

Large Language Models in AI

A Quick Overview

Michael Hahsler



This work is licensed under a [Creative Commons Attribution-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-sa/4.0/).



Online Material

Example Conversation With a Large Language Model (LLM)



Can you help me with my english homework?



Of course! As an AI language model, I would be happy to assist you with your English homework. Just let me know what you need help with, and I will do my best to provide you with the information and guidance you need.



Please write an 1000 word essay on the works of Shakespear.



Certainly! Here is a 1000-word essay on the works of Shakespeare:

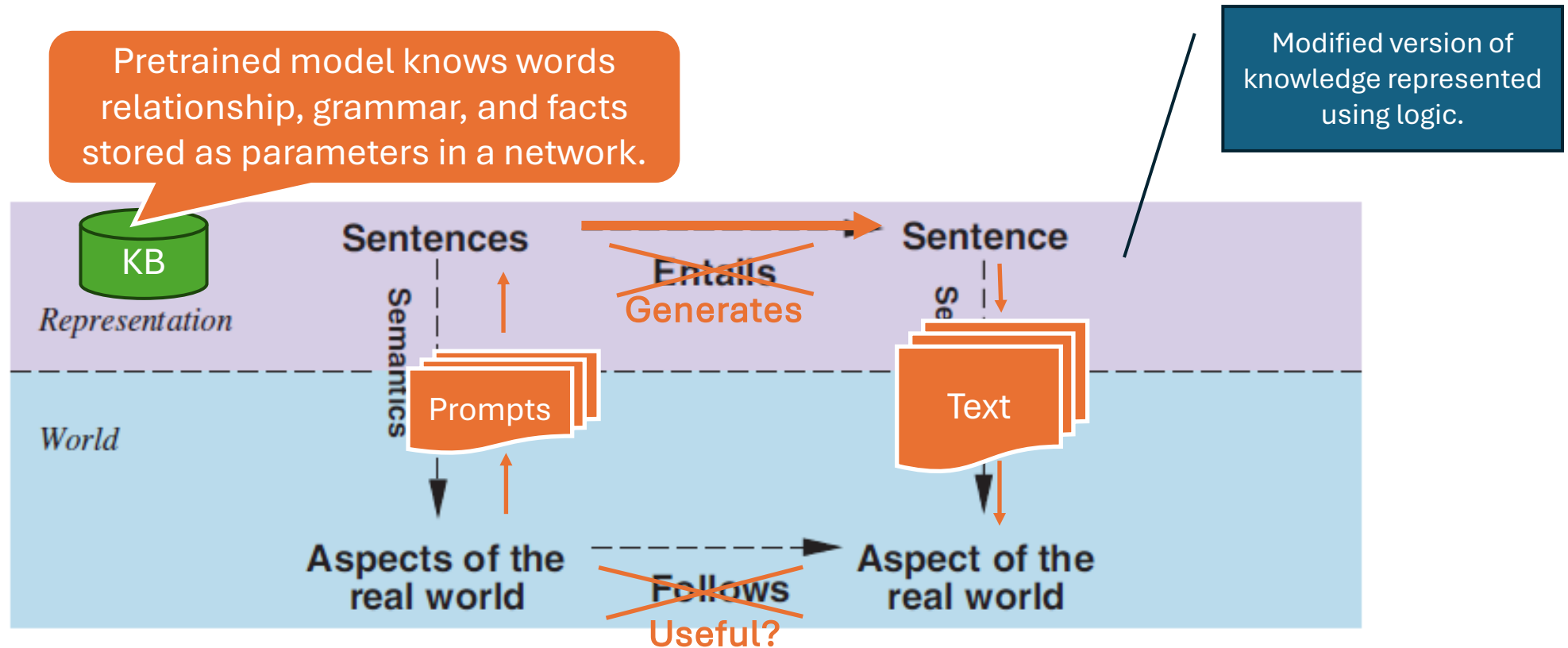
William Shakespeare is widely regarded as one of the greatest writers in the history of the English language. Born in Stratford-upon-Avon, England, in 1564, Shakespeare began his career as an actor before eventually becoming a playwright. He wrote plays, sonnets, and other works that have been studied, performed, and celebrated for centuries.

Shakespeare's plays are known for their rich characters, intricate plots, and poetic language. His works have been categorized into three main genres: tragedies, comedies, and histories. Each of these genres explores different themes and motifs, but they all share Shakespeare's distinct voice and style. ...

This knowledge
has to come
from
somewhere!

Source: <https://chat.openai.com/>

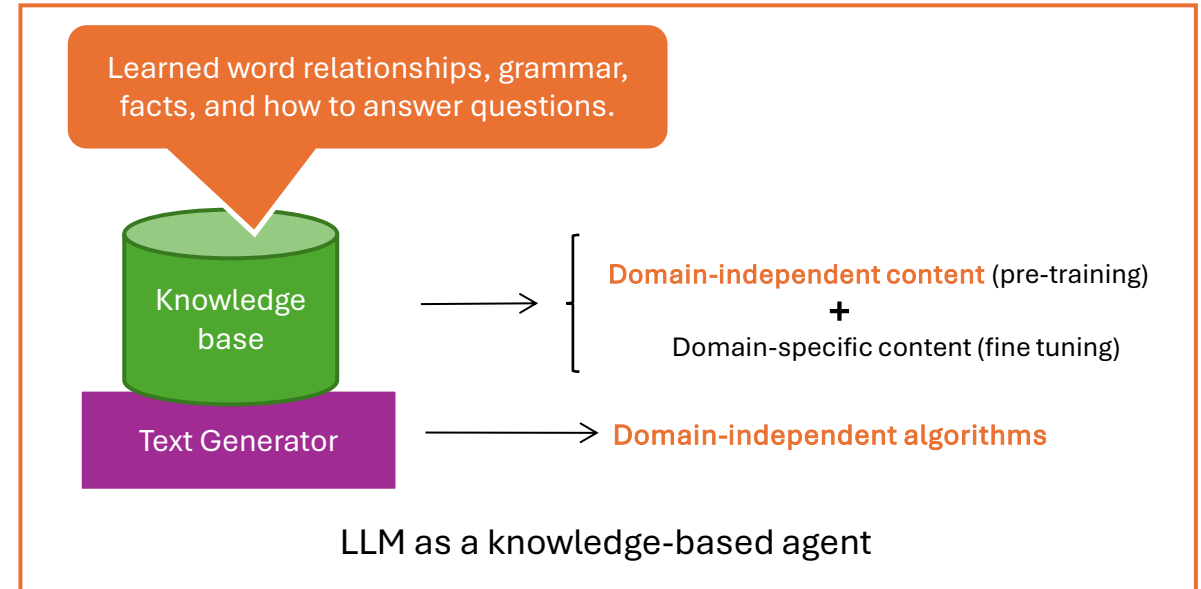
Using Natural Language for Knowledge Representation



- The user formulates a question about the real world as a natural language **prompt** (a sequence of tokens).
- The LLM **generates text** using a pretrained model that represents its knowledge base.
- The text (hopefully) is useful in the real world. The **objective function** is not clear. Maybe it is implied in the prompt?

What are LLMs

- Large language models are **knowledge-based agents**.
- The knowledge base is a large, deep artificial neural network using the **transformer architecture**.
- The model is **pretrained** for the task of predicting the next word using self-supervised learning on a large text corpus.
- The pretrained model is specialized for a specific task using **fine-tuning** on examples. For chatbots, this includes
 - supervised instruction training followed by
 - reinforcement learning from human feedback (RLHF).
- The model can generate text following a “**prompt**.”

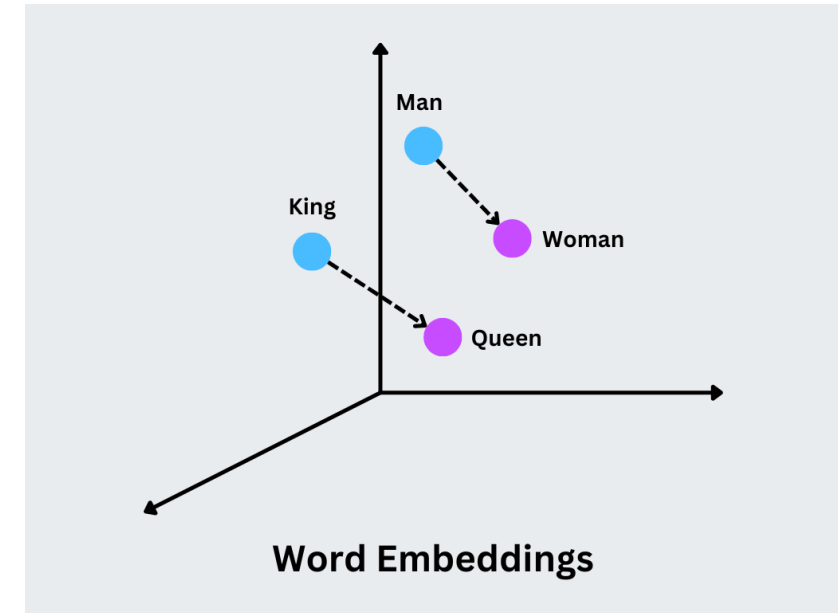


Pretrained models can be downloaded from
Hugging Face: <https://huggingface.co/models>

How LLMs Understand Text

- LLMs can only compute with numbers!
- Prompts are converted into numbers:
 1. **Tokenization:**
 - Text is converted into a sequence of tokens (ids).
 - Tokens represent words, parts of words, parts of numbers, etc.
 2. **Embedding:**
 - Tokens are converted to large vectors indicating a position in a high-dimensional space called the embedding space.
 - Positions in this space have meaning.
- Computation is done by calculations (moving around) in the embedding space.
- New words are generated from resulting vectors by learning a function that maps vectors back to tokens.

Natural Language Processing (NLP)



Source: <https://zilliz.com/glossary/word2vec>

Prompts and Generation

- LLMs can only perform **next-token prediction**.
- The most likely next word is predicted based on the **context**, which is the input window (percept) for the LLM.
- The context is typically a combination of the
 - a) **system prompt** (general instructions you often don't see)
 - b) more **context information** (documents, additional information)
 - c) the **user prompt**
- The context window size is limited by the model, and the model “forgets” everything that drops out of the window.
- The complete response is created **token-by-token** (autoregressive generation). LLMs may hallucinate by creating low-probability tokens, and autoregression will amplify this issue.

Example Context:

You are a helpful assistant who answers user questions based on the following information.

This week's schedule:

Monday 10 AM – AI class

Wednesday 12 AM – Data Mining class

When do I need to go to the AI class?

System Prompt

User Prompt

Additional context (e.g., from a database)

How LLMs Predict the Next Word

- Example:** “Tomorrow is a good” ?

x_{i-4}
 x_{i-3}
 x_{i-2}
 x_{i-1}
 x_i

Context window

Predict the next word

X_i is a random variable representing a word/token at position i .
- Model:** The optimal MAP decision for the next word using the Bayes rule is

$$\operatorname{argmax}_{x_i} P(X_i = x_i | X_{i-1} = x_{i-1}, X_{i-2} = x_{i-2}, \dots, X_{i-l} = x_{i-l}) \propto P(x_{i-1}, x_{i-2}, \dots, x_l | x_i) P(x_i)$$

Mask the next word = self-supervised learning
- Learning/Estimation:** From a large amount of data “Tomorrow is a good [day].”

$$P(x_i) = \frac{N(x_i)}{N}$$

$$P(x_{i-1}, x_{i-2}, \dots, x_l | x_i) = \frac{N(x_i, x_{i-1}, x_{i-2}, \dots, x_{i-l})}{N(x_i)}$$

Joint Probability as a table has all the information $P(X_i, X_{i-1}, X_{i-2}, \dots, X_{i-l})$
- Model size:** vocabulary size d , context length $l \rightarrow O(d^l)$
- Example GPT 3** uses $d = 50,000, l = 2000$
 - Bayesian classifier : $O(50000^{2000}) \approx O(10^{9400})$
 - GPT 3 models $P(X_i | X_{i-1}, X_{i-2}, \dots, X_{i-l})$ with only 175 billion parameters ($O(10^{11})$)

This does not fit in memory and we cannot estimate that many numbers!

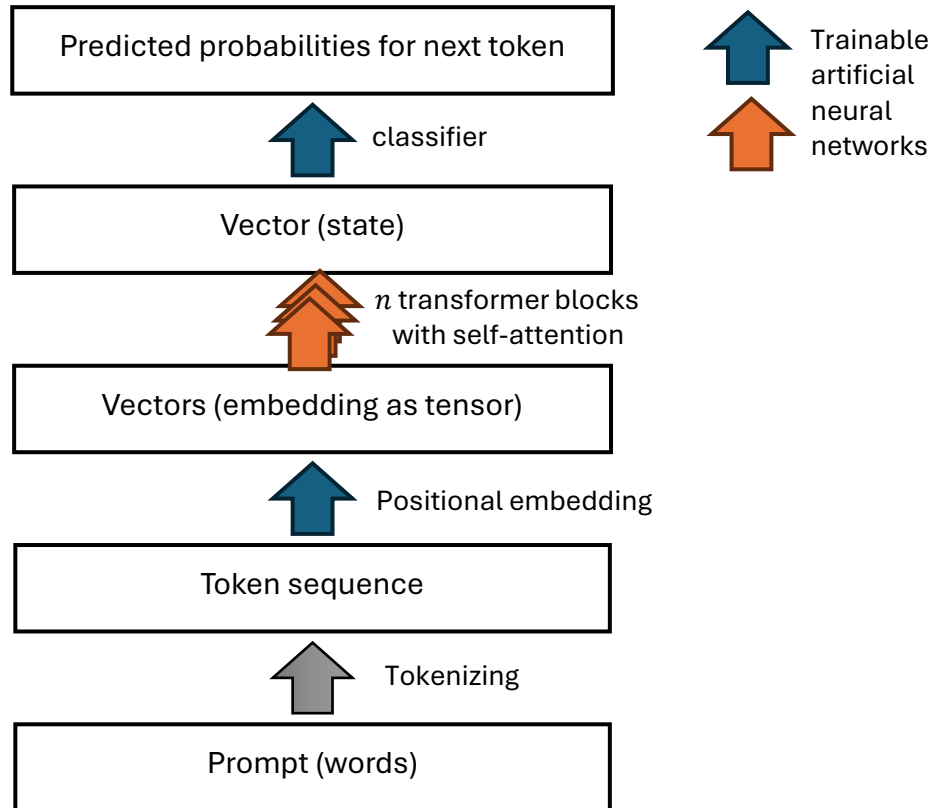
Tricks for compression in transformers are positional encoding and the attention mechanism.

With 16-bit floats, this needs > 350 GB RAM

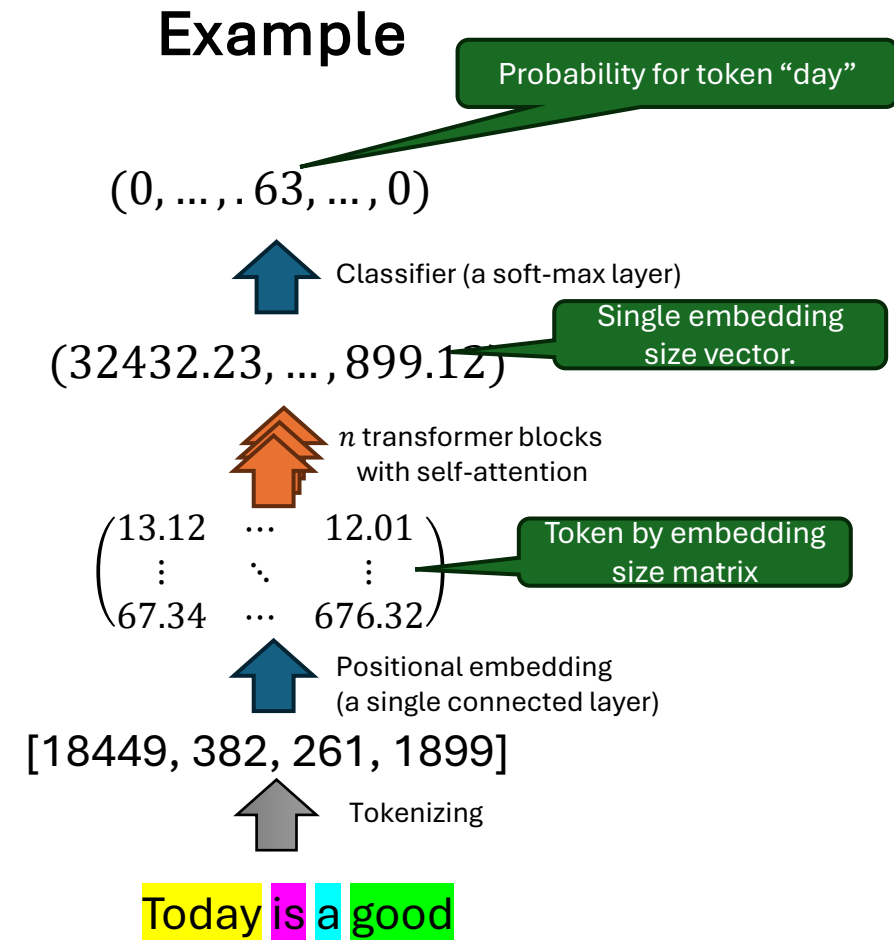
X_{i-l}	...	X_{i-3}	X_{i-2}	X_{i-1}	X_i	P
-	...	is	a	good	day	0.01
-	...	is	a	good	reason	1e-12

$O(d^l)$ rows!

Structure of a GPT-type LLM



Structure of a GPT-type LLM using a decoder only transformer architecture



Prompt Engineering / Context Engineering

- Design effective natural language instructions (prompts) to guide generative AI models toward desired outputs, such as text, images, or code.
 - **Zero-shot prompt:** directly instructs the model to perform a task without any additional examples.
 - **Few-Shot Prompting:** Provide demonstrations in the prompt to steer the model to better performance. This is also called “in-context learning”
 - **Chain-of-thought:** solve a problem as a series of intermediate steps before giving a final answer.
Note: Many modern chatbots include chain-of-thought into the system prompt and call this **reasoning models**!

Examples:

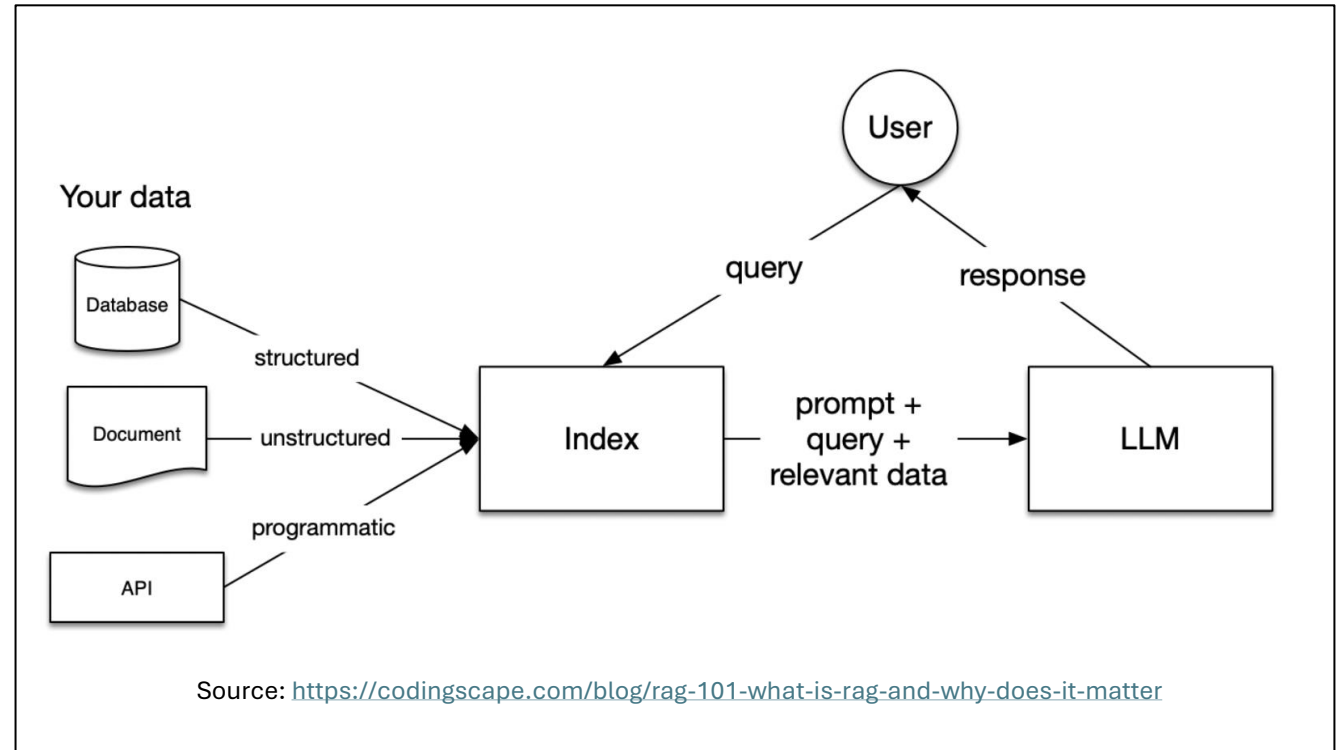
6 times 15 minus 3 is

Review: Loved it! Output: Positive
Review: Hated it. Output: Negative
Review: It was good. Output:

How much is 6 times 15 minus 3? *Let's think step-by-step.*

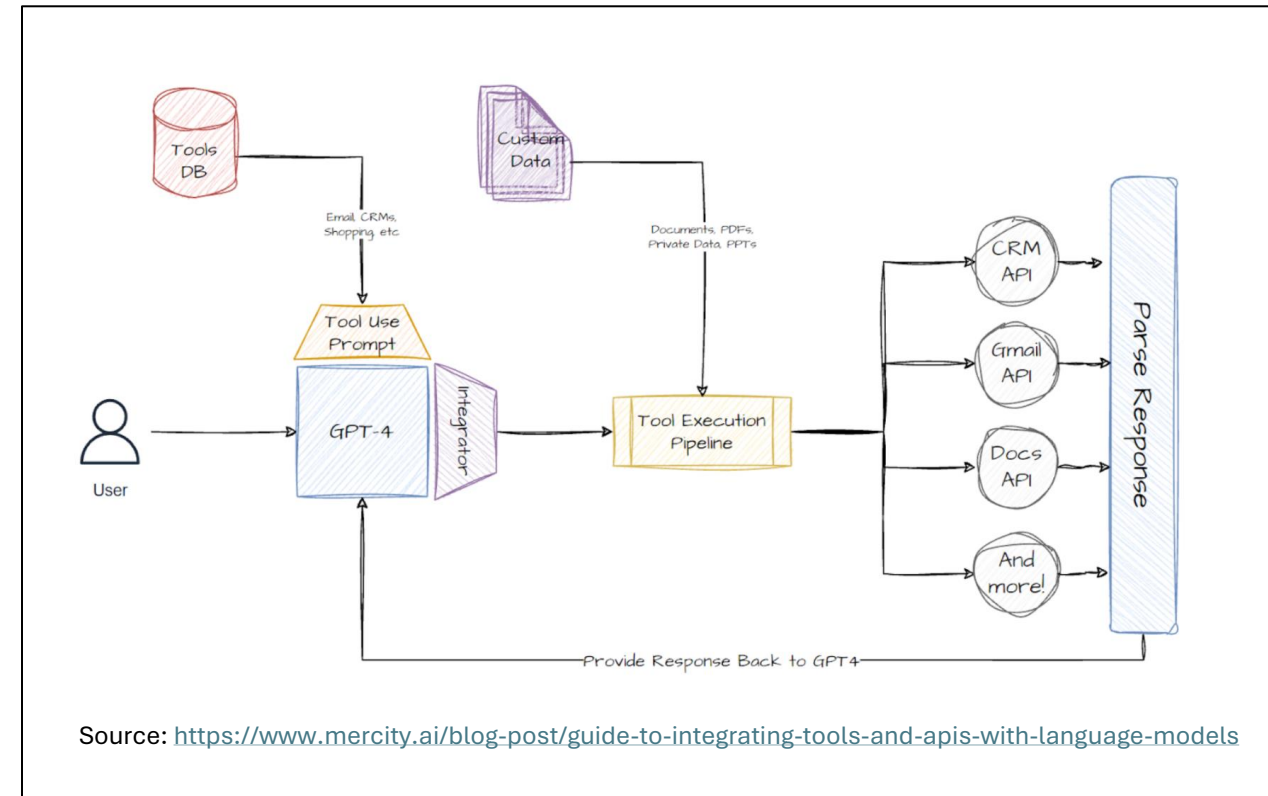
Retrieval Augmented Generation (RAG)

- RAG combines an information retrieval component with a text generator model.
- Approach:
 1. RAG takes an input and retrieves a set of relevant/supporting documents given a source (e.g., Wikipedia).
 2. The documents are concatenated as context with the original input prompt and fed to the text generator LLM.
 3. The LLM produces the final output.
- This approach can reduce issues with hallucination and introduce information that was not in the training data.
- Tools: LangChain, LambdaIndex, Flowise,...



LLMs that use Tools

- Enables LLMs to use tools like browsing the web, a calculator, or executing code.
- **Training:** LLMs are finetuned using demonstrations of how the tool is correctly used.
- **Execution:**
 1. The LLM generates a structured output that signals a tool call. E.g.: “browser(http://...)”
 2. The tool call is intercepted by the surrounding program, and the tool is executed.
 3. The tool’s result is returned to the LLM to create the final output.
- **Frameworks:** LangChain, LangGraph, OpenAI Agents SDK, ...



LLMs and Images

- **Image interpretation:** image as the input
 - LLMs **convert visual data into locations in the embedding space** (using image encoders and multimodal alignment).
 - The location in the embedding space connects the image to tokens. This connection enables LLMs to describe the image.
 - **Training:** LLMs that process images are usually trained with multimodal datasets and tasks like image-text matching, predicting missing parts of an image, etc.
- **Image generation:** Image as the output
 - LLMs do not directly generate images.
 - The LLM is trained to generate a textual description of the desired image and then uses an **external tool** like a diffusion model (DALL-E, Stable Diffusion, Imagen).

Argentic AI

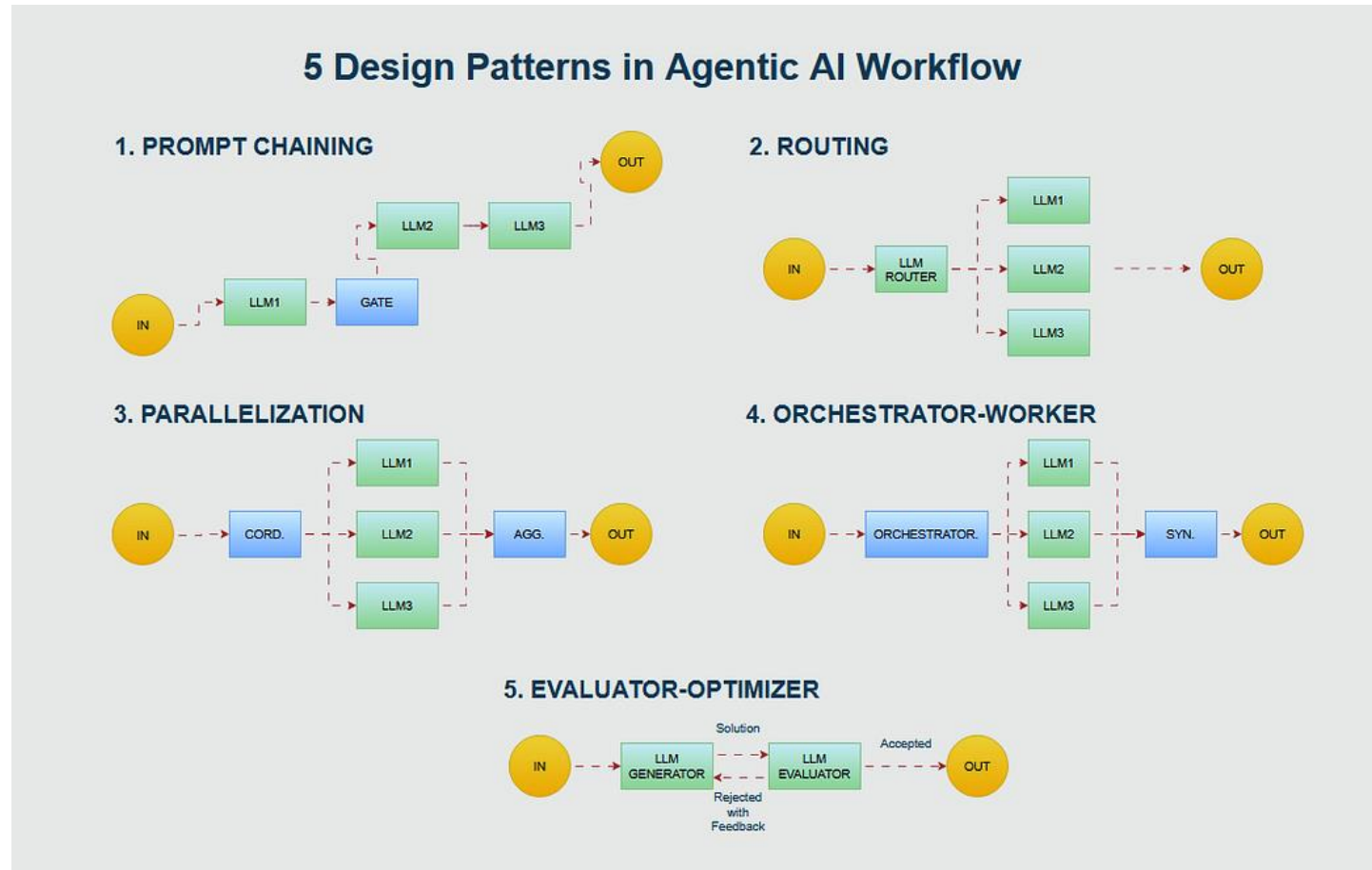
An AI solution that uses a set of specially prompted LLM calls.

The solution involves any or all of these:

- **Multiple LLM calls**
 - LLMs can use **tools** (browse the web, access files, etc.) to interact with an environment.
 - A **planner** coordinates the activities of the agents: Can be a developer-defined workflow or use an LLM to plan its own.
-
- **Tools:** LangChain, LangGraph, CrewAI, ...

Argentic AI Strategies

- **Prompt chaining:** each step has a focused job, making each LLM query simpler and more accurate.
- **Routing** is about smartly directing inputs to the right handler (e.g., LLMs trained differently).
- **Parallelization** lets the agent do multiple things at the same time.
- **Orchestration:** An agent takes the input, breaks it into sub-tasks on the fly, and then dispatches each subtask to separate “worker” agents.
- **Evaluator–optimizer** is a feedback loop. One agent generates a solution, and another evaluates it and suggests improvements.



Source: <https://pub.towardsai.net/5-design-patterns-in-agentic-ai-workflow-c972c83f77e4>

Conclusion

- LLMs are **knowledge-based agents** trained for the next-word prediction task.
- LLMs can be fine-tuned for specific purposes (e.g., a chatbot), to **understand images**, and to **use tools**.
- **Prompt engineering** (context engineering) is important to produce better responses from the LLM.
- **Retrieval Augmented Generation (RAG)** can be used to include information that was not in the training data.
- **Agentic AI** uses multiple LLM calls to deal with more complicated problems.
- Some issues:
 - **Hallucinations**; the LLM does not say when it is not sure. RAG can help.
 - **Privacy and data security** when a third party hosts the LLM. Training data can be poisoned.
 - **Safety**: It is easy to avoid guardrails (i.e., jailbreak) leading to unsafe responses.
 - LLMs currently have no (or very little) **common sense**.
 - LLMs' performance is influenced by small prompt changes. The results are not very **dependable**.
 - **Reasoning** capabilities are very limited. Agentic AI tries to mitigate this issue by focusing on smaller tasks.

