AI&MLOps Module 4: Generative AI

Deepak Subramani
Assistant Professor
Dept. of Computational and Data Science
Indian Institute of Science Bengaluru

Outline for Jan 25

- Part 1: Decoder only GPT Model
  - What are GPT-class Generative Large Language Models
  - Data preparation for GPT model training
  - GPT finetuning (Assignment)
- Part 2: LLMs and Interacting with them
  - Commercial and open source LLMs
  - What are the main issues in LLMs to be aware of?
  - Taxonomy of interaction with LLMs
  - Prompting Strategies – ZSL, FSL, CoT, ReACT, DSP
  - Parameter Efficient Fine Tuning (LoRA, QLoRA)

Outline for Feb 01

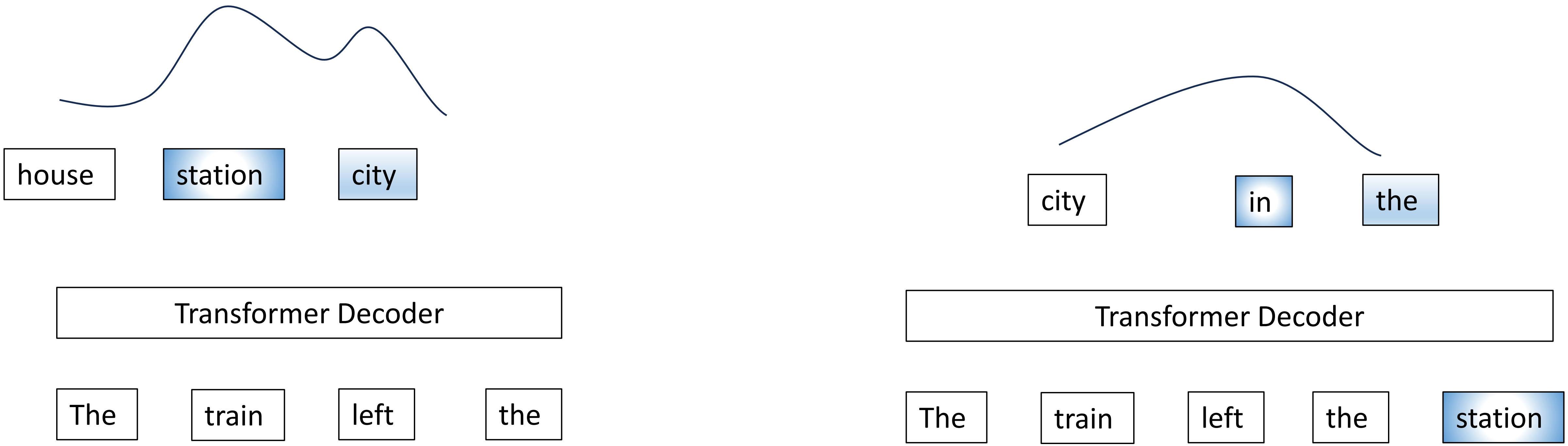
- Part 1:
  - Instruction Tuning
- Part 2:
  - Orchestration
  - Retrieval Augmented Generation
- Part 3:
  - LLM Guardrails
  - LLM Agents

Generative Pretrained Transformers (GPT)

- These are decoder only models.
- Since there is no encoder in this set up, these decoder layers would not have the encoder-decoder attention sublayer that vanilla transformer decoder layers have.
- It only has the masked self attention layer.
- The model predict the next word using massive datasets.

Radford et al., 2018

What does GPT do?



GPT

- 2 step training:
  - Generative pretraining
  - Finetuning with instructions and human feedback
- GPT 1 and GPT 2 Specifics
  - Transformer decoder with 12 blocks, 117M parameters.
  - 512-sequence length, 768-dimensional hidden states, 3072-dimensional feed-forward hidden layers.
  - Trained on BooksCorpus: over 7000 unique books.

GPT

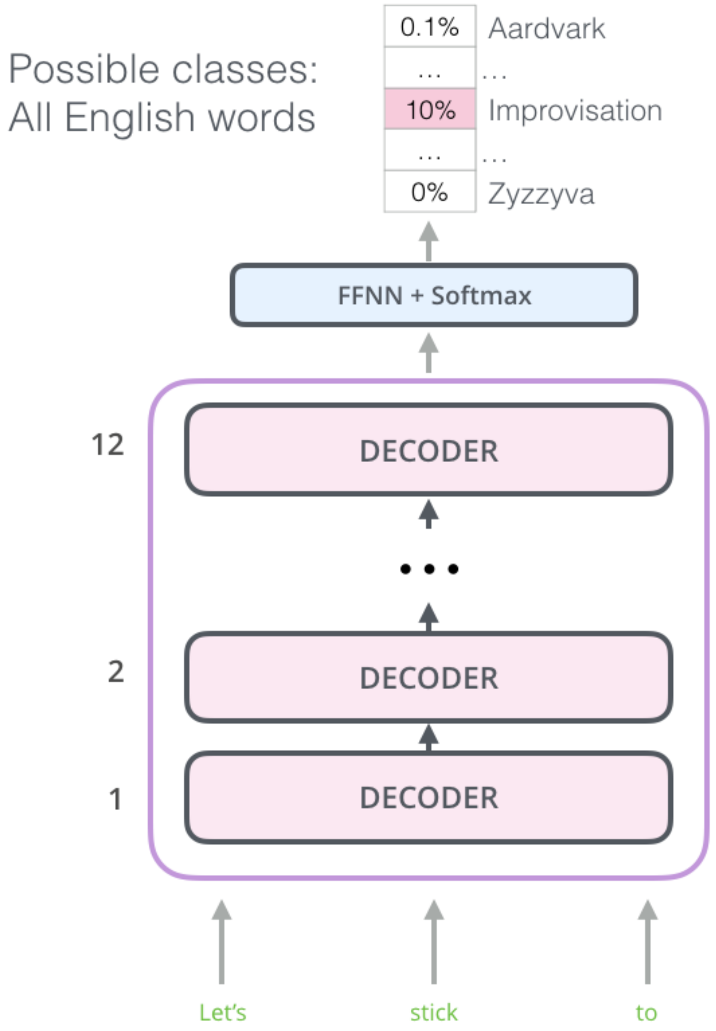
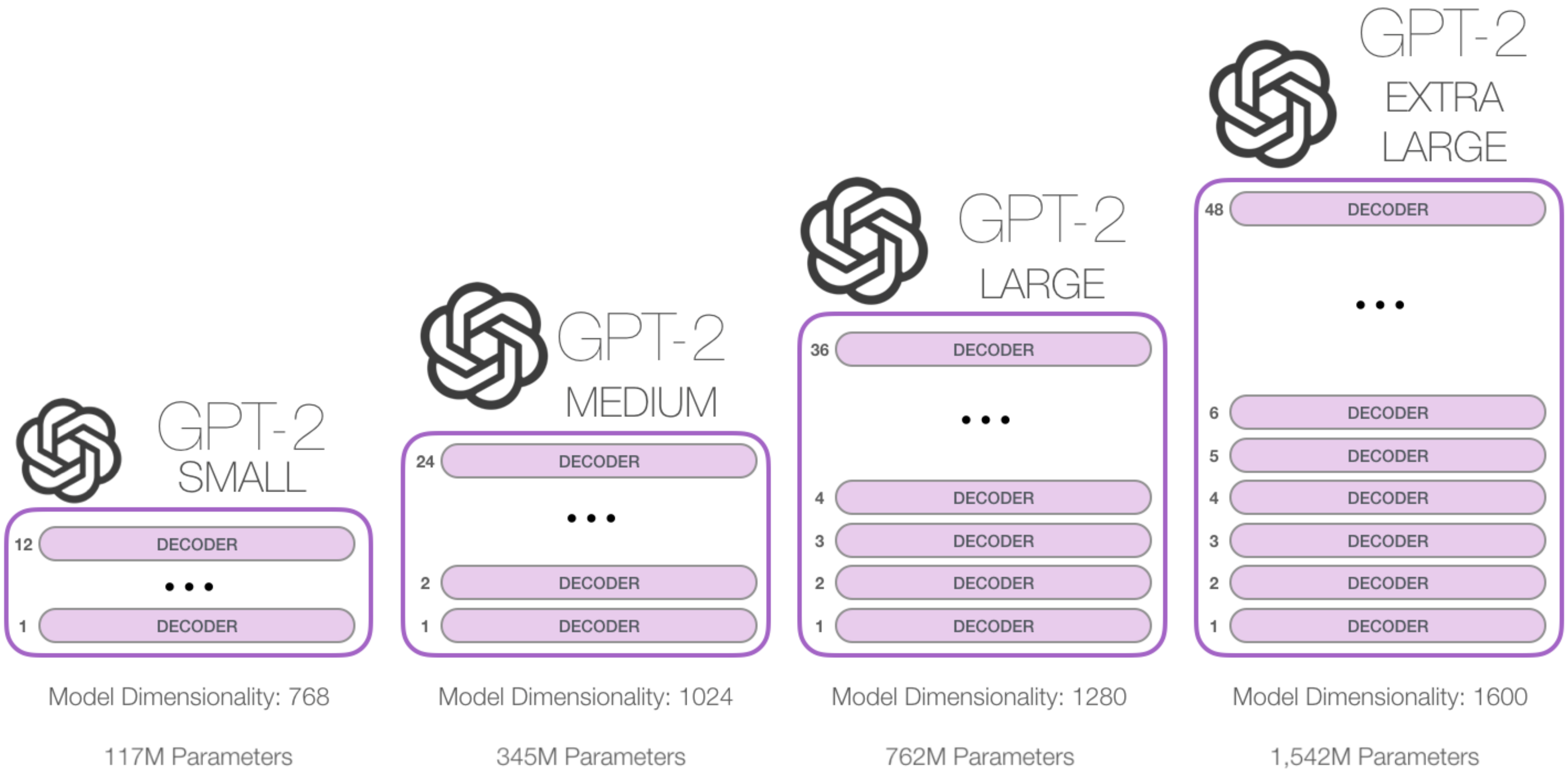


Image source: https://jalammar.github.io/illustrated-bert/

GPT-2



[Radford et al., 2018](#)
Image source: https://jalammar.github.io/illustrated-gpt2/

GPT-2

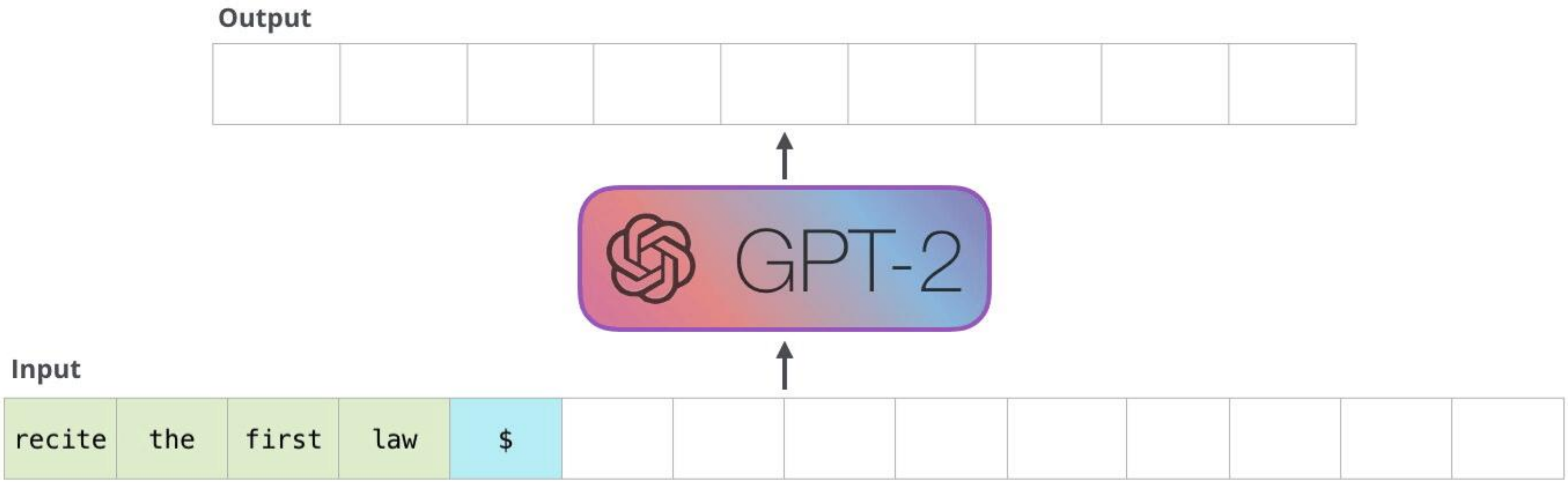


Image source: https://jalammar.github.io/illustrated-gpt2/

Masked Self-Attention

Queries

robot	must	obey	orders

X

Keys

robot	must	obey	orders
robot	must	obey	orders
robot	must	obey	orders
robot	must	obey	orders

=

Scores (before softmax)

0.11	0.00	0.81	0.79
0.19	0.50	0.30	0.48
0.53	0.98	0.95	0.14
0.81	0.86	0.38	0.90

Apply Attention Mask →

Masked Scores (before softmax)

0.11	-inf	-inf	-inf
0.19	0.50	-inf	-inf
0.53	0.98	0.95	-inf
0.81	0.86	0.38	0.90

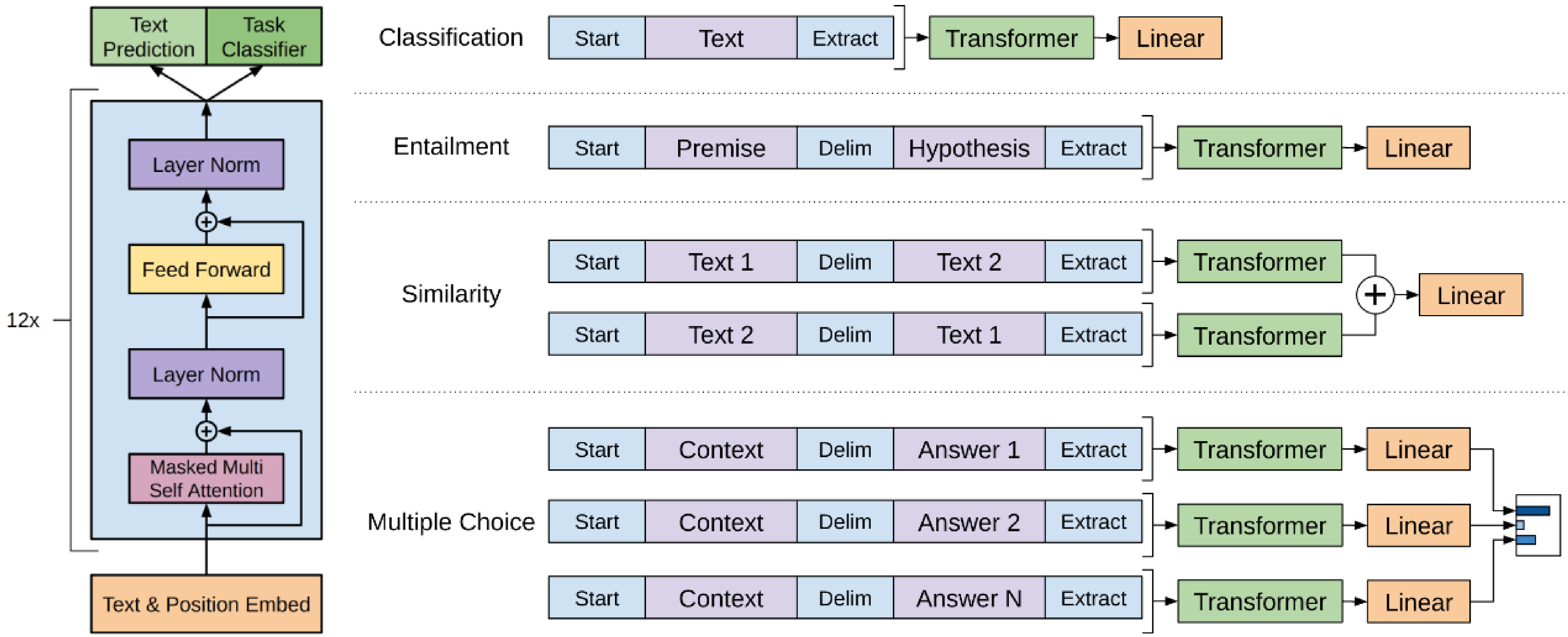
↓

Scores

1	0	0	0
0.48	0.52	0	0
0.31	0.35	0.34	0
0.25	0.26	0.23	0.26

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

GPT 1 Capabilities



[Radford et al., 2018]

Machine Translation with GPT-2

Training Dataset

I	am	a	student	<to-fr>	je	suis	étudiant
let	them	eat	cake	<to-fr>	Qu'ils	mangent	de
good	morning	<to-fr>	Bonjour				

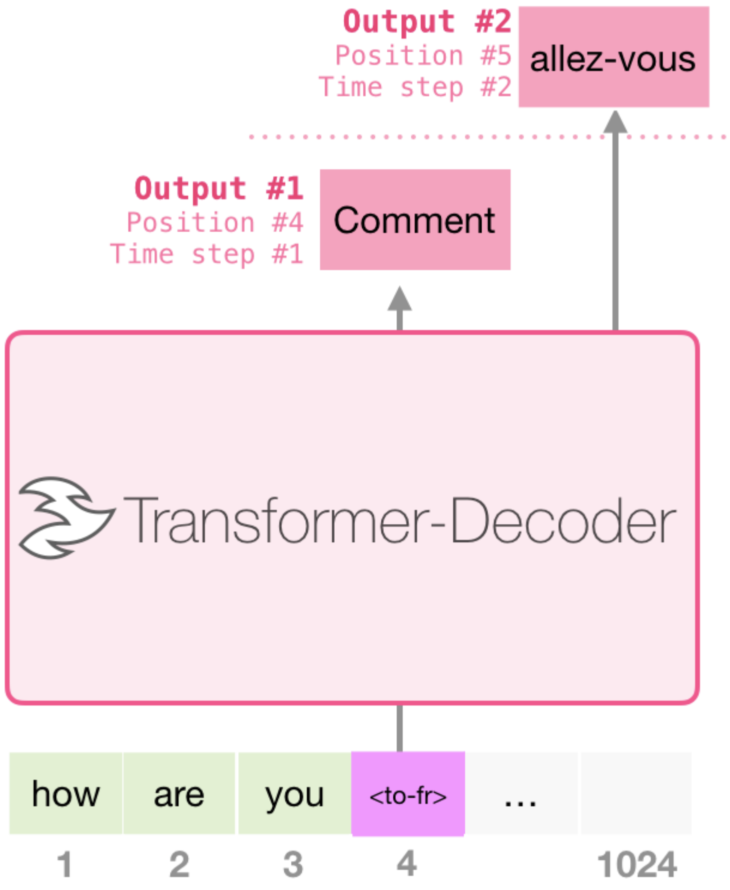
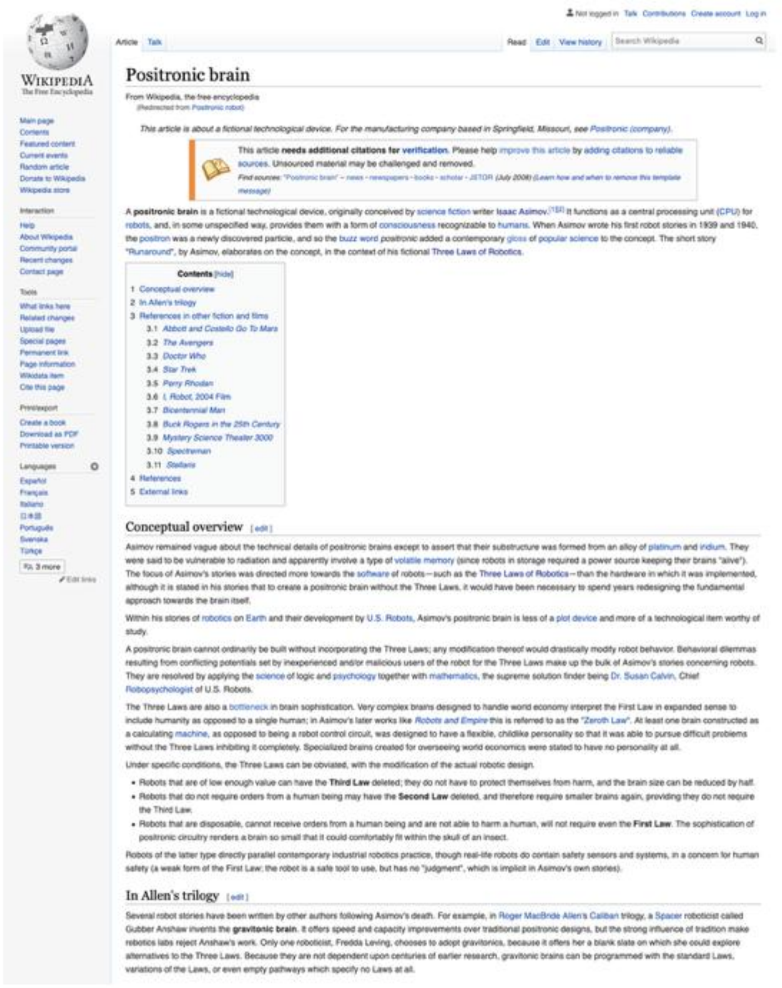


Image source: https://jalammar.github.io/illustrated-gpt2/

Summarization with GPT-2



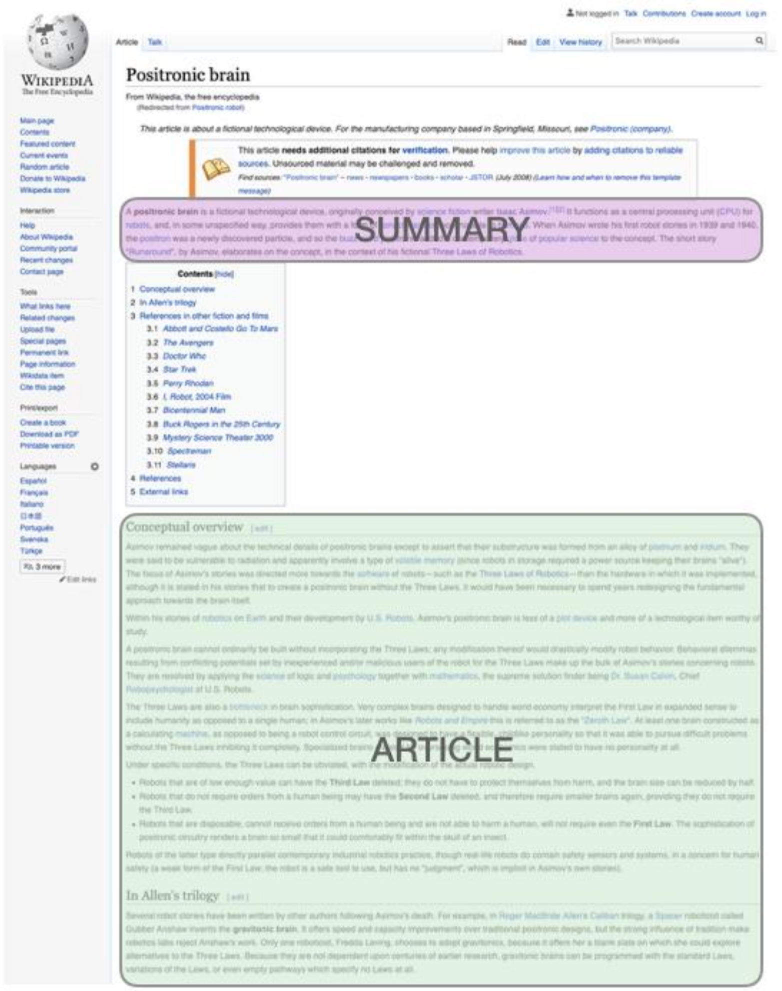


Image source: https://jalammar.github.io/illustrated-gpt2/

Summarization with GPT-2

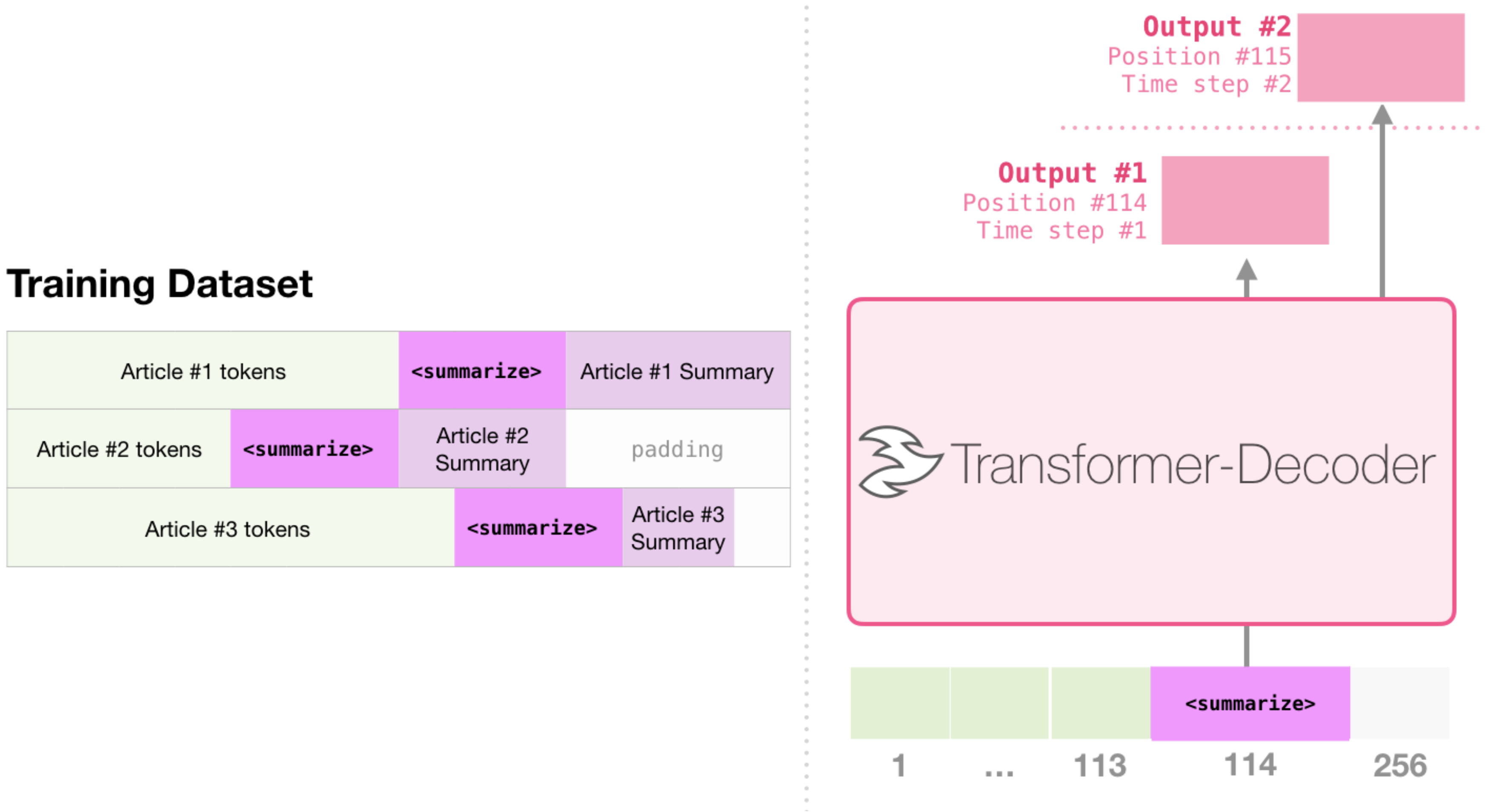
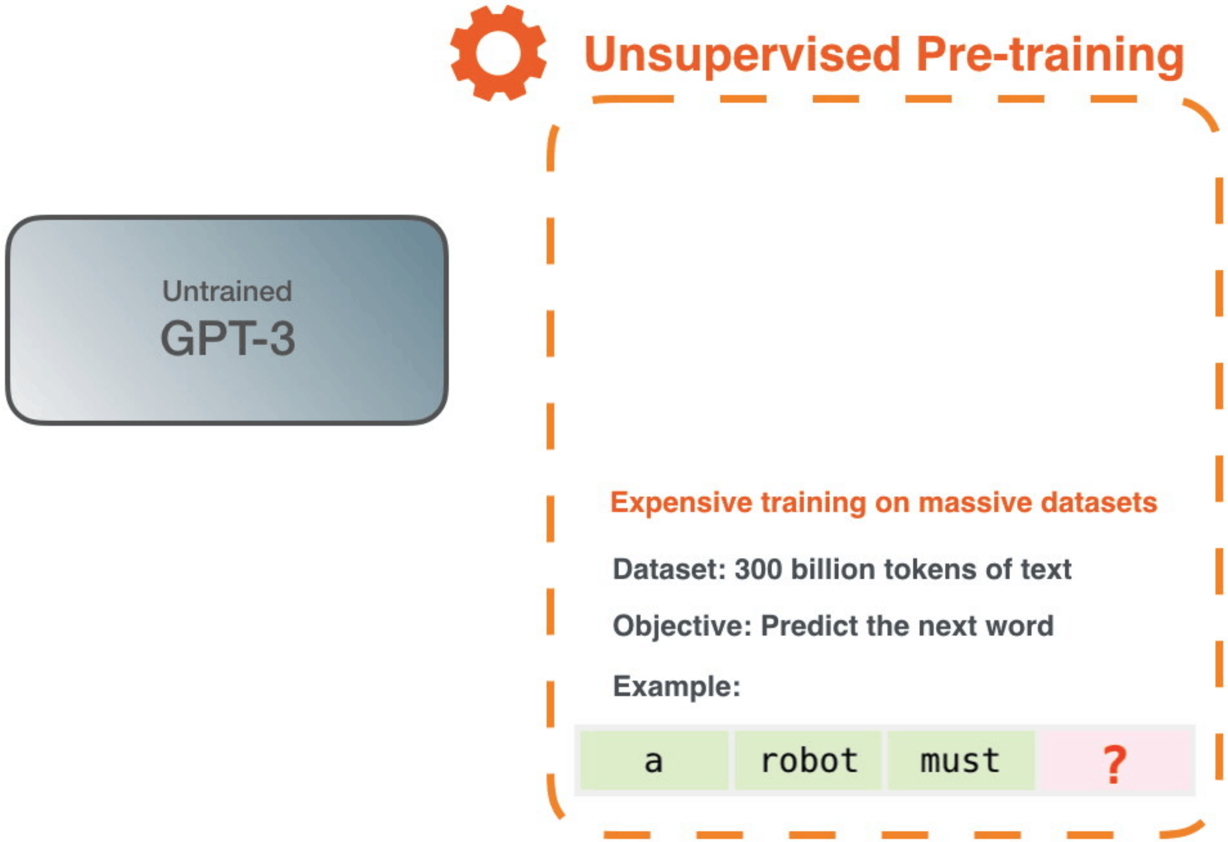


Image source: https://jalammar.github.io/illustrated-gpt2/

GPT-3



Source: https://jalammar.github.io/how-gpt3-works-visualizations-animations/

GPT-3 Specifics

Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
GPT-3 XL	1.3B	24	2048	24	128	1M	2.0×10^{-4}
GPT-3 2.7B	2.7B	32	2560	32	80	1M	1.6×10^{-4}
GPT-3 6.7B	6.7B	32	4096	32	128	2M	1.2×10^{-4}
GPT-3 13B	13.0B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3 175B or "GPT-3"	175.0B	96	12288	96	128	3.2M	0.6×10^{-4}

Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

Brown et al., 2020

GPT-3 Code Generation

Source: https://jalammar.github.io/how-gpt3-works-visualizations-animations/

LLM Research

- BIG-Bench – Beyond the Imitation Game Benchmark
- https://github.com/google/BIG-bench/blob/main/bigbench/benchmark_tasks/README.md

Outline for Jan 25

- Part 1: Decoder only GPT Model
  - What are GPT-class Generative Large Language Models
  - Data preparation for GPT model training
  - GPT finetuning (Assignment)
- Part 2: LLMs and Interacting with them
  - Commercial and open source LLMs
  - What are the main issues in LLMs to be aware of?
  - Taxonomy of interaction with LLMs
  - Prompting Strategies – ZSL, FSL, CoT, ReACT, DSP
  - Parameter Efficient Fine Tuning (LoRA, QLoRA)

Commercial and Open Source LLMs

- Commercial – GPT3.5 (ChatGPT), GPT4, Gemini Pro, Claude 3
- Open Source – Gemma, Llama-2, Mistral, Zephyr
- Parameter count in 1-200 Billion Range

- How to understand size of LLMs?
  - In terms of parameter count
    - context length
    - Embedding dimension
    - number of weights and biases
    - Attention heads
    - Vocabulary size during tokenization
    - Training data size (typically in terms of number of tokens), source
- Links:
  - https://github.com/eugeneyan/open-llms
  - https://crfm.stanford.edu/helm/classic/latest/
  - https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard

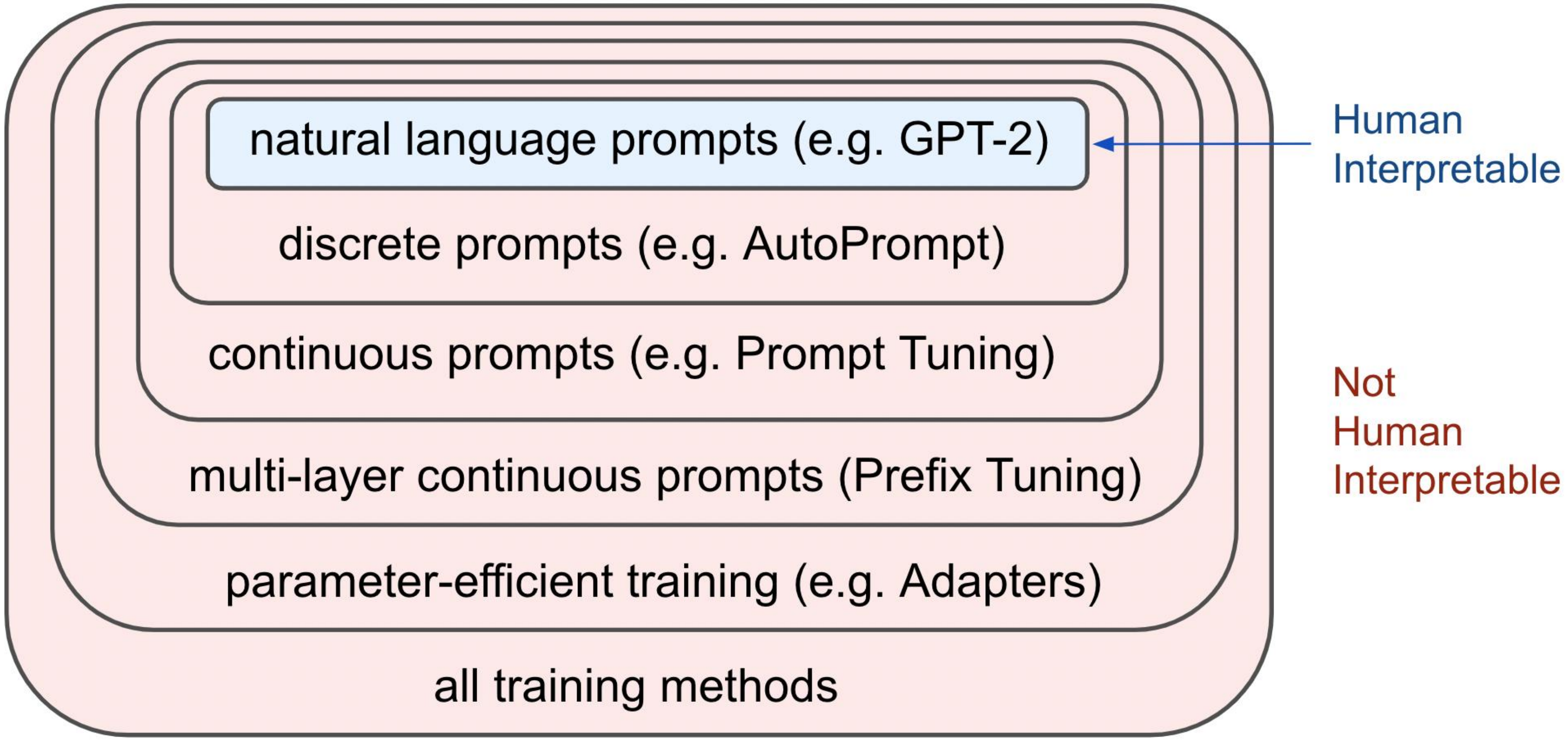
Challenges with LLM

- Size
- Cost
- Out of date facts
- Hallucination
- Harmful content

Ways to interact with a LLM

- Use case: New domain, proprietary data, want to perform a NLP/Generative Task
- 2 Major ways to achieve results
  - Zero-Shot/Few-Shot Learning using Prompt Engineering
  - Fine Tuning – Start with a LLM and do weight updates
- Prompt Engineering
  - Create manual or machine generated prompts to achieve specific tasks
  - Prompt Tuning, Prefix tuning, Auto Prompt – machine learning for prompts
  - Can be done with all LLMs
- Fine Tuning
  - Update all weights and biases of a LLM
  - Parameter efficient fine tuning – Adapters, LoRA
  - Can be done only with open source/open weight models

Taxonomy of Interaction/Prompting Methods



Prompt Engineering

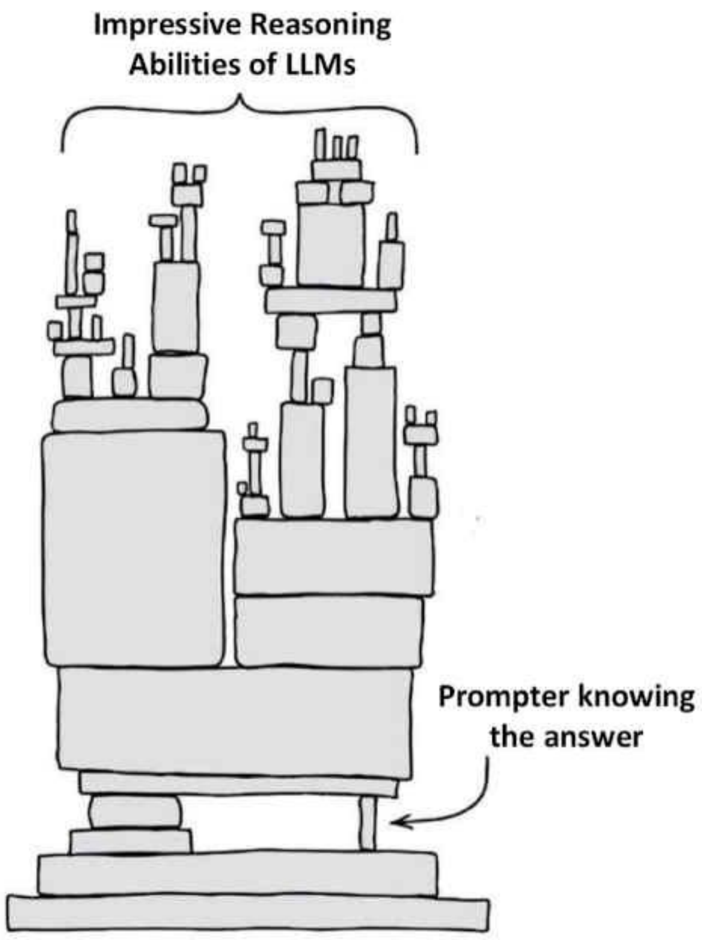
- A prompt is natural language text describing the task that an AI should perform
- Examples:
  - "what is Neural Network?”
  - "write a poem about leaves falling",
  - a short statement of feedback - "too verbose", "too formal", "rephrase again", "omit this word" or
  - a longer statement including context, instructions and input data.
- Prompt Engineering: The process of structuring text that can be interpreted and understood by a generative AI model

Prompting, Coding, Searching

- Different ways of interacting with a computer
- Coding – instructions to execute an algorithm in a high level computer language
- Searching – Keywords, Phrases as input to a webscale search engine
- Prompting – Natural Language instructions to a LLM

- All cases – activities are done to obtain a desirable output

Prompt Engineering: Our Future?



Basic Prompting

- Appending the text at the beginning of a sequence for completion:
- Example Prompt: "When a dog sees a squirrel, it will usually..."
- GPT-2 Small Output:
  - "be afraid of anything unusual. As an exception, that's when a squirrel is usually afraid to bite."
- GPT-2 XL Output:
  - "lick the squirrel. It will also touch its nose to the squirrel on the tail and nose if it can."

Radford et al. 2018

Few-shot Prompting

- Specify the task by giving examples of the task.
- Also called in-context learning.

Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
Translate English to French:     ← task description
cheese =>                        ← prompt
```

One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

```
Translate English to French:     ← task description
sea otter => loutre de mer       ← example
cheese =>                        ← prompt
```

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
Translate English to French:     ← task description
sea otter => loutre de mer       ← examples
peppermint => menthe poivrée
plush girafe => girafe peluche
cheese =>                        ← prompt
```

Brown et al. 2020

Zero Shot Learning

- If an LLM can do a novel task without any examples it is called Zero Shot Learning
- Example:
  - Explain this joke:
- In previous versions of LLMs, explain the joke was solved using a 2-shot prompt.
- With scale, many tasks have become an "emergent property"

Few Shot Learning

- You give 1, 2, or a few examples of the input and output that you want
- Then stop the prompt with the last input, and then LLM produces an output

FSL - Need

To do a "farduddle" means to jump up and down really fast. An example of a sentence that uses the word farduddle is:

View other drafts

I couldn't find any reliable sources that define the word "farduddle" as "to jump up and down really fast."

The word "farduddle" is a nonsense word, meaning it is a made-up word that does not have a real meaning. It is sometimes used as a placeholder word when someone is trying to think of the real word. For example, someone might say "I farduddle what you mean" when they mean "I understand what you mean."

In the prompt you gave me, the word "farduddle" is being used as a verb. However, it is not clear what the verb means. It could mean to jump up and down really fast, but it could also mean something else entirely. Without more context, it is impossible to say for sure what the word means in that sentence.

I hope this helps!

FSL - Example

A "whatpu" is a small, furry animal native to Tanzania. An example of a sentence that uses the word whatpu is: We were travelling in Africa and we saw these very cute whatpus. To do a "farduddle" means to jump up and down really fast. An example of a sentence that uses the word farduddle is:

Sure, here are some examples of sentences that use the word "farduddle":

- The excited child fardaddled up and down when she saw the new puppy.
- The cheerleaders fardaddled on the sidelines to cheer on their team.
- The basketball players fardaddled in celebration after winning the game.
- The crowd fardaddled when the fireworks went off.
- The children fardaddled in the snow.

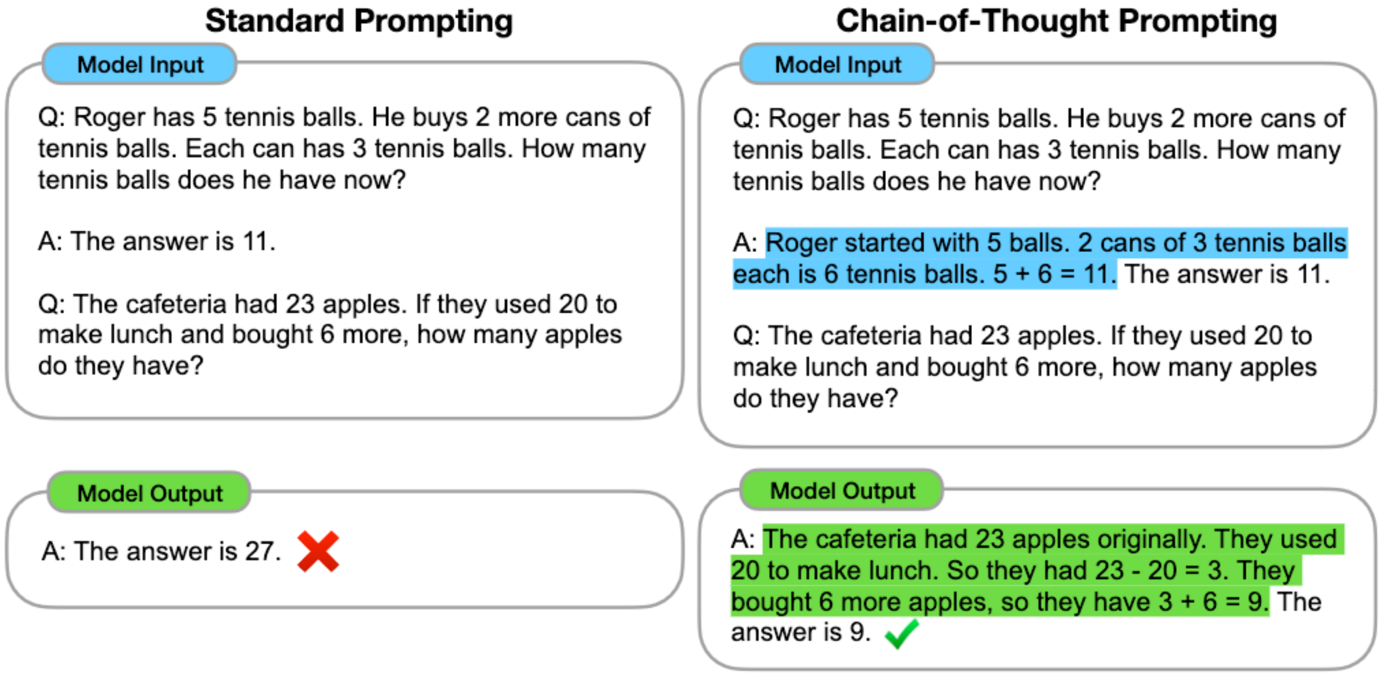
I hope this helps!

List of Prompting Strategies

1. Chain of Thought Prompting (CoT)
2. Tree of Thought Prompting
3. Prompts with Instructions
4. Prompt Chaining
5. Directed Stimulus Prompting
6. ReAct

Chain-of-Thought Prompting

- Some tasks require the language models to perform complex reasoning.
- Chain-of-Thought(CoT) prompts enable the model to explain its reasoning before producing an answer.



Wei et al. 2022

Department of Computational and Data Sciences

INDIAN INSTITUTE OF SCIENCE
भारतीय विज्ञान संस्थान

Can LMs do reasoning?

Standard Prompting

Example Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

Example Output

A: The answer is 11.

Prompt

The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Response

The answer is 50.

Chain of thought prompting

Example Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

Example Output

Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Prompt

The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Response

The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23-20 = 3. They bought 6 more apples, so they have 3+6=9. The answer is 9.

Source: Google PaLM

Chain of Thought Prompting

- Chain-of-thought (CoT) prompting is a technique that allows large language models (LLMs) to solve a problem as a series of intermediate steps before giving a final answer.
- Chain-of-thought prompting improves reasoning ability by inducing the model to answer a multi-step problem with steps of reasoning that mimic a train of thought.
- It allows large language models to overcome difficulties with some reasoning tasks that require logical thinking and multiple steps to solve, such as arithmetic or commonsense reasoning questions.

Using LLM for your task and Data

- Fine Tuning
- Low Rank Adaptation
- Quantized Low Rank Adaptation

Parameter Efficient Fine Tuning

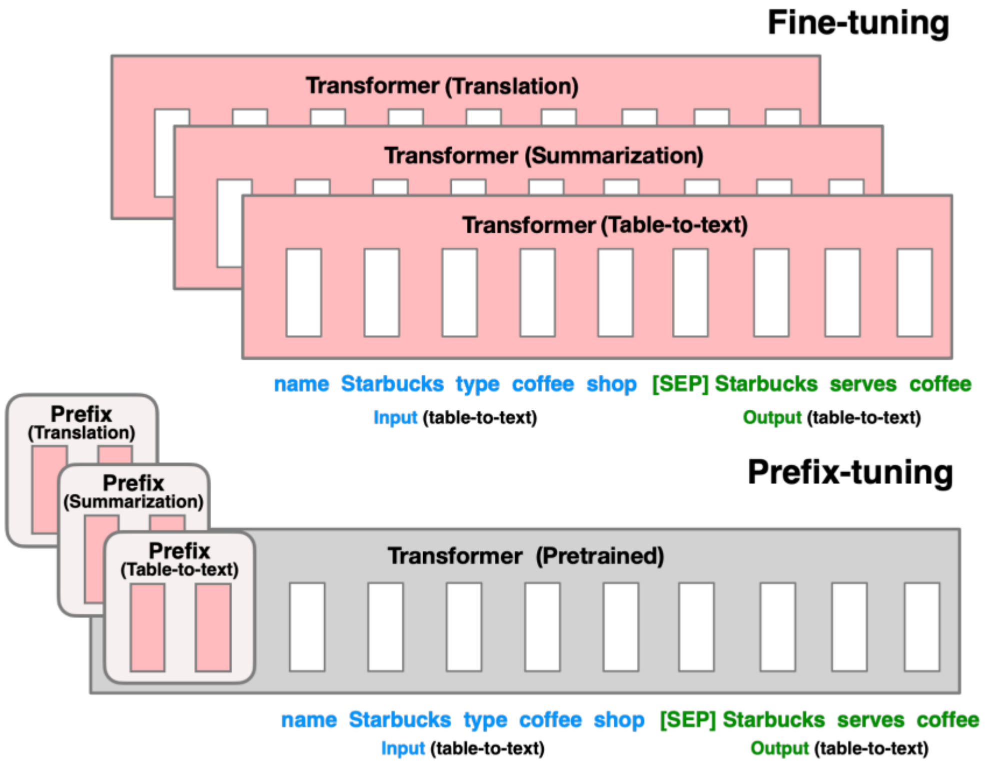
Prompt modifications	Adapter methods	Reparameterization
“Hard” prompt tuning	Adapters	Low rank adaptation (LoRA)
“Soft” prompt tuning		
Prefix-tuning	LLaMA-Adapter	

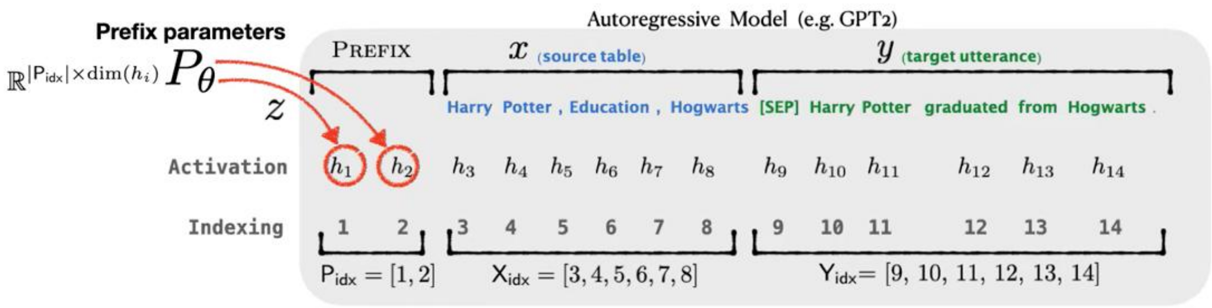
Hard Prompt Tuning

```
1) "Translate the English sentence '{english_sentence}' into German: {german_translation}"

2) "English: '{english_sentence}' | German: {german_translation}"

3) "From English to German: '{english_sentence}' -> {german_translation}"
```

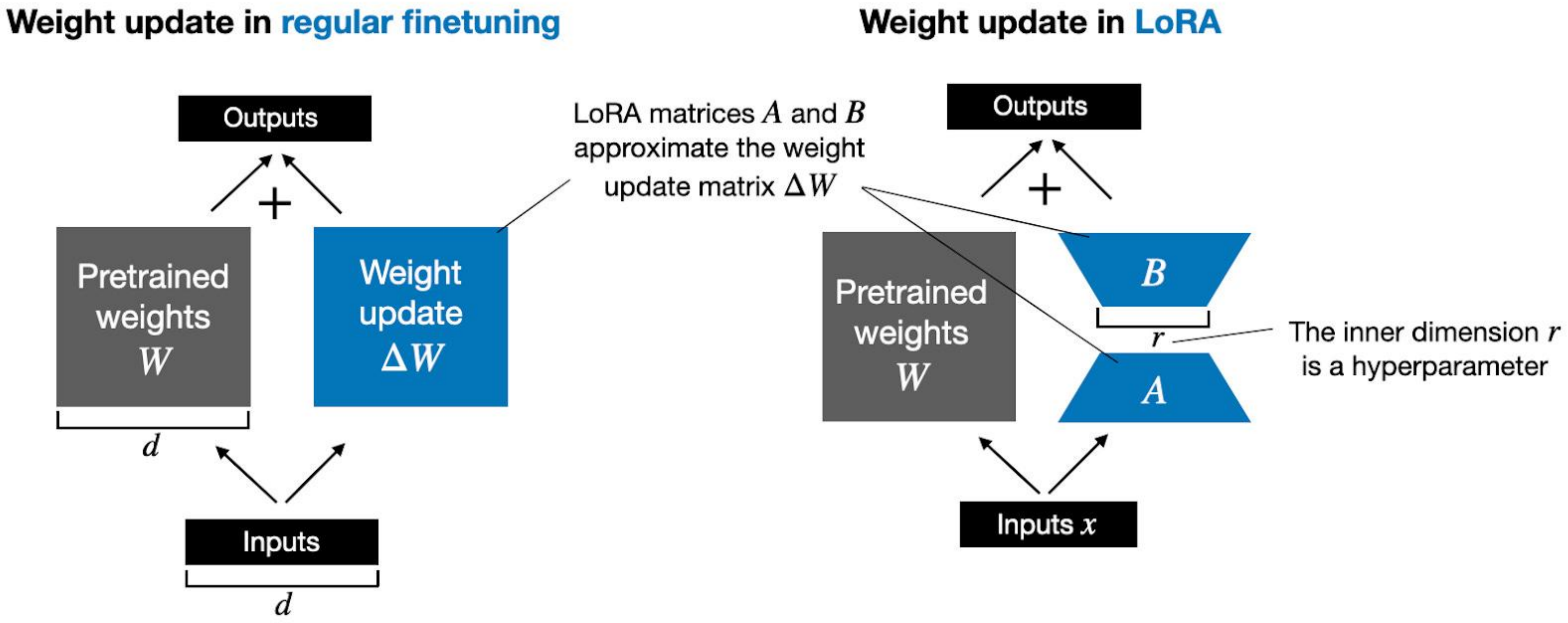
Prefix Tuning

- Add prefix parameters that are learnt during the training of GPT





LoRA – Low Rank Adaptation



Quantization

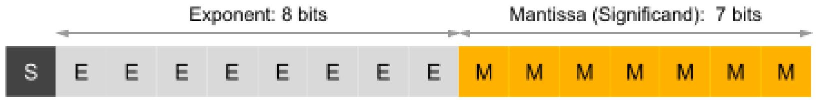
- Technique to reduce the size of deep neural networks (including LLMs) by changing the precision of the weights and biases data structure
- Pros: Lower model size allowing for deployment on edge device
- Cons: Lower accuracy
- Concept:
  - Typical computation happens in Floating Point 32 precision (FP32) or FP16
  - Quantized models are converted to INT4 either
    - Post training (PTQ – Post Training Quantization)
    - During training (QAT – Quantization Aware Training)
  - PTQ is easier than QAT
- HuggingFace hub has quantized models that you can use and deploy in LLMOps

Floating Point Sizes

Floating Point Formats

bfloat16: Brain Floating Point Format

Range: ~$1e^{-38}$ to ~$3e^{38}$

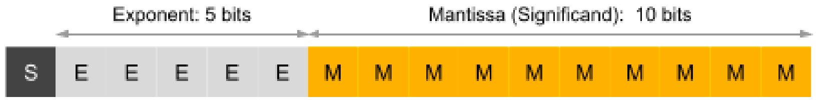


fp32: Single-precision IEEE Floating Point Format

Range: ~$1e^{-38}$ to ~$3e^{38}$

fp16: Half-precision IEEE Floating Point Format

Range: ~$5.96e^{-8}$ to 65504



Size of Quantized Models

Model	Original Size (FP16)	Quantized Size (INT4)
Llama2-7B	13.5 GB	3.9 GB
Llama2-13B	26.1 GB	7.3 GB
Llama2-70B	138 GB	40.7 GB

Q-LoRA

- quantized LoRA - a technique that further reduces memory usage during finetuning.
  - During backpropagation, QLoRA quantizes the pretrained weights to 4-bit precision and uses paged optimizers to handle memory spikes.
- But Q-LoRA comes with a runtime penalty

Default LoRA with 16-bit brain floating point precision:

- Training time: 1.85 h
- Memory used: 21.33 GB

QLoRA with 4-bit *Normal Floats:*

- Training time: 2.79 h
- Memory used: 14.18 GB

SOURCE: https://magazine.sebastianraschka.com/p/practical-tips-for-finetuning-llms