Large Language Models (LLMs)

Naeemullah Khan
naeemullah.khan@kaust.edu.sa

July 3, 2025

Large Language Models

BCV

Application Layer

Copywriting
Jasper
copy.ai
Headlime
co:here
HyperWrite
Writesonic
Contenda
unbounce
copysmith

Coding
tabnine
MUTABLEAI
Codiga
co:here
GitHub Copilot
CODEGEN

Dev Tools
algolia
warp
Mintlify
cogram
Debuild
repl.it

Chat / Comms
MessageBird
Replier.ai
Sapling
FABLE

BizOps
viable
Enterpret
tabulate
Anecdote
OTHERSIDE AI
casetext
Dover

Infrastructure Layer

Model Creation
AI21labs
OpenAI
NVIDIA
Adept
EleutherAI
ANTHROPIC
Google AI

Hardware
habana
NVIDIA
SambaNova SYSTEMS
cerebras
GRAPHCORE

Fine Tuning
Google AI
aws
OpenAI
Hugging Face

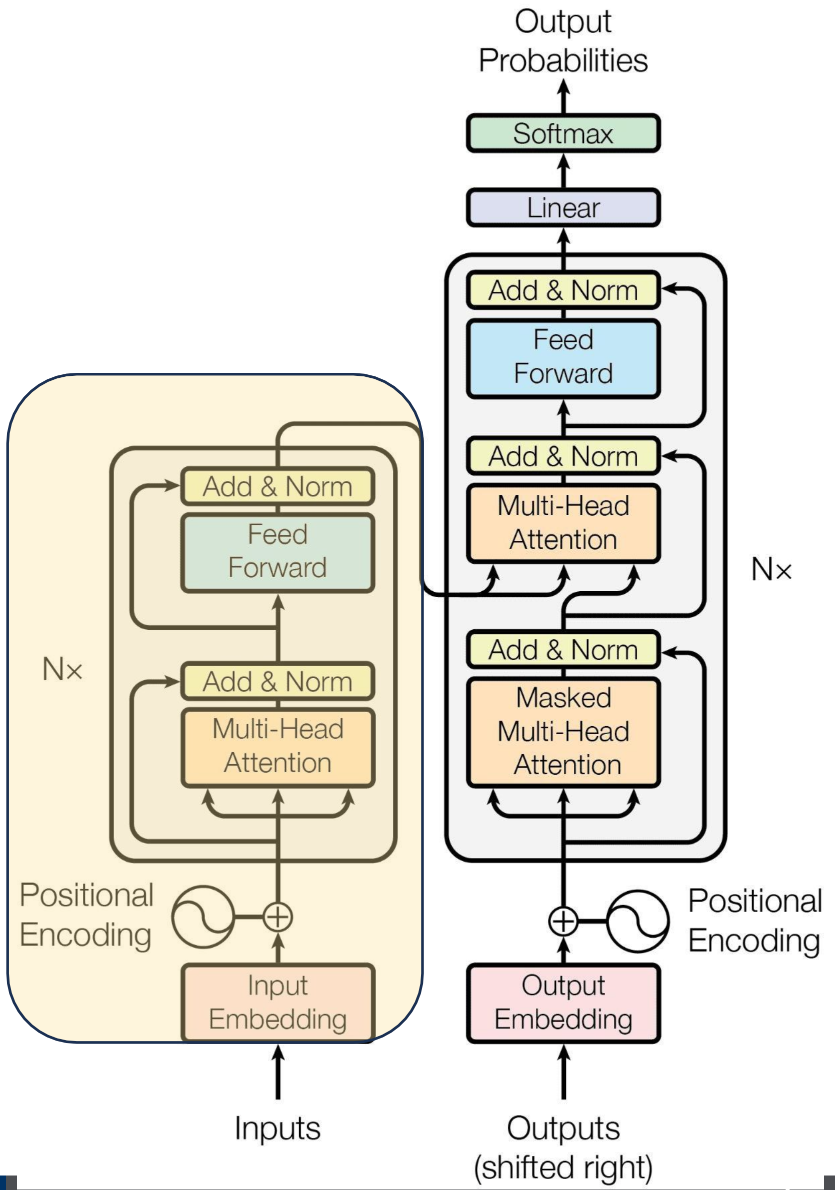
Inference
OpenAI
Hugging Face

BERT and GPT Concepts

- **BERT (Bidirectional Encoder Representations from Transformers)**
  - Uses transformer encoder only.
  - Trained with Masked Language Modeling (MLM).
  - **Bidirectional:** Considers both left and right context.
  - Fine-tuned for tasks like QA, classification.
  - **Architecture:**
    - Layers of encoder blocks.
    - Positional encodings added to embeddings.
    - Self-attention heads capture dependencies.

BERT - Bidirectional Encoder **Representations**

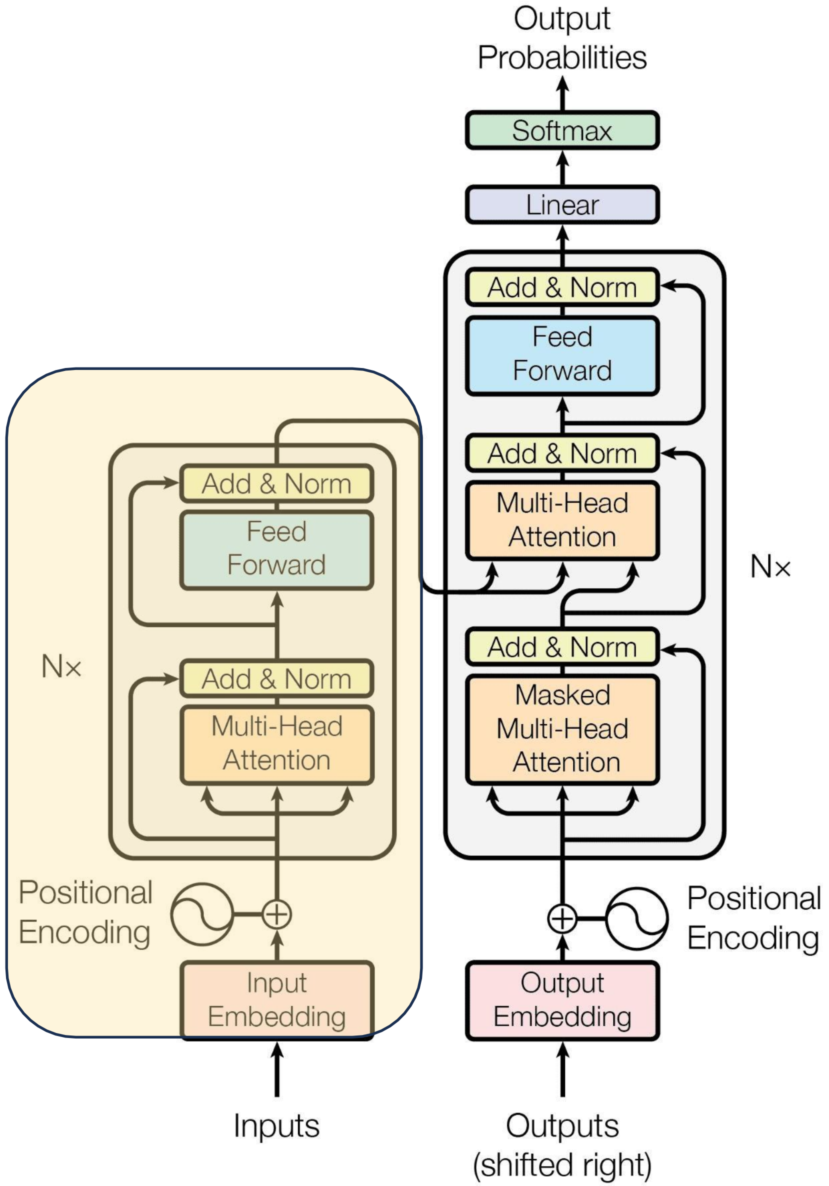
- One of the biggest challenges in LM-building used to be the lack of task-specific training data.
- What if we learn an effective representation that can be applied to a variety of downstream tasks?
  - Word2vec (2013)
  - GloVe (2014)



BERT - Bidirectional Encoder Representations

BERT Pre-Training Corpus:

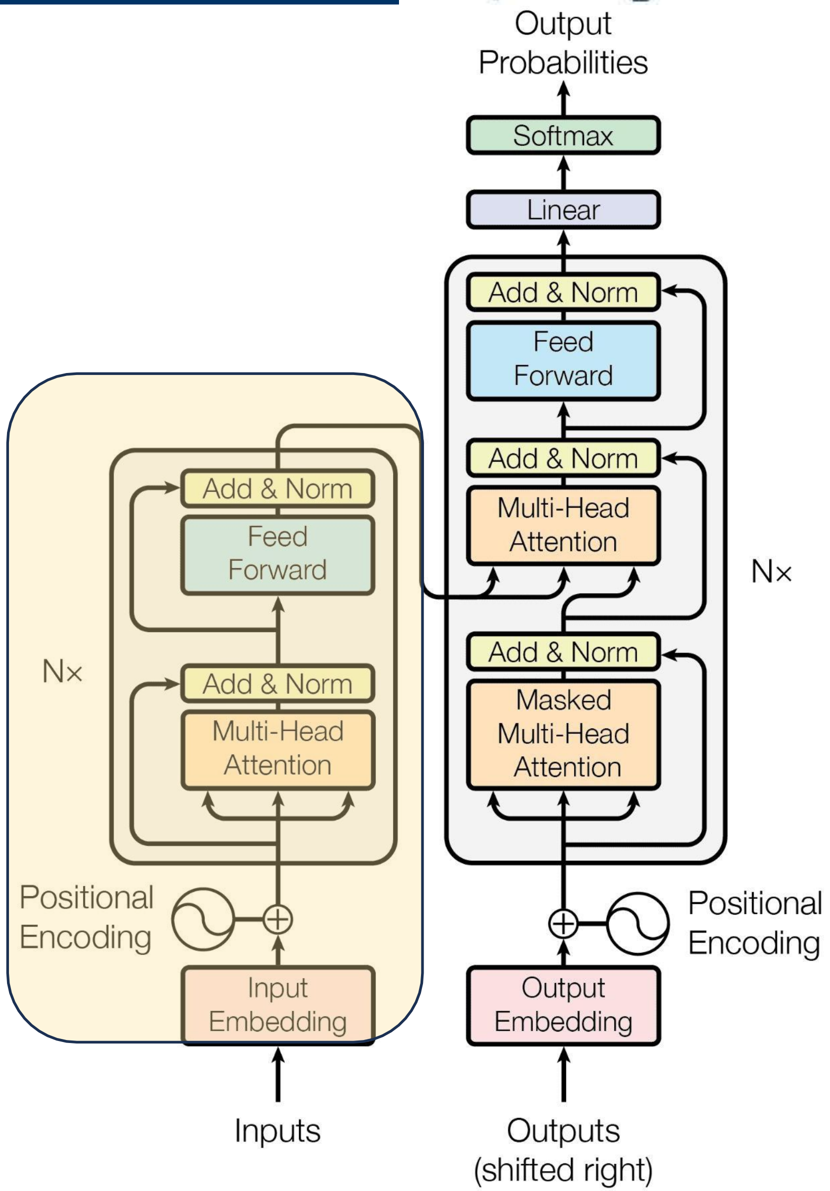
- English Wikipedia - 2,500 million words
- Book Corpus - 800 million words



BERT - Bidirectional Encoder **Representations**

BERT Pre-Training Corpus:
- English Wikipedia - 2,500 million words
- Book Corpus - 800 million words

BERT Pre-Training Tasks:
- MLM (Masked Language Modeling)
- NSP (Next Sentence Prediction)



BERT - Bidirectional Encoder Representations

BERT Pre-Training Corpus:

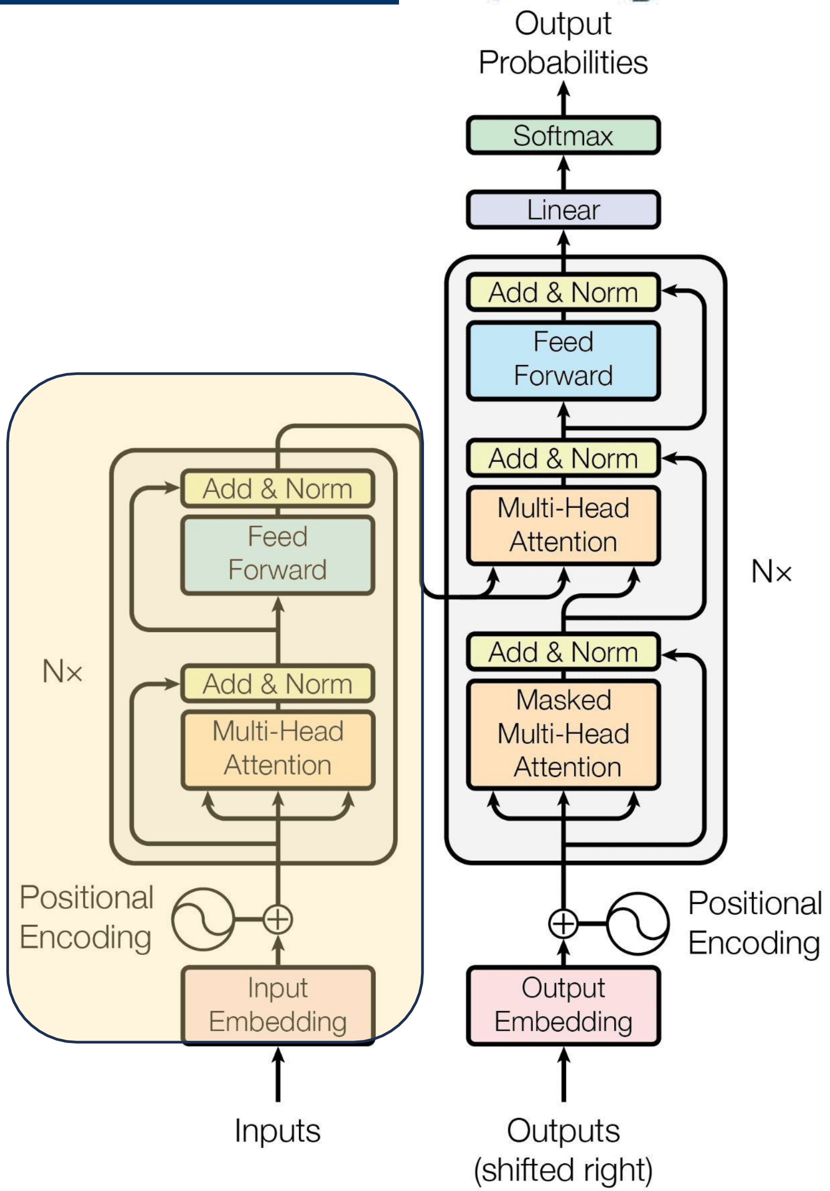
- English Wikipedia - 2,500 million words
- Book Corpus - 800 million words

BERT Pre-Training Tasks:

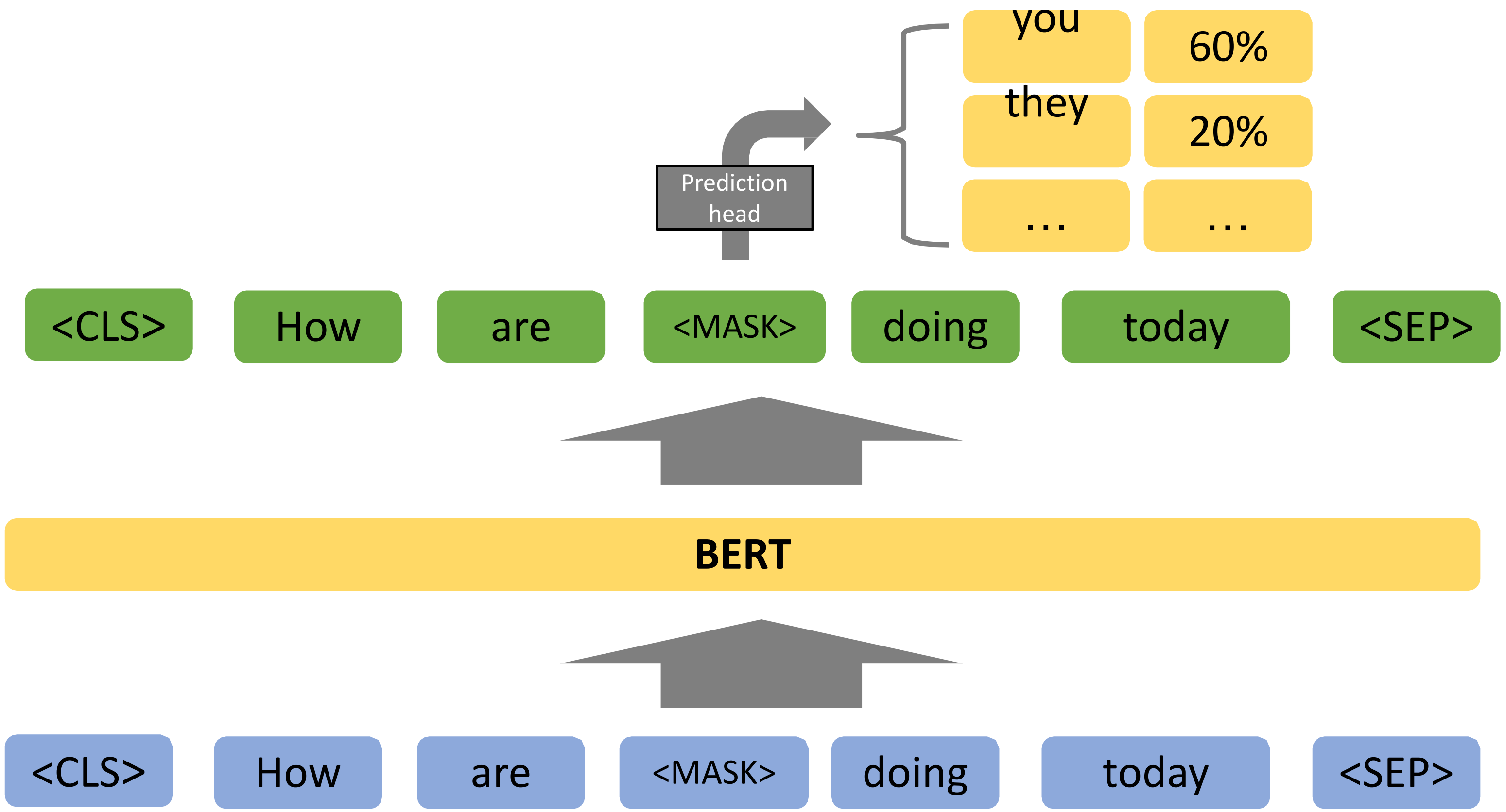
- MLM (Masked Language Modeling)
- NSP (Next Sentence Prediction)

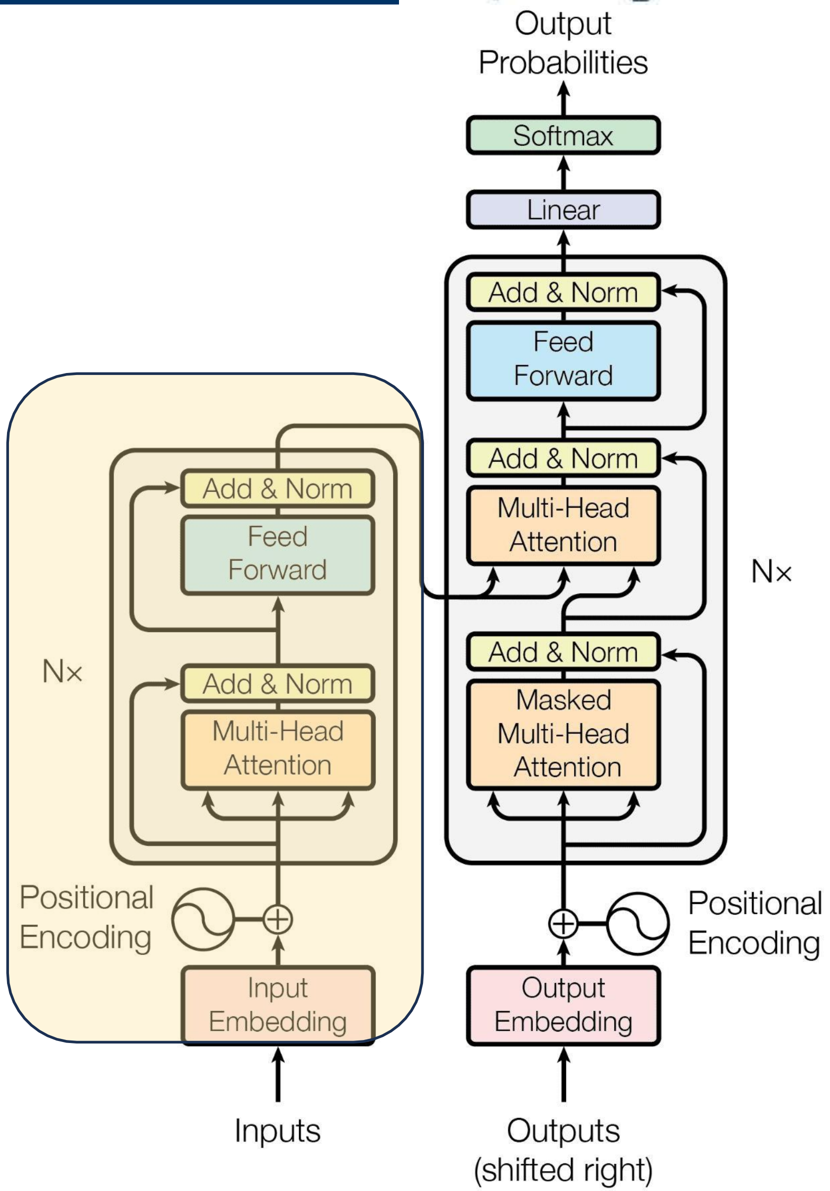
BERT Pre-Training Results:

- BERT-Base – 110M Params
- BERT-Large – 340M Params

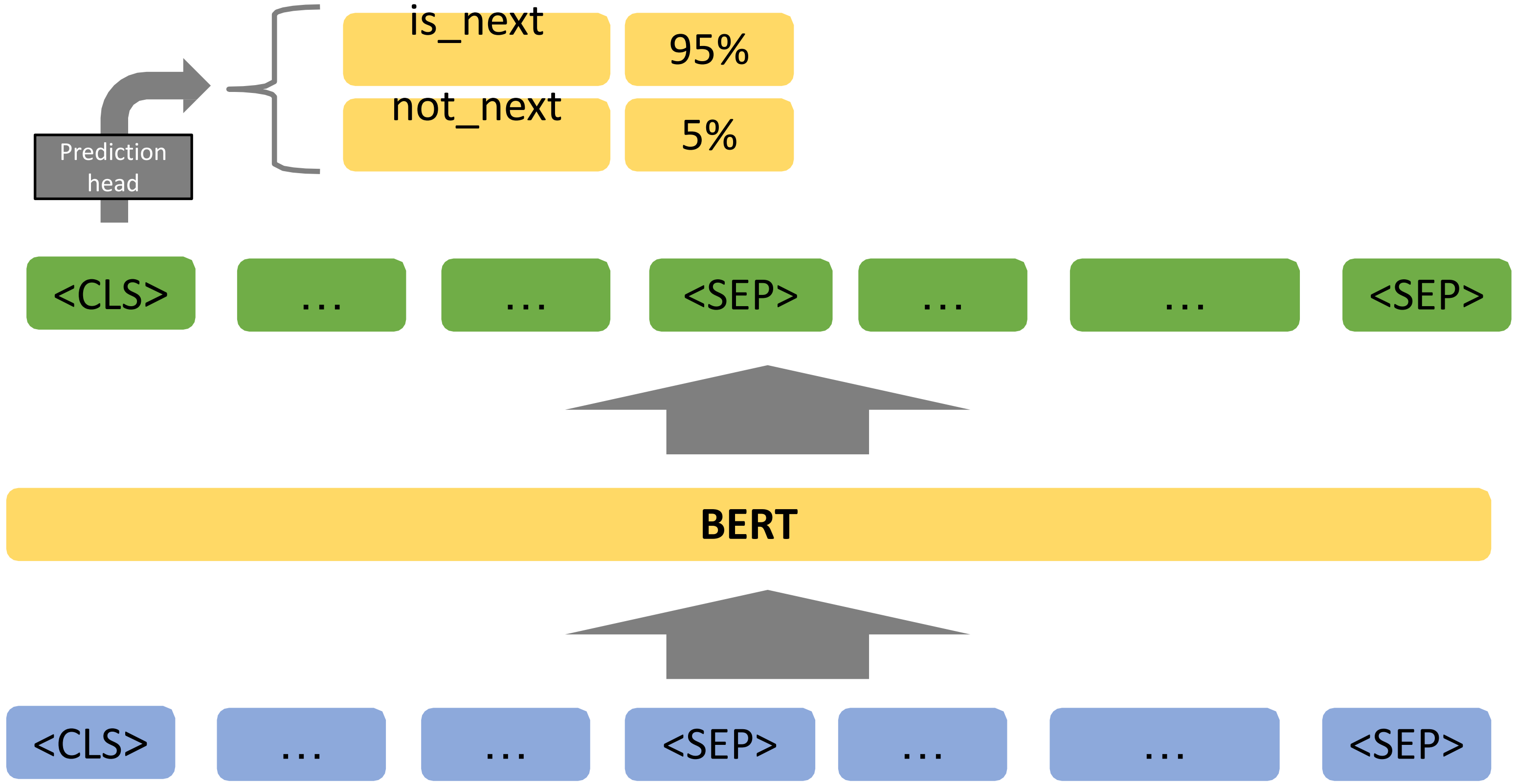


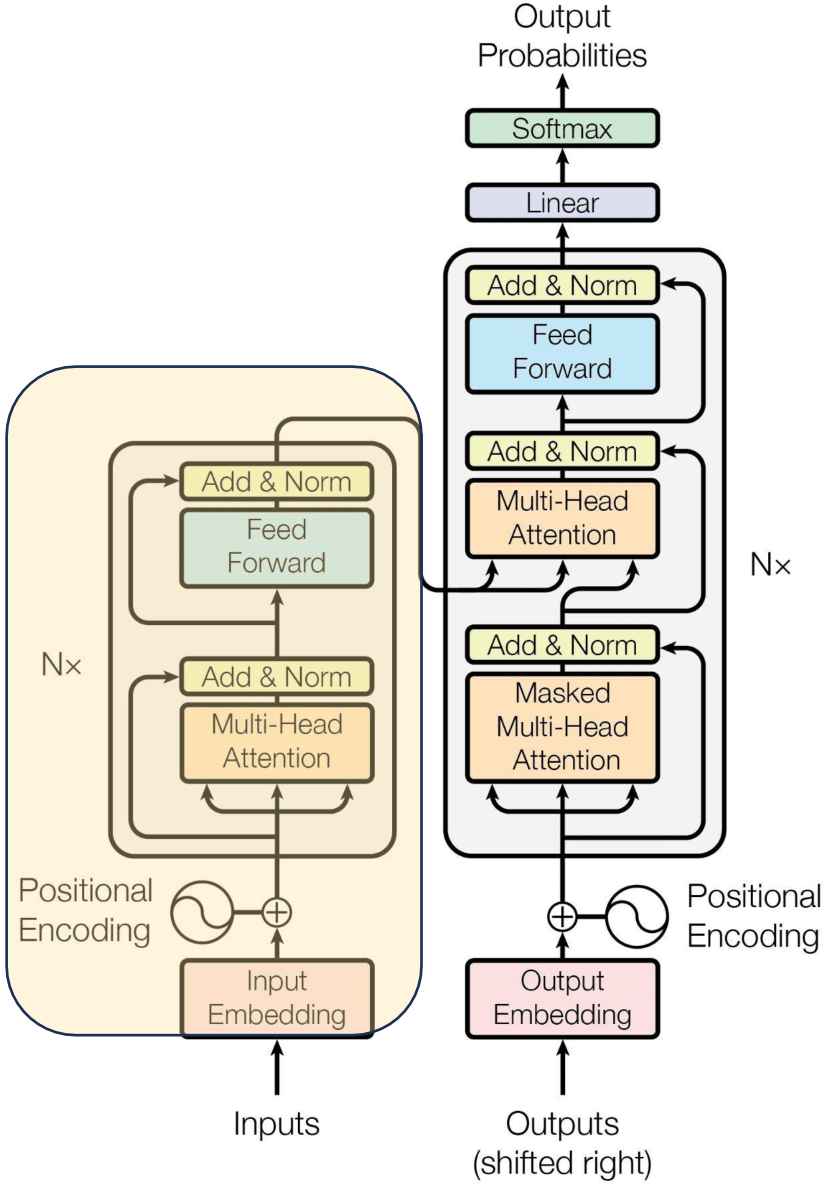
MLM (Masked Language Modeling)





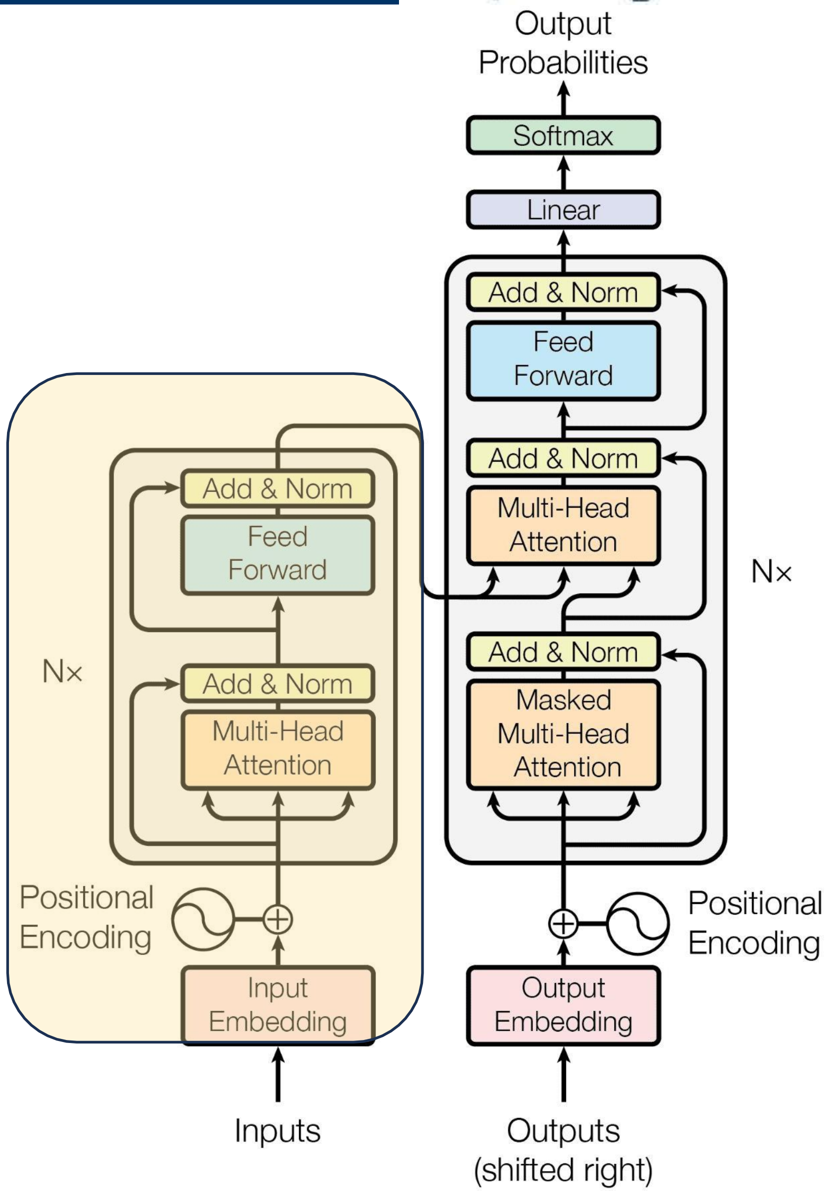
NSP (Next Sentence Prediction)





BERT Fine-Tuning:

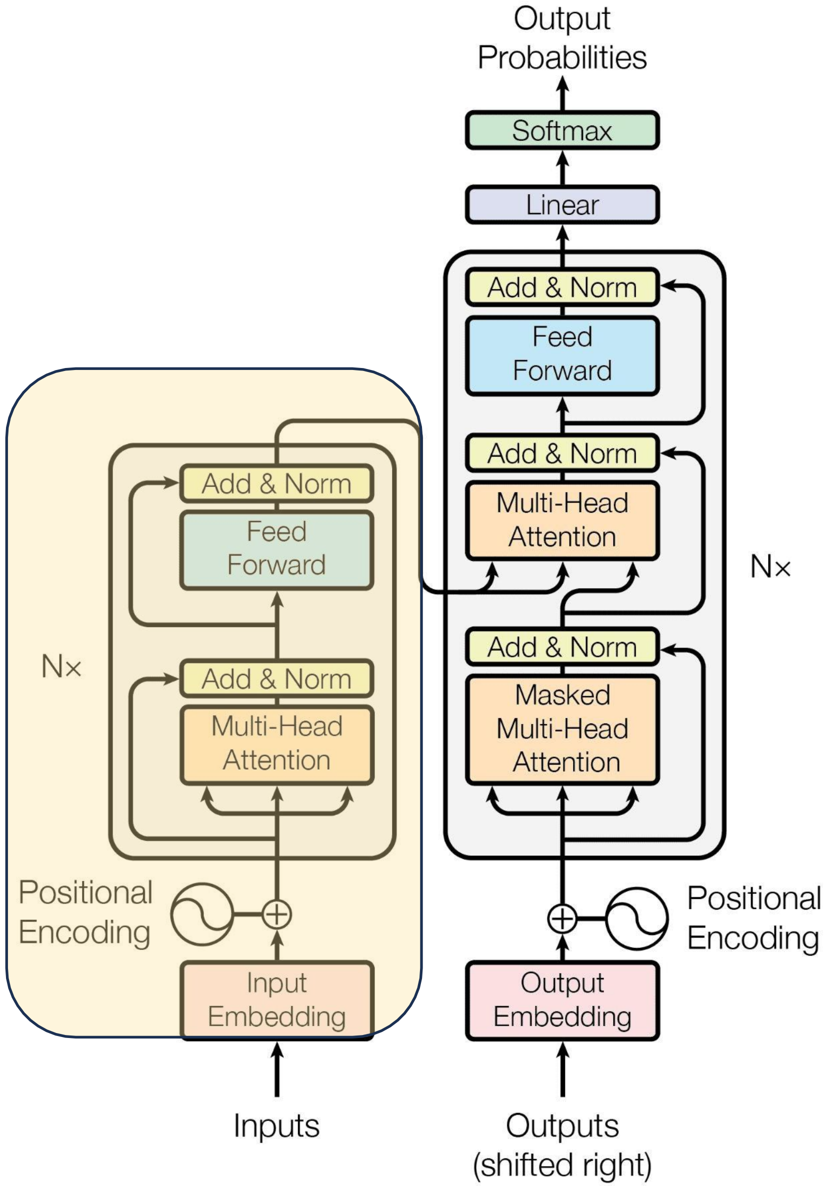
- Simply add a task-specific module after the last encoder layer to map it to the desired dimension.
  - *Classification Tasks:*
    - *Add a feed-forward layer on top of the encoder output for the [CLS] token*
  - *Question Answering Tasks:*
    - *Train two extra vectors to mark the beginning and end of answer from paragraph*
  - *…*



BERT Evaluation:

- General Language Understanding Evaluation (GLUE)
  - Sentence pair tasks
  - Single sentence classification

- Stanford Question Answering Dataset (SQuAD)

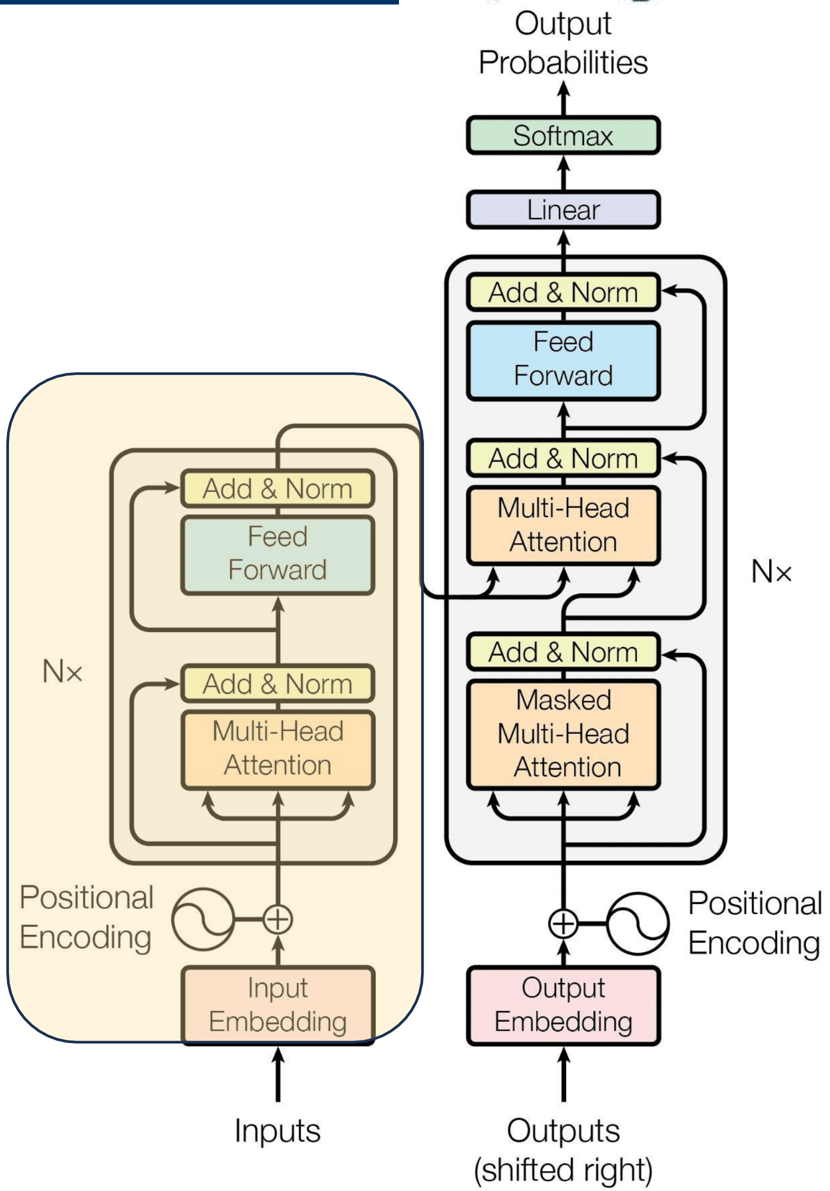


BERT Evaluation:

System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Average
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

System	Dev EM	Dev F1	Test EM	Test F1
Leaderboard (Oct 8th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
#1 Single - nlnet	-	-	83.5	90.1
#2 Single - QANet	-	-	82.5	89.3
Published				
BiDAF+ELMo (Single)	-	85.8	-	-
R.M. Reader (Single)	78.9	86.3	79.5	86.6
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
Ours				
$\text{BERT}_{\text{BASE}}$ (Single)	80.8	88.5	-	-
$\text{BERT}_{\text{LARGE}}$ (Single)	84.1	90.9	-	-
$\text{BERT}_{\text{LARGE}}$ (Ensemble)	85.8	91.8	-	-
$\text{BERT}_{\text{LARGE}}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
$\text{BERT}_{\text{LARGE}}$ (Ens.+TriviaQA)	**86.2**	**92.2**	**87.4**	**93.2**

Table 2: SQuAD results. The BERT ensemble is 7x systems which use different pre-training checkpoints and fine-tuning seeds.



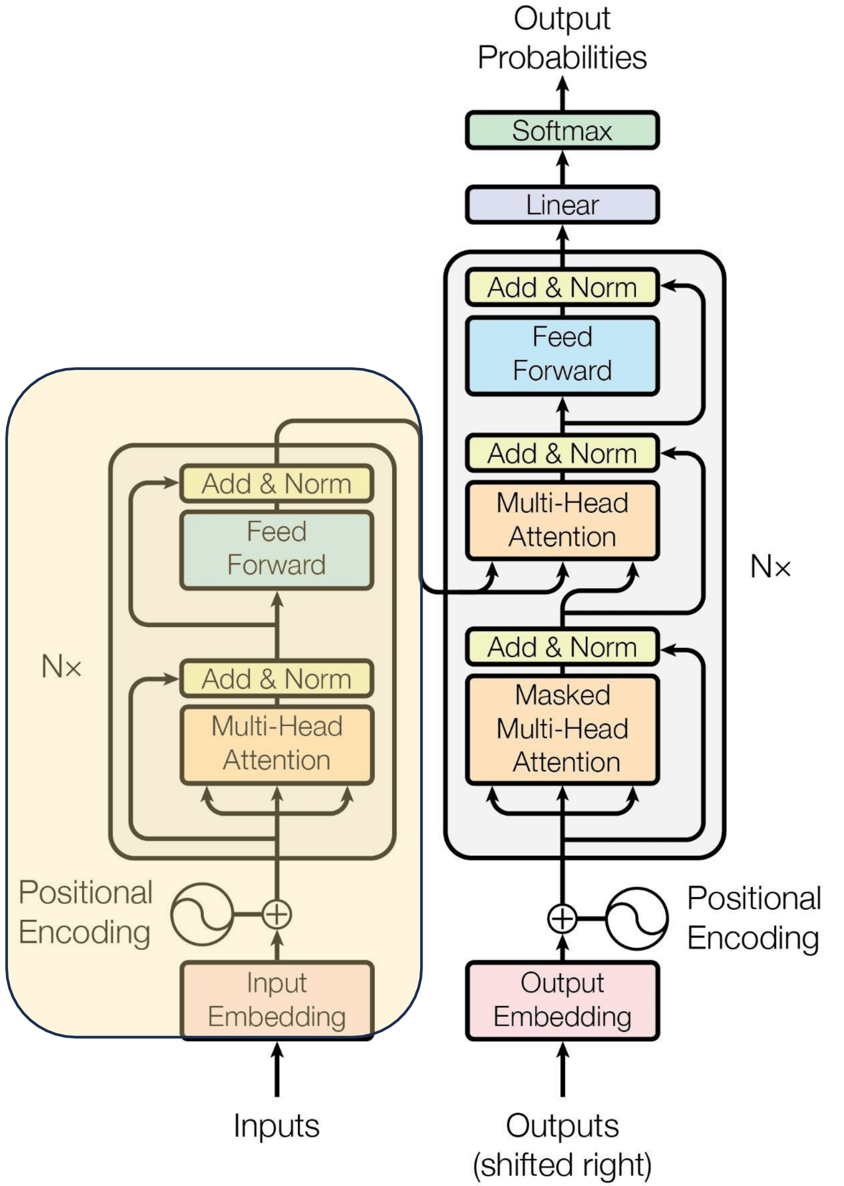
BERT - Bidirectional Encoder **Representations**

أكاديمية كاوست
KAUST ACADEMY

LMH
Lady Margaret Hall

What is our takeaway from BERT?

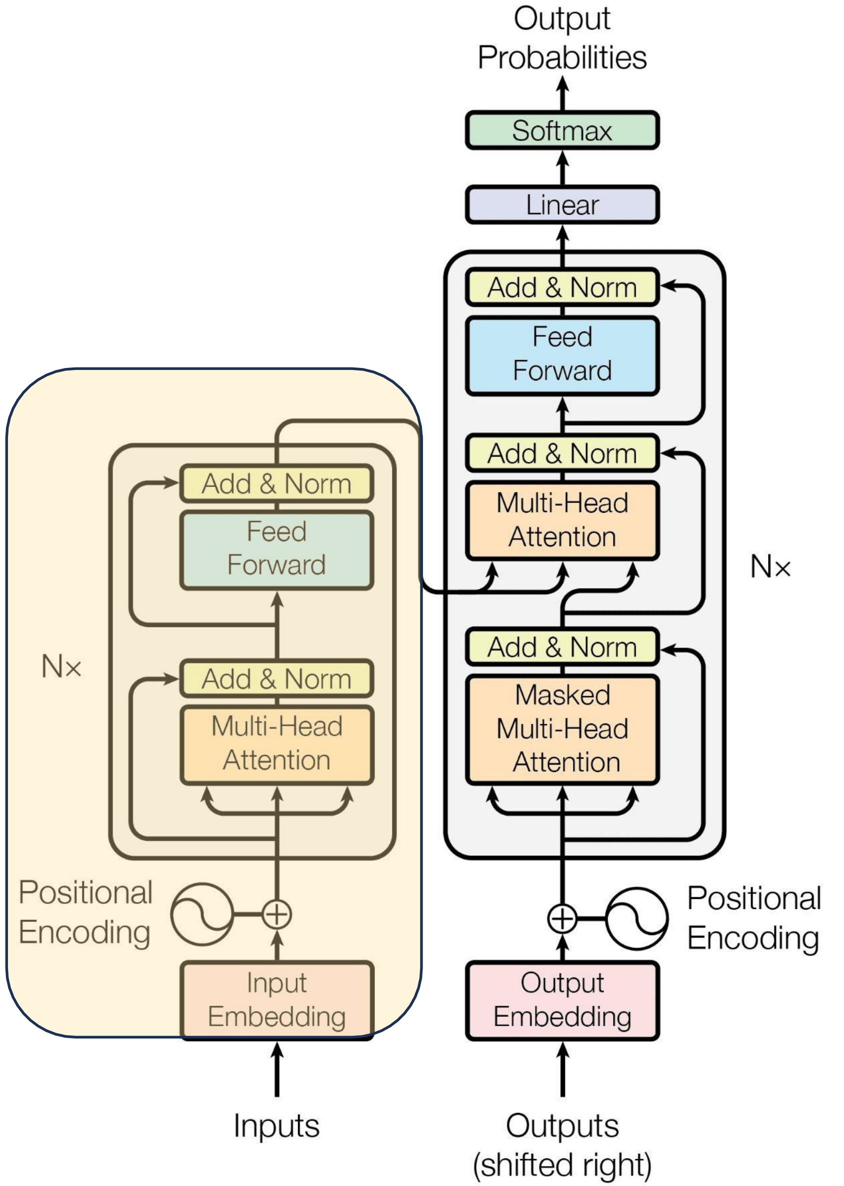
- **Pre-training tasks can be invented flexibly…**
  - Effective representations can be derived from a flexible regime of pre-training tasks.



أكاديمية كاوست
KAUST ACADEMY

Lady Margaret Hall

What is our takeaway from BERT?

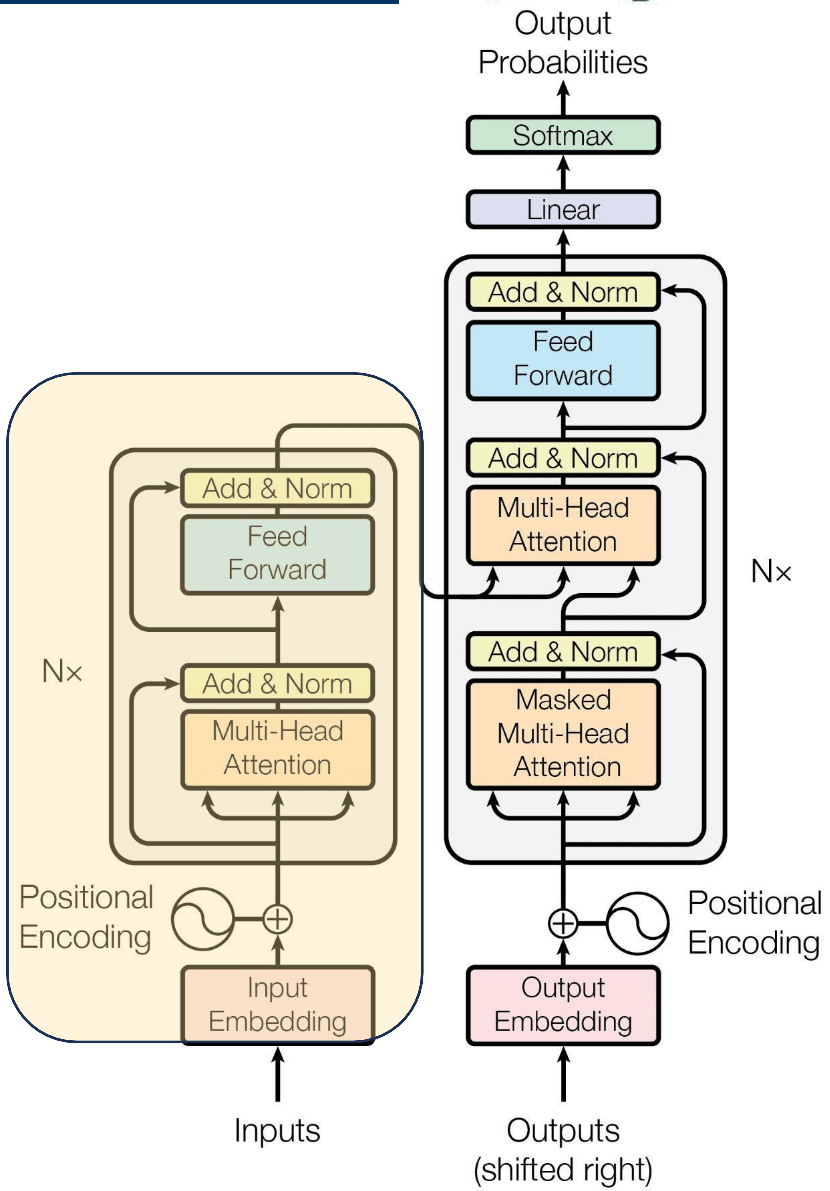


- **Pre-training tasks can be invented flexibly…**
  - Effective representations can be derived from a flexible regime of pre-training tasks.

- **Different NLP tasks seem to be highly transferable with each other...**
  - As long as we have effective representations, that seems to form a general model which can serve as the backbone for many specialized models.

What is our takeaway from BERT?

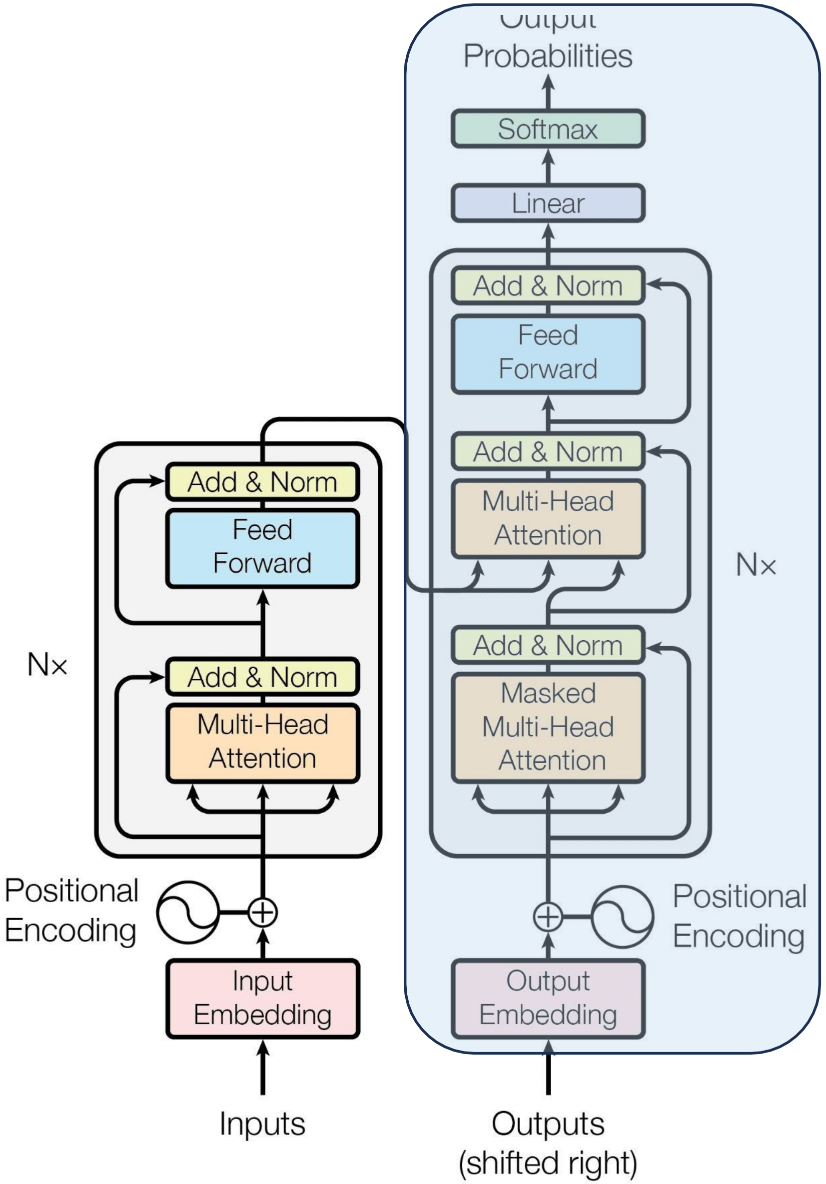
- **Pre-training tasks can be invented flexibly…**
  - Effective representations can be derived from a flexible regime of pre-training tasks.
- **Different NLP tasks seem to be highly transferable with each other...**
  - As long as we have effective representations, that seems to form a general model which can serve as the backbone for many specialized models.
- **And scaling works!!!**
  - 340M was considered large in 2018



- **GPT (Generative Pre-trained Transformer)**
  - Uses transformer decoder only.
  - Trained with next-token prediction.
  - **Unidirectional:** Considers left context only.
  - Fine-tuned for text generation, dialogue systems.
  - **Architecture:**
    - Layers of decoder blocks.
    - Causal masking to prevent future token access.
    - Self-attention heads capture sequential dependencies.

GPT – Generative Pretrained Transformer

- Similarly motivated as BERT, though differently designed
  - Can we leverage large amounts of unlabeled data to pretrain an LM that understands general patterns?

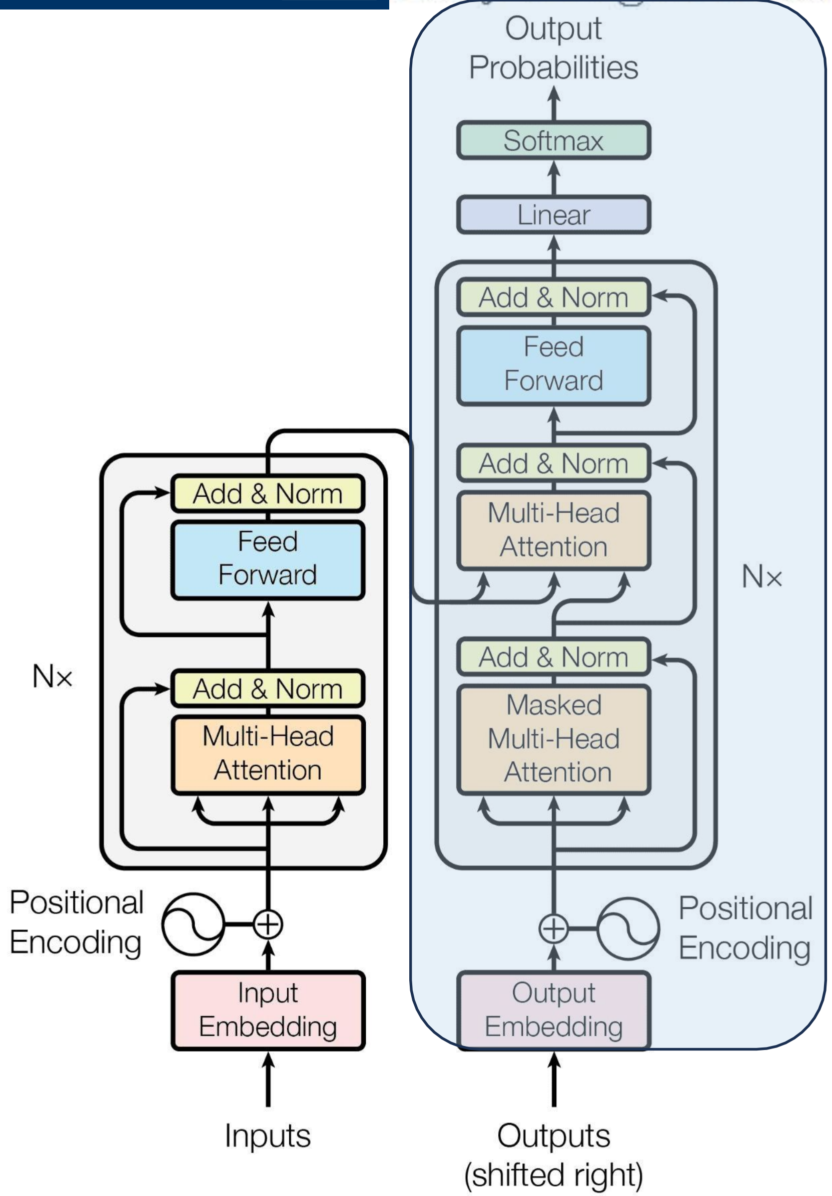


GPT – Generative Pretrained Transformer

GPT Pre-Training Corpus:
- Similarly, BooksCorpus and English Wikipedia

GPT Pre-Training Tasks:
- Predict the next token, given the previous tokens
  - More learning signals than MLM

GPT Pre-Training Results:
- GPT – 117M Params
  - Similarly competitive on GLUE and SQuAD



GPT – Generative Pretrained Transformer

GPT Fine-Tuning:

- Prompt-format task-specific text as a continuous stream for the model to fit

Summarization

Summarize this article:

The summary is:

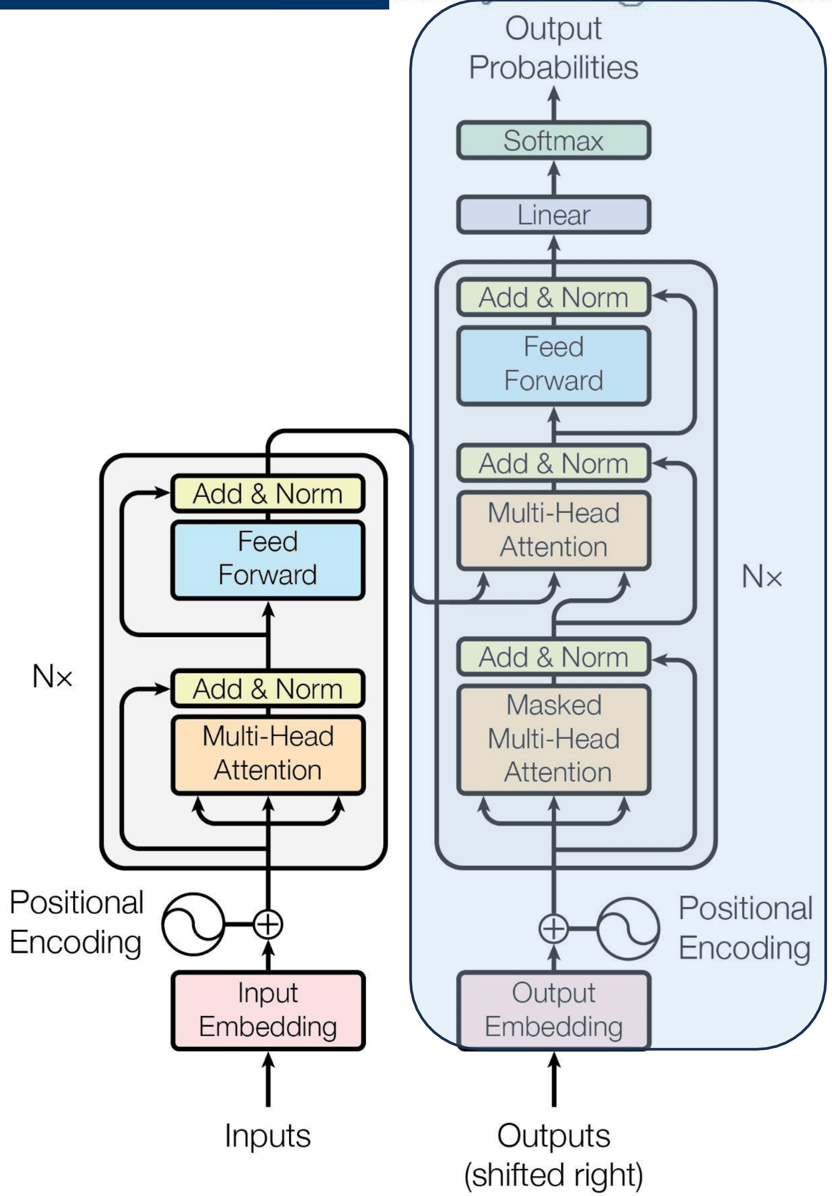
QA

Answer the question based on the context.

Context:

Question:

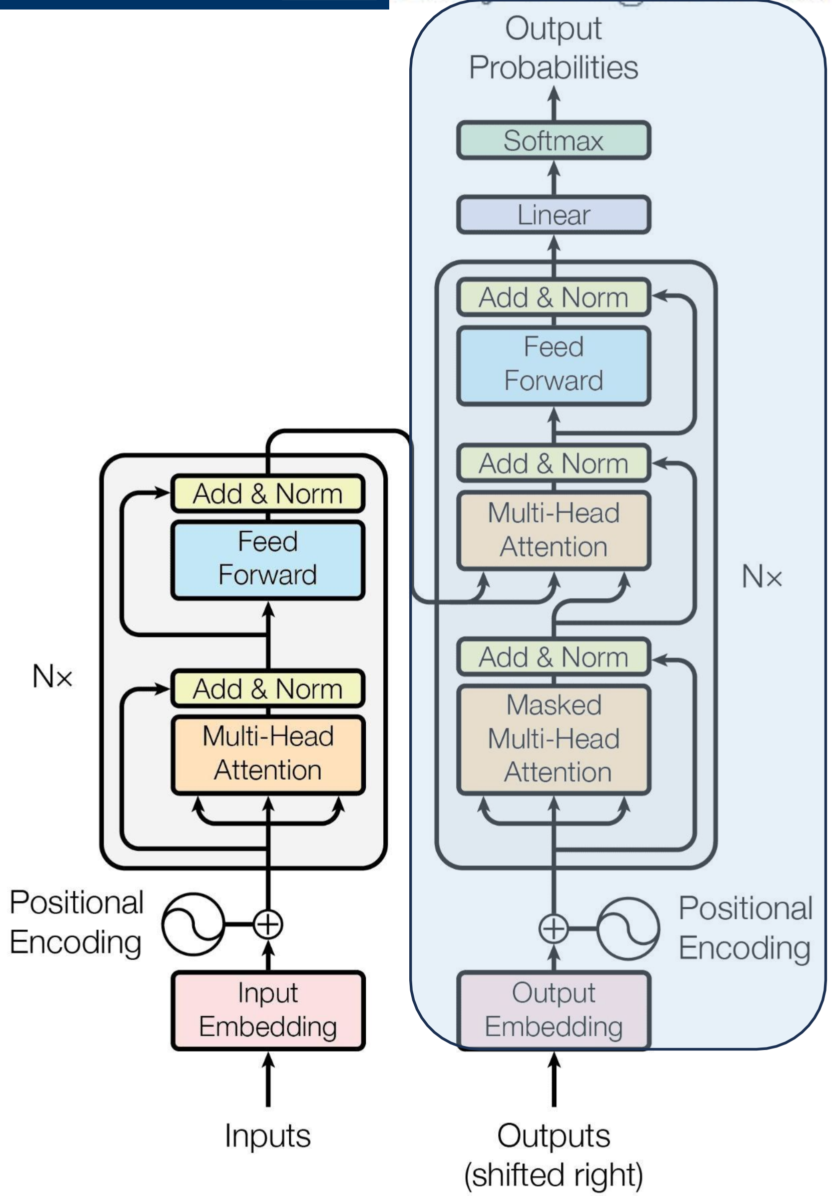
Answer:



GPT – Generative Pretrained Transformer

What is our takeaway from GPT?

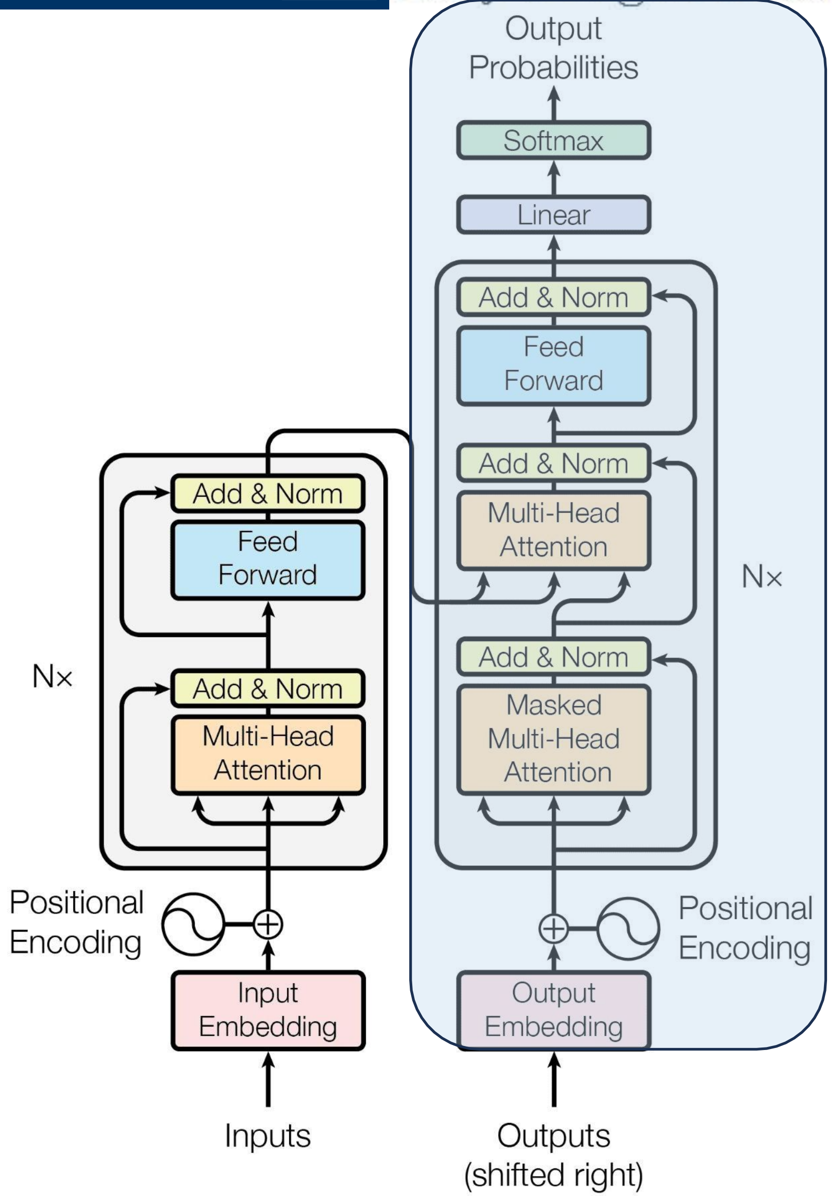
- **The Effectiveness of Self-Supervised Learning**
  - Specifically, the model seems to be able to learn from generating the language *itself*, rather than from any specific task we might cook up.



What is our takeaway from GPT?

- **The Effectiveness of Self-Supervised Learning**
  - Specifically, the model seems to be able to learn from generating the language *itself*, rather than from any specific task we might cook up.

- **Language Model as a Knowledge Base**
  - Specifically, a generatively pretrained model seems to have a decent zero-shot performance on a range of NLP tasks.



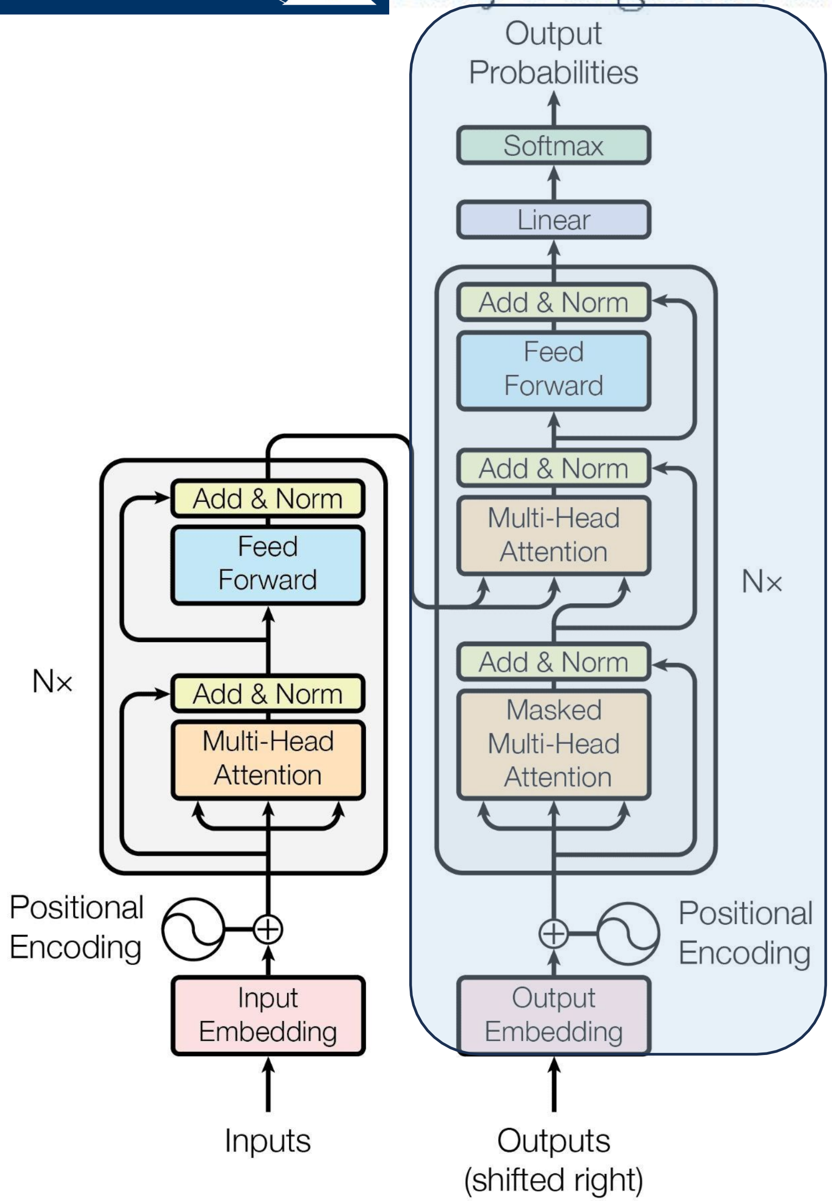
GPT – Generative Pretrained Transformer

What is our takeaway from GPT?

- **The Effectiveness of Self-Supervised Learning**
  - Specifically, the model seems to be able to learn from generating the language *itself*, rather than from any specific task we might cook up.

- **Language Model as a Knowledge Base**
  - Specifically, a generatively pretrained model seems to have a decent zero-shot performance on a range of NLP tasks.

- **And scaling works!!!**



- **Key Differences:**

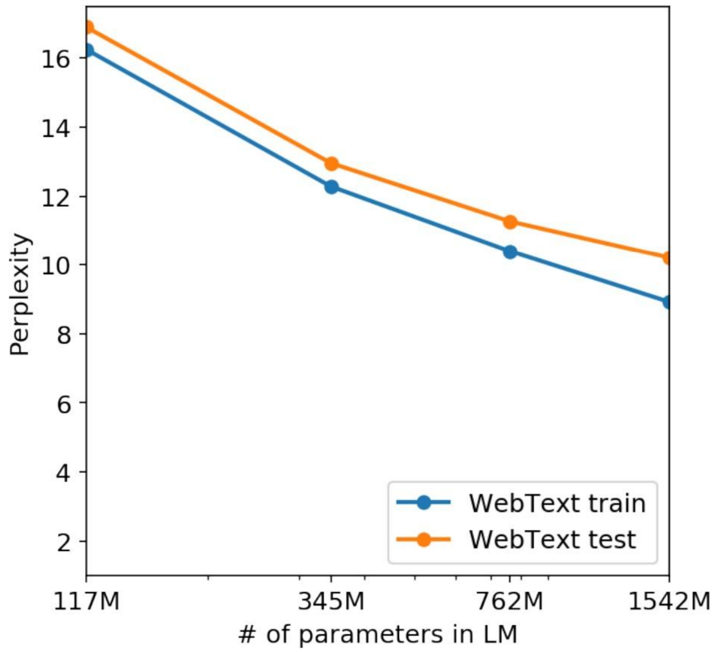
Feature	**BERT**	**GPT**
Directionality	Bidirectional	Unidirectional
Objective	Masked Language Modeling (MLM)	Causal Language Modeling (CLM)
Output	Contextual embeddings	Text generation
Usage	Downstream tasks (e.g., classification, QA)	Generation, few-shot learning

Scaling Laws for LLMs

- **Kaplan et al. (2020):** "Scaling Laws for Neural Language Models"
- Performance improves predictably with:
  - More parameters
  - More compute
  - Larger datasets
- **Optimal allocation of compute:** Train bigger models with less data, rather than small models with lots of data.
- **Implication:** LLMs like GPT-3 (175B), GPT-4 (est. >500B) are products of scaling laws.

Scaling in GPT-2

- Scaling improves the perplexity of the LM and improves performance

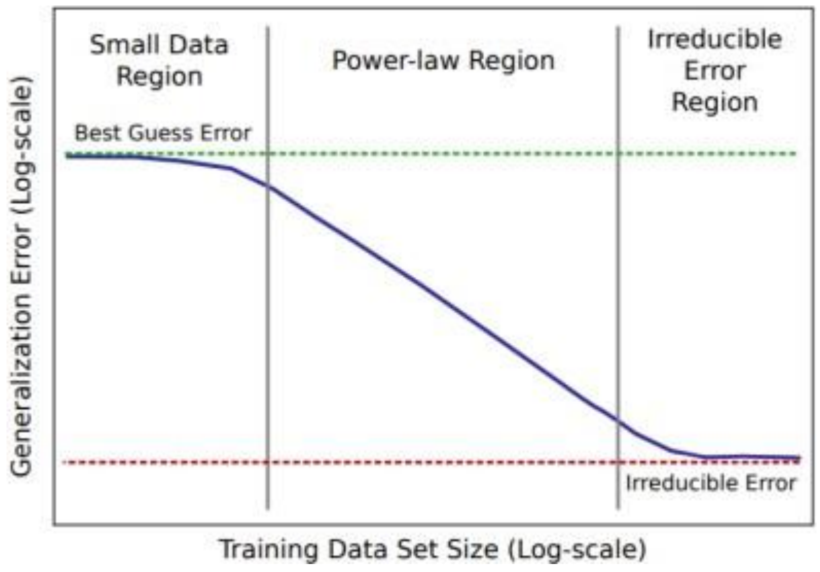


Why is this interesting? Look at data scaling

أكاديمية كاوست
KAUST ACADEMY

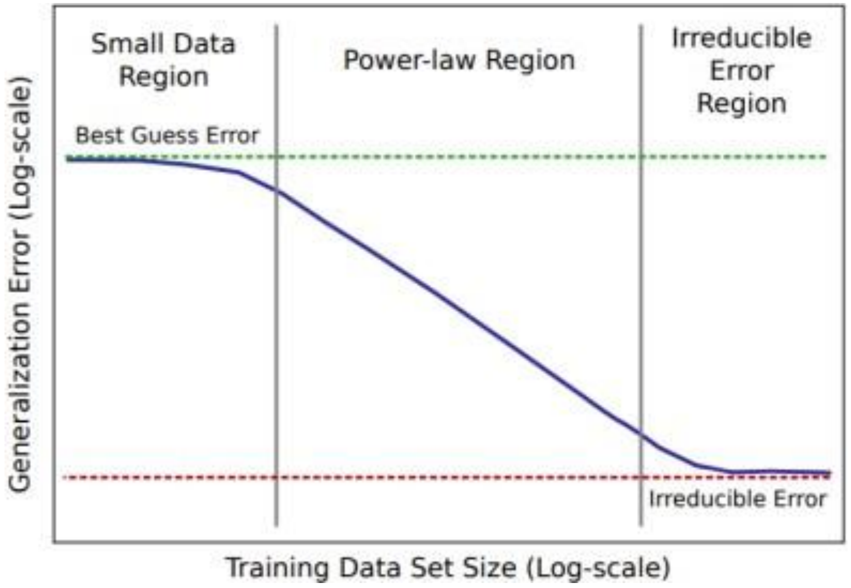
Lady Margaret Hall

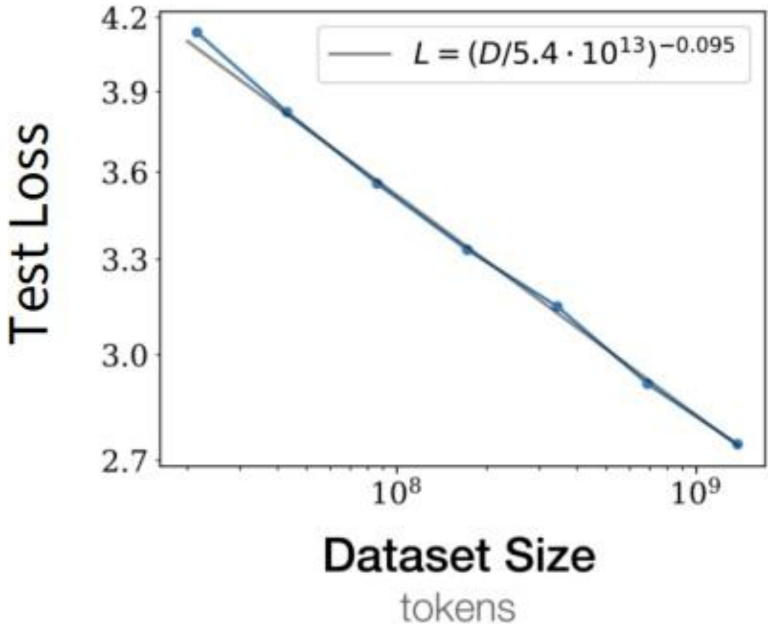
- We know that typical scaling effects look like this when we increase the amount of training data



Why is this interesting? Look at data scaling

- Loss and dataset size is linear on a log-log plot
- This is “power-law scaling”





Scaling - (Kaplan,2020)

- Can we understand scaling by positing scaling laws ?
- With scaling laws, we can make decisions on architecture, data, hyperparameters by training smaller models
- Open AI Study : **Scaling Laws for Neural Language Models** (Kaplan et al. 2020)

Scaling - (Kaplan,2020)

- Open AI Study : **Scaling Laws for Neural Language Models** (Kaplan et al. 2020)

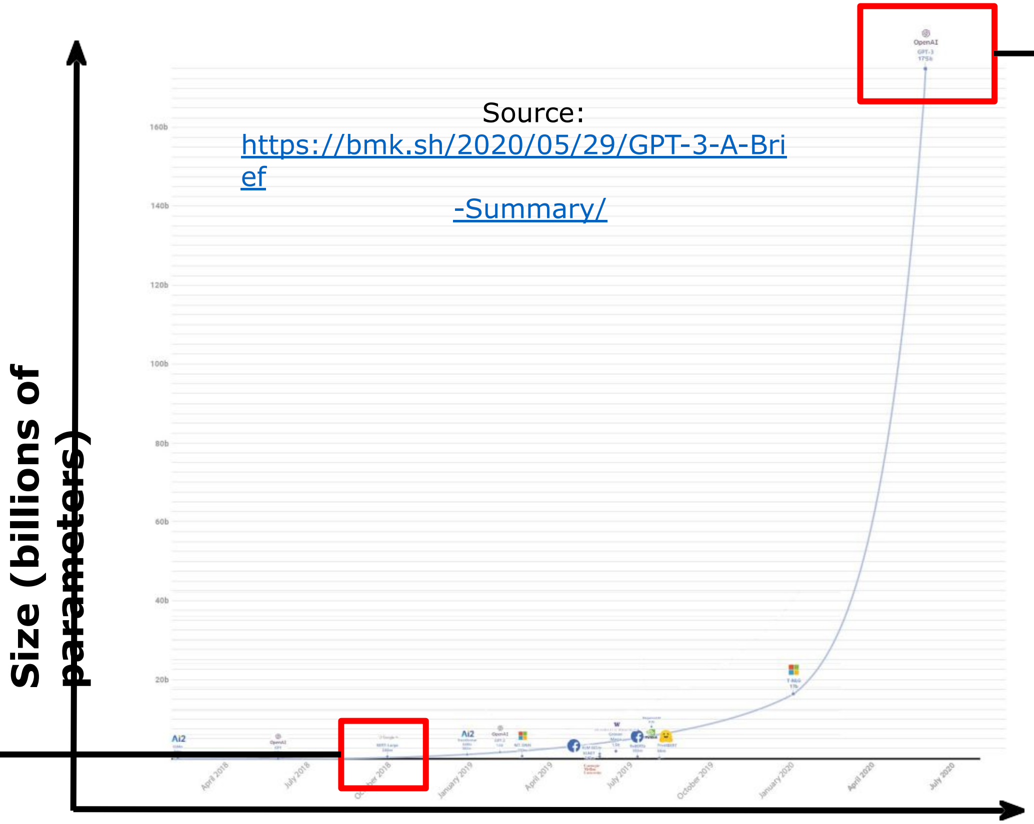
- Key Findings:
  - Performance depends strongly on scale, and weakly on the model shape
  - Larger models are more sample-efficient
  - Smooth power laws ($y = ax^k$) b/w empirical performance & N - parameters, D - dataset size, C - compute

Scaling Effects

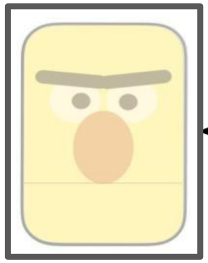
- The effect of some hyperparameters on big LMs can be predicted before training – optimizer (Adam v/s SGD), model depth, LSTM v/s Transformer

- Idea:
  - Train a few smaller models
  - Establish a scaling law (e.g. ADAM vs SGD scaling law)
  - Select optimal hyper param based on the scaling law prediction

Model Scaling: GPT-3



Source: https://bmk.sh/2020/05/29/GPT-3-A-Brief-Summary/



175b params! GPT-2 was 1.5b

Emergent Abilities with GPT-3 – Wei et. al 2022

- Emergent abilities:
  - not present in smaller models but is present in larger models
  - Do LLMs like GPT3 have these ?

- Findings:
  - GPT-3 trained on text can do arithmetic problems like addition and subtraction
  - Different abilities “emerge” at different scales

Emergent Abilities with GPT-3 – Wei et. al 2022

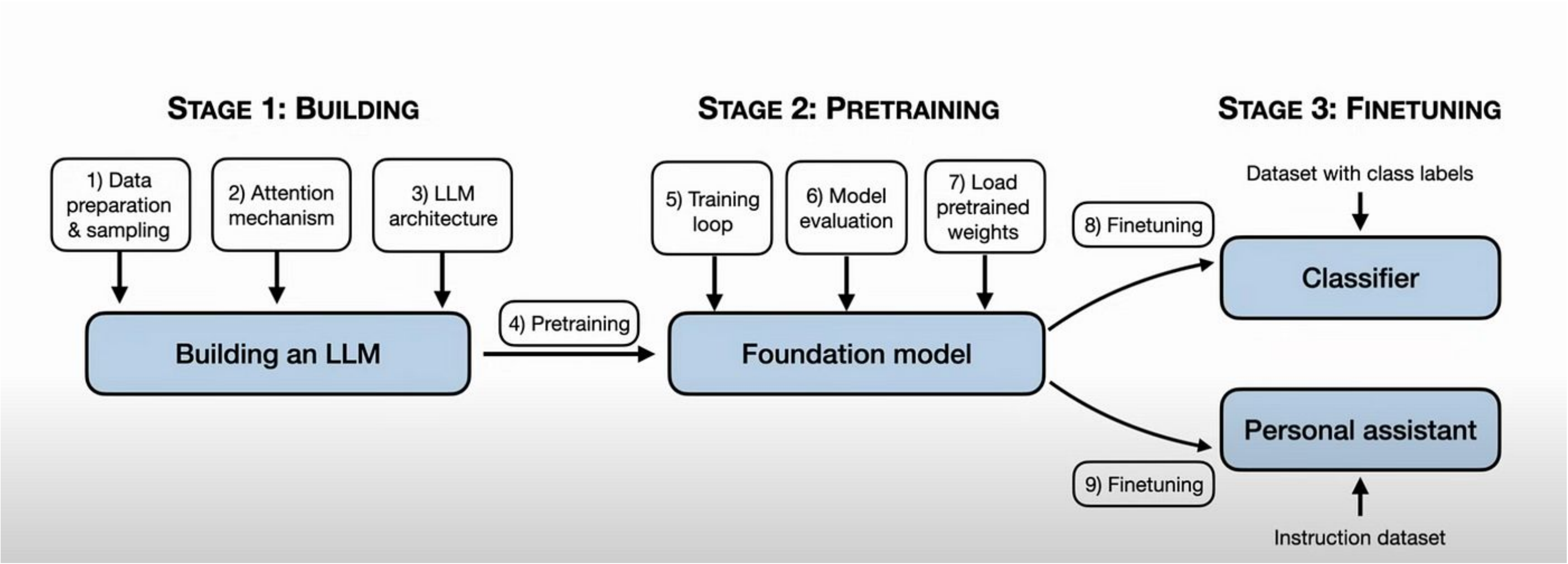
- Emergent abilities:
  - not present in smaller models but is present in larger models
  - Do LLMs like GPT3 have these ?

- Findings:
  - GPT-3 trained on text can do arithmetic problems like addition and subtraction
  - Different abilities “emerge” at different scales
  - Model scale is not the only contributor to emergence – for 14 BIG-Bench tasks, LaMDA 137B and GPT-3 175B models perform at near-random, but PaLM 62B achieves above-random performance
  - Problems LLMs can’t solve today may be emergent for future LLMs

Pre-training Overview

- **Pre-training Phase:**
  - Large-scale unsupervised training on corpus (e.g., Common Crawl, Books)
  - Objective: learn general-purpose language representations
- **Why Pre-train?**
  - Data-efficient fine-tuning
  - Enables zero-shot and few-shot capabilities
  - Foundation for instruction tuning, alignment
- **Challenges:**
  - Massive compute costs
  - Environmental concerns (carbon footprint)

Pre-training Overview



1. Auto-regressive Pre-training - Train to predict the next token on very large-scale corpora (~3 trillion tokens)

Training of Decoder-only LLMs – Llama 2

أكاديمية كاوست
KAUST ACADEMY

Lady Margaret Hall

1. Auto-regressive Pre-training - Train to predict the next token on very large scale corpora (~3 trillion tokens)
2. Instruction Fine-tuning/ Supervised Fine-tuning (SFT) - Fine-tune the pre-trained model with pairs of (instruction+input,output) with large dataset and then with small high-quality dataset

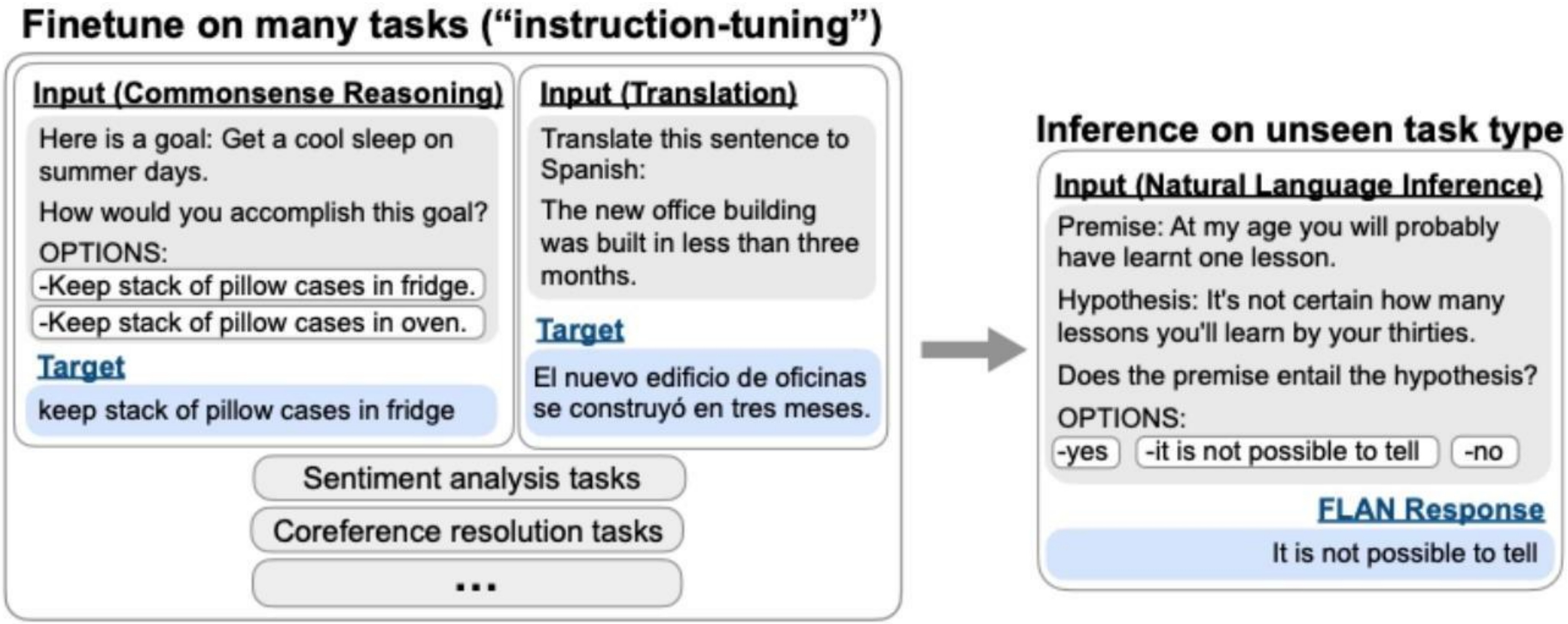
Instruction fine-tuning provides as a prefix a natural language description of the task along with the input.

- E.g. Translate into French this sentence: my name is -> je m’appelle

Supervised Fine-tuning versus Pre-training

- Objective function
  - Loss computed only for target tokens in SFT, all tokens are targets in pre-training
- Input and Target
  - Instruction + input as input with the target in SFT and only input as input with shifted input as target
- Purpose
  - Pre-training makes good generalist auto-completes but good SFT builds models that can do many unseen tasks
  - SFT can also guide nature of outputs in terms of safety and helpfulness

Instruction Tuning (Wei et. al. 2021)



Fine-Tuning Methods for LLMs

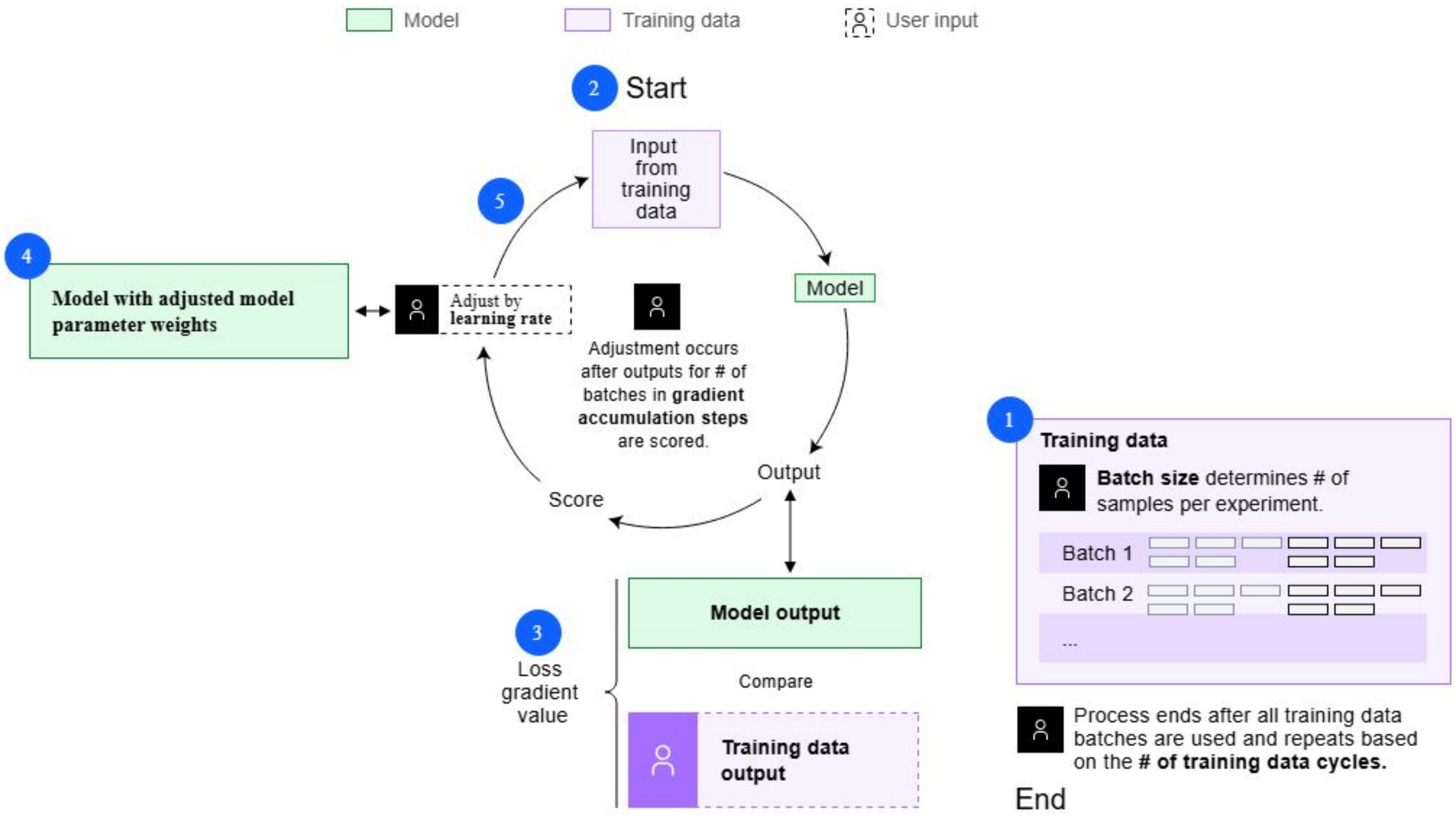
Categories of Fine-Tuning

Method	Description	Parameters Updated	Efficiency
Full Fine-Tuning	Retrain all model weights	All	Expensive
Adapter Tuning	Add small bottlenecks (e.g., Houlsby)	Few	Efficient
Prefix Tuning	Tune soft prompts	Few tokens	Efficient
LoRA / QLoRA	Low-rank decomposition of weight deltas	Very few	Very Efficient
Instruction Tuning	Fine-tune on instruction-following datasets	All or partial	

Full Fine-Tuning (FT)

- **Definition:** Fine-tune all parameters of the pre-trained model on downstream data.
- **Historical Use:** Common in early GPT-2 and BERT applications.
- **Pros:**
  - Maximum flexibility and performance
- **Cons:**
  - Expensive (requires large compute resources)
  - Prone to catastrophic forgetting
  - Not efficient for large models

Full fine-tuning workflow

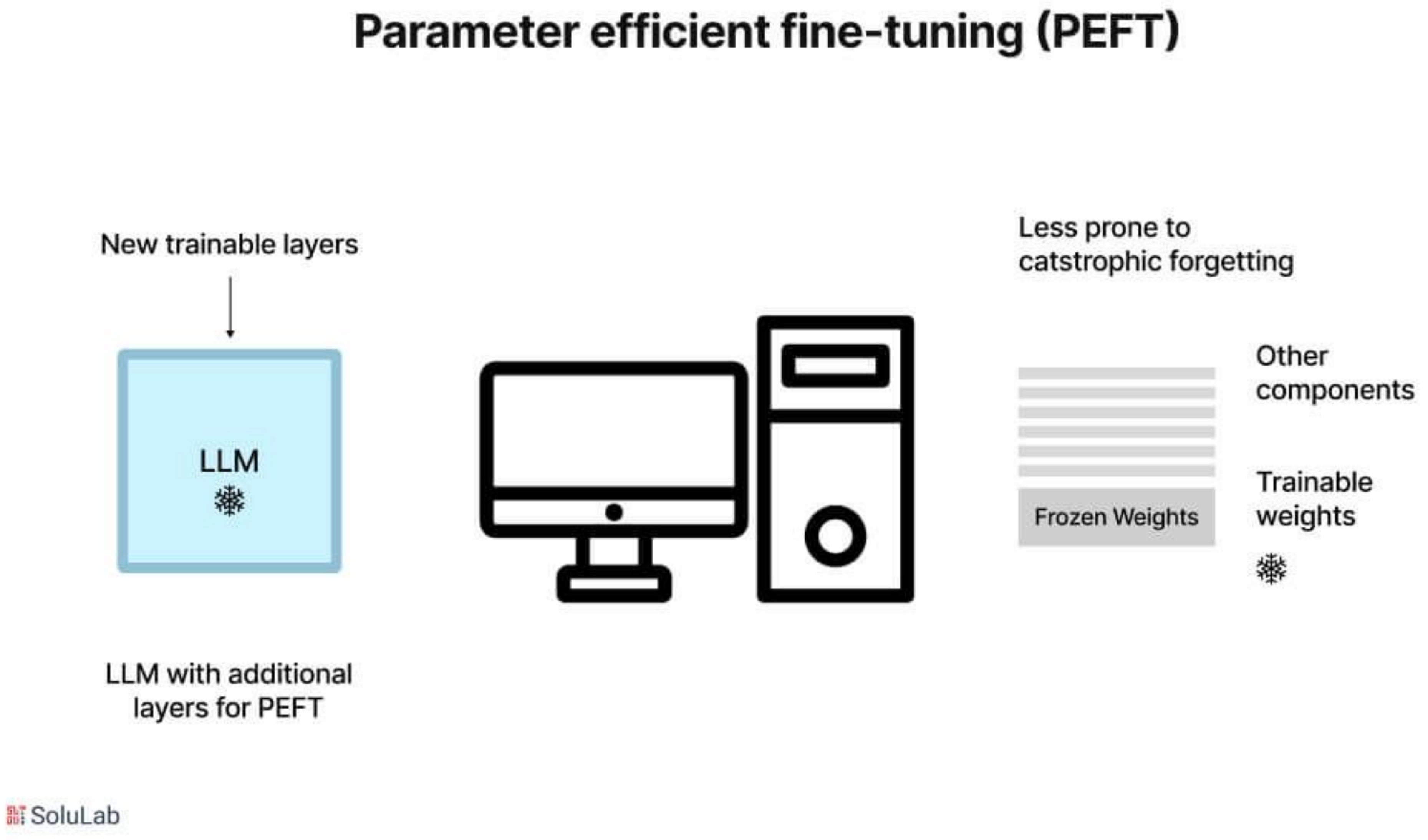


Parameter-Efficient Fine-Tuning (PEFT)

أكاديمية كاوست
KAUST ACADEMY

LMH
Lady Margaret Hall

- **Key Idea:** Keep the base model frozen, tune only a small subset of parameters.
- **Types:**
  - Adapter Modules (Houlsby et al., 2019)
  - Prompt Tuning / Prefix Tuning
  - LoRA / QLoRA
- **Benefits:**
  - Enables multi-tasking and personalization
  - Suitable for low-resource adaptation
  - Reduces compute and memory requirements



Parameter efficient fine-tuning (PEFT)
New trainable layers
LLM
LLM with additional layers for PEFT
Less prone to catstrophic forgetting
Other components
Frozen Weights
Trainable weights
SoluLab

Supervised Fine-Tuning (SFT)

What is Supervised Fine-Tuning (SFT)?

أكاديمية كاوست
KAUST ACADEMY

LMH
Lady Margaret Hall

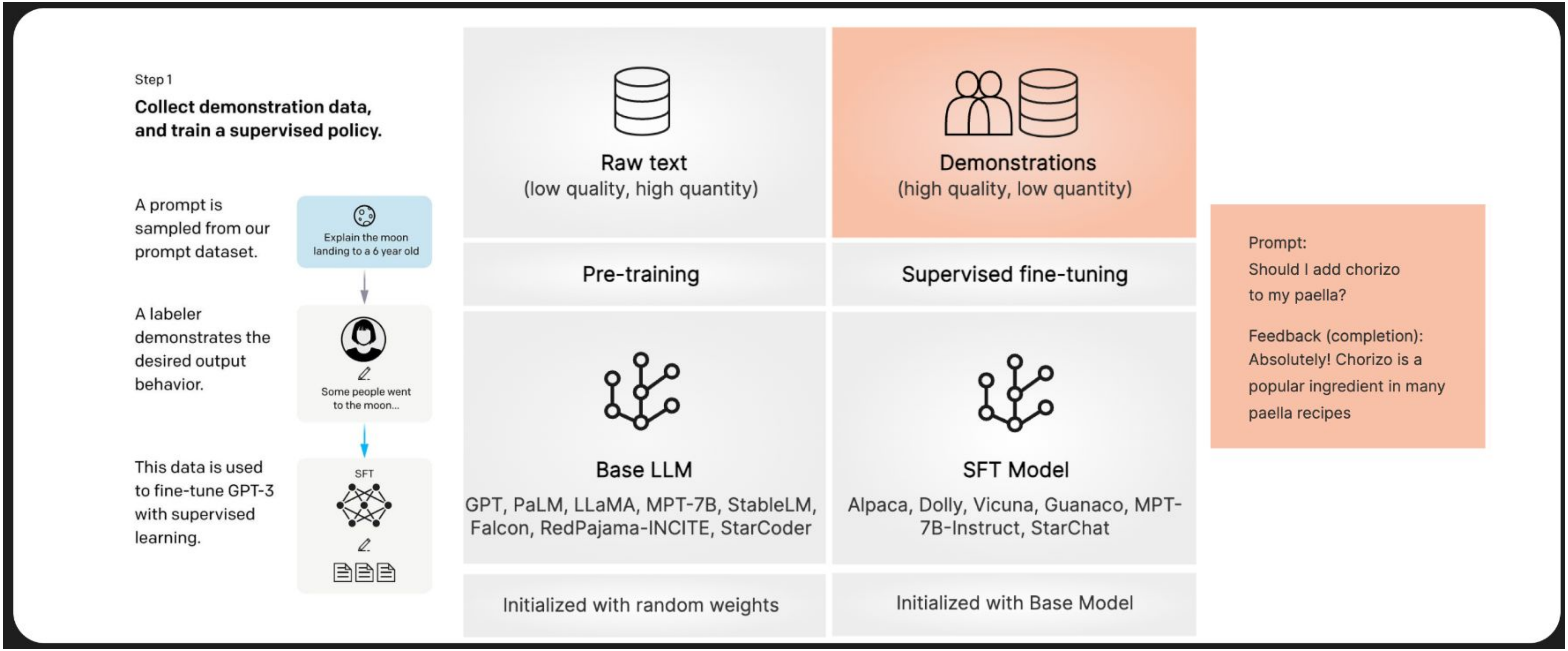
- **Definition:** SFT is training on labeled instruction-response pairs.
- **Example:**
  - *Prompt:* "Explain black holes to a 5-year-old"
  - *Response:* "Black holes are like big vacuum cleaners in space. . . "
- **Objective:** Optimize the log-likelihood of the correct response given the prompt.

Loss Function:

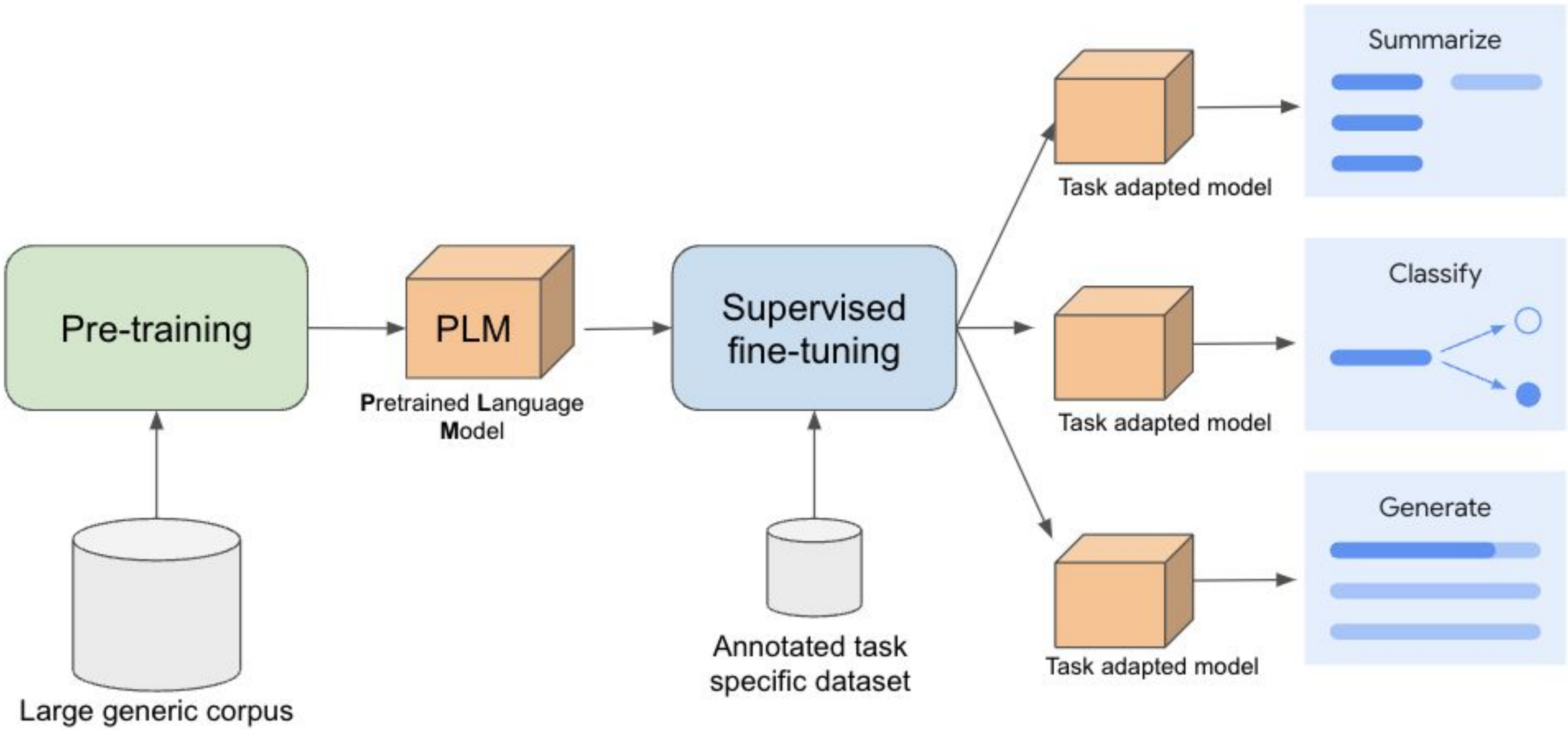
$$L_{\mathrm{SFT}} = -\sum_{t=1}^{T} \log p_\theta(y_t \mid y_{<t}, x)$$

where x is the prompt, y is the response, and T is the response length.

What is Supervised Fine-Tuning (SFT)?



Supervised Fine Tuning for Gemini LLM



Supervised Fine Tuning for Gemini LLM

Datasets for Supervised Fine-Tuning (SFT)

- **Instructional datasets:**
  - OpenAI: InstructGPT (Anthropic Helpfulness data)
  - Stanford Alpaca (52k GPT-3 generated instructions)
  - Dolly, ShareGPT, OASST, UltraChat
- **Note:** Quality of supervision greatly affects model behavior.

Limitations of Supervised Fine-Tuning (SFT)

- Cannot capture nuanced human preferences or values.
- May reinforce existing biases or hallucinations present in the data.
- Risk of overfitting, especially on synthetic or noisy datasets.
- Often leads to safe but bland and generic responses.

LoRA and Quantized LoRA

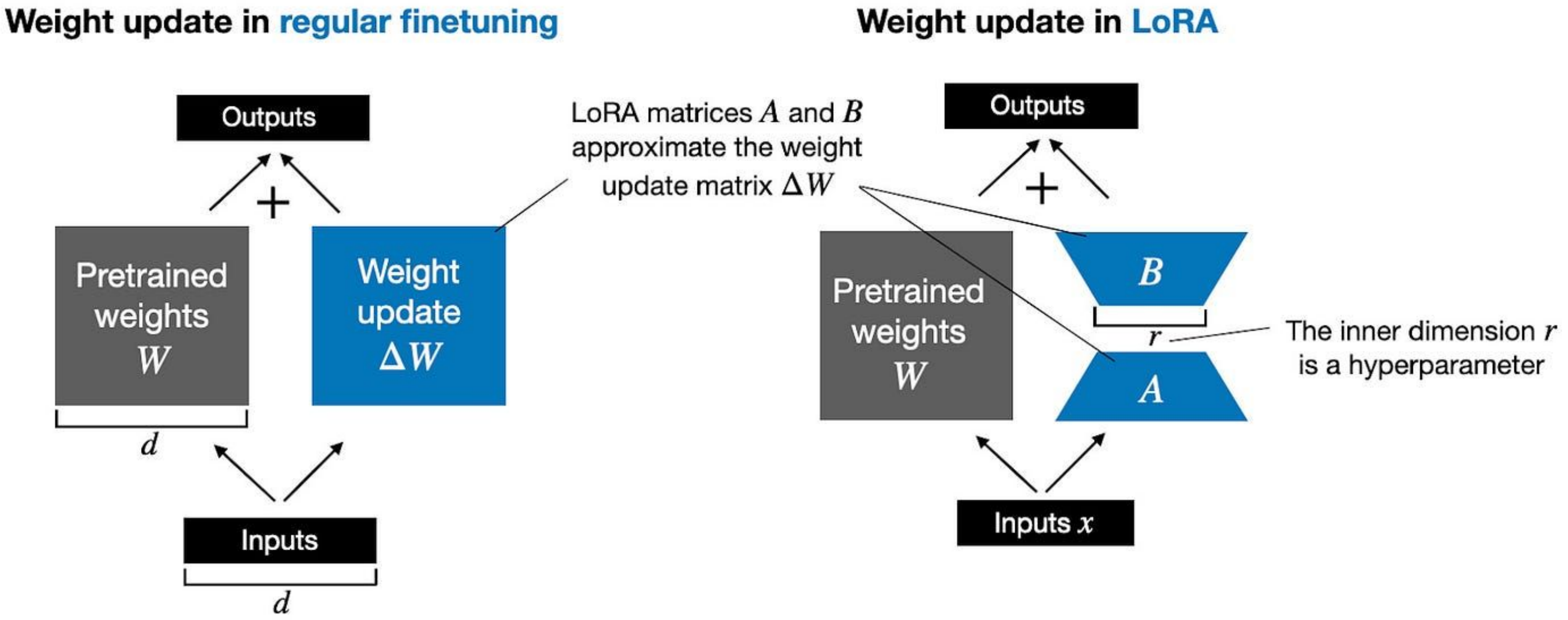
What is LoRA?

- **Key Idea:** Update low-rank matrices instead of full weights.
- For a weight matrix $W \in \mathbb{R}^{d \times k}$:

$$W' = W + \Delta W, \quad \Delta W = AB^T$$

  where $A \in \mathbb{R}^{d \times r}$, $B \in \mathbb{R}^{k \times r}$.
- Only train A, B $\rightarrow$ drastically reduces parameters.

What is LoRA?


Weight update in regular finetuning
Weight update in LoRA
Outputs
Pretrained weights W
Weight update ΔW
d
Inputs
d
LoRA matrices A and B approximate the weight update matrix ΔW
Outputs
Pretrained weights W
B
r
A
The inner dimension r is a hyperparameter
Inputs x

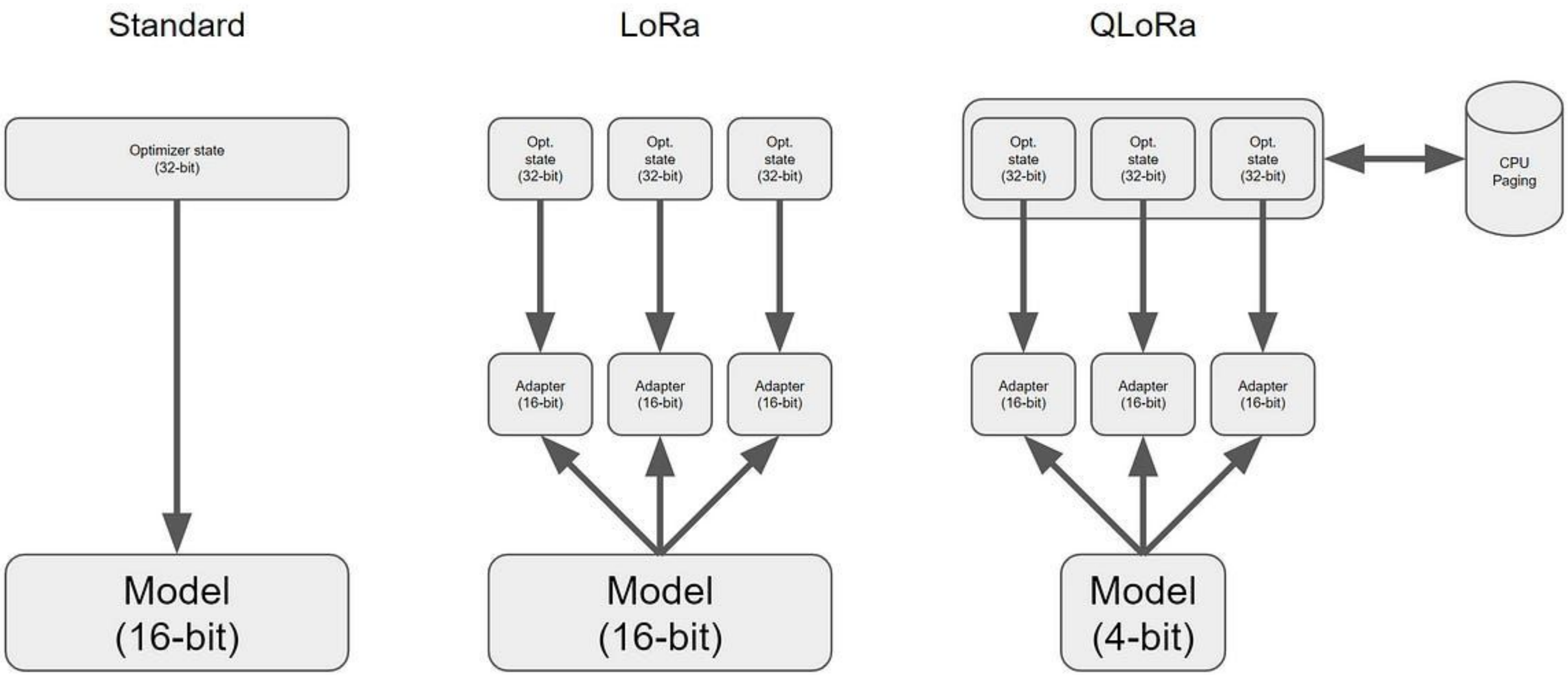
Benefits of LoRA

- **Very lightweight:** Only 0.1%–1% of parameters are trainable.
- **Hardware friendly:** Enables fine-tuning on consumer GPUs and even laptops.
- **Modular:** Supports plug-and-play adapters for different tasks or users.
- **Personalization:** Allows efficient user- or domain-specific adaptation.

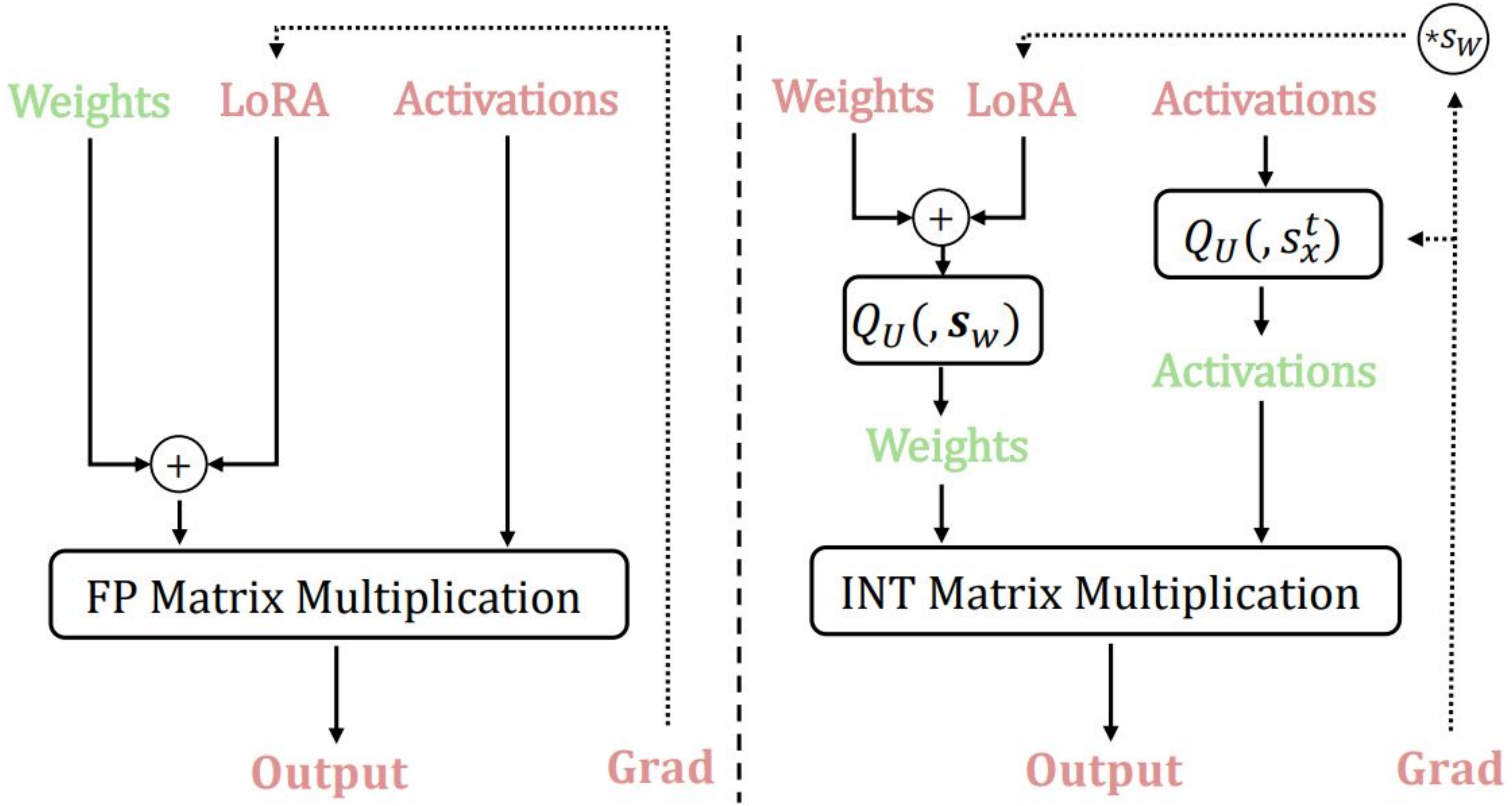
QLoRA — Quantized LoRA

- **Key Idea:** Fine-tune large language models in 4-bit precision using LoRA adapters.
- **Scalability:** Enables fine-tuning of 65B parameter models on a single 48GB GPU.
- **Efficiency:** Highly memory-efficient with no significant performance loss (Dettmers et al., 2023).
- **Techniques:**
  - Double quantization
  - Paged optimizers
  - NF4 (NormalFloat 4-bit) quantization format

QLoRA — Quantized LoRA



QLoRA — Quantized LoRA



(a) QLoRA layer

(b) QALoRA layer

Applications of LoRA / QLoRA

- **Instruction tuning for resource-constrained settings:**
  - Enables fine-tuning large models on modest hardware (e.g., consumer GPUs, laptops).
  - Used for domain adaptation, personalization, and rapid prototyping.
- **Popular fine-tuned models:**
  - **Alpaca, Guanaco, Vicuna, Mistral:** All leverage LoRA/QLoRA for efficient instruction tuning.
- **Model merging and compositionality:**
  - Merge multiple LoRA adapters for multi-domain or multi-task capabilities.
  - Compose adapters for new tasks without retraining the base model.

Evaluation Metrics

Evaluation Metrics for LLMs

Perplexity Measures how well a model predicts text; lower is better

BLEU Compares generated text to reference using n-gram overlap

ROUGE Evaluates overlap with human-generated text; used for summarization

METEOR Considers synonyms and word order; improved BLEU for translation

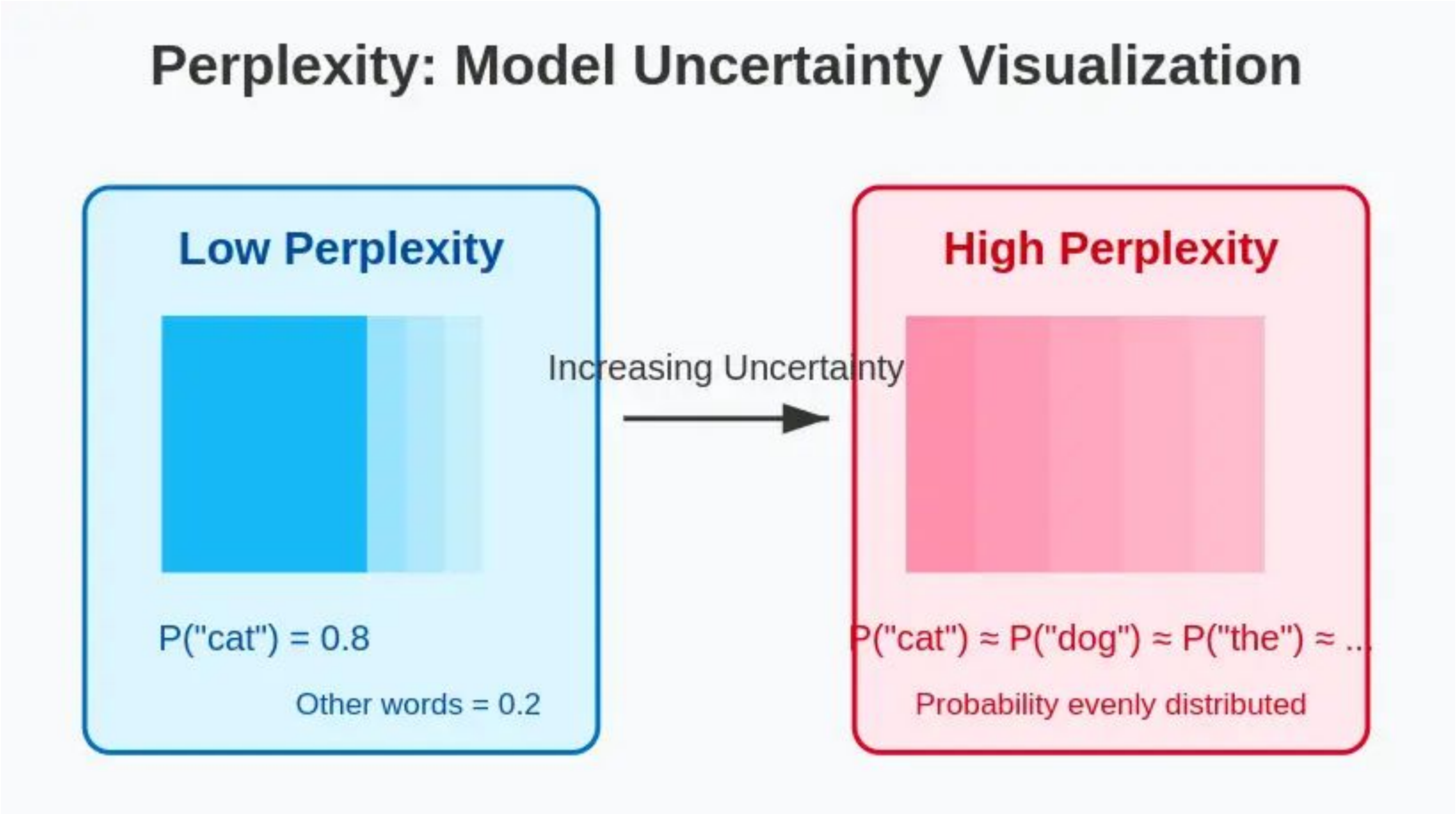
BERTScore Calculates semantic similarity with contextual embeddings

Human Evaluation Involves rating outouts for qualities like relevance and coherence

Task-Specific Metrics E.g.. Exact Match for QA, Pass@k for code generation

Prompt-Injected Evaluation Uses LLMs to assess outputs; scalable but may be biased

Perplexity


Perplexity: Model Uncertainty Visualization
Low Perplexity
P("cat") = 0.8
Other words = 0.2
Increasing Uncertainty
High Perplexity
P("cat") ≈ P("dog") ≈ P("the") ≈ ...
Probability evenly distributed

Perplexity

$$\text{perplexity} = \underbrace{\prod_{t=1}^{T} \left(\frac{1}{P_{\text{LM}}(\boldsymbol{x}^{(t+1)} \mid \boldsymbol{x}^{(t)}, \ldots, \boldsymbol{x}^{(1)})} \right)}_{\text{Inverse probability of corpus, according to Language Model}}^{1/T}$$

Normalized by number of words

Dan Jurafsky

Perplexity

The best language model is one that best predicts an unseen test set

- Gives the highest P(sentence)

Perplexity is the inverse probability of the test set, normalized by the number of words:

$$PP(W) = P(w_1 w_2 \ldots w_N)^{-\frac{1}{N}}$$

$$= \sqrt[N]{\frac{1}{P(w_1 w_2 \ldots w_N)}}$$

Chain rule:

$$\mathrm{PP}(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_1 \ldots w_{i-1})}}$$

For bigrams:

$$\mathrm{PP}(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_{i-1})}}$$

Minimizing perplexity is the same as maximizing probability

BLEU

- N-gram overlap between machine translation output and reference translation
- Compute precision for n-grams of size 1 to 4
- Add brevity penalty (for too short translations)

$$\text{BLEU} = \min\left(1, \frac{\textit{output-length}}{\textit{reference-length}}\right) \left(\prod_{i=1}^{4} \textit{precision}_i\right)^{\frac{1}{4}}$$

- Typically computed over the entire corpus, not single sentences

BLEU (Bilingual Evaluation Understudy)

أكاديمية كاوست
KAUST ACADEMY

LMH
Lady Margaret Hall

BLEU Evaluation Metric

Reference (Human) translation:

The U.S. island of Guam is maintaining a high state of alert after the Guam airport and its offices both received an e-mail from someone calling himself the Saudi Arabian Osama bin Laden and threatening a biological/chemical attack against public places such as the airport .

Machine translation:

The American [?] international airport and its the office all receives one calls self the sand Arab rich business [?] and so on electronic mail , which sends out ; The threat will be able after public place and so on the airport to start the biochemistry attack , [?] highly alerts after the maintenance.

- Score is between 0 and 1 (sometimes normalized to a number between 0 and 100)
- **What percentage of MT output n-grams (text string clusters) can be found in the reference translation?**
- Usually calculated on ~**1000 test sentences**.
- Important to reward the right things and there is brevity penalty
- Getting larger word clusters to match provides better scores

Example from C. Wayne

LLM Evaluation - Metrics - ROUGE clipping

Reference (human):

It is cold outside.

Generated output:

cold cold cold cold

$$\text{ROUGE-1 Precision} = \frac{\text{unigram matches}}{\text{unigrams in output}} = \frac{4}{4} = 1.0$$

$$\text{Modified precision} = \frac{\text{clip(unigram matches)}}{\text{unigrams in output}} = \frac{1}{4} = 0.25$$

Generated output:

outside cold it is

$$\text{Modified precision} = \frac{\text{clip(unigram matches)}}{\text{unigrams in output}} = \frac{4}{4} = 1.0$$

أكاديمية كاوست
KAUST ACADEMY

LMH
Lady Margaret Hall

LLM Evaluation - Metrics - ROUGE-L

Reference (human):

It is cold outside.

Generated output:

It is very cold outside.

LCS:
Longest common subsequence

$$\text{ROUGE-L Recall:} = \frac{\text{LCS(Gen, Ref)}}{\text{unigrams in reference}} = \frac{2}{4} = 0.5$$

$$\text{ROUGE-L Precision:} = \frac{\text{LCS(Gen, Ref)}}{\text{unigrams in output}} = \frac{2}{5} = 0.4$$

$$\text{ROUGE-L F1:} = 2\,\frac{\text{precision x recall}}{\text{precision + recall}} = 2\,\frac{0.2}{0.9} = 0.44$$

ROUGE

ROUGE-N

$$= \frac{\sum_{S \in \{ReferemceSummaries\}} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count(gram_n)} \quad (1)$$

References

These slides have been adapted from

- Bhiksha Raj & Rita Singh, 11-785 Introduction to Deep Learning, CMU

References

[1] Kaplan et al., “Scaling Laws for Neural Language Models”, 2020.

[2] Devlin et al., “BERT: Pre-training of Deep Bidirectional Transformers”, 2018.

[3] Brown et al., “Language Models are Few-Shot Learners (GPT-3)”, 2020.

[4] Xue et al., “ByT5: Towards a token-free future with pre-trained byte-to-byte models”, 2022.

[5] Beltagy et al., “Longformer: The Long-Document Transformer”, 2020.

[6] Tay et al., “Efficient Transformers: A Survey”, 2020.

[7] OpenAI, “Technical Report on GPT-4”, 2023.

References

[8] Google Research Blog: Scaling Transformer Models.

[9] Anthropic, “Claude 3.5 Release Notes”, 2024.

[10] FlashAttention: https://arxiv.org/abs/2205.14135

Credits

Dr. Prashant Aparajeya

Computer Vision Scientist — Director(AISimply Ltd)

p.aparajeya@aisimply.uk

This project benefited from external collaboration, and we acknowledge their contribution with gratitude.