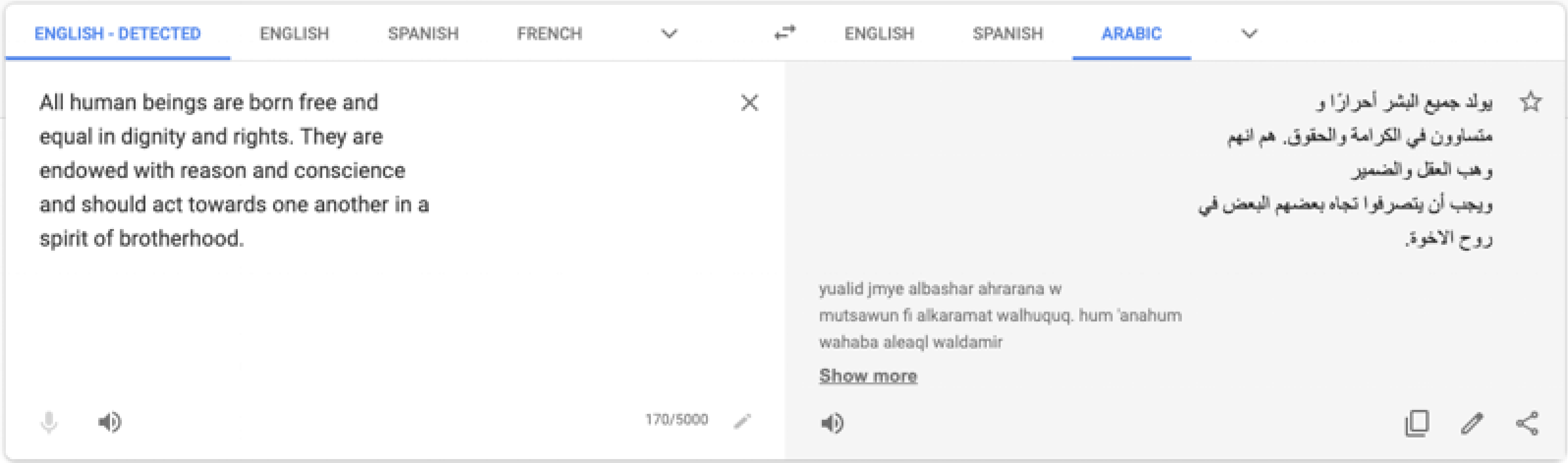
Transformers

Transformers

- Have completely blown up in the past several years
- Are the basis for most state of the art papers
- You WILL hear of them!

Machine Translation



1950s Machine Translation

Neural Machine Translation

- MT with a single neural network
- The architecture is called a *sequence-to-sequence* (*seq2seq*) model or an *encoder-decoder* model

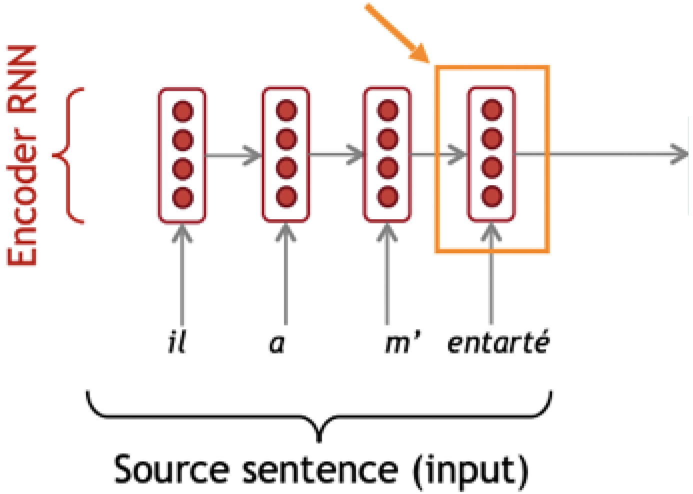
The sequence-to-sequence model

il a m' entarté

Source sentence (input)

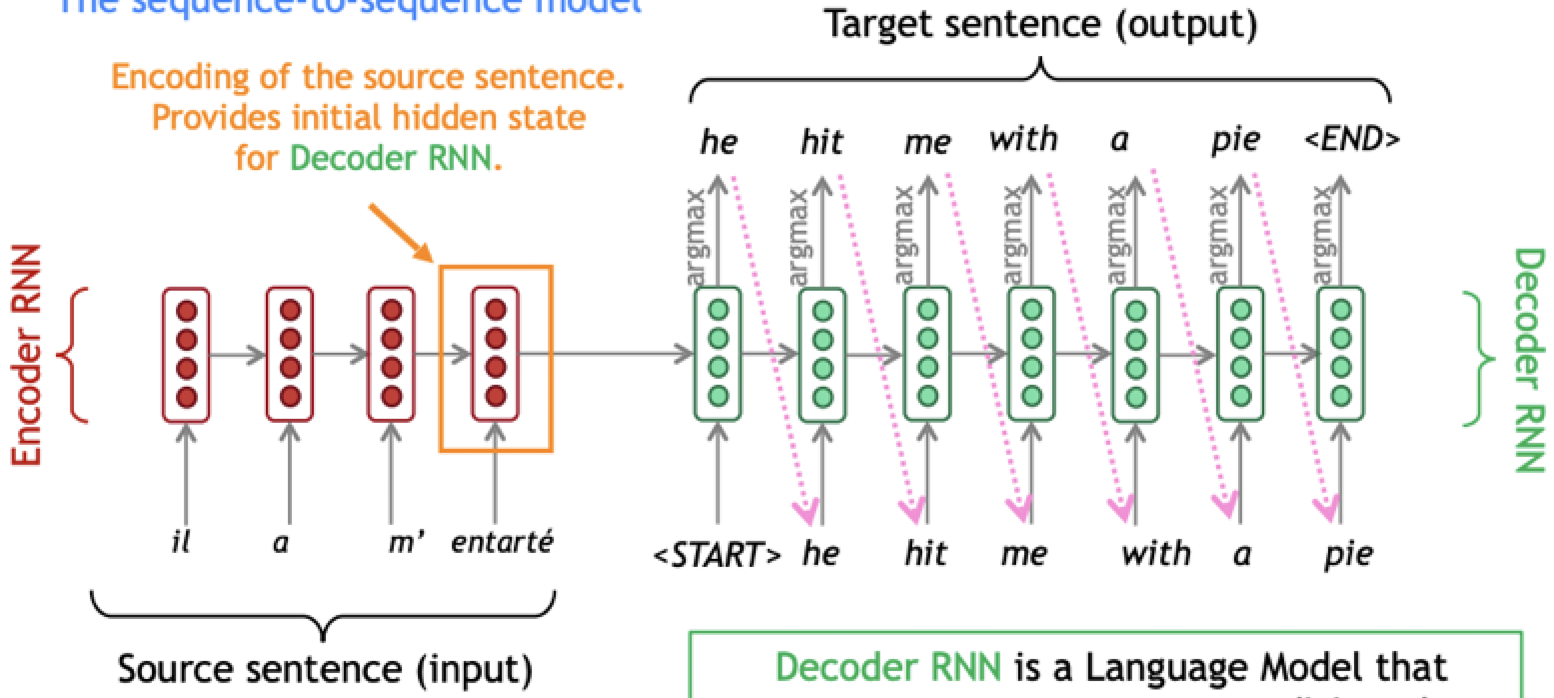
The sequence-to-sequence model

Encoding of the source sentence.
Provides initial hidden state
for Decoder RNN.



Encoder RNN produces an encoding of the source sentence.

The sequence-to-sequence model

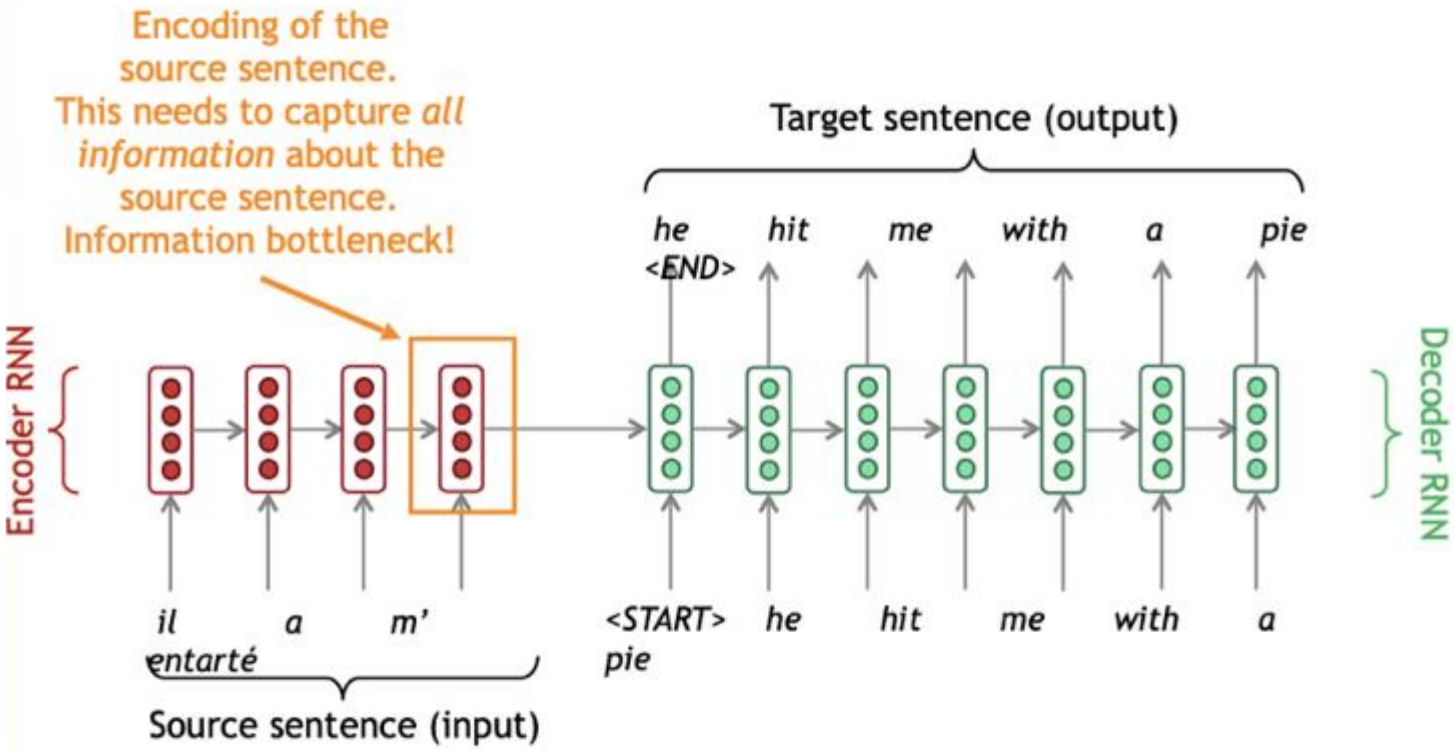


Encoder RNN produces an *encoding* of the source sentence.

Decoder RNN is a Language Model that generates target sentence, *conditioned on encoding*.

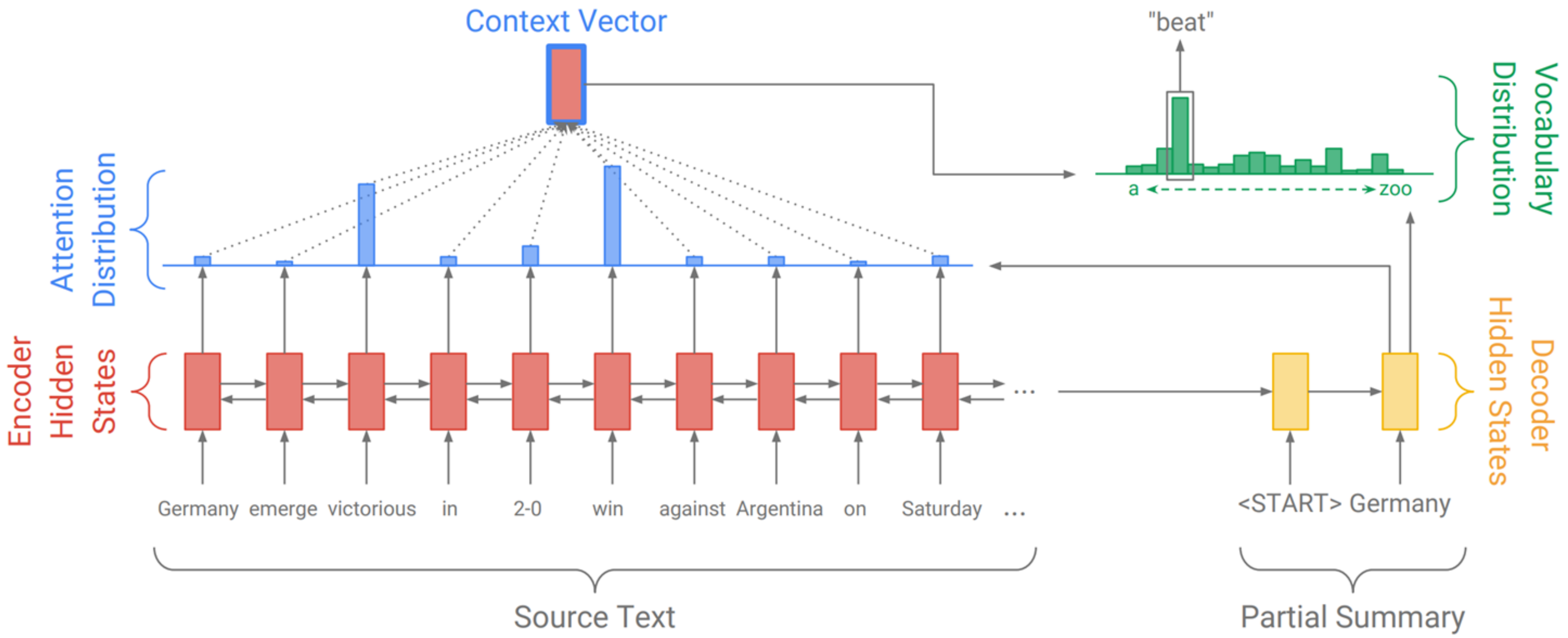
Note: This diagram shows **test time** behavior: decoder output is fed in ·····> as next step's input

Sequence-to-sequence: the bottleneck problem



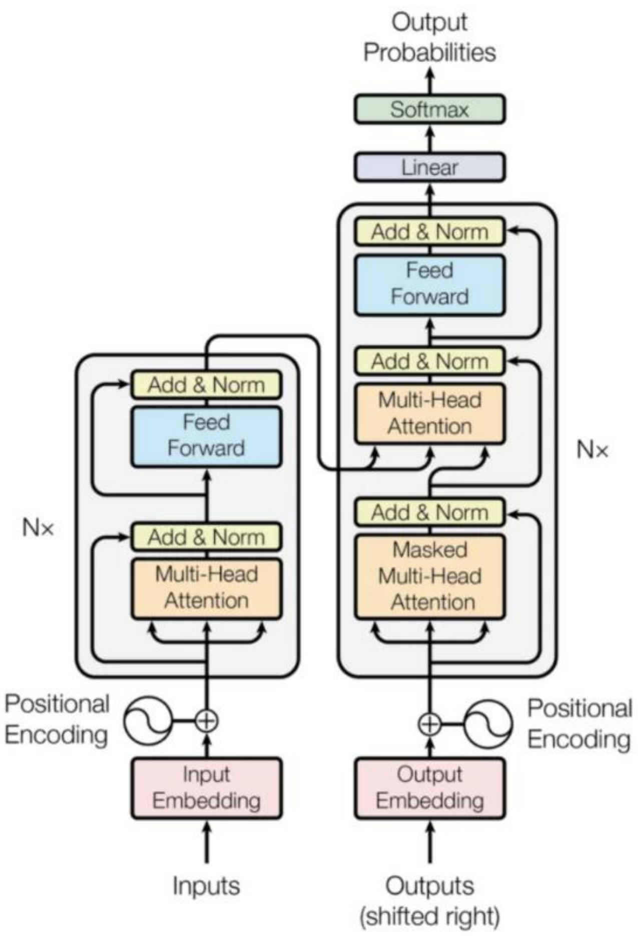
Attention

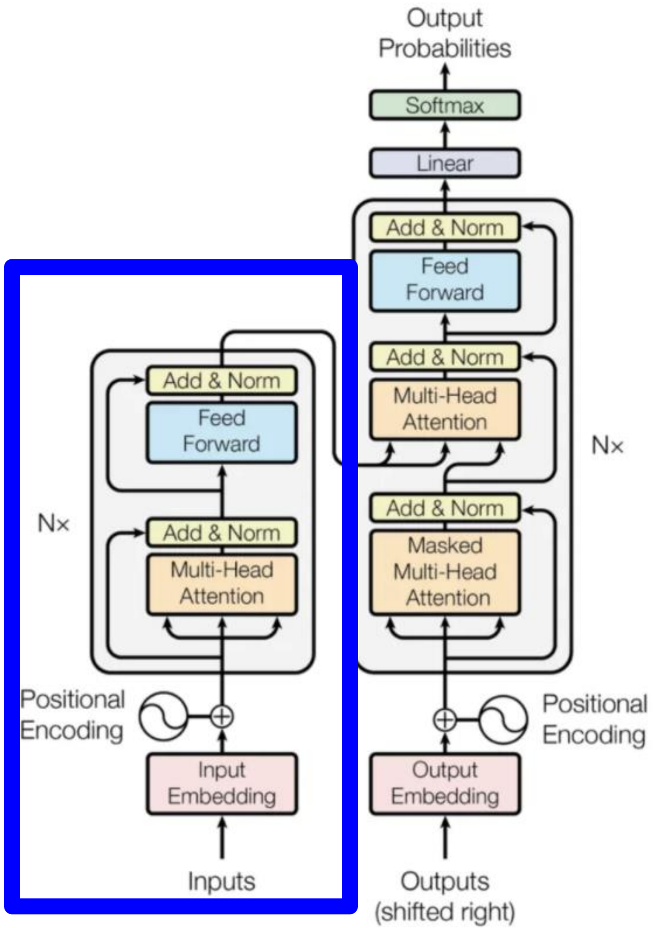
- Attention solves the bottleneck problem
- It includes a context matrix that includes information about a word’s importance in the sentence

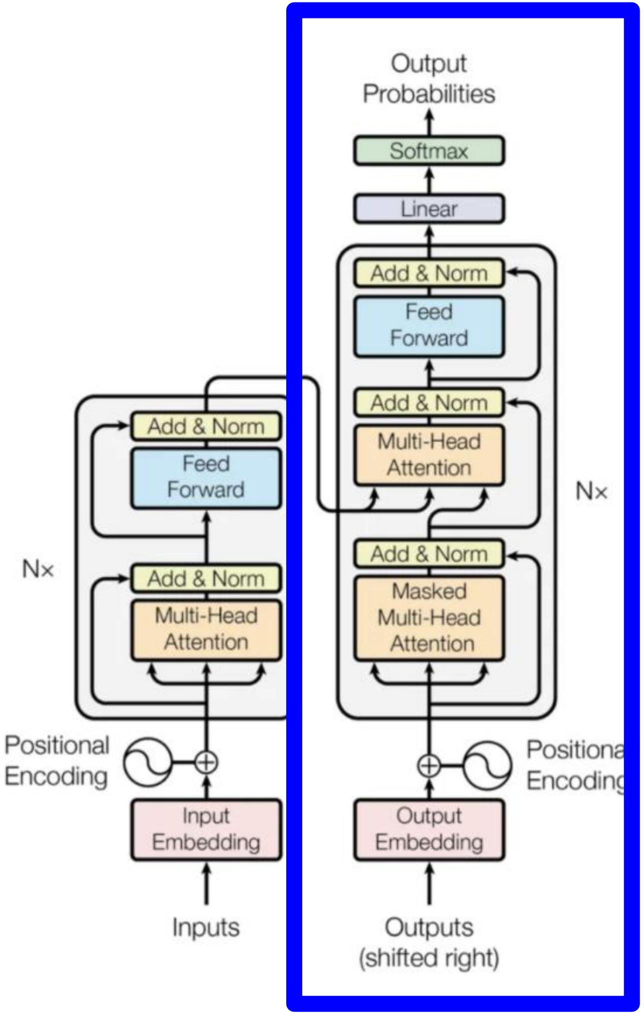


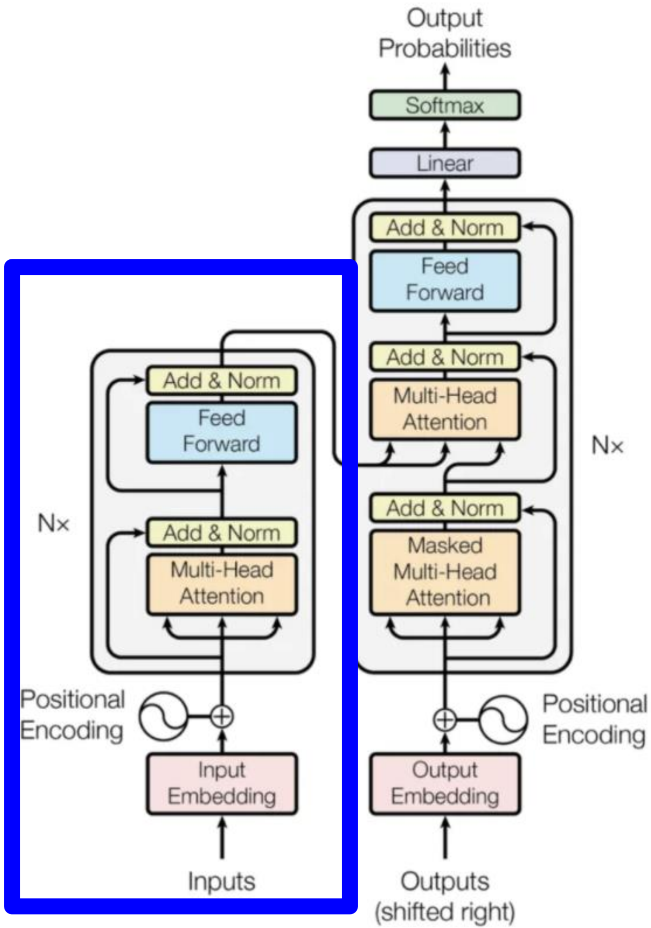
Transformers

- Previously, RNNs had to be processed sequentially
- Transformers are able to maintain some contextual information while also being able to be processed in parallel

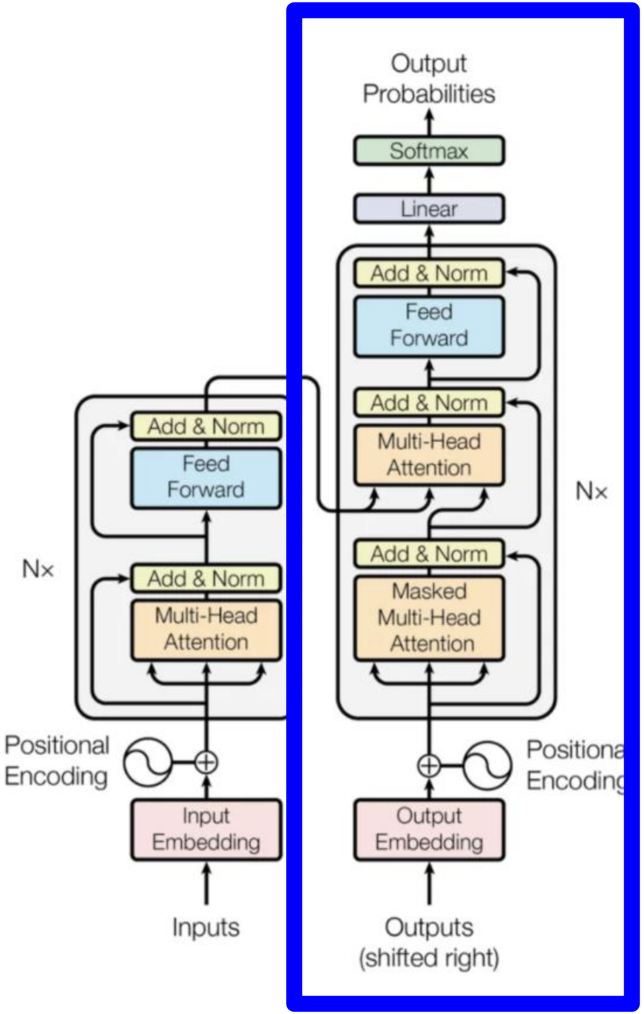

Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Positional
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)

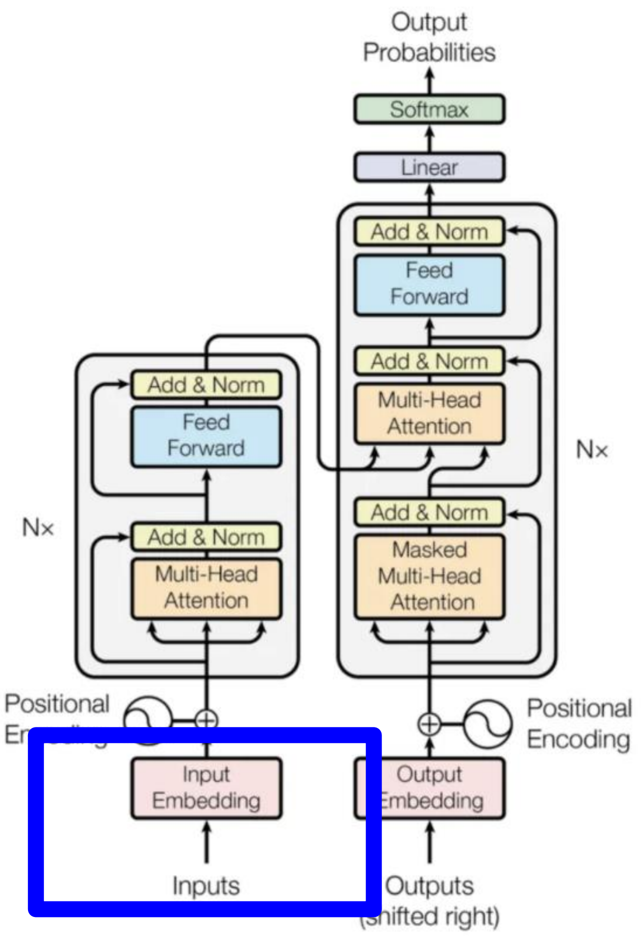

Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Input
Embedding
Inputs
Output
Embedding
Positional
Encoding
Outputs
(shifted right)

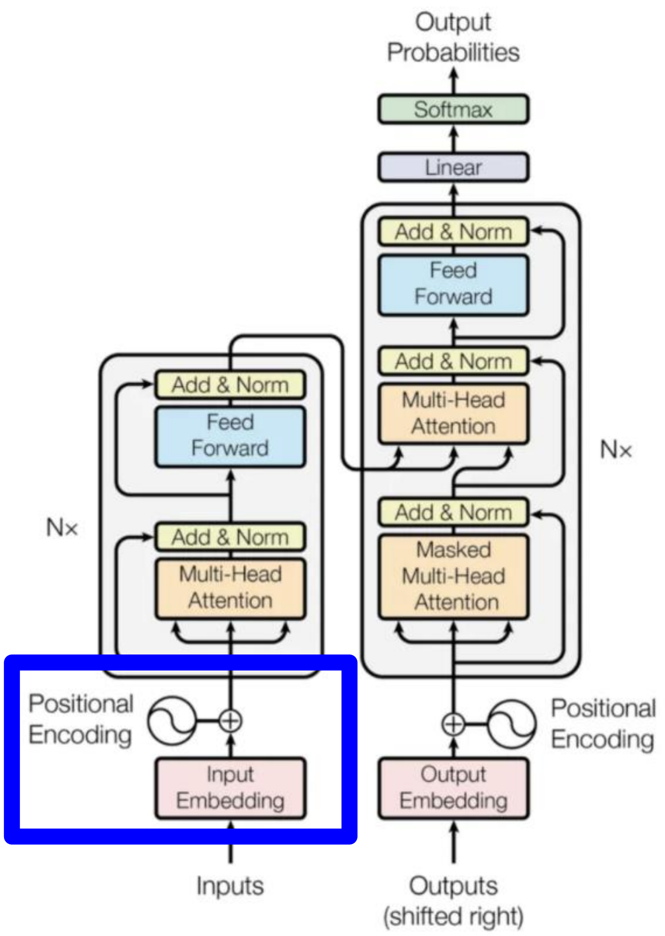

Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Input
Embedding
Inputs
Output
Embedding
Outputs
(shifted right)

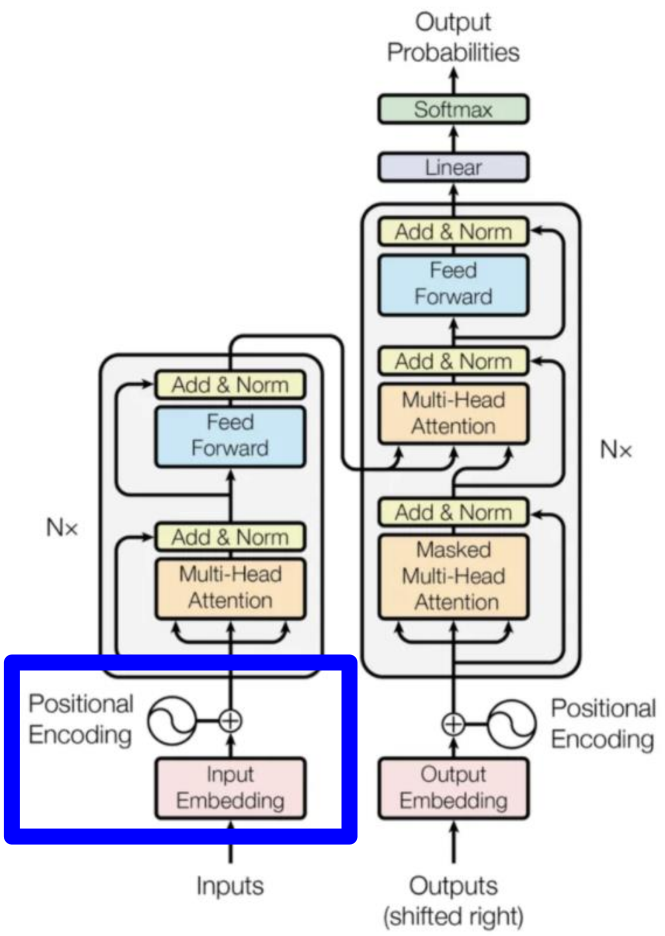


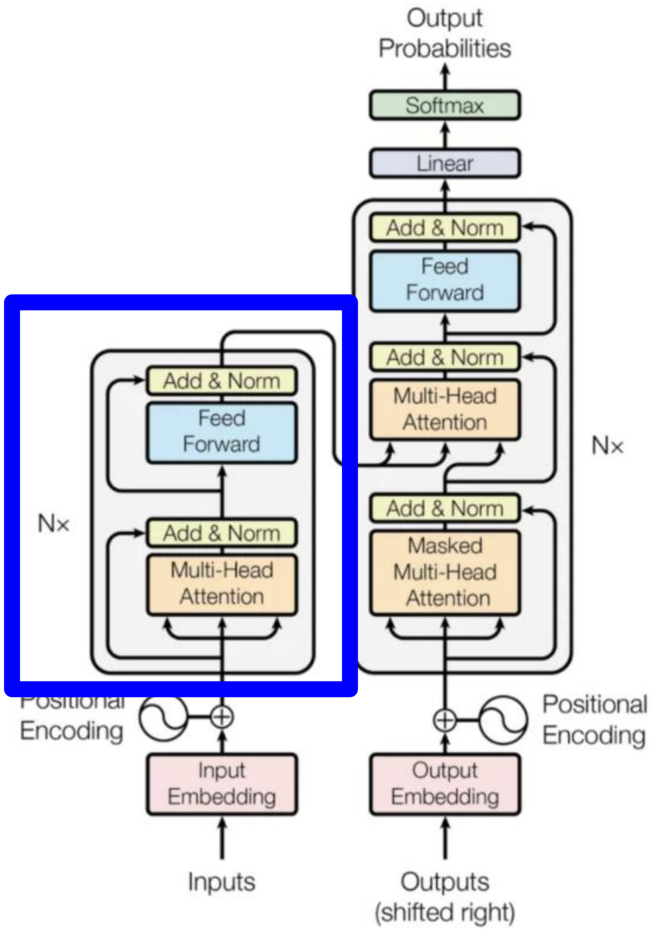
Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Input
Embedding
Inputs
Positional
Encoding
Output
Embedding
Outputs
(shifted right)

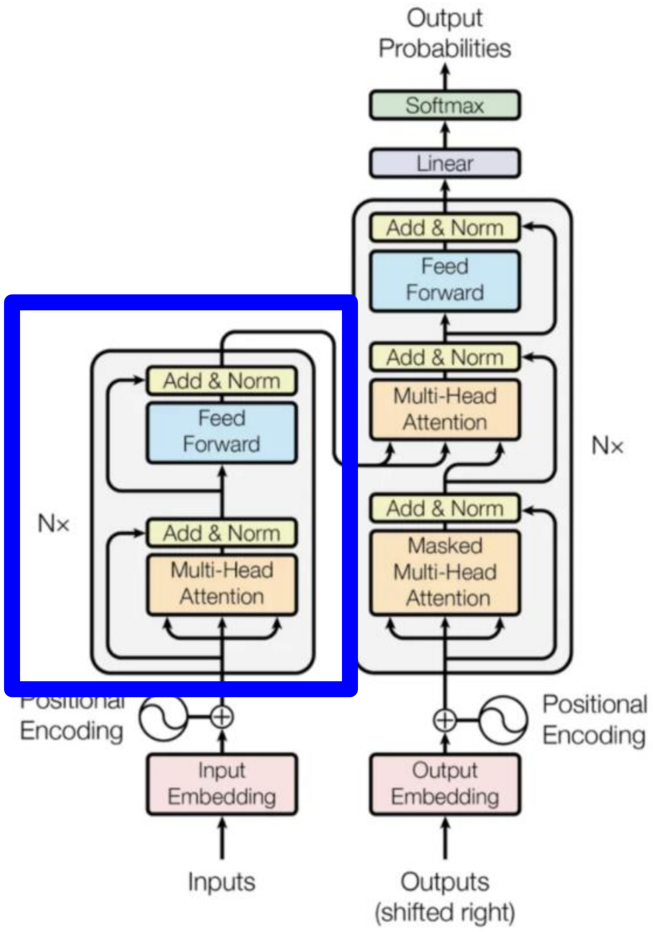

Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Positional
Encoding
Output
Embedding
Outputs
(shifted right)
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Input
Embedding
Inputs


Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

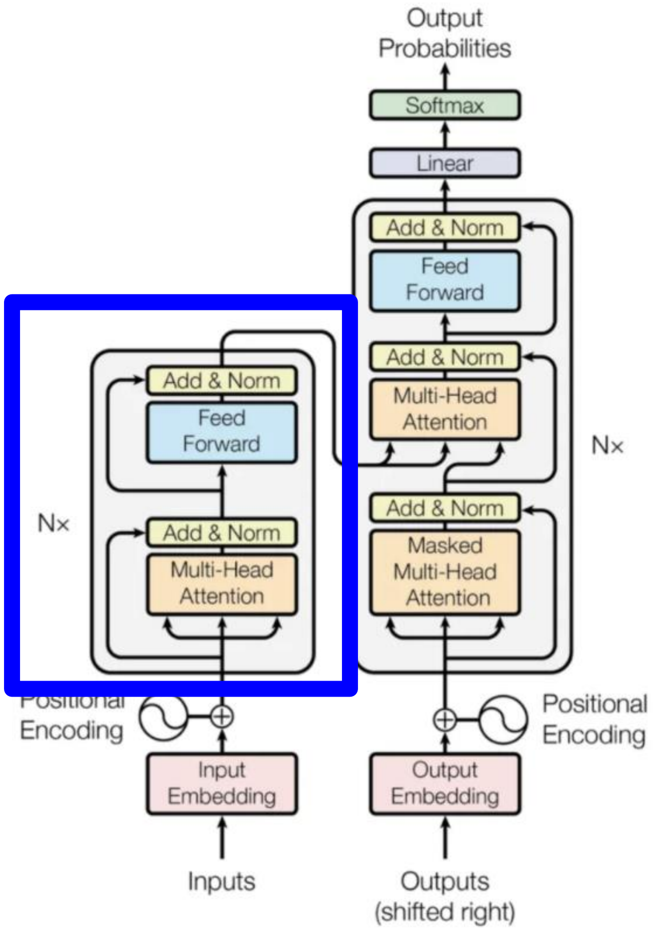

Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Add & Norm
Masked Multi-Head Attention
Add & Norm
Feed Forward
Nx
Add & Norm
Multi-Head Attention
Positional Encoding
Input Embedding
Output Embedding
Positional Encoding
Inputs
Outputs (shifted right)


Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Input
Embedding
Output
Embedding
Positional
Encoding
Inputs
Outputs
(shifted right)

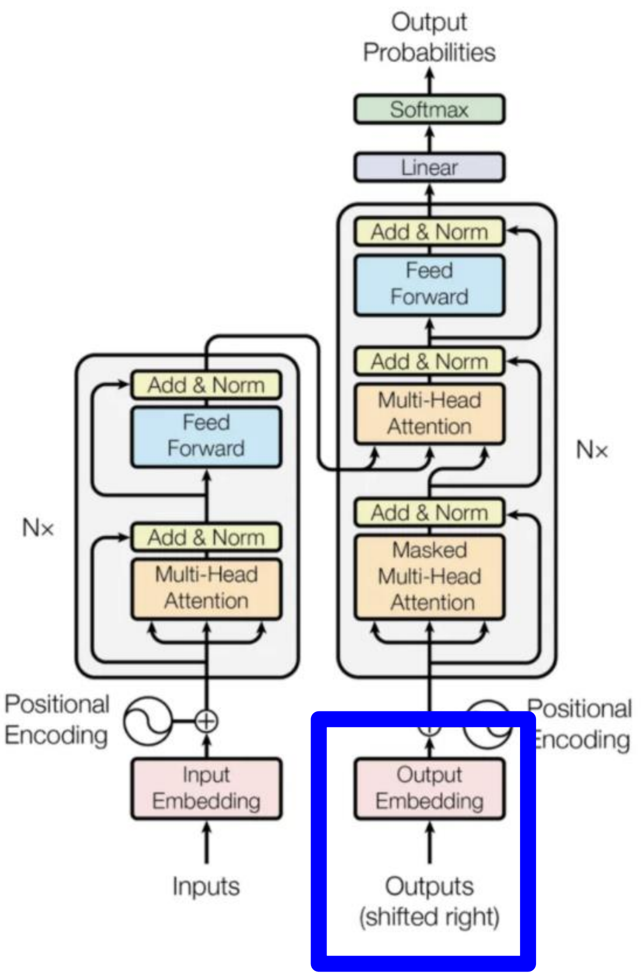

Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Add & Norm
Masked
Multi-Head
Attention
Add & Norm
Feed
Forward
Nx
Add & Norm
Multi-Head
Attention
Positional
Encoding
Positional
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)



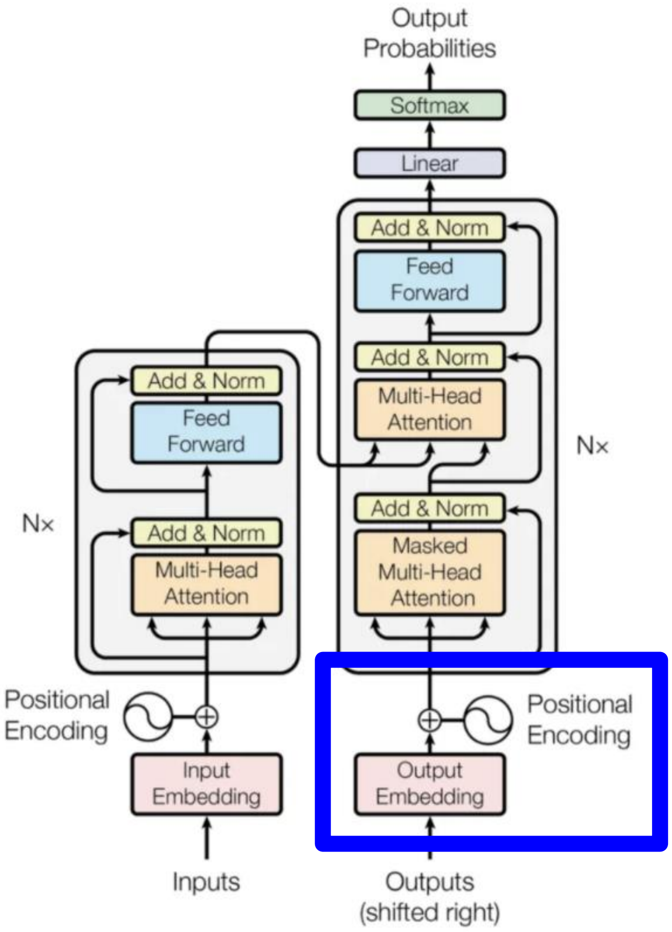
Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Input
Embedding
Inputs
Output
Embedding
Outputs
(shifted right)
Positional
Encoding

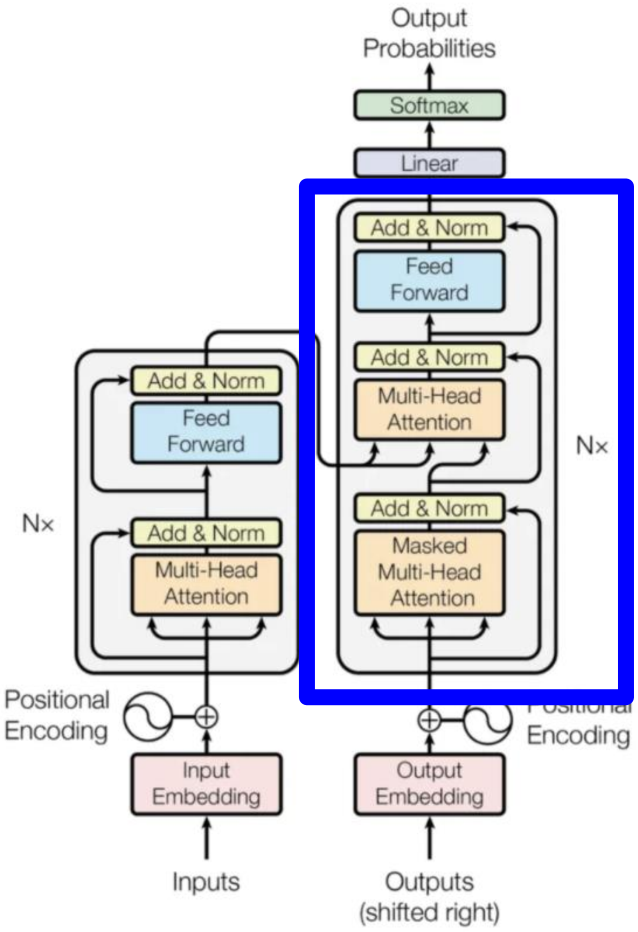


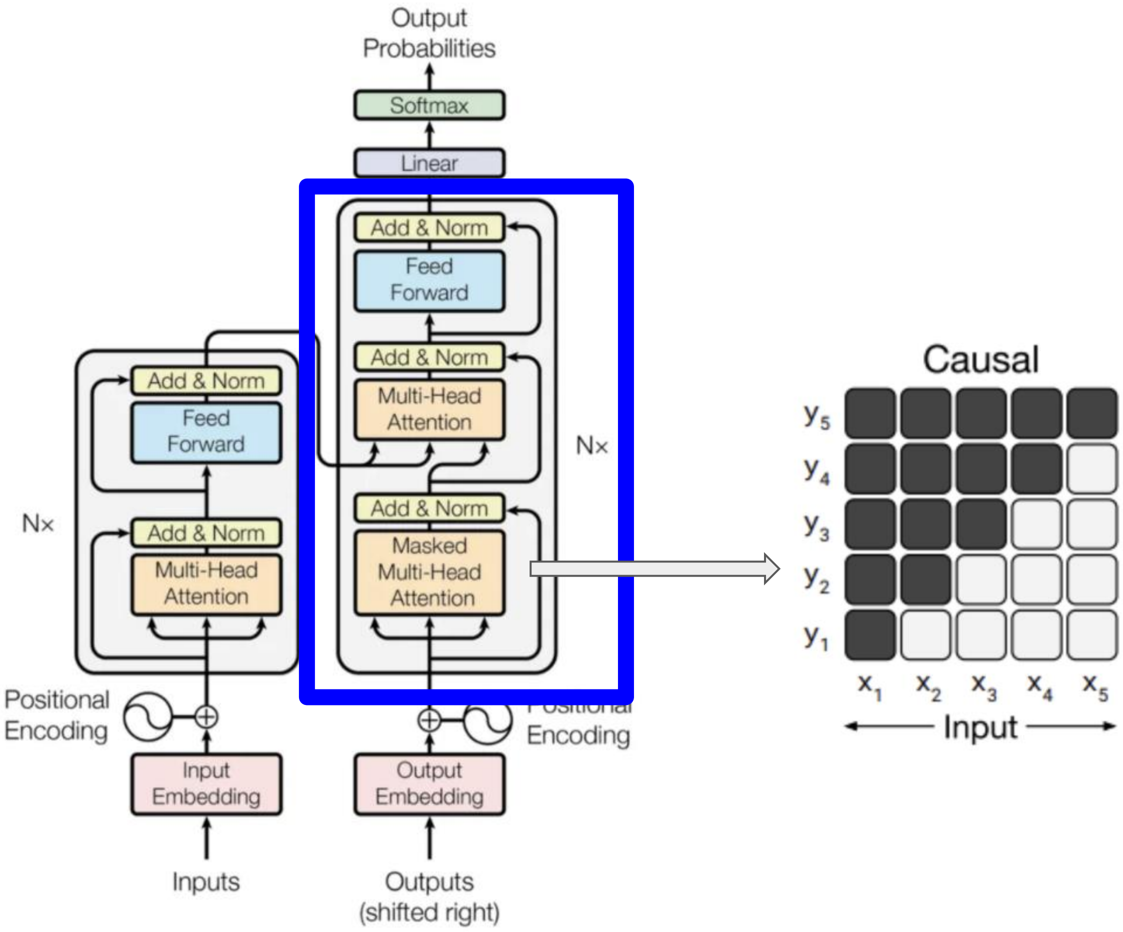
Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Encoding
Positional
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)



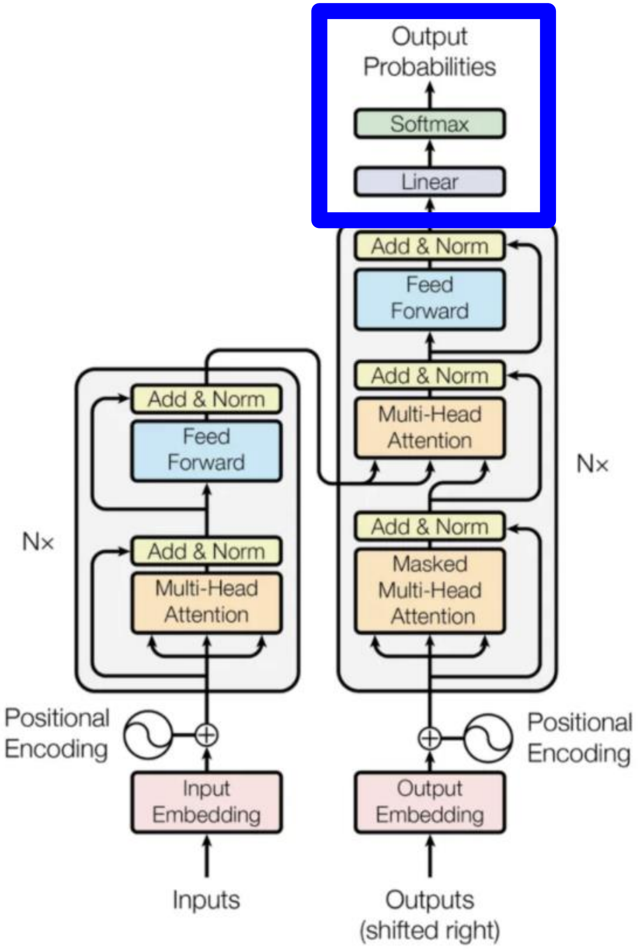
Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Positional
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)


Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Input
Embedding
Inputs
Positional
Encoding
Output
Embedding
Outputs
(shifted right)


Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

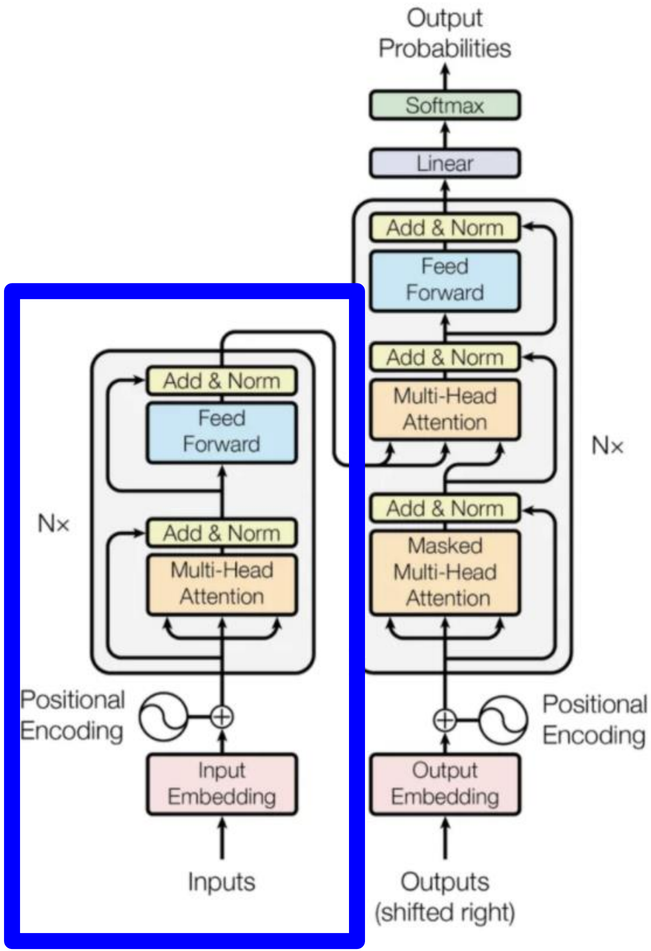


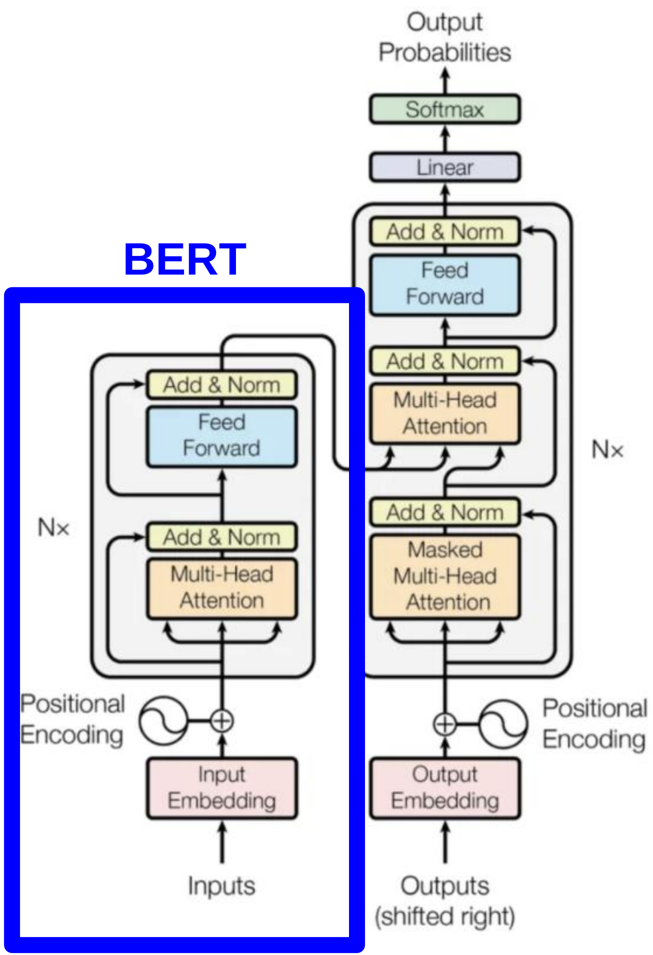
Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)
Causal
y_5
y_4
y_3
y_2
y_1
x_1 x_2 x_3 x_4 x_5
Input


Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Positional
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)

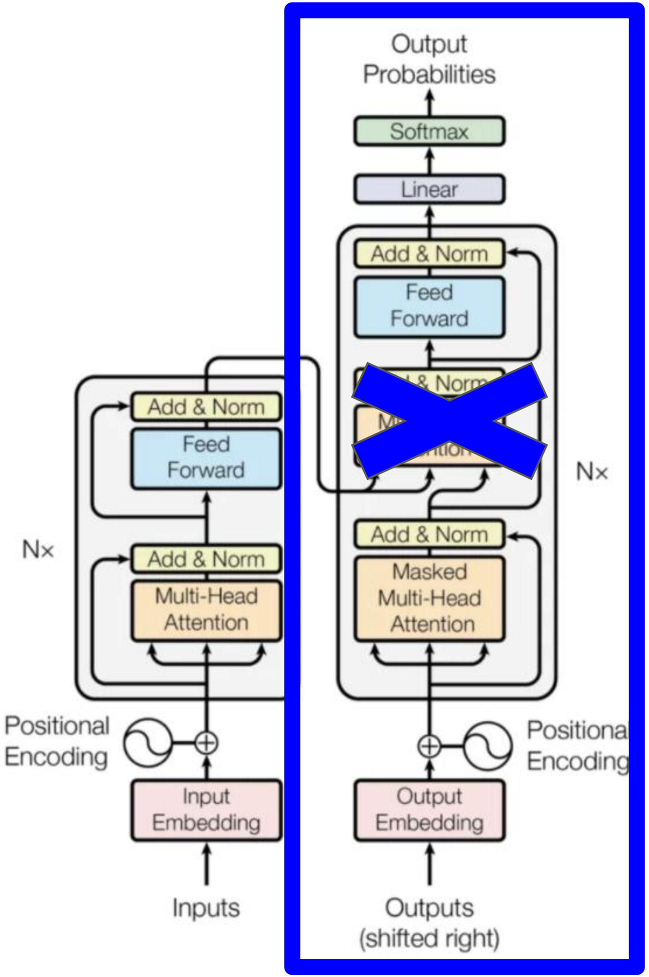
Applications of Transformers

- Machine Translation
- Text Summarization
- Language modeling
- Contextual word embeddings


Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Input
Embedding
Inputs
Positional
Encoding
Output
Embedding
Outputs
(shifted right)


BERT
Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Add & Norm
Masked Multi-Head Attention
Positional Encoding
Output Embedding
Outputs (shifted right)
Add & Norm
Feed Forward
Nx
Add & Norm
Multi-Head Attention
Positional Encoding
Input Embedding
Inputs

GPT-3



Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Feed
Forward
Nx
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Positional
Encoding
Positiona
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)

Contextual Embeddings

1. Words are not numbers

2. Input can be different lengths

3. Words can mean different things in different contexts

Issues with Word Vectors

“Did I show you this **clip** of a dog skateboarding?”

“I need to get a chip **clip**”

“He runs at a good **clip**”

“I have to **clip** my dog’s nails”

Issues with Word Vectors

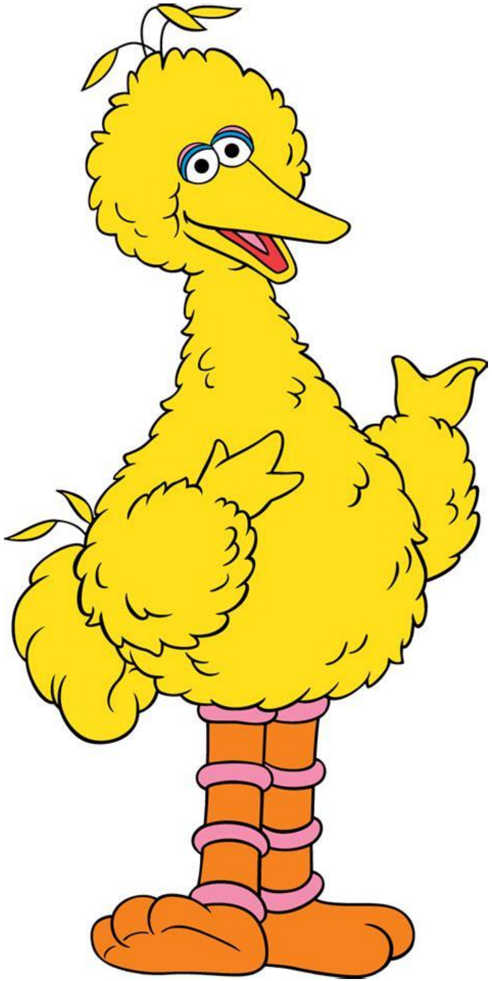
"Did I show you this **clip** of a dog skateboarding?"

"I need to get a chip **clip**"

"He runs at a good **clip**"

"I have to **clip** my dog's nails"

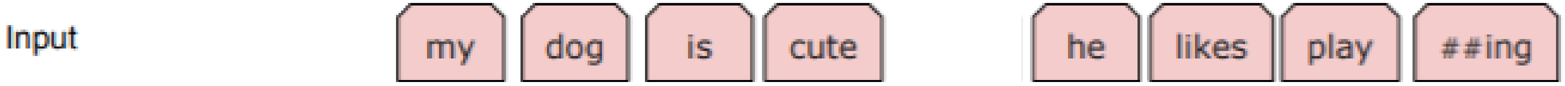
Contextual Word Embeddings



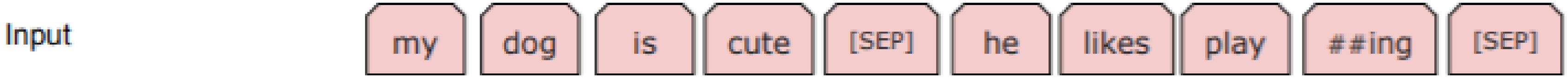
BERT

- Trained on enormous datasets by Google
- Works by using semi-supervised bidirectional language modeling on 15% masked tokens
- Creates a [CLS] token with an aggregate value for all of the tokens in the sentence - a great starting point!
- Allows us to use fine tuning/transfer learning to achieve great performance on downstream tasks

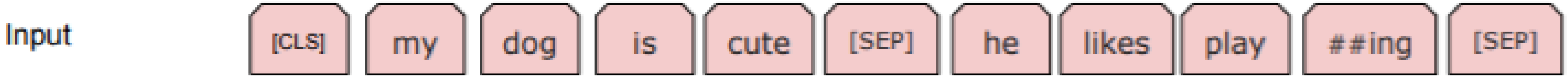
BERT



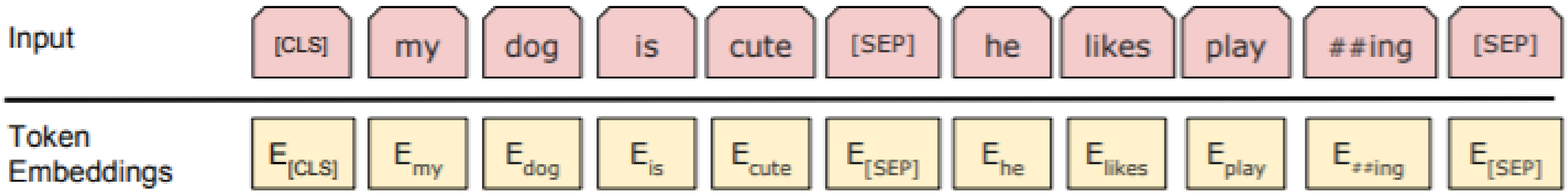
BERT



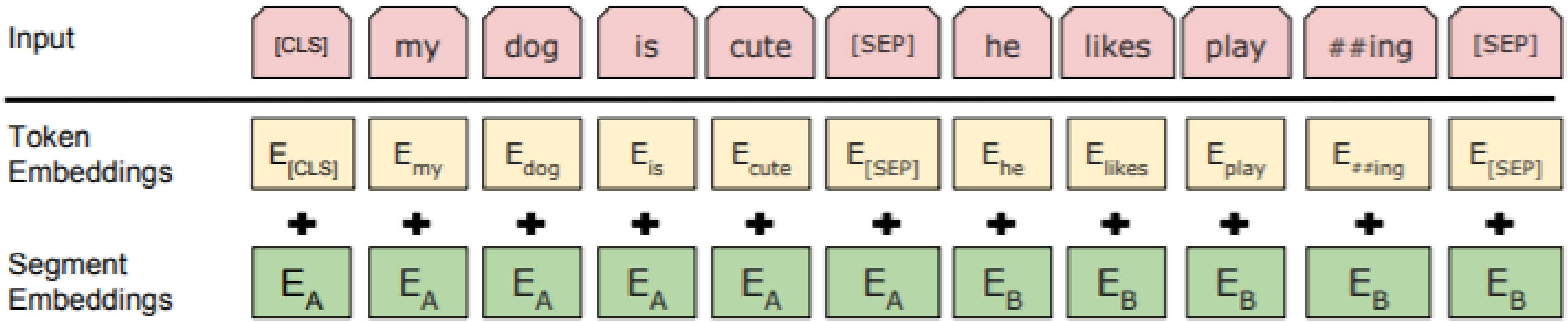
BERT



BERT


Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$
E_{my}
E_{dog}
E_{is}
E_{cute}
$E_{[SEP]}$
E_{he}
E_{likes}
E_{play}
$E_{\#\#ing}$
$E_{[SEP]}$

BERT

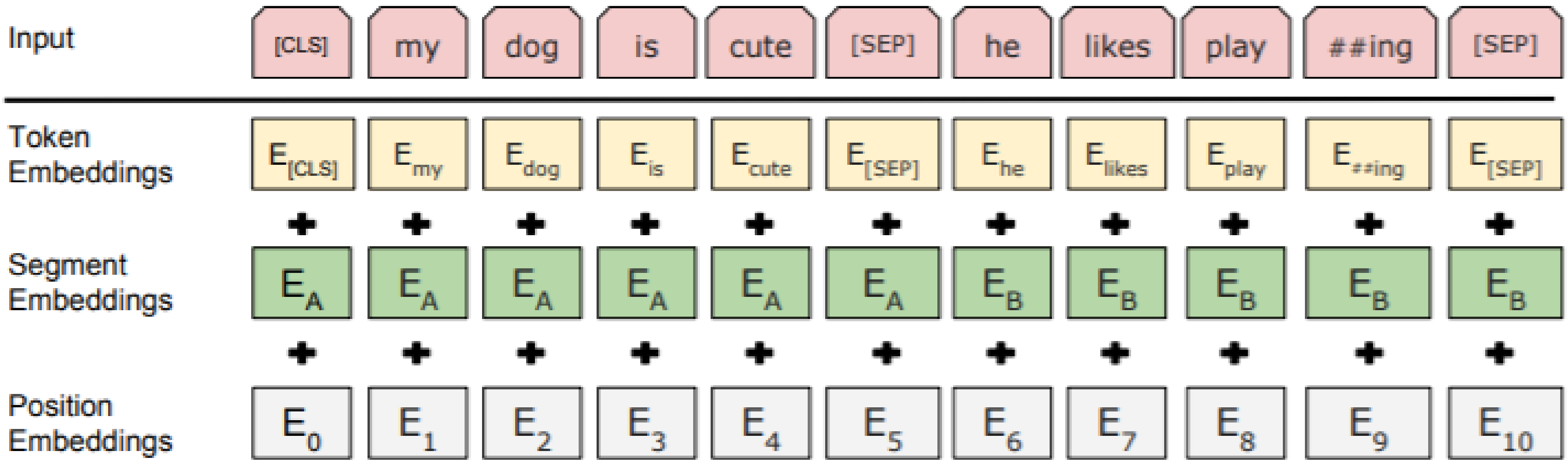


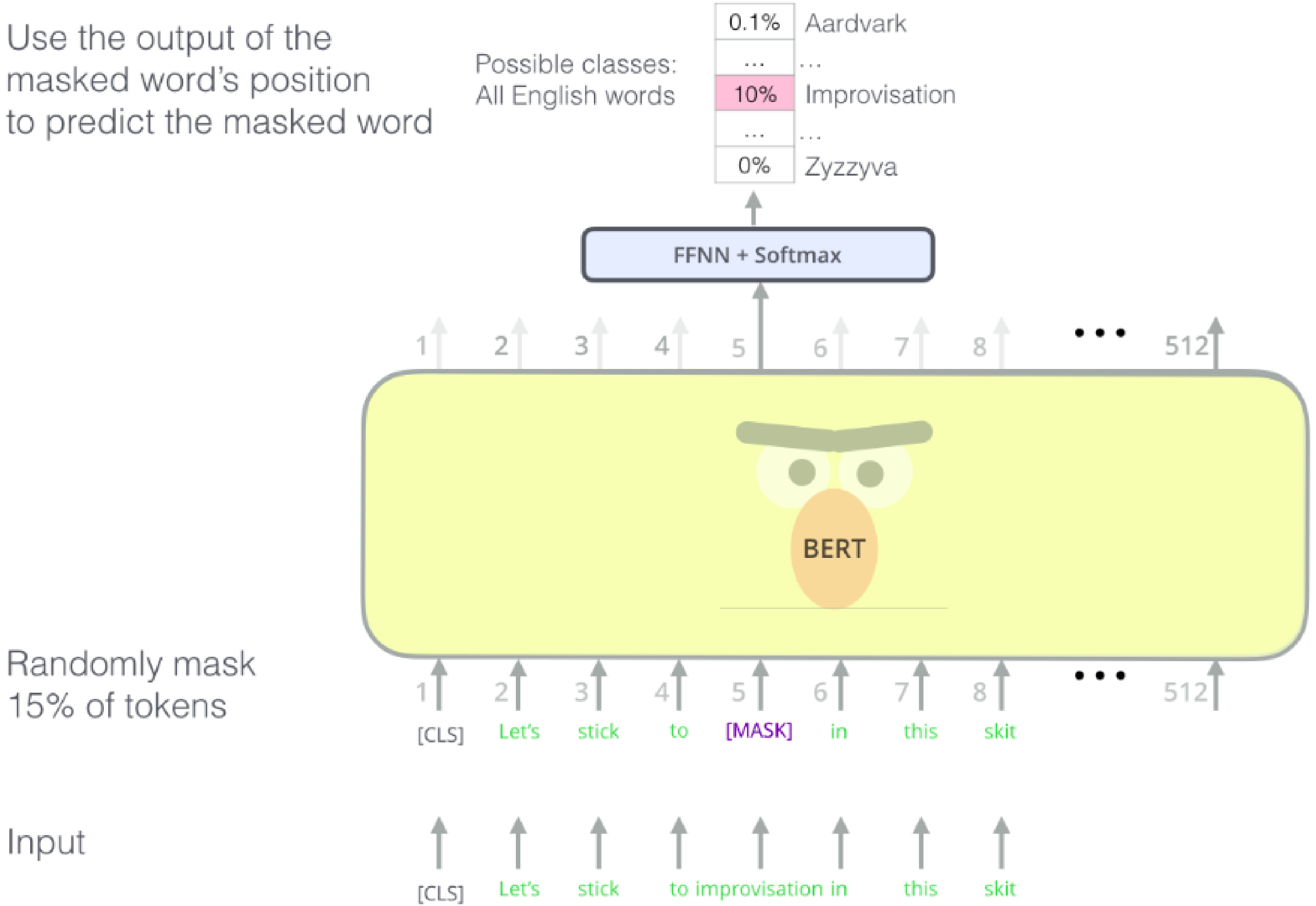
Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
E[CLS]
Emy
Edog
Eis
Ecute
E[SEP]
Ehe
Elikes
Eplay
E##ing
E[SEP]
Segment Embeddings
EA
EA
EA
EA
EA
EA
EB
EB
EB
EB
EB

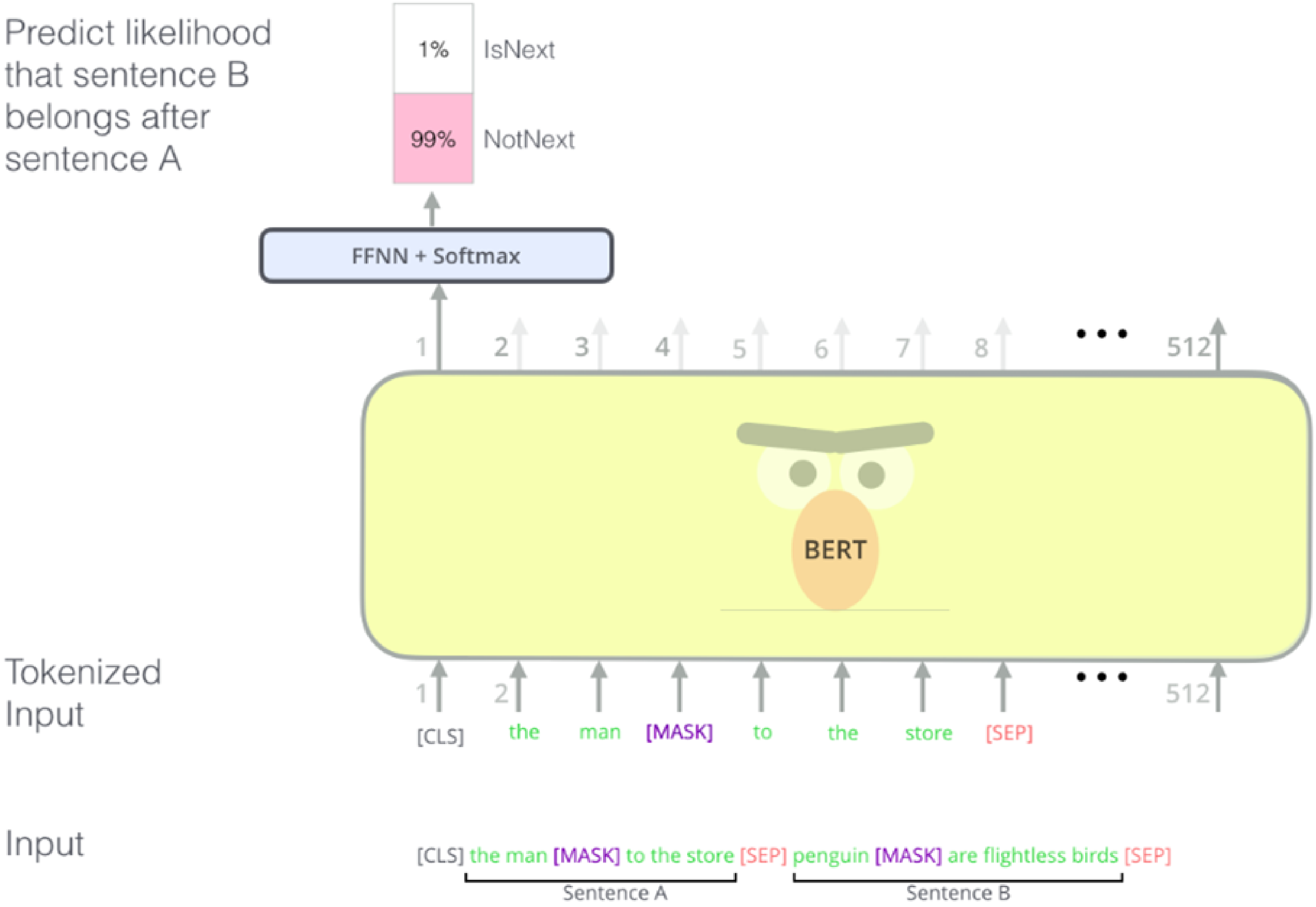
BERT - Segmentation

[CLS]	I	LIKE	CATS	[SEP]	I	LIKE	DOGS
0	0	0	0	0	1	1	1

BERT


Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

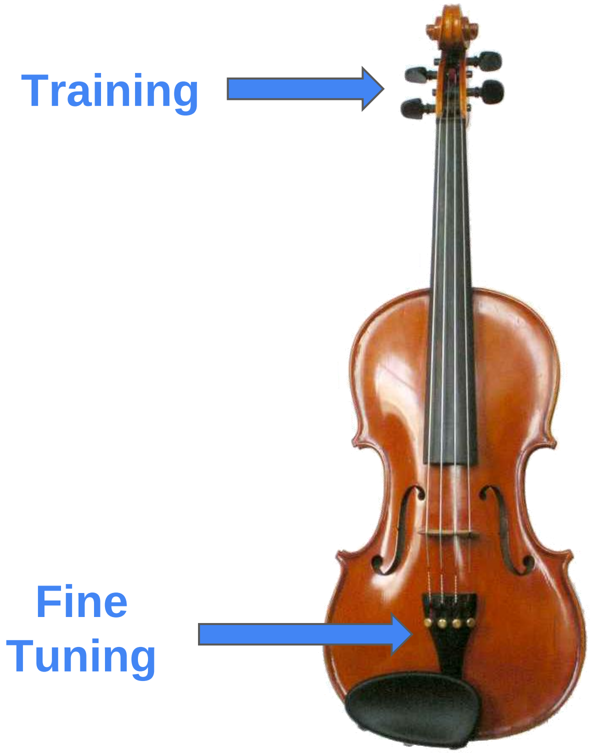

Use the output of the
masked word's position
to predict the masked word
Possible classes:
All English words
0.1% Aardvark
...
10% Improvisation
...
0% Zyzzyva
FFNN + Softmax
1 2 3 4 5 6 7 8 ••• 512
BERT
Randomly mask
15% of tokens
[CLS] Let's stick to [MASK] in this skit
Input
[CLS] Let's stick to improvisation in this skit



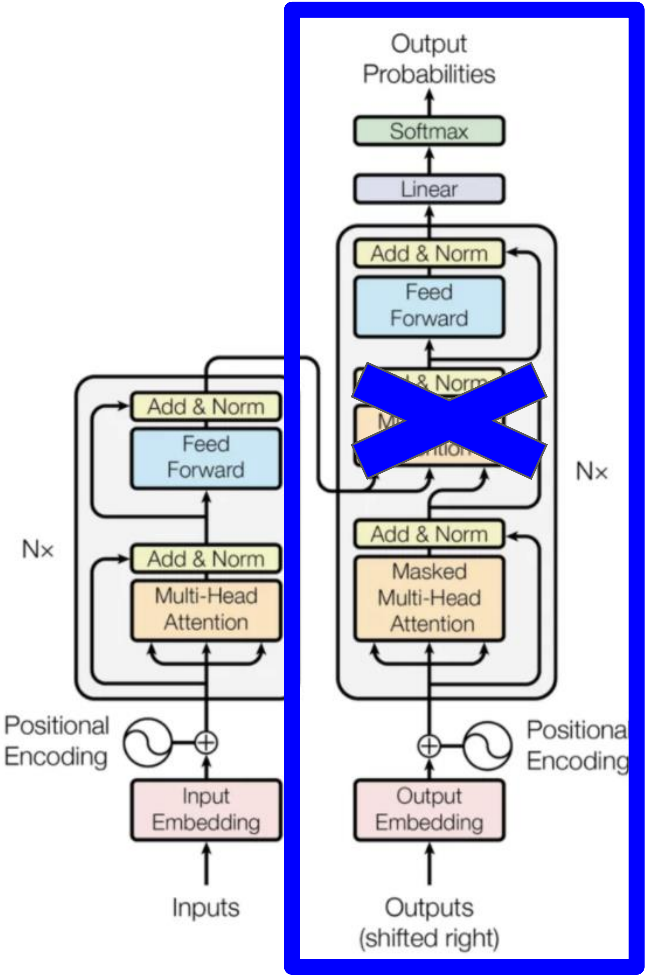
Predict likelihood that sentence B belongs after sentence A
1%
IsNext
99%
NotNext
FFNN + Softmax
1
2
3
4
5
6
7
8
512
BERT
Tokenized Input
1
2
512
[CLS]
the
man
[MASK]
to
the
store
[SEP]
Input
[CLS] the man [MASK] to the store [SEP] penguin [MASK] are flightless birds [SEP]
Sentence A
Sentence B

Fine Tuning vs. Training

- Fine tuning is simply a type of training!
- Fine tuning:
  - Works as a kind of transfer learning
  - Smaller learning rate (but we usually use an optimizer like ADAM)



Large Language Models


Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Add & Norm
Masked
Multi-Head
Attention
Nx
Positional
Encoding
Positiona
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)

GPT-3

The age of BIG COMPUTE

- Until recently, we used GPUs (Graphics Processing Units) for deep learning
    - GPUs are originally for gaming, optimized for matrix multiplication and use multiple cores, which is necessary for parallel computing
    - Transformers can capture contextual information while doing parallel computing - GPUs make it possible to harness this potential
- We live in the age of the Tensor Processing Unit (TPU)
    - TPUs are only available to Google and they’re hardware designed specifically for machine learning
- We’ve gone from data science requiring a good deal of feature engineering to really truly brute force

Language Modeling Revisited

- The task of predicting the probability of a sentence

“I’ll text you when I get ______”

Large Language Models

- “Large” because of the number of parameters
  - GPT-2 - 1.5 billion parameters
  - GPT-3 - 175 billion parameters
  - GPT-4 - “over one trillion parameters”
  - ChatGPT - ?

Large Language Models

- “Large” because of the number of parameters
  - GPT-2 - 1.5 billion parameters
  - GPT-3 - 175 billion parameters
  - GPT-4 - “over one trillion parameters”
  - ChatGPT - 175 billion parameters!

Large Language Models

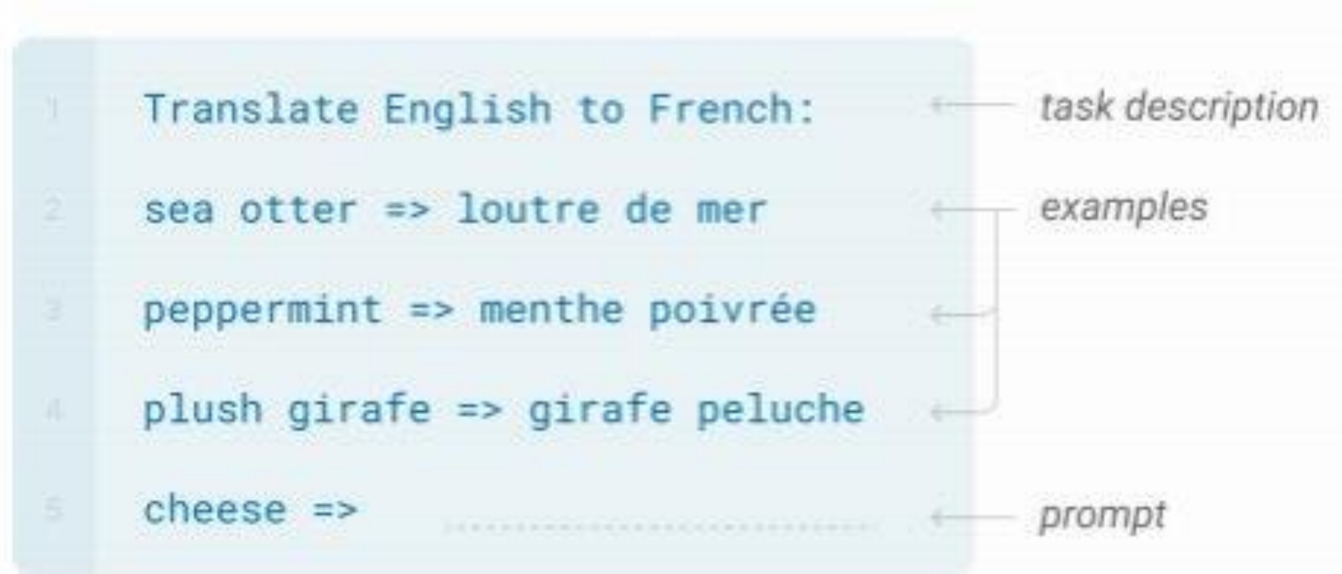
- "Large" because of the number of parameters
  - GPT-2 - 1.5 billion parameters
  - GPT-3 - 175 billion parameters
  - ChatGPT - 175 billion parameters!
- ChatGPT is based on GPT-3
  - Fine-tuned using Reinforcement Learning from Human Feedback (RLHF) - a combination of unsupervised methods double-checked by human labelers

Prompt Engineering & Prompt Tuning

Prompt Engineering

“Instead of training the model, the model trains you”

Prompt Engineering

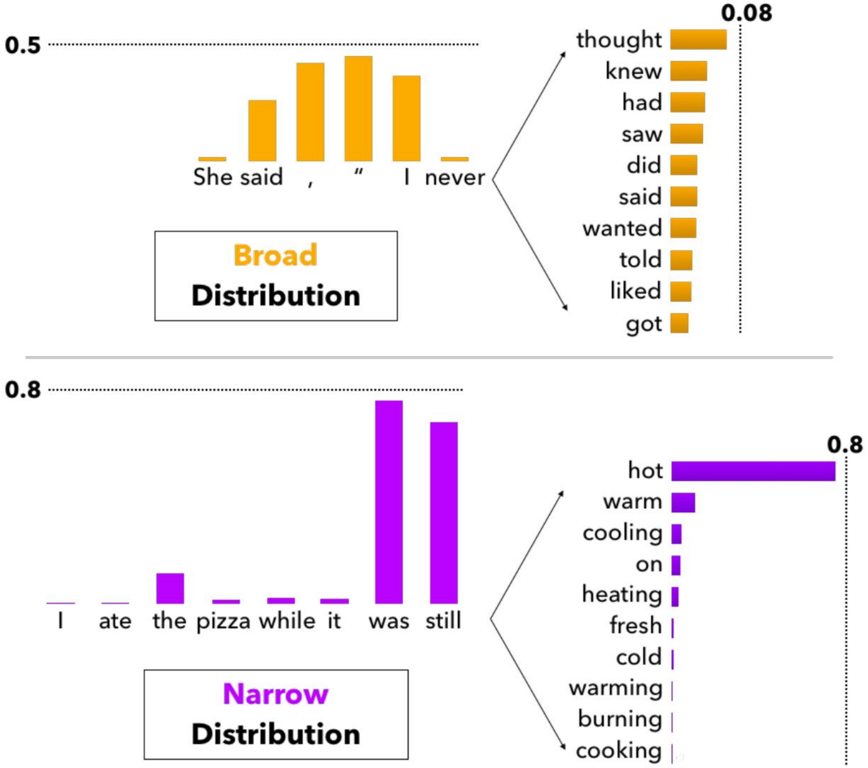


Prompt Engineering - Parameters

- Max Response: Sets a limit on the number of tokens per model response.
- Frequency Penalty: Reduce the chance of repeating a token proportionally based on how often it has appeared in the text so far.
- Presence Penalty: Reduce the chance of repeating any token that has appeared in the text at all so far.

Prompt Engineering - Parameters

- **Temperature:** Controls how “creative” your LLM will be by increasing entropy and creating a broader output distribution - higher values will be more creative because probabilities will be more more similar to each other
- **Top P:** Similar to temperature, this controls creativity by “allowing” tokens with a probability above a certain threshold (P) to be generated



Prompt Engineering - Practical Tips

- Restricting the length of the output can prevent lying
- Be specific - include tone, style, mention specific things you want to be left out, etc.
- You can use yes/no questions with inputs for judgements (more in a second) or information extraction
- If you don’t want to do a whole Generative AI model (cost is a concern), you can use Generative AI to create datasets for lighter-weight models (test this thoroughly first, based on the use case)

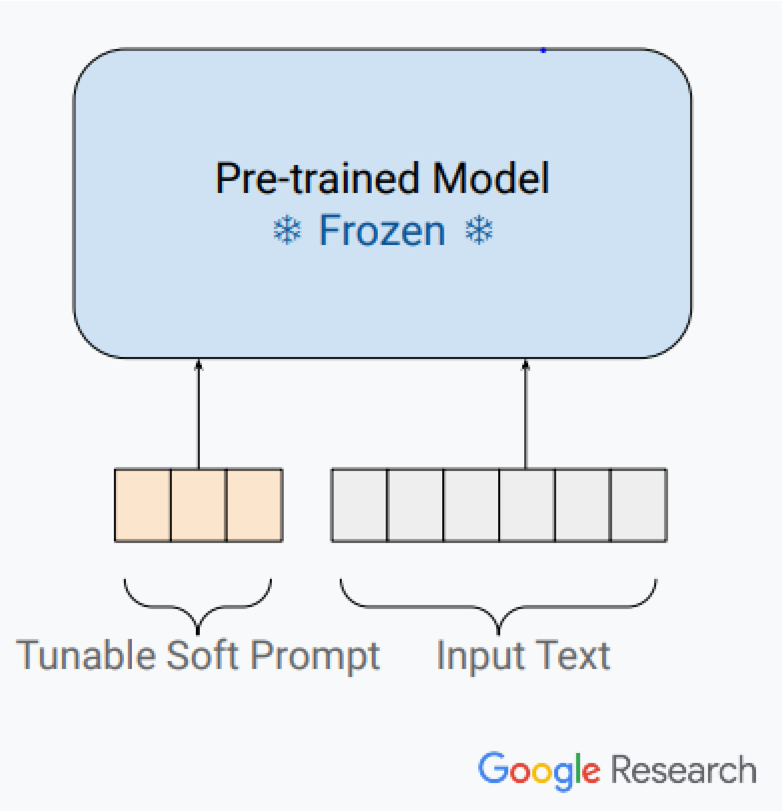
Prompt Tuning

- Remember: one of the benefits of Transformers was that you could fine-tune
  - However, the number of parameters is huge, and still requires parallel computation
  - You now have multiple copies of the same model, tweaked - which requires a lot of disk space

- Solution: having a smaller “prefix” (prompt) can be trained to your specific task for much less compute and with good performance

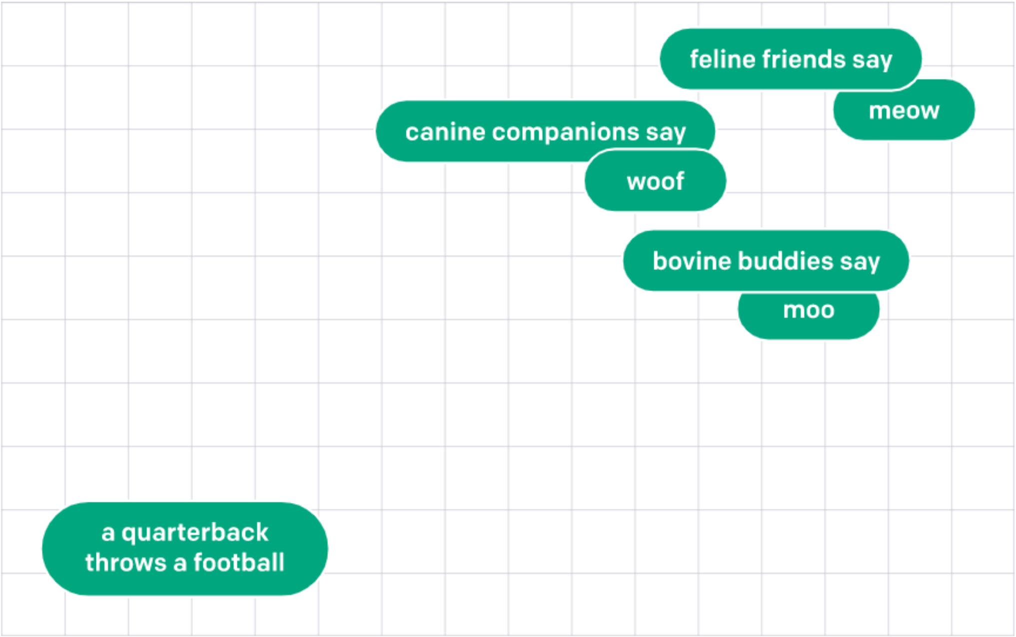
Prompt Tuning

- Prepend **virtual tokens** (e.g. fake words) to the input
- Learn embeddings for only these special tokens
- Benefits:
  - Learn a much smaller number of parameters
  - No manual time needed
  - We can learn from the whole dataset, rather than specific context



Prompt Tuning: Nearest Neighbors

Turn prompt-tuned virtual tokens into **real words** with a simple **nearest neighbor** search using **cosine similarity**!



Retrieval-Augmented Generation (RAG)?

- Allows us to combine the knowledge of LLMs, which are trained on huge amounts of data but are not necessarily up-to-date or aware (out of the box) of the specifics of your organization’s data, with a specific knowledge base without retraining or even fine-tuning the model
- Works by “chunking” and creating embedding vectors for your prompt and your documents (text-based documentation, database, etc.) and doing similarity comparisons, then feeding the closest “answer” into the LLM to return a nice natural-language answer to a question
- Very useful for LLM-based chatbots

Evaluating LLMs Linguistically

- Perplexity

Evaluating LLMs for Factual Accuracy

- Still an unsolved problem!
- Cosine similarity (seriously)
- LLM-as-a-judge
  - Ask the LLM to determine whether the output is acceptable based on certain criteria
  - Do question-answering - yes/no questions about the content of the output
- Human evaluation
- Watch this space!


Thank You!