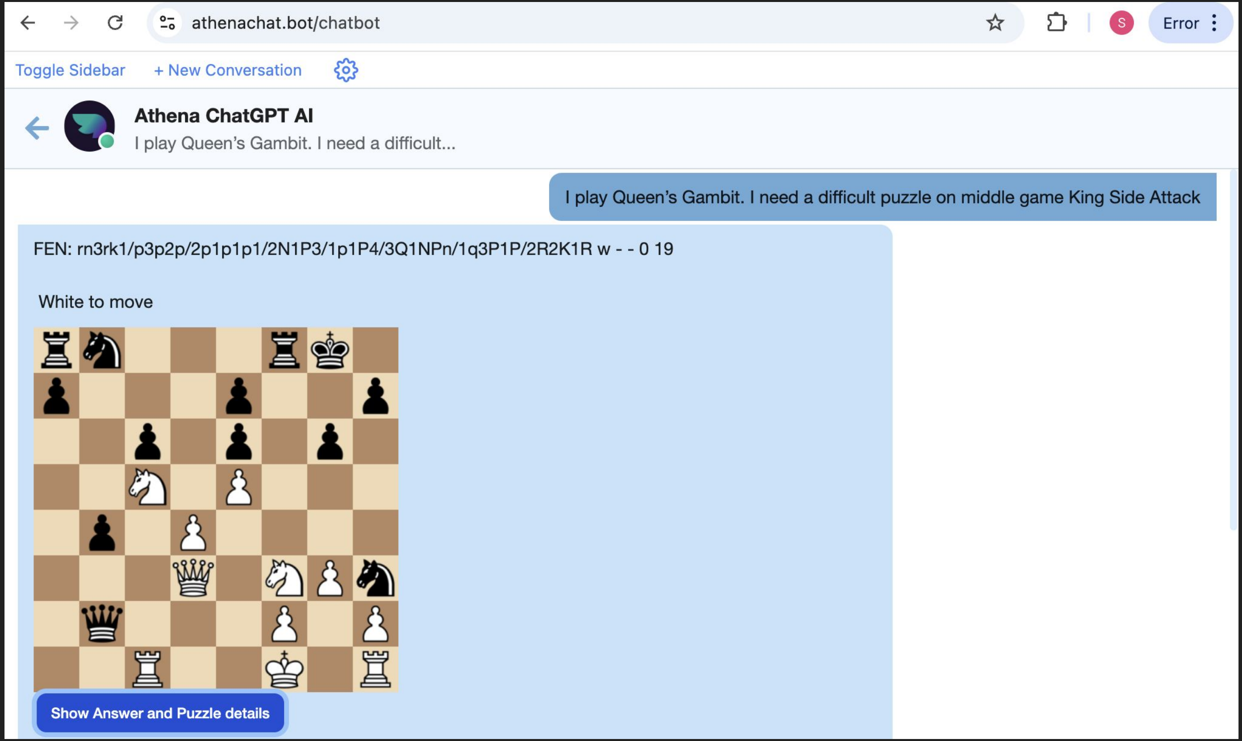
User prompt: Same as last slide.

Example: Chess puzzles recommender

Improvement on chess puzzles recommendations:

Fine-tune a model for query generation to reduce hallucinations and the amount of code in the system prompt.

Example: Chess puzzles recommender



I play Queen's Gambit. I need a difficult puzzle on middle game King Side Attack

FEN: rn3rk1/p3p2p/2p1p1p1/2N1P3/1p1P4/3Q1NPn/1q3P1P/2R2K1R w - - 0 19

White to move

Resources on other LLM techniques

Structured Output:
https://platform.openai.com/docs/guides/structured-outputs

Prompt Evaluation:

https://mirascope.com/blog/llm-as-judge/

Optimize prompts:
https://cloud.google.com/vertex-ai/generative-ai/docs/learn/prompts/prompt-optimizer

Thank You!

Please let me know if you have any questions!

My email: shaohang.he@devfortress.com

Add me on linkedin: https://www.linkedin.com/in/shao-hang-he/

Shao Hang He

Owner of DevFortress and MailMagic |
CTO at Fanstories