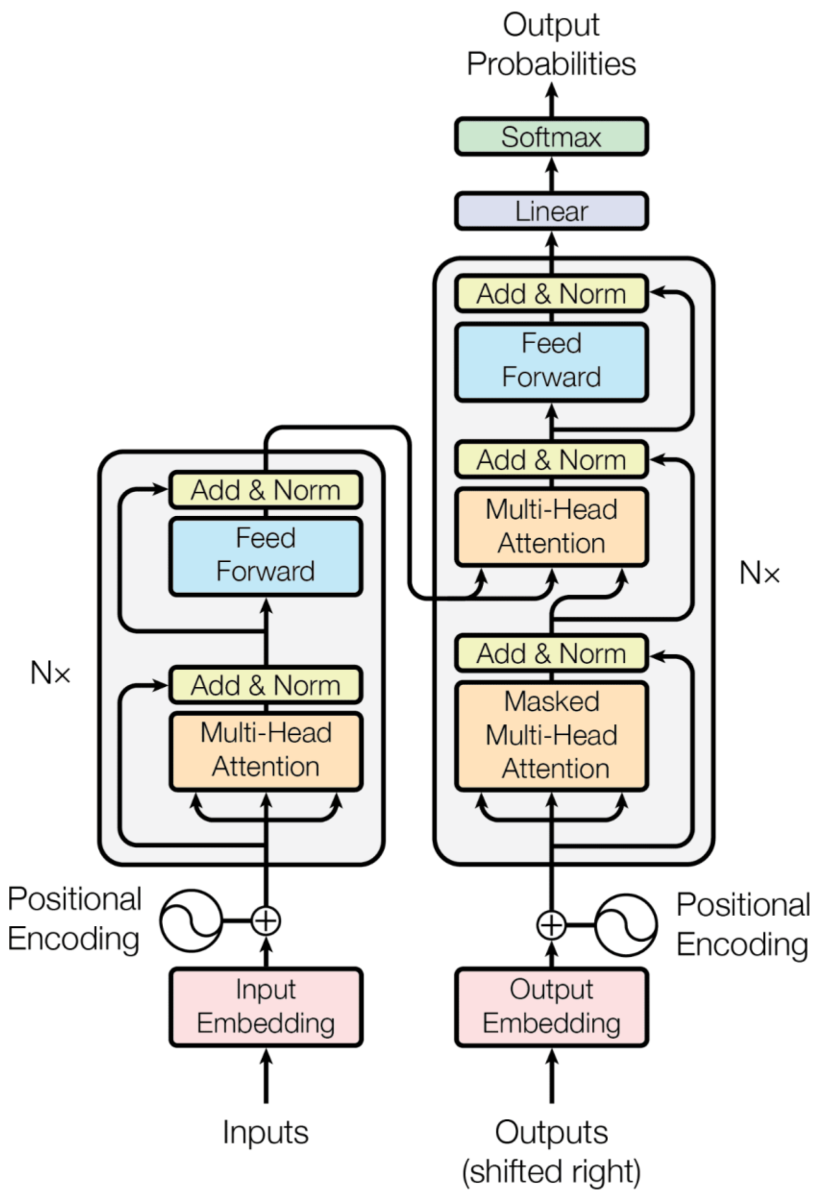
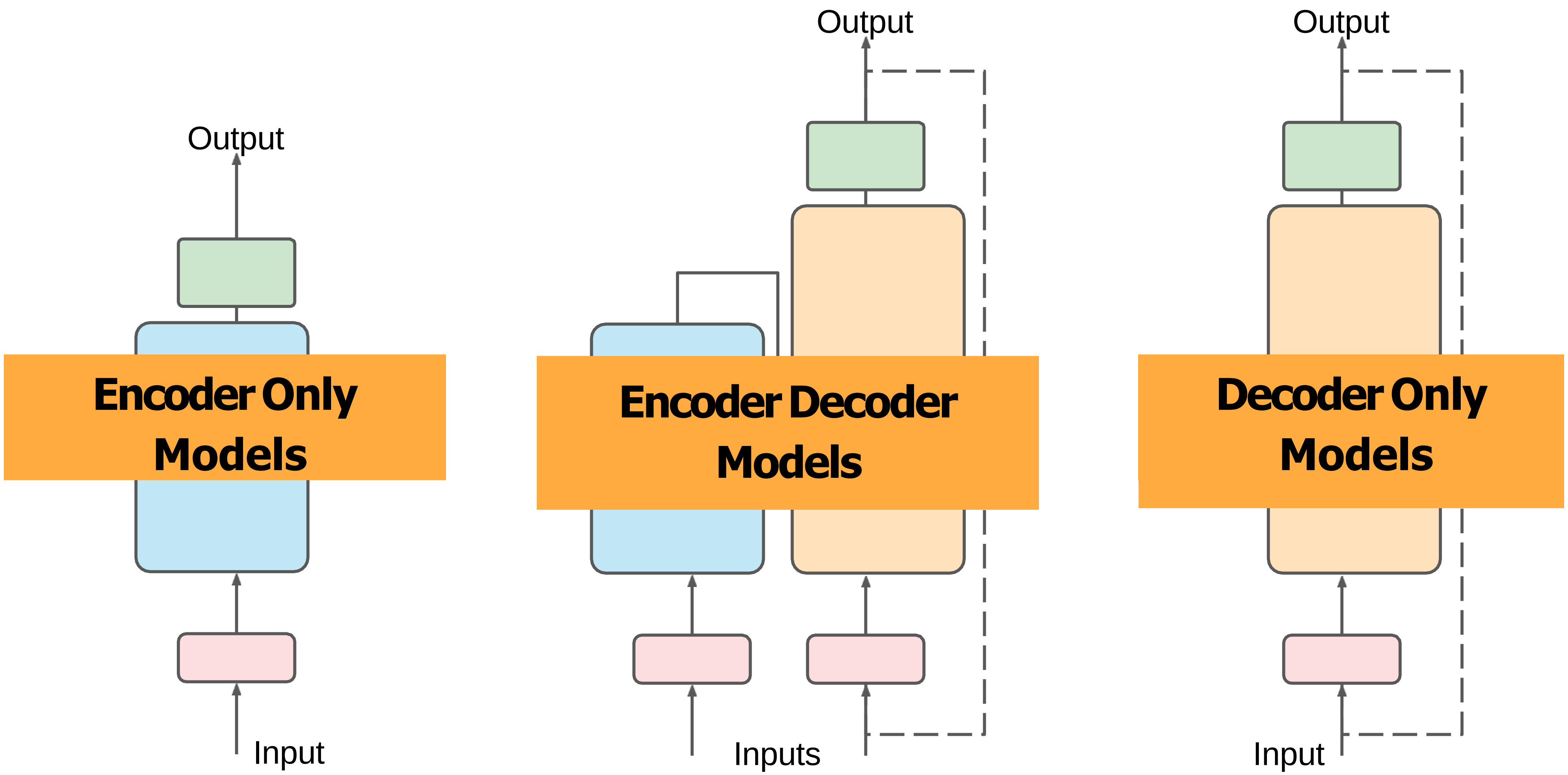
TRAINING LLMs



Transformers



Output

Encoder Only Models

Input

Output

Encoder Decoder Models

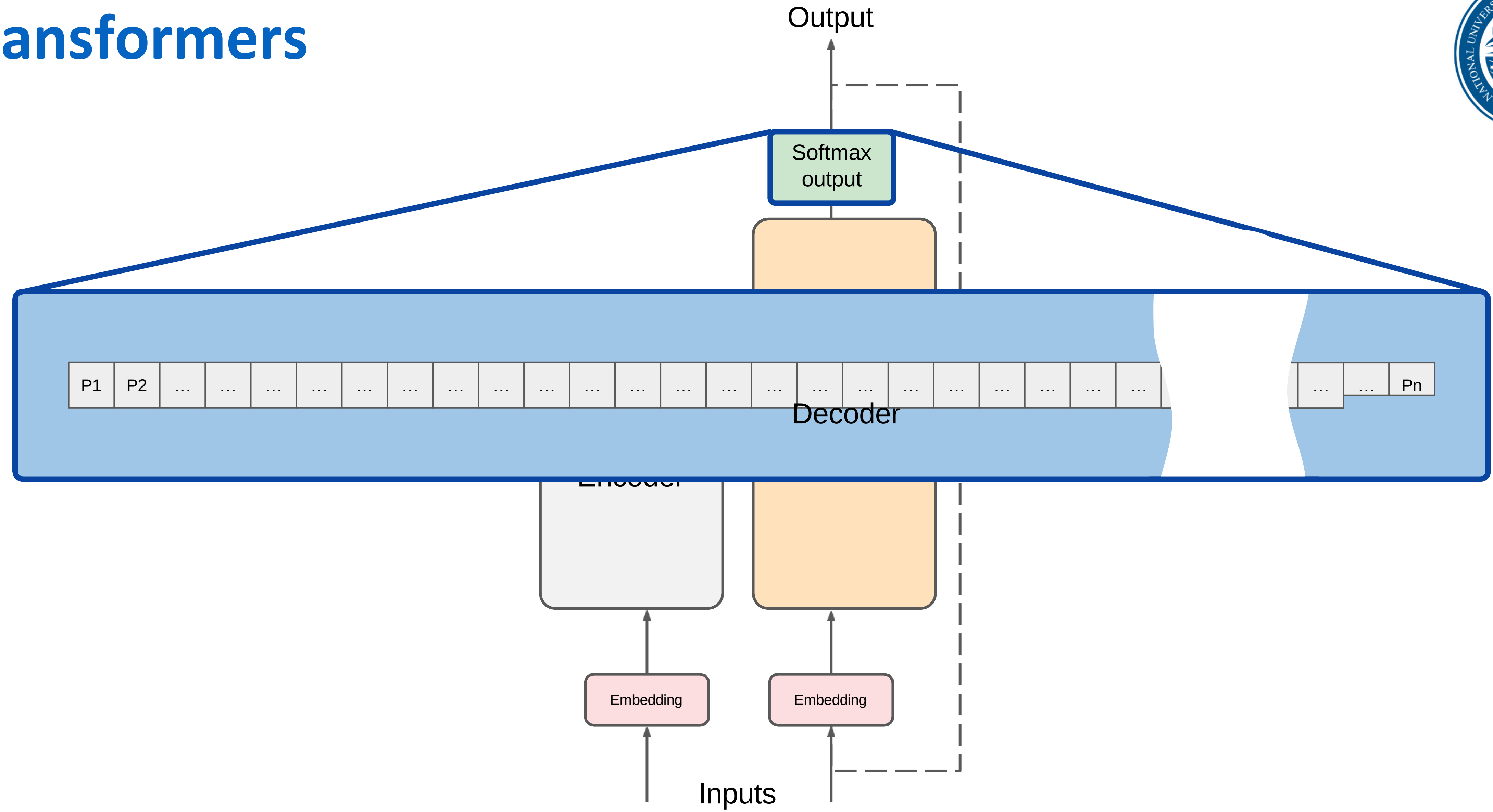
Inputs

Output

Decoder Only Models

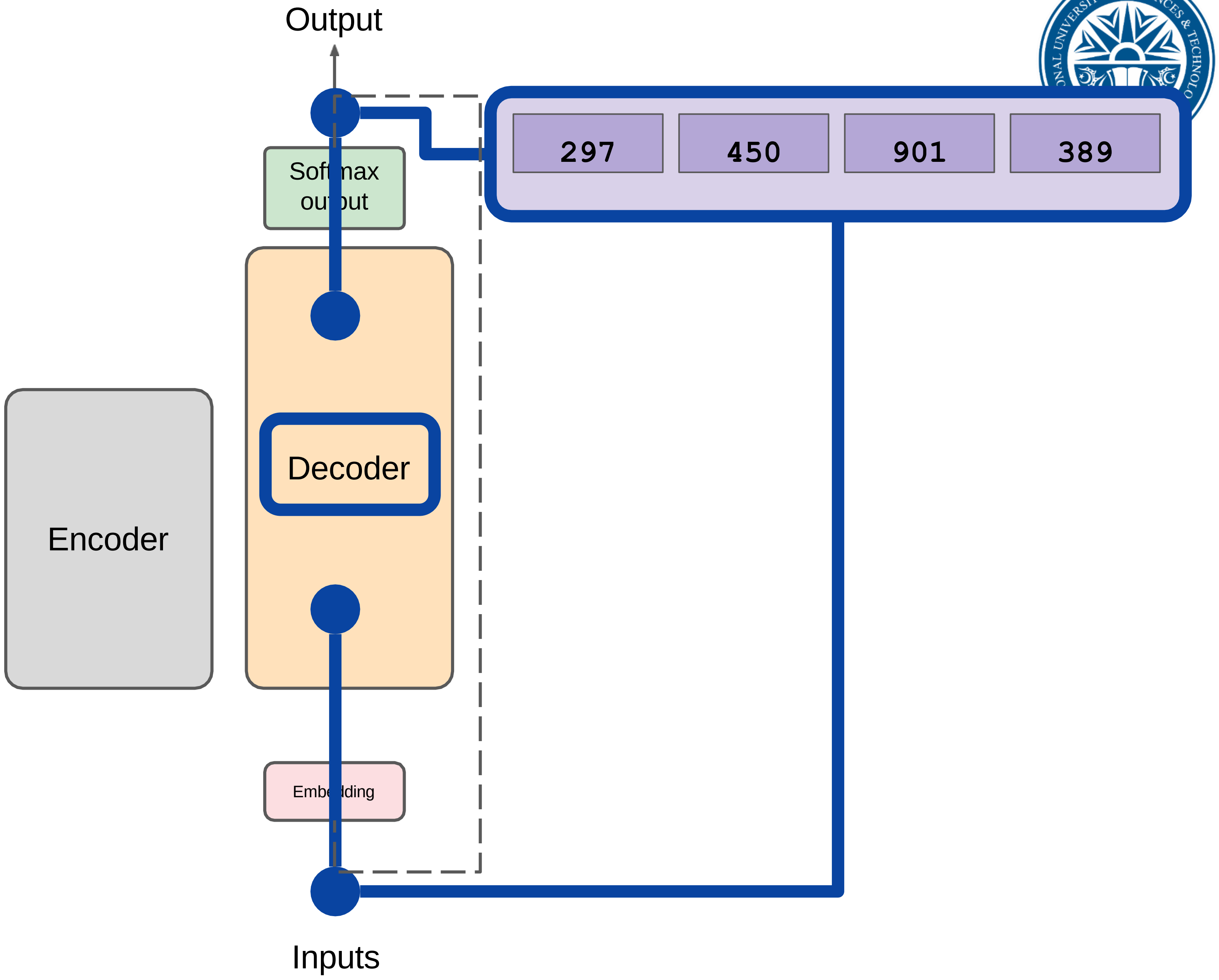
Input

Transformers



Transformers

Language Generation

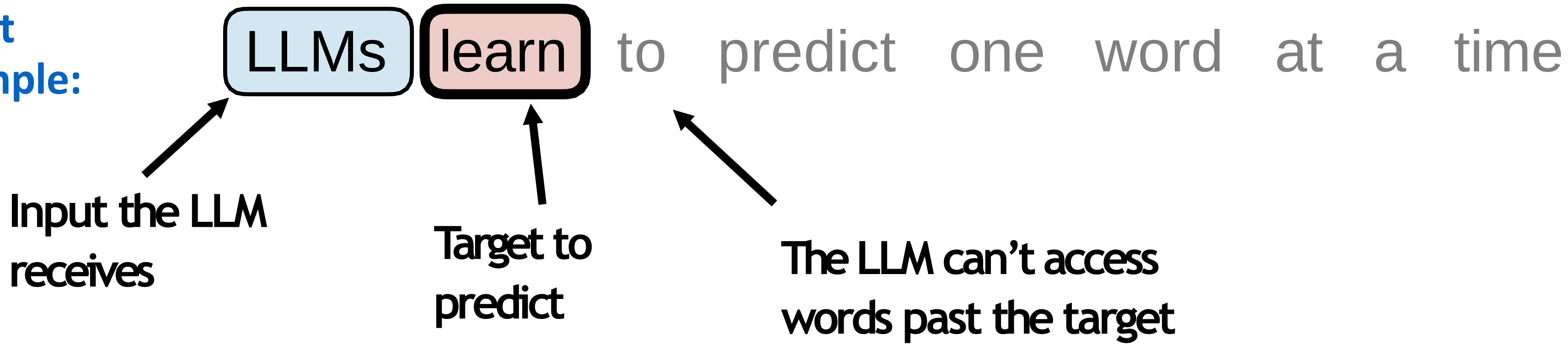


Next word (/token) prediction

Text sample:

LLMs learn to predict one word at a time

Text sample:

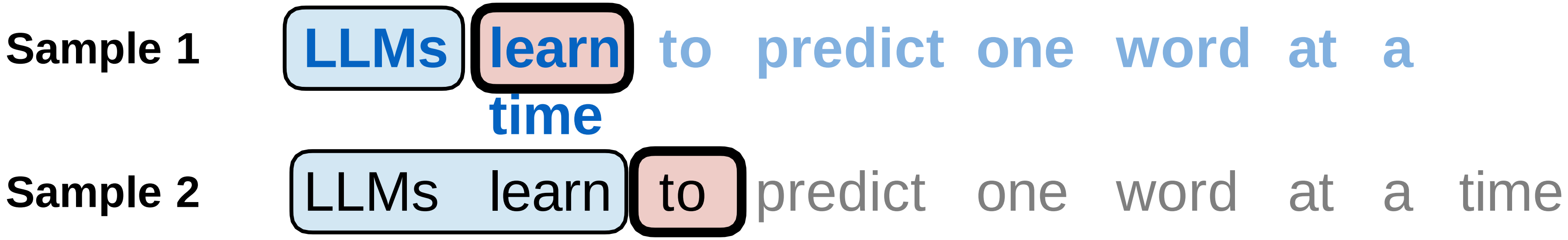


LLMs learn to predict one word at a time

Input the LLM receives

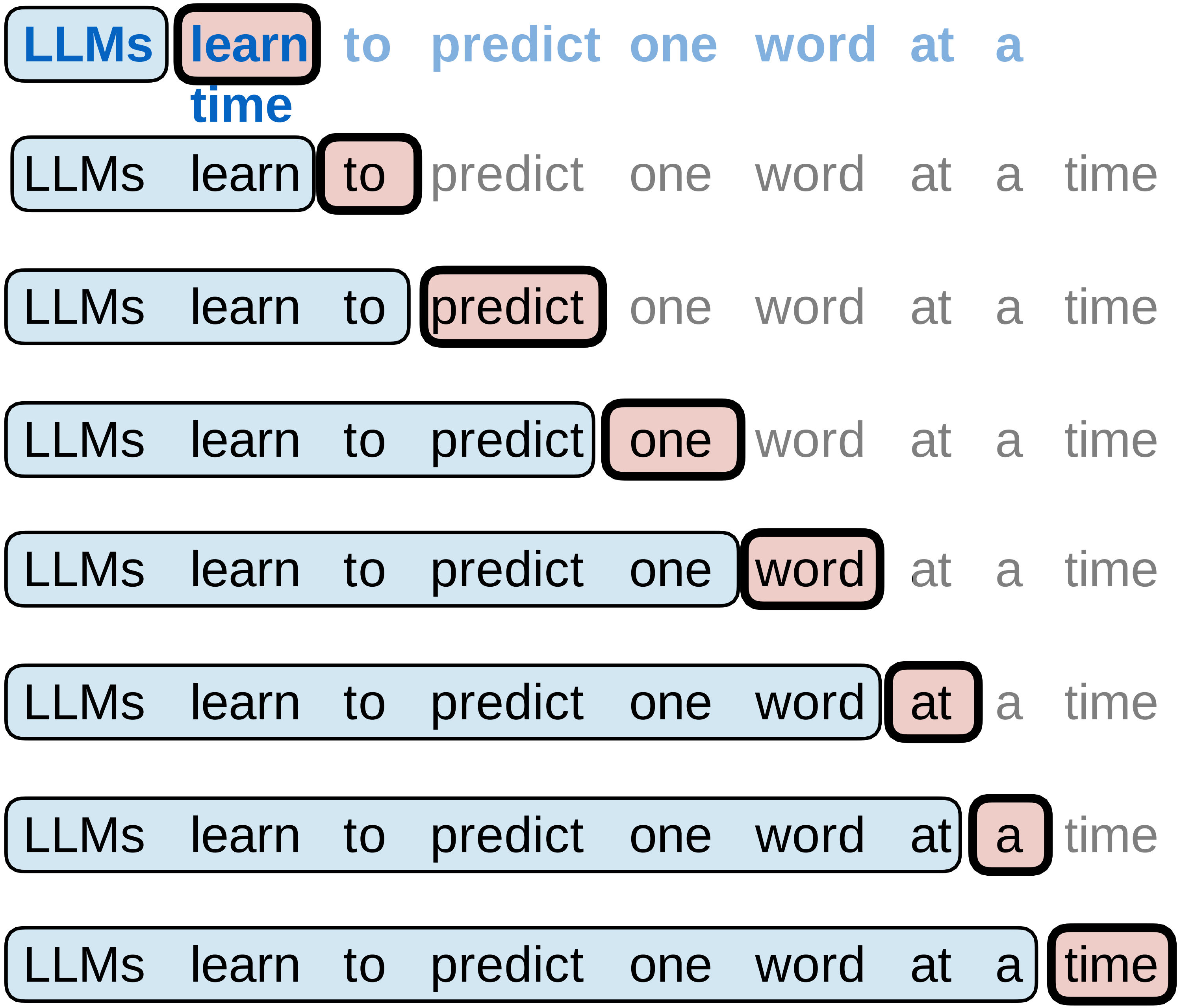
Target to predict

The LLM can’t access words past the target



Sample 1 LLMs learn to predict one word at a time

Sample 2 LLMs learn to predict one word at a time



Sample 1 LLMs learn to predict one word at a time

Sample 2 LLMs learn to predict one word at a time

Sample 3 LLMs learn to predict one word at a time

Sample 4 LLMs learn to predict one word at a time

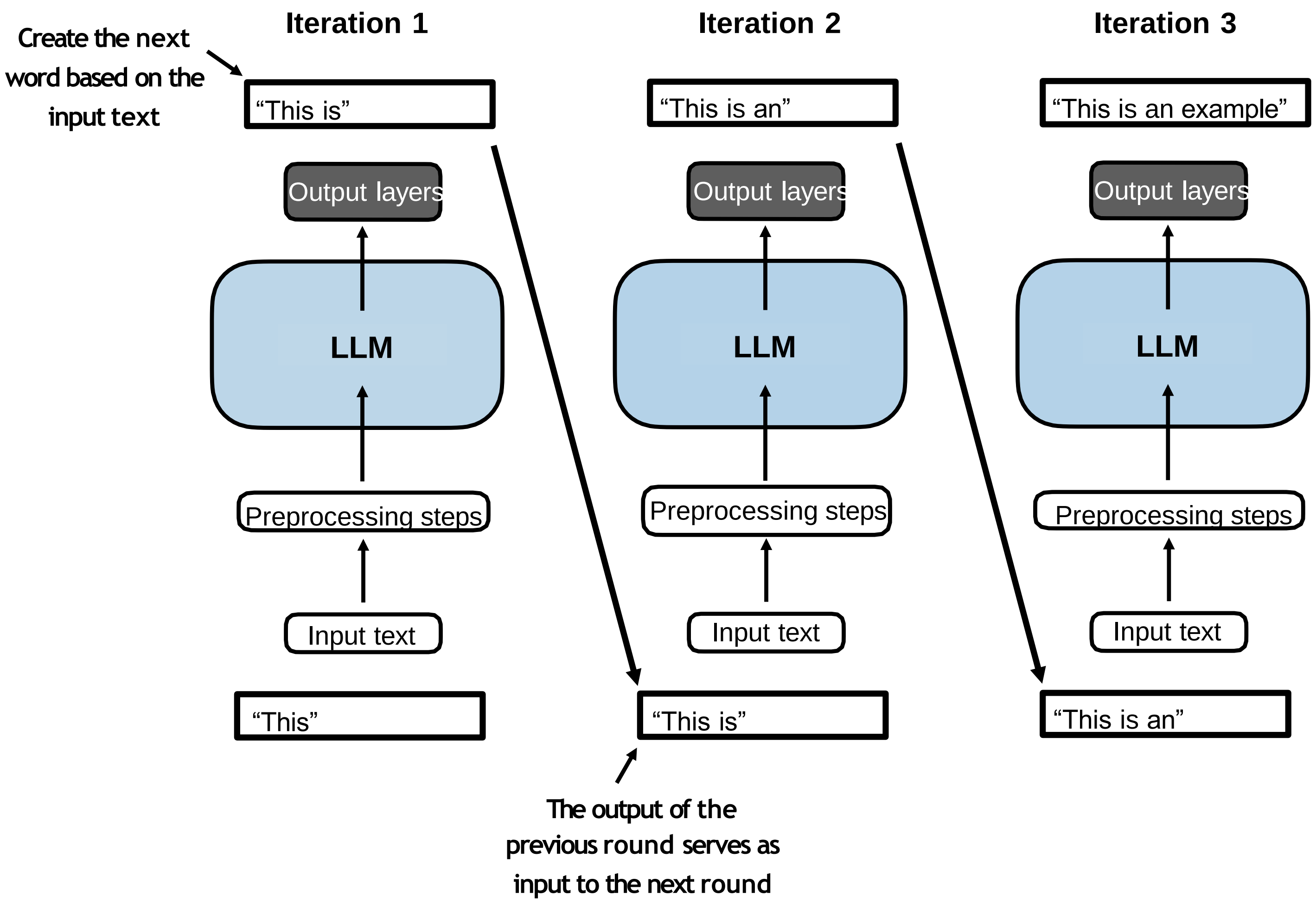
Sample 5 LLMs learn to predict one word at a time

Sample 6 LLMs learn to predict one word at a time

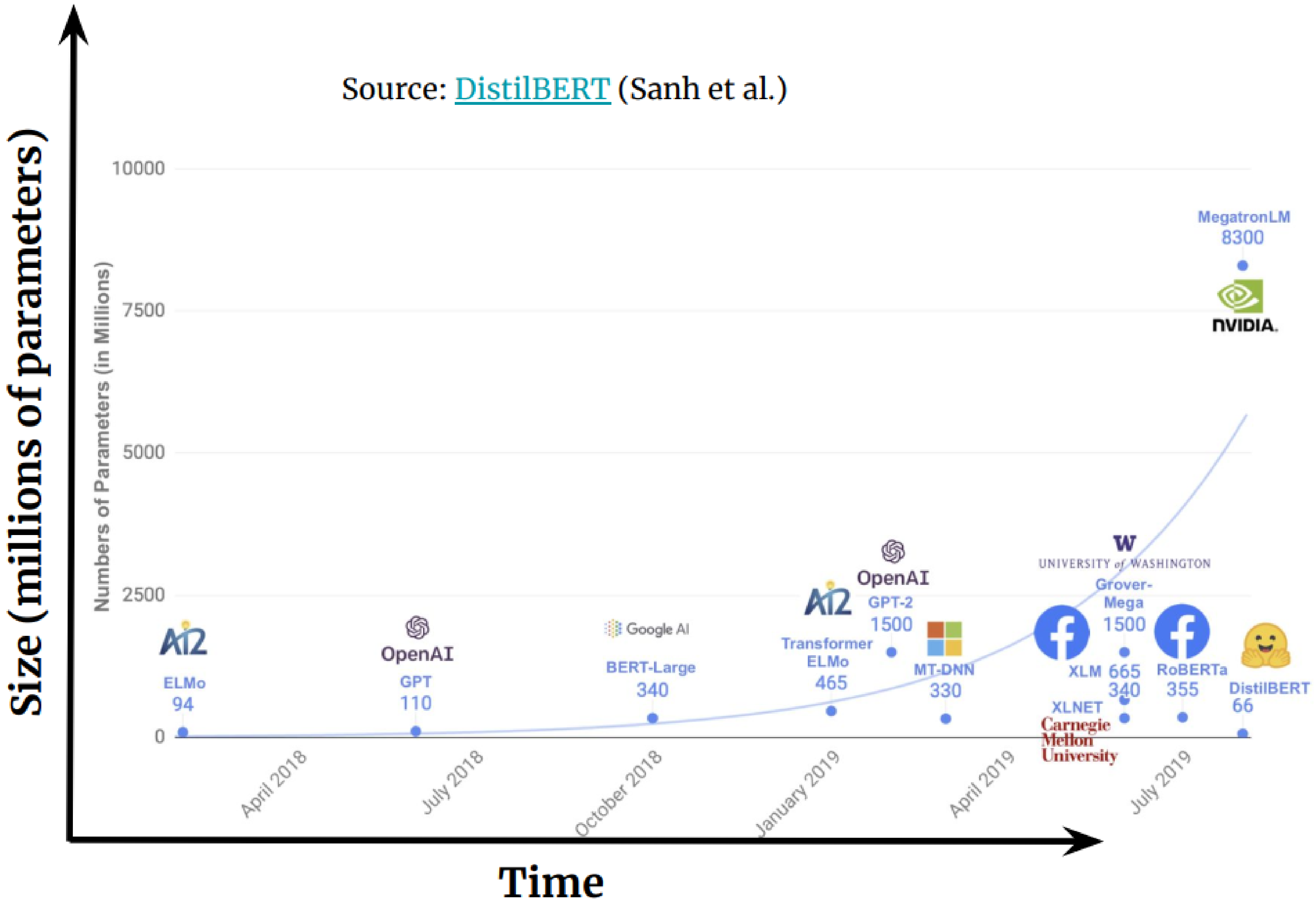
Sample 7 LLMs learn to predict one word at a time

Sample 8 LLMs learn to predict one word at a time

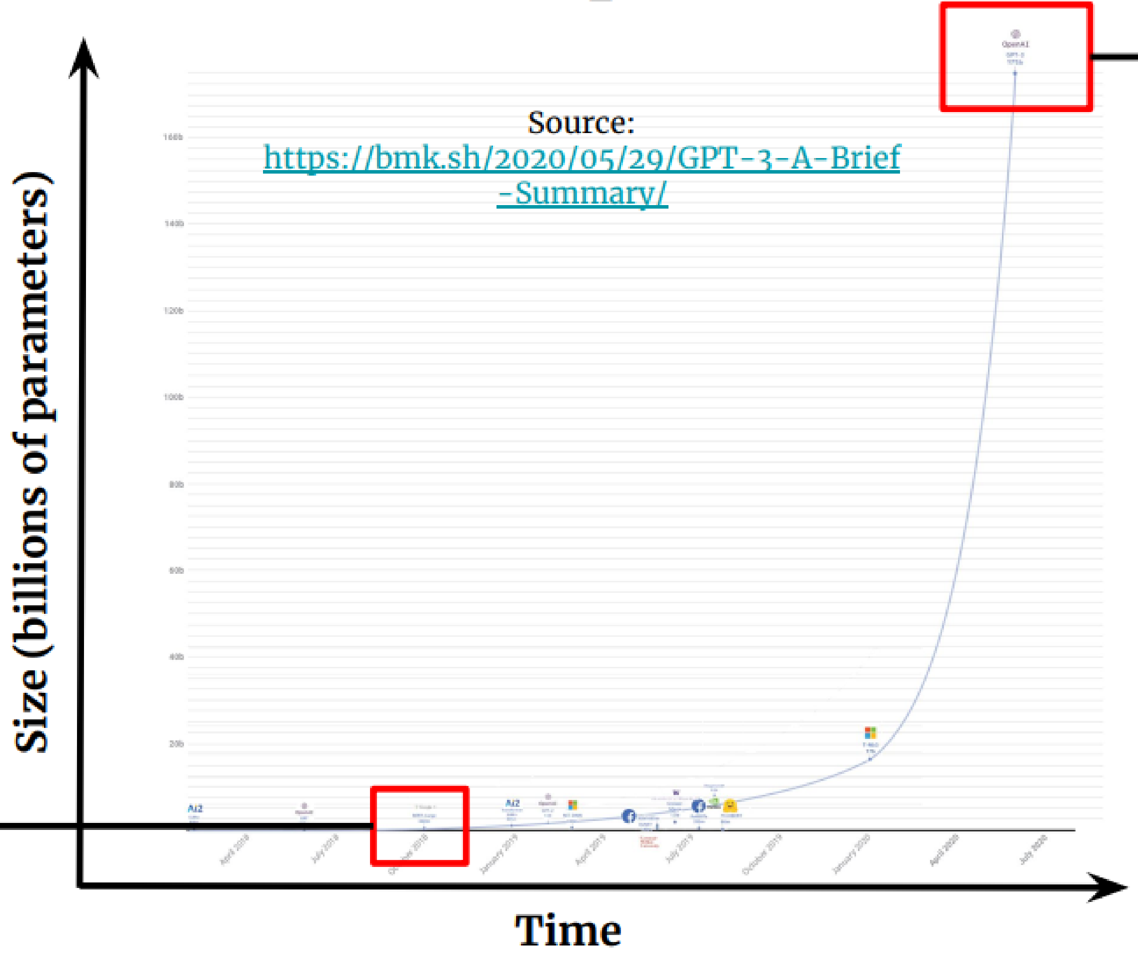
How do LLMs generate multi-word outputs?



LM Landscape pre GPT-3



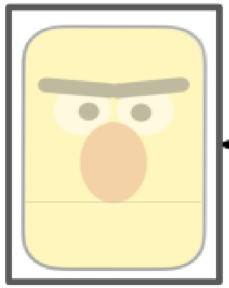
LM Landscape with GPT-3



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

175b params!
GPT-2 was 1.5b

340m params!

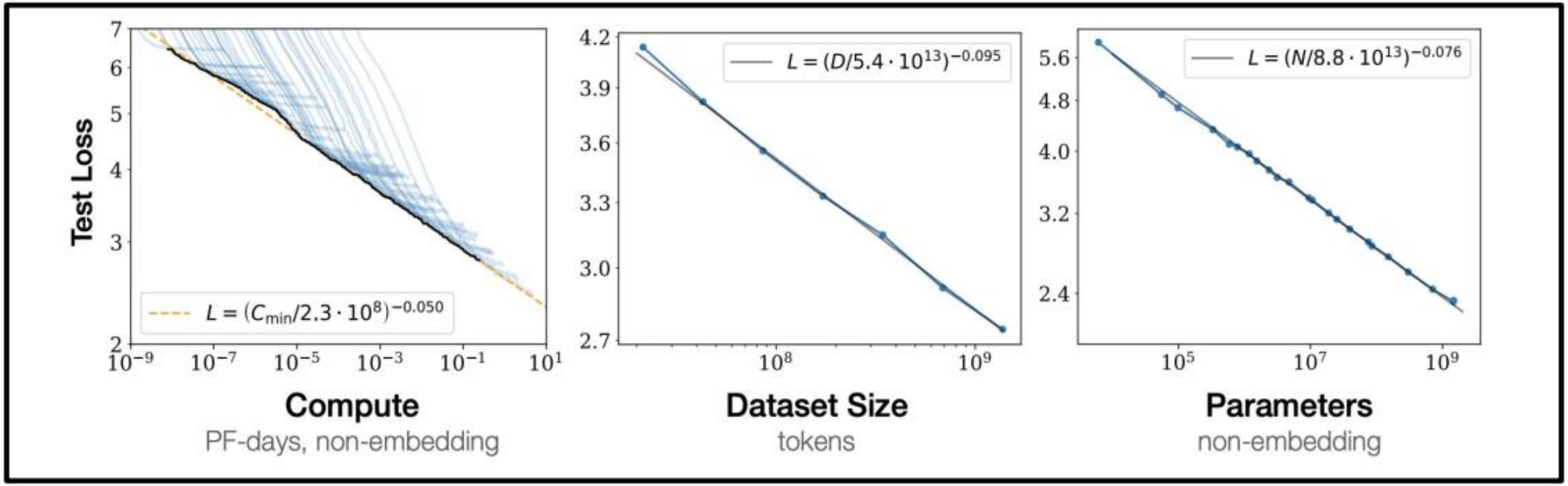


10

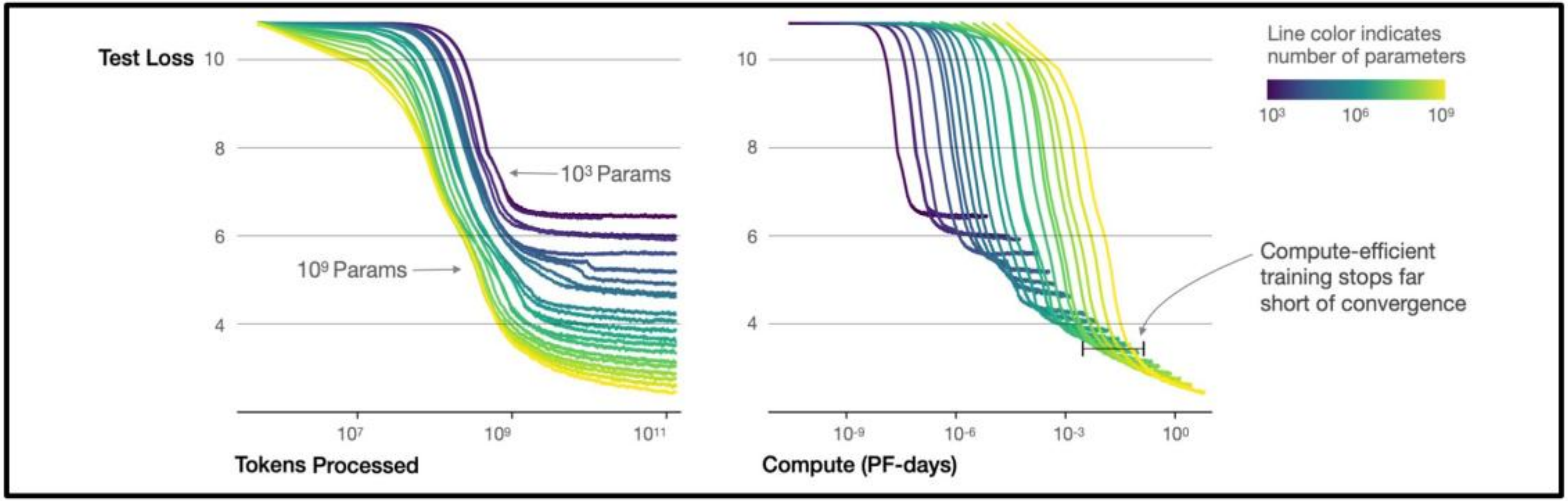
Why Scale?

- Study conducted by OpenAI → **Scaling Laws for Neural Language Models** (Kaplan et al. 2020)
- A few **key findings**:
  - Performance depends strongly on scale, weakly on model shape
  - Smooth power laws ($y = ax^k$) b/w empirical performance & N - parameters, D - dataset size, C - compute
  - Transfer improves with test performance
  - Larger models are more sample efficient

Bigger is Better!

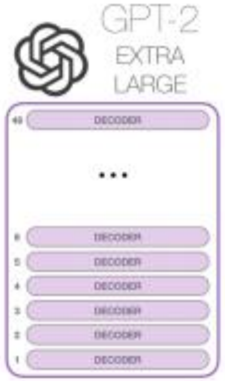


Bigger is Better!



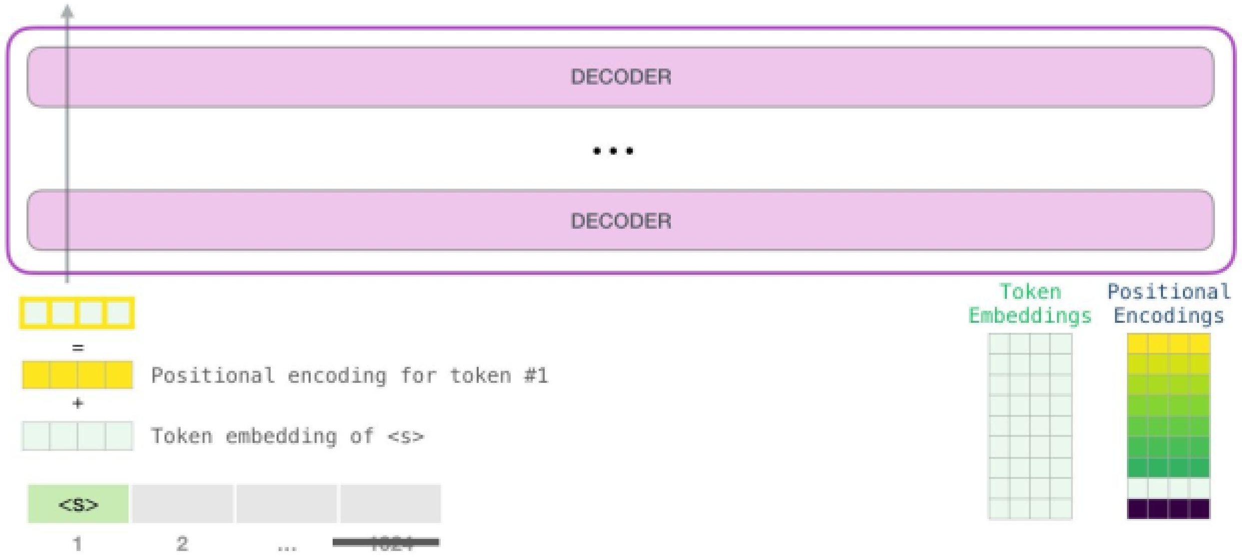
GPT-3 → GPT-2

= A very big GPT-2



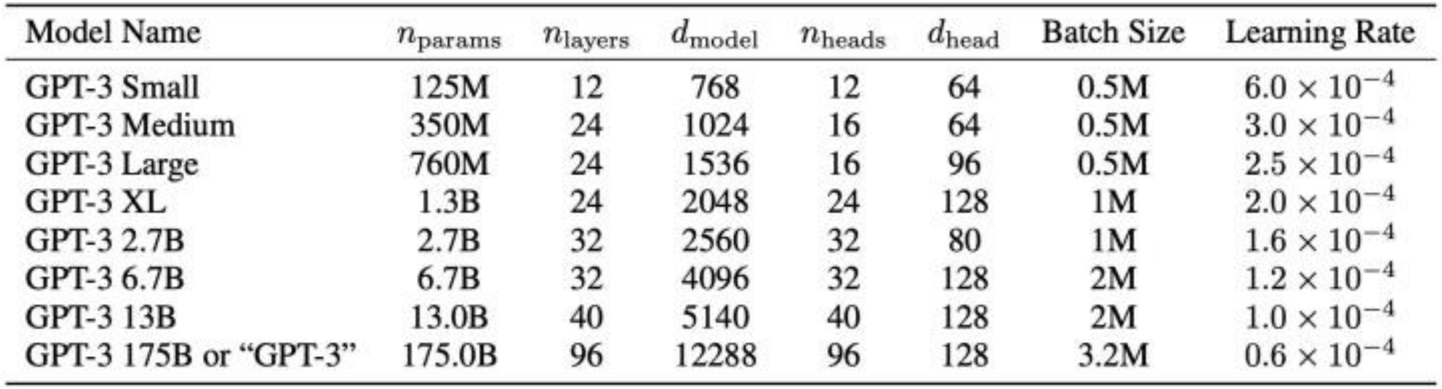
- more layers & parameters
- bigger dataset
- longer training
- larger embeddings
- larger context window → few-shot (whereas GPT-2 was zero-shot only)

GPT-3 is MASSIVE!

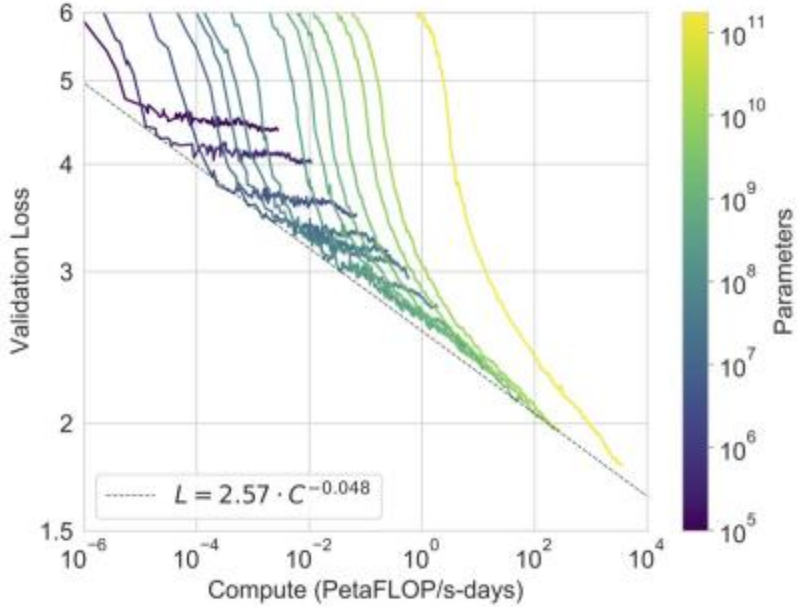


- 96 decoder blocks (2x GPT-2)
- Context size: 2048 (2x GPT-2)
- Embedding size: 12288 (~8x GPT-2)
- Params: 175b (~117x GPT-2)

GPT-3 is MASSIVE!



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}



- All models were trained on 300B tokens
- Follows power law argued in Kaplan et al.
- "GPT-3" → GPT-3 175B

In-Context Learning

	No Prompt	Prompt
Zero-shot (0s)	skicts = sticks	Please unscramble the letters into a word, and write that word: skicts = sticks

1-shot
(1s)
	chiar = chair
	skicts = sticks
		Please unscramble the letters into a word, and write that word:
		chiar = chair
		skicts = sticks

Few-shot
(FS)
	chiar = chair
	[...]
	pciinc = picnic
	skicts = sticks
		Please unscramble the letters into a word, and write that word:
		chiar = chair
		[...]
		pciinc = picnic
		skicts = sticks

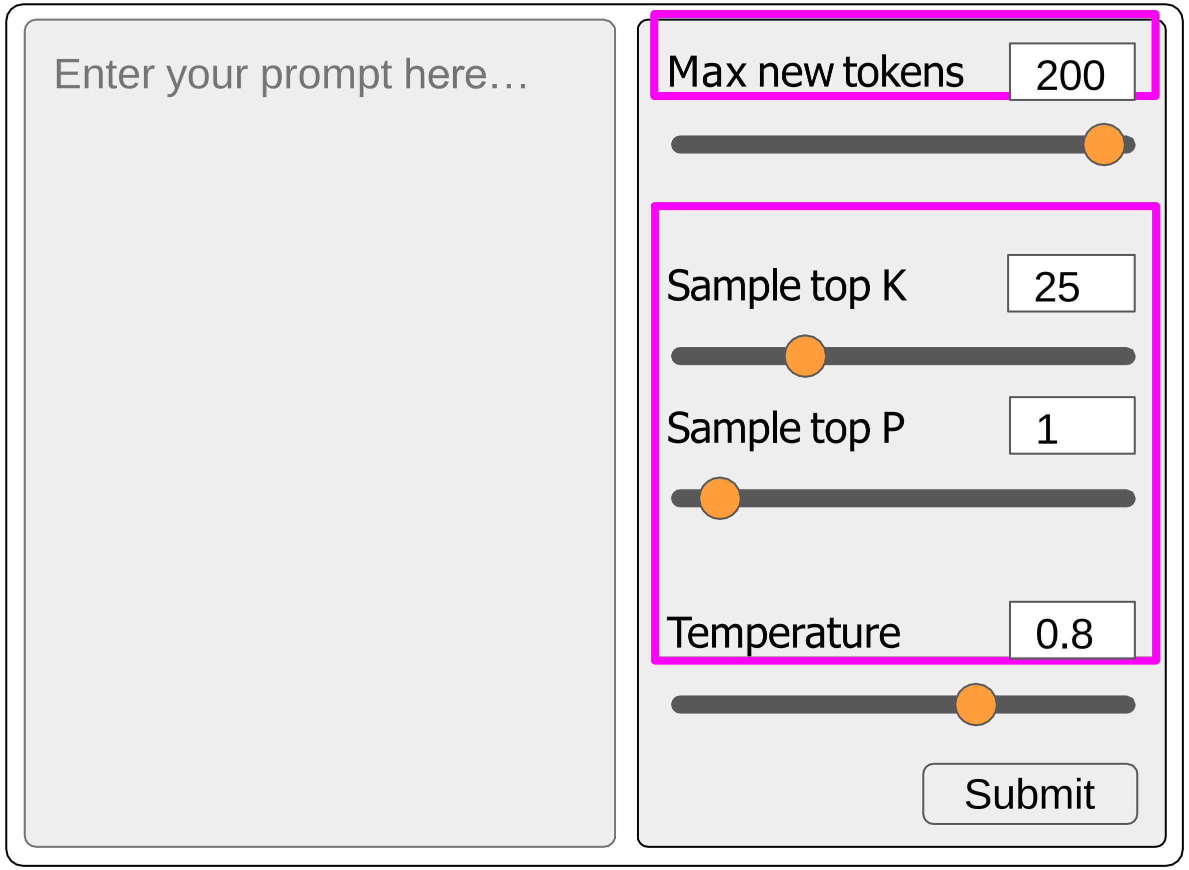
GPT training pipeline

Stage	Pretraining	Supervised Finetuning	Reward Modeling	Reinforcement Learning
Dataset	**Raw internet** text trillions of words low-quality, large quantity	**Demonstrations** Ideal Assistant responses, ~10-100K (prompt, response) written by contractors low quantity, high quality	**Comparisons** 100K –1M comparisons written by contractors low quantity, high quality	**Prompts** ~10K-100K prompts written by contractors low quantity, high quality
	↓	↓	↓	↓
Algorithm	**Language modeling** predict the next token	**Language modeling** predict the next token	**Binary classification** predict rewards consistent w preferences	**Reinforcement Learning** generate tokens that maximize the reward
	↓	↗ init from ↓	↗ init from ↓	↗ init from SFT use RM ↓
Model	**Base model**	**SFT model**	**RM model**	**RL model**
Notes	1000s of GPUs months of training ex: GPT, LLaMA, PaLM **can deploy this model**	1-100 GPUs days of training ex: Vicuna-13B **can deploy this model**	1-100 GPUs days of training	1-100 GPUs days of training ex: ChatGPT, Claude **can deploy this model**

Credits : @karpathy

Generative configuration parameters for inference

Generative configuration - inference parameters



Enter your prompt here…

Max new tokens 200

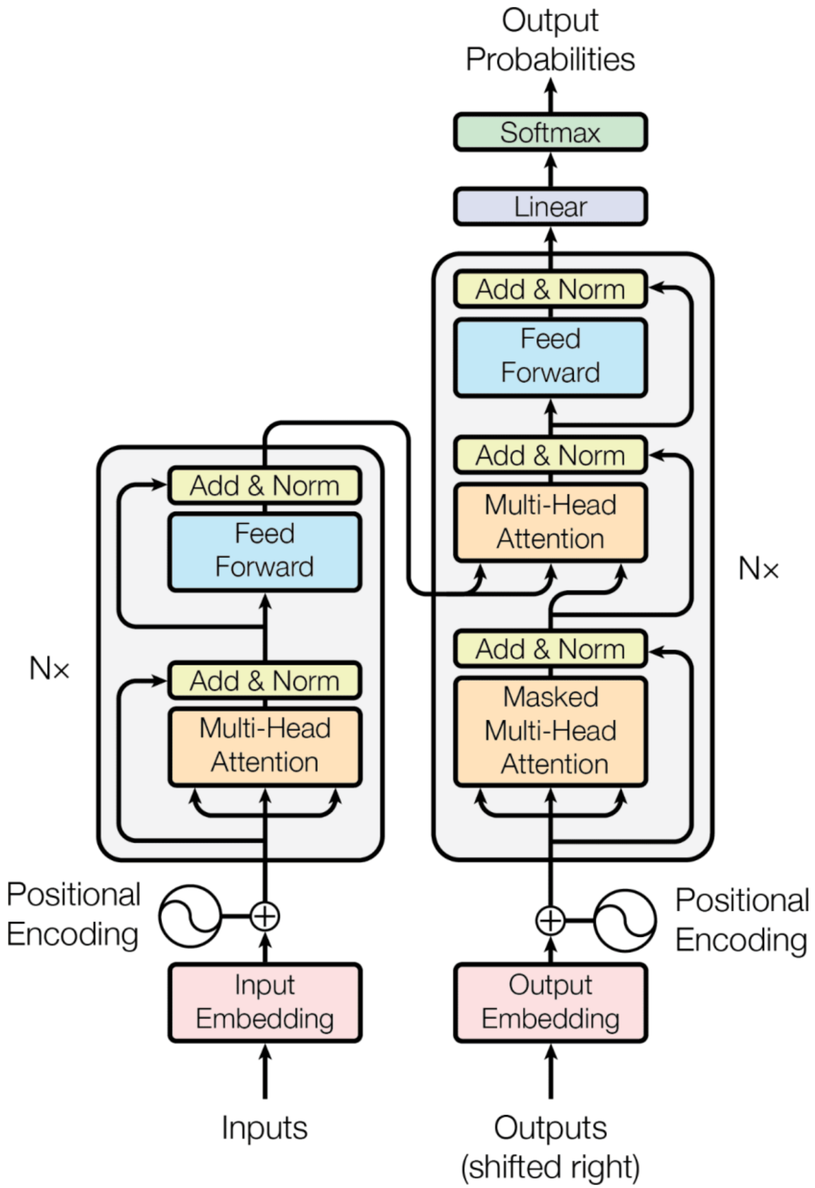
Sample top K 25

Sample top P 1

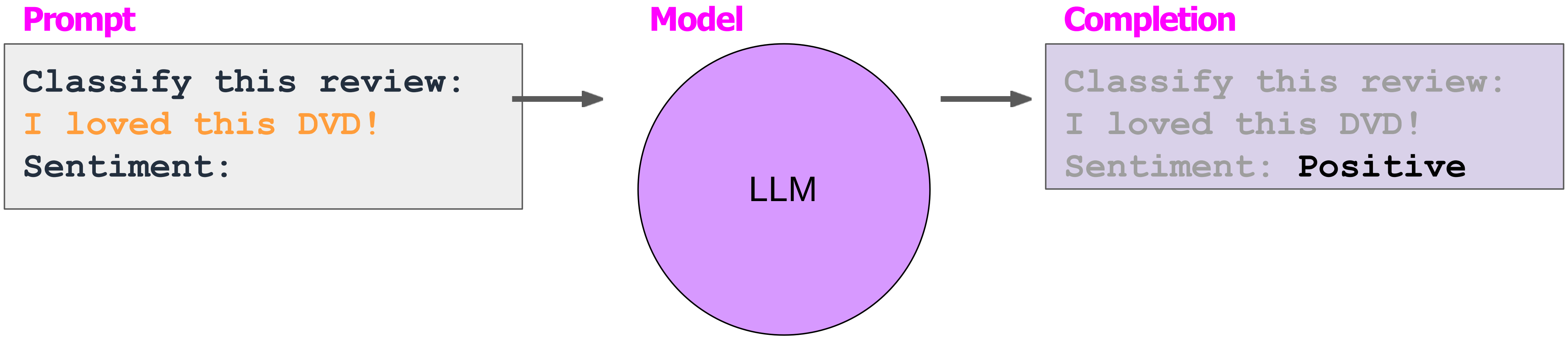
Temperature 0.8

Inference configuration parameters

Fine-tuning an LLM with instruction prompts



In-context learning (ICL) - zero shot inference



Prompt

```
Classify this review:
I loved this DVD!
Sentiment:
```

Model

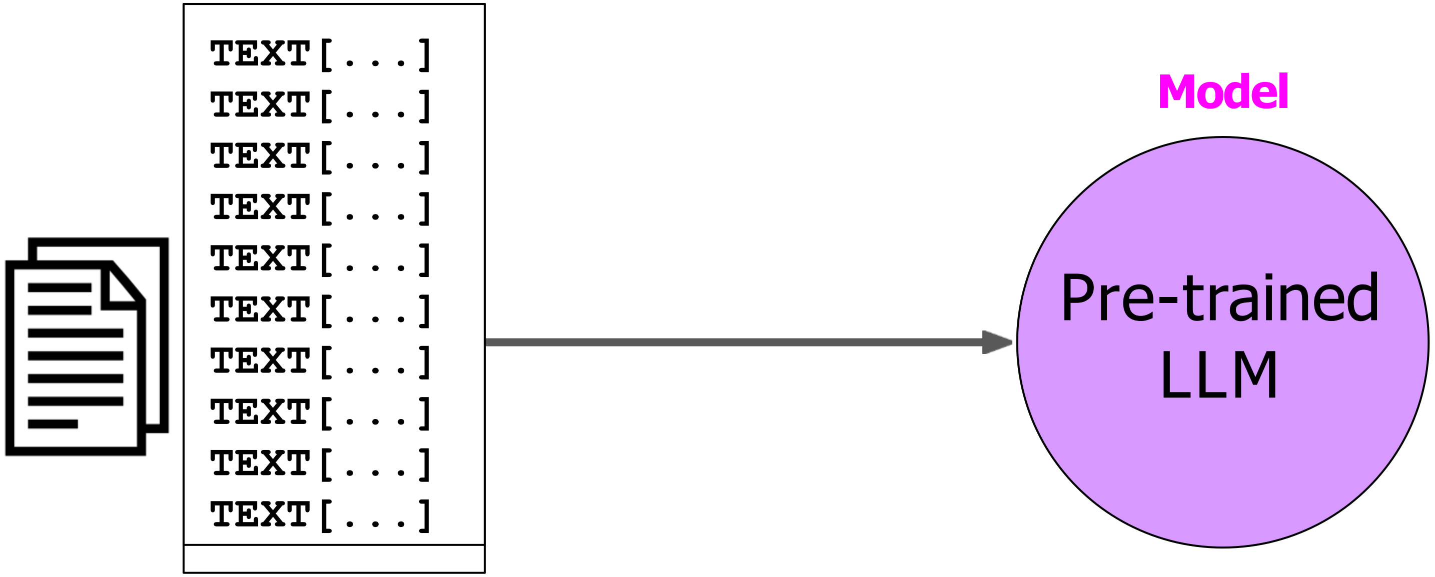
LLM

Completion

```
Classify this review:
I loved this DVD!
Sentiment: Positive
```

LLM fine-tuning at a high level

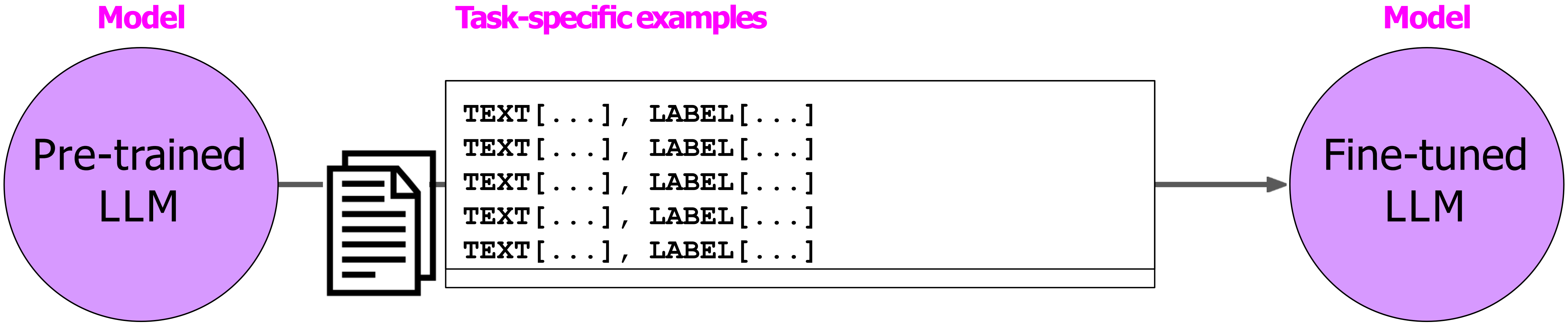
LLM pre-training



GB - TB - PB
of unstructured textual data

LLM fine-tuning at a high level

LLM fine-tuning

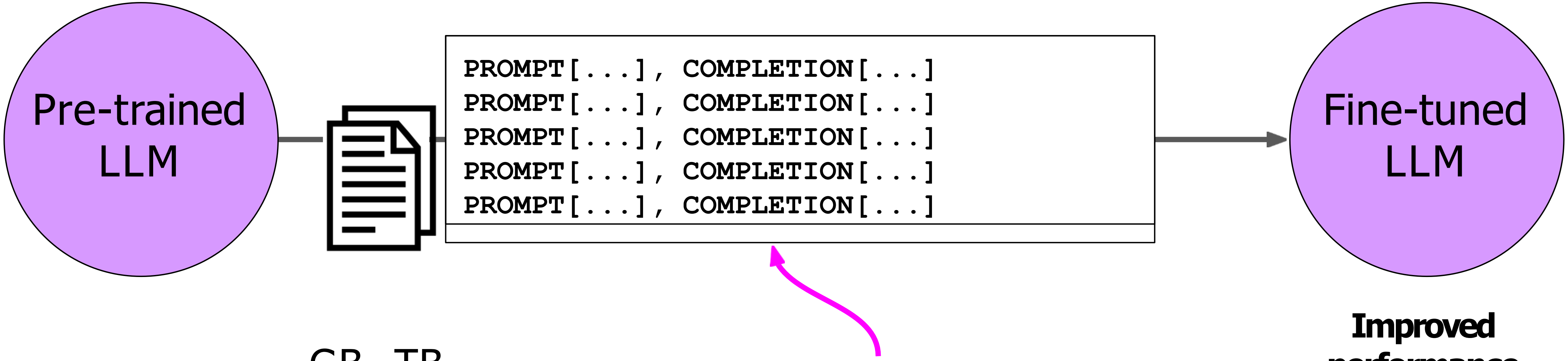


```
TEXT[...], LABEL[...]
TEXT[...], LABEL[...]
TEXT[...], LABEL[...]
TEXT[...], LABEL[...]
TEXT[...], LABEL[...]
```

GB - TB
of labeled examples for a specific
task or set of tasks

LLM fine-tuning at a high level

LLM fine-tuning



Model

Task-specific examples

Model

Pre-trained LLM

```
PROMPT[...], COMPLETION[...]
PROMPT[...], COMPLETION[...]
PROMPT[...], COMPLETION[...]
PROMPT[...], COMPLETION[...]
PROMPT[...], COMPLETION[...]
```

Fine-tuned LLM

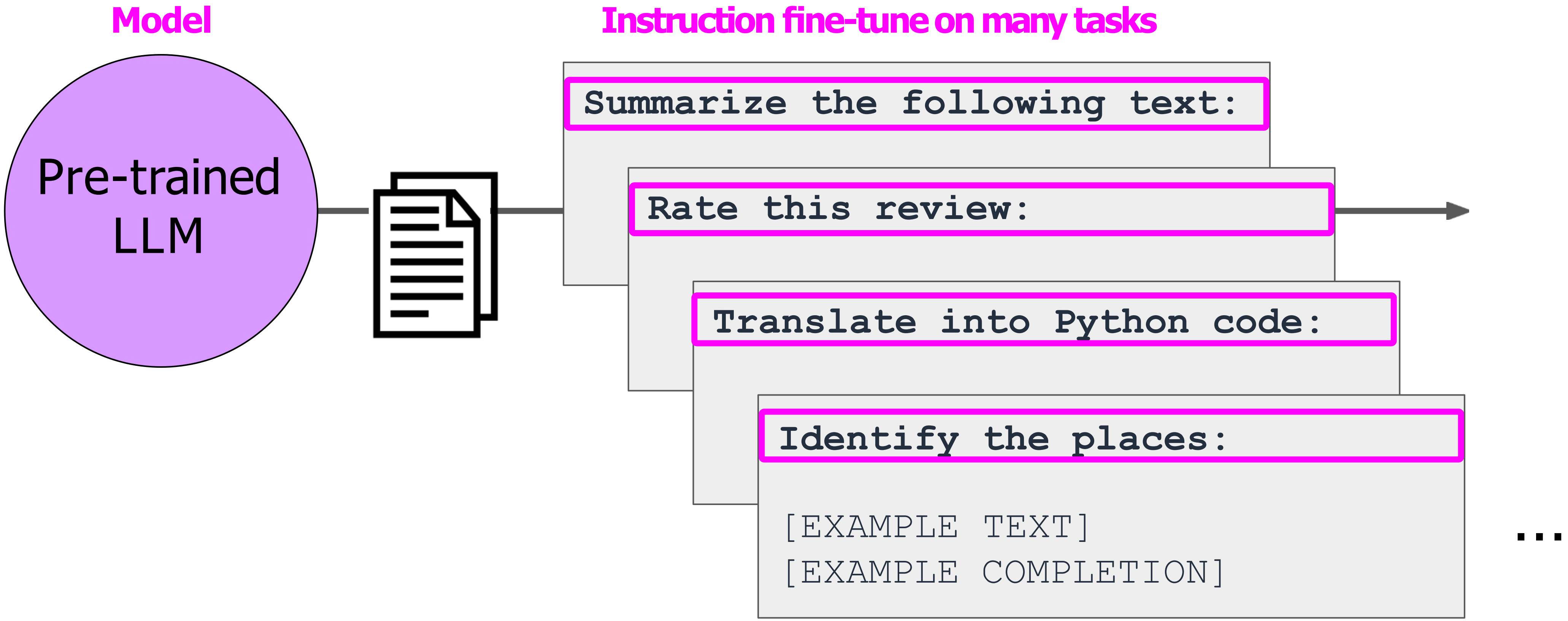
Improved performance

GB - TB
of labeled examples for a specific
task or set of tasks

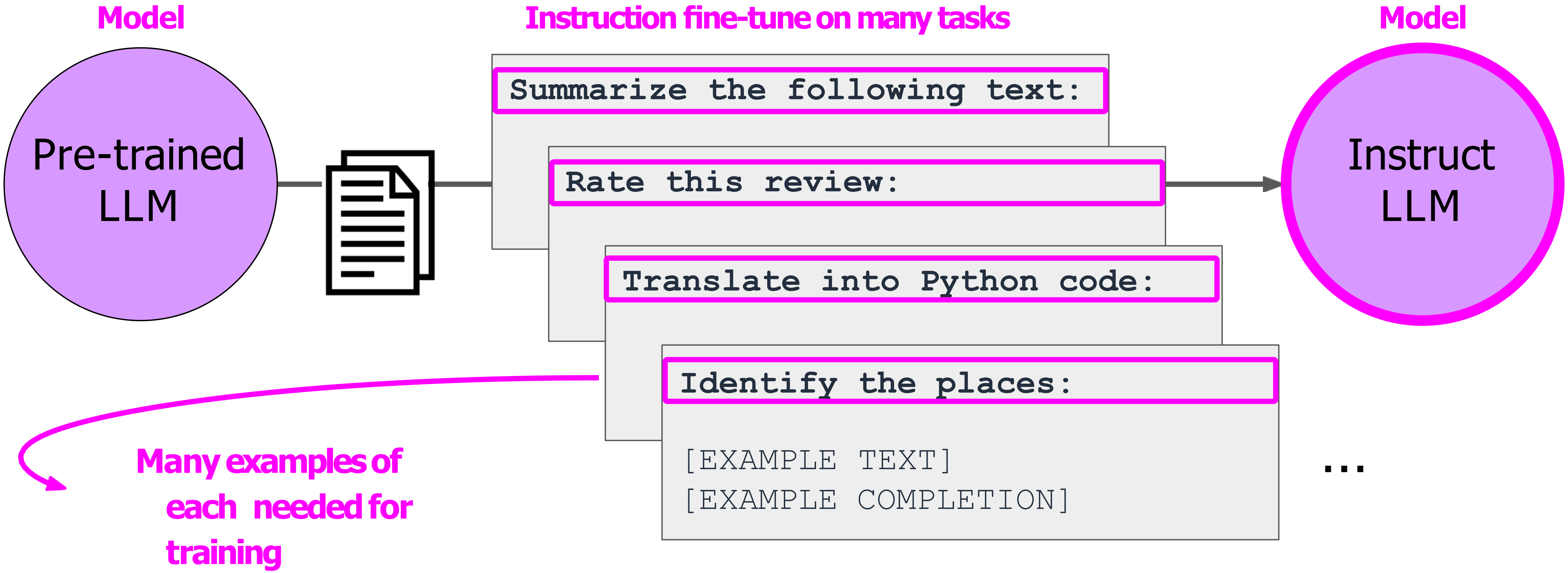
Prompt-completion pairs

Multi-task, instruction fine-tuning

Multi-task, instruction fine-tuning



Multi-task, instruction fine-tuning


Model
Pre-trained LLM
Instruction fine-tune on many tasks
Summarize the following text:
Rate this review:
Translate into Python code:
Identify the places:
[EXAMPLE TEXT]
[EXAMPLE COMPLETION]
...
Model
Instruct LLM
Many examples of each needed for training