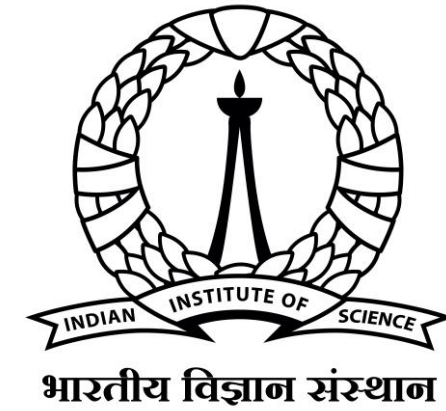




Department of Computational and Data Sciences



# AI&MLOps Module 4: Generative AI

Deepak Subramani

Assistant Professor

Dept. of Computational and Data Science

Indian Institute of Science Bengaluru

# Outline for Week 01

- Part 1: Decoder only GPT Model
  - What are GPT-class Generative Large Language Models
  - Generative AI Use Cases
  - 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
  - Parameter Efficient Fine Tuning (LoRA, QLoRA)



# Outline for Week 02

- Part 1:
  - Prompting Strategies – ZSL, FSL, CoT, ReACT, DSP
- Part 2:
  - Instruction Tuning
- Part 3:
  - Orchestration
  - Retrieval Augmented Generation
- Part 4:
  - LLM Guardrails
  - LLM Agents

# Outline for Week 03

- Part 1:
  - Deep learning as a representation learning system
  - Autoencoders for pre-training new situations
- Part 2:
  - Modern GenAI Image Pipelines
  - CLIP
  - Stable Diffusion
- Part 3: (May Know)
  - GANs – Generative Adversarial Networks
  - Variational Auto Encoders
  - Resource for Math of Diffusion Models (Link Shared in Part 2)

# Predictive AI and Generative AI

- Predictive AI
  - Input: Any of the data modality
  - Output: Continuous or Categorical
- Generative AI
  - Input: Any of the data modality
  - Output: Text, Image, Video, Audio



# Generative AI Use Cases

- **Healthcare Assistance** – Offering support in areas like patient interaction, medical documentation, and even as assistive tools for diagnosis and treatment planning, though they don't replace professional advice.
- **Personal Assistants** – Managing schedules, setting reminders, answering questions, and even helping with email management and other administrative tasks.
- **Legal and Compliance Assistance** – Assisting in legal research, document review, and drafting legal documents (without replacing professional legal advice).
- **Accessibility Tools** – Enhancing accessibility through tools like voice-to-text conversion, reading assistance, and simplifying complex text.
- **Interactive Entertainment** – In gaming and interactive storytelling, creating dynamic narratives, character dialogue, and responsive storytelling elements.
- **Marketing and Customer Insights** – Analyzing customer feedback, conducting sentiment analysis, and generating marketing content, providing valuable insights into consumer behavior.
- **Social Media Management** – Managing social media content, from generating posts to analyzing trends and engaging with audiences.
- **Human Resources Management** – Aiding in resume screening, answering employee queries, and even in training and development activities.

# Generative AI Use Cases

- **Customer Service and Support** – Providing customer support, handling inquiries, resolving issues, and offering information 24/7.
- **Content Creation and Copywriting** – Generating creative content, such as articles, blogs, scripts, and advertising copy.
- **Language Translation and Localization** – Translation services for various content types, aiding in bridging language barriers and localizing content for different regions.
- **Education and Tutoring** – Functioning as personalized tutors, providing explanations, answering questions, and assisting with learning materials in a wide range of subjects.
- **Programming and Code Generation** – Writing, reviewing, and debugging code, thereby speeding up the development process and helping in learning programming languages.
- **Research and Data Analysis** – Sifting through large volumes of text, summarizing information, and extracting relevant data, which is invaluable for research and analysis.

# Generative AI

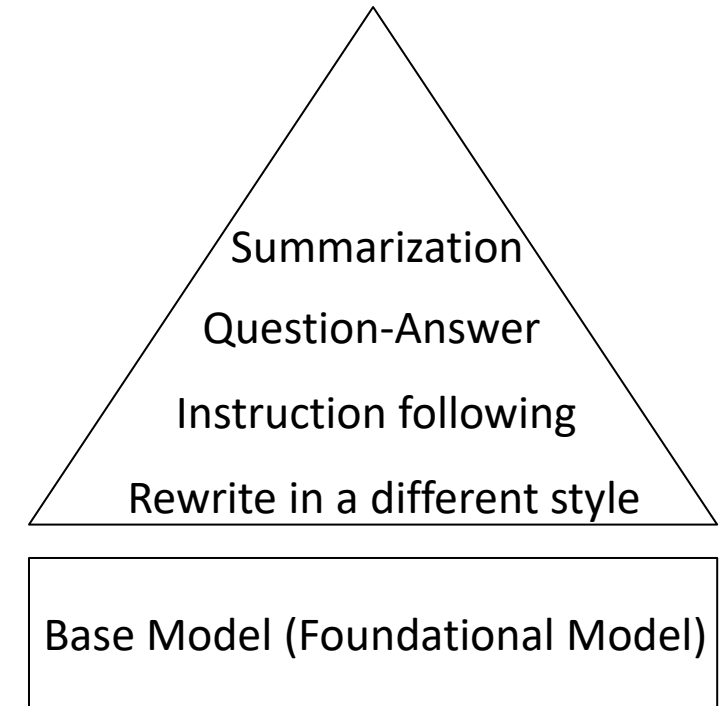
- Text to Text
  - Text to Image/Video
  - Image/Video to Text
  - Image/Video to Image/Video
  - Text to Audio
  - Audio to Text
  - Text/Image to Code
- 
- Input is the “Prompt”; Model is a Large Language/Vision Model; Output is Image/Video/Text/Speech





# Foundational Model

- Large-scale AI model trained on vast amounts of diverse data
- Serves as a base for multiple downstream tasks and applications
- Key characteristics:
  - Broad knowledge and capabilities
  - Prompt engineering to make it perform tasks
  - Retrieval Augmented Generation for tapping into specific data
  - Adaptable through fine-tuning
  - Generalize to new tasks with minimal additional training
- **Examples:** GPT, BERT, T5



# Latest Developments

- Anthropic
  - Claude 3.5 Sonnet
- Microsoft-OpenAI integration
  - Bing search
  - PowerBI with ChatGPT
- Generative Image
  - Photoshop
  - MidJourney
- Generative Videos
  - Sora
- LLMOps pipeline
  - LangChain/LlamaIndex + OpenAI/Anthropic/Llama

# Language Model

- Any model that can predict the probability of the next token in a sequence of text input (converted to embeddings) is called a Language Model
- LM captures the latent space of language: its statistical structure
- Large Language Models are trained on large text corpora (trillions of tokens) and have billions of parameters
- They have emergent abilities
  - Can do tasks for which it is not explicitly trained
  - Today, we don't take a chance and make it learn to follow instructions
- Finally, LLMs would all be a sophisticated lookup table!

# Language Modeling Approaches

- Masked Language Modeling
  - Tokens in a document are randomly masked
  - Neural Models are trained to predict the masked token correctly
  - This is a fill-in-the-blank task
  - Example:
    - The cat sits on the mat.
    - The [MASK] sits on the mat.
    - The model's task is to predict "cat" based on the context
- Sentence Completion Modeling (Next token prediction)
  - Model is set up in an autoregressive mode
  - At each inference step, the model predicts the next token (from the vocabulary as a probability distribution)
  - (k+1)st token is predicted with (prompt+predicted k tokens) as input
  - (k+2)nd token is predicted with (prompt+predicted k+1 tokens) as input

# Transformers

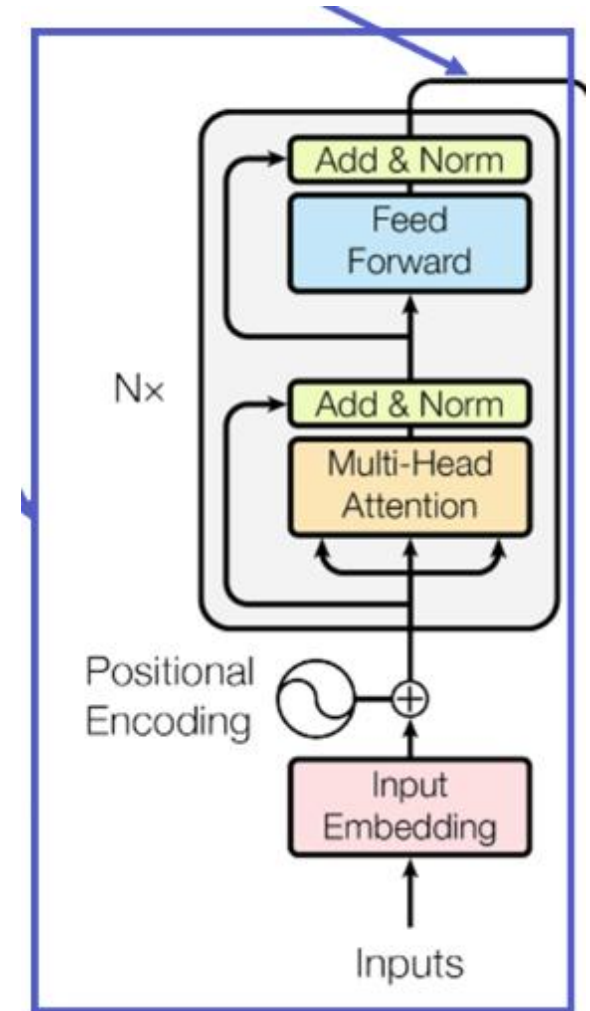
- Transformer Encoder
  - Converts a sequence of words to a vector representation
  - This vector representation can be used for text-understanding tasks
  - Trained using fill-in-the-blanks tasks - MLM
- Transformer Decoder
  - Uses the context of the sequence of words so far (sometimes with an additional context from encoder or retrieval) to predict next token in the sequence
  - Trained using next-token-prediction tasks



# Transformer Encoder

## Steps in a Transformer Encoder

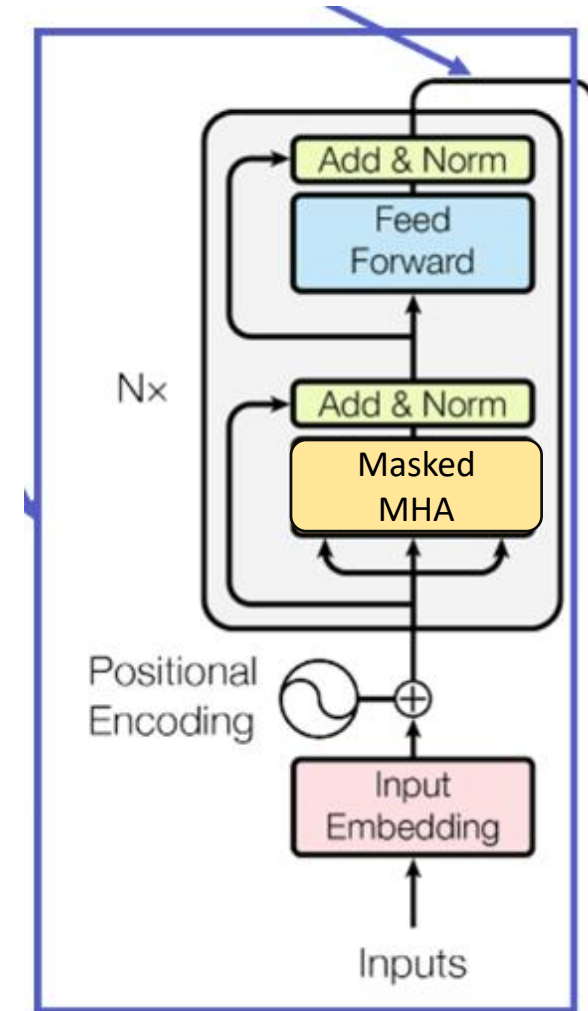
1. Tokenize (append special tokens – [CLS])
2. Get Encoded sequence added with position
3. Multi-headed attention
  - Converts encoded sequence to context aware representation (still a sequence)
4. Residual and layer normalization
5. Dense Layers for further representation learning
6. Encoder block outputs a encoded representation
7. Use the representation of [CLS] token to perform text understanding tasks

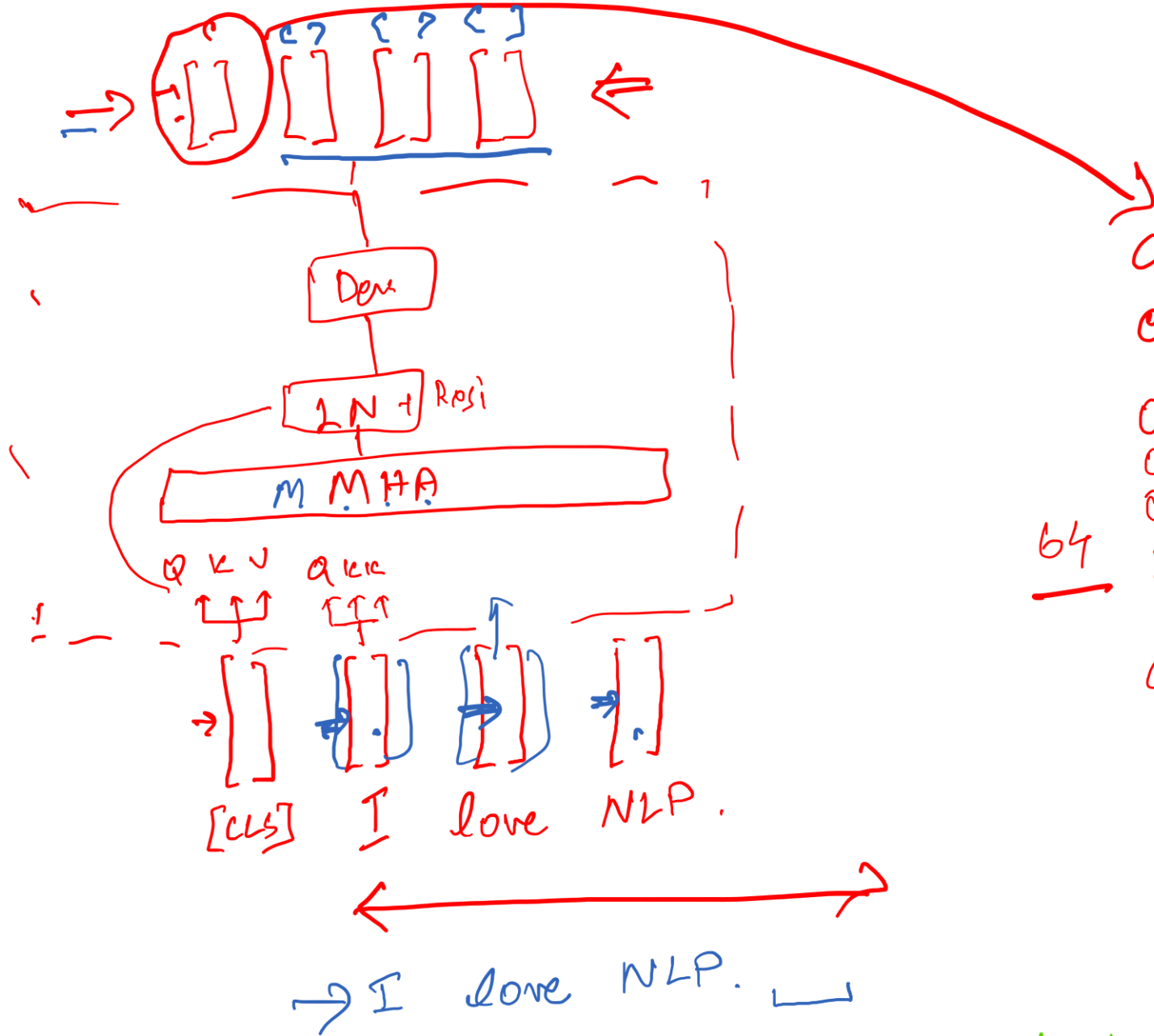


# Transformer Decoder

## Steps in a Transformer Decoder

1. Tokenize the prompt
2. Get Encoded sequence added with position
3. Masked Multi-headed attention
  - Masking makes the attention attend only to tokens to the left
4. Residual and layer normalization
5. Dense Layers for further representation learning
6. Decoder block outputs sequence of representations
7. Use the representation of last token to generate the next token
8. Repeat by including the generated token as part of the prompt





CLS token's  
encoded rep.

64

0		
0	0	
0	0	0 →
:	0	1
0		

LM Head

→ I love NLP. →

That is great. How can I help you?



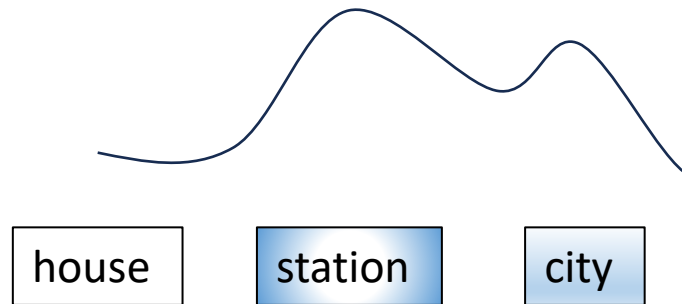
# NLP View

- $NLP = NLU + NLG$
- Encoder-Only Models (BERT) for NLU
- Decoder Only Models (GPT) for NLG

# Generative Pretrained Transformers (GPT)

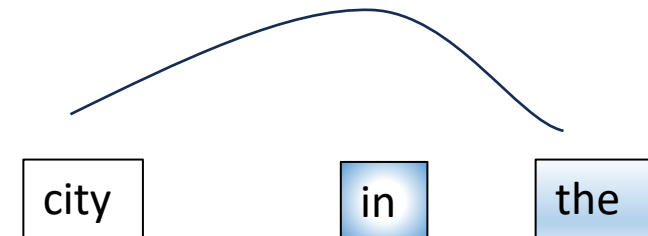
- These are decoder only models.
- Since there is no encoder in this set up, these decoder layers would not have the encoder-decoder attention sublayer that vanilla transformer decoder layers have.
- It only has the masked self attention layer.
- The model predict the next word using massive datasets.

# What does GPT do?



Transformer Decoder

The train left the



Transformer Decoder

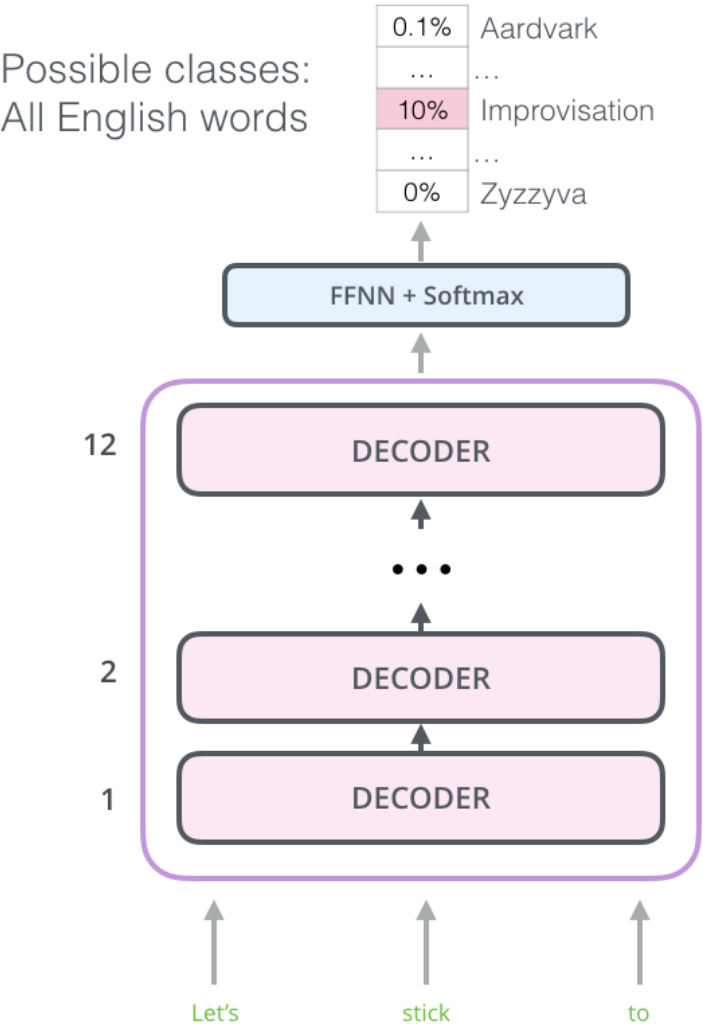
The train left the station

# 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.

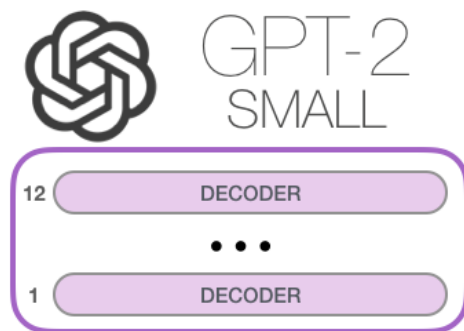


# GPT



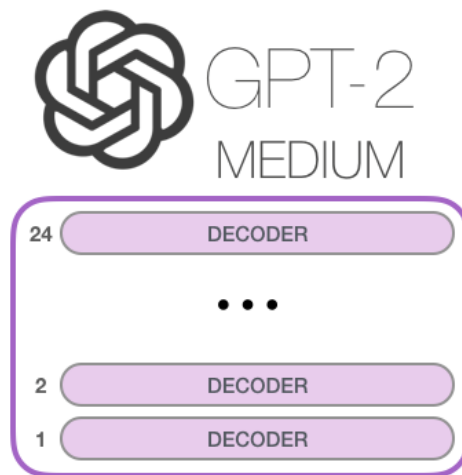


# GPT-2



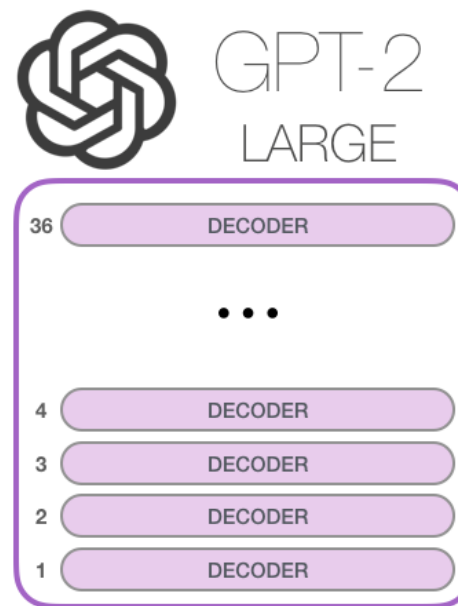
Model Dimensionality: 768

117M Parameters



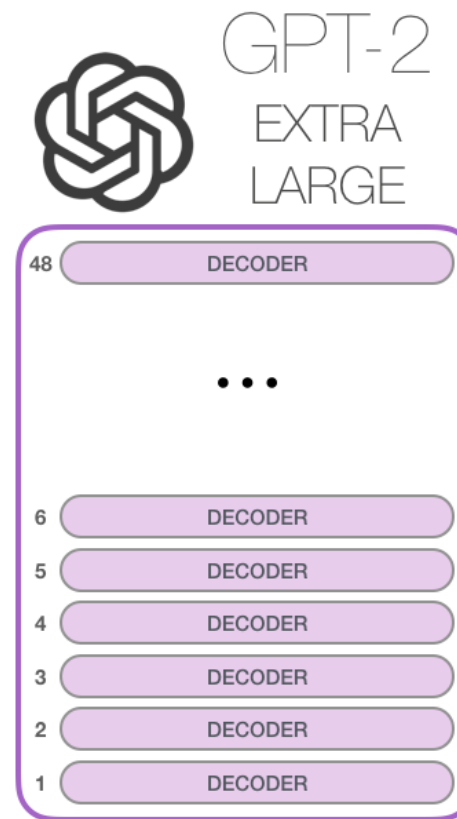
Model Dimensionality: 1024

345M Parameters



Model Dimensionality: 1280

762M Parameters



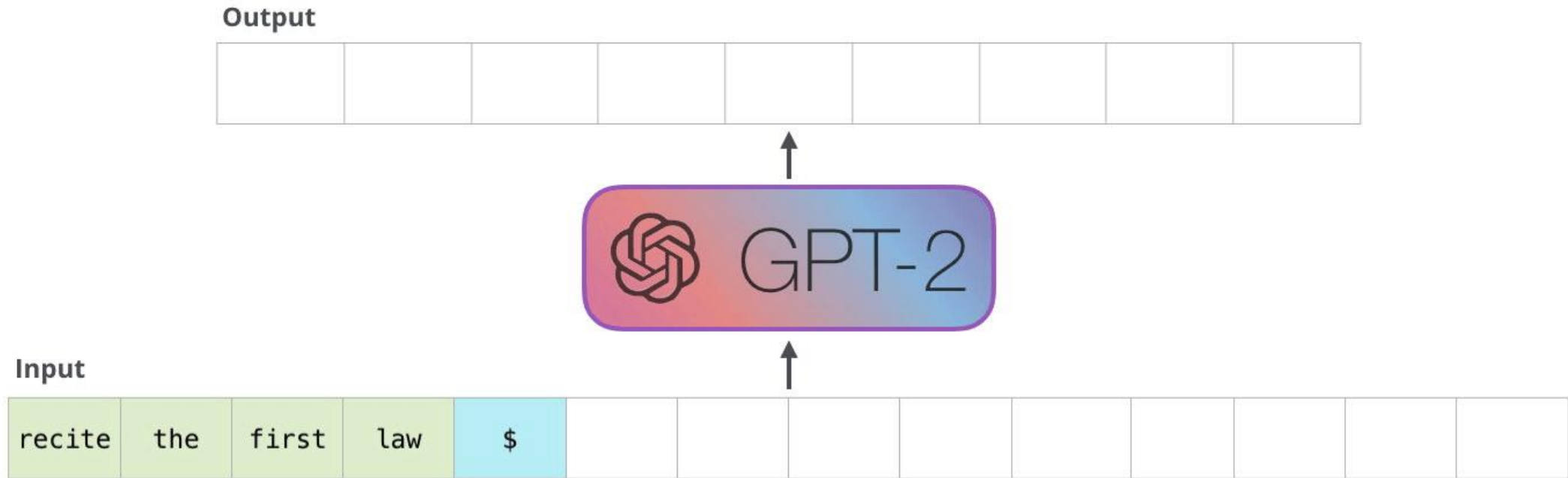
Model Dimensionality: 1600

1,542M Parameters

[Radford et al., 2018](#)

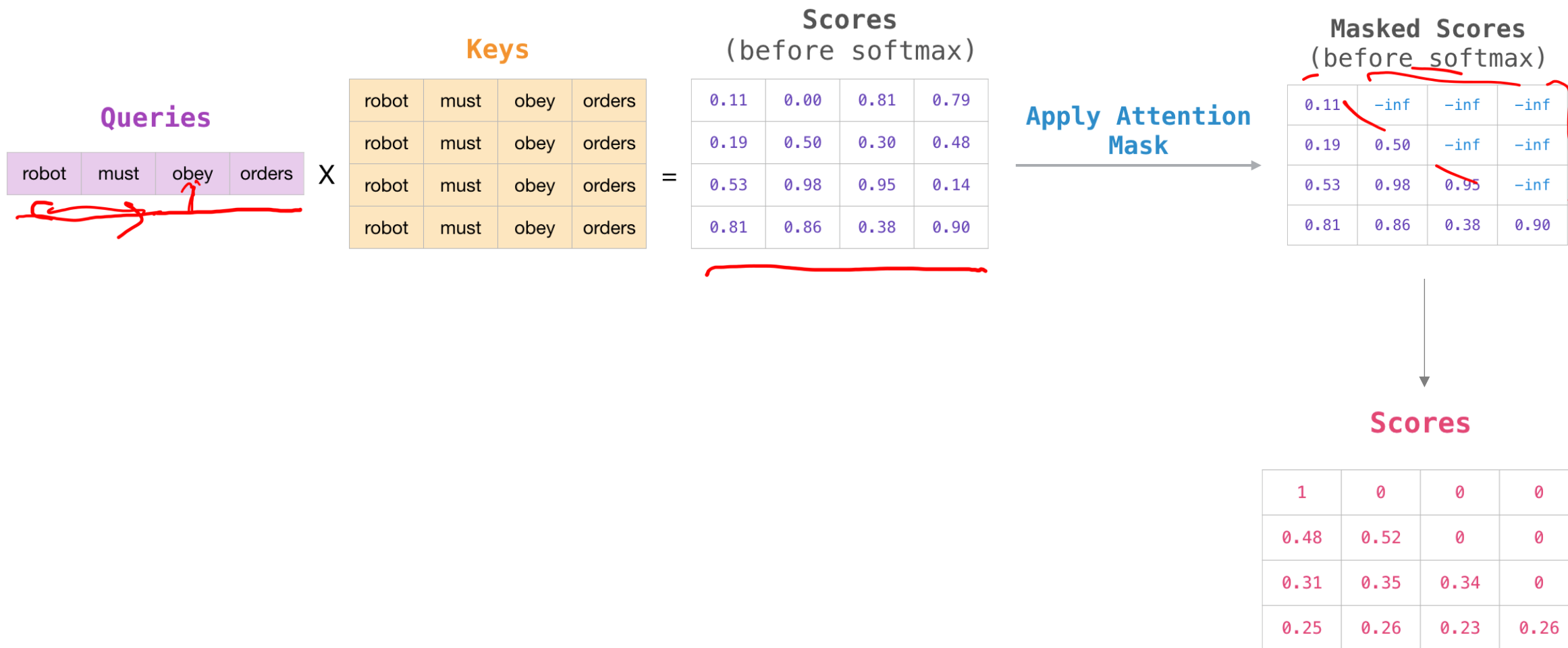
Image source: <https://jalammar.github.io/illustrated-gpt2/>

# GPT-2





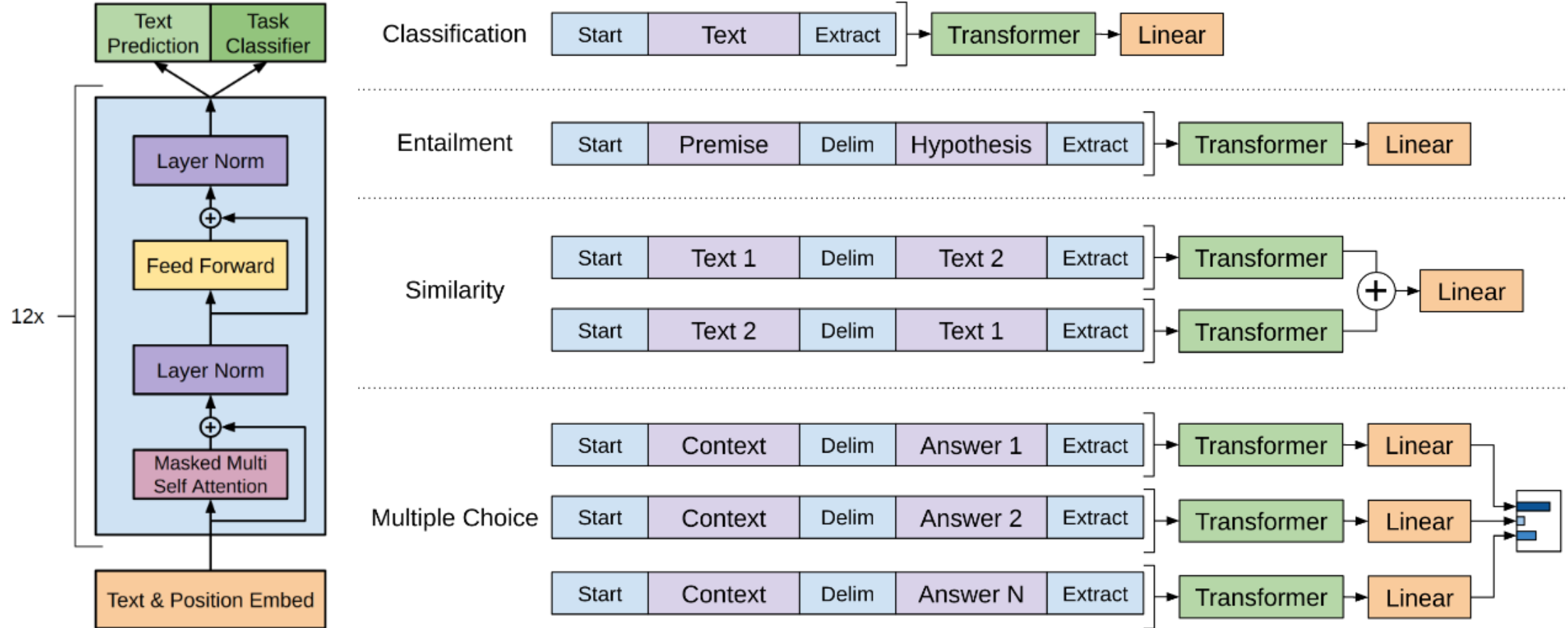
# Masked Self-Attention







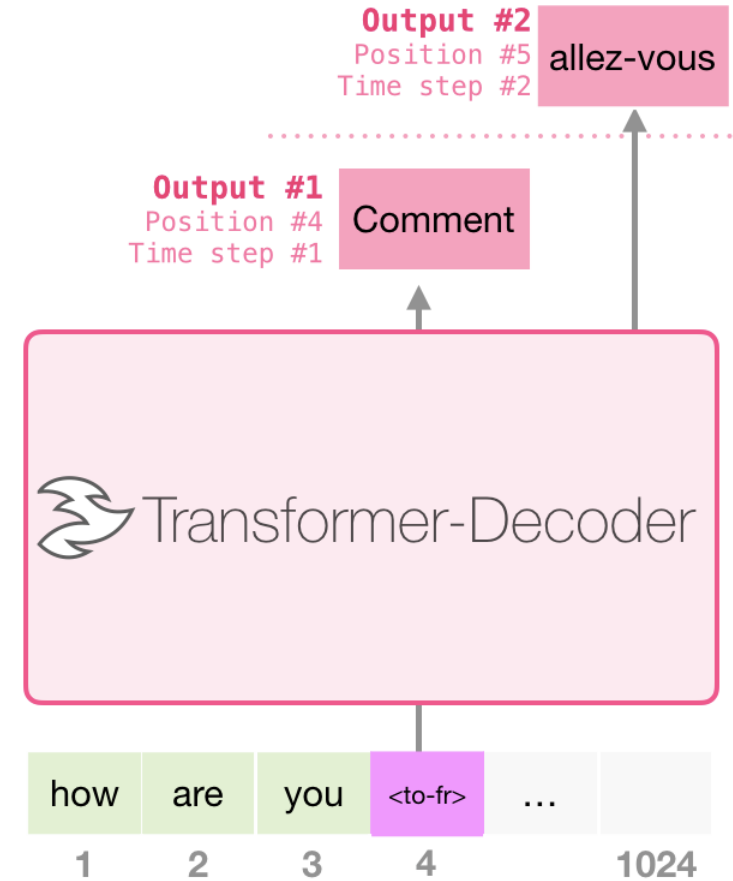
# GPT 1 Capabilities



# 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				





# Summarization with GPT-2

WIKIPEDIA The Free Encyclopedia

Article Talk

Positronic brain

From Wikipedia, the free encyclopedia  
(Redirected from Positronic robot)

This article is about a fictional technological device. For the manufacturing company based in Springfield, Missouri, see *Positronic* (company).

This article needs additional citations for verification. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed.  
Find sources: "Positronic brain" – news – newspapers – books – scholar – JSTOR (July 2008) (Learn how and when to remove this template message)

A **positronic brain** is a fictional technological device, originally conceived by science fiction writer Isaac Asimov<sup>[1][2]</sup> It functions as a central processing unit (CPU) for robots, and, in some unspecified way, provides them with a form of consciousness recognizable to humans. When Asimov wrote his first robot stories in 1939 and 1940, the positron was a newly discovered particle, and so the buzz word positronic added a contemporary gloss of popular science to the concept. The short story "Runaround", by Asimov, elaborates on the concept, in the context of his fictional Three Laws of Robotics.

**Contents** [hide]

- Conceptual overview
- In Allen's trilogy
- References in other fiction and films
  - 3.1 Abbott and Costello Go To Mars
  - 3.2 The Avengers
  - 3.3 Doctor Who
  - 3.4 Star Trek
  - 3.5 Perry Rhodan
  - 3.6 I, Robot, 2004 Film
  - 3.7 Bicentennial Man
  - 3.8 Bulk Rogers in the 25th Century
  - 3.9 Mystery Science Theater 2000
  - 3.10 Spectreman
  - 3.11 Star Wars
- References
- External links

**Conceptual overview** [ edit ]

Asimov remained vague about the technical details of positronic brains except to assert that their substructure was formed from an alloy of platinum and indium. They were said to be vulnerable to radiation and apparently involve a type of *volatile memory* (since robots in storage required a power source keeping their brains "alive"). The focus of Asimov's stories was directed more towards the *software* of robots—such as the Three Laws of Robotics—than the hardware in which it was implemented, although it is stated in his stories that to create a positronic brain without the Three Laws, it would have been necessary to spend years redesigning the fundamental approach towards the brain itself.

Within his stories of *robotics on Earth* and their development by U.S. Robots, Asimov's positronic brain is less of a *plot device* and more of a technological item worthy of study.

A positronic brain cannot ordinarily be built without incorporating the Three Laws; any modification thereof would drastically modify robot behavior. Behavioral dilemmas resulting from conflicting potentials set by inexperienced and/or malicious users of the robot for the Three Laws make up the bulk of Asimov's stories concerning robots. They are resolved by applying the science of logic and psychology together with *mathematics*, the supreme solution finder being Dr. Susan Calvin, Chief Roboticspsychologist at U.S. Robots.

The Three Laws are also a *bottleneck* in brain sophistication. Very complex brains designed to handle world economy interpret the First Law in expanded sense to include humanity as opposed to a single human; in Asimov's later works like *Robots and Empire* this is referred to as the "Zeroth Law". At least one brain constructed as a calculating *machine*, as opposed to being a robot control circuit, was designed to have a flexible, childlike personality so that it was able to pursue difficult problems without the Three Laws inhibiting it completely. Specialized brains created for overseeing world economics were stated to have no personality at all.

Under specific conditions, the Three Laws can be obviated, with the modification of the actual robotic design.

- Robots that are of low enough value can have the **Third Law** deleted; they do not have to protect themselves from harm, and the brain size can be reduced by half.
- Robots that do not require orders from a human being may have the **Second Law** deleted, and therefore require smaller brains again, providing they do not require the Third Law.
- Robots that are disposable, cannot receive orders from a human being and are not able to harm a human, will not require even the **First Law**. The sophistication of positronic circuitry renders a brain so small that it could comfortably fit within the skull of an insect.

Robots of the latter type directly parallel contemporary industrial robotics practice, though real-life robots do contain safety sensors and systems, in a concern for human safety (a weak form of the First Law; the robot is a safe tool to use, but has no "judgment", which is implicit in Asimov's own stories).

**In Allen's trilogy** [ edit ]

Several robot stories have been written by other authors following Asimov's death. For example, in Roger MacBride Allen's *Caliban* trilogy, a Spacer robotist called Gubber Anshaw invents the **gravitronic brain**. It offers speed and capacity improvements over traditional positronic designs, but the strong influence of tradition make robotics fans reject Anshaw's work. Only one robotist, Freddie Leving, chooses to adopt gravitronics, because it offers her a blank slate on which she could explore alternatives to the Three Laws. Because they are not dependent upon centuries of earlier research, gravitronic brains can be programmed with the standard Laws, variations of the Laws, or even empty pathways which specify no Laws at all.

WIKIPEDIA The Free Encyclopedia

Article Talk

Positronic brain

From Wikipedia, the free encyclopedia  
(Redirected from Positronic robot)

This article is about a fictional technological device. For the manufacturing company based in Springfield, Missouri, see *Positronic* (company).

This article needs additional citations for verification. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed.  
Find sources: "Positronic brain" – news – newspapers – books – scholar – JSTOR (July 2008) (Learn how and when to remove this template message)

A **positronic brain** is a fictional technological device, originally conceived by science fiction writer Isaac Asimov<sup>[1][2]</sup> It functions as a central processing unit (CPU) for robots, and, in some unspecified way, provides them with a form of consciousness recognizable to humans. When Asimov wrote his first robot stories in 1939 and 1940, the positron was a newly discovered particle, and so the buzz word positronic added a contemporary gloss of popular science to the concept. The short story "Runaround", by Asimov, elaborates on the concept, in the context of his fictional Three Laws of Robotics.

**SUMMARY**

**Contents** [hide]

- Conceptual overview
- In Allen's trilogy
- References in other fiction and films
  - 3.1 Abbott and Costello Go To Mars
  - 3.2 The Avengers
  - 3.3 Doctor Who
  - 3.4 Star Trek
  - 3.5 Perry Rhodan
  - 3.6 I, Robot, 2004 Film
  - 3.7 Bicentennial Man
  - 3.8 Bulk Rogers in the 25th Century
  - 3.9 Mystery Science Theater 2000
  - 3.10 Spectreman
  - 3.11 Star Wars
- References
- External links

**Conceptual overview** [ edit ]

Asimov remained vague about the technical details of positronic brains except to assert that their substructure was formed from an alloy of platinum and indium. They were said to be vulnerable to radiation and apparently involve a type of *volatile memory* (since robots in storage required a power source keeping their brains "alive"). The focus of Asimov's stories was directed more towards the *software* of robots—such as the Three Laws of Robotics—than the hardware in which it was implemented, although it is stated in his stories that to create a positronic brain without the Three Laws, it would have been necessary to spend years redesigning the fundamental approach towards the brain itself.

Within his stories of *robotics on Earth* and their development by U.S. Robots, Asimov's positronic brain is less of a *plot device* and more of a technological item worthy of study.

A positronic brain cannot ordinarily be built without incorporating the Three Laws; any modification thereof would drastically modify robot behavior. Behavioral dilemmas resulting from conflicting potentials set by inexperienced and/or malicious users of the robot for the Three Laws make up the bulk of Asimov's stories concerning robots. They are resolved by applying the science of logic and psychology together with *mathematics*, the supreme solution finder being Dr. Susan Calvin, Chief Roboticspsychologist at U.S. Robots.

The Three Laws are also a *bottleneck* in brain sophistication. Very complex brains designed to handle world economy interpret the First Law in expanded sense to include humanity as opposed to a single human; in Asimov's later works like *Robots and Empire* this is referred to as the "Zeroth Law". At least one brain constructed as a calculating *machine*, as opposed to being a robot control circuit, was designed to have a flexible, childlike personality so that it was able to pursue difficult problems without the Three Laws inhibiting it completely. Specialized brains created for overseeing world economics were stated to have no personality at all.

Under specific conditions, the Three Laws can be obviated, with the modification of the actual robotic design.

- Robots that are of low enough value can have the **Third Law** deleted; they do not have to protect themselves from harm, and the brain size can be reduced by half.
- Robots that do not require orders from a human being may have the **Second Law** deleted, and therefore require smaller brains again, providing they do not require the Third Law.
- Robots that are disposable, cannot receive orders from a human being and are not able to harm a human, will not require even the **First Law**. The sophistication of positronic circuitry renders a brain so small that it could comfortably fit within the skull of an insect.

Robots of the latter type directly parallel contemporary industrial robotics practice, though real-life robots do contain safety sensors and systems, in a concern for human safety (a weak form of the First Law; the robot is a safe tool to use, but has no "judgment", which is implicit in Asimov's own stories).

**In Allen's trilogy** [ edit ]

Several robot stories have been written by other authors following Asimov's death. For example, in Roger MacBride Allen's *Caliban* trilogy, a Spacer robotist called Gubber Anshaw invents the **gravitronic brain**. It offers speed and capacity improvements over traditional positronic designs, but the strong influence of tradition make robotics fans reject Anshaw's work. Only one robotist, Freddie Leving, chooses to adopt gravitronics, because it offers her a blank slate on which she could explore alternatives to the Three Laws. Because they are not dependent upon centuries of earlier research, gravitronic brains can be programmed with the standard Laws, variations of the Laws, or even empty pathways which specify no Laws at all.

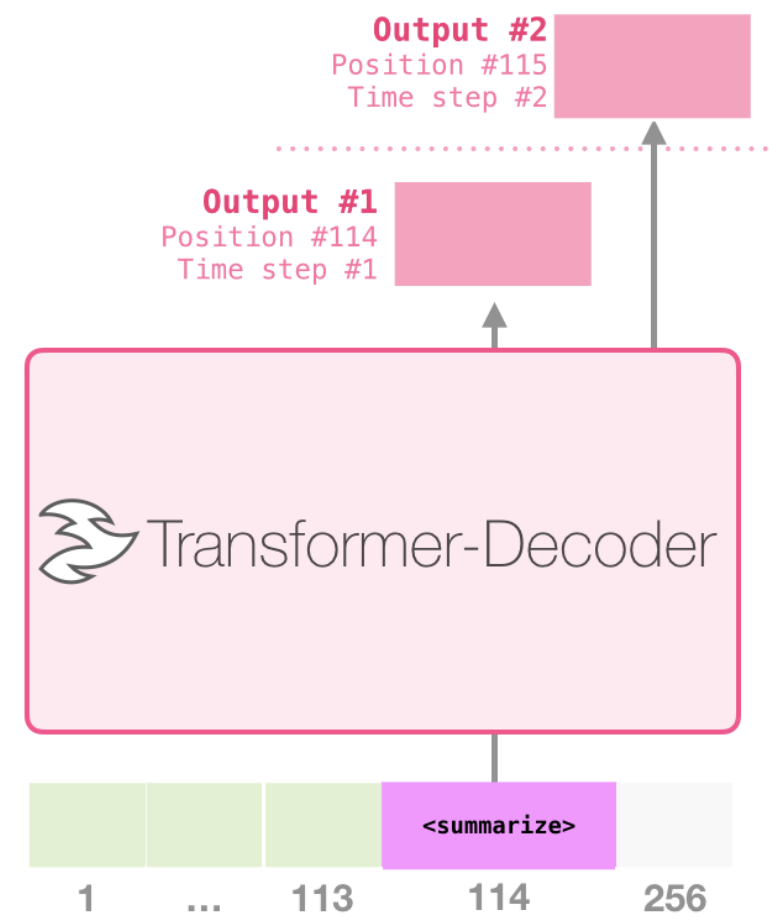
**ARTICLE**



# Summarization with GPT-2

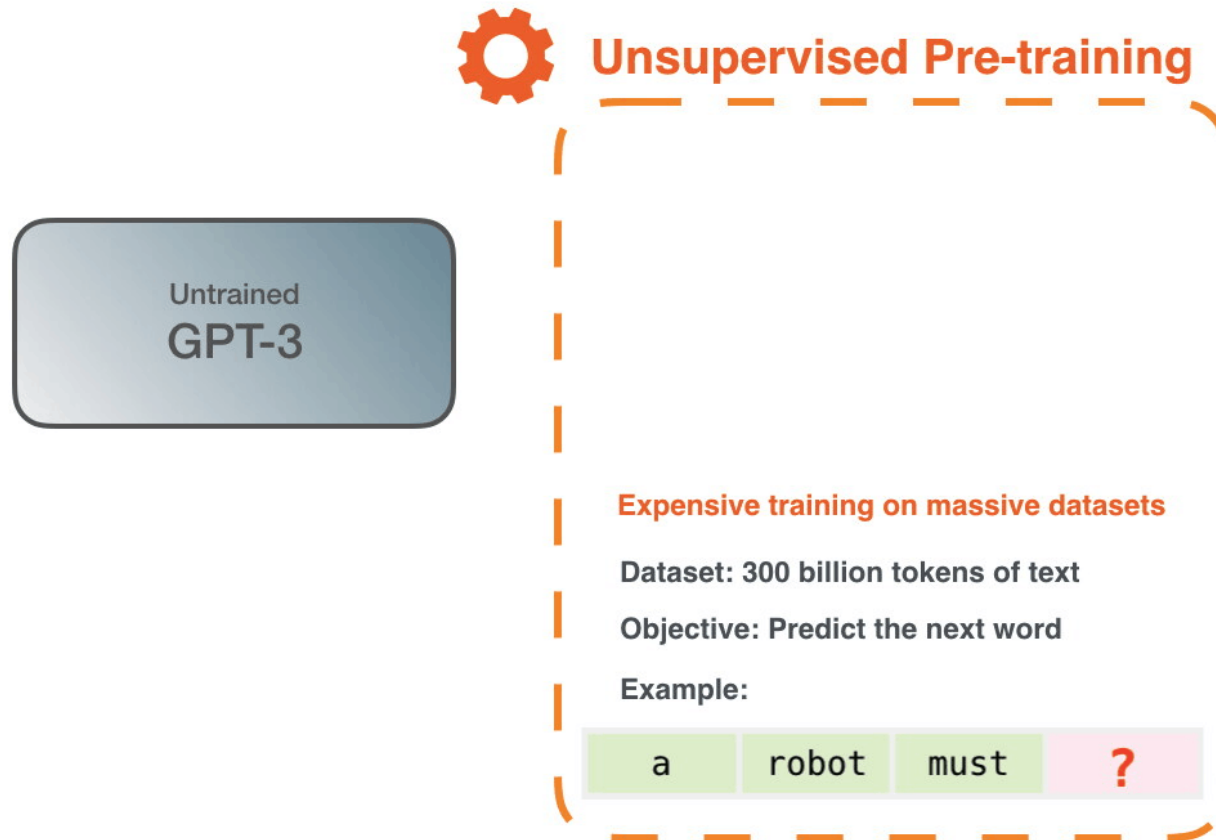
## Training Dataset

Article #1 tokens	<summarize>	Article #1 Summary
Article #2 tokens	<summarize>	Article #2 Summary
Article #3 tokens	<summarize>	Article #3 Summary





# GPT-3





# GPT-3 Specifics

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

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

# GPT-3 Code Generation

[example] an input that says "search" [toCode] Class App extends React Component... </div> } } }

[example] a button that says "I'm feeling lucky" [toCode] Class App extends React Component...

[example] an input that says "enter a todo" [toCode]





# LLM Research

- BIG-Bench – Beyond the Imitation Game Benchmark
- [https://github.com/google/BIG-bench/blob/main/bigbench/benchmark\\_tasks/README.md](https://github.com/google/BIG-bench/blob/main/bigbench/benchmark_tasks/README.md)



# Outline for Week 01

- 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, LoRA, Context Length and issues
  - Commercial and open source LLMs
  - What are the main issues in LLMs to be aware of?
  - Taxonomy of interaction with LLMs
  - Finetuning, Adapters, Quantization
  - Prompting Strategies



# 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](https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard)



Department of Computational and Data Sciences

# Challenges with LLM



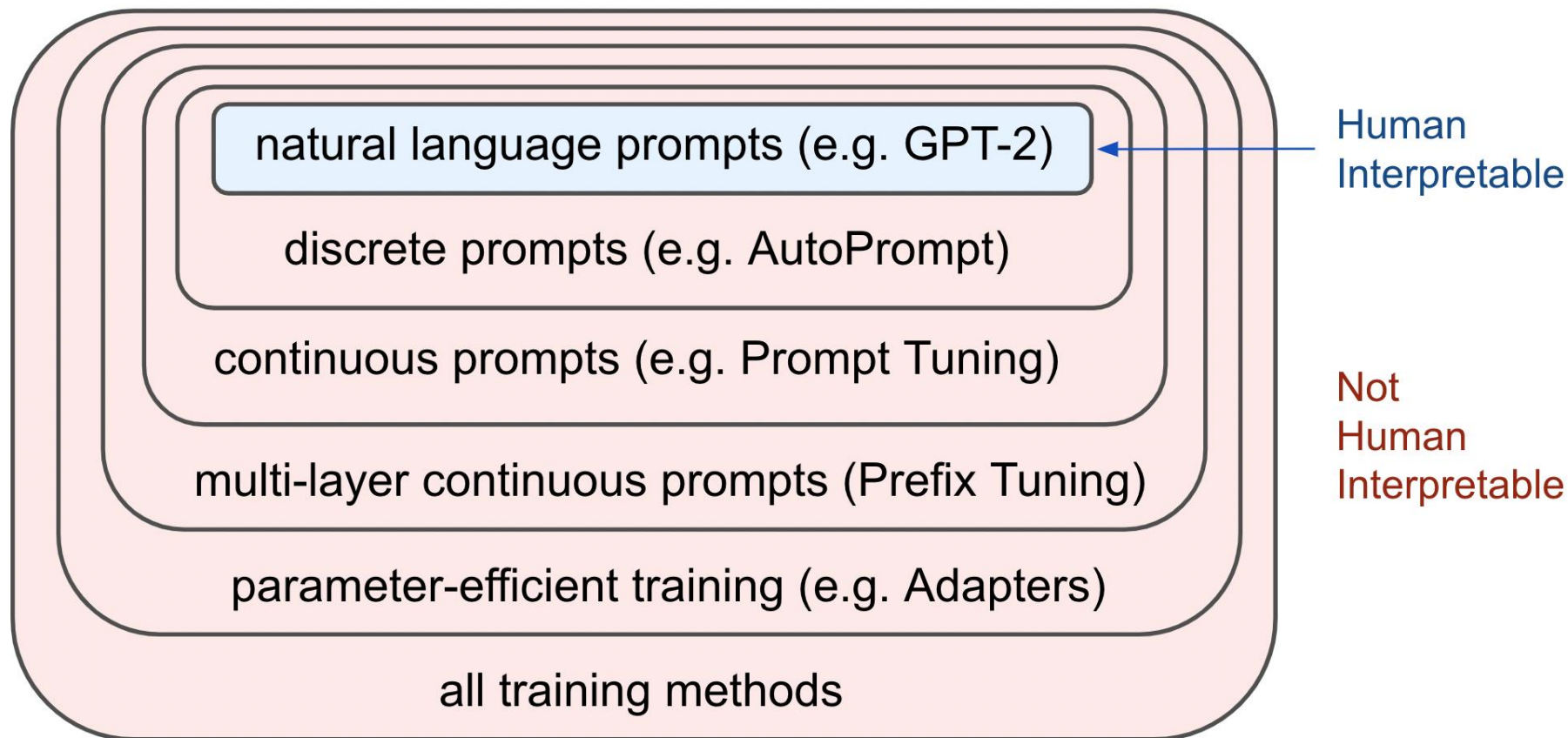
- Size
- Cost
- Out of date facts
- Hallucination
- Harmful content

# 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



Department of Computational and Data Sciences

# Using LLM for your task and Data



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



# Parameter Efficient Fine Tuning

Prompt modifications

“Hard” prompt tuning

“Soft” prompt tuning

Prefix-tuning — LLaMA-Adapter

Adapter methods

Adapters

Reparameterization

Low rank adaptation (LoRA)

# Hard Prompt Tuning

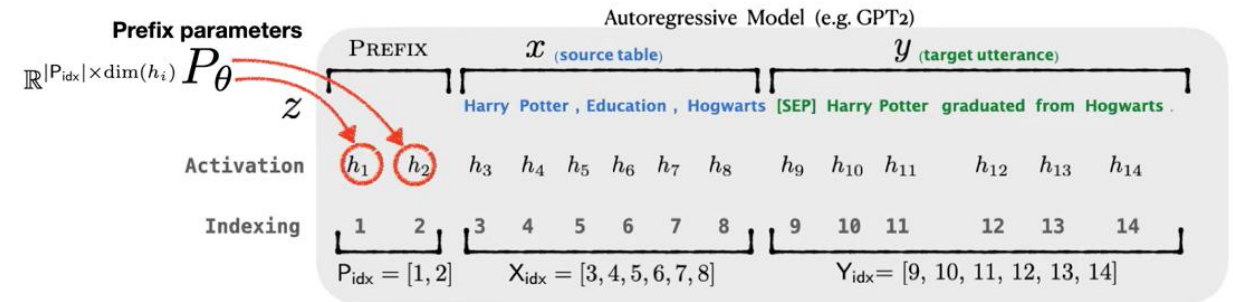
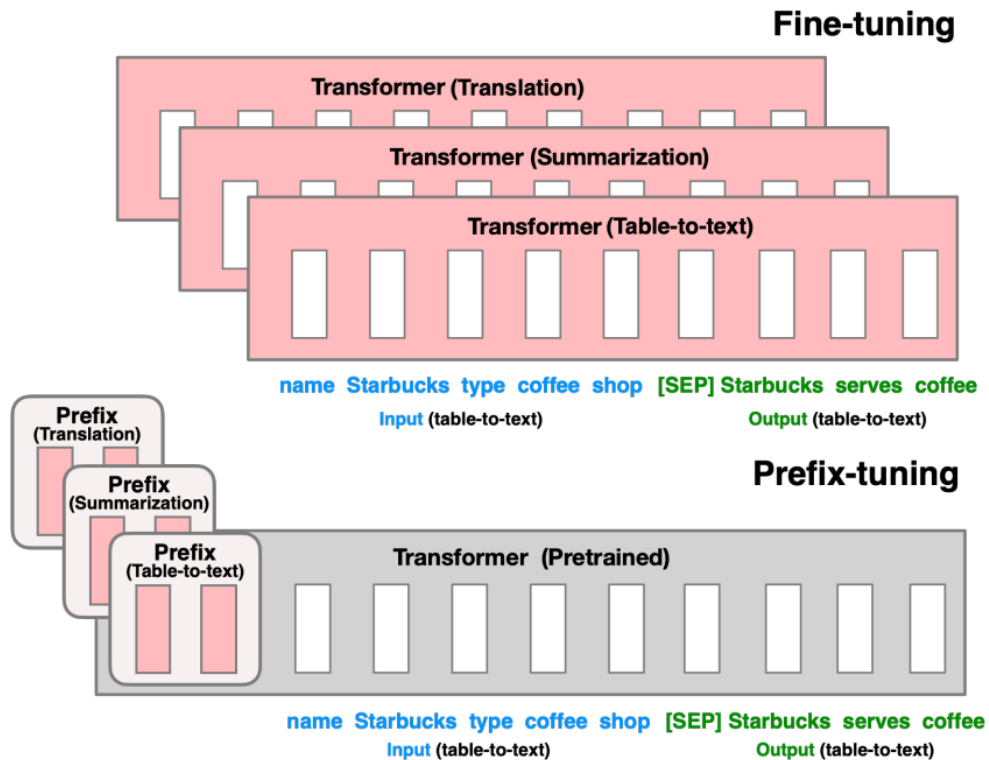


```
1 1) "Translate the English sentence '{english_sentence}' into German: {german_translation}"
2
3 2) "English: '{english_sentence}' | German: {german_translation}"
4
5 3) "From English to German: '{english_sentence}' -> {german_translation}"
```



# Prefix Tuning

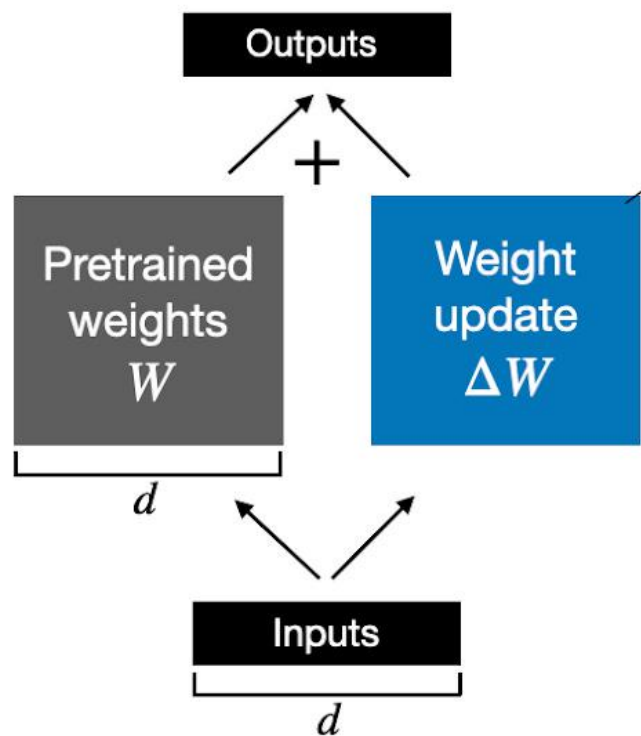
- Add prefix parameters that are learnt during the training of GPT





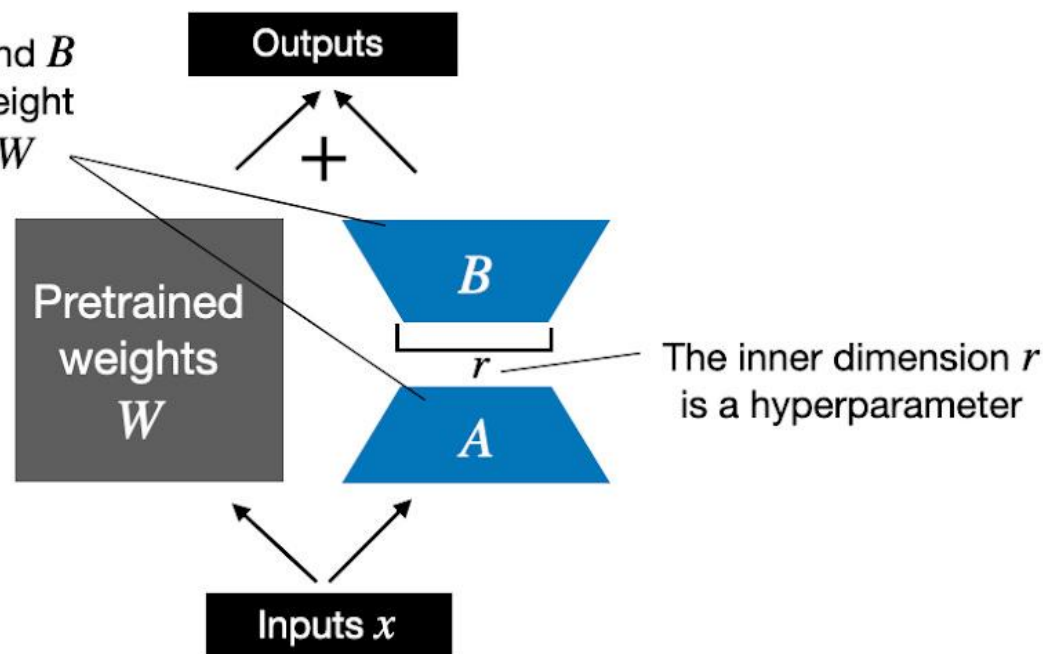
# LoRA – Low Rank Adaptation

## Weight update in regular finetuning



LoRA matrices  $A$  and  $B$  approximate the weight update matrix  $\Delta W$

## Weight update in LoRA



# 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 LLM Ops



# Floating Point Sizes

## Floating Point Formats

bfloat16: Brain Floating Point Format

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



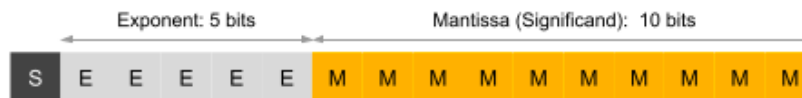
fp32: Single-precision IEEE Floating Point Format

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



fp16: Half-precision IEEE Floating Point Format

Range:  $\sim 5.96e^{-8}$  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>