Workshop 4
Globalizing Knowledge: Leveraging Large Language Models to Enhance Accessibility of ETDs

ETD 2024 Conference

Yinlin Chen[1], William A. Ingram[1], Edward A. Fox[2]

{ylchen, waingram, fox}@vt.edu

Virginia Tech: [1]University Libraries, [2]Dept. of Computer Science

Today’s Topics

- Introduction
- Generative AI
- Large Language Models (LLMs)
- Retrieval-Augmented Generation (RAG)
- Demonstration
- Hands-On Exercise
- Discussion
- Q & A

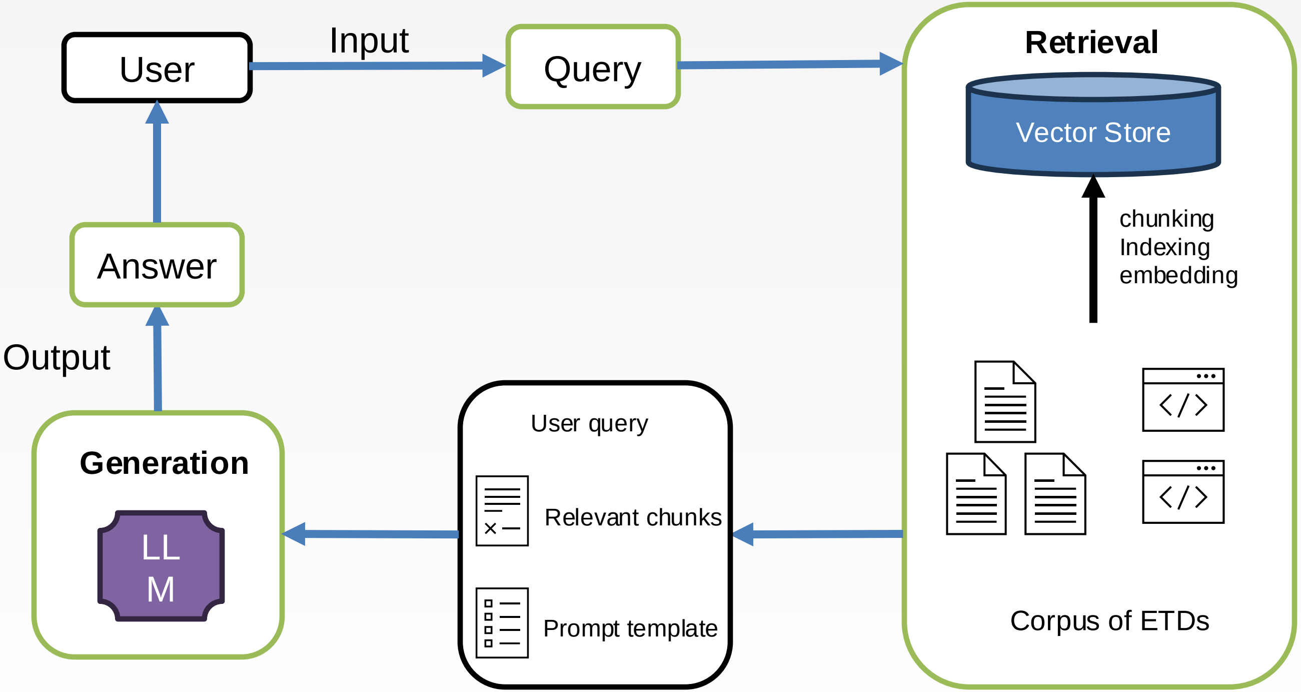
Welcome & Introduction

- Introduce yourselves
  - Name, role and institution
- Experience with AI and machine learning
- Experience with large language models
- What outcomes are you hoping for from this workshop?

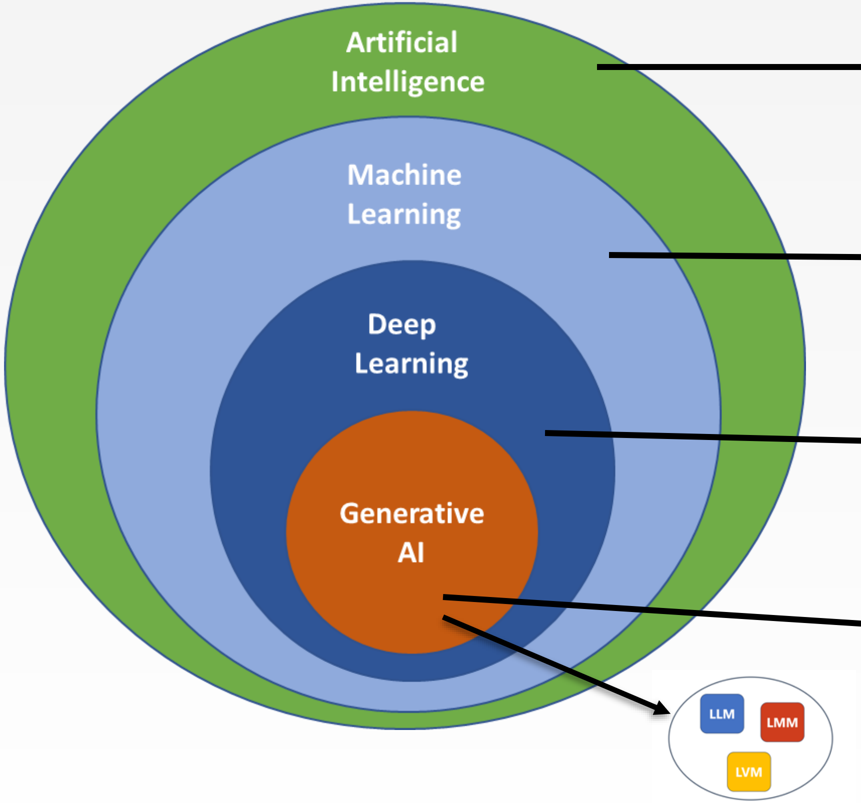
Learning Objectives

- By the end of this workshop, participants will be able to:
  - Understand the potential of Large Language Models (LLMs) in improving ETD accessibility and discoverability
  - Grasp the fundamentals of Retrieval-Augmented Generation (RAG) architecture
  - Learn how to implement LLM-based solutions for translating and standardizing ETD metadata and content
  - Develop skills to create and query unified vector databases for ETD collections
  - Gain practical experience in integrating LLM technologies into existing digital library systems

Workshop Overview



Artificial Intelligence (AI) and Generative AI (GenAI)



A computer science field that involves the science and engineering of making intelligent machines, especially intelligent computer programs

A field that focuses on developing algorithms and statistical models that enable computers to perform tasks without explicit instructions, by identifying patterns and making inferences from data.

A field that uses artificial neural networks, which mimic the structure and function of the human brain, to learn from large datasets and recognize complex patterns.

A field that focuses on creating algorithms capable of generating new, human-like content, such as text, images, and videos, based on patterns it learns from existing data.

Generative Adversarial Network

- GANs are powerful machine learning models capable of generating realistic image, video, and voice outputs
- Applications:
  - Transform photo editing
  - Generate special effects for media and entertainment
  - Contribute to creative products from industrial design to fine art
  - Augment datasets in small data problems in fields from autonomous driving to manufacturing

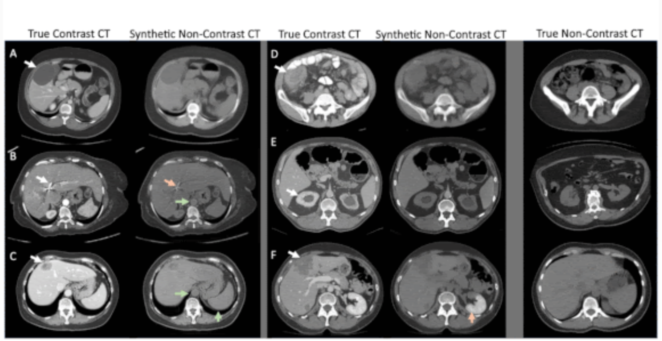


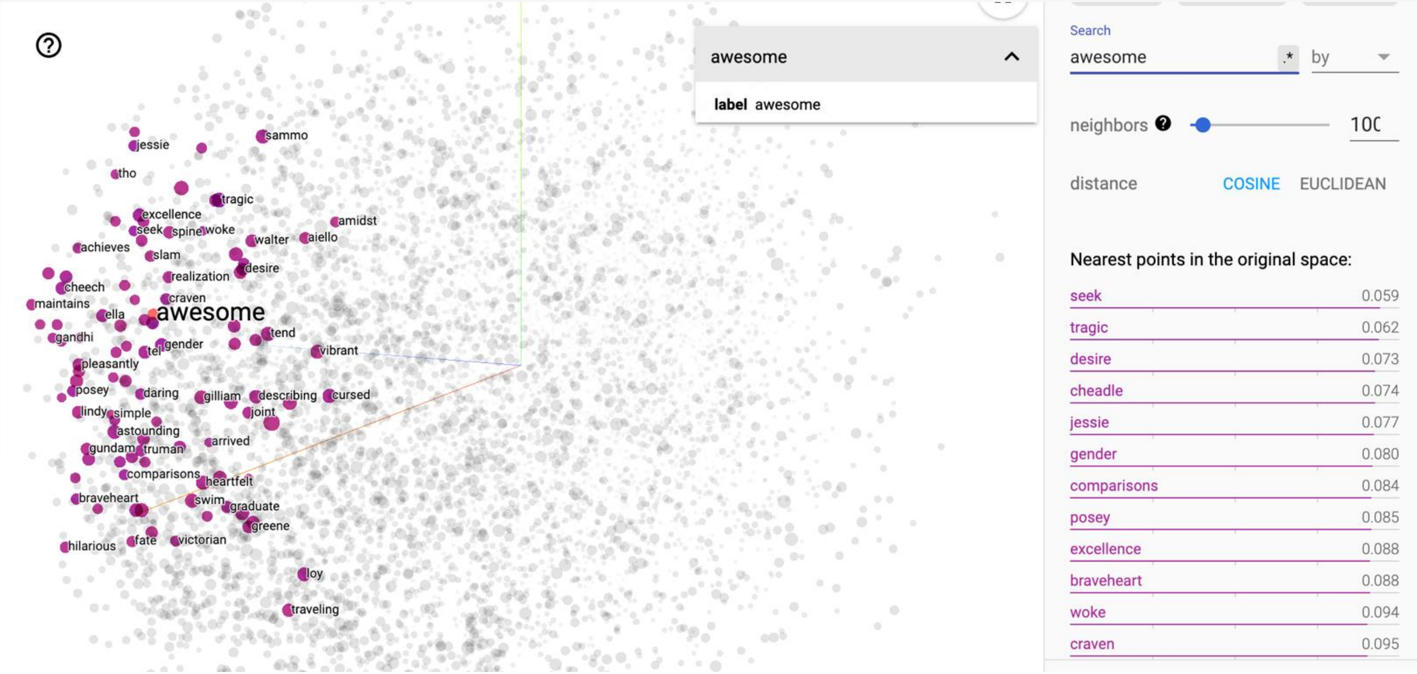
Image source: (up) https://richzhang.github.io/colorization/ (down) DeepLearning.AI

Natural Language Processing (NLP), Natural Language Understanding (NLU)

- Enables machines to understand, interpret, and generate human language
- Facilitates human-computer interaction: e.g., Alexa, Siri, Google Home, ChatGPT
- Empowers language tools: Assists in language acquisition and translations
- Drives linguistic research: Analyzes datasets to uncover linguistic patterns and trends
- Document classification, sentiment analysis, name entity recognition, etc.

Word Embedding

- Use an efficient, dense representation in which similar words have a similar encoding



Search

awesome .* by

neighbors 100

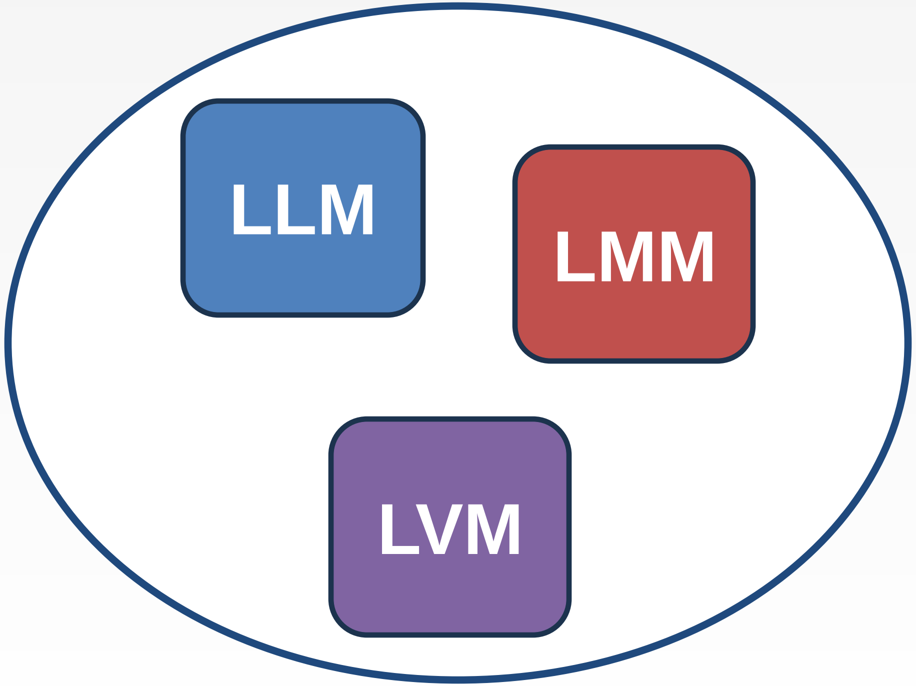
distance COSINE EUCLIDEAN

Nearest points in the original space:

seek	0.059
tragic	0.062
desire	0.073
cheadle	0.074
jessie	0.077
gender	0.080
comparisons	0.084
posey	0.085
excellence	0.088
braveheart	0.088
woke	0.094
craven	0.095

Large Language Models (LLMs) and Large Multimodal Models (LMMs)

- GPT-4 (OpenAI)
- Llama (Meta AI)
- Gemini (Google)
- Claude (Anthropic)
- Grok-1 (xAI)
- Mistral (Mistral AI)
- Phi-3 (Microsoft)
- DALL-E (OpenAI)
- Many more……



Language Model

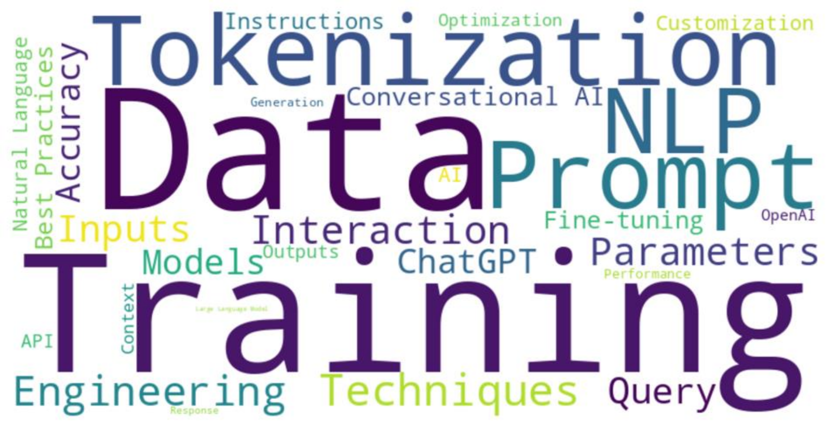
- A machine agent that understands and generates human language
- A subset of Nature Language Processing (NLP) focused on text generation and comprehension
- Uses statistical and machine learning techniques to predict and produce language sequences
- Acts as a tool within NLP for various applications like chatbots, translation services, and virtual assistants

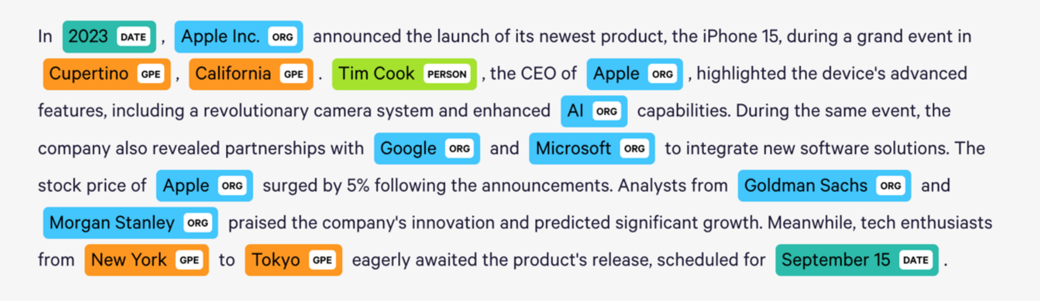
Large Language Models (LLMs)

- Predicts the probability of word sequences to generate text that mimics human speech and writing
- Trained on extensive textual data, these models excel in predicting the next word in a sentence based on the preceding words.
- Enhance the functionality of applications like chatbots, creative writing tools, translation software, and platforms that analyze customer sentiment
- Examples of such models include OpenAI's GPT-3 and GPT-4, as well as Google's BERT, among others.

NLP Tasks Where AI Models Excel

- Text Generation
- Text Summarization
- Sentiment Analysis
- Named Entity Recognition (NER)
- Question Answering
- Text Classification
- Paraphrasing
- Translation




In 2023 DATE , Apple Inc. ORG announced the launch of its newest product, the iPhone 15, during a grand event in Cupertino GPE , California GPE . Tim Cook PERSON , the CEO of Apple ORG , highlighted the device's advanced features, including a revolutionary camera system and enhanced AI ORG capabilities. During the same event, the company also revealed partnerships with Google ORG and Microsoft ORG to integrate new software solutions. The stock price of Apple ORG surged by 5% following the announcements. Analysts from Goldman Sachs ORG and Morgan Stanley ORG praised the company's innovation and predicted significant growth. Meanwhile, tech enthusiasts from New York GPE to Tokyo GPE eagerly awaited the product's release, scheduled for September 15 DATE .

Closed vs. Open Models

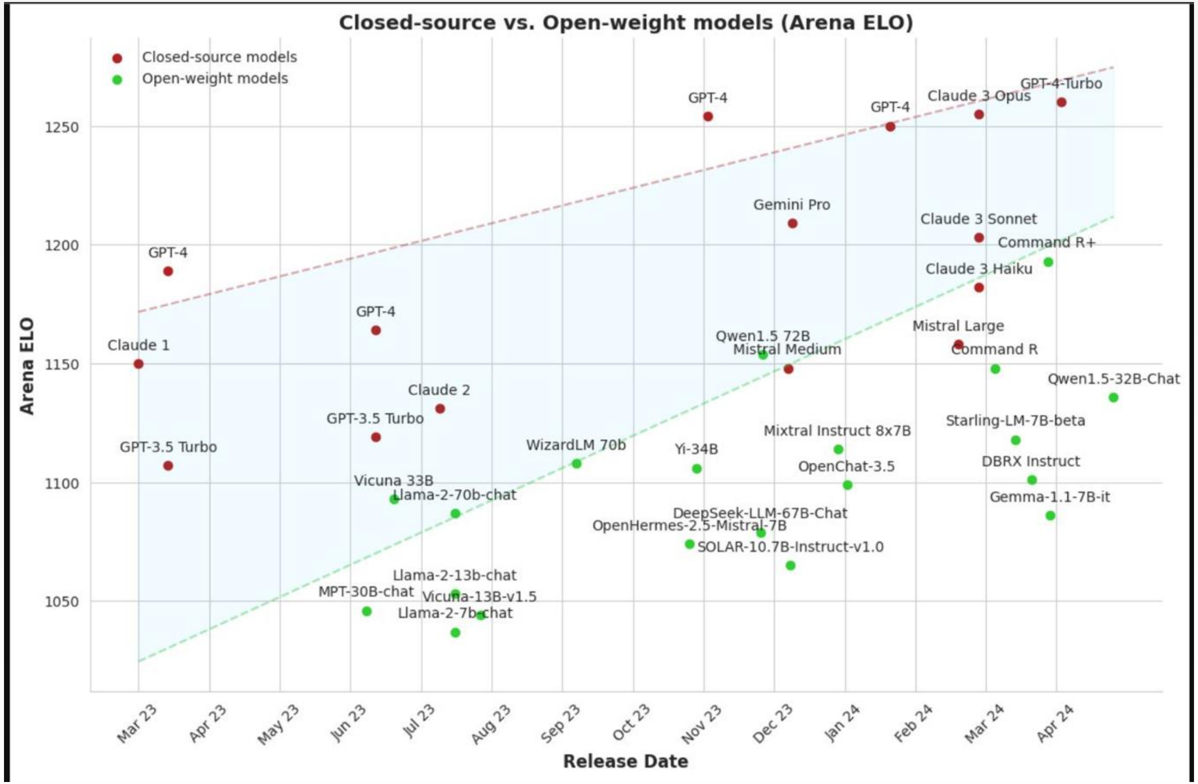
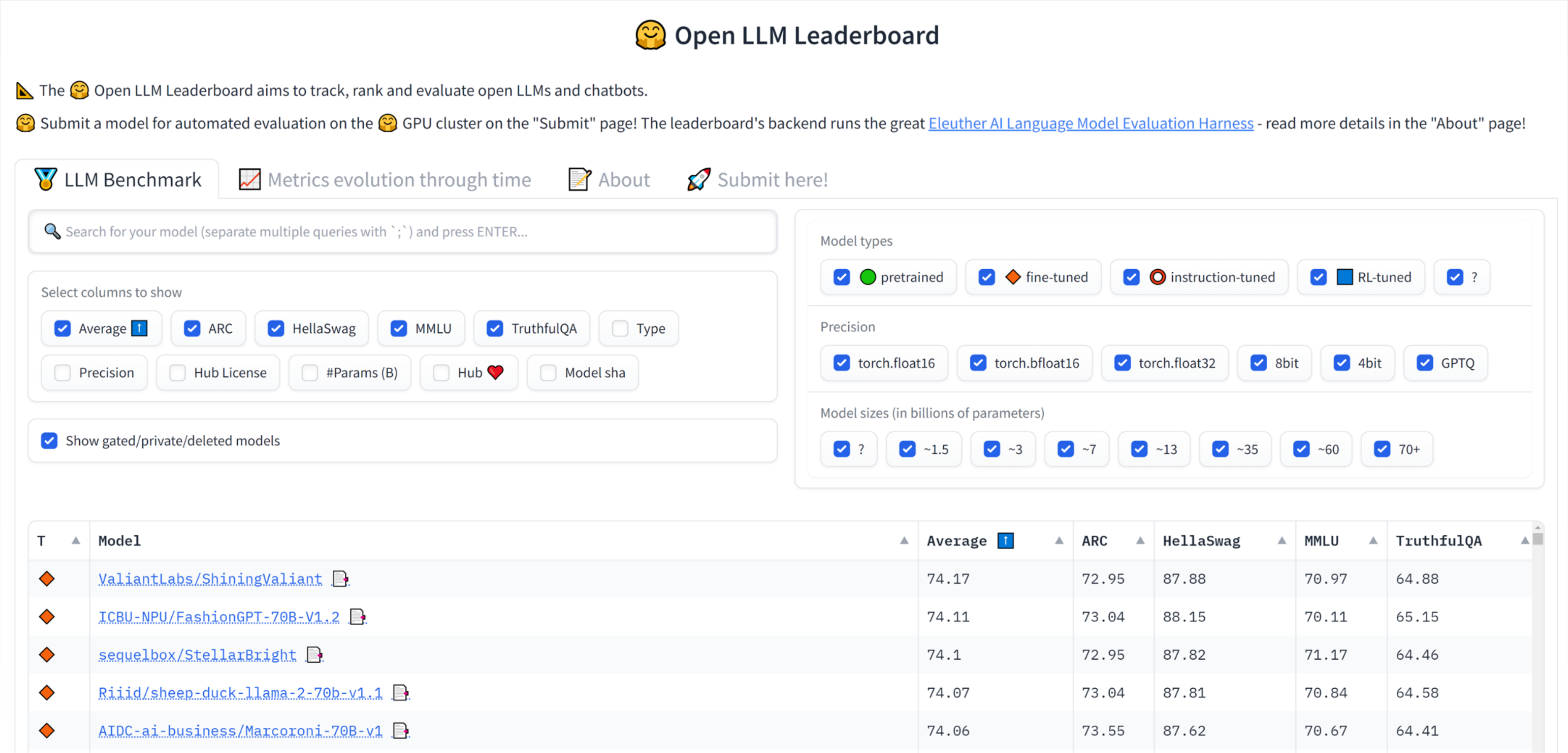


Image: https://twitter.com/maximelabonne/status/1779801605702836454



🤗 Open LLM Leaderboard

📐 The 🤗 Open LLM Leaderboard aims to track, rank and evaluate open LLMs and chatbots.

🤗 Submit a model for automated evaluation on the 🤗 GPU cluster on the "Submit" page! The leaderboard's backend runs the great Eleuther AI Language Model Evaluation Harness - read more details in the "About" page!

🏅 LLM Benchmark
📈 Metrics evolution through time
📝 About
🚀 Submit here!
🔍 Search for your model (separate multiple queries with `;`) and press ENTER...

Select columns to show

☑ Average ⬆️ ☑ ARC ☑ HellaSwag ☑ MMLU ☑ TruthfulQA ☐ Type

☐ Precision ☐ Hub License ☐ #Params (B) ☐ Hub ❤️ ☐ Model sha

☑ Show gated/private/deleted models

Model types

☑ ● pretrained ☑ ◆ fine-tuned ☑ ⭕ instruction-tuned ☑ 🟦 RL-tuned ☑ ?

Precision

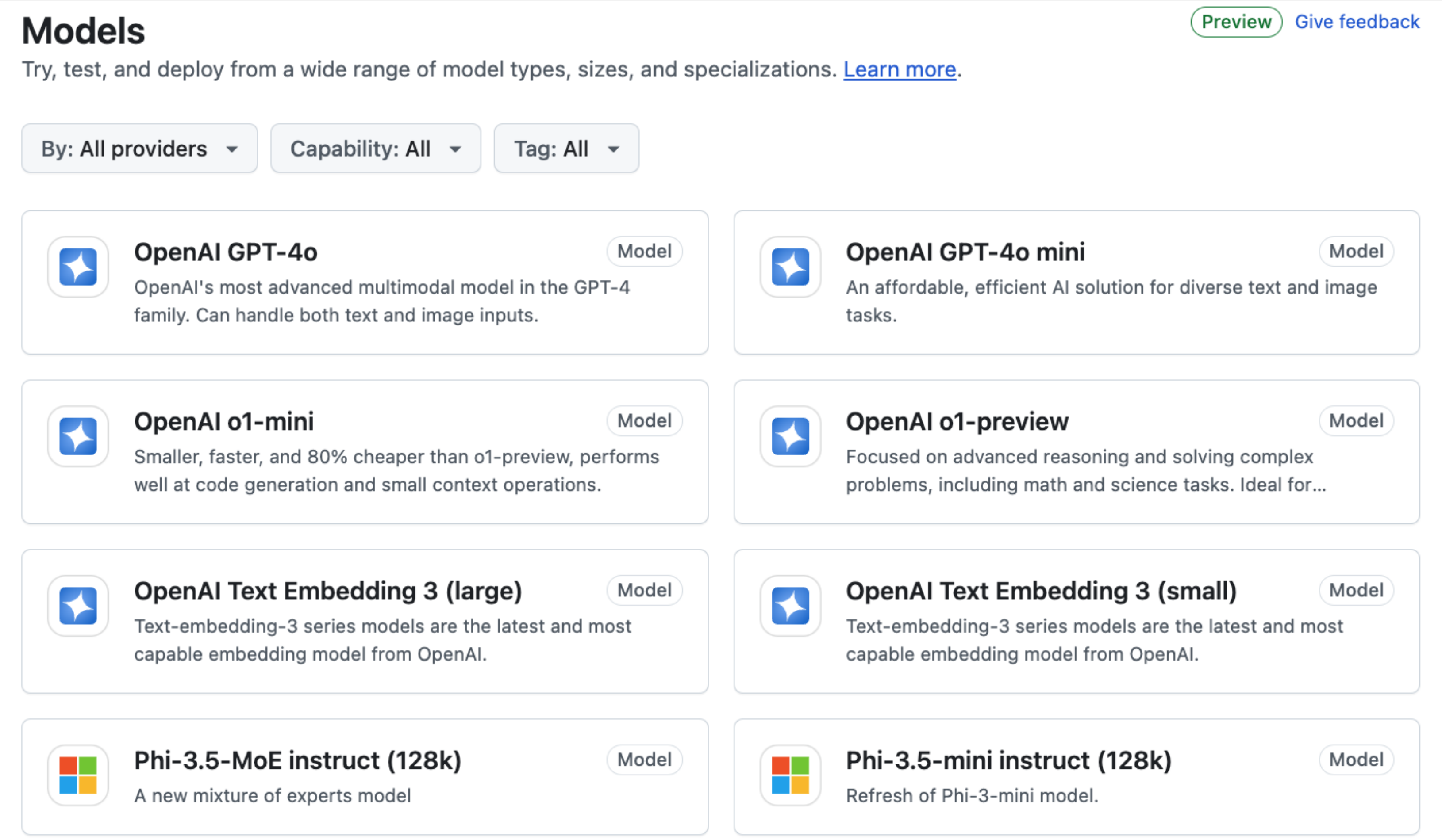
☑ torch.float16 ☑ torch.bfloat16 ☑ torch.float32 ☑ 8bit ☑ 4bit ☑ GPTQ

Model sizes (in billions of parameters)

☑ ? ☑ ~1.5 ☑ ~3 ☑ ~7 ☑ ~13 ☑ ~35 ☑ ~60 ☑ 70+

T	Model	Average ⬆️	ARC	HellaSwag	MMLU	TruthfulQA
◆	ValiantLabs/ShiningValiant	74.17	72.95	87.88	70.97	64.88
◆	ICBU-NPU/FashionGPT-70B-V1.2	74.11	73.04	88.15	70.11	65.15
◆	sequelbox/StellarBright	74.1	72.95	87.82	71.17	64.46
◆	Riiid/sheep-duck-llama-2-70b-v1.1	74.07	73.04	87.81	70.84	64.58
◆	AIDC-ai-business/Marcoroni-70B-v1	74.06	73.55	87.62	70.67	64.41

Source: https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard



Models

Try, test, and deploy from a wide range of model types, sizes, and specializations. Learn more.

By: All providers

Capability: All

Tag: All

OpenAI GPT-4o
OpenAI's most advanced multimodal model in the GPT-4 family. Can handle both text and image inputs.

OpenAI GPT-4o mini
An affordable, efficient AI solution for diverse text and image tasks.

OpenAI o1-mini
Smaller, faster, and 80% cheaper than o1-preview, performs well at code generation and small context operations.

OpenAI o1-preview
Focused on advanced reasoning and solving complex problems, including math and science tasks. Ideal for...

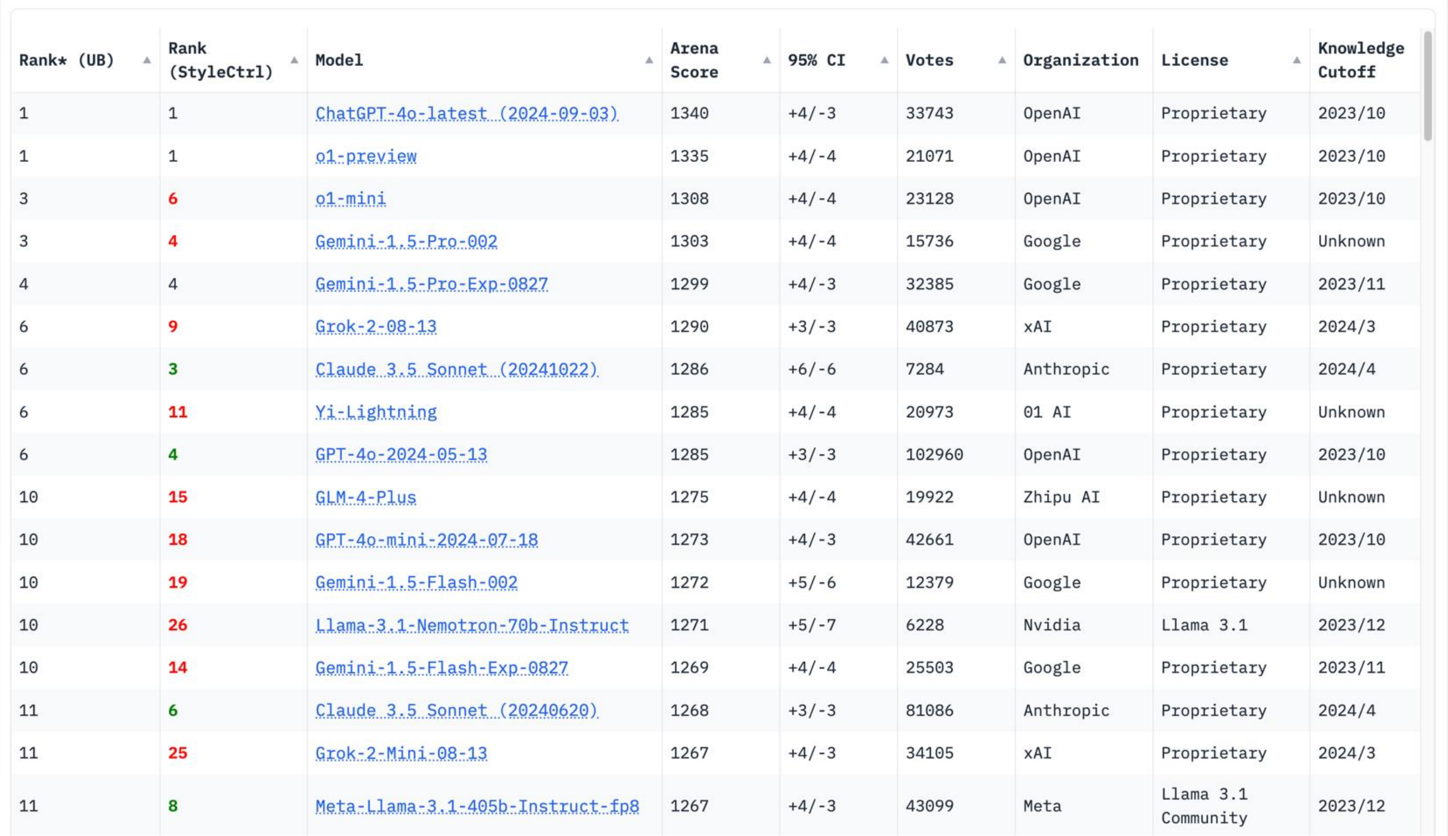
OpenAI Text Embedding 3 (large)
Text-embedding-3 series models are the latest and most capable embedding model from OpenAI.

OpenAI Text Embedding 3 (small)
Text-embedding-3 series models are the latest and most capable embedding model from OpenAI.

Phi-3.5-MoE instruct (128k)
A new mixture of experts model

Phi-3.5-mini instruct (128k)
Refresh of Phi-3-mini model.

LMSYS Chatbot Arena Leaderboard



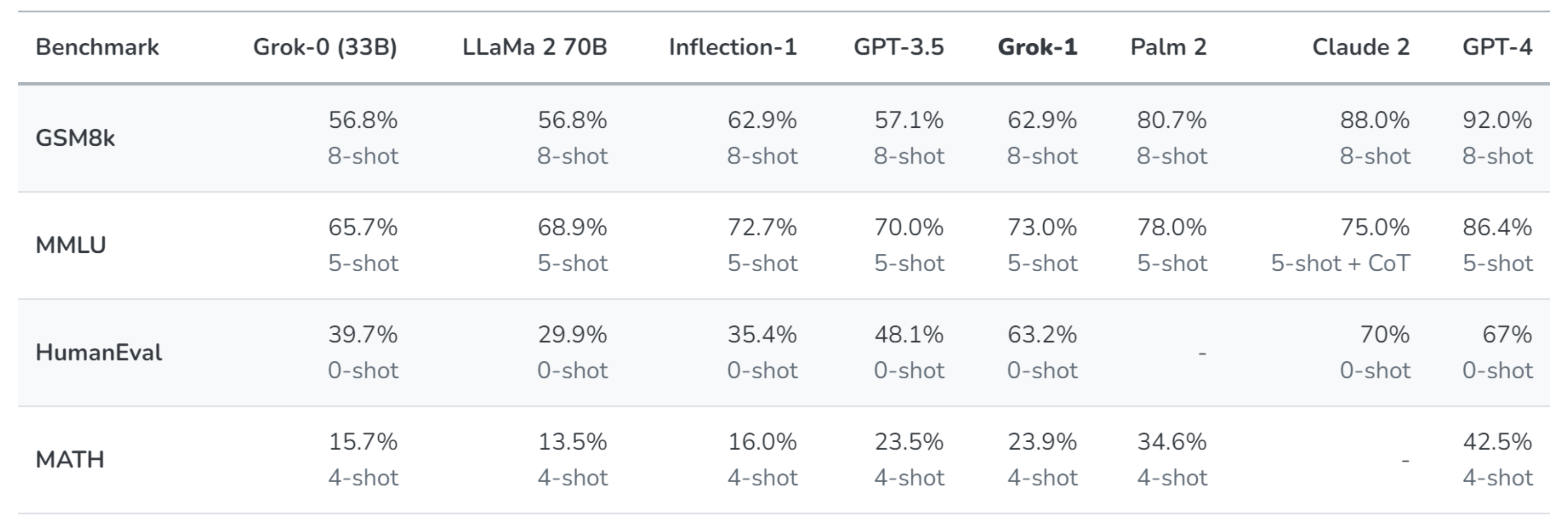
Rank* (UB)	Rank (StyleCtrl)	Model	Arena Score	95% CI	Votes	Organization	License	Knowledge Cutoff
1	1	ChatGPT-4o-latest (2024-09-03)	1340	+4/-3	33743	OpenAI	Proprietary	2023/10
1	1	o1-preview	1335	+4/-4	21071	OpenAI	Proprietary	2023/10
3	6	o1-mini	1308	+4/-4	23128	OpenAI	Proprietary	2023/10
3	4	Gemini-1.5-Pro-002	1303	+4/-4	15736	Google	Proprietary	Unknown
4	4	Gemini-1.5-Pro-Exp-0827	1299	+4/-3	32385	Google	Proprietary	2023/11
6	9	Grok-2-08-13	1290	+3/-3	40873	xAI	Proprietary	2024/3
6	3	Claude 3.5 Sonnet (20241022)	1286	+6/-6	7284	Anthropic	Proprietary	2024/4
6	11	Yi-Lightning	1285	+4/-4	20973	01 AI	Proprietary	Unknown
6	4	GPT-4o-2024-05-13	1285	+3/-3	102960	OpenAI	Proprietary	2023/10
10	15	GLM-4-Plus	1275	+4/-4	19922	Zhipu AI	Proprietary	Unknown
10	18	GPT-4o-mini-2024-07-18	1273	+4/-3	42661	OpenAI	Proprietary	2023/10
10	19	Gemini-1.5-Flash-002	1272	+5/-6	12379	Google	Proprietary	Unknown
10	26	Llama-3.1-Nemotron-70b-Instruct	1271	+5/-7	6228	Nvidia	Llama 3.1	2023/12
10	14	Gemini-1.5-Flash-Exp-0827	1269	+4/-4	25503	Google	Proprietary	2023/11
11	6	Claude 3.5 Sonnet (20240620)	1268	+3/-3	81086	Anthropic	Proprietary	2024/4
11	25	Grok-2-Mini-08-13	1267	+4/-3	34105	xAI	Proprietary	2024/3
11	8	Meta-Llama-3.1-405b-Instruct-fp8	1267	+4/-3	43099	Meta	Llama 3.1 Community	2023/12

Source: https://chat.lmsys.org/?leaderboard

LLM Capabilities and Features

- Agents
- Conversation
- Multilingual
- Multimodal
- Multipurpose
- Reasoning
- Understanding

Model Benchmark



Benchmark	Grok-0 (33B)	LLaMa 2 70B	Inflection-1	GPT-3.5	**Grok-1**	Palm 2	Claude 2	GPT-4
GSM8k	56.8% 8-shot	56.8% 8-shot	62.9% 8-shot	57.1% 8-shot	62.9% 8-shot	80.7% 8-shot	88.0% 8-shot	92.0% 8-shot
MMLU	65.7% 5-shot	68.9% 5-shot	72.7% 5-shot	70.0% 5-shot	73.0% 5-shot	78.0% 5-shot	75.0% 5-shot + CoT	86.4% 5-shot
HumanEval	39.7% 0-shot	29.9% 0-shot	35.4% 0-shot	48.1% 0-shot	63.2% 0-shot	-	70% 0-shot	67% 0-shot
MATH	15.7% 4-shot	13.5% 4-shot	16.0% 4-shot	23.5% 4-shot	23.9% 4-shot	34.6% 4-shot	-	42.5% 4-shot

Source: https://x.ai/

LLMs for Human-Computer Interaction

- Interpret and respond to natural language input.
- Engage in dynamic dialogues with users.
- Maintain context over multi-turn conversations.
- Provide personalized responses based on user preferences.
- Offer real-time language translation and multilingual support.
- Integrate with external systems for enhanced interactivity.
- Learn and adapt from user interactions to improve over time.

LLM language support

- Different models have different language capabilities.
- Most major LLMs are trained predominantly on English data.
- Primary languages (like English, Spanish, French) typically have strong support.
- Less common languages often have poor performance or no support.
- Model-Specific Variations:
  - GPT models show strong performance in widely-spoken languages.
  - Open-source models may have more limited language support.
  - Specialized models may focus on specific language families or regions.

Chatbot Limitations

- **Context Limitation:** Can only consider a fixed amount of text at a time, potentially missing broader context.
- **Usage Limit**: ChatGPT (GPT-4) allows 40 messages every 3 hours, while Copilot permits 30 requests per topic.
- **Outdated Knowledge:** Knowledge stops at the last training update, with no updates on newer advancements or changes.
- **Data Dependency:** Model performance directly correlates to the quality and breadth of its training data.
- **Input Sensitive:** Output quality heavily relies on the specificity and clarity of user prompts.
- **Hallucinations:** May confidently generate incorrect or misleading information.

Augmentation and Optimization Methods of LLMs

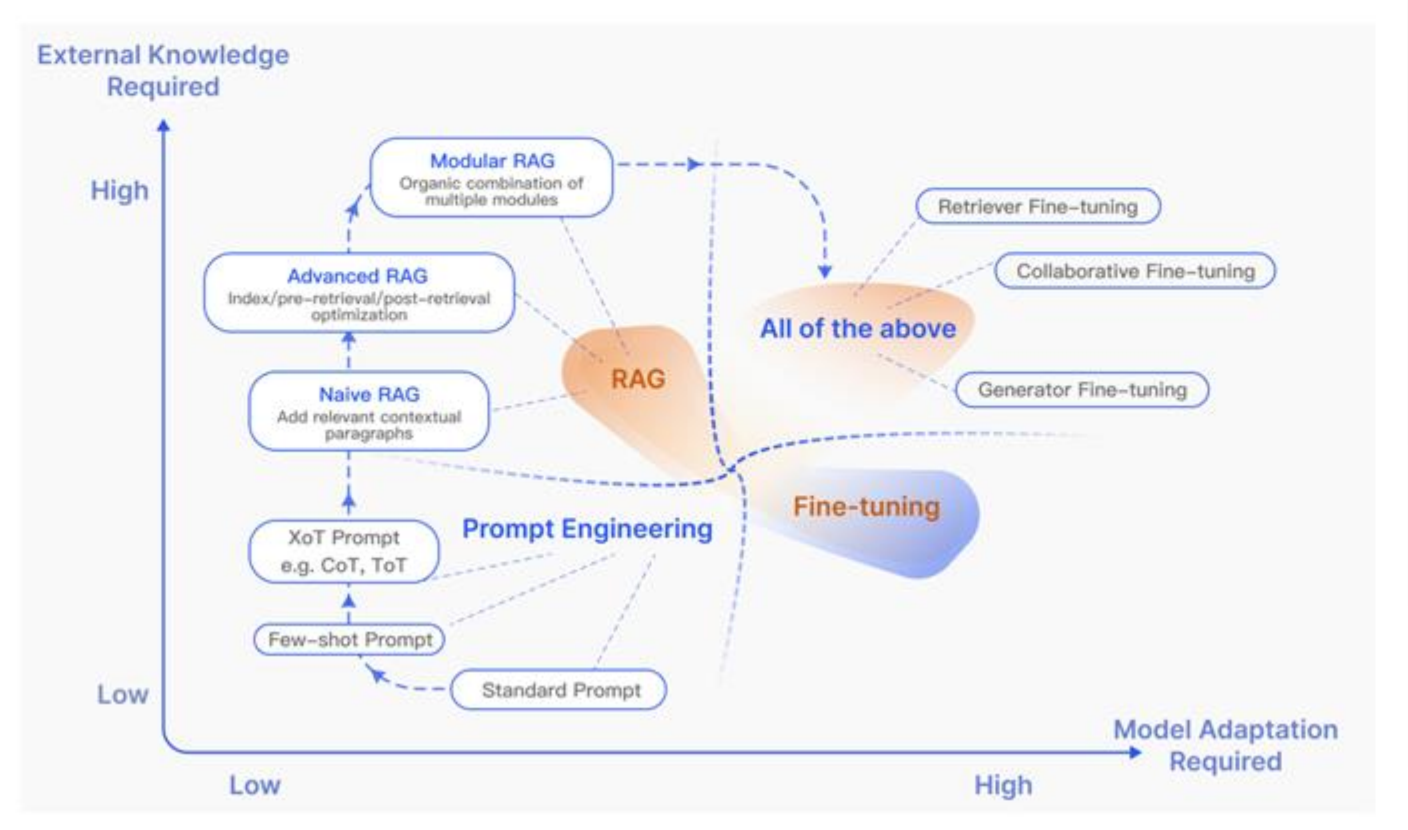

External Knowledge Required
High
Low
Modular RAG
Organic combination of multiple modules
Advanced RAG
Index/pre-retrieval/post-retrieval optimization
Naive RAG
Add relevant contextual paragraphs
XoT Prompt
e.g. CoT, ToT
Few-shot Prompt
Standard Prompt
Prompt Engineering
RAG
All of the above
Fine-tuning
Retriever Fine-tuning
Collaborative Fine-tuning
Generator Fine-tuning
Model Adaptation Required
Low
High

Image source: Retrieval-Augmented Generation for Large Language Models: A Survey

Prompt engineering vs RAG vs Fine-tuning

Prompt engineering

- Crafting prompts to elicit desired responses from a language model
- Requires understanding of the model's behavior and capabilities
- Often involves iterative testing and refining of prompts

RAG

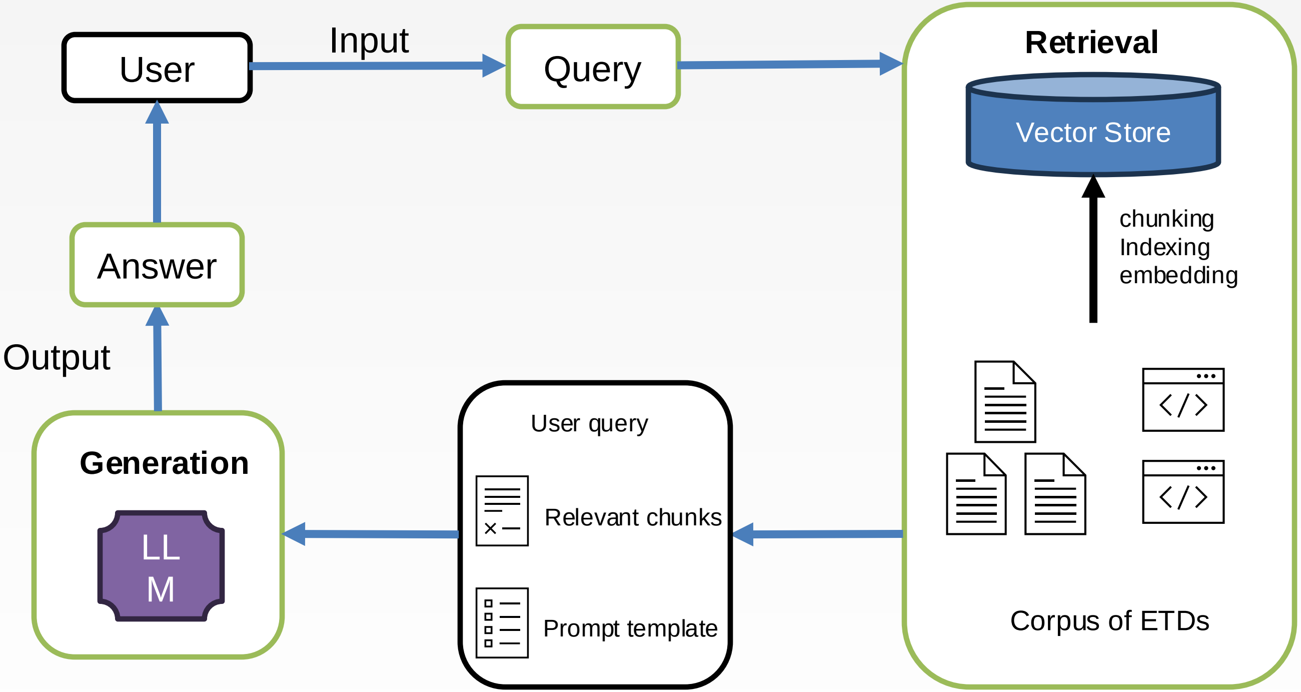
- Integrate external knowledge bases with LLM
- Access real-time data not included in the LLM's training set
- Reduce the likelihood of hallucinations

Fine-tuning

- Adjusts a pre-trained language model using a smaller, specialized dataset
- Adapts the model to specific tasks or domains
- Improves performance on the target task
- More expensive than using a pre-trained model

Retrieval-Augmented Generation (RAG)

Retrieval-Augmented Generation (RAG) Overview



Retrieval-Augmented Generation (RAG)

- RAG is valuable for use cases requiring knowledge beyond the model's pre-trained information.
- It enables semantic queries based on user input.
- Prompts guide both the retrieval process and the generation of responses.
- By incorporating relevant information, RAG improves response quality and reduces hallucinations.

Retrieval-Augmented Generation (RAG)

- Retrieval: The system retrieves relevant documents or information from a large corpus or database based on a query.

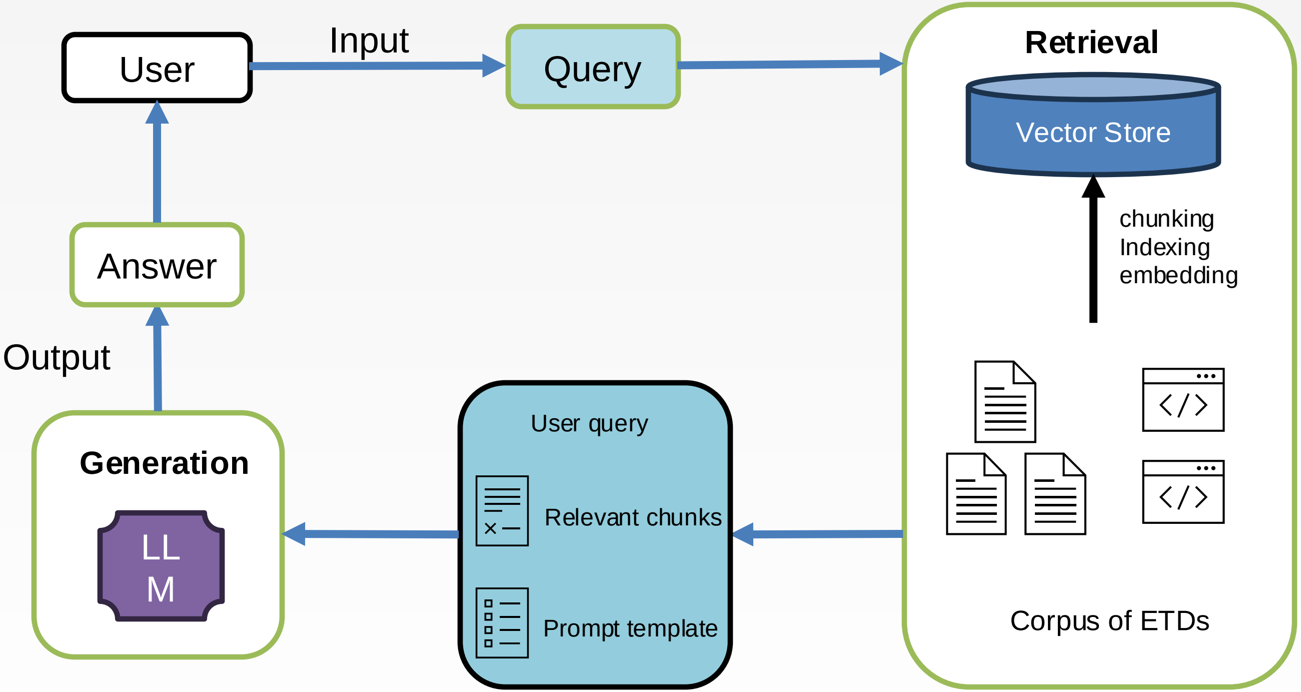
- Augmentation: The retrieved information is then augmented into the generative model’s input. This augmentation helps the model avoid hallucinations and produce better-informed results.

- Generation: The generative model creates an output using both its internal knowledge and the retrieved information, which improves the quality of responses, especially for complex or knowledge-intensive queries.

Key Skills for RAG Implementation

- Programming (Python)
- Prompt Engineering
- Retrieval Techniques (e.g., BM25, TF-IDF, embedding-based retrieval, etc.)
- Natural Language Processing (NLP)

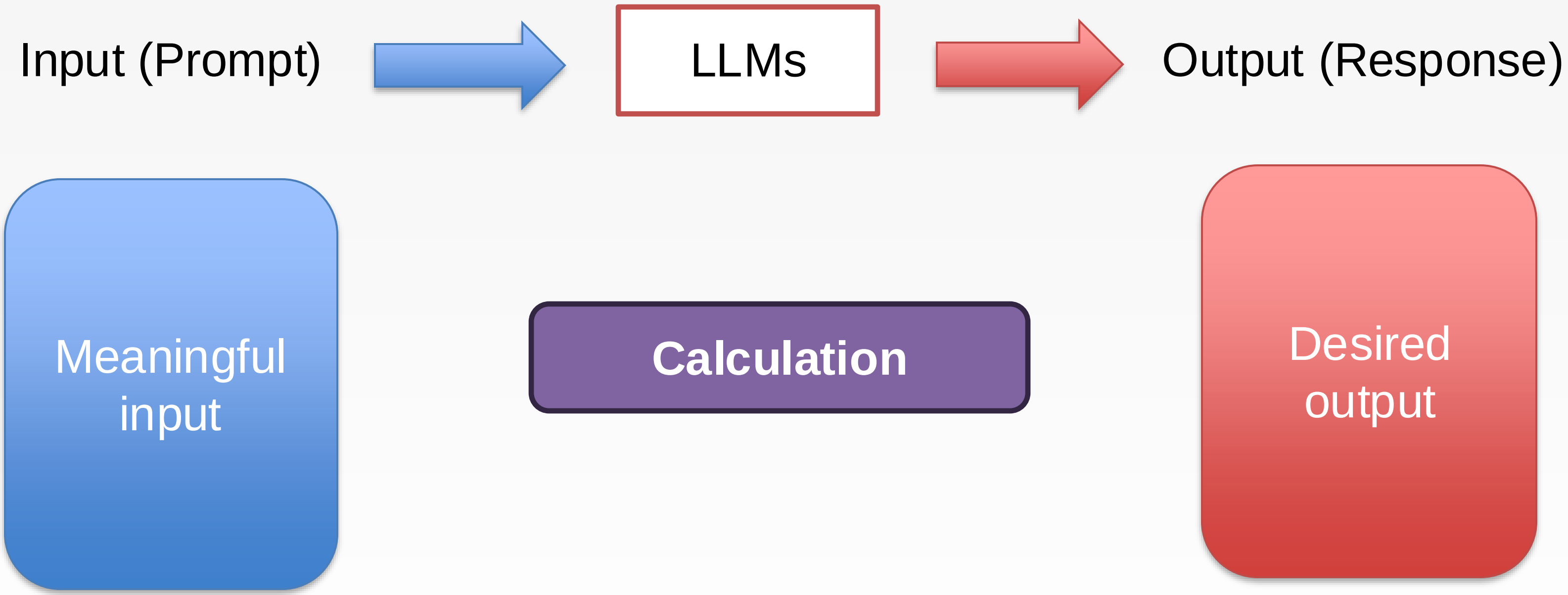
Retrieval-Augmented Generation (RAG) Overview



Prompt Engineering

- A skill of crafting prompts to elicit the desired responses or results from models.
- Crafting prompts aligned with the model's capabilities and limitations to maximize effectiveness.
- Employing a result-oriented communication approach to lead to the intended outcome.
- Incorporating contextual information within prompts to guide model understanding.
- Including examples within prompts to guide response generation.
- Guiding formatting instructions to shape the output.
- Specifying personas to instruct language models on behavior or perspective.
- Critically refining prompts based on model responses to optimize relevance and accuracy iteratively.

Workflow in LLM-based Application



Input (Prompt)

LLMs

Output (Response)

Meaningful input

Calculation

Desired output

Prompt (Input)

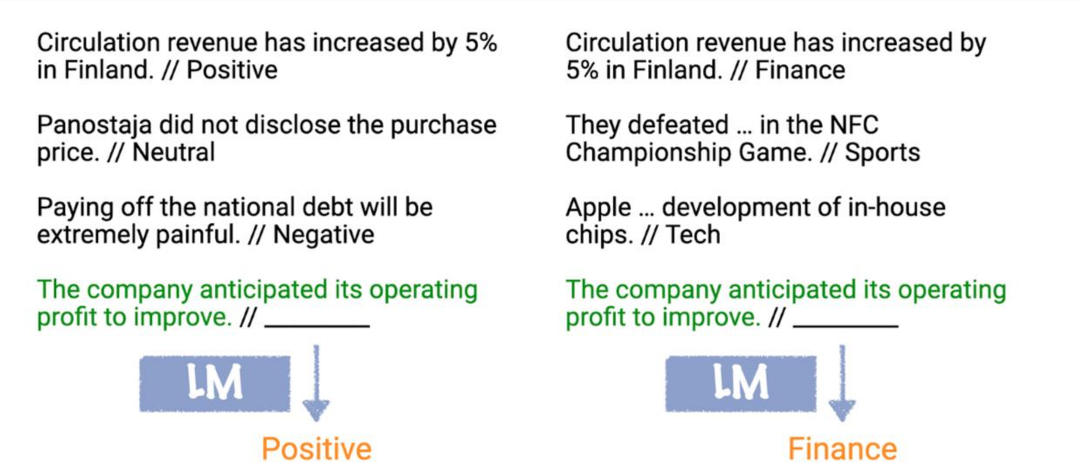
- Serves as the input method for Language Models
- **Describes** the task to be performed by the model
- Supplies necessary **context** for generating relevant responses
- May include **formatting instructions** for the output
- Specifies a **role** to instruct LLMs on behavior or perspective
- May contain examples to **guide** response generation

Prompt Techniques

- **Prompt-Based**: User provides a "prompt" or initial input, and the model generates a continuation.
- **Zero-Shot**: Model makes predictions about a task without seeing examples of the task during training.
- **Few-Shot**: User provides a few examples of the desired task within the prompt, and the model generalizes from these examples to complete the task.
- **Chain-of-Thought (CoT)**: Enables reasoning via intermediate steps. When combined with few-shot prompting, it improves results on tasks needing complex reasoning.

In Context Learning (ICL)

- Methods that enhance capabilities without adjusting (fine-tuning) the underlying model.
  - Fine-tuning involves modifying the original model for specific tasks, which can be costly.
- Perform a new task from directly learning a small set of examples presented within the prompt
- Utilizes off-the-shelf LLMs, thus ensuring broad applicability and ease of integration.



Circulation revenue has increased by 5% in Finland. // Positive

Panostaja did not disclose the purchase price. // Neutral

Paying off the national debt will be extremely painful. // Negative

The company anticipated its operating profit to improve. // ________

Circulation revenue has increased by 5% in Finland. // Finance

They defeated ... in the NFC Championship Game. // Sports

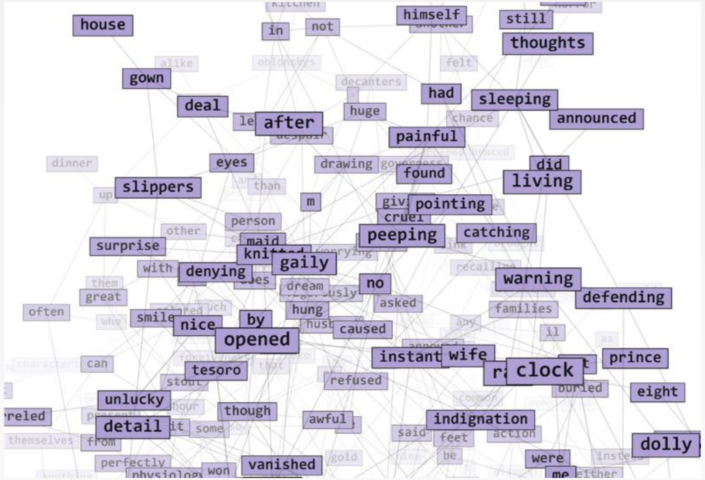
Apple ... development of in-house chips. // Tech

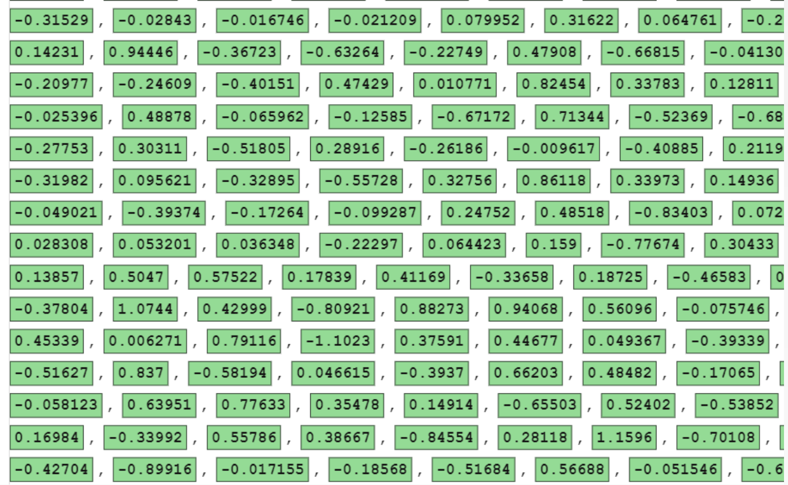
The company anticipated its operating profit to improve. // ________

Image: https://ai.stanford.edu/blog/understanding-incontext/

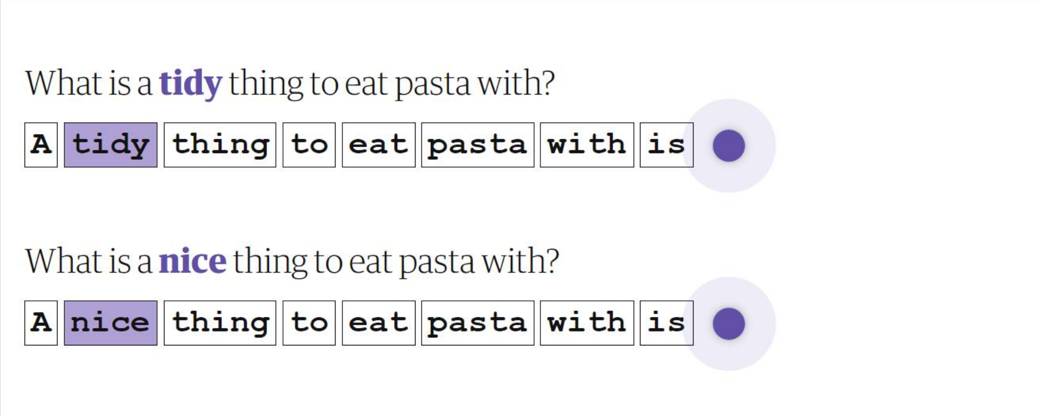
Embeddings in Language Models

- **Semantic Representation**: for the model to understand language nuances, context, and meaning.
- **Contextual Understanding**: for the model to grasp the dictionary meaning of words and their specific meanings in different sentences and scenarios.
- **Generalization**: for the model to apply learned information about one word to similar words, thereby improving its ability to handle new or related vocabulary.
- **Transfer Learning**: For the model to efficiently apply knowledge gained from one task to another, enhancing the model’s versatility across various applications and languages.





```
-0.31529 , -0.02843 , -0.016746 , -0.021209 , 0.079952 , 0.31622 , 0.064761 , -0.2
0.14231 , 0.94446 , -0.36723 , -0.63264 , -0.22749 , 0.47908 , -0.66815 , -0.04130
-0.20977 , -0.24609 , -0.40151 , 0.47429 , 0.010771 , 0.82454 , 0.33783 , 0.12811
-0.025396 , 0.48878 , -0.065962 , -0.12585 , -0.67172 , 0.71344 , -0.52369 , -0.68
-0.27753 , 0.30311 , -0.51805 , 0.28916 , -0.26186 , -0.009617 , -0.40885 , 0.2119
-0.31982 , 0.095621 , -0.32895 , -0.55728 , 0.32756 , 0.86118 , 0.33973 , 0.14936
-0.049021 , -0.39374 , -0.17264 , -0.099287 , 0.24752 , 0.48518 , -0.83403 , 0.072
0.028308 , 0.053201 , 0.036348 , -0.22297 , 0.064423 , 0.159 , -0.77674 , 0.30433
0.13857 , 0.5047 , 0.57522 , 0.17839 , 0.41169 , -0.33658 , 0.18725 , -0.46583 , 0
-0.37804 , 1.0744 , 0.42999 , -0.80921 , 0.88273 , 0.94068 , 0.56096 , -0.075746 ,
0.45339 , 0.006271 , 0.79116 , -1.1023 , 0.37591 , 0.44677 , 0.049367 , -0.39339 ,
-0.51627 , 0.837 , -0.58194 , 0.046615 , -0.3937 , 0.66203 , 0.48482 , -0.17065 ,
-0.058123 , 0.63951 , 0.77633 , 0.35478 , 0.14914 , -0.65503 , 0.52402 , -0.53852
0.16984 , -0.33992 , 0.55786 , 0.38667 , -0.84554 , 0.28118 , 1.1596 , -0.70108 ,
-0.42704 , -0.89916 , -0.017155 , -0.18568 , -0.51684 , 0.56688 , -0.051546 , -0.6
```

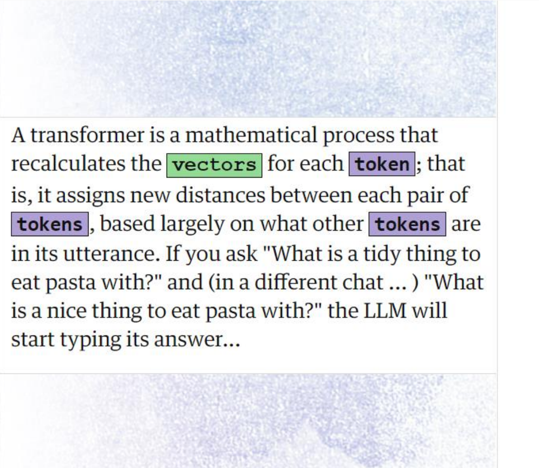


What is a **tidy** thing to eat pasta with?

`A` `tidy` `thing` `to` `eat` `pasta` `with` `is` ●

What is a **nice** thing to eat pasta with?

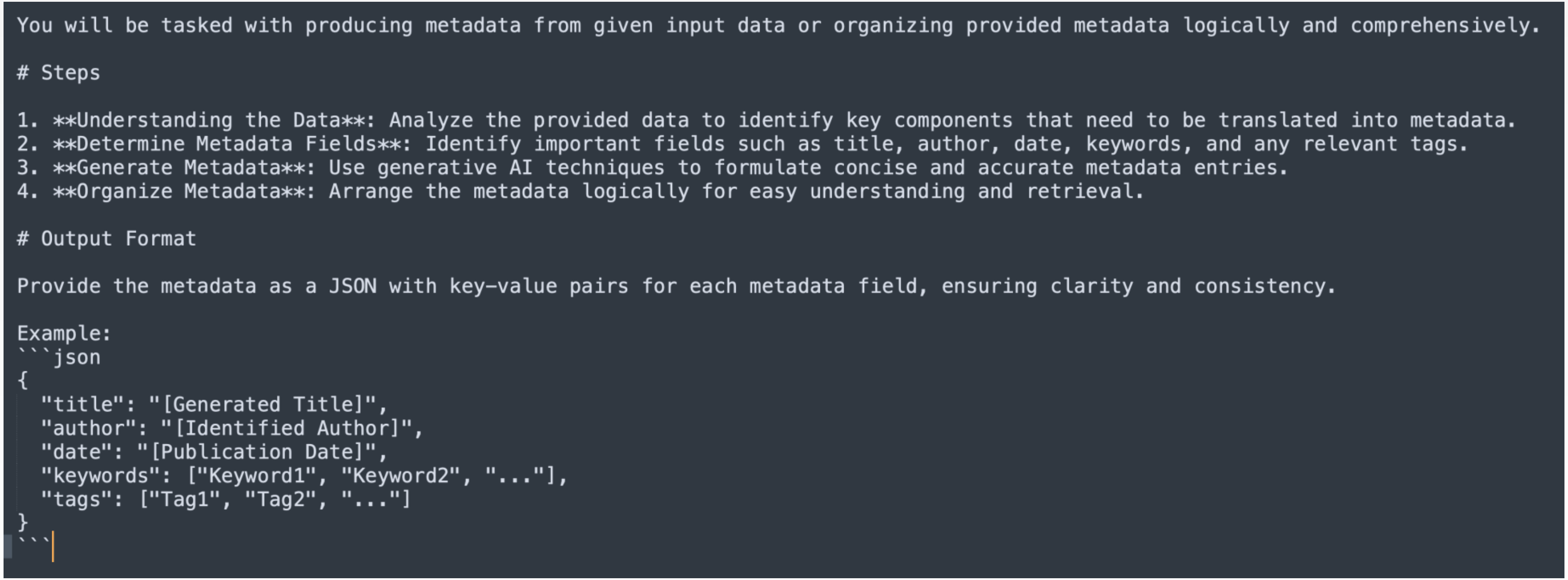
`A` `nice` `thing` `to` `eat` `pasta` `with` `is` ●



A transformer is a mathematical process that recalculates the `vectors` for each `token`; that is, it assigns new distances between each pair of `tokens`, based largely on what other `tokens` are in its utterance. If you ask "What is a tidy thing to eat pasta with?" and (in a different chat ...) "What is a nice thing to eat pasta with?" the LLM will start typing its answer...

* https://www.theguardian.com/technology/ng-interactive/2023/nov/01/how-ai-chatbots-like-chatgpt-or-bard-work-visual-explainer

Prompt Example



~~~
You will be tasked with producing metadata from given input data or organizing provided metadata logically and comprehensively.

# Steps

1. **Understanding the Data**: Analyze the provided data to identify key components that need to be translated into metadata.
2. **Determine Metadata Fields**: Identify important fields such as title, author, date, keywords, and any relevant tags.
3. **Generate Metadata**: Use generative AI techniques to formulate concise and accurate metadata entries.
4. **Organize Metadata**: Arrange the metadata logically for easy understanding and retrieval.

# Output Format

Provide the metadata as a JSON with key-value pairs for each metadata field, ensuring clarity and consistency.

Example:
```json
{
  "title": "[Generated Title]",
  "author": "[Identified Author]",
  "date": "[Publication Date]",
  "keywords": ["Keyword1", "Keyword2", "..."],
  "tags": ["Tag1", "Tag2", "..."]
}
```
~~~

Basic Prompt Techniques

- **Zero-Shot**
  - Palatucci, M., Pomerleau, D., Hinton, G. E., & Mitchell, T. M. (2009). Zero-shot learning with semantic output codes. Advances in neural information processing systems, 22.
- **One-Shot**
- **Few-Shot**
  - Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. Advances in neural information processing systems, 33, 1877-1901.

Zero-Shot Prompting

- **Direct Prompting**: Issue a request without providing examples.
- **Model-Dependent**: Relies entirely on the pre-trained knowledge of the model.
- **Advantages**: Immediate use, no requirement for example data.
- **Limitations**: You may receive unpredictable or less accurate responses.
- **Use Cases**: Ideal for straightforward tasks or general queries.

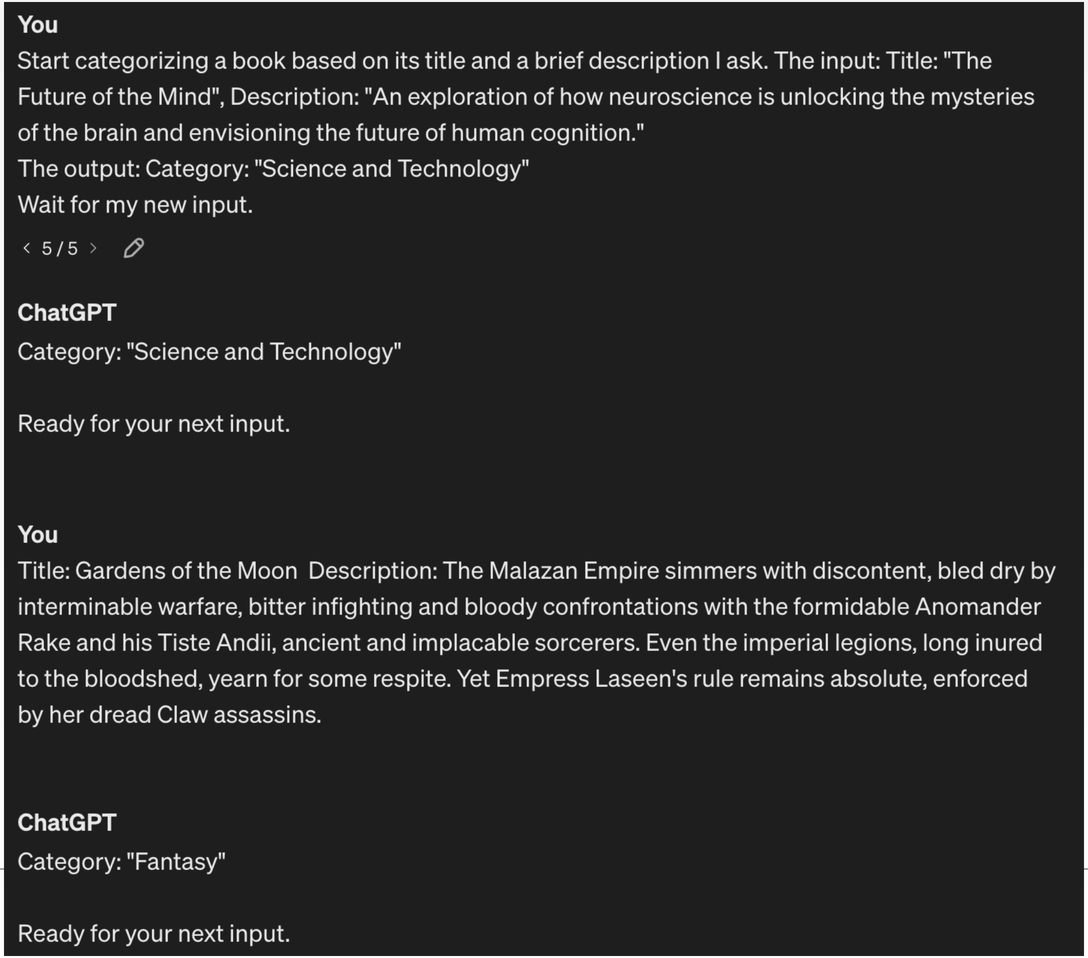
Zero-Shot Examples

- "Given a title and a brief description, categorize this book into one of the following genres: fiction, non-fiction, science, history, or biography."
- "Provide a summary explanation suitable for an 8th grader about why the Earth orbits the sun."
- "List potential themes and activities for a summer reading program aimed at teenagers, focusing on adventure and mystery genres."
- "Suggest three books similar to 'Harry Potter' that are suitable for readers aged 10-12, highlighting magical themes and strong character development."
- "Draft a proposal for a library workshop that teaches basic digital literacy skills to senior citizens, including course objectives and expected outcomes."

One-Shot Prompting

- **Single Example Guidance**: Uses one example to direct the model's response.
- **In-Context Learning**: Utilizes the context from the provided example and pre-trained knowledge.
- **Advantages**: Provides a balance between ease of use and accuracy with just one example.
- **Limitations**: Depends on the quality and relevance of the single example provided.
- **Use Cases**: For tasks needing responses in a particular style. Offers more guidance than zero-shot but less than few-shot.

One-Shot Example



You

Start categorizing a book based on its title and a brief description I ask. The input: Title: "The Future of the Mind", Description: "An exploration of how neuroscience is unlocking the mysteries of the brain and envisioning the future of human cognition."
The output: Category: "Science and Technology"
Wait for my new input.

ChatGPT

Category: "Science and Technology"

Ready for your next input.

You

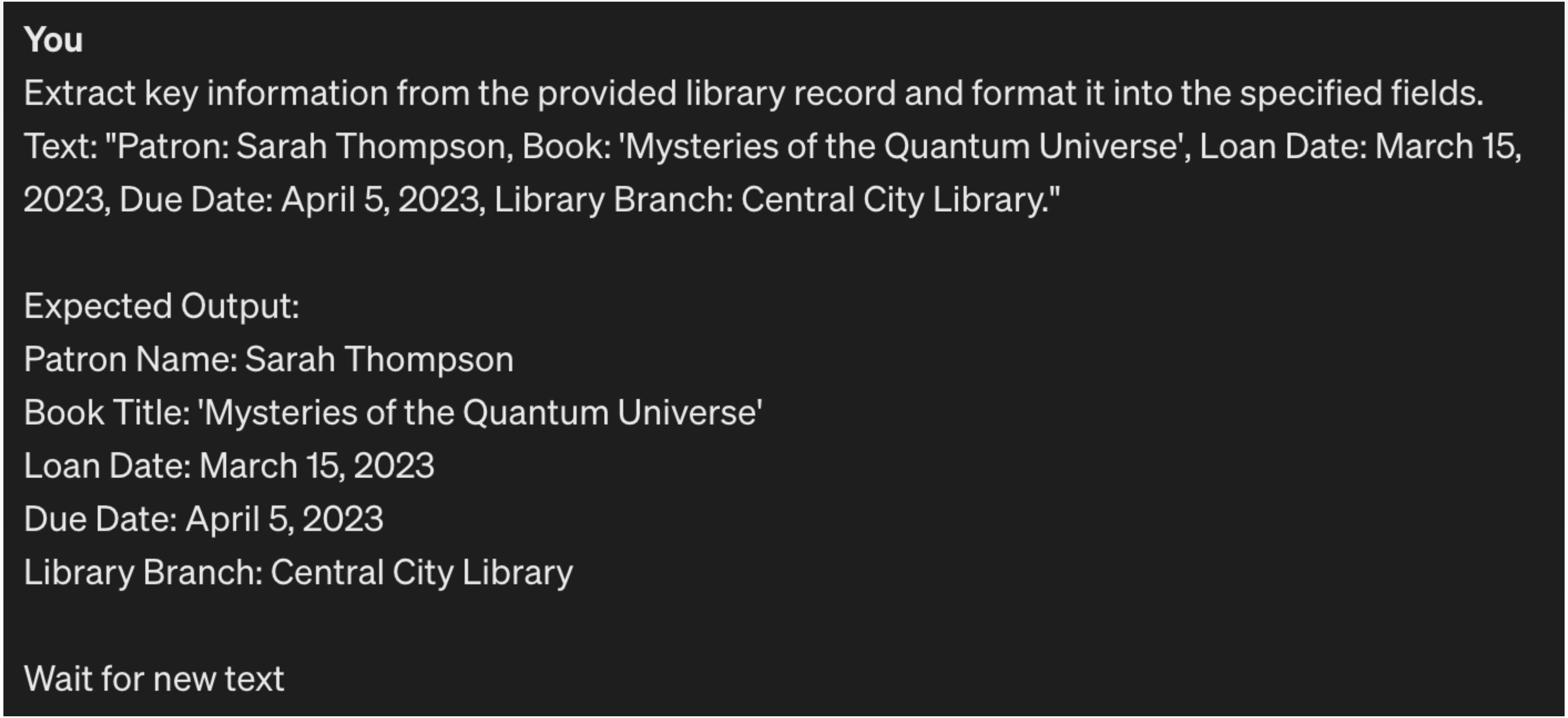
Title: Gardens of the Moon Description: The Malazan Empire simmers with discontent, bled dry by interminable warfare, bitter infighting and bloody confrontations with the formidable Anomander Rake and his Tiste Andii, ancient and implacable sorcerers. Even the imperial legions, long inured to the bloodshed, yearn for some respite. Yet Empress Laseen's rule remains absolute, enforced by her dread Claw assassins.

ChatGPT

Category: "Fantasy"

Ready for your next input.

One-Shot Example (cont’d)



You

Extract key information from the provided library record and format it into the specified fields.

Text: "Patron: Sarah Thompson, Book: 'Mysteries of the Quantum Universe', Loan Date: March 15, 2023, Due Date: April 5, 2023, Library Branch: Central City Library."

Expected Output:

Patron Name: Sarah Thompson

Book Title: 'Mysteries of the Quantum Universe'

Loan Date: March 15, 2023

Due Date: April 5, 2023

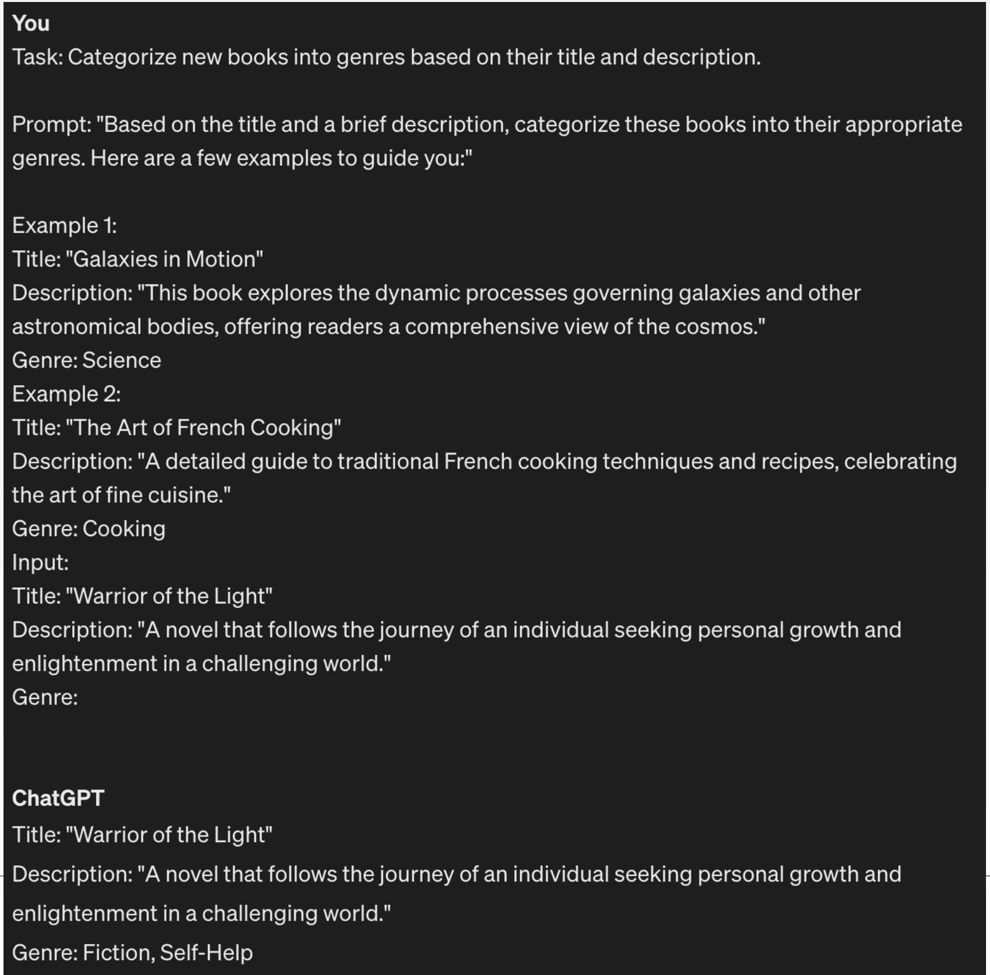
Library Branch: Central City Library

Wait for new text

Few-Shot Prompting

- **Few Example Guidance**: Uses a few carefully selected examples to guide the model's response.
- **Contextual Enhancement**: Leverages multiple examples to improve the model’s understanding and response accuracy.
- **Advantages**: Increases reliability and precision over one-shot and zero-shot prompting.
- **Limitations**: More preparation is required to create practical examples.
- **Use Cases**: Effective for tasks that have diverse outcomes or specific requirements that are difficult to describe.

Few-Shot Example



You

Task: Categorize new books into genres based on their title and description.

Prompt: "Based on the title and a brief description, categorize these books into their appropriate genres. Here are a few examples to guide you:"

Example 1:
Title: "Galaxies in Motion"
Description: "This book explores the dynamic processes governing galaxies and other astronomical bodies, offering readers a comprehensive view of the cosmos."
Genre: Science
Example 2:
Title: "The Art of French Cooking"
Description: "A detailed guide to traditional French cooking techniques and recipes, celebrating the art of fine cuisine."
Genre: Cooking
Input:
Title: "Warrior of the Light"
Description: "A novel that follows the journey of an individual seeking personal growth and enlightenment in a challenging world."
Genre:

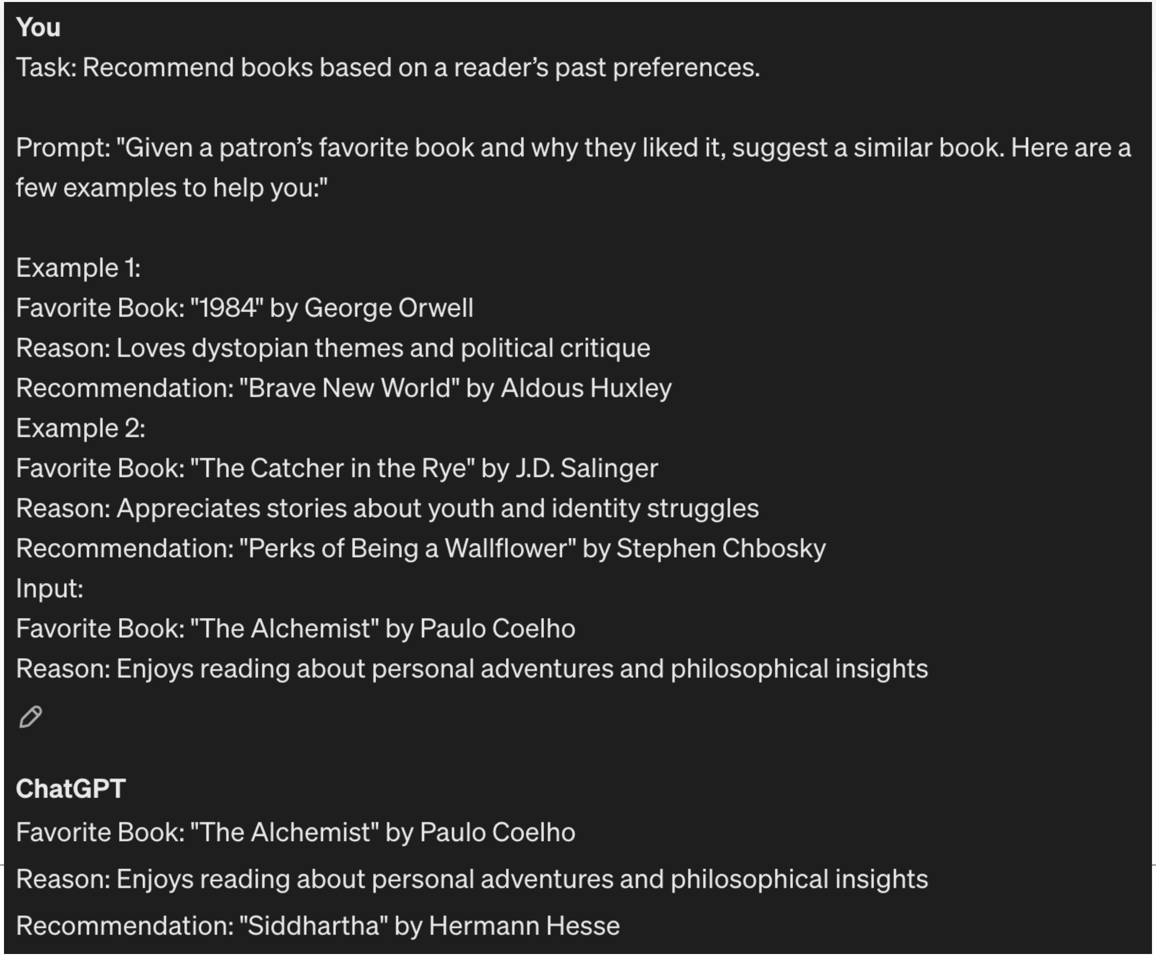
ChatGPT

Title: "Warrior of the Light"

Description: "A novel that follows the journey of an individual seeking personal growth and enlightenment in a challenging world."

Genre: Fiction, Self-Help

Few-Shot Example (cont'd)



You

Task: Recommend books based on a reader's past preferences.

Prompt: "Given a patron's favorite book and why they liked it, suggest a similar book. Here are a few examples to help you:"

Example 1:
Favorite Book: "1984" by George Orwell
Reason: Loves dystopian themes and political critique
Recommendation: "Brave New World" by Aldous Huxley
Example 2:
Favorite Book: "The Catcher in the Rye" by J.D. Salinger
Reason: Appreciates stories about youth and identity struggles
Recommendation: "Perks of Being a Wallflower" by Stephen Chbosky
Input:
Favorite Book: "The Alchemist" by Paulo Coelho
Reason: Enjoys reading about personal adventures and philosophical insights

ChatGPT

Favorite Book: "The Alchemist" by Paulo Coelho

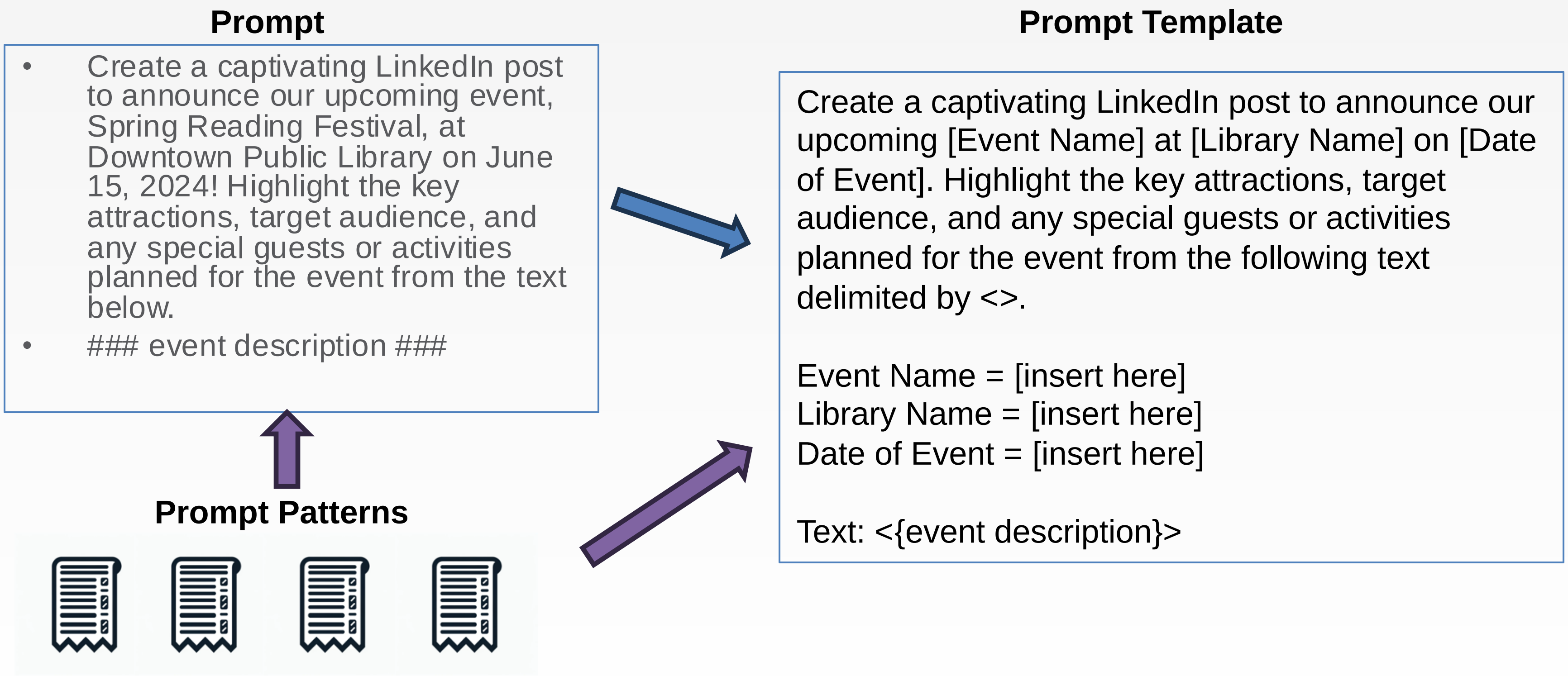
Reason: Enjoys reading about personal adventures and philosophical insights

Recommendation: "Siddhartha" by Hermann Hesse

Prompt Patterns

- A structured approach to guide user input, ensuring consistent, relevant, and comprehensive responses from AI models.
- They offer clear and uniform solutions to predefined common problems.
- They are abstracted from various prompts and reusable, aiming to save time when creating effective prompts.
- Adaptable and customizable for both domain-independent and domain-specific applications.
- Allows users to create prompt templates, providing a foundation for efficiently generating consistent and high-quality prompts.

Prompt Template



Prompt

- Create a captivating LinkedIn post to announce our upcoming event, Spring Reading Festival, at Downtown Public Library on June 15, 2024! Highlight the key attractions, target audience, and any special guests or activities planned for the event from the text below.
- ### event description ###

Prompt Patterns

Prompt Template

Create a captivating LinkedIn post to announce our upcoming [Event Name] at [Library Name] on [Date of Event]. Highlight the key attractions, target audience, and any special guests or activities planned for the event from the following text delimited by <>.

Event Name = [insert here]
Library Name = [insert here]
Date of Event = [insert here]

Text: <{event description}>

Prompt Template

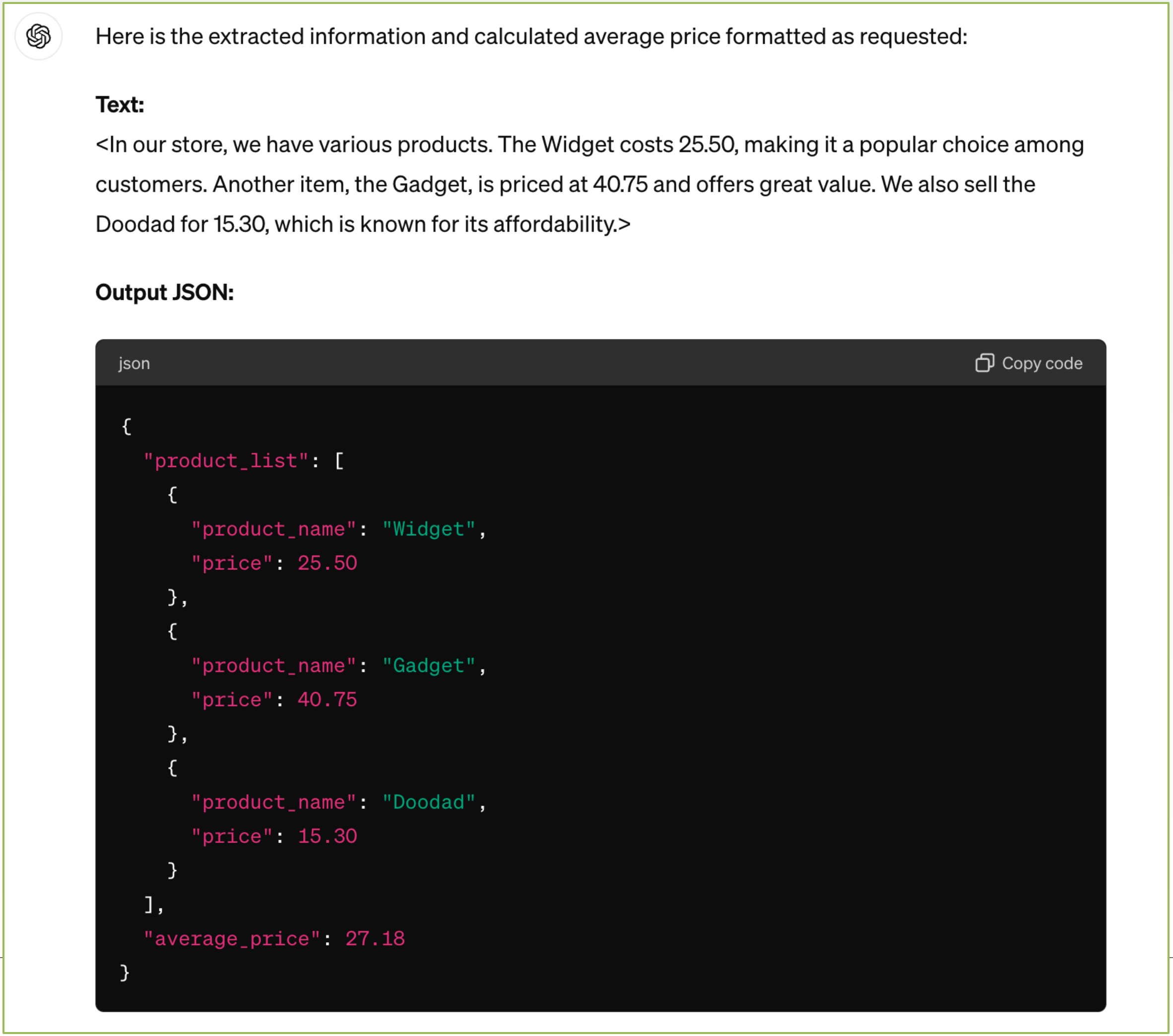
- Provides a foundation for creating multiple prompts without the need to craft each one individually, saving time and effort.
- **Consistency**: Ensures that prompts maintain a uniform structure and tone.
- **Clarity**: Offers direct guidelines on the necessary information to include for better communication with the AI model.
- **Reliability**: Reduces the chance of missing critical information and minimizes errors.
- **Efficiency**: Saves time and effort by using templates to create prompts quickly and effectively.
- **Adaptability**: Allows customization across various scenarios while maintaining a fundamental structure.
- **Scalability**: Enables effective management of increasing prompts through template reuse.

Template pattern

- **Definition:** Use a predefined structure or format to instruct an AI model to follow and respond with a consistent **output**.
- **Purpose:** Ensure responses are organized and adhere to a specific format for a particular intended purpose.
- **Application:** Ideal for tasks requiring structured responses, such as data interchange formats or any output needing a consistent format.
- **Guidelines:** Define the structure clearly, using headings and subheadings to organize content. Include placeholders for variable content. Make the template clear and intuitive for reuse.
- **Notes:** This pattern filters the AI model's responses, which may eliminate some useful information.

Template pattern Prompt Example

- Perform the following actions:
    1. Parse the following text delimited by <>.
    2. Extract the "product_name" and "price" fields from the input text.
    3. Output a new JSON array containing only these fields for each entry.
    4. Calculate the average price of the products and include it in the JSON output.
- Use the following format:
- Text: <text to parse>
- Output JSON: { "product_list": [{ "product_name": "", "price": }], "average_price": }



Here is the extracted information and calculated average price formatted as requested:

Text:

<In our store, we have various products. The Widget costs 25.50, making it a popular choice among customers. Another item, the Gadget, is priced at 40.75 and offers great value. We also sell the Doodad for 15.30, which is known for its affordability.>

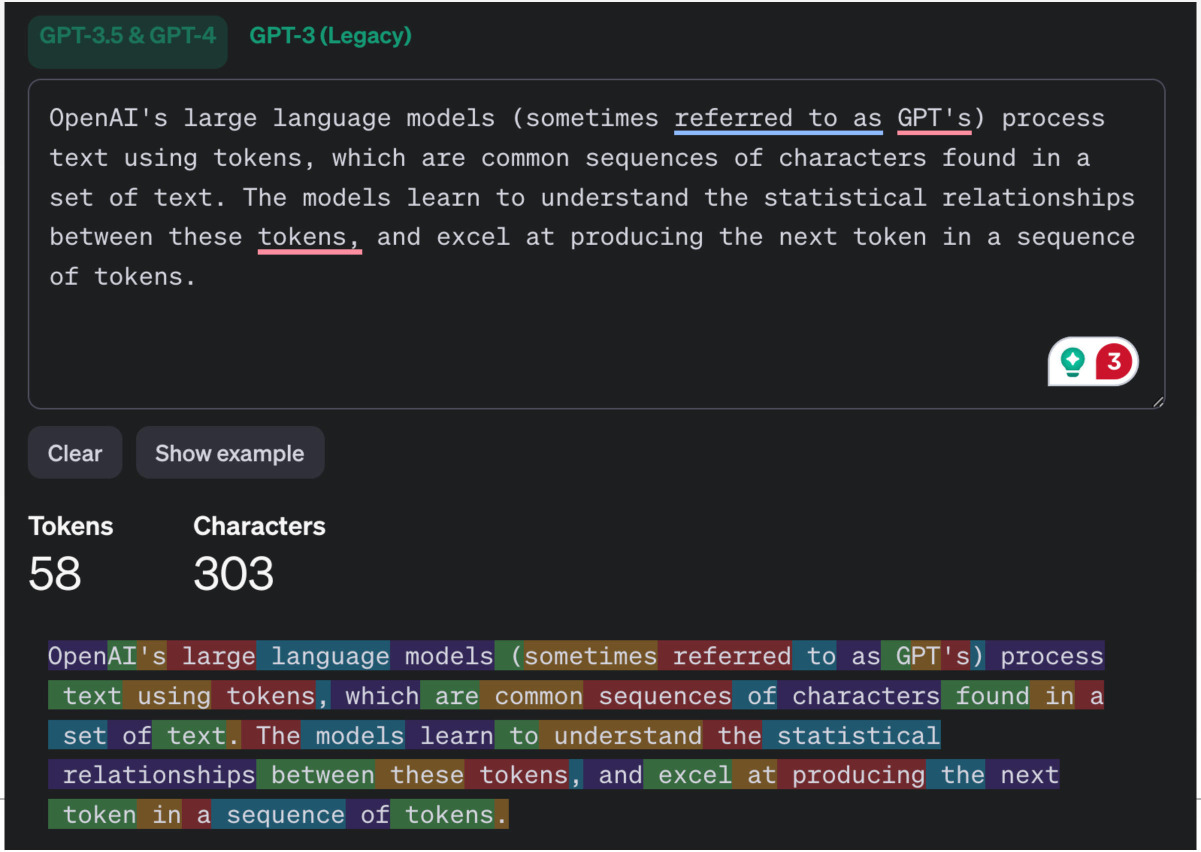
Output JSON:

```json
{
  "product_list": [
    {
      "product_name": "Widget",
      "price": 25.50
    },
    {
      "product_name": "Gadget",
      "price": 40.75
    },
    {
      "product_name": "Doodad",
      "price": 15.30
    }
  ],
  "average_price": 27.18
}
```

LLM has Context window

- Models have a maximum number of tokens they can process at once, called context window or token limit.
- The context window includes the input prompt and output response.
- Exceeding the token limit results in the loss of the earliest conversation details.
- Larger context windows improve the model's ability to comprehend lengthy inputs but are more computationally expensive.
- Prompt engineering is one strategy to include necessary information and optimize content to achieve the desired response without exceeding the token limit.

Tokenization



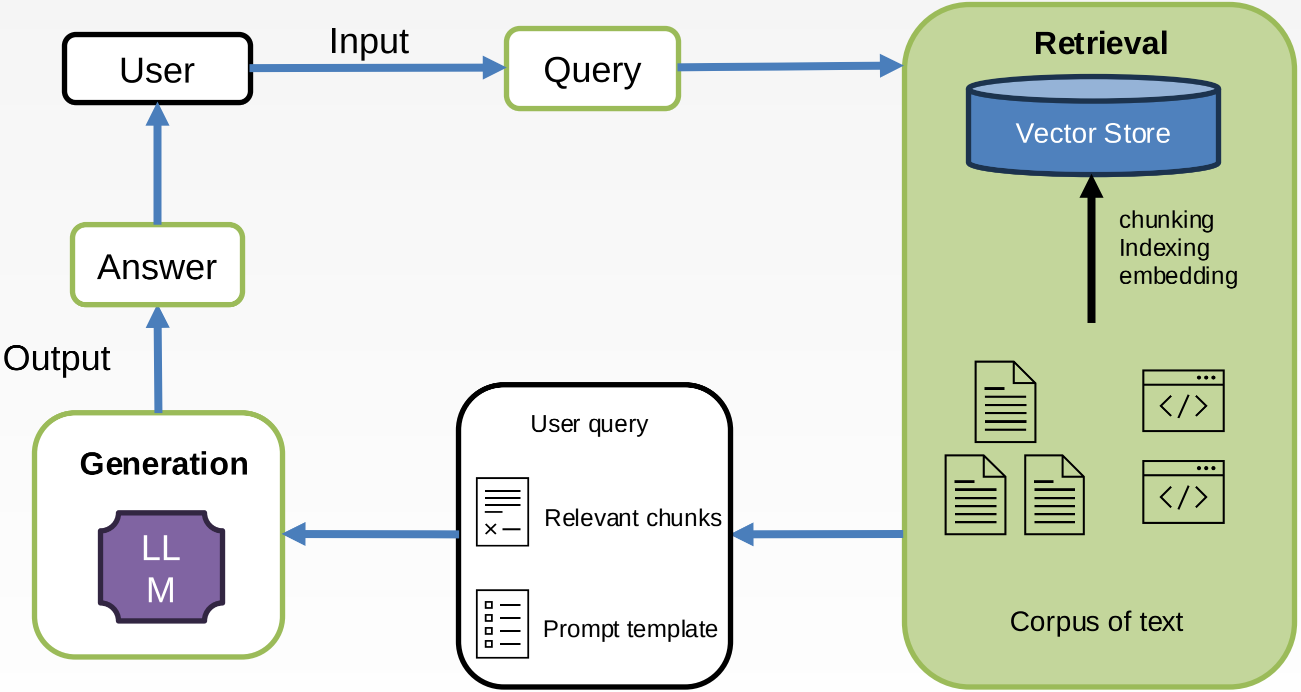
OpenAI's large language models (sometimes referred to as GPT's) process text using tokens, which are common sequences of characters found in a set of text. The models learn to understand the statistical relationships between these tokens, and excel at producing the next token in a sequence of tokens.

Tokens 58

Characters 303

OpenAI's large language models (sometimes referred to as GPT's) process text using tokens, which are common sequences of characters found in a set of text. The models learn to understand the statistical relationships between these tokens, and excel at producing the next token in a sequence of tokens.

Retrieval-Augmented Generation (RAG)



User

Input

Query

Retrieval

Vector Store

chunking
Indexing
embedding

Corpus of text

User query

Relevant chunks

Prompt template

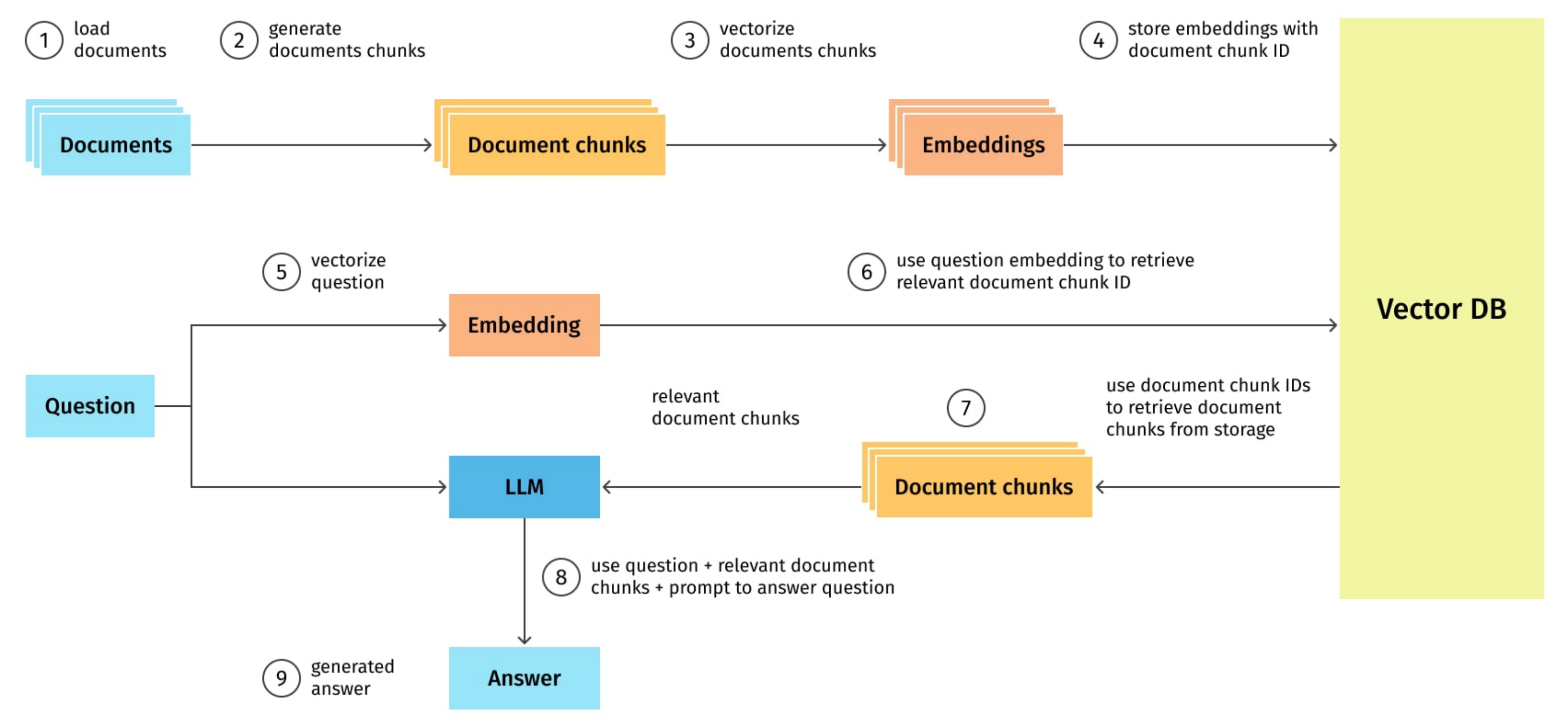
Generation

LL
M

Answer

Output

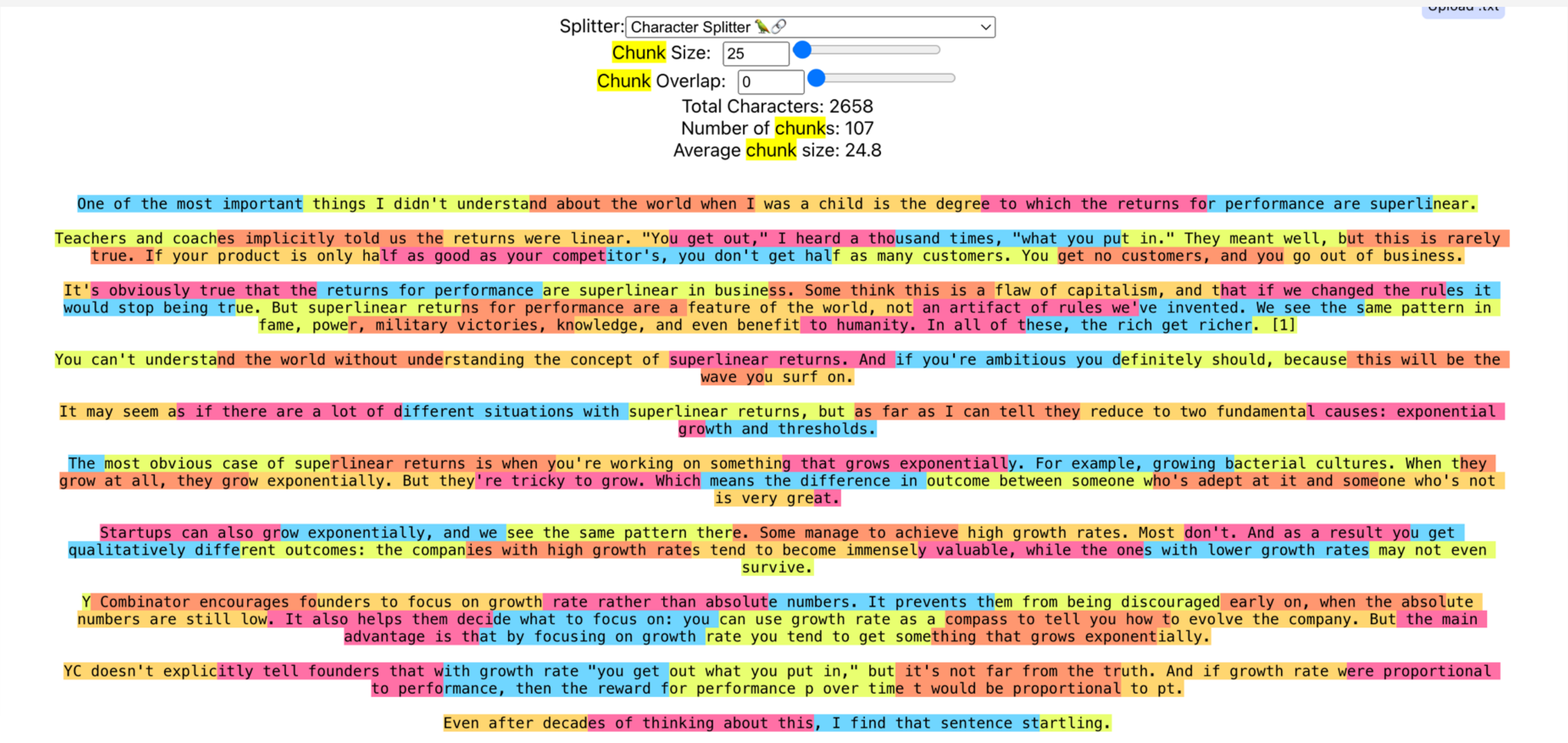
Schematic representation of the RAG process flow



① load documents
② generate documents chunks
③ vectorize documents chunks
④ store embeddings with document chunk ID
Documents
Document chunks
Embeddings
⑤ vectorize question
⑥ use question embedding to retrieve relevant document chunk ID
Embedding
Vector DB
Question
relevant document chunks
⑦
use document chunk IDs to retrieve document chunks from storage
LLM
Document chunks
⑧ use question + relevant document chunks + prompt to answer question
⑨ generated answer
Answer

Image source : https://www.griddynamics.com/blog/retrieval-augmented-generation-llm

Chunking



Splitter: Character Splitter

Chunk Size: 25

Chunk Overlap: 0

Total Characters: 2658
Number of chunks: 107
Average chunk size: 24.8

One of the most important things I didn't understand about the world when I was a child is the degree to which the returns for performance are superlinear.

Teachers and coaches implicitly told us the returns were linear. "You get out," I heard a thousand times, "what you put in." They meant well, but this is rarely true. If your product is only half as good as your competitor's, you don't get half as many customers. You get no customers, and you go out of business.

It's obviously true that the returns for performance are superlinear in business. Some think this is a flaw of capitalism, and that if we changed the rules it would stop being true. But superlinear returns for performance are a feature of the world, not an artifact of rules we've invented. We see the same pattern in fame, power, military victories, knowledge, and even benefit to humanity. In all of these, the rich get richer. [1]

You can't understand the world without understanding the concept of superlinear returns. And if you're ambitious you definitely should, because this will be the wave you surf on.

It may seem as if there are a lot of different situations with superlinear returns, but as far as I can tell they reduce to two fundamental causes: exponential growth and thresholds.

The most obvious case of superlinear returns is when you're working on something that grows exponentially. For example, growing bacterial cultures. When they grow at all, they grow exponentially. But they're tricky to grow. Which means the difference in outcome between someone who's adept at it and someone who's not is very great.

Startups can also grow exponentially, and we see the same pattern there. Some manage to achieve high growth rates. Most don't. And as a result you get qualitatively different outcomes: the companies with high growth rates tend to become immensely valuable, while the ones with lower growth rates may not even survive.

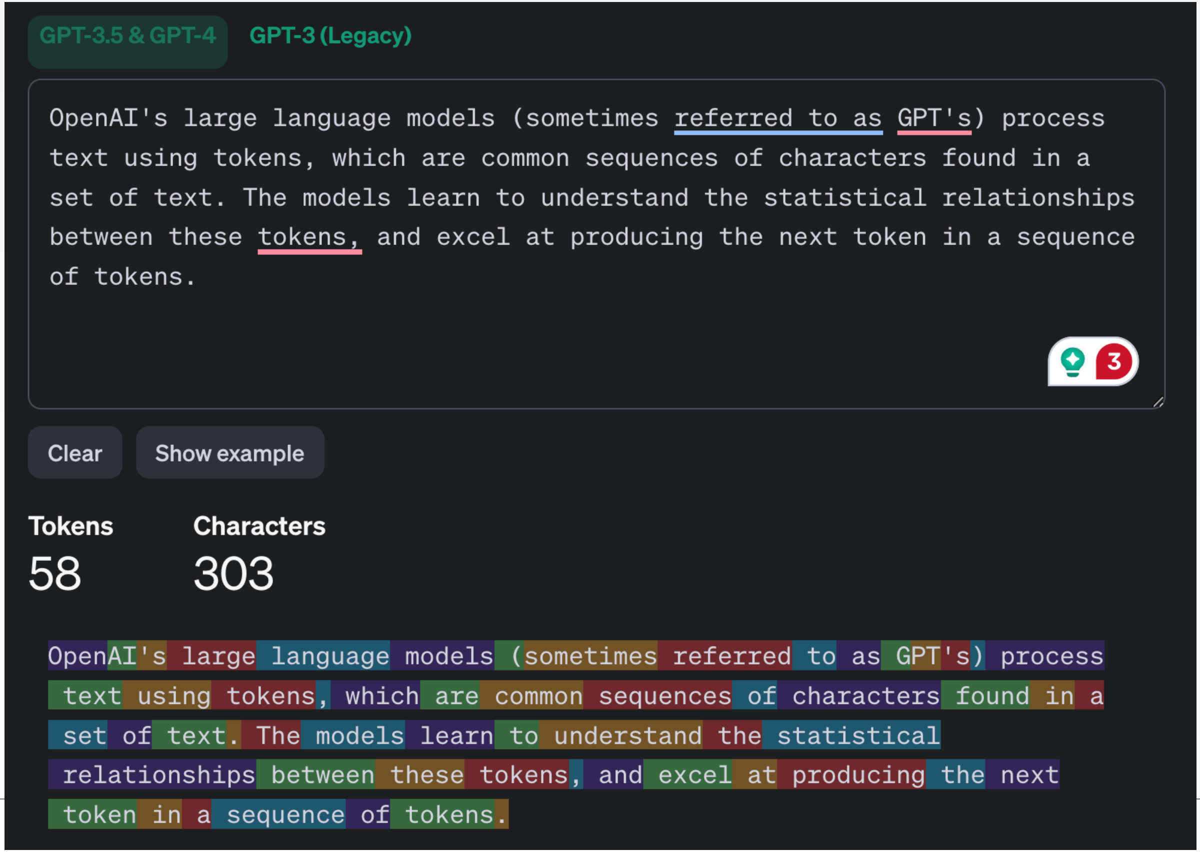
Y Combinator encourages founders to focus on growth rate rather than absolute numbers. It prevents them from being discouraged early on, when the absolute numbers are still low. It also helps them decide what to focus on: you can use growth rate as a compass to tell you how to evolve the company. But the main advantage is that by focusing on growth rate you tend to get something that grows exponentially.

YC doesn't explicitly tell founders that with growth rate "you get out what you put in," but it's not far from the truth. And if growth rate were proportional to performance, then the reward for performance p over time t would be proportional to pt.

Even after decades of thinking about this, I find that sentence startling.

Image source: https://chunkviz.up.railway.app/

Tokenization



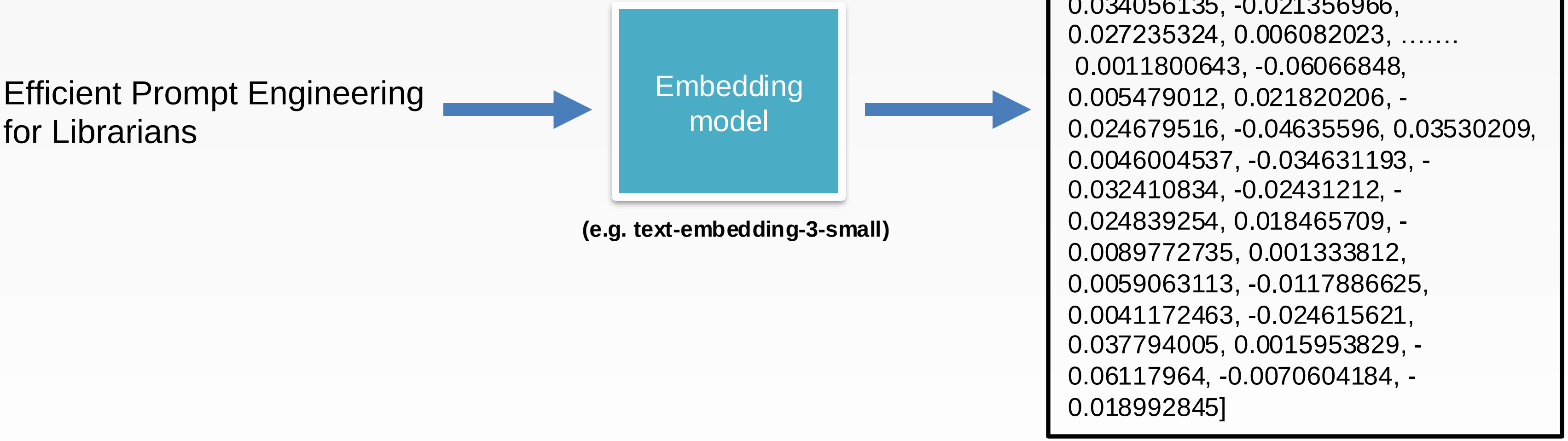
OpenAI's large language models (sometimes referred to as GPT's) process text using tokens, which are common sequences of characters found in a set of text. The models learn to understand the statistical relationships between these tokens, and excel at producing the next token in a sequence of tokens.

Tokens
58

Characters
303

OpenAI's large language models (sometimes referred to as GPT's) process text using tokens, which are common sequences of characters found in a set of text. The models learn to understand the statistical relationships between these tokens, and excel at producing the next token in a sequence of tokens.

Embedding



Efficient Prompt Engineering for Librarians

Embedding model

(e.g. text-embedding-3-small)

```
[-0.021229176, 0.0014725844, -
0.0215806, -0.002835349, 0.009168959,
0.026947796, -0.019695692,
0.040956814, -0.037570372,
0.0042530233, 0.044151574, -
0.045333635, -0.00263368, -
0.026867926, 0.03808153, 0.008078747,
0.030861376, 0.0009085097, 0.0338325,
-0.0048760017, 0.017187806,
0.0027954145, 0.007835147,
0.034056135, -0.021356966,
0.027235324, 0.006082023, …….
 0.0011800643, -0.06066848,
0.005479012, 0.021820206, -
0.024679516, -0.04635596, 0.03530209,
0.0046004537, -0.034631193, -
0.032410834, -0.02431212, -
0.024839254, 0.018465709, -
0.0089772735, 0.001333812,
0.0059063113, -0.0117886625,
0.0041172463, -0.024615621,
0.037794005, 0.0015953829, -
0.06117964, -0.0070604184, -
0.018992845]
```

A 1536-dimensional vector

Indexing

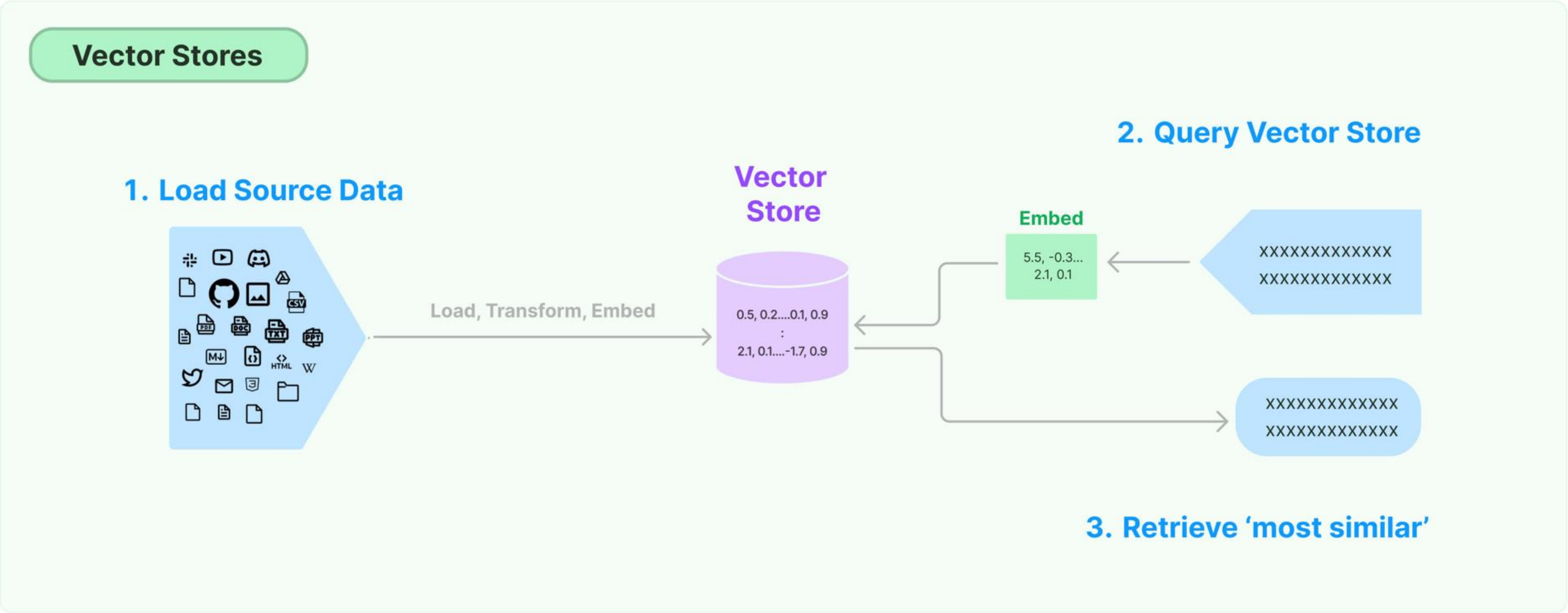
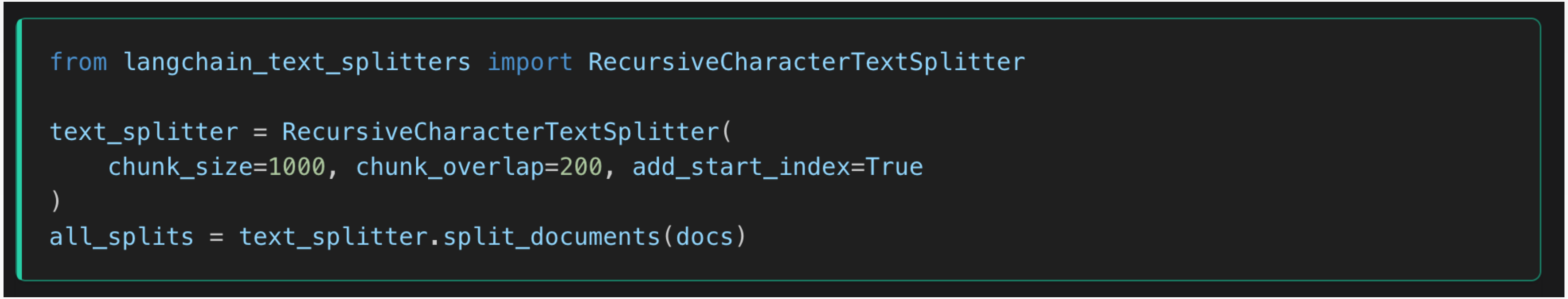
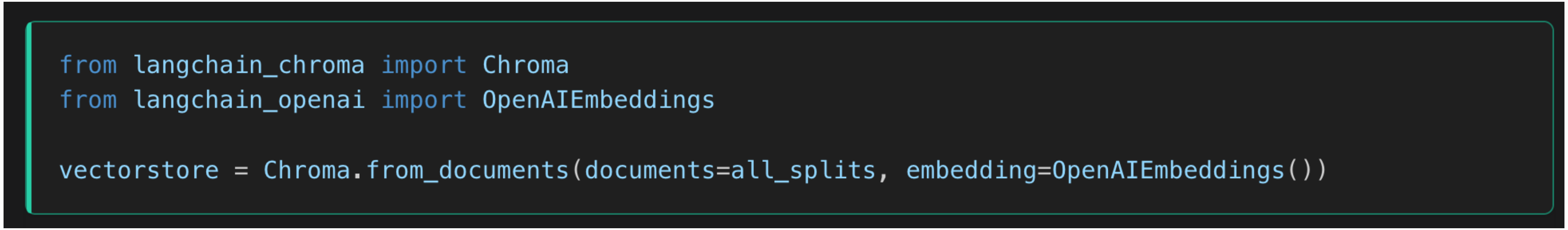


Image source: https://python.langchain.com/v0.1/docs/modules/data_connection/vectorstores/

RAG Code Example



```
from langchain_text_splitters import RecursiveCharacterTextSplitter

text_splitter = RecursiveCharacterTextSplitter(
    chunk_size=1000, chunk_overlap=200, add_start_index=True
)
all_splits = text_splitter.split_documents(docs)
```

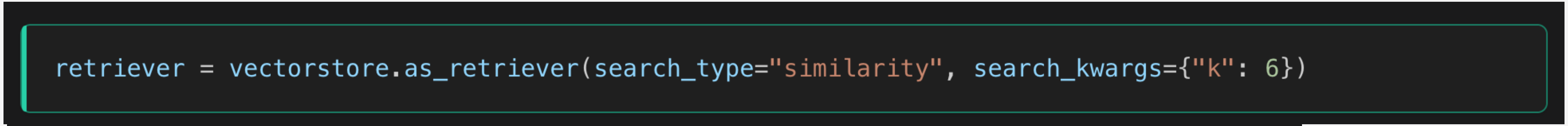


```
from langchain_chroma import Chroma
from langchain_openai import OpenAIEmbeddings

vectorstore = Chroma.from_documents(documents=all_splits, embedding=OpenAIEmbeddings())
```

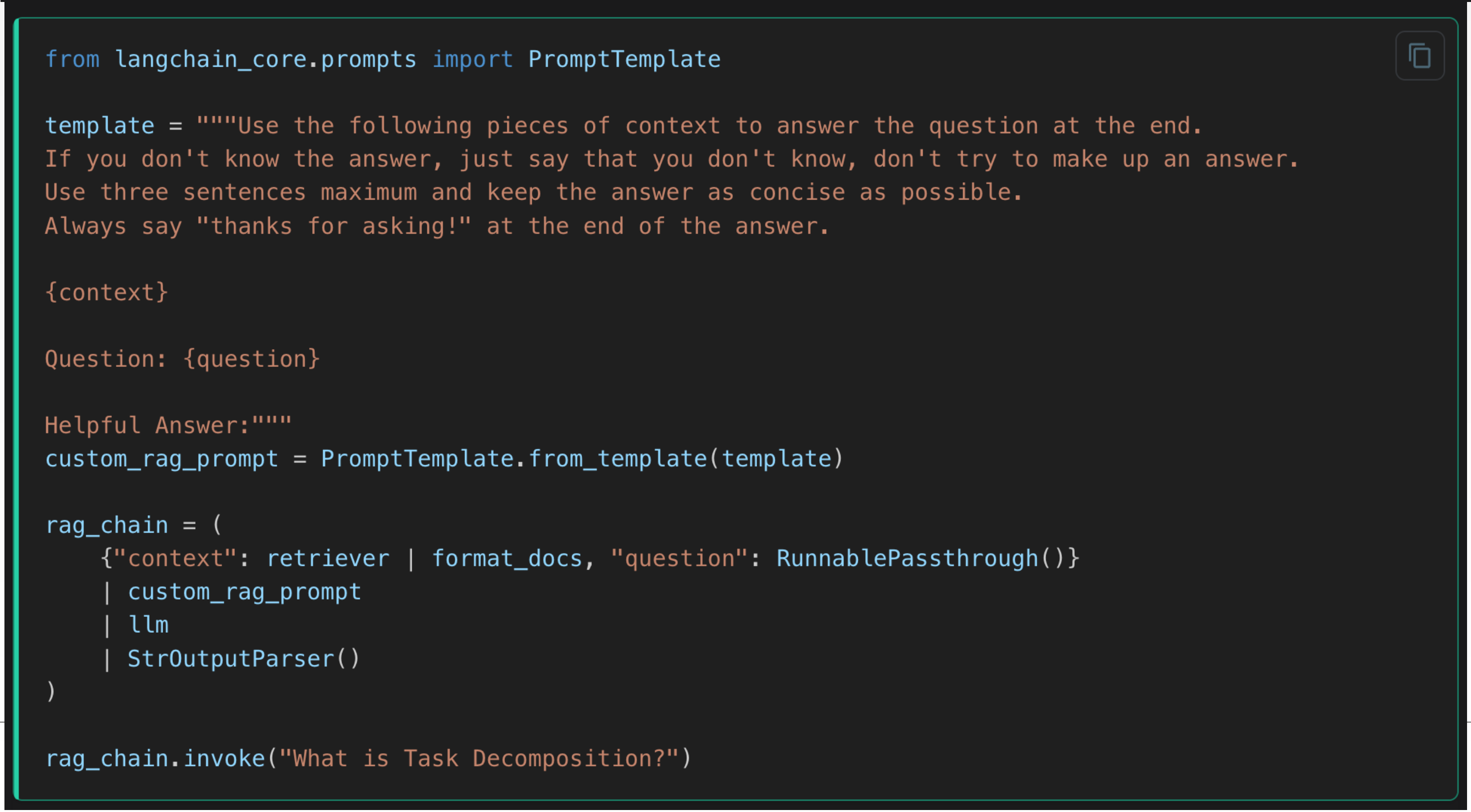
Source : https://python.langchain.com/v0.1/docs/use_cases/question_answering/quickstart/

RAG Code Example

Source : https://python.langchain.com/v0.1/docs/use_cases/question_answering/quickstart/



```python
retriever = vectorstore.as_retriever(search_type="similarity", search_kwargs={"k": 6})
```



```python
from langchain_core.prompts import PromptTemplate

template = """Use the following pieces of context to answer the question at the end.
If you don't know the answer, just say that you don't know, don't try to make up an answer.
Use three sentences maximum and keep the answer as concise as possible.
Always say "thanks for asking!" at the end of the answer.

{context}

Question: {question}

Helpful Answer:"""
custom_rag_prompt = PromptTemplate.from_template(template)

rag_chain = (
    {"context": retriever | format_docs, "question": RunnablePassthrough()}
    | custom_rag_prompt
    | llm
    | StrOutputParser()
)

rag_chain.invoke("What is Task Decomposition?")
```

RAG Evaluation

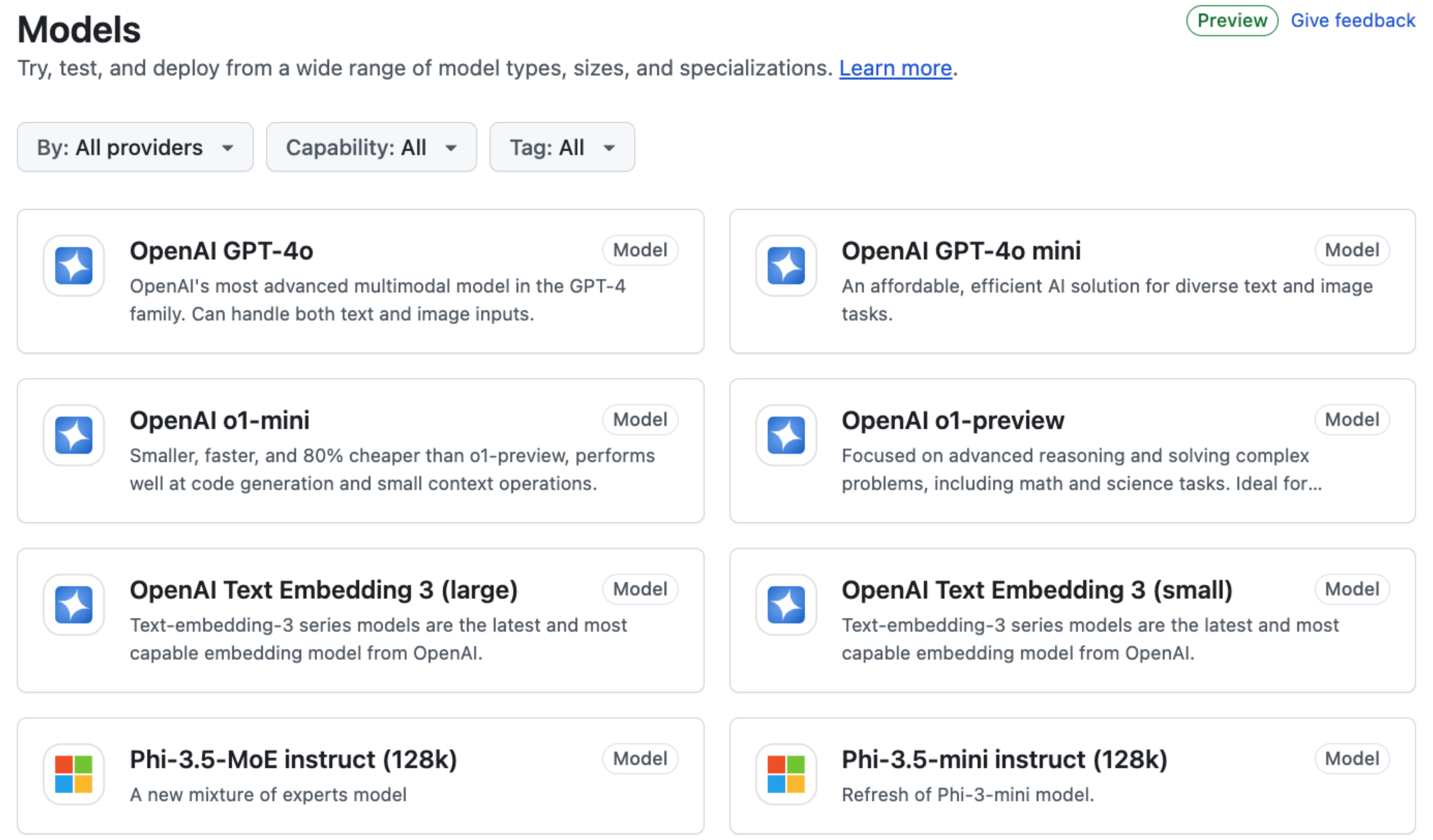
- ROUGE (Recall-Oriented Understudy for Gisting Evaluation)
- Measures the overlap between generated and reference responses
- Evaluates using n-grams, word sequences, and word pairs
- Considers both precision and recall aspects
- Evaluating the generation component of RAG systems by comparing the output text against reference answers.

Demonstration and Hands-On Exercise

Ways to Interact with LLM/LMM models

- Web/Application UI
- OpenAI / Google Gemini / Anthropic APIs
- Open-source Models
- GitHub Models
- Hugging Face Models
- Anaconda AI Navigator
- Etc.

Web/Application UI



Models

Try, test, and deploy from a wide range of model types, sizes, and specializations. Learn more.

OpenAI GPT-4o Model
OpenAI's most advanced multimodal model in the GPT-4 family. Can handle both text and image inputs.

OpenAI GPT-4o mini Model
An affordable, efficient AI solution for diverse text and image tasks.

OpenAI o1-mini Model
Smaller, faster, and 80% cheaper than o1-preview, performs well at code generation and small context operations.

OpenAI o1-preview Model
Focused on advanced reasoning and solving complex problems, including math and science tasks. Ideal for...

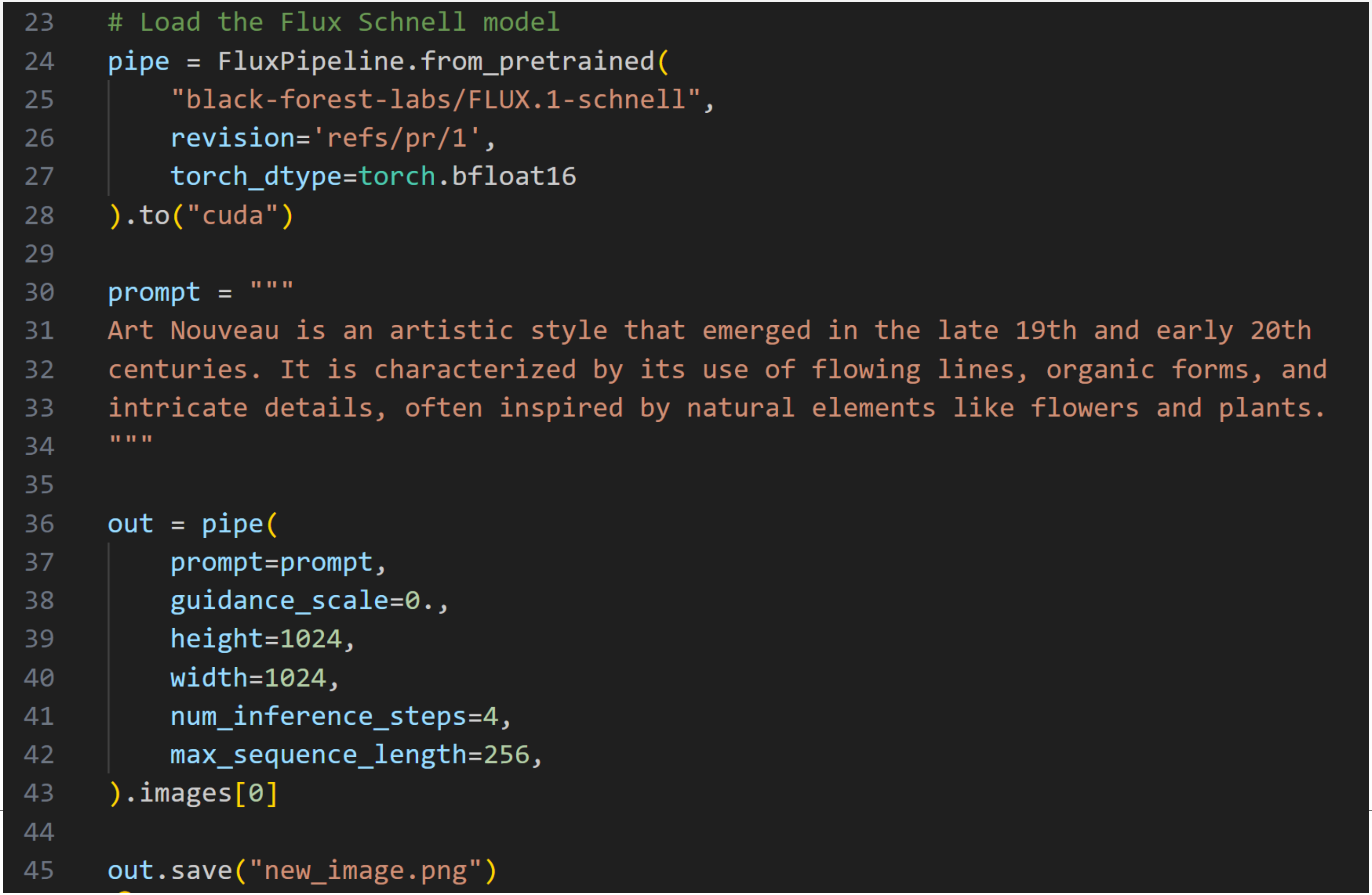
OpenAI Text Embedding 3 (large) Model
Text-embedding-3 series models are the latest and most capable embedding model from OpenAI.

OpenAI Text Embedding 3 (small) Model
Text-embedding-3 series models are the latest and most capable embedding model from OpenAI.

Phi-3.5-MoE instruct (128k) Model
A new mixture of experts model

Phi-3.5-mini instruct (128k) Model
Refresh of Phi-3-mini model.

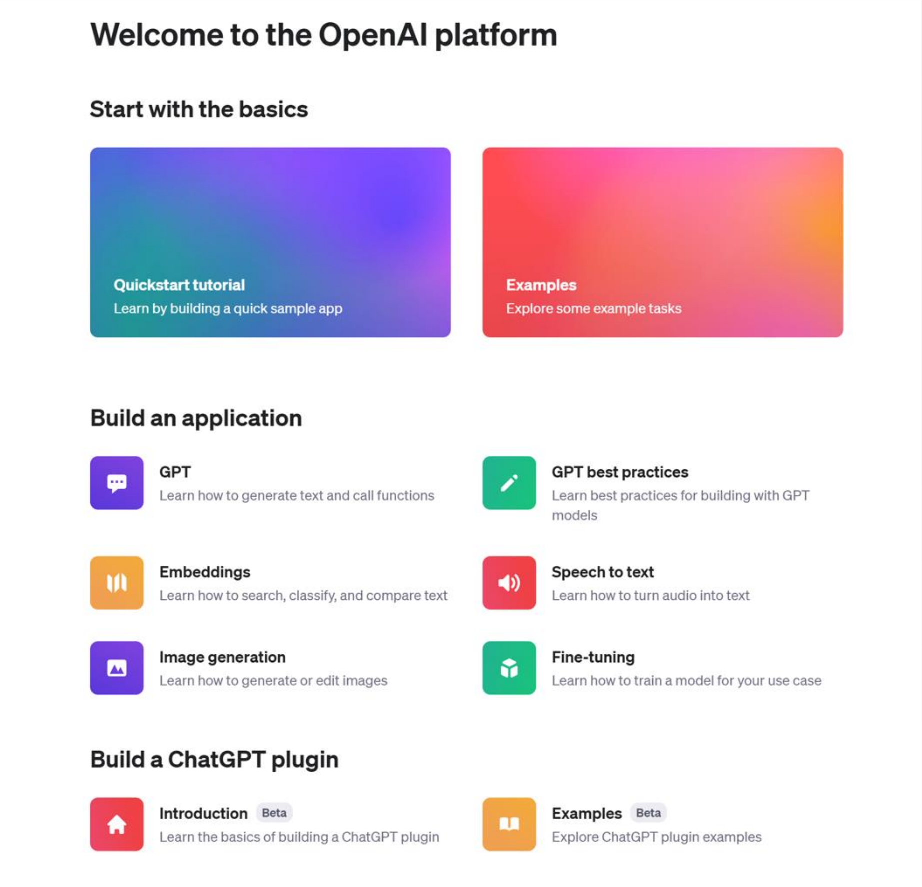
Interact with Model via Program



```
# Load the Flux Schnell model
pipe = FluxPipeline.from_pretrained(
    "black-forest-labs/FLUX.1-schnell",
    revision='refs/pr/1',
    torch_dtype=torch.bfloat16
).to("cuda")

prompt = """
Art Nouveau is an artistic style that emerged in the late 19th and early 20th
centuries. It is characterized by its use of flowing lines, organic forms, and
intricate details, often inspired by natural elements like flowers and plants.
"""


out = pipe(
    prompt=prompt,
    guidance_scale=0.,
    height=1024,
    width=1024,
    num_inference_steps=4,
    max_sequence_length=256,
).images[0]

out.save("new_image.png")
```

OpenAI API

- A cloud-based service for accessing OpenAI's GPT (Generative Pre-trained Transformer) models
  - GPT-4 and GPT-3.5: Models can understand as well as generate natural language or code
  - DALL·E: A model that can generate and edit images given a natural language prompt
  - Whisper: A model that can convert audio into text
  - Embeddings: A set of models that can convert text into a numerical form
  - Moderation: A fine-tuned model that can detect whether text may be sensitive or unsafe



Welcome to the OpenAI platform

Start with the basics

Quickstart tutorial
Learn by building a quick sample app

Examples
Explore some example tasks

Build an application

GPT
Learn how to generate text and call functions

GPT best practices
Learn best practices for building with GPT models

Embeddings
Learn how to search, classify, and compare text

Speech to text
Learn how to turn audio into text

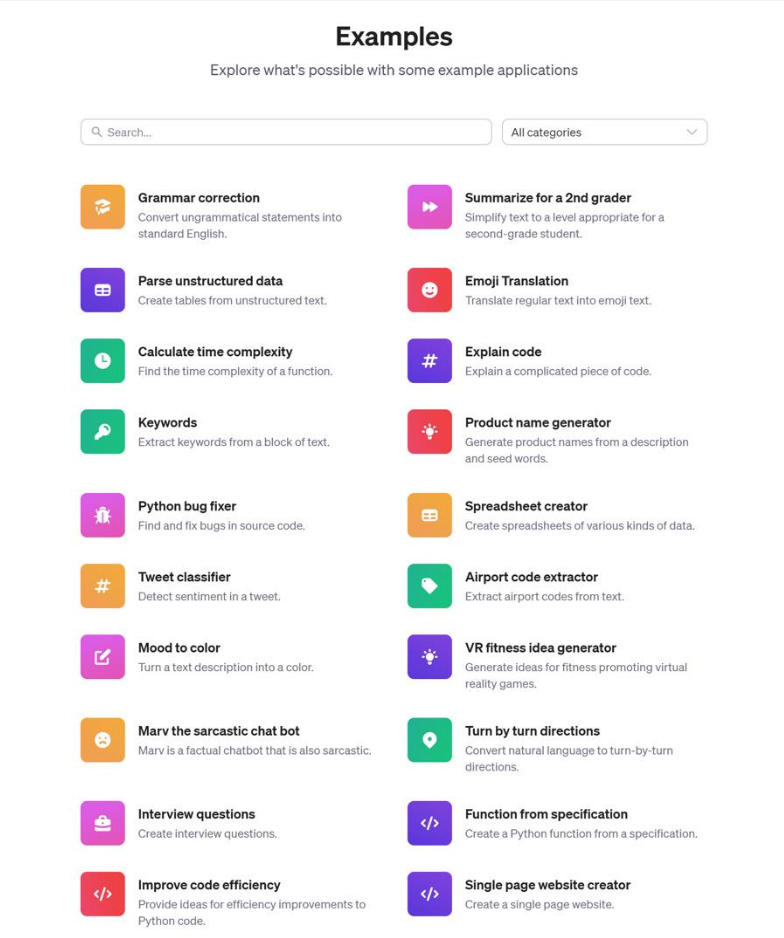
Image generation
Learn how to generate or edit images

Fine-tuning
Learn how to train a model for your use case

Build a ChatGPT plugin

Introduction Beta
Learn the basics of building a ChatGPT plugin

Examples Beta
Explore ChatGPT plugin examples



Examples

Explore what's possible with some example applications

Grammar correction
Convert ungrammatical statements into standard English.

Summarize for a 2nd grader
Simplify text to a level appropriate for a second-grade student.

Parse unstructured data
Create tables from unstructured text.

Emoji Translation
Translate regular text into emoji text.

Calculate time complexity
Find the time complexity of a function.

Explain code
Explain a complicated piece of code.

Keywords
Extract keywords from a block of text.

Product name generator
Generate product names from a description and seed words.

Python bug fixer
Find and fix bugs in source code.

Spreadsheet creator
Create spreadsheets of various kinds of data.

Tweet classifier
Detect sentiment in a tweet.

Airport code extractor
Extract airport codes from text.

Mood to color
Turn a text description into a color.

VR fitness idea generator
Generate ideas for fitness promoting virtual reality games.

Marv the sarcastic chat bot
Marv is a factual chatbot that is also sarcastic.

Turn by turn directions
Convert natural language to turn-by-turn directions.

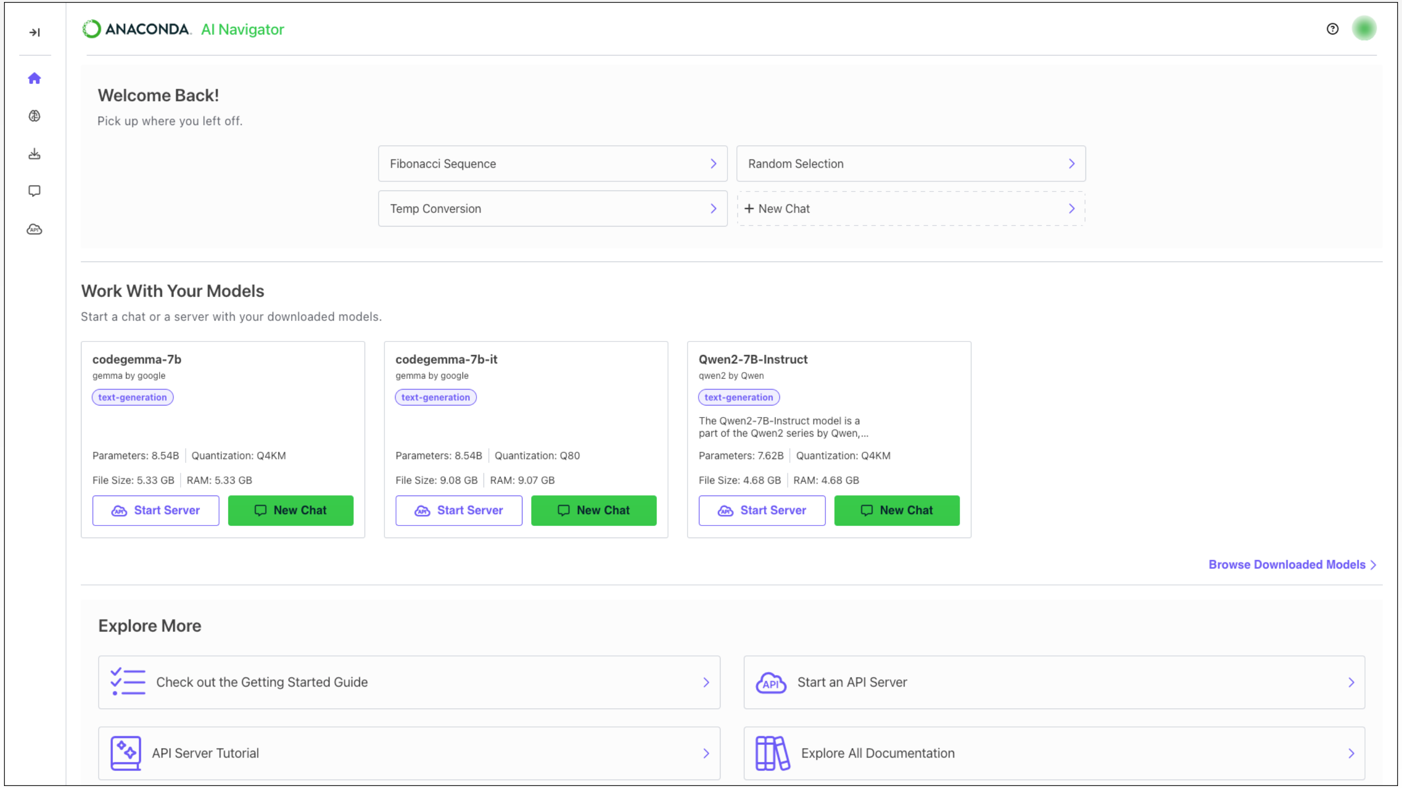
Interview questions
Create interview questions.

Function from specification
Create a Python function from a specification.

Improve code efficiency
Provide ideas for efficiency improvements to Python code.

Single page website creator
Create a single page website.

Development Environment



Welcome Back!

Pick up where you left off.

Work With Your Models

Start a chat or a server with your downloaded models.

codegemma-7b
gemma by google

Parameters: 8.54B | Quantization: Q4KM
File Size: 5.33 GB | RAM: 5.33 GB

codegemma-7b-it
gemma by google

Parameters: 8.54B | Quantization: Q80
File Size: 9.08 GB | RAM: 9.07 GB

Qwen2-7B-Instruct
qwen2 by Qwen

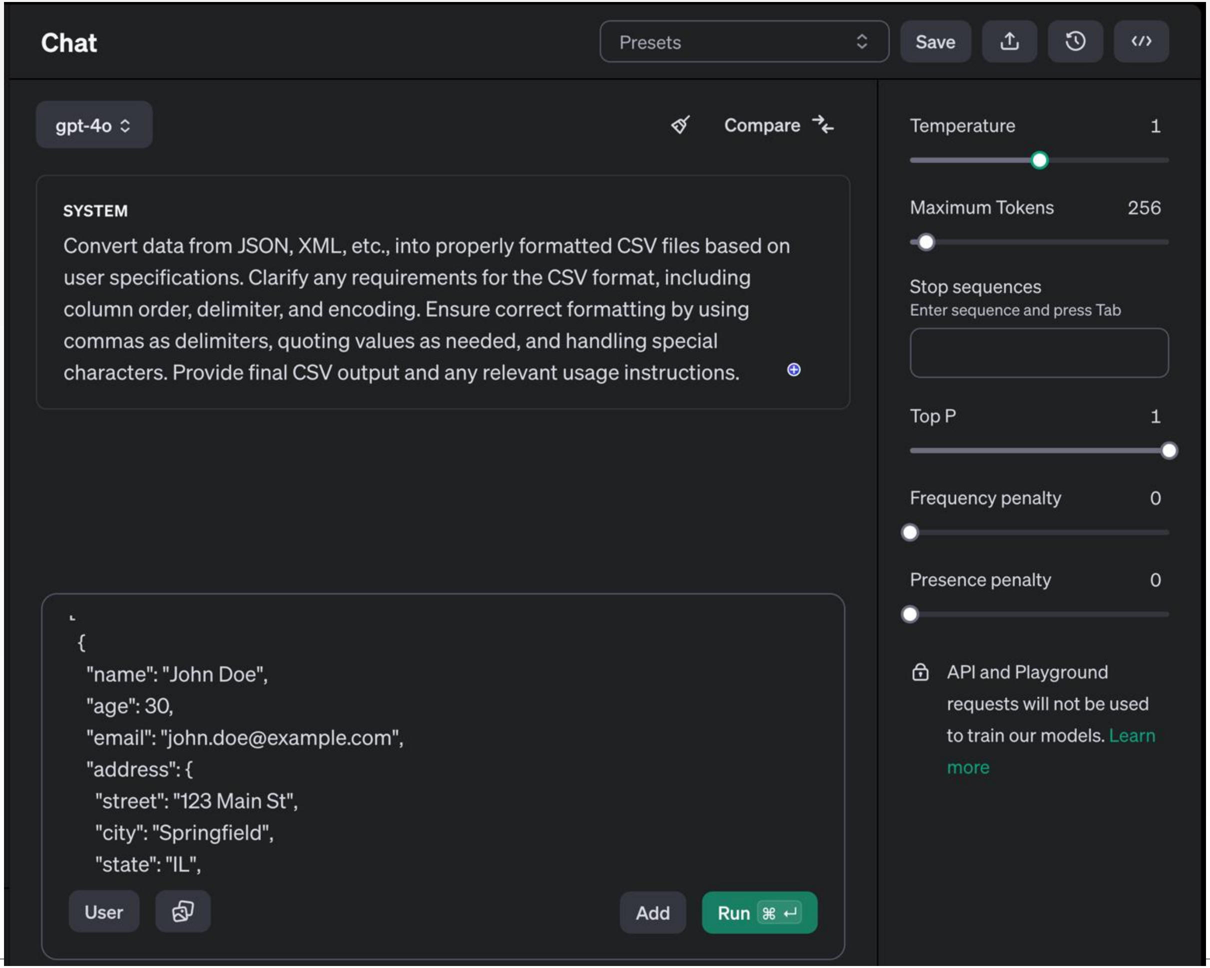
The Qwen2-7B-Instruct model is a part of the Qwen2 series by Qwen,...
Parameters: 7.62B | Quantization: Q4KM
File Size: 4.68 GB | RAM: 4.68 GB

Explore More

Check out the Getting Started Guide
Start an API Server
API Server Tutorial
Explore All Documentation

Image source: https://www.anaconda.com/products/ai-navigator

Use the Model API



gpt-4o

SYSTEM

Convert data from JSON, XML, etc., into properly formatted CSV files based on user specifications. Clarify any requirements for the CSV format, including column order, delimiter, and encoding. Ensure correct formatting by using commas as delimiters, quoting values as needed, and handling special characters. Provide final CSV output and any relevant usage instructions.

```
`
 {
  "name": "John Doe",
  "age": 30,
  "email": "john.doe@example.com",
  "address": {
   "street": "123 Main St",
   "city": "Springfield",
   "state": "IL",
```

Temperature 1

Maximum Tokens 256

Stop sequences
Enter sequence and press Tab

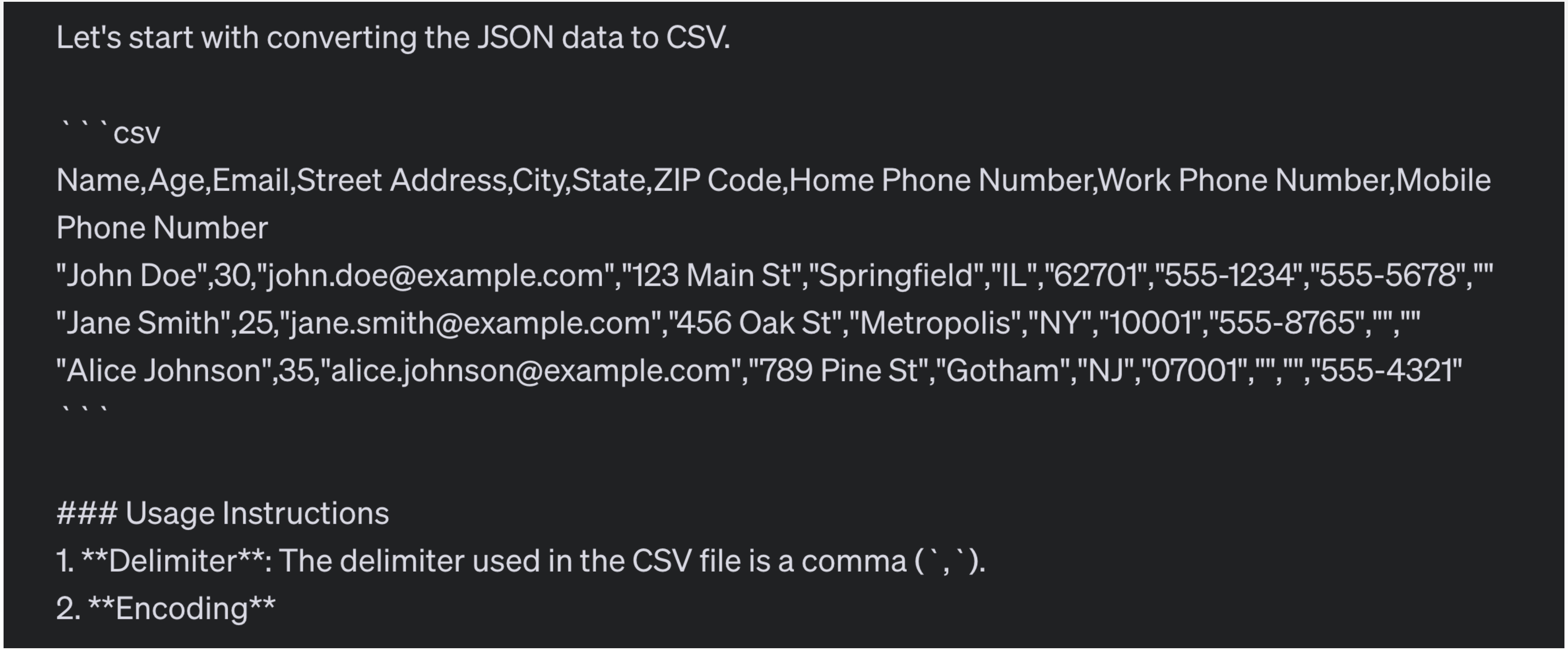
Top P 1

Frequency penalty 0

Presence penalty 0

API and Playground requests will not be used to train our models. Learn more

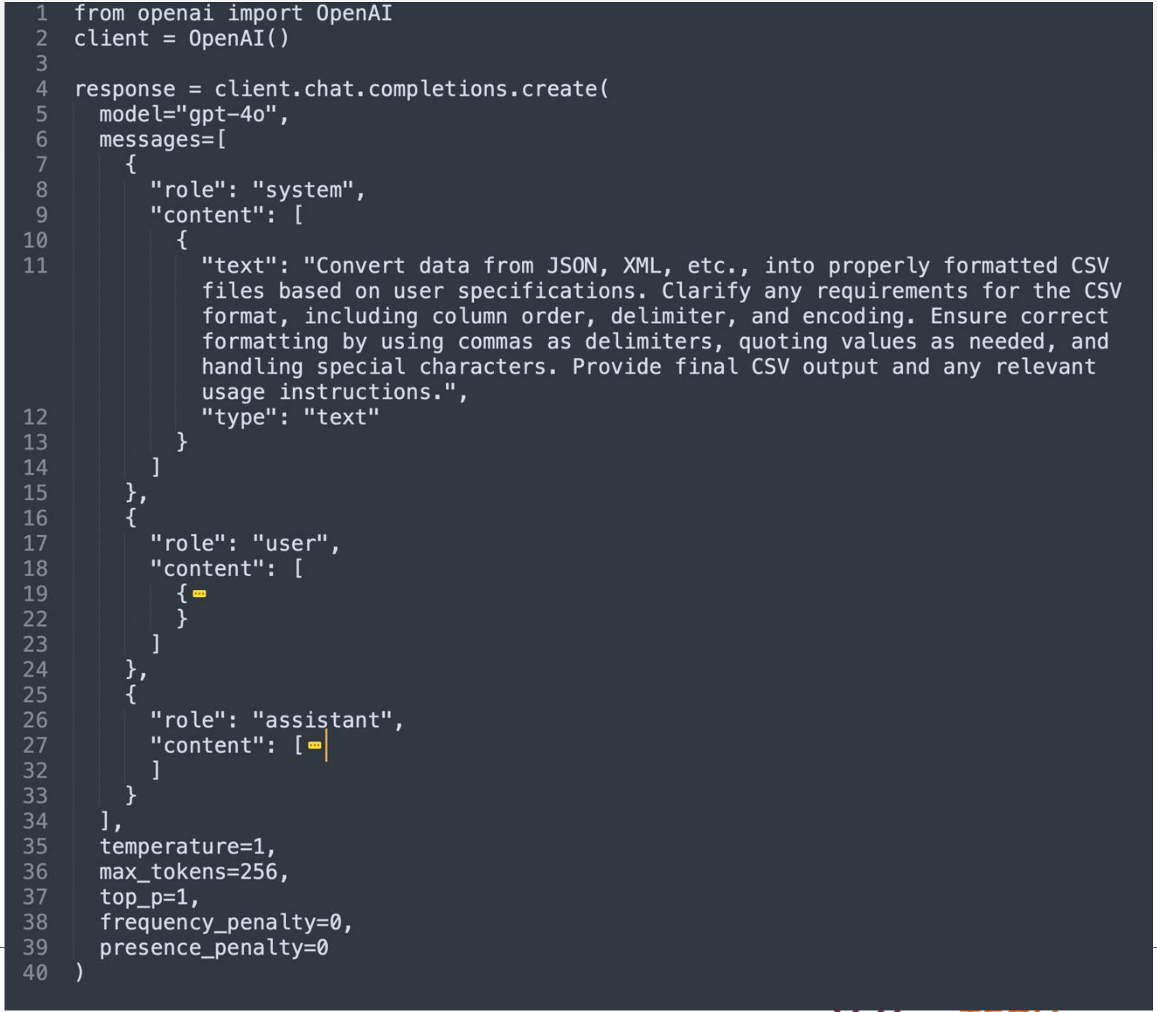
Use the Model API (Output)



~~~
Let's start with converting the JSON data to CSV.

```csv
Name,Age,Email,Street Address,City,State,ZIP Code,Home Phone Number,Work Phone Number,Mobile Phone Number
"John Doe",30,"john.doe@example.com","123 Main St","Springfield","IL","62701","555-1234","555-5678",""
"Jane Smith",25,"jane.smith@example.com","456 Oak St","Metropolis","NY","10001","555-8765","",""
"Alice Johnson",35,"alice.johnson@example.com","789 Pine St","Gotham","NJ","07001","","","555-4321"
```


### Usage Instructions
1. **Delimiter**: The delimiter used in the CSV file is a comma (`,`).
2. **Encoding**
~~~

Use the Model API

- System: Instruction prompt to the system or prompt template
- User: User query or user input
- Assistant: Assistant response



```
from openai import OpenAI
client = OpenAI()

response = client.chat.completions.create(
  model="gpt-4o",
  messages=[
    {
      "role": "system",
      "content": [
        {
          "text": "Convert data from JSON, XML, etc., into properly formatted CSV files based on user specifications. Clarify any requirements for the CSV format, including column order, delimiter, and encoding. Ensure correct formatting by using commas as delimiters, quoting values as needed, and handling special characters. Provide final CSV output and any relevant usage instructions.",
          "type": "text"
        }
      ]
    },
    {
      "role": "user",
      "content": [
        {…
        }
      ]
    },
    {
      "role": "assistant",
      "content": […
      ]
    }
  ],
  temperature=1,
  max_tokens=256,
  top_p=1,
  frequency_penalty=0,
  presence_penalty=0
)
```

🦜🔗 LangChain

- Framework for developing language model-powered applications
- Enables context-aware and reasoning applications.
- Modular components and off-the-shelf chains for ease of use and customization.
- Standard, extendable modules: Model I/O, Retrieval, Chains, Agents, Memory, Callbacks.
- Building applications with LLMs through composability.

https://www.langchain.com/

Implementation Consideration on ETDs

- Document Processing Considerations
  - Handle multi-level document structure (chapters, sections, appendices, references)
  - Process mixed content types (text, equations, figures, tables, citations)
  - Manage large PDF files (100+ pages) efficiently during ingestion
  - Extract and maintain hierarchical relationships between sections
  - Preserve academic citation networks and reference mappings

Implementation Consideration on ETDs

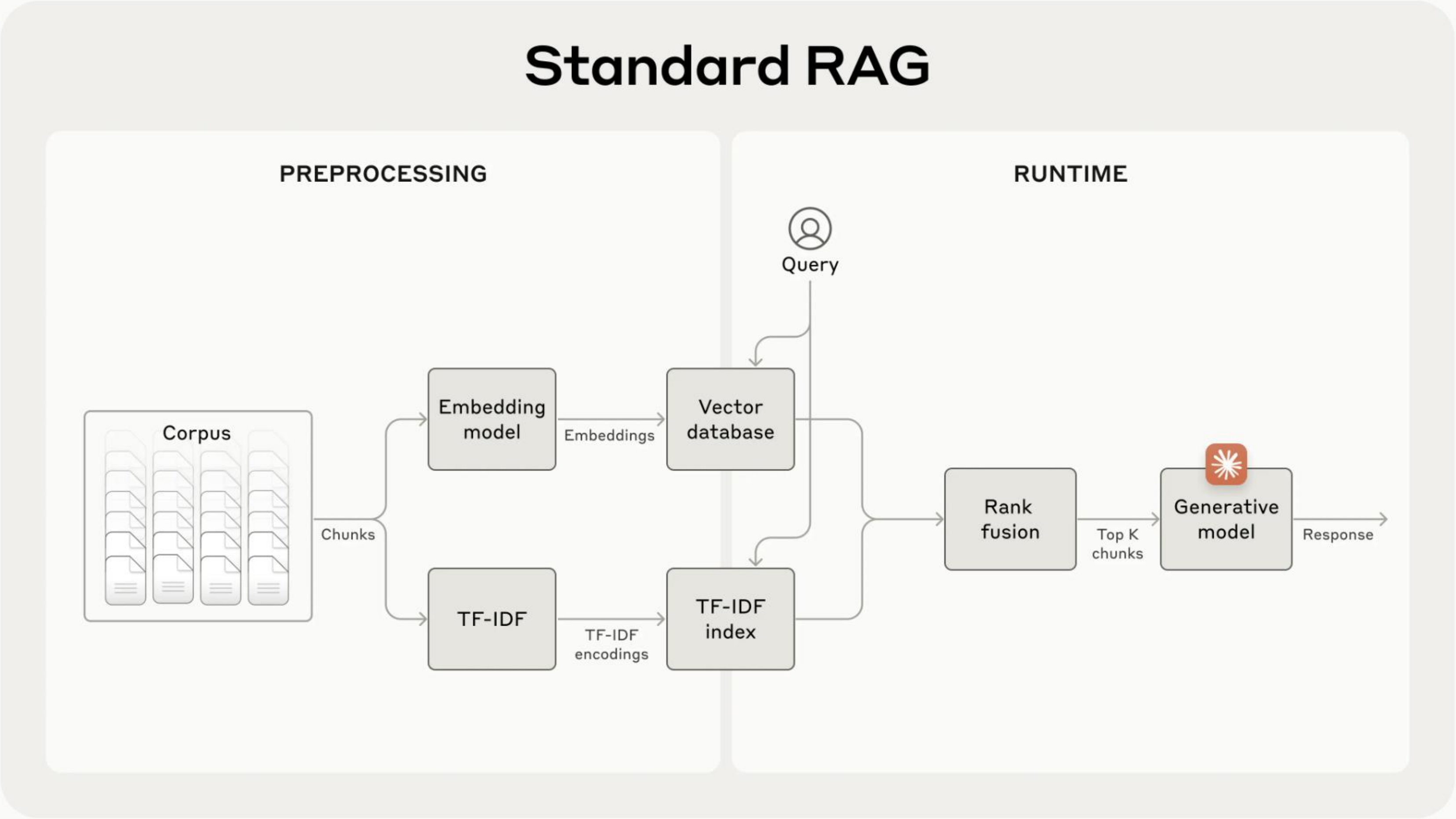
- Chunking Strategies
  - Implement semantic chunking based on section boundaries
  - Balance chunk size for context retention vs. token limits
  - Maintain cross-references between related chunks
  - Consider hierarchical chunking (chapter → section → subsection)
  - Preserve metadata for each chunk (figure references, citations)

Implementation Consideration on ETDs

- Retrieval Optimization
  - Index by semantic sections rather than arbitrary splits
  - Implement citation-aware retrieval for reference validation
  - Use hybrid retrieval combining semantic and keyword search
  - Create section-specific embeddings for targeted queries
  - Maintain context windows across related chunks

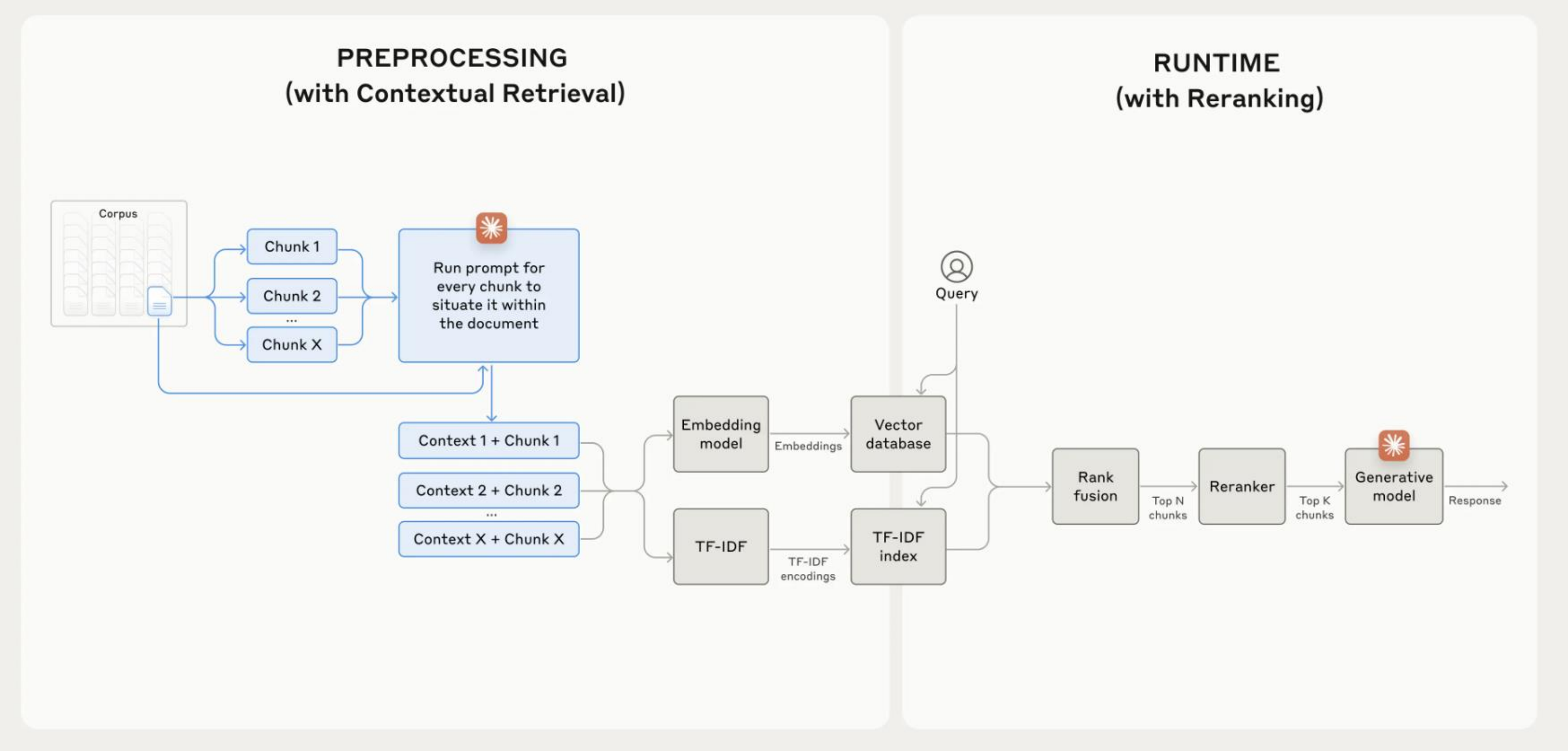
Variant RAGs

Image source: https://www.anthropic.com/news/contextual-retrieval



Variant RAGs

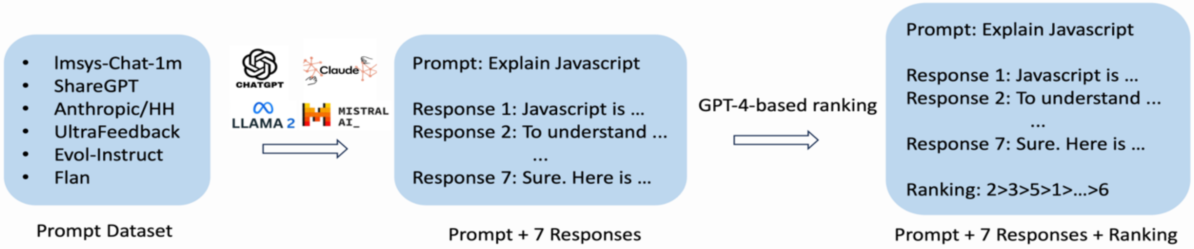
Image source: https://www.anthropic.com/news/contextual-retrieval



Fine-tuning

Starling-7B: Increasing LLM Helpfulness & Harmlessness with RLAIF

https://starling.cs.berkeley.edu/



- lmsys-Chat-1m
- ShareGPT
- Anthropic/HH
- UltraFeedback
- Evol-Instruct
- Flan

Prompt Dataset

Prompt: Explain Javascript

Response 1: Javascript is ...
Response 2: To understand ...
...
Response 7: Sure. Here is ...

Prompt + 7 Responses

GPT-4-based ranking

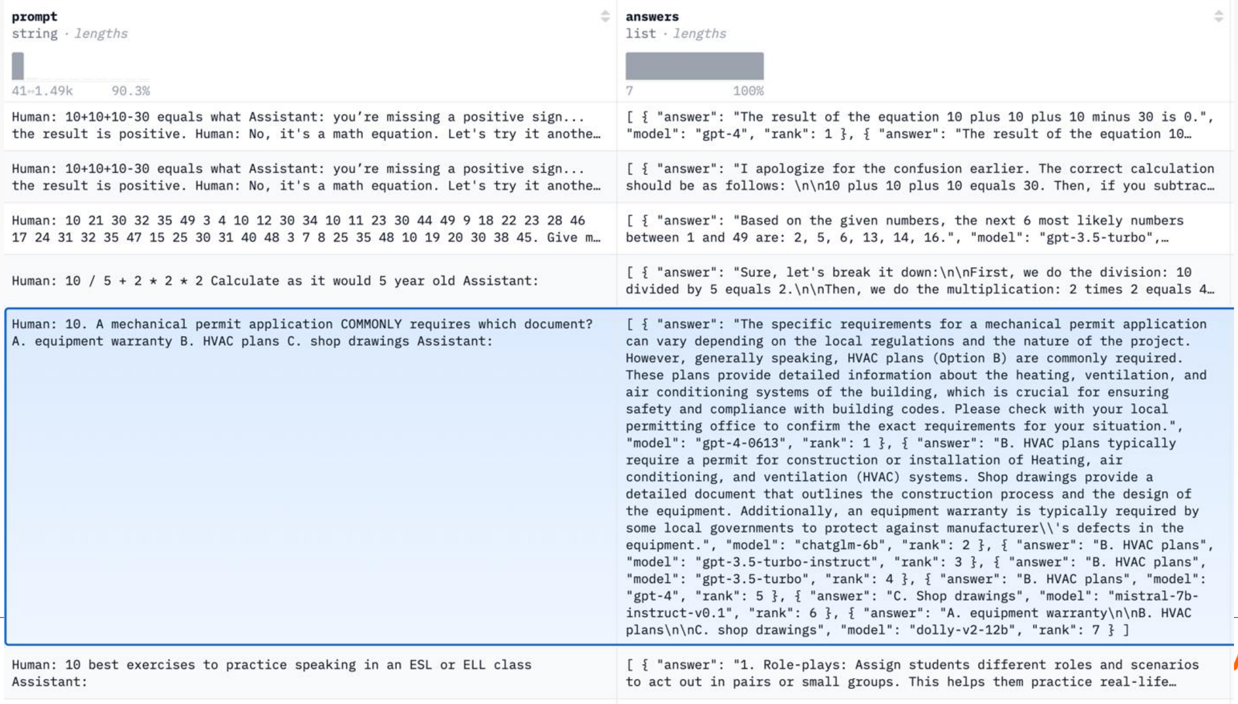
Prompt: Explain Javascript

Response 1: Javascript is ...
Response 2: To understand ...
...
Response 7: Sure. Here is ...

Ranking: 2>3>5>1>...>6

Prompt + 7 Responses + Ranking

*Illustrating the creation process of Nectar, a 7-wise comparison dataset for RLAIF.



prompt string · *lengths* 41↔1.49k 90.3%	answers list · *lengths* 7 100%
Human: 10+10+10-30 equals what Assistant: you're missing a positive sign... the result is positive. Human: No, it's a math equation. Let's try it anothe…	[{ "answer": "The result of the equation 10 plus 10 plus 10 minus 30 is 0.", "model": "gpt-4", "rank": 1 }, { "answer": "The result of the equation 10…
Human: 10+10+10-30 equals what Assistant: you're missing a positive sign... the result is positive. Human: No, it's a math equation. Let's try it anothe…	[{ "answer": "I apologize for the confusion earlier. The correct calculation should be as follows: \n\n10 plus 10 plus 10 equals 30. Then, if you subtrac…
Human: 10 21 30 32 35 49 3 4 10 12 30 34 10 11 23 30 44 49 9 18 22 23 28 46 17 24 31 32 35 47 15 25 30 31 40 48 3 7 8 25 35 48 10 19 20 30 38 45. Give m…	[{ "answer": "Based on the given numbers, the next 6 most likely numbers between 1 and 49 are: 2, 5, 6, 13, 14, 16.", "model": "gpt-3.5-turbo",…
Human: 10 / 5 + 2 * 2 * 2 Calculate as it would 5 year old Assistant:	[{ "answer": "Sure, let's break it down:\n\nFirst, we do the division: 10 divided by 5 equals 2.\n\nThen, we do the multiplication: 2 times 2 equals 4…
Human: 10. A mechanical permit application COMMONLY requires which document? A. equipment warranty B. HVAC plans C. shop drawings Assistant:	[{ "answer": "The specific requirements for a mechanical permit application can vary depending on the local regulations and the nature of the project. However, generally speaking, HVAC plans (Option B) are commonly required. These plans provide detailed information about the heating, ventilation, and air conditioning systems of the building, which is crucial for ensuring safety and compliance with building codes. Please check with your local permitting office to confirm the exact requirements for your situation.", "model": "gpt-4-0613", "rank": 1 }, { "answer": "B. HVAC plans typically require a permit for construction or installation of Heating, air conditioning, and ventilation (HVAC) systems. Shop drawings provide a detailed document that outlines the construction process and the design of the equipment. Additionally, an equipment warranty is typically required by some local governments to protect against manufacturer\\'s defects in the equipment.", "model": "chatglm-6b", "rank": 2 }, { "answer": "B. HVAC plans", "model": "gpt-3.5-turbo-instruct", "rank": 3 }, { "answer": "B. HVAC plans", "model": "gpt-3.5-turbo", "rank": 4 }, { "answer": "B. HVAC plans", "model": "gpt-4", "rank": 5 }, { "answer": "C. Shop drawings", "model": "mistral-7b-instruct-v0.1", "rank": 6 }, { "answer": "A. equipment warranty\n\nB. HVAC plans\n\nC. shop drawings", "model": "dolly-v2-12b", "rank": 7 }]
Human: 10 best exercises to practice speaking in an ESL or ELL class Assistant:	[{ "answer": "1. Role-plays: Assign students different roles and scenarios to act out in pairs or small groups. This helps them practice real-life…

Resources

- A Simple Guide to Retrieval Augmented Generation [eBook]
- AI-Powered Search [eBook]
- LangChain: A software framework that facilitates the integration of large language models into applications
- Anaconda AI Navigator: Interacting with open-source LLMs directly on your computer
- PaperQA2: a RAG for answering questions from scientific documents with citations
- Pyserini: a Python toolkit for reproducible information retrieval research

Discussion

Discussion

- Research and applications of Large Language Models (LLMs) in ETDs
- Improving ETD accessibility using LLMs
- Enhancing ETD discoverability through LLMs
- Joint discussion and sharing ideas

Future Workshops

- Call for RAG Implementation in ETDs Workshop
  - RAG Architecture and Design
  - Advanced RAG Research
  - Specialized RAG Topics

Q & A

Thank You!