Large Language Model (LLM)

Neural network designed to understand, generate & respond to human like text → - deep neural network trained on massive amount of text data

LLM vs earlier NLP models

LLM → can do a wide range of task

earlier NLP models → Designed to specific task like translation

How LLM works ?

- A Language model is a probabilistic model of text
- predicting a next word
- eg.

I wrote to the zoo to send me a pet. They sent me a ____

word	lion	elephant	dog	cat	panther	----
probability	0.1	0.1	0.3	0.2	0.05	

- Large Language model

Large → Large number of parameters

GPT-3 parameters , Source Dataset

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

To think

- What else LLM can do?
- How do we affect the distribution over the vocabulary?
- How do LLM generate text using these distributions?

We Should study

- What else LLM can do?

  LLM architexture

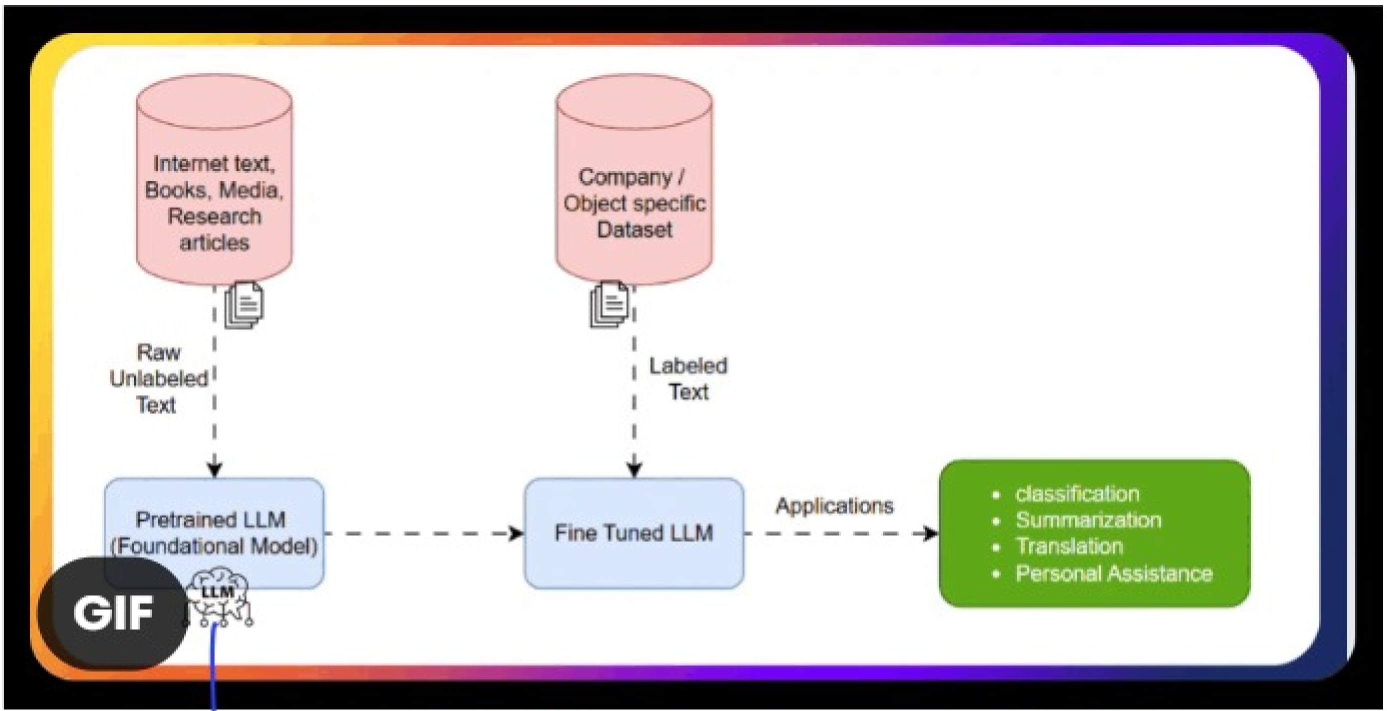
- How do we affect the distribution over the vocabulary?

  Prompting & training

- How do LLM generate text using these distributions

  Decoding

Understanding building blocks of LLM



→ ChatGPT
→ Claude

→ - Text completion
- Few-shot capabilites

Prompting & Prompt Engineering

- The simplest way to affect the distribution over the vocabulary is to change the prompt
- prompt

eg.
I wrote to the zoo to send me a pet. They sent me a ____

word	lion	elephant	dog	cat	panther	----
probability	0.1	0.1	0.3	0.2	0.05	

eg.
I wrote to the zoo to send me a pet. They sent me a little ____

word	lion	elephant	dog	cat	panther	----
probability	0.03	0.02	0.45	0.4	0.05	

In-context Learning.

- Prompting an LLM with instructions and/or demonstrations of the task it ment to complete.
- K-shot prompting - explicitly providing K-examples of the intended task in the prompt

zero shot prompting

```
Translate English to French:
cheese =>
```

one shot prompting

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

Few-shot prompting

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

Chain-of-thought

- prompt the LLM to emit intermediate reasoning steps

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

[Wei et al, 2022]

Least-to-most

Prompt the LLM to decompose the problem and solve easy first

```
Q: "think, machine, learning"
A: "think", "think, machine", "think, machine, learning"
The last letter of "think" is "k". The last letter of "machine" is
"e". Concatenating "k", "e" leads to "ke".
"think, machine" outputs "ke". The last letter of "learning is "g".
So, "think, machine, learning" outputs "keg".
```

[Zhou et al, 2022]

What makes LLM so good?

- What is a secret sauce?

What makes LLM so good?

Transformer

- 2017
- Translation
- 1,42,180 citations

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

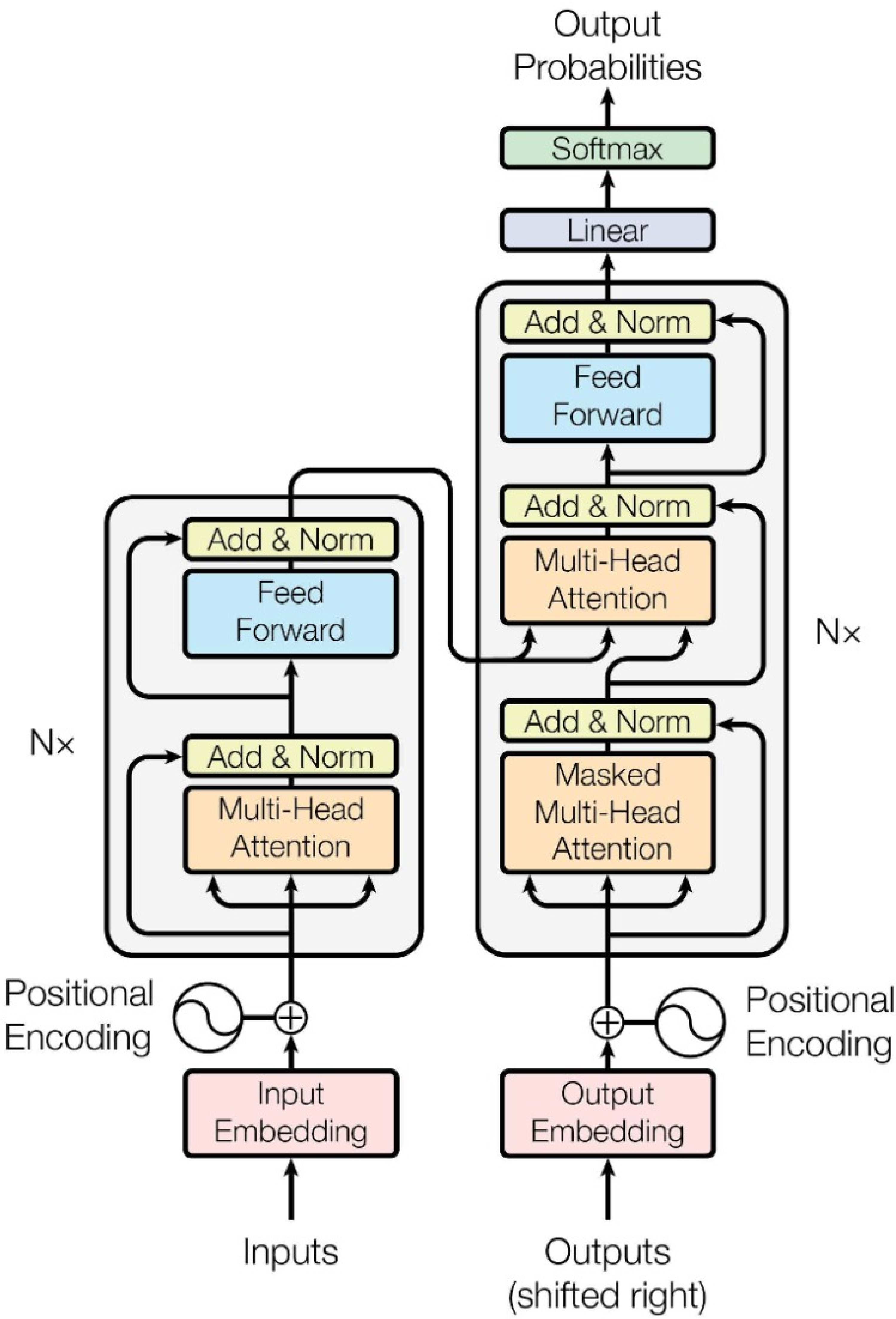
The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer,

NIPS papers
https://papers.nips.cc › paper › 7181-attention-is-all-yo...

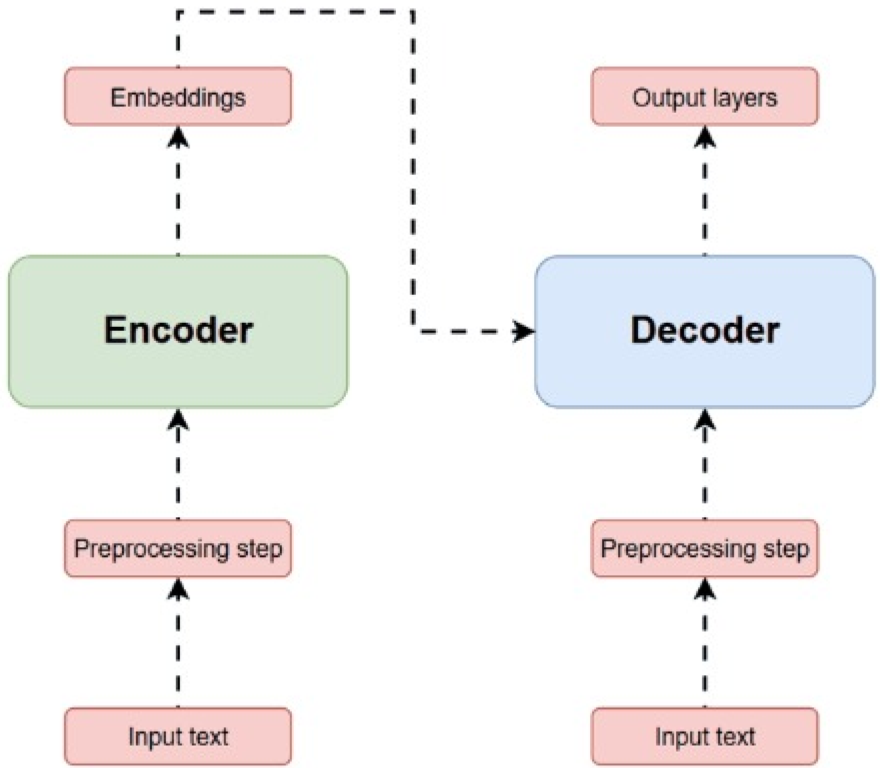
Attention is All you Need

by A Vaswani · 2017 · Cited by 142180 — **Attention is All you Need**. Part of Advances in Neural Information Processing Systems 30 (NIPS 2017) · Bibtex Metadata Paper Reviews....

Transformer Architecture



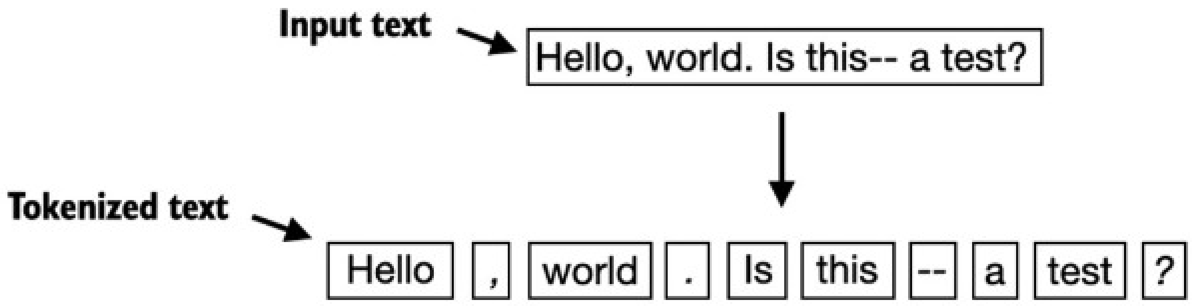
Simplified Transformer Architecture



GIF

Tokenizing text

- The concept of how we split the text into indivisual token
- Simplest one is word based tokenizer



- Oxford English Dictionary
  around - 170,000 → in current use
  around - 47,000 → obsolute words
- Character based tokenizer
  256 characters in English
- Byte Pair Encoding (BPE)
  single token represent roughly
  4 charachters of text
  GPT 3.5 & GPT 4 has 1,00,256 token

Example [edit]

Suppose the data to be encoded is

```
aaabdaaabac
```

The byte pair "aa" occurs most often, so it will be replaced by a byte that is not used in the data, such as "Z". Now there is the following data and replacement table:

```
ZabdZabac
Z=aa
```

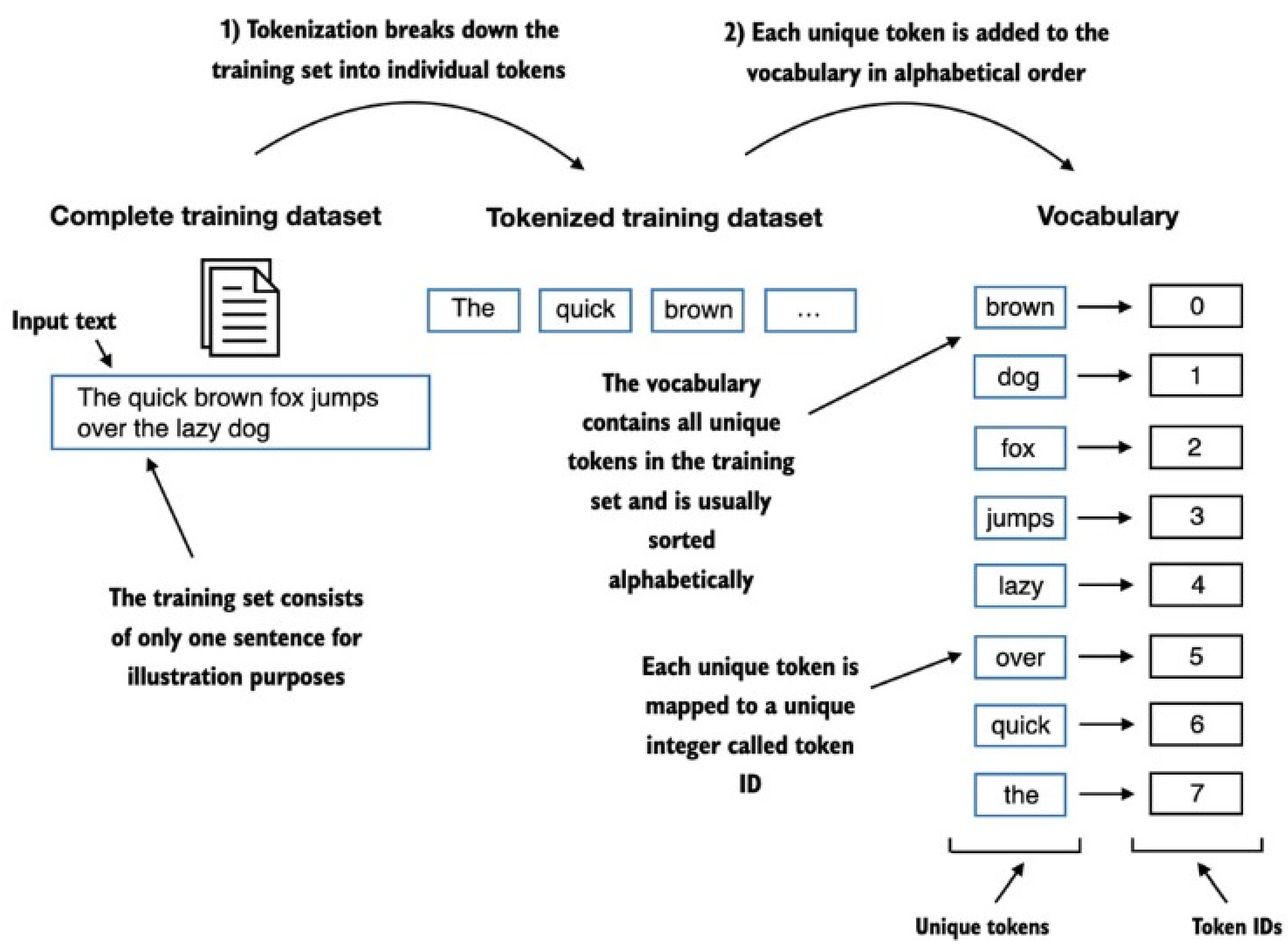
Then the process is repeated with byte pair "ab", replacing it with "Y":

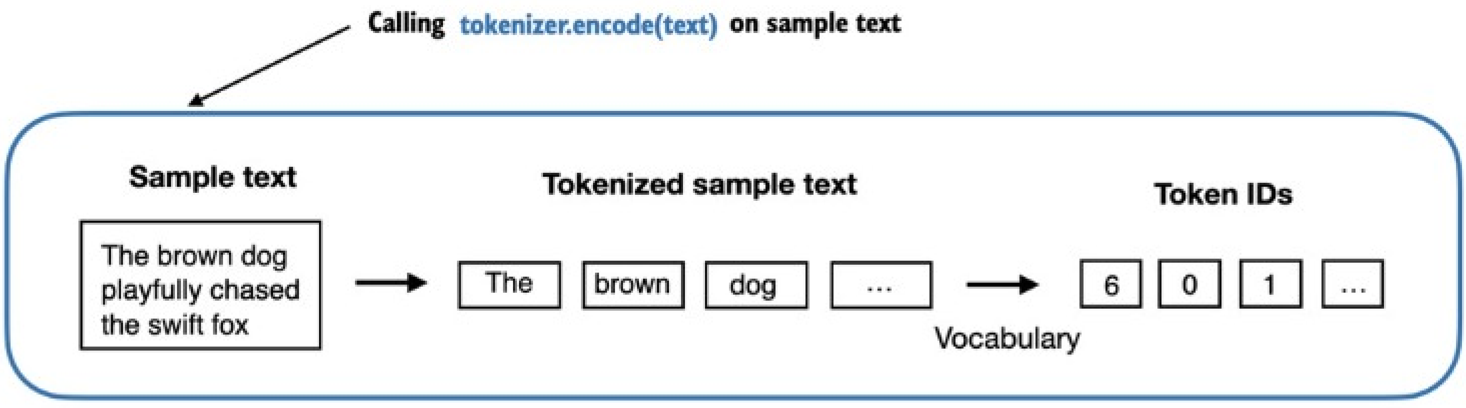
```
ZYdZYac
Y=ab
Z=aa
```

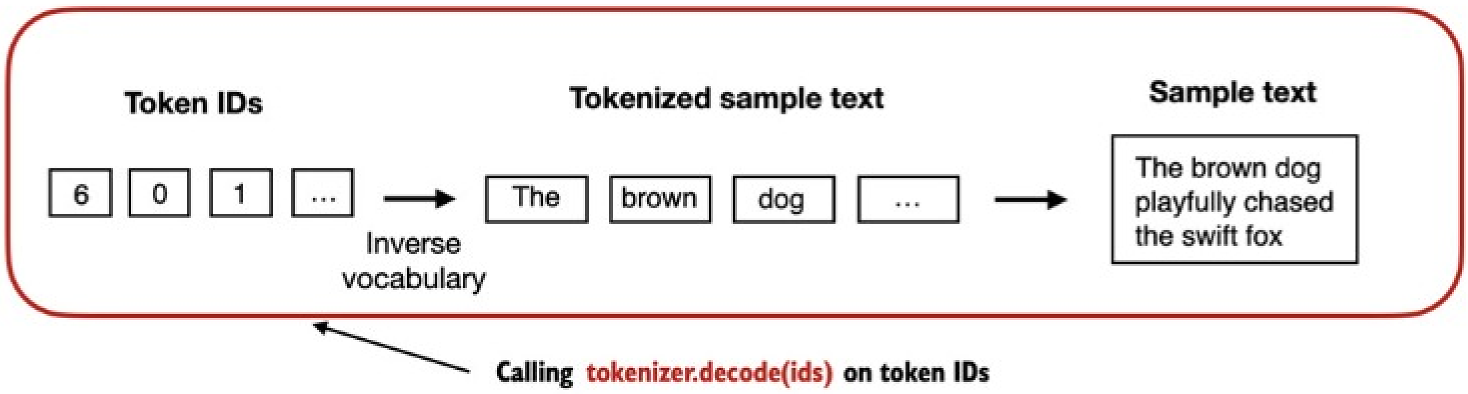
The only literal byte pair left occurs only once, and the encoding might stop here. Alternatively, the process could continue with recursive byte pair encoding, replacing "ZY" with "X":

```
XdXac
X=ZY
Y=ab
Z=aa
```

Converting token into token-ID



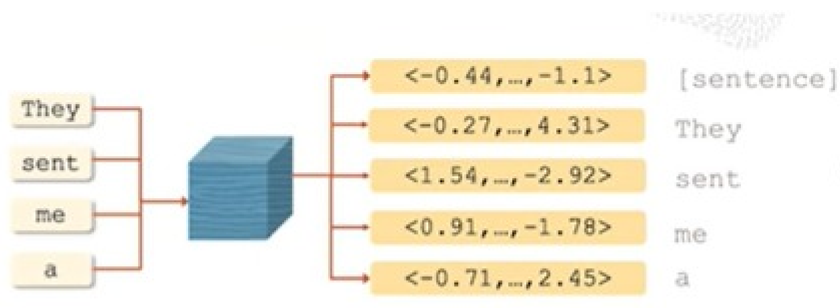


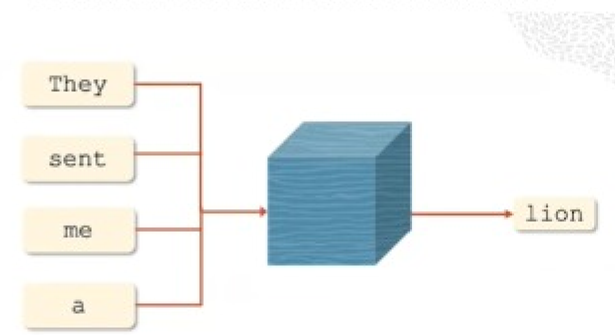


Encoder model, Decoder model

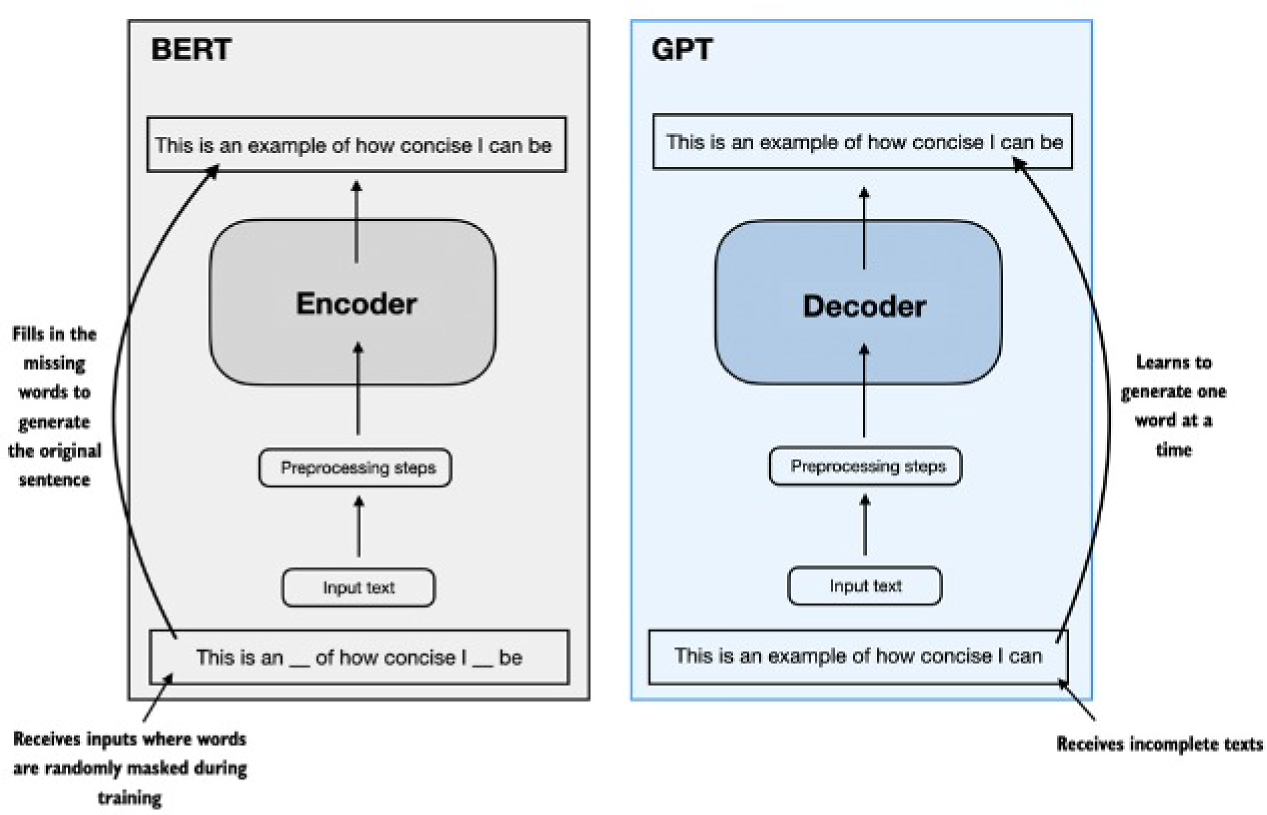
Encoder - model that convert a sequence of words to an embedding (vector representation)

Decoder - model take a sequence of words and output next word





BERT vs GPT

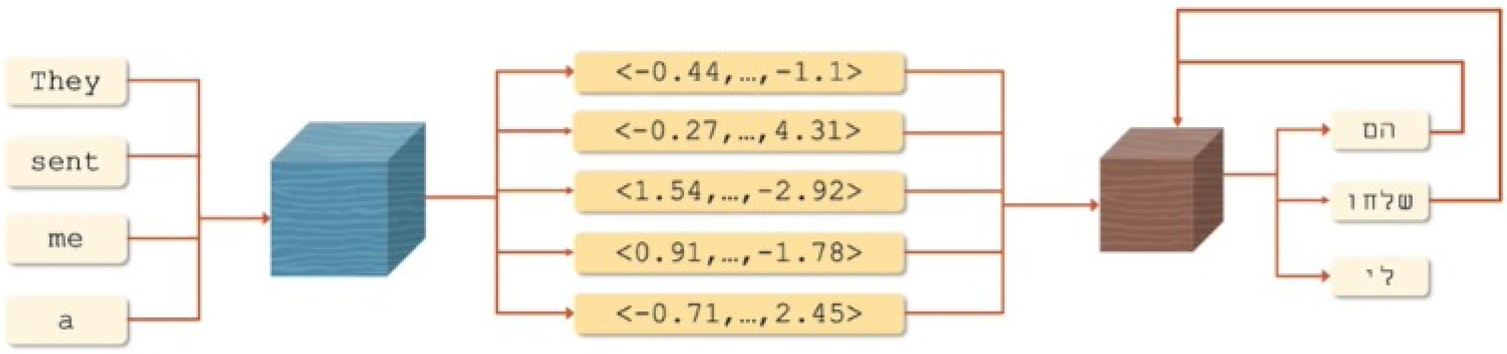


- Bidirectional Encoder Representation from Transformer

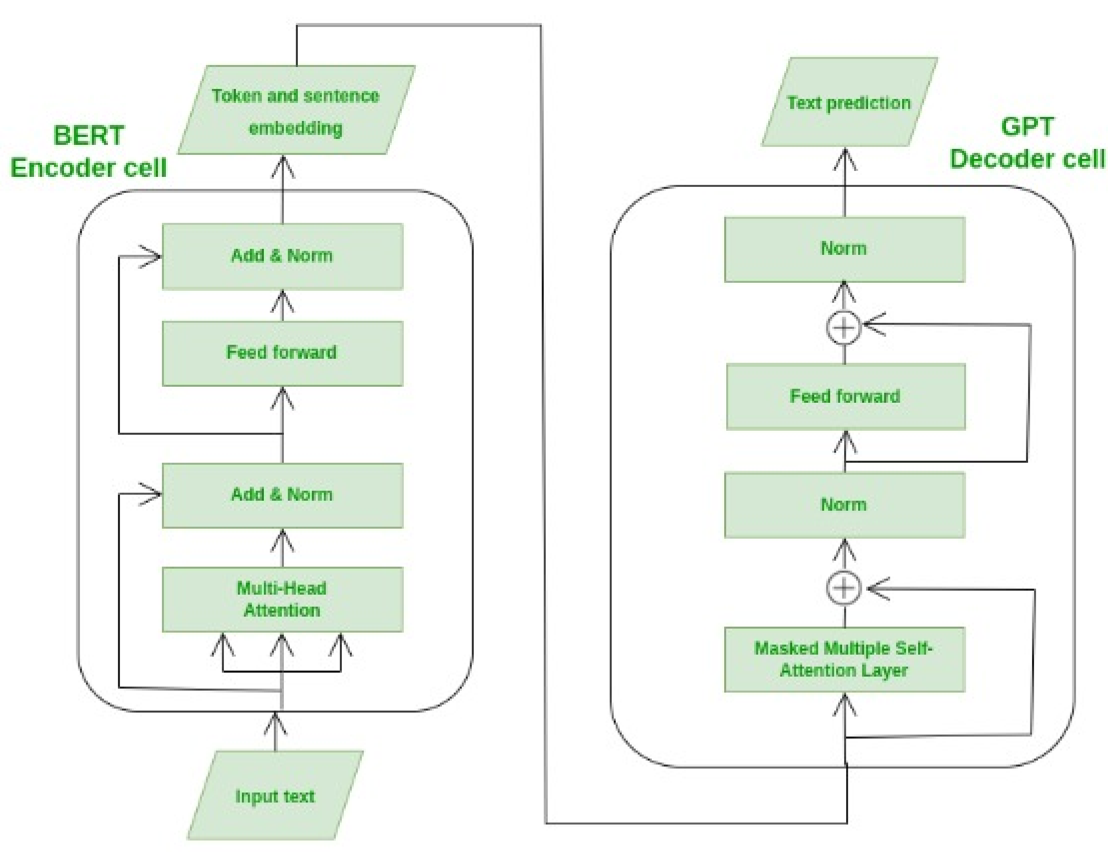
- Generative Pre-trained Transformer

Encoder - Decoder model

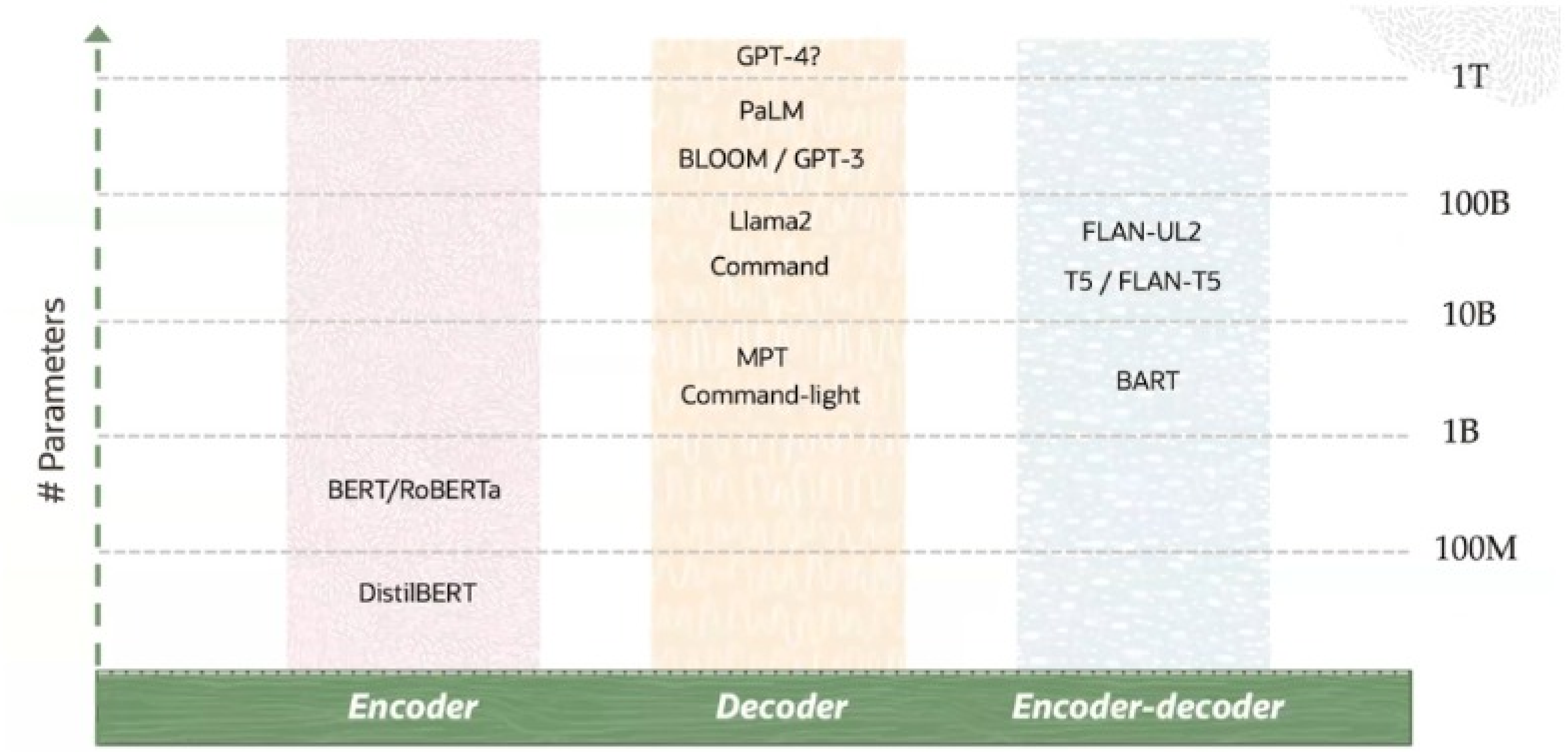
Encoders - Decoders
encodes a sequence of words and use the encoding + to output a next word.



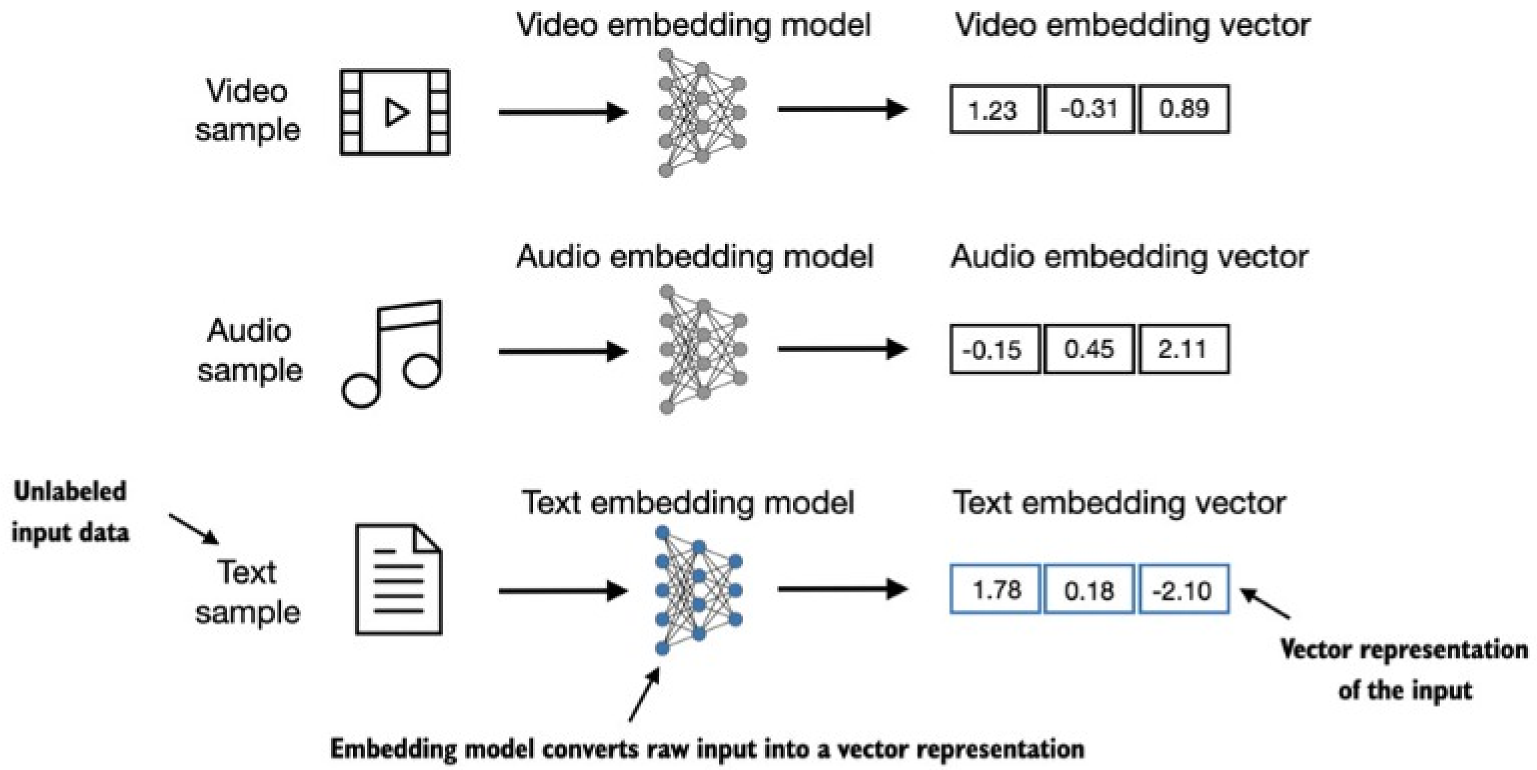
BART model



- Bidirectional and
  Auto
  Regressive
  Transformer



What encoder actually do?



How GPT works

- GPT models are simply trained on next-word prediction task

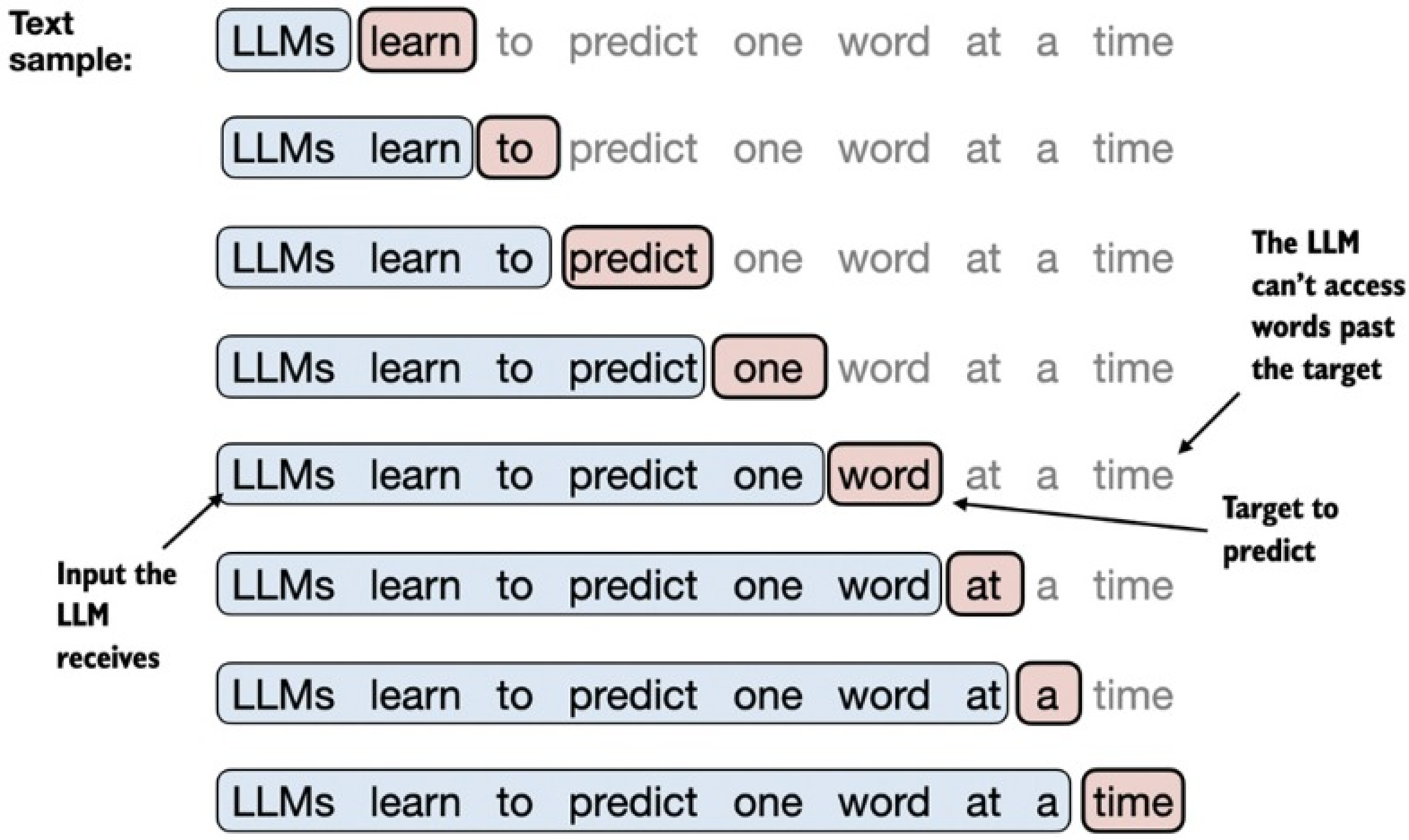
The lion roam in the jungle

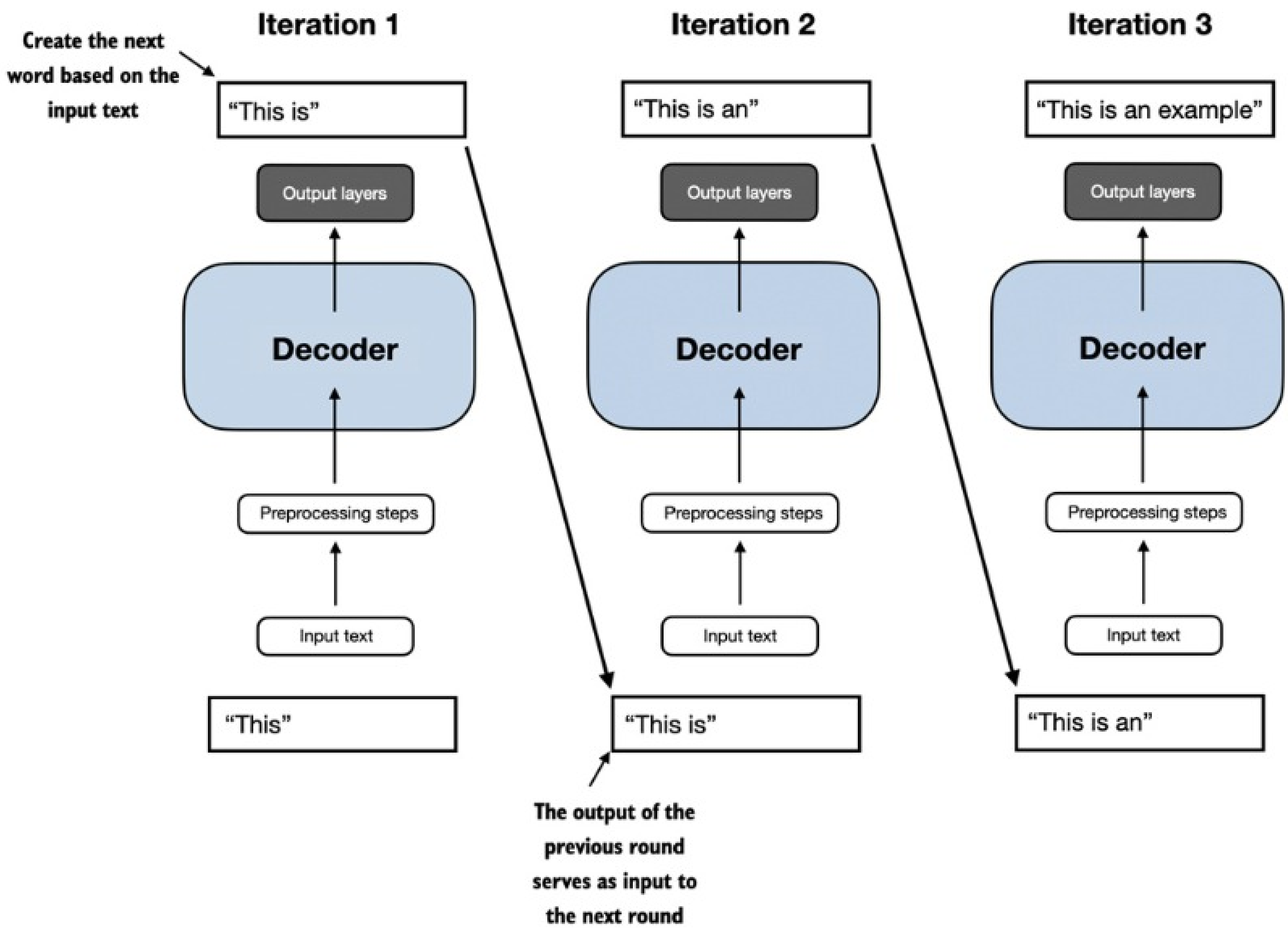
jungle → next word

Generated training examples

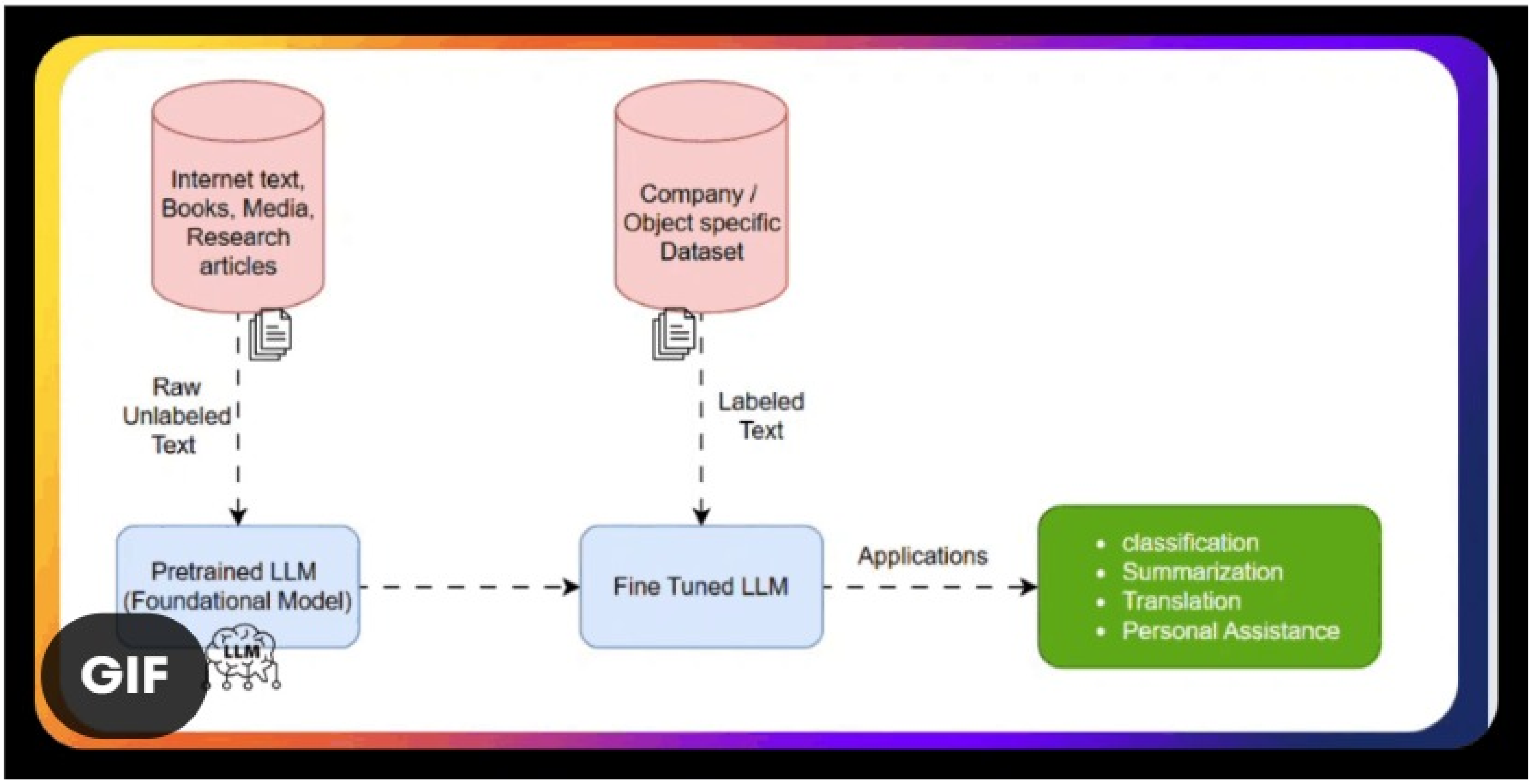
Example #	Input (features)	Correct output (labels)
1	Second law of robotics :	a
2	Second law of robotics : a	robot
3	Second law of robotics : a robot	must
...		

- Next word prediction
  self supervised learning
- Auto regressive model:
  use previous output as input for future predictions.


Text sample:
LLMs learn to predict one word at a time
The LLM can't access words past the target
Target to predict
Input the LLM receives


Iteration 1
Iteration 2
Iteration 3
Create the next word based on the input text
"This is"
"This is an"
"This is an example"
Output layers
Decoder
Preprocessing steps
Input text
"This"
"This is"
"This is an"
The output of the previous round serves as input to the next round

Understanding building blocks of LLM



Fine-tuning LLM for domain specific task

Training Style	Modifies	Data	Summary
Fine-tuning (FT)	All parameters	Labeled, task-specific	Classic ML training
Param. Efficient FT	Few, new parameters	Labeled, task-specific	+Learnable params to LLM
Soft prompting	Few, new parameters	Labeled, task-specific	Learnable prompt params
(cont.) pre-training	All parameters	unlabeled	Same as LLM pre-training