Generative AI and Large Language Model in PHP

Enrico Zimuel, *Tech Lead & Principal Software Engineer*

May 17, Verona 2024

Agenda

- Generative AI
- Deep neural network
- Large Language Model
- Prompt engineering
- OpenAI for PHP
- Retrieval Augmented Generation (RAG)
- Embedding and Vector Search
- LLPhant for PHP

Image generated using dall-e-3

elastic

Examples generated using OpenAI

Image generated using dall-e-3 with the prompt:
A busy developer working on a laptop during a conference

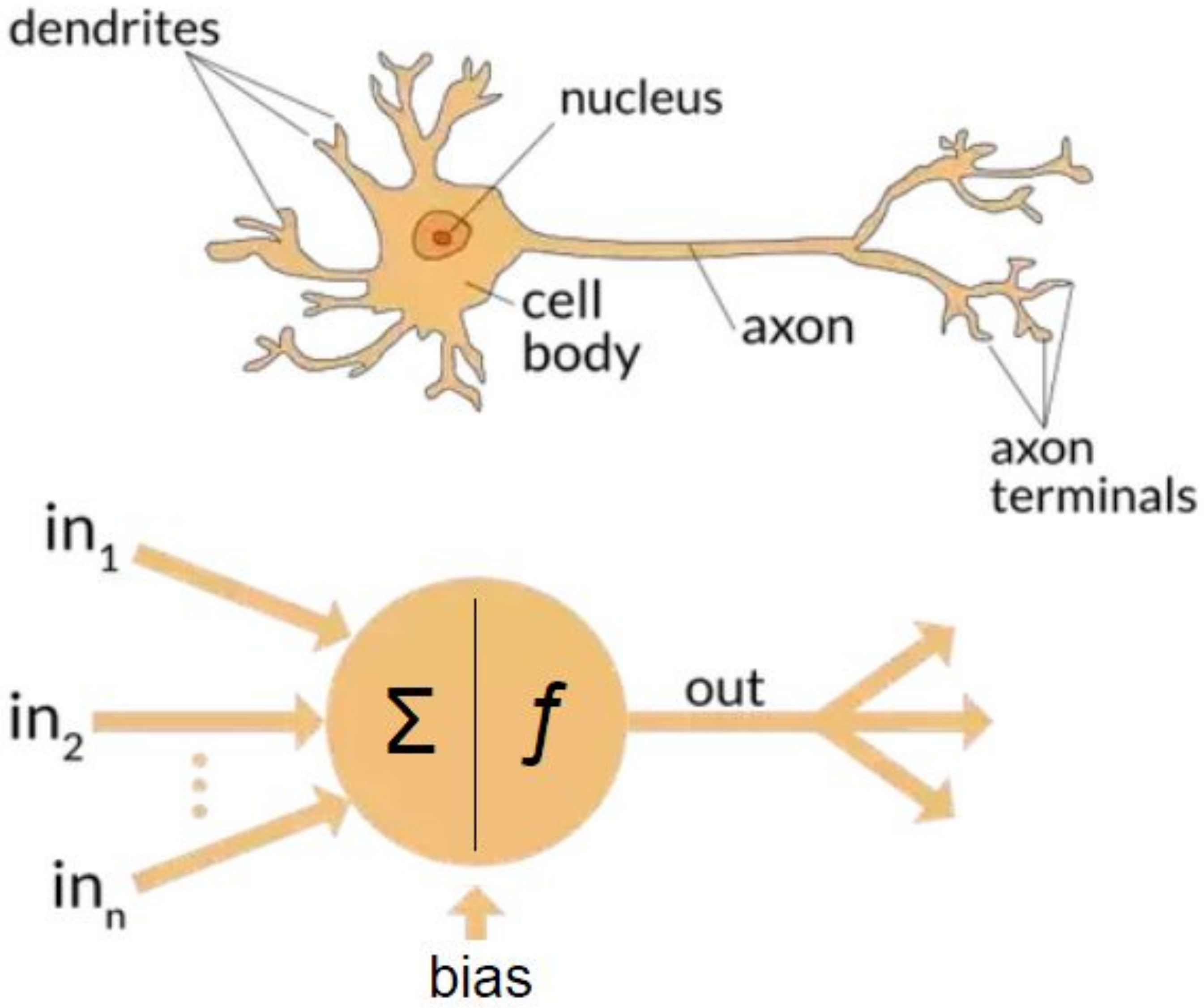
Audio file generated using tts-1

Generative AI

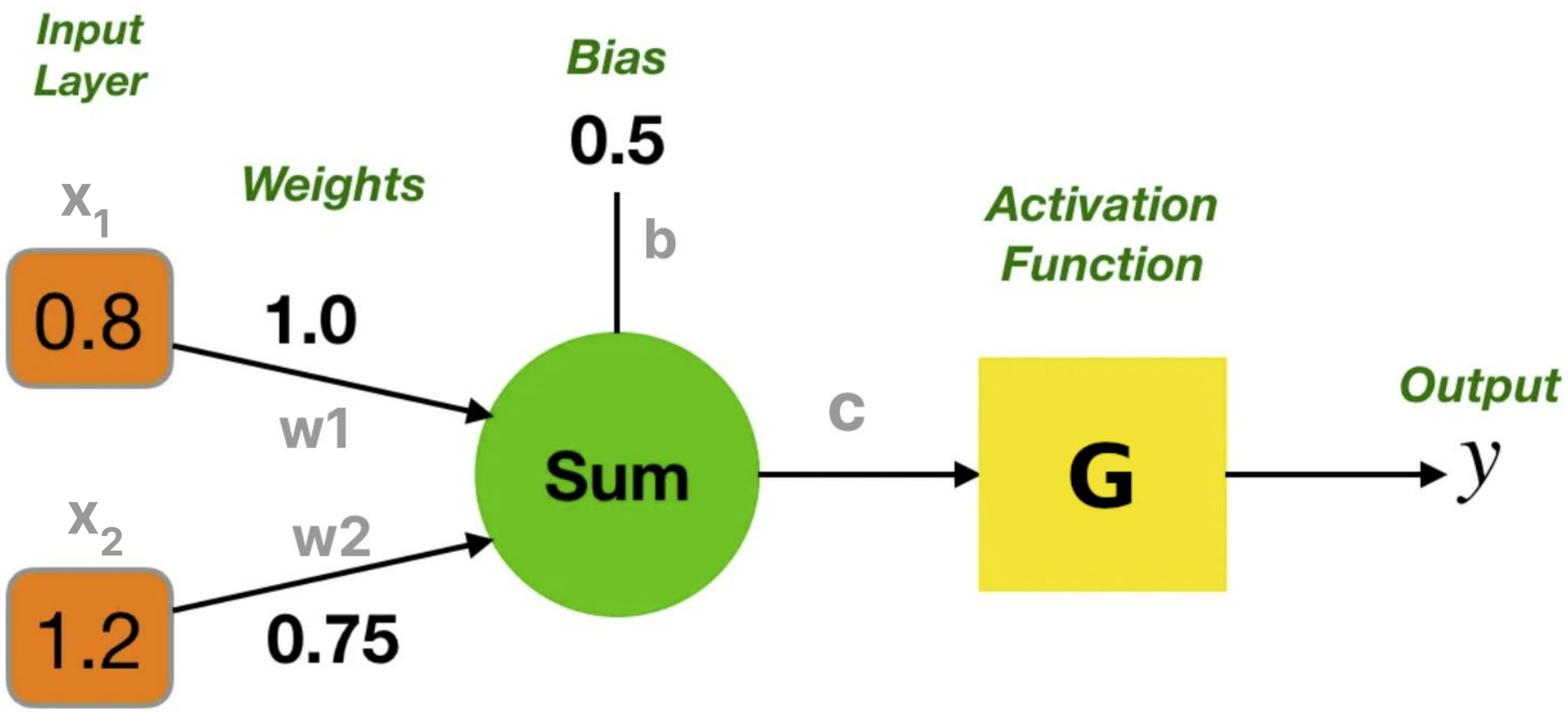
- **Generative Artificial Intelligence** (GenAI) is a subset of **deep learning** capable of generating text, images, or other media, using generative models
- GenAI models **learn the patterns and structure** of their input training data and then generate new data that has **similar characteristics**
- It's used in many industries: art, writing, coding, healthcare, finance, gaming, marketing, etc
- The global generative ai market was valued at $10.5 billion in 2022, and is projected to reach $191.8 billion by 2032

Neural Network

- A **neural network** is a method that teaches computers to **process data** in a way that is inspired by the human brain
- Collection of **nodes** (artificial neurons) with inputs and outputs. A node computes some non-linear function of the sum of its inputs
- The nodes are collected in **layers**
- If the number of layers > 3 we call it **deep learning network**

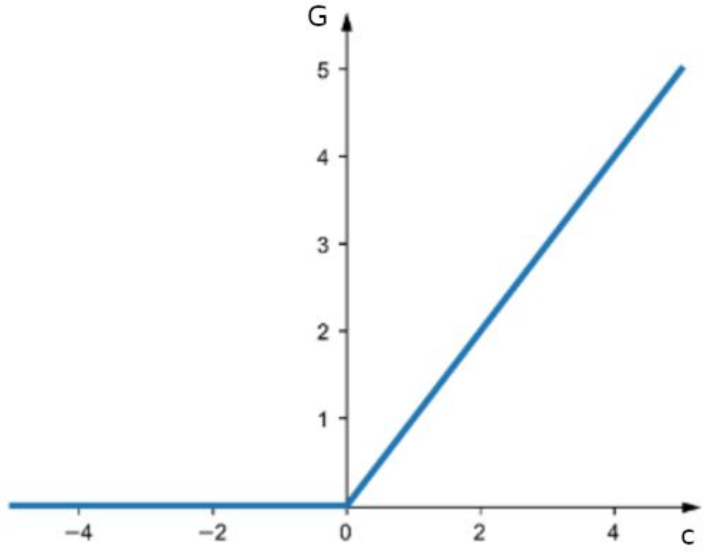


Single layer neuron



Example: ReLU activation

G(c)= c≥0 ? c : 0

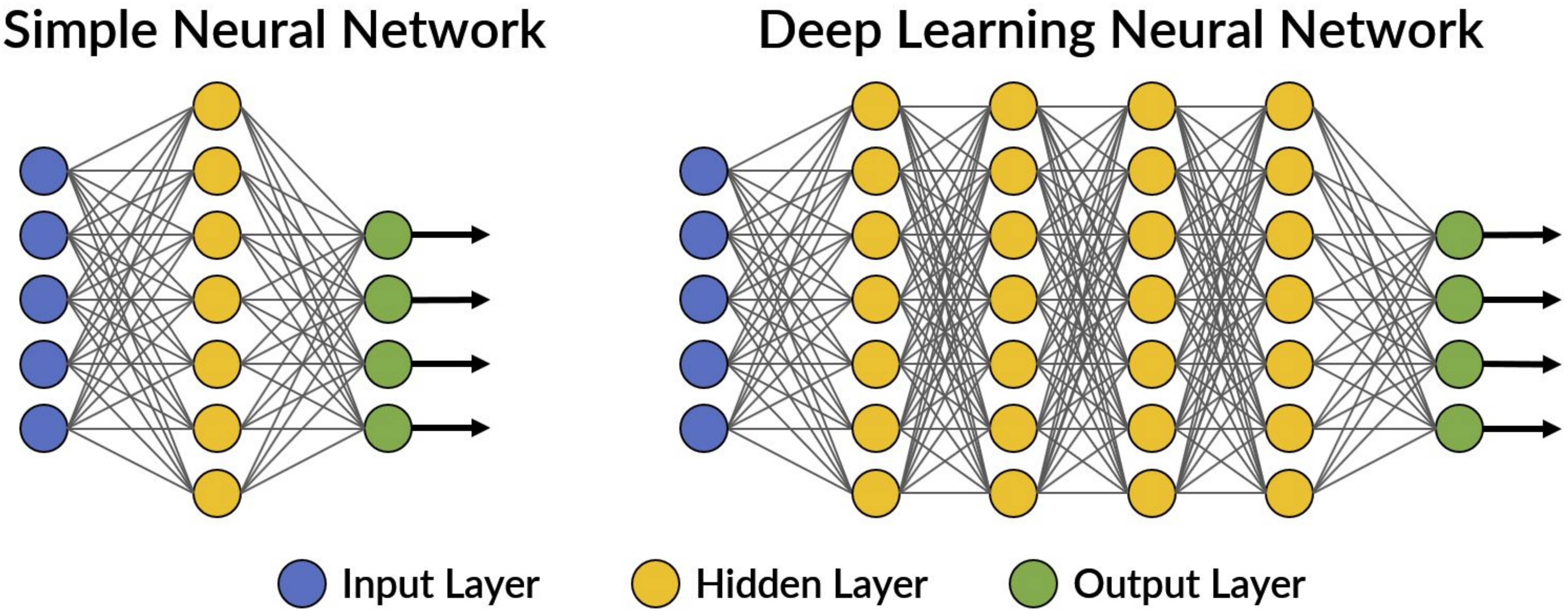


$c = x_1 * w_1 + x_2 * w_2 + b$

$= 0.8*1.0 + 1.2*0.75 + 0.5$

$= 2.2$

$y = G(c) = G(2.2) = 2.2$



Simple Neural Network

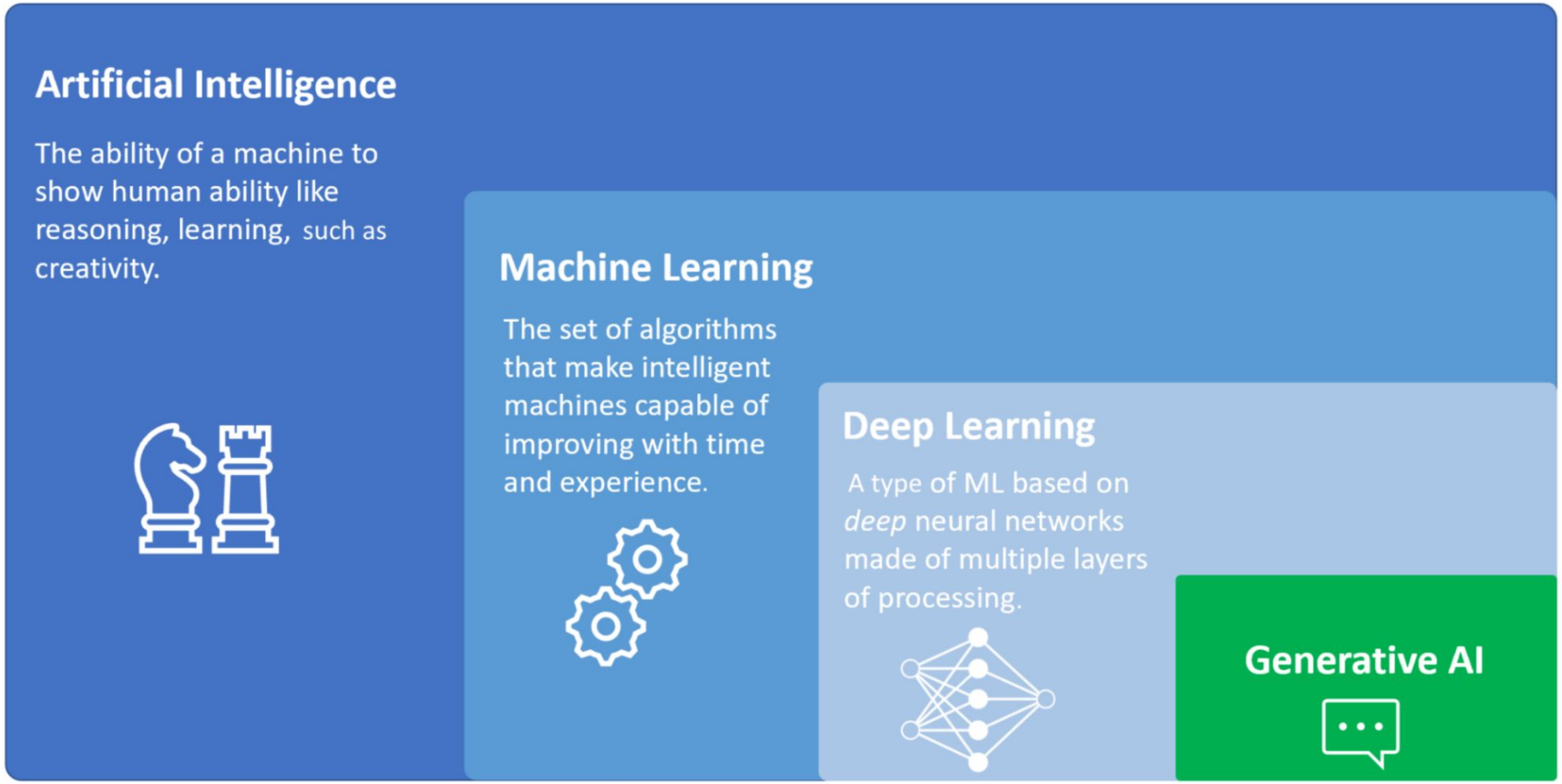
Deep Learning Neural Network

● Input Layer

● Hidden Layer

● Output Layer

AI ⊃ ML ⊃ DL ⊃ GenAI



Artificial Intelligence

The ability of a machine to show human ability like reasoning, learning, such as creativity.

Machine Learning

The set of algorithms that make intelligent machines capable of improving with time and experience.

Deep Learning

A type of ML based on *deep* neural networks made of multiple layers of processing.

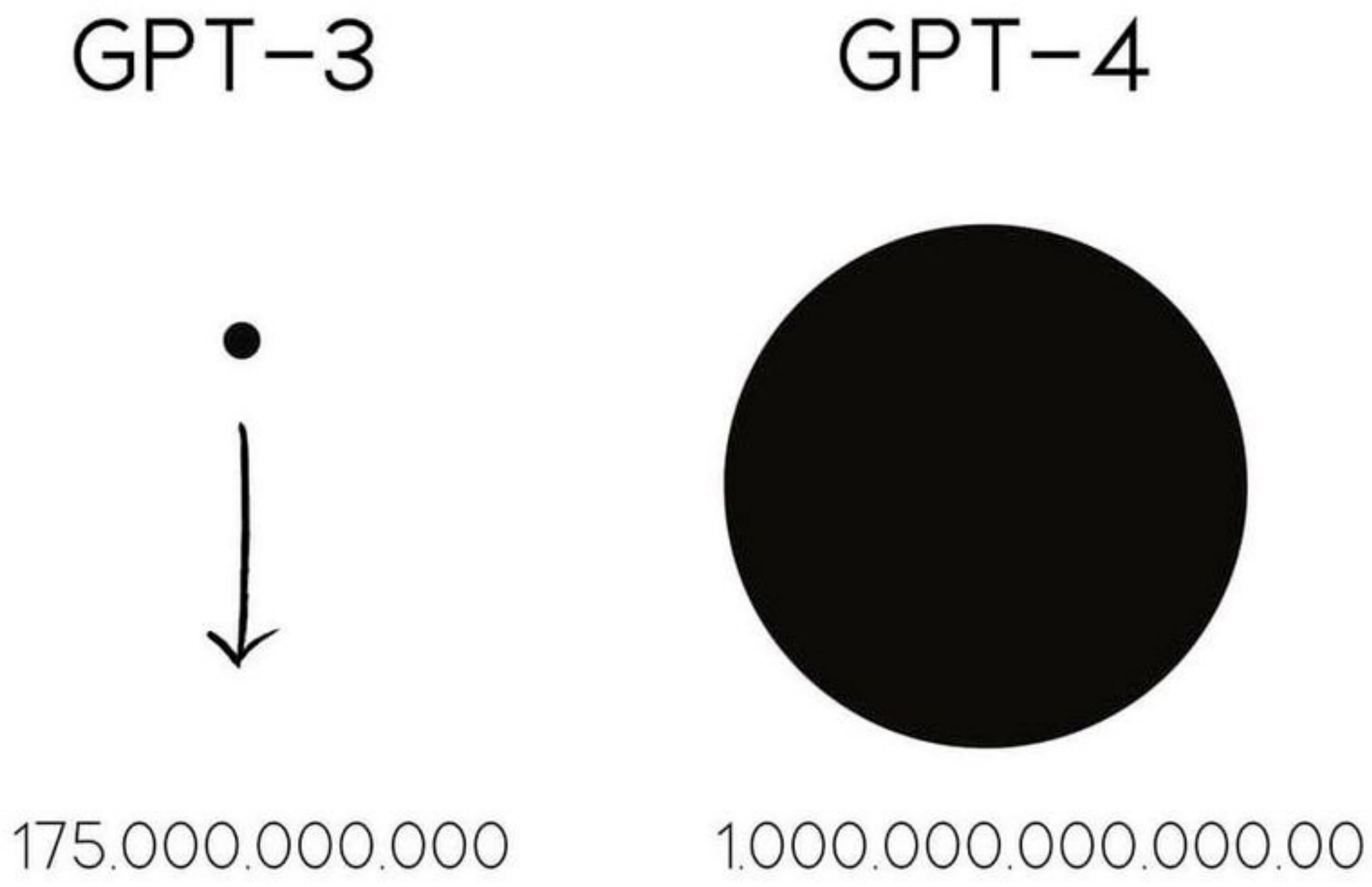
Generative AI

LLM

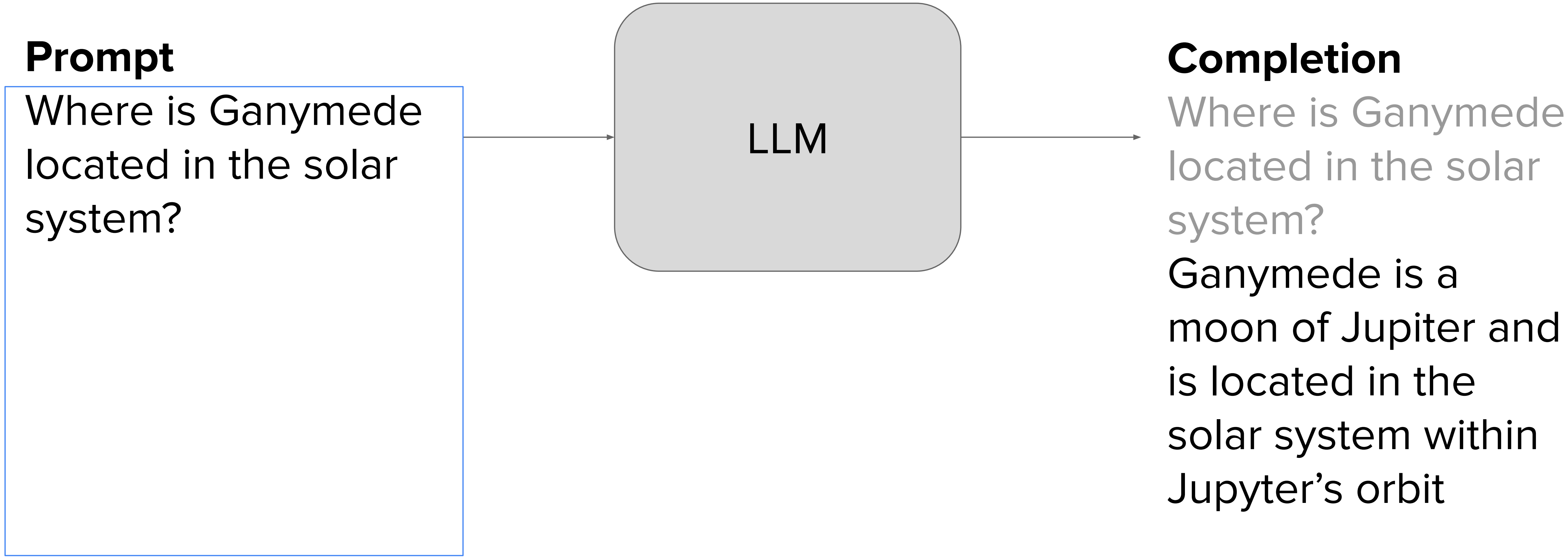
- **Large Language Model** (LLM) consisting of a neural network with many parameters (typically billions of weights or more), trained on large quantities of unlabelled text using self-supervised learning
- A message is splitted in **tokens**
- Each token is translated in a number using an operation called **embeddings**
- LLM works by taking an input text and **repeatedly predicting** the next token or word
- Since it's based on really big deep learning networks, no one fully understand how it works internally

Size of GPT-4

- Around **1.76 trillion** parameters
- Neural network with **120** layers
- Process up to **25,000** words at once
- Estimated training cost is $200M using 10,000 Nvidia A100 GPU for 11 months



Prompt



Prompt

Where is Ganymede located in the solar system?

LLM

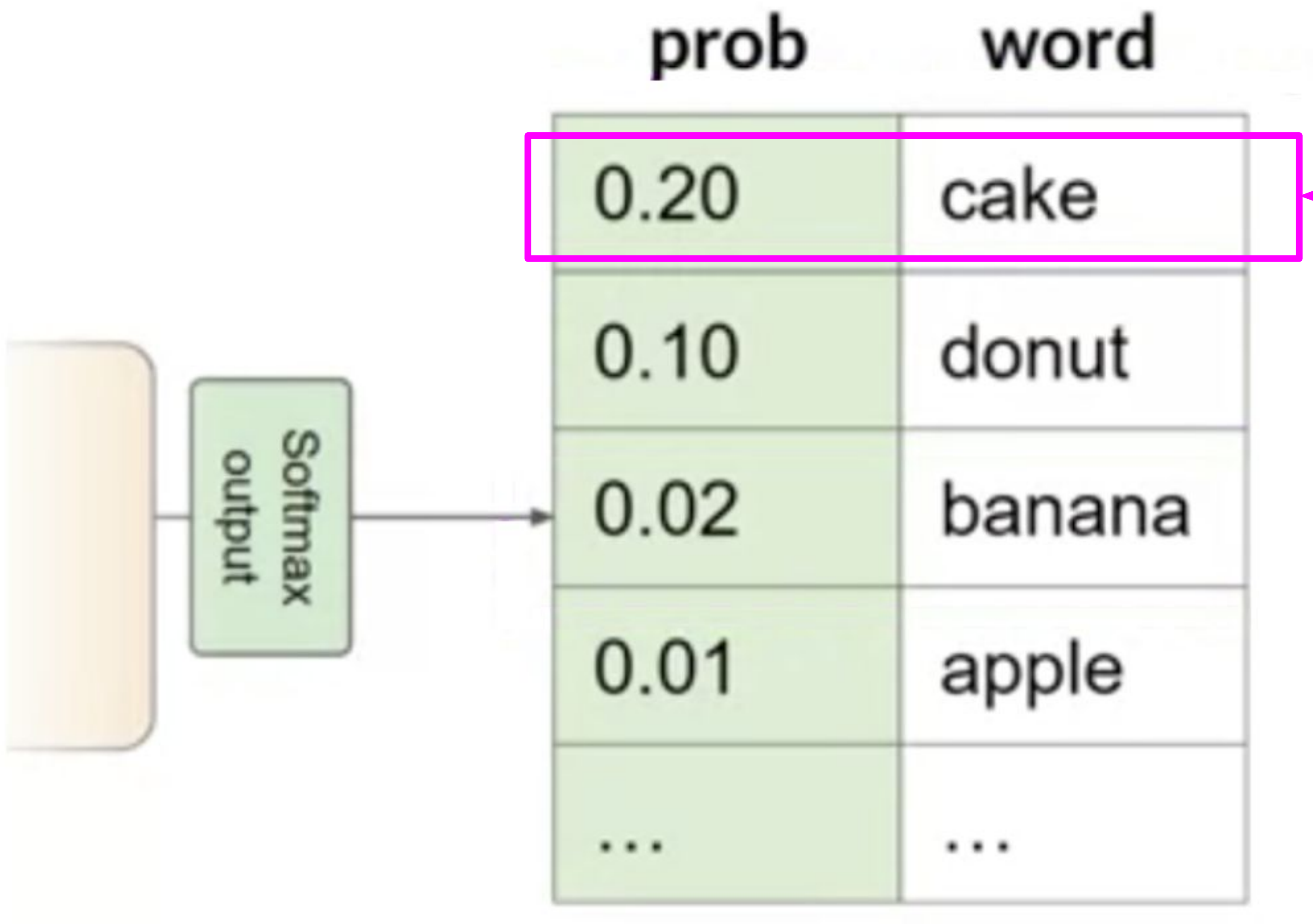
Completion

Where is Ganymede located in the solar system?

Ganymede is a moon of Jupiter and is located in the solar system within Jupyter’s orbit

Context window: few thousand words

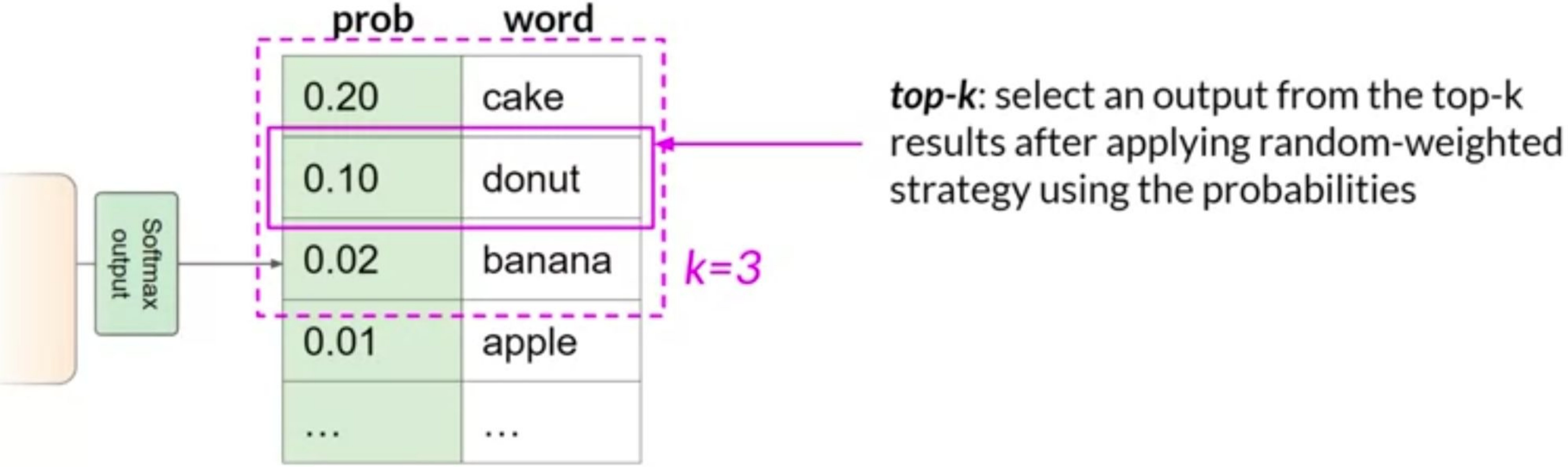
Predict the next word



prob	word
0.20	cake
0.10	donut
0.02	banana
0.01	apple
...	...

Choose the one with greatest probability (greedy algorithm)

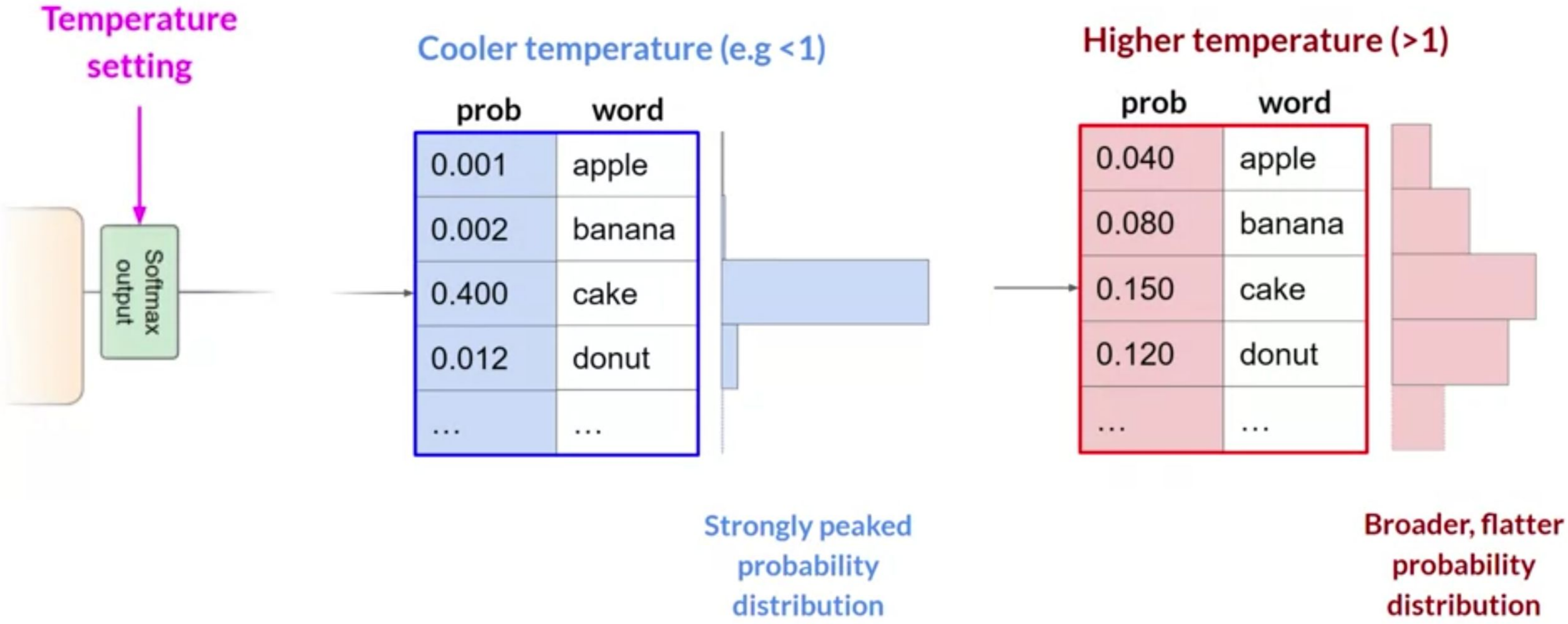
Top-k



prob	word
0.20	cake
0.10	donut
0.02	banana
0.01	apple
...	...

top-k: select an output from the top-k results after applying random-weighted strategy using the probabilities

Temperature



Cooler temperature (e.g <1)

prob	word
0.001	apple
0.002	banana
0.400	cake
0.012	donut
...	...

Strongly peaked probability distribution

Higher temperature (>1)

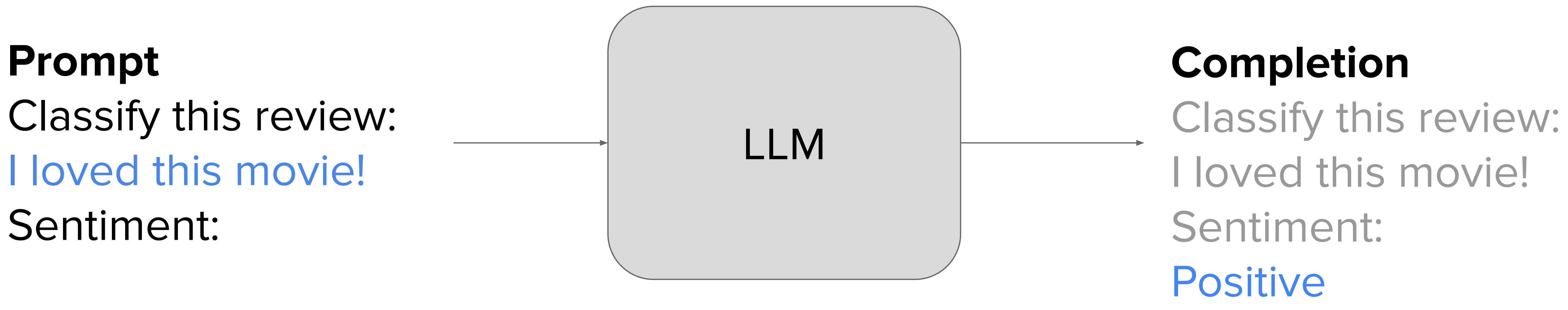
prob	word
0.040	apple
0.080	banana
0.150	cake
0.120	donut
...	...

Broader, flatter probability distribution

Prompt engineering

- You can encounter situations where the model doesn't produce the outcome that you want on the first try
- You may have to revisit the language several times to get a good answer
- The development and improvement of the prompt is known as **prompt engineering**
- One powerful strategy is to include examples of the task that you want the model to carry out inside the prompt
- This is called **In-Context Learning (ICL)**

ICL - zero shot inference

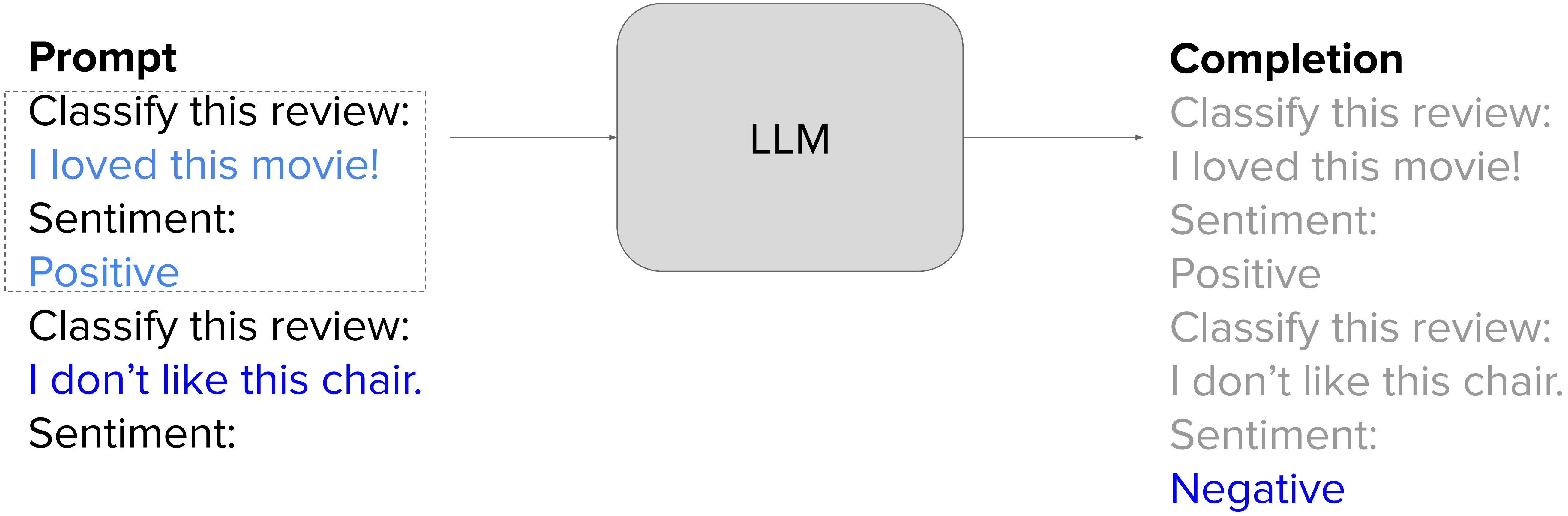


Prompt
Classify this review:
I loved this movie!
Sentiment:

LLM

Completion
Classify this review:
I loved this movie!
Sentiment:
Positive

ICL - one shot inference



Prompt

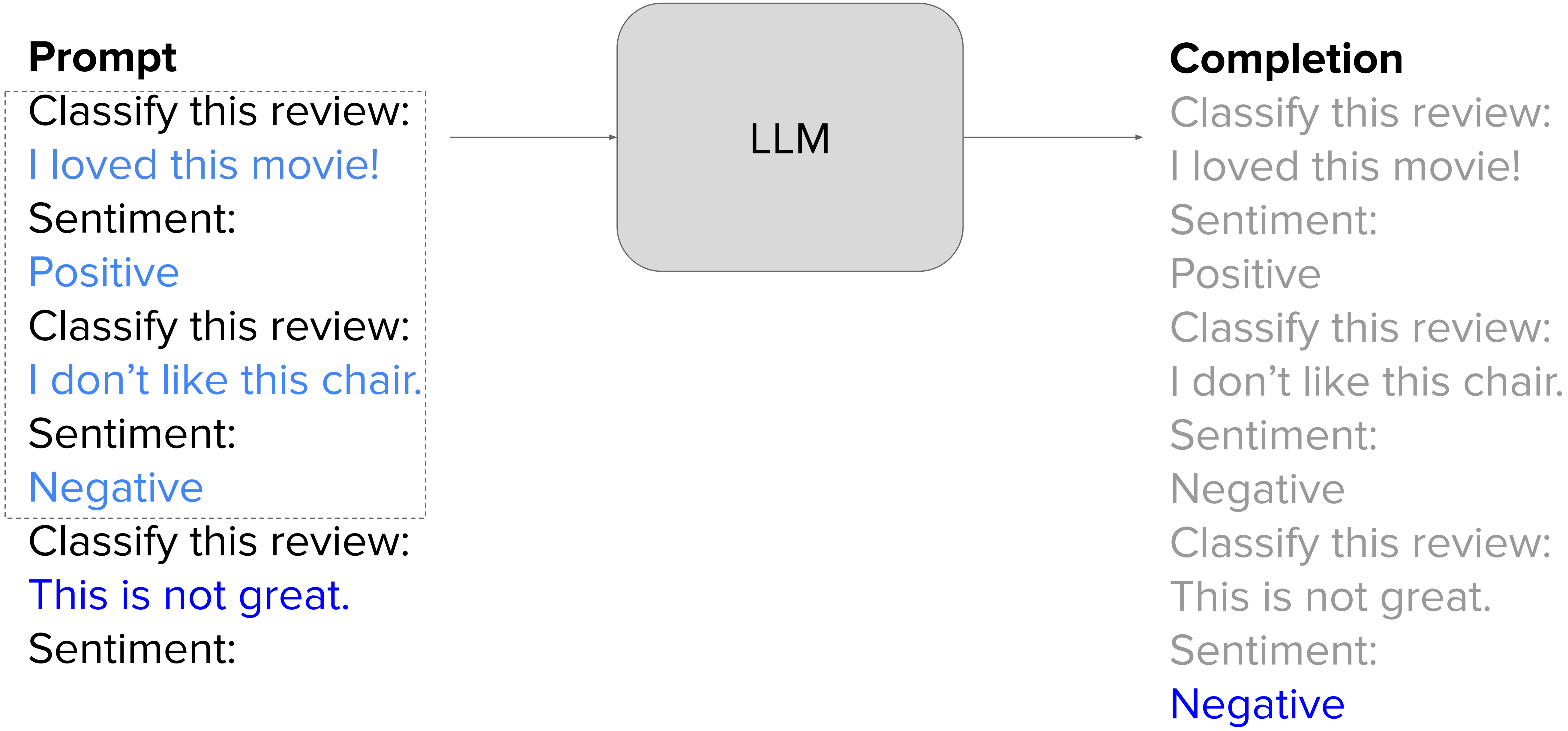
Classify this review:
I loved this movie!
Sentiment:
Positive
Classify this review:
I don’t like this chair.
Sentiment:

LLM

Completion

Classify this review:
I loved this movie!
Sentiment:
Positive
Classify this review:
I don’t like this chair.
Sentiment:
Negative

ICL - few shot inference



Prompt

Classify this review:
I loved this movie!
Sentiment:
Positive
Classify this review:
I don’t like this chair.
Sentiment:
Negative
Classify this review:
This is not great.
Sentiment:

Completion

Classify this review:
I loved this movie!
Sentiment:
Positive
Classify this review:
I don’t like this chair.
Sentiment:
Negative
Classify this review:
This is not great.
Sentiment:
Negative

OpenAI and PHP

- In PHP we can use the openai-php/client open source project (MIT license) to use the **GPT models**
- This is a community project driven by Nuno Maduro and Sandro Gehri
- It performs HTTP call using the OpenAI API
- You need to have an API key from OpenAI

Example: generate text

```php
$client = OpenAI::client(getenv('OPENAI_API_KEY'));

$result = $client->chat()->create([
   'model' => 'gpt-3.5-turbo',
   'messages' => [
     ['role' => 'system', 'content' => 'You are an helpful assistant.'],
     ['role' => 'user', 'content' => 'What is the capital of Italy?'],
   ],
]);
// Answer: The capital city of Italy is Rome
printf("Answer: %s\n", $result->choices[0]->message->content);
```

More information text generation guide

Example: generate image

```php
$client = OpenAI::client(getenv('OPENAI_API_KEY'));

$response = $client->images()->create([
   'model' => 'dall-e-3',
   'prompt' => 'A busy developer working on a laptop during a conference',
   'n' => 1,
   'size' => '1024x1024',
   'response_format' => 'b64_json',
]);

file_put_contents("image.png", base64_decode($response->data[0]->b64_json));
```

More information [image generation guide]

Example: text to speech

```php
$client = OpenAI::client(getenv('OPENAI_API_KEY'));

$response = $client->audio()->speech([
    'model' => 'tts-1',
    'input' => 'Good morning PHP developers',
    'voice' => 'onyx',
    'speed' => 0.95
]);

file_put_contents('audio.mp3', $response);
```

More information [text-to-speech guide]()

Ollama

- Ollama is a software for downloading and running (open source) LLMs
- Llama 2/3, Phi 3, Mistral, Gemma, and other models
- Simple interface:
  - ollama pull llama3
  - ollama run llama3

LLPhant

- LLPhant is a comprehensive open source Generative AI framework for PHP
- The goal is to offer an easy to use library to build GenAI applications in PHP
- LLM supported: OpenAI, Ollama, Mistral
- Vector databases: Elasticsearch, File, Memory, Milvus, Qdrant, Redis
- Started by Maxime Thoonsen and sponsored by Theodo

Example: LLPhant with Llama3

```php
use LLPhant\Chat\OllamaChat;
use LLPhant\OllamaConfig;

$config = new OllamaConfig();
$config->model = 'llama3';
$chat = new OllamaChat($config);

$response = $chat->generateText('What is the capital of Italy?');
// The capital city of Italy is Rome
printf("%s\n", $response);
```

Retrieval-Augmented Generation (RAG)

- **RAG** is a technique in natural language processing that combines information retrieval systems with **Large Language Models** (LLM) to generate more informed and accurate responses
- It is composed by the following parts:
  - **Retrieval-Augmented**
  - **Generation**

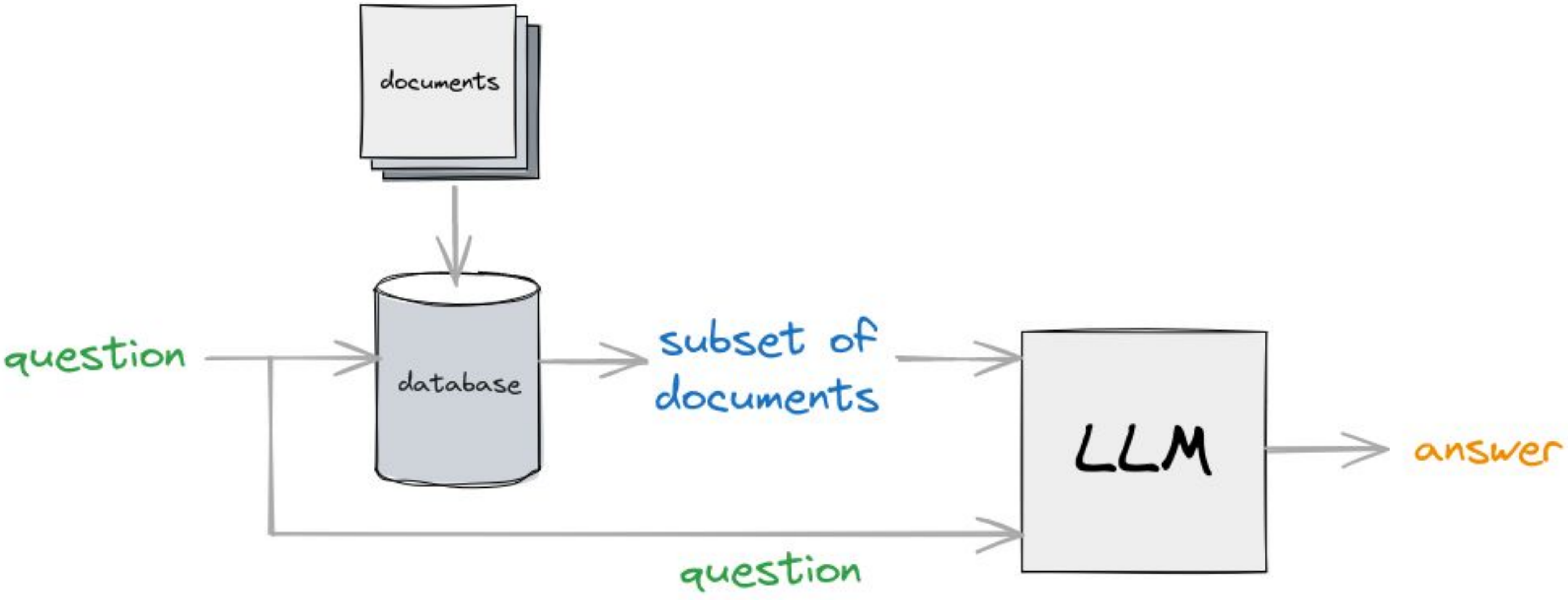
Generation

- LLMs are very powerful but have some limitations:
  - **No source** (potential hallucinations)
    - How can I verify the information coming from an LLM?
    - What sources has been used to generate the answer?
  - **Out of date**
    - An LLM is trained in a period of time
    - For update we need to retraining the model (very expensive)

Retrieval-Augmented

- We collect sets of private or public document
- We build a **retrieval system** (e.g. a database) to extract a subset of documents using a **question**
- Then we pass the **question + documents found** to an LLM as prompt with a context
- The LLM can give an answer using the updated documents

RAG architecture



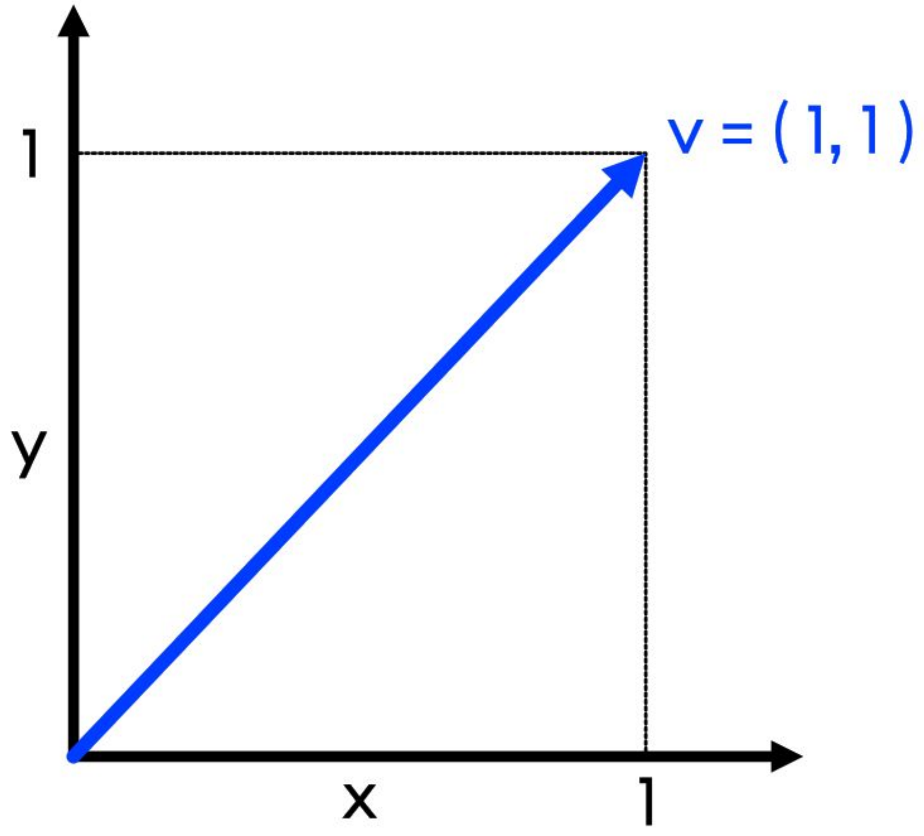
elastic

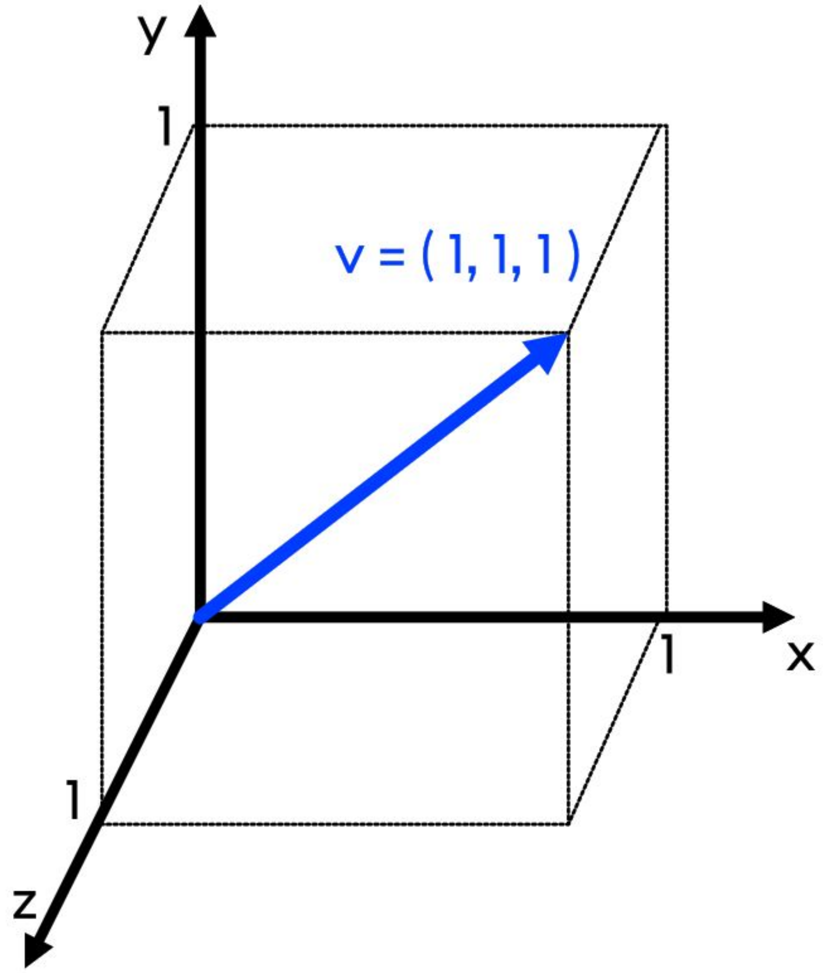
Retrieve documents from a question

- How we can retrieve documents in a database using a question?
- We need to use **semantic search**
- One solution is to use a **vector database** (eg. Elasticsearch)
- A vector database is a system that uses **vectors** (set of numbers) to retrieve information

What is a vector?

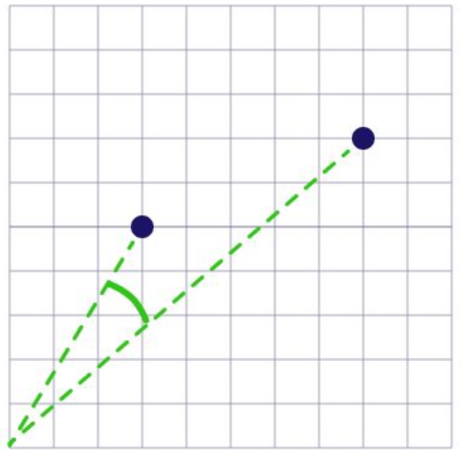
- A vector is a set of numbers
- Example: a vector of 3 elements [2, 5, -10]
- A vector can be represented in a multi-dimensional space (eg. GPT uses 1536 dimensions)





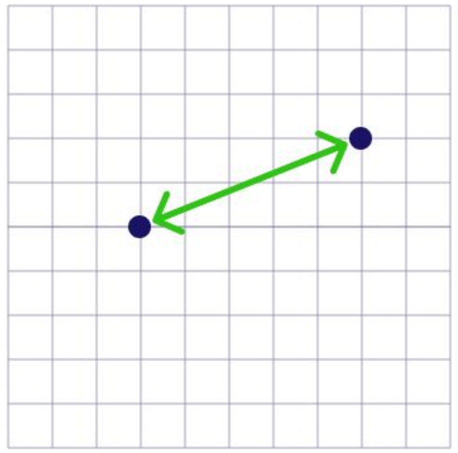
Similarity between two vectors

- Two vectors are (semantically) similar if they are close to each other
- We need to define a way to measure the similarity



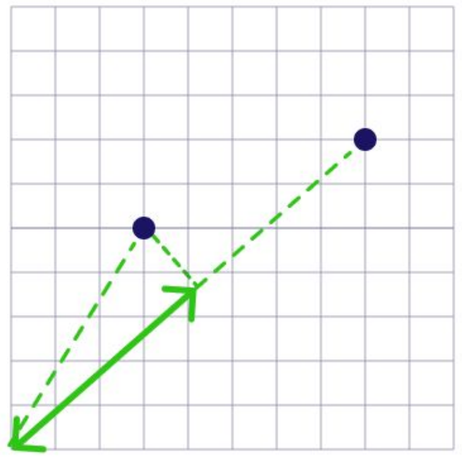
Cosine Distance

$$1 - \frac{A \cdot B}{||A|| \quad ||B||}$$



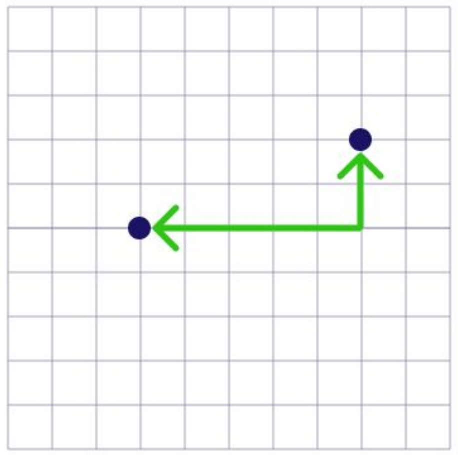
Squared Euclidean (L2 Squared)

$$\sum_{i=1}^{n} (x_i - y_i)^2$$



Dot Product

$$A \cdot B = \sum_{i=1}^{n} A_i B_i$$

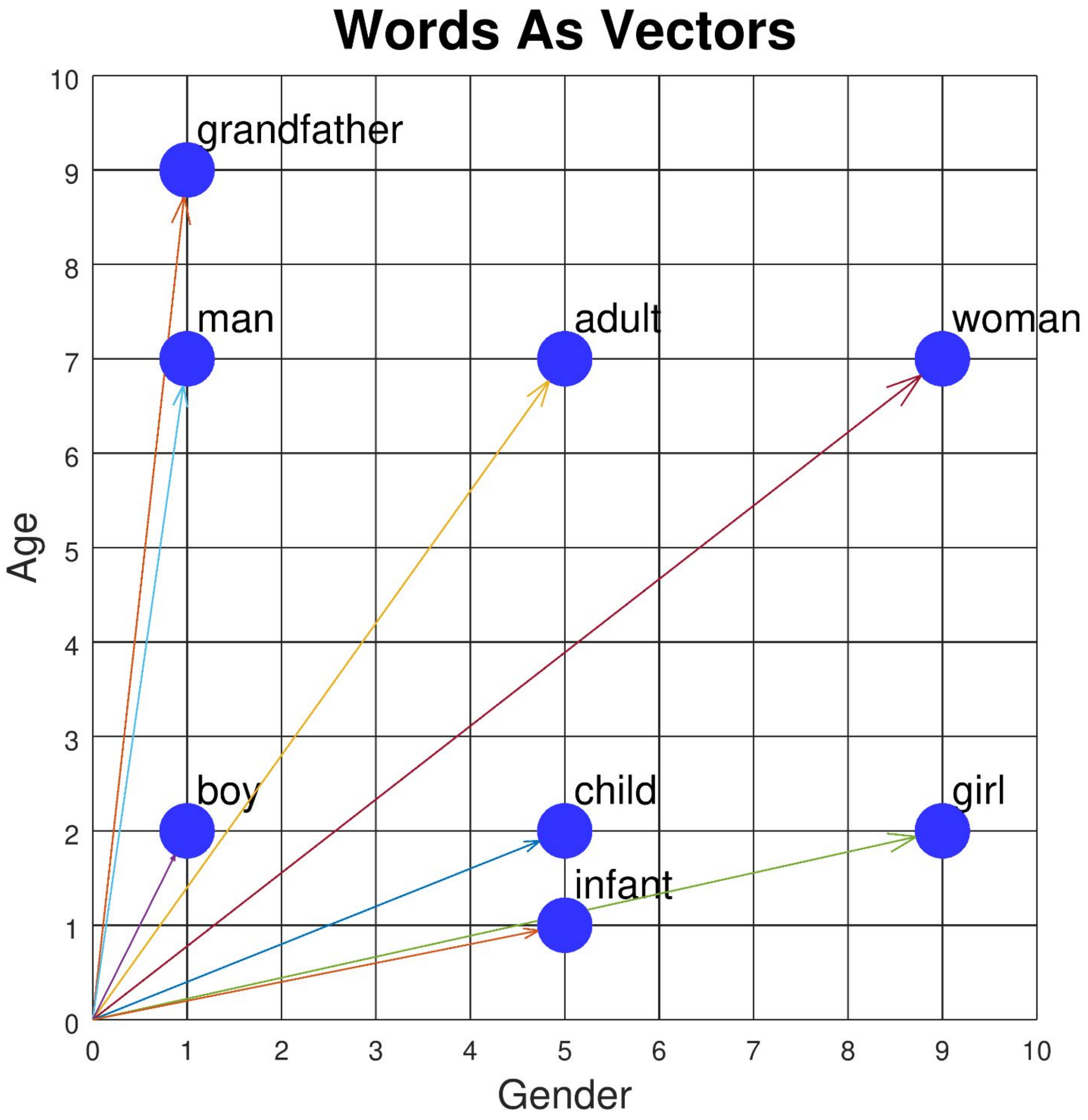


Manhattan (L1)

$$\sum_{i=1}^{n} |x_i - y_i|$$

Embedding

- Embedding is the translation of an input (document, image, sound, movie, etc) to a vector
- There are many techniques, using an LLM typically this is done by a neural network
- The goal is to group information that are semantically related to each other



elastic

Vector database + LLM

- The search query (***question***) is in natural language
- We use semantic search to retrieve top-n relevant documents (***context***)
- We send the following prompt to the LLM:
    - *Given the following* ***{context}*** *answer to the following* ***{question}***

Split the documents in chunk

- We need to store data in the vector database using chunk of information
- We cannot use big documents since we need to pass it in the context part of the prompt for an LLM that typically has a token limit (e.g. gpt-3.5-turbo up to 16k)
- We need to split the documents in chunk (eg. number of words)

RAG demo:

OpenAI + Elasticsearch + LLPhant

Available on github: ezimuel/php-llm-examples

elastic

References

- What is retrieval-augmented generation? IBM research
- GPT-4 Technical Report, OpenAI
- Nathan Benaich, State of AI Report 2023, Air Street Capital
- Valentina Alto, Modern Generative AI with ChatGPT and OpenAI Models, Packt, 2023
- Ashish Vaswan et al., Attention Is All You Need, Proceedings of 31st Conference on Neural Information Processing Systems (NIPS 2017)
- Will Oremus, Google's AI passed a famous test — and showed how the test is broken, Washington Post, June 17, 2022
- Scott Mayer McKinney et al., International evaluation of an AI system for breast cancer screening, Nature, Vol. 577, 2 January 2020
- Albert Ziegler, John Berryman, A developer's guide to prompt engineering and LLMs, Github blog post
- Saurabh Mhatre, What Is The Relation Between Artificial And Biological Neuron?

Thanks!

More information: www.elastic.co

elastic