Transformers, ChatGPT, Prompt Engineering

John Tan Chong Min

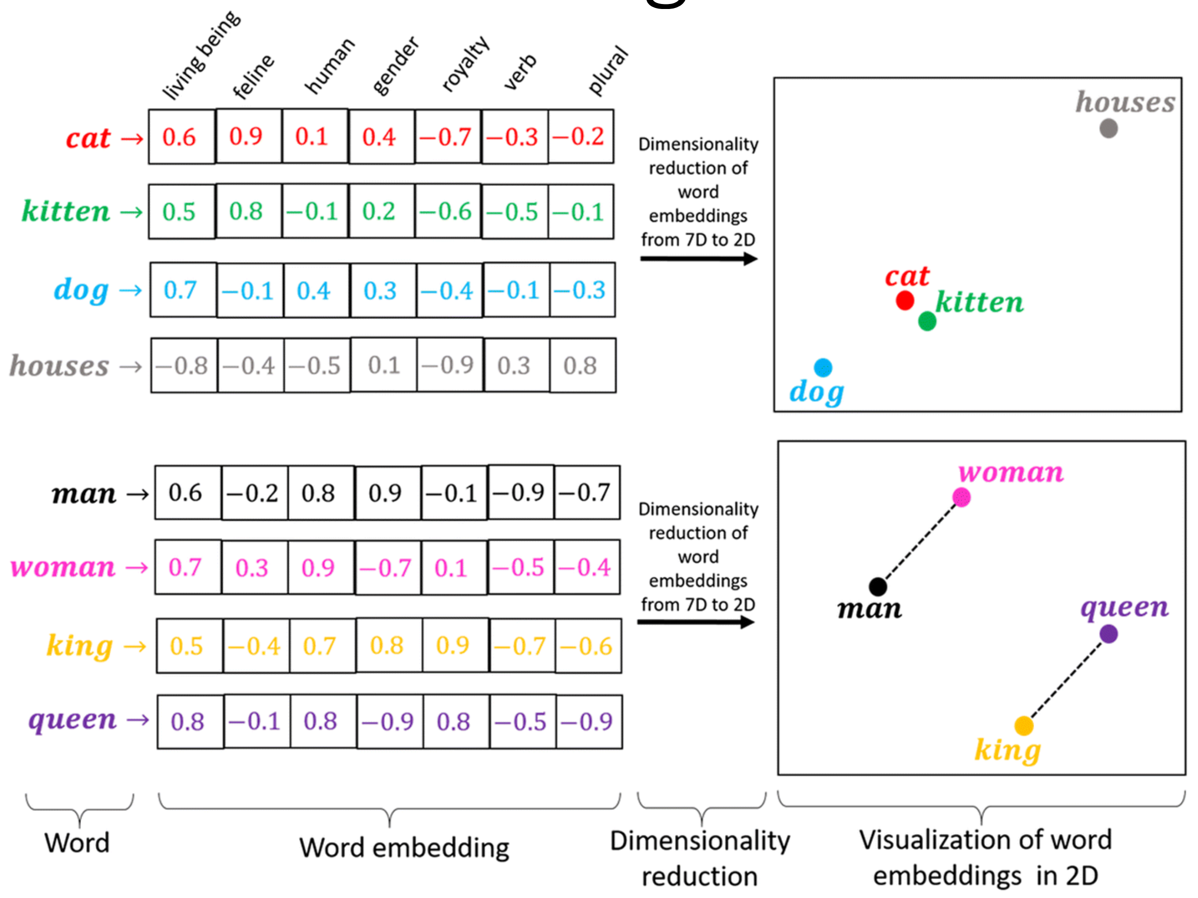
Transformers

Attention Is All You Need. 2017. Vaswani et al.

Embedding Space

Semantic Meaning

Word Embeddings



	living being	feline	human	gender	royalty	verb	plural
cat →	0.6	0.9	0.1	0.4	−0.7	−0.3	−0.2
kitten →	0.5	0.8	−0.1	0.2	−0.6	−0.5	−0.1
dog →	0.7	−0.1	0.4	0.3	−0.4	−0.1	−0.3
houses →	−0.8	−0.4	−0.5	0.1	−0.9	0.3	0.8

man →	0.6	−0.2	0.8	0.9	−0.1	−0.9	−0.7
woman →	0.7	0.3	0.9	−0.7	0.1	−0.5	−0.4
king →	0.5	−0.4	0.7	0.8	0.9	−0.7	−0.6
queen →	0.8	−0.1	0.8	−0.9	0.8	−0.5	−0.9

Dimensionality reduction of word embeddings from 7D to 2D
houses
cat
kitten
dog
Dimensionality reduction of word embeddings from 7D to 2D
woman
man
queen
king
Word
Word embedding
Dimensionality reduction
Visualization of word embeddings in 2D

- Extracting semantic meaning in higher-dimensional space
- Number of dimensions depends on use case

Taken from: https://medium.com/@hari4om/word-embedding-d816f643140

Transformer Architecture

Summarized with illustrations from

https://jalammar.github.io/illustrated-transformer/

Transformers: Overall

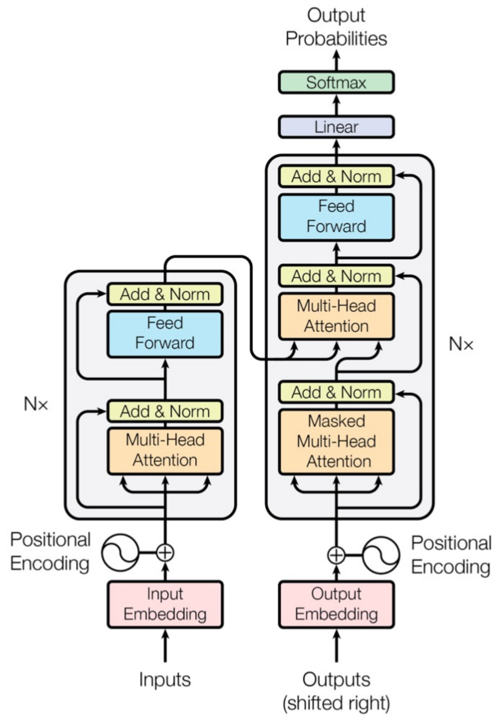


Figure 1: The Transformer - model architecture.

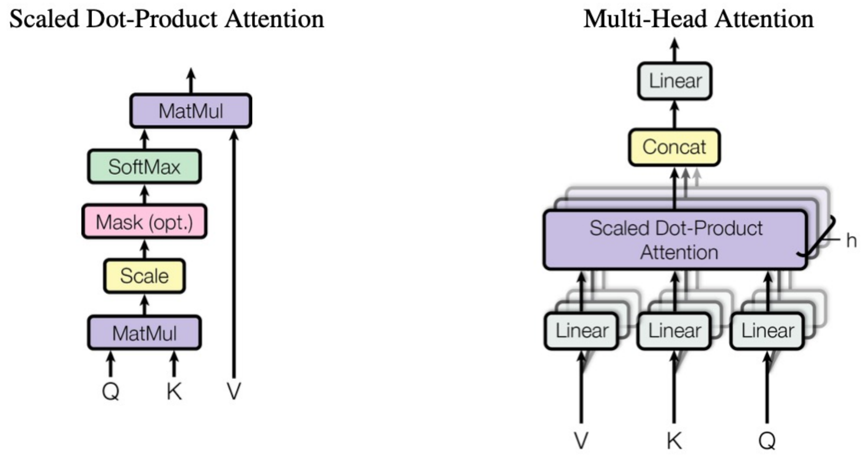


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

$$\mathrm{Attention}(Q,K,V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Taken from: Attention is all you need. Vaswani et al. (2017)

Encoder-Decoder link

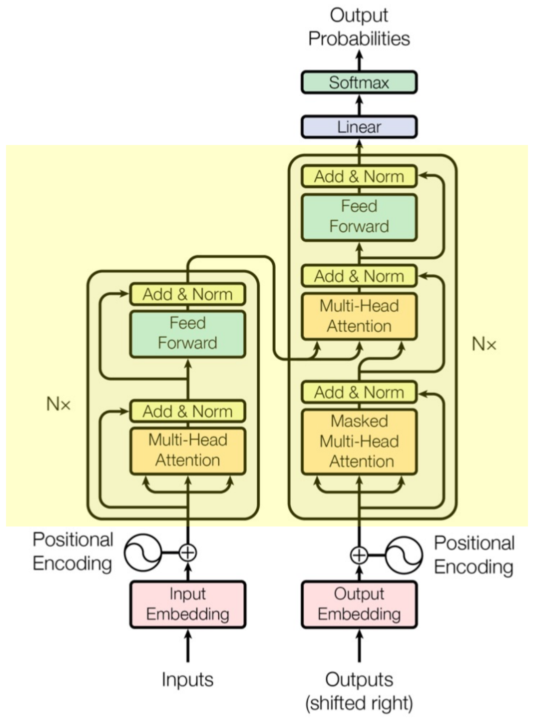
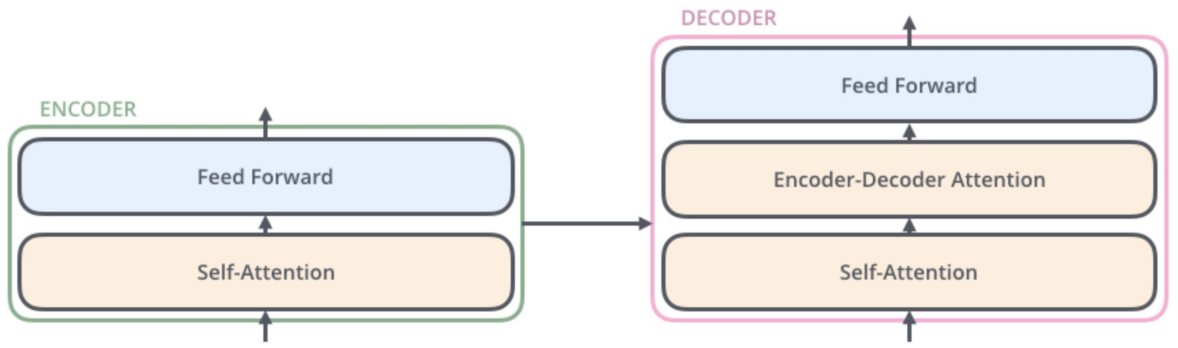


Figure 1: The Transformer - model architecture.

Self-attention in the Encoder & Decoder block,
Then Encoder-Decoder Attention



https://jalammar.github.io/illustrated-transformer/

Self-Attention block

Converting input dimension to embedding dimensions

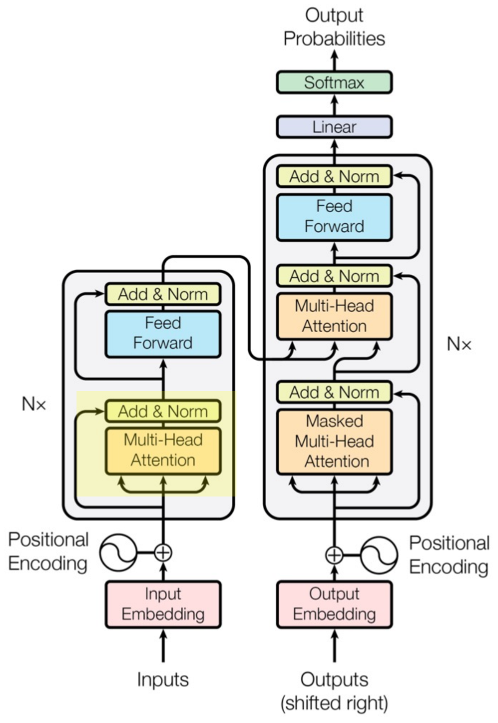
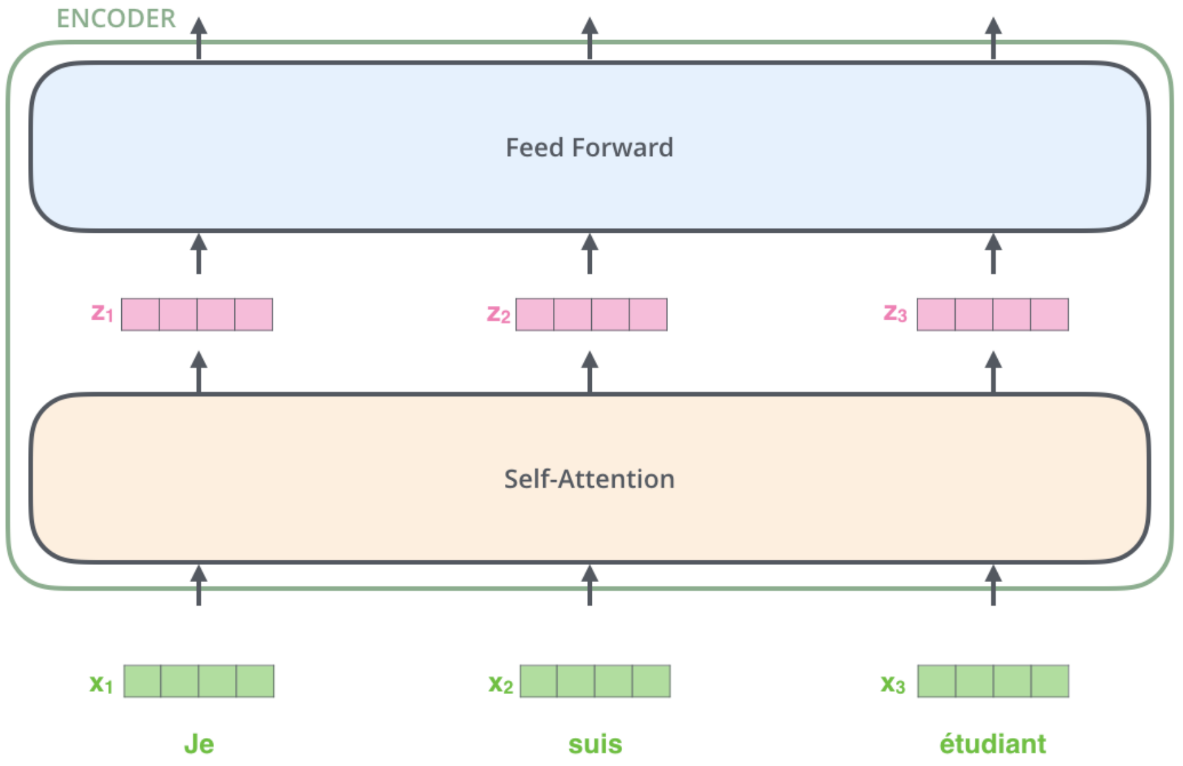


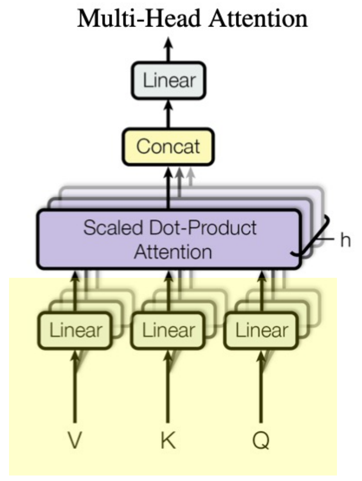
Figure 1: The Transformer - model architecture.

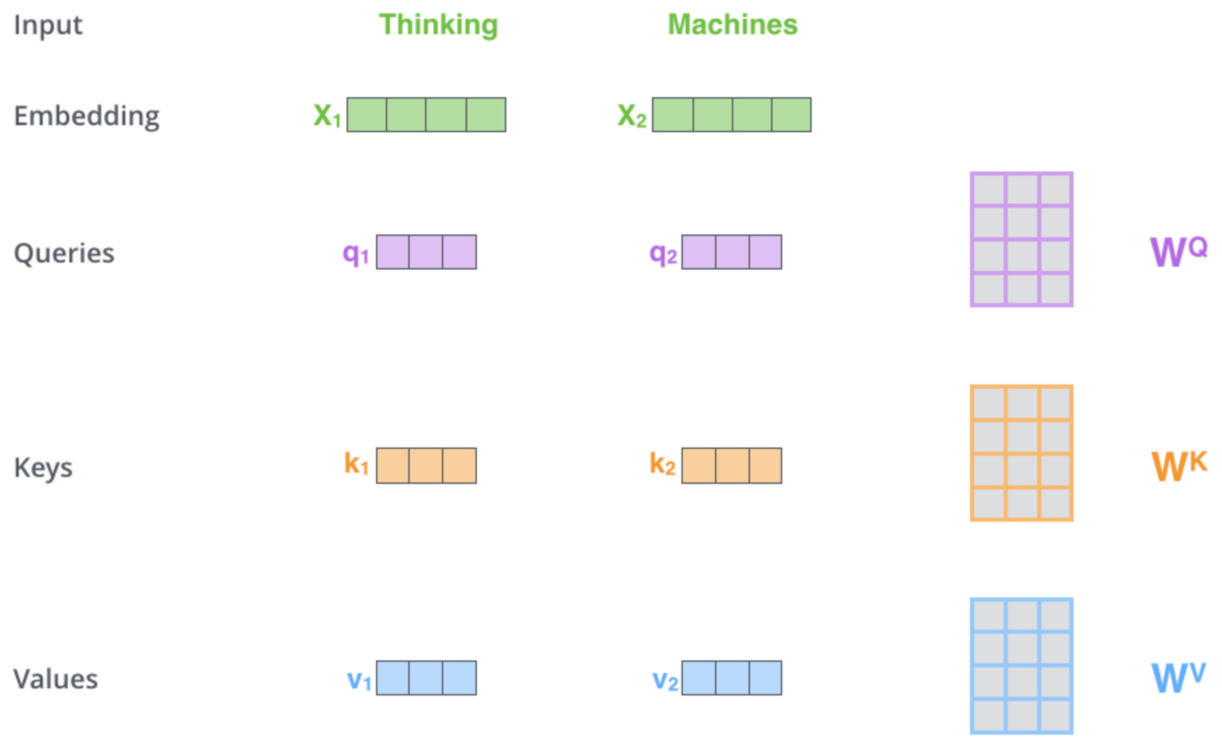


https://jalammar.github.io/illustrated-transformer/

Generate Embedding Space

Convert Q, K, V to embedding space dimensions



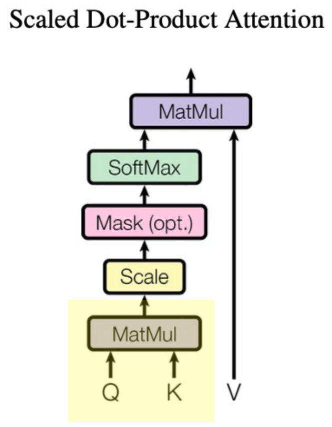


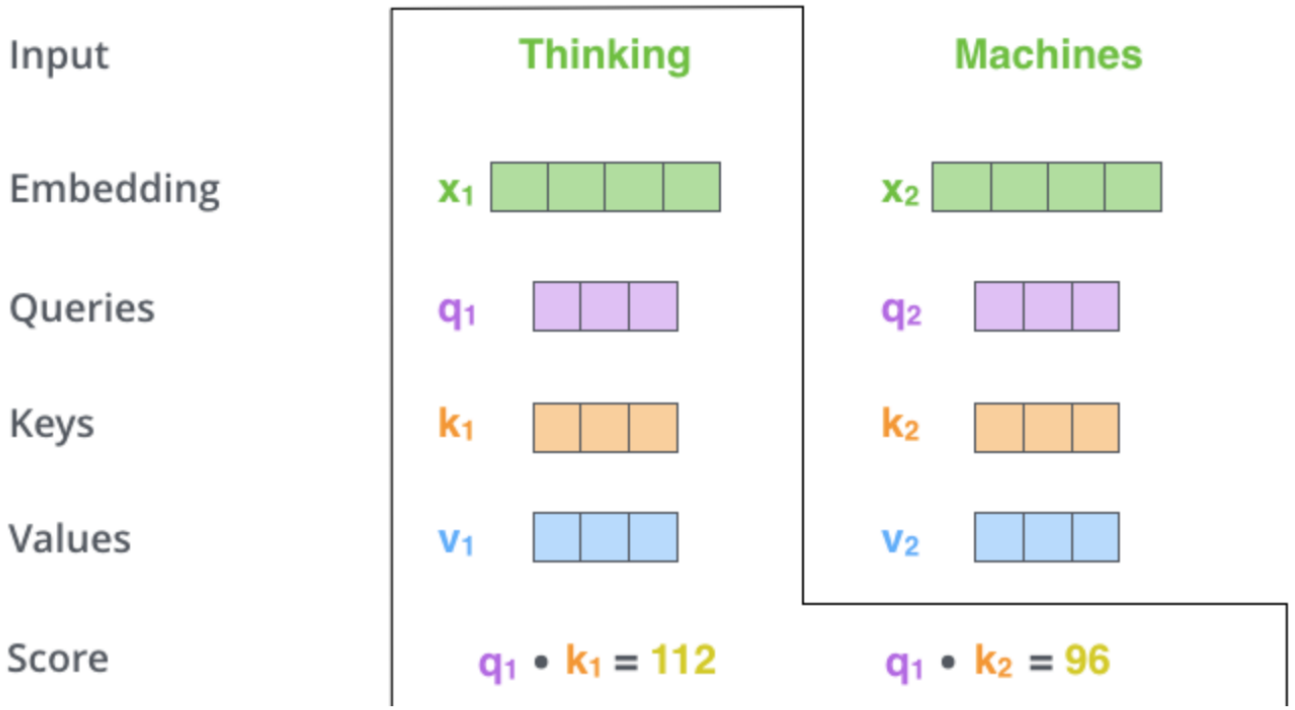
$$\mathrm{Attention}(Q,K,V)=\mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

https://jalammar.github.io/illustrated-transformer/

Dot Product between Q and K

Dot product to measure similarity between embedding vectors of Q and K



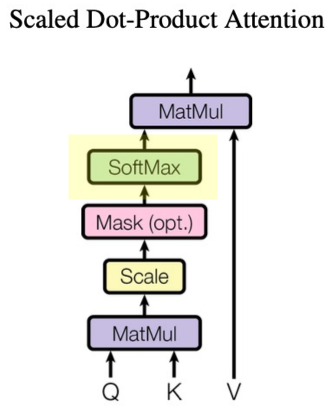


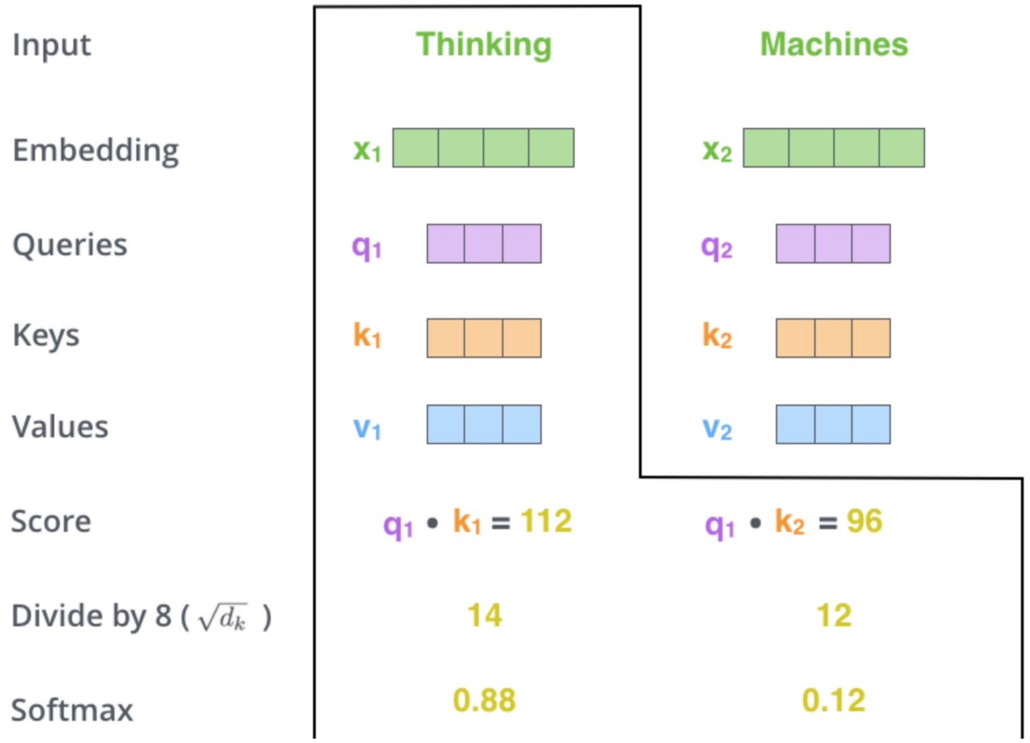
$$\mathrm{Attention}(Q,K,V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

https://jalammar.github.io/illustrated-transformer/

Softmax the dot product

Softmax to maintain overall magnitude scale of value vectors (softmax sums to 1)





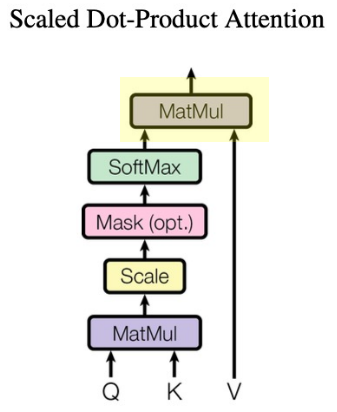
Division to smooth out softmax

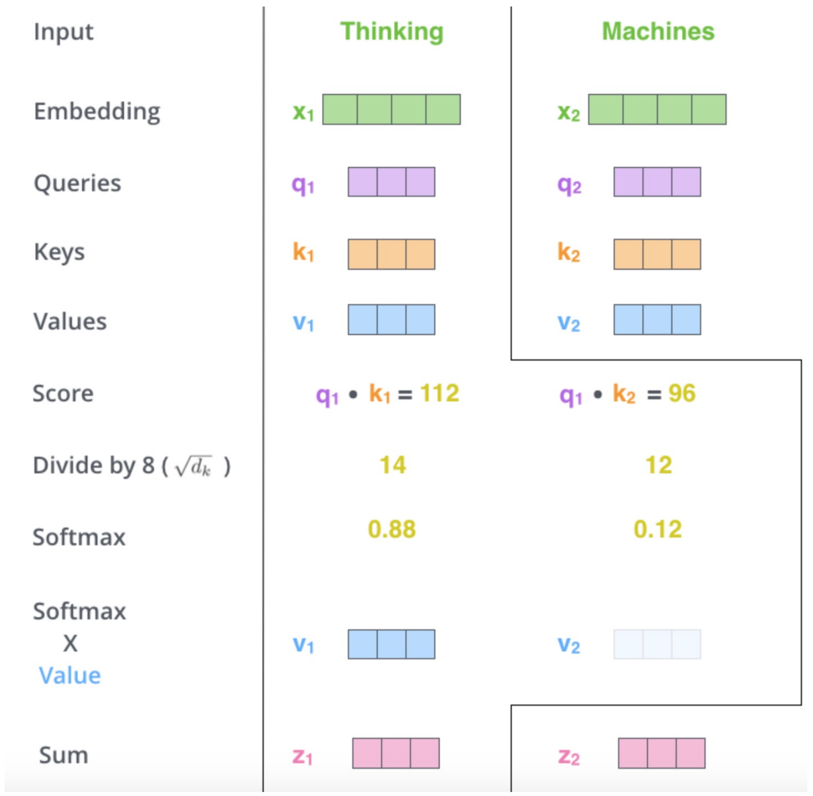
$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

https://jalammar.github.io/illustrated-transformer/

Sum up the values weighted by softmax

Softmax to maintain overall magnitude scale of value vectors (softmax sums to 1)

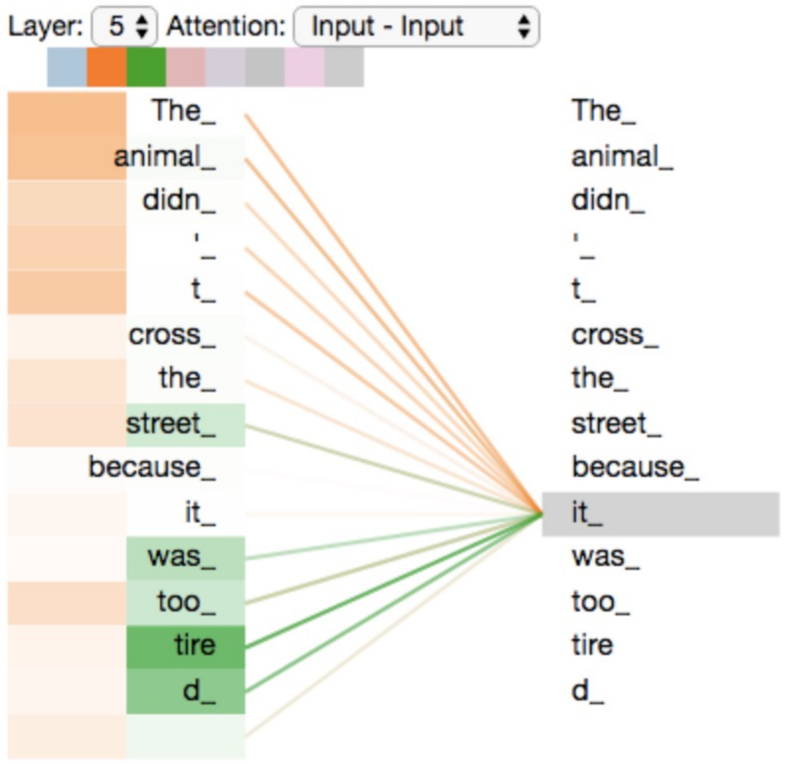




$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

https://jalammar.github.io/illustrated-transformer/

Visualizing attention heads



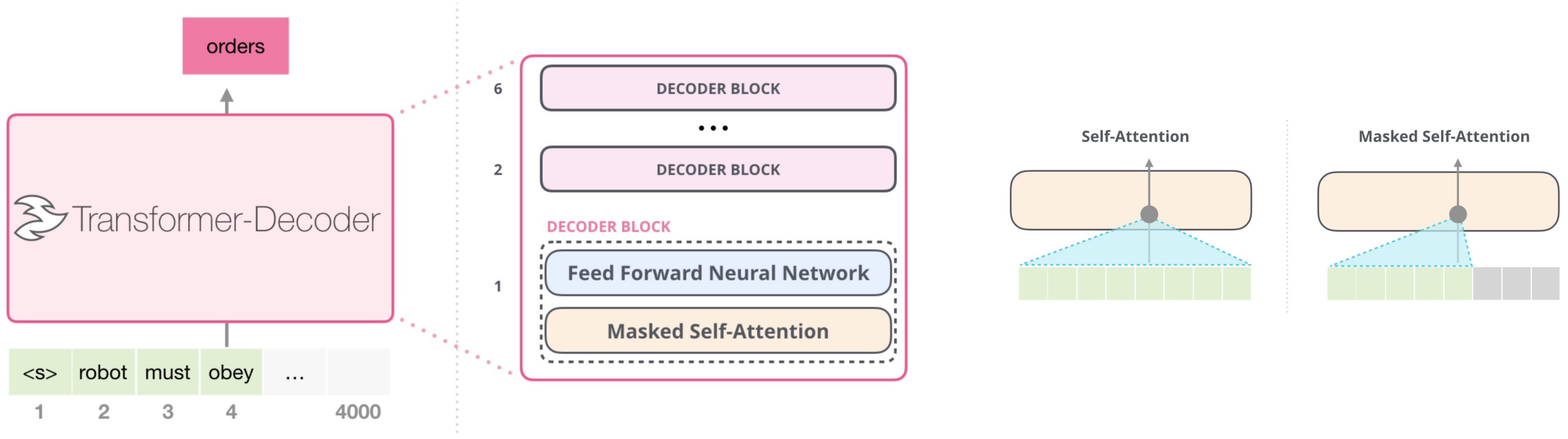
https://jalammar.github.io/illustrated-transformer/

GPT

Improving language understanding by generative pre-training. 2018. Radford et al.

(GPT paper)

GPT is decoder-only, with masked attention



- Replace encoder memory with context! Fewer weights to learn.

https://jalammar.github.io/illustrated-gpt2/

Training

- Unsupervised Pre-training
  - Given a series of earlier words, predict the next one
  - Can be done on any web data without labelling!
  - Example: The cat is on the mat
    - The <masked> : predict cat
    - The cat is <masked> : predict on
    - The cat is on the <masked> : predict mat

- Fine-tuning
  - Given a prompt, predict the sequence of tokens to match the response
  - Done using well-labelled prompt-response datasets

Next token loss

Given an unsupervised corpus of tokens $\mathcal{U} = \{u_1, \ldots, u_n\}$, we use a standard language modeling objective to maximize the following likelihood:

$$L_1(\mathcal{U}) = \sum_i \log P(u_i | u_{i-k}, \ldots, u_{i-1}; \Theta) \quad (1)$$

where k is the size of the context window, and the conditional probability P is modeled using a neural network with parameters Θ. These parameters are trained using stochastic gradient descent [51].

- For supervised loss involving an entire output prompt, it is simply the summation of the log probability of each individual output token given all previous tokens
  - E.g. Q: What is twenty plus two? A: Twenty two
  - Next token loss = log P(Twenty| .) + log P(two| .)

Next-token generation

- Generates the next token based on sampling the probability distribution at the output layer

- Example:
  - **Input: Thinking machines**
  - Probability of next tokens: are (80%), were (20%)
  - Output: Thinking machines are

  - **Input: Thinking machines are**
  - Probability of next tokens: useful (80%), good (20%)
  - Output: Thinking machines are useful

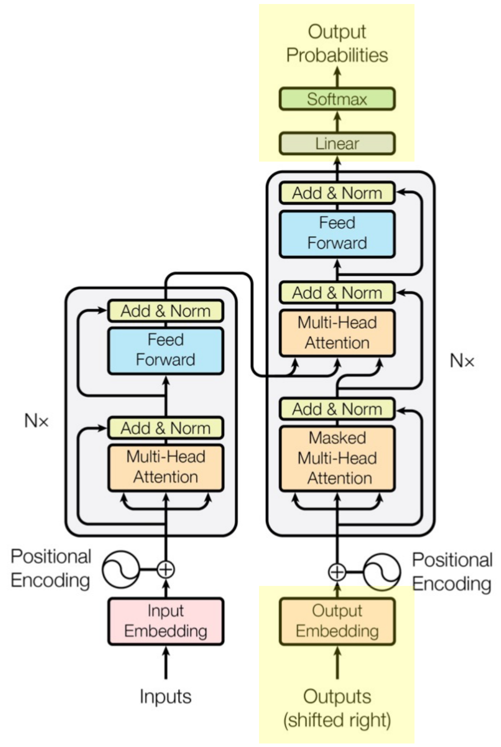


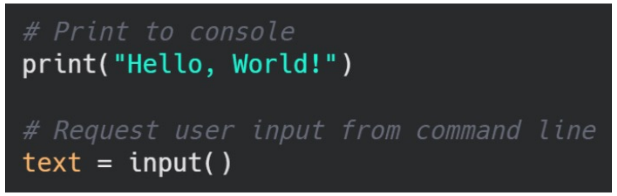
Figure 1: The Transformer - model architecture.

Prompt Engineering Tips

LLMs vs Computer Programs (Part 1)

ChatGPT

- Customizable with zero-shot / few-shot prompting out of the box
- Performance may not be replicable
- Can do intent processing well, even for out of distribution cases



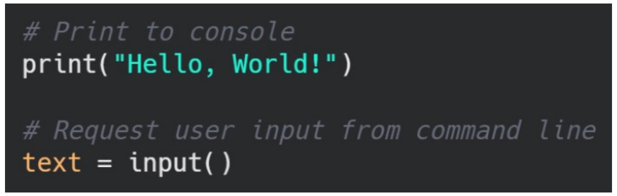
```python
# Print to console
print("Hello, World!")

# Request user input from command line
text = input()
```

- Needs to be programmed extensively to perform a task
- Performance replicable
- Intent processing only based on what is programmed in

LLMs vs Computer Programs (Part 2)

ChatGPT

- Inference time lengthy due to need for recursive token generation
- Costs money per use
- Highly flexible and can accept multiple input formats in multiple languages



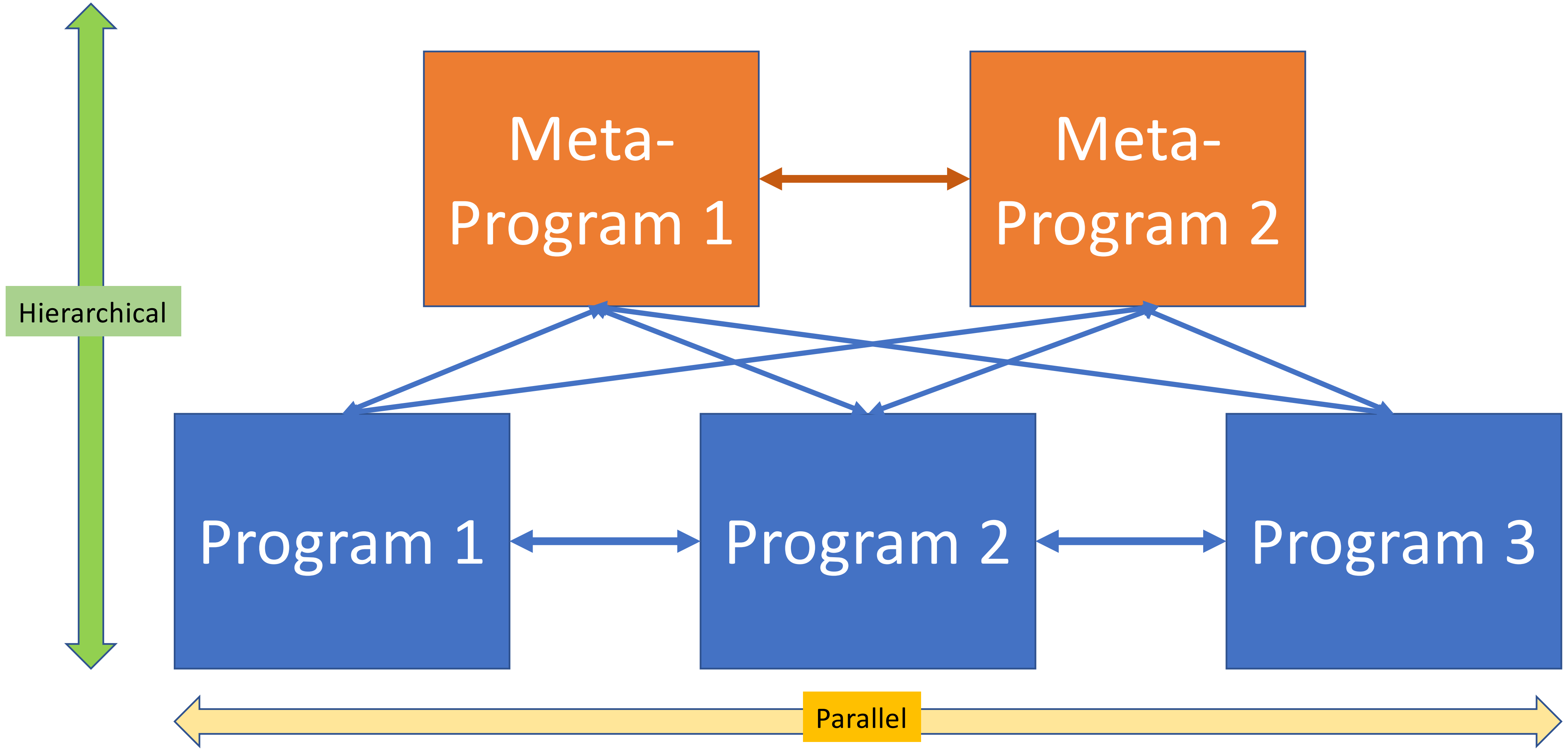
```python
# Print to console
print("Hello, World!")

# Request user input from command line
text = input()
```

- Relatively fast inference time depending on use case
- Free to use
- Input malleability limited to what is programmed in

What is the key benefit of LLMs

- Flexible
- Can perform most tasks without any pre-training / programming
- If you are specific enough, can get quite reliable and consistent outputs

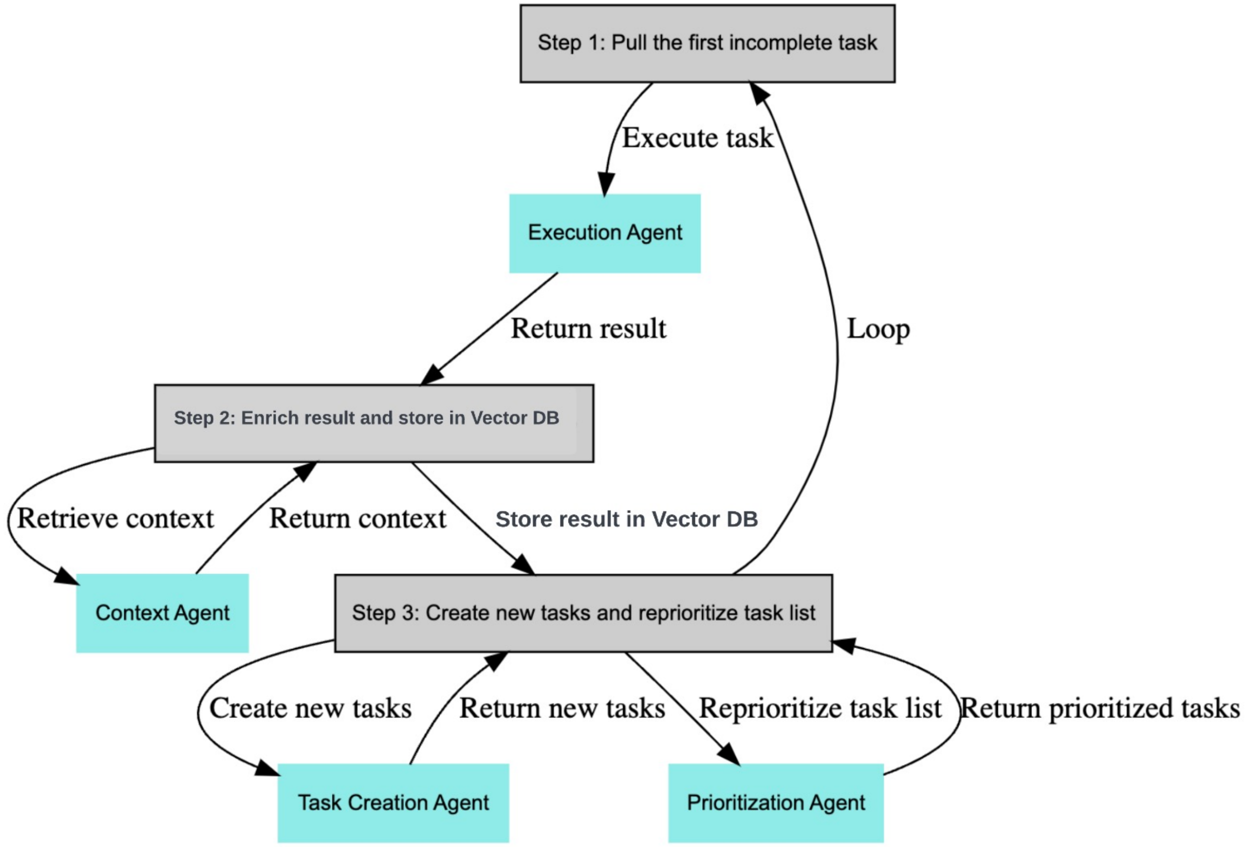
Systems-level approach to LLMs



BabyAGI

- Uses GPT4 to in a larger ecosystem to:
    - Create Tasks
    - Execute Tasks
    - Prioritize Tasks

- Uses memory to store and retrieve task/result pairs
    - Stored by vector embeddings



https://github.com/yoheinakajima/babyagi

Zero-shot prompting

- Only applicable if natural language can already describe the categories / use case

- Simply describe the categories / use case in natural text

Zero-shot prompting

- Example:
  - You are a classification machine. You are to classify the context of each sentence. The various contexts are given as {Context Letter}: {Description}:
    - A: On a mountain
    - B: In the classroom
    - C: In the garden
  - Classify the following and give the answer as {Number}:{Context Letter} for each line. Only provide the context letter without the description:
    - 1. Why, what a steep slope
    - 2. I can’t find my eraser!
    - 3. Those plants are not going to be watering themselves, are they?

Few-shot prompting

- You are a number classifier. The following are examples of classifications:

    - Input 1: 3
    - Output 1: Odd

    - Input 2: 4
    - Ouput 2: Even

- What is the classification of the number 6?

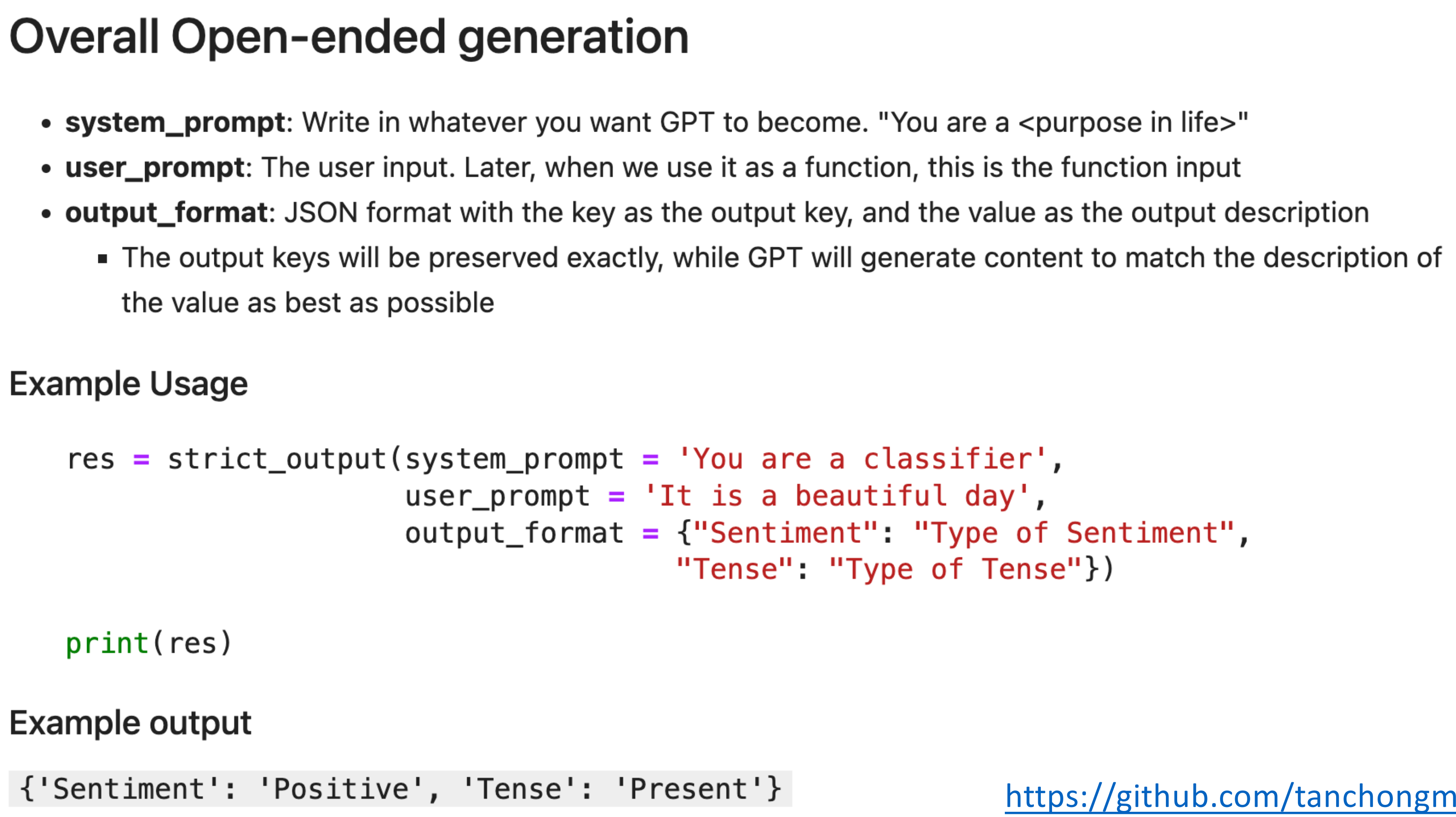
How to program intent?

- You are <purpose in life>
- Example format (optional):
  - Input: <Input>
  - Output: <Output>
- Input: <Your input>

ChatGPT talks too much, what can I do?

- Give it a format to output information in
- JSON framework provides a good template

Fixed JSON Framework



Overall Open-ended generation

- **system_prompt**: Write in whatever you want GPT to become. "You are a <purpose in life>"
- **user_prompt**: The user input. Later, when we use it as a function, this is the function input
- **output_format**: JSON format with the key as the output key, and the value as the output description
  - The output keys will be preserved exactly, while GPT will generate content to match the description of the value as best as possible

Example Usage

```
res = strict_output(system_prompt = 'You are a classifier',
                    user_prompt = 'It is a beautiful day',
                    output_format = {"Sentiment": "Type of Sentiment",
                                    "Tense": "Type of Tense"})

print(res)
```

Example output

```
{'Sentiment': 'Positive', 'Tense': 'Present'}
```

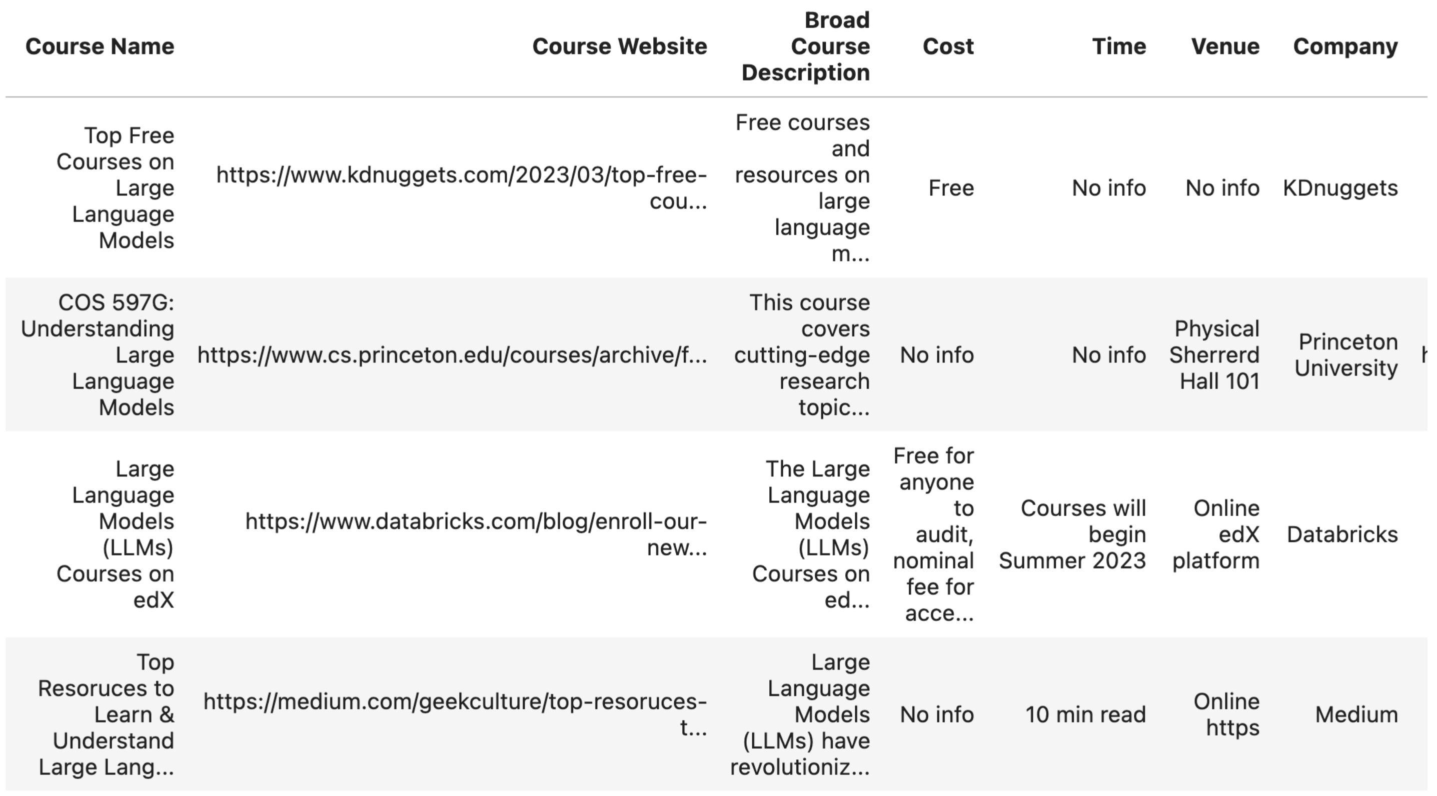
https://github.com/tanchongmin/strictjson

Projects

Possible Projects

- ChatBot using Retrieval-Augmented Generation (ask some questions to a document, website, products etc.)
- Game using ChatGPT API + Stable Diffusion/DALL-E API
- Website Crawler using ChatGPT API + Selenium
- Your other projects!

Web Scraper!

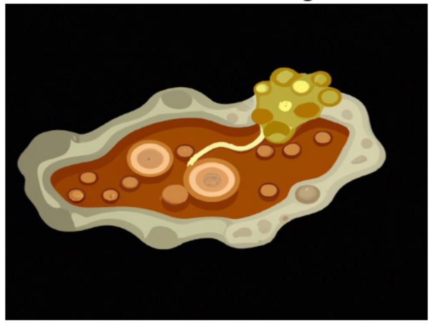
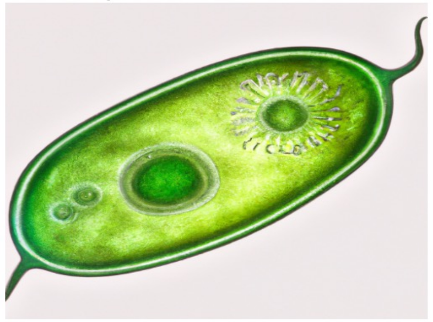


Course Name	Course Website	Broad Course Description	Cost	Time	Venue	Company
Top Free Courses on Large Language Models	https://www.kdnuggets.com/2023/03/top-free-cou...	Free courses and resources on large language m...	Free	No info	No info	KDnuggets
COS 597G: Understanding Large Language Models	https://www.cs.princeton.edu/courses/archive/f...	This course covers cutting-edge research topic...	No info	No info	Physical Sherrerd Hall 101	Princeton University
Large Language Models (LLMs) Courses on edX	https://www.databricks.com/blog/enroll-our-new...	The Large Language Models (LLMs) Courses on ed...	Free for anyone to audit, nominal fee for acce...	Courses will begin Summer 2023	Online edX platform	Databricks
Top Resoruces to Learn & Understand Large Lang...	https://medium.com/geekculture/top-resoruces-t...	Large Language Models (LLMs) have revolutioniz...	No info	10 min read	Online https	Medium

https://github.com/tanchongmin/TensorFlow-Implementations/tree/main/Tutorial

Evolution Game

```
Outcome: You have chosen the attribute Motile. As a result, you gain the ability to move independently, increasing
your chances of finding food and escaping predators. Your simple structure allows you to move using cilia or flagel
la, propelling yourself through the water. With this new attribute, you are now more adaptable and have a higher ch
ance of survival in your aquatic pond habitat.
Species: Protozoa
Attribute: Resilient, Motile
Habitat: Aquatic Pond
Phase: Fight a Creature
Option 1: Creature: Amoeba - A single-celled organism similar to you, but lacks the motility attribute. It moves by
extending pseudopods and engulfs its food through phagocytosis.
Option 2: Creature: Paramecium - Another single-celled organism with motility attribute. It moves using cilia and f
eeds on bacteria and other small organisms.
Option 3: Creature: Euglena - A single-celled organism with both motility and photosynthetic attributes. It moves u
sing a whip-like tail called a flagellum and can produce its own food through photosynthesis.
```

```
Enter your choice
 3
```

https://github.com/tanchongmin/TensorFlow-Implementations/tree/main/Tutorial