

# Pretraining and Contextualized Word Embeddings

# Contextualized Word Embeddings

---

- Pre-neural NLP: manually crafted features, most supervised learning
- Labeled data is scarce, unlabeled data is abundant
  - And there are much we can learn from unlabeled data
  - Yet, supervised learning requires labeled data...
- Word embeddings partially tackle this:
  - Using unlabeled data we can get embeddings, which in turn can serve as features for supervised learning
- But bag-of-words embeddings give up on a lot of useful information:
  - Word order
  - Different senses/uses of a word are conflated

# Contextualized Word Embeddings

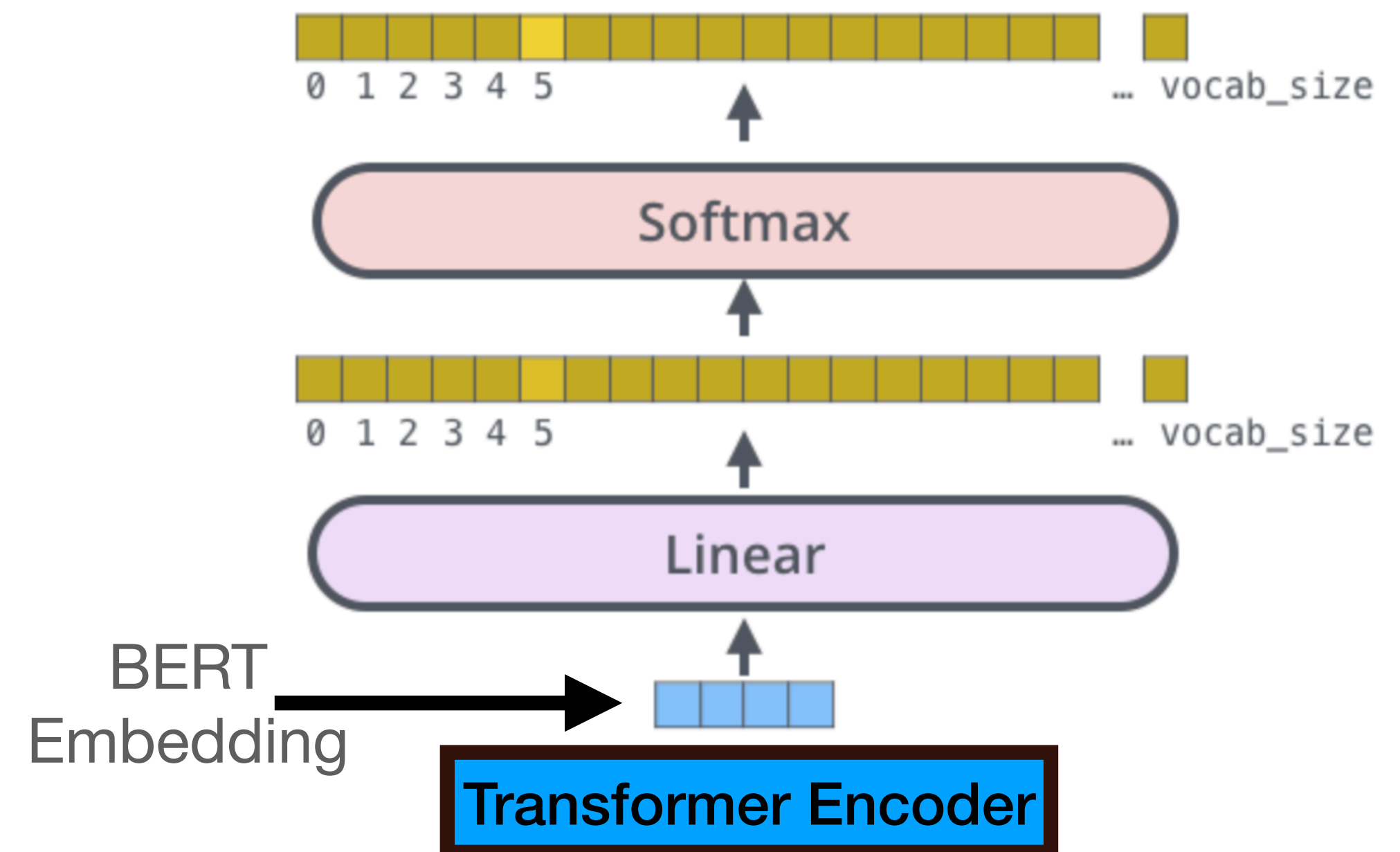
---

- Neural language models implicitly represent much information about words
  - To predict the next word, we need to represent not only the neighbors of a word, but also their senses, their order etc.
  - They therefore implicitly induce embeddings from unlabeled data that contain richer information than, say, *word2vec*
- The same can potentially hold for sentences:
  - A sentence can be represented as the representation that can help predict the preceding and following sentence

# The BERT Model

---

- The BERT model is a standard model for obtaining contextualized word embeddings
- The model is based on the *Transformer* encoder
  - We will explain what that means in a bit
- The BERT embedding of a token is the activation of its penultimate layer








# Masked Language Models

---

- BERT is trained as a **masked language model** (MLM)
- MLMs are statistical models that, given a sentence where part of the tokens are replaced with masks, predict the identity of the masked tokens

*My dog is [MASK] and likes to [MASK] the entire [MASK].*

## Mask 1

Prediction	Score
My dog is <b>hungry</b> and likes to [MASK2] the entire [MASK3] .	 5.6%
My dog is <b>friendly</b> and likes to [MASK2] the entire [MASK3] .	 4.8%
My dog is <b>cute</b> and likes to [MASK2] the entire [MASK3] .	 3.7%
My dog is <b>nice</b> and likes to [MASK2] the entire [MASK3] .	 2.9%
My dog is <b>smart</b> and likes to [MASK2] the entire [MASK3] .	 2.4%

# BERT's Training

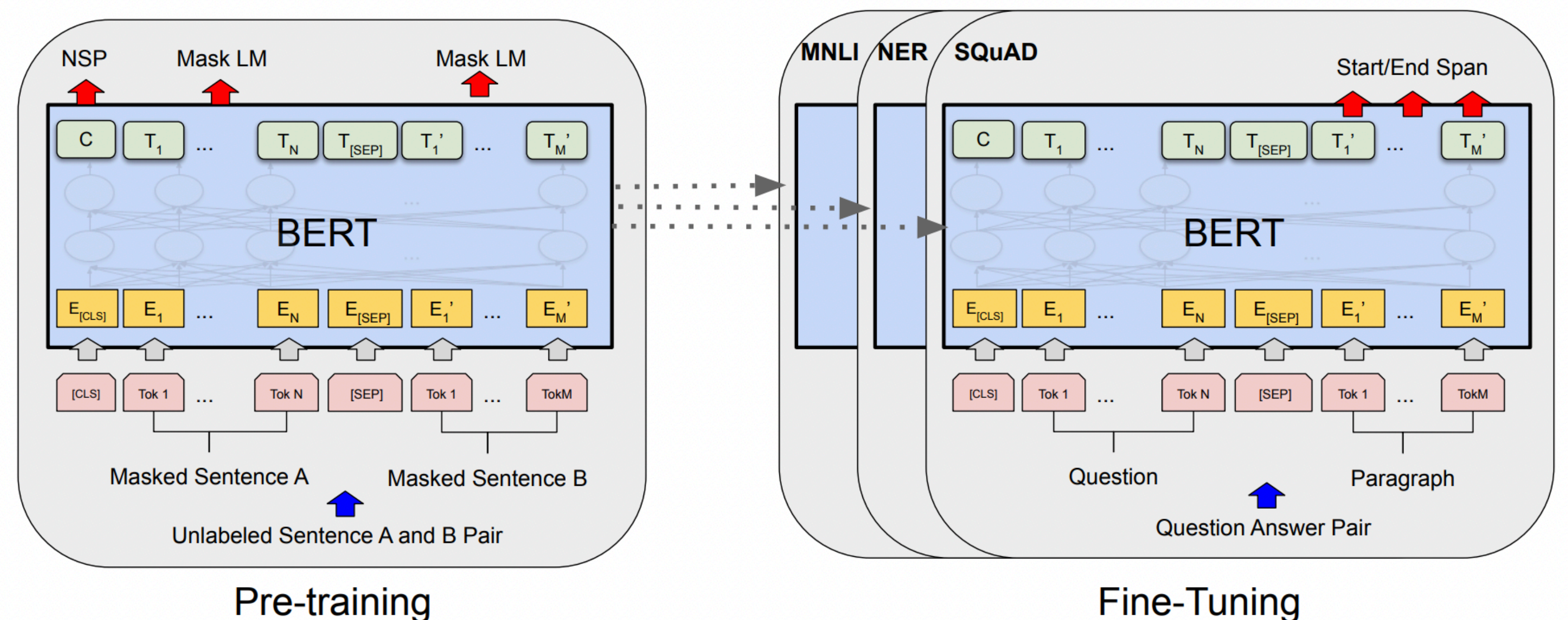
---

- BERT's training is carried out by segmenting texts into windows in the size of the model's input window, and selecting 15% of the tokens (denote the set with  $S$ ). Of the tokens in  $S$ :
  - 80% of the time, replace the token with <MASK>
  - 10% of the time, replace the token with a random token
  - 10% of the time, keep the token the same
- Denote the modified sentence with  $\mathbf{x}'$
- The loss function is then the average log probability of the original word in the selected positions  $S$ :

$$L(\theta) = \frac{1}{|S|} \sum_{i \in S} \log(p_{\theta}(x_i | \mathbf{x}'))$$

# Fine Tuning BERT

- The standard paradigm today in NLP includes a phase where BERT is trained as a masked language model (and using other methods, that will not be discussed here)
- The resulting model is then used to initialize the parameters of the model in learning further tasks
  - Known as fine-tuning





# Contextualized Word Embeddings

---

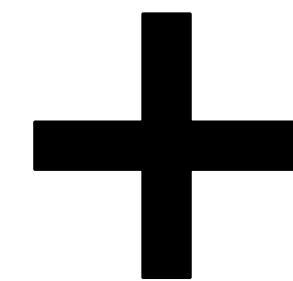
- Contextualized word embeddings are an extremely powerful method used through present-day NLP
- Contextualized word embeddings are different from the previous non-contextualized ones in:
  - Being sensitive to the context in which the word appears
  - Being a result of a complex computation, rather than a look-up table
  - Can be adjusted or fine-tuned to a specific end-goal
- Standard contextualized word embeddings today are trained over billions of words, and are derived from networks with hundreds of millions of parameters



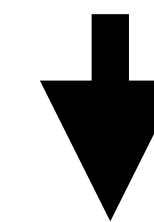
# Zero-shot Prediction: Alternative to using MLM for embeddings

- (Masked) language models can also be used to directly decode the answer as a token (without pre-training). This is called zero-shot prediction.
- Take this relation extraction example

On November 22, 2001, Pitt made a guest appearance in the [eighth season](#) of the television series [Friends](#), playing a man with a grudge against [Rachel Green](#), played by [Jennifer Aniston](#), to whom Pitt was married at the time.<sup>[86]</sup> For this performance he was nominated for an [Emmy Award](#) in the category of [Outstanding Guest Actor in a Comedy Series](#).<sup>[87]</sup> In December 2001, Pitt played [Rusty Ryan](#) in the heist film [Ocean's Eleven](#), a remake of the 1960 [Rat Pack original](#). He joined an ensemble cast including [George Clooney](#), [Matt Damon](#), [Andy García](#), and Julia Roberts.<sup>[88]</sup> Well received by critics, *Ocean's Eleven* was highly successful at the box office, earning \$450 million worldwide.<sup>[32]</sup> Pitt appeared in two episodes of MTV's reality series [Jackass](#) in February 2002, first running through the streets of Los Angeles with several cast members in gorilla suits,<sup>[89]</sup> and in a subsequent episode participating in his own staged abduction.<sup>[90]</sup> In the same year, Pitt had a cameo role in George Clooney's directorial debut [Confessions of a Dangerous Mind](#).<sup>[91]</sup> He took on his first voice-acting roles in 2003, speaking as the titular character of the [DreamWorks](#) animated film [Sinbad: Legend of the Seven Seas](#)<sup>[92]</sup> and playing [Boomhauer's](#) brother, [Patch](#), in an episode of the animated television series [King of the Hill](#).<sup>[93]</sup>



Pitt appeared in the movie [MASK]



0.3	Ocean's Eleven
0.15	Confessions of a Dangerous Mind
0.01	Friends
0.01	Jackass
...	

decoder output

# The Transformer Architecture

# Attention and Transformers

---

- We learned about attention as a way to model an implicit soft alignment between the words in the source sentence and the words in the target
  - Helps “select” what words in the source are most important for generating the next word
  - Inspired by human “attention”: limited in its capacity
- More generally, the same mechanism can be used for defining context as a distribution over tokens
  - We can also think of the previous layer as context for the next layer: this is called ‘self-attention’

# Self-Attention

- Self-attention is the use of attention to generate, from a sequence of word embeddings, a new sequence of word embeddings whose representation is some aggregation of the word embeddings in the previous layer
- Why would we want to do that?

## Attention Visualizations

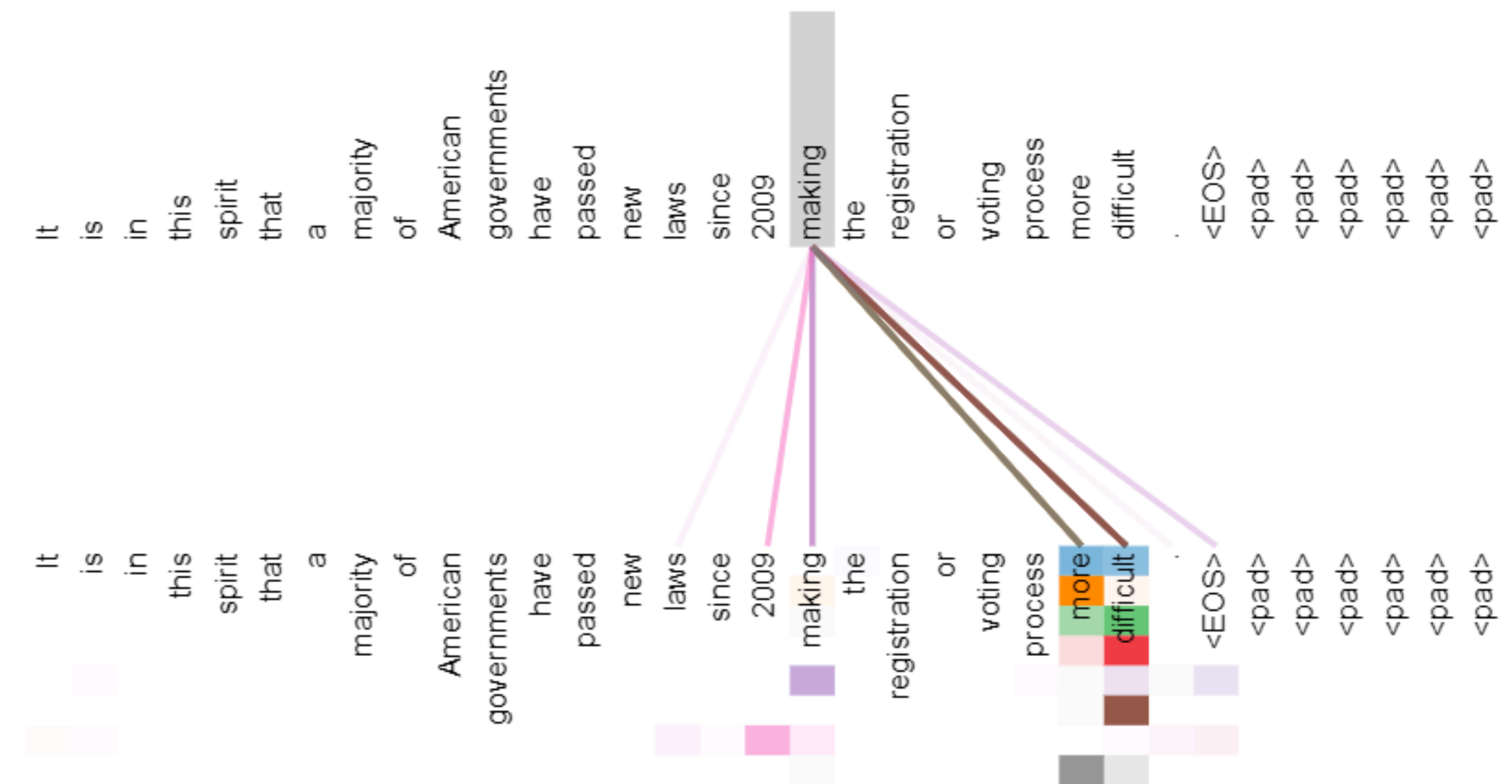


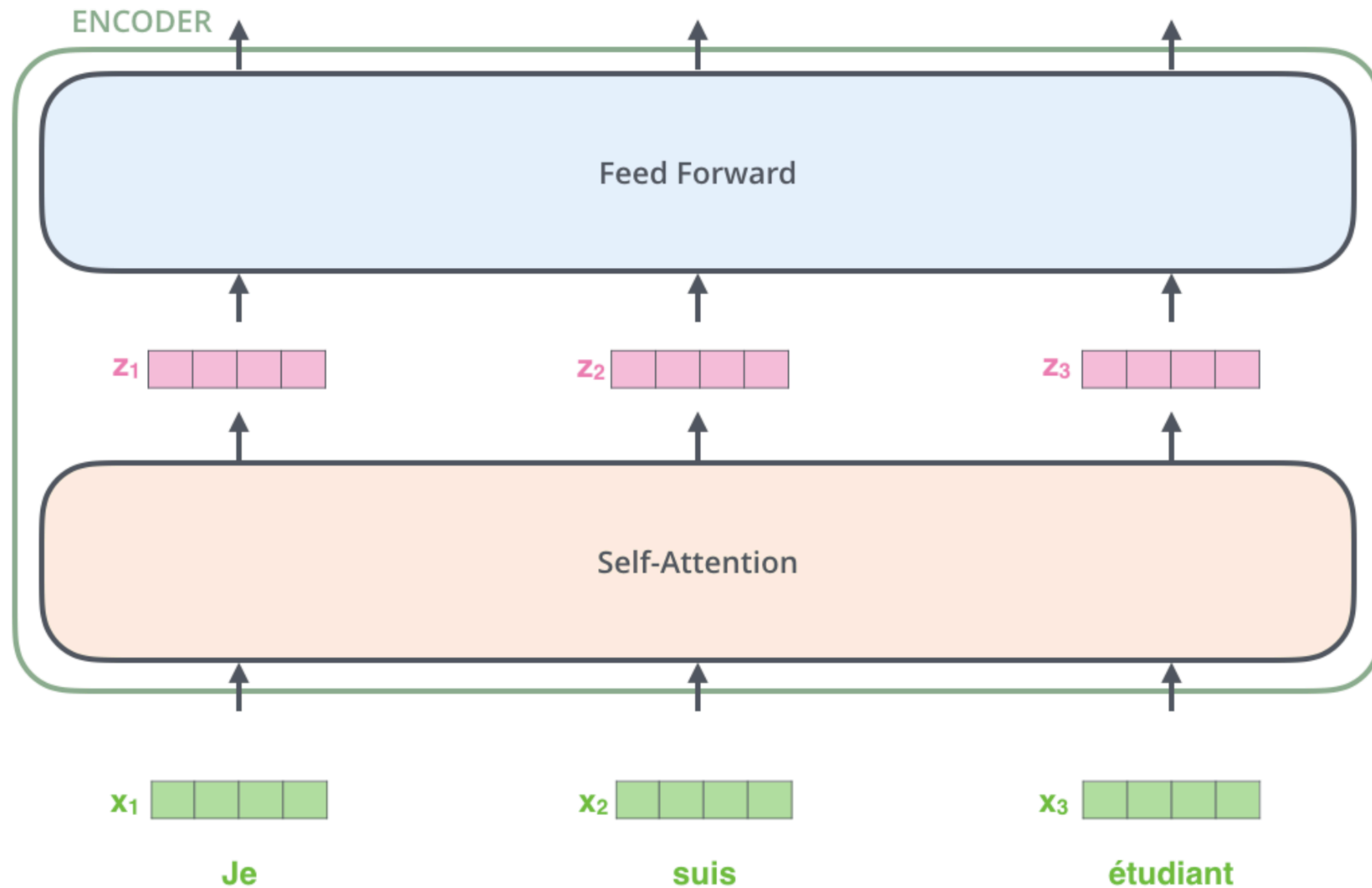
Figure 3: An example of the attention mechanism following long-distance dependencies in the encoder self-attention in layer 5 of 6. Many of the attention heads attend to a distant dependency of the verb 'making', completing the phrase 'making...more difficult'. Attentions here shown only for the word 'making'. Different colors represent different heads. Best viewed in color.

# Self-Attention

---

- Traditional ('non-contextualized') embeddings of a word represent the word form over all its different senses
  - For example, the word *make* could be used in many different senses (e.g., להצחיק — make ... laugh, להקשות — make difficult, להכין — make a cake)
  - The context of a word may disambiguate it, resulting in a representation that is tailored to the sentence it appears in
- Self-attention begins with embedding every word according to its form, and an embedding of its position
- It then iteratively constructs a representation of the word by self-attending to the representations in the previous layer

# The Self-Attention Mechanism



**The Illustrated Transformer**

Jay Alammar

<https://jalammar.github.io/illustrated-transformer/>



# The Illustrated Transformer

Jay Alammar

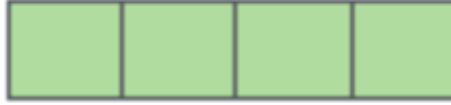
<https://jalammar.github.io/illustrated-transformer/>

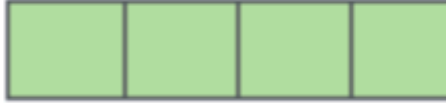
Input

Thinking

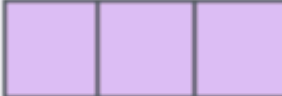
Machines

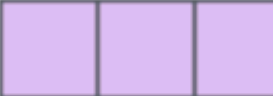
Embedding

$x_1$  

$x_2$  

Queries

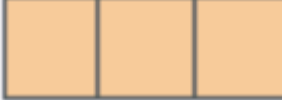
$q_1$  

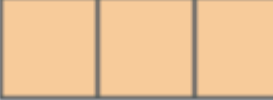
$q_2$  



$W^Q$

Keys


$k_1$  

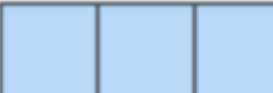
$k_2$  



$W^K$

Values

$v_1$  

$v_2$  



$W^V$

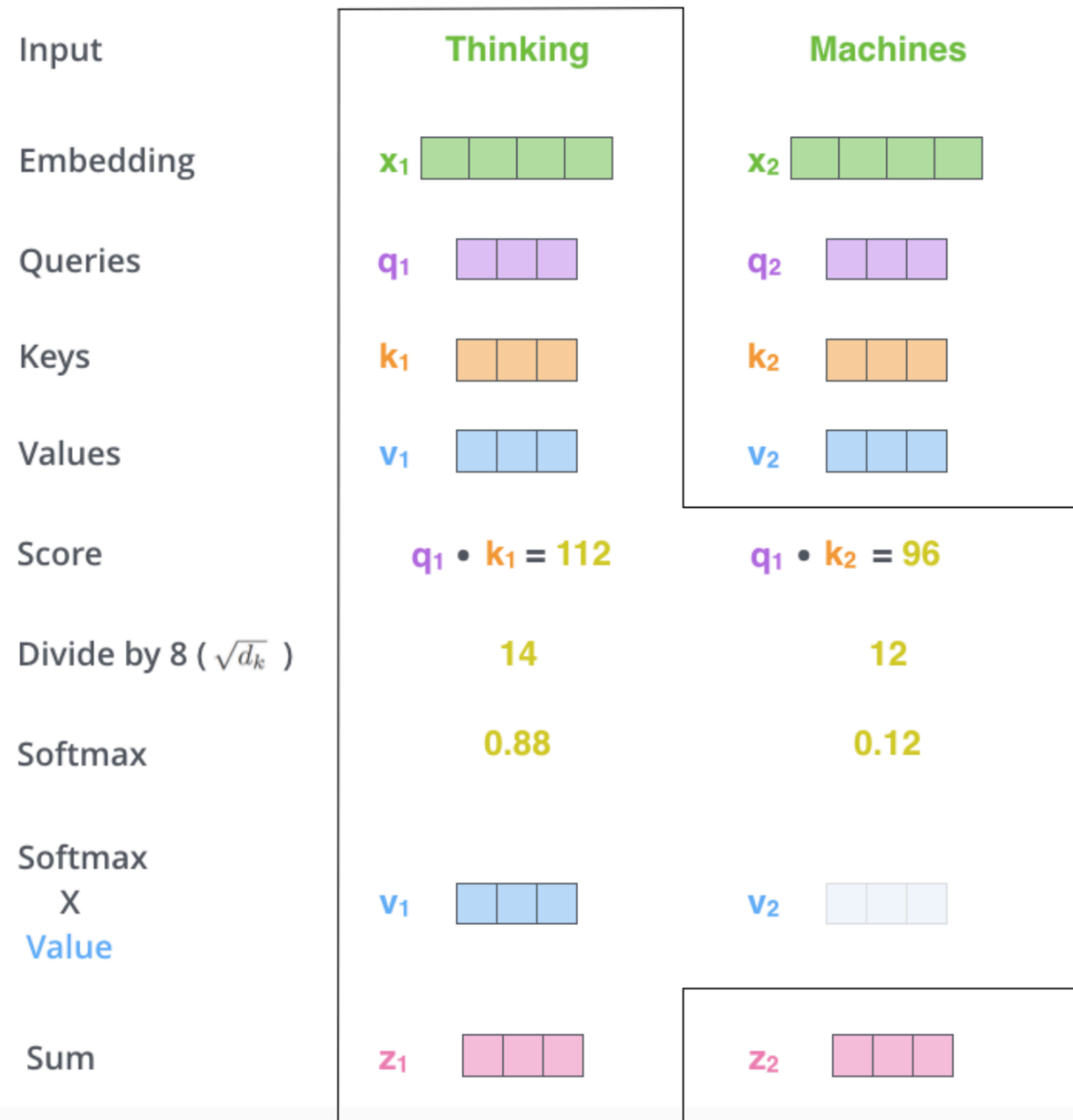
Multiplying  $x_1$  by the  $W^Q$  weight matrix produces  $q_1$ , the "query" vector associated with that word. We end up creating a "query", a "key", and a "value" projection of each word in the input sentence.



# The Illustrated Transformer

Jay Alammar

<https://jalammar.github.io/illustrated-transformer/>

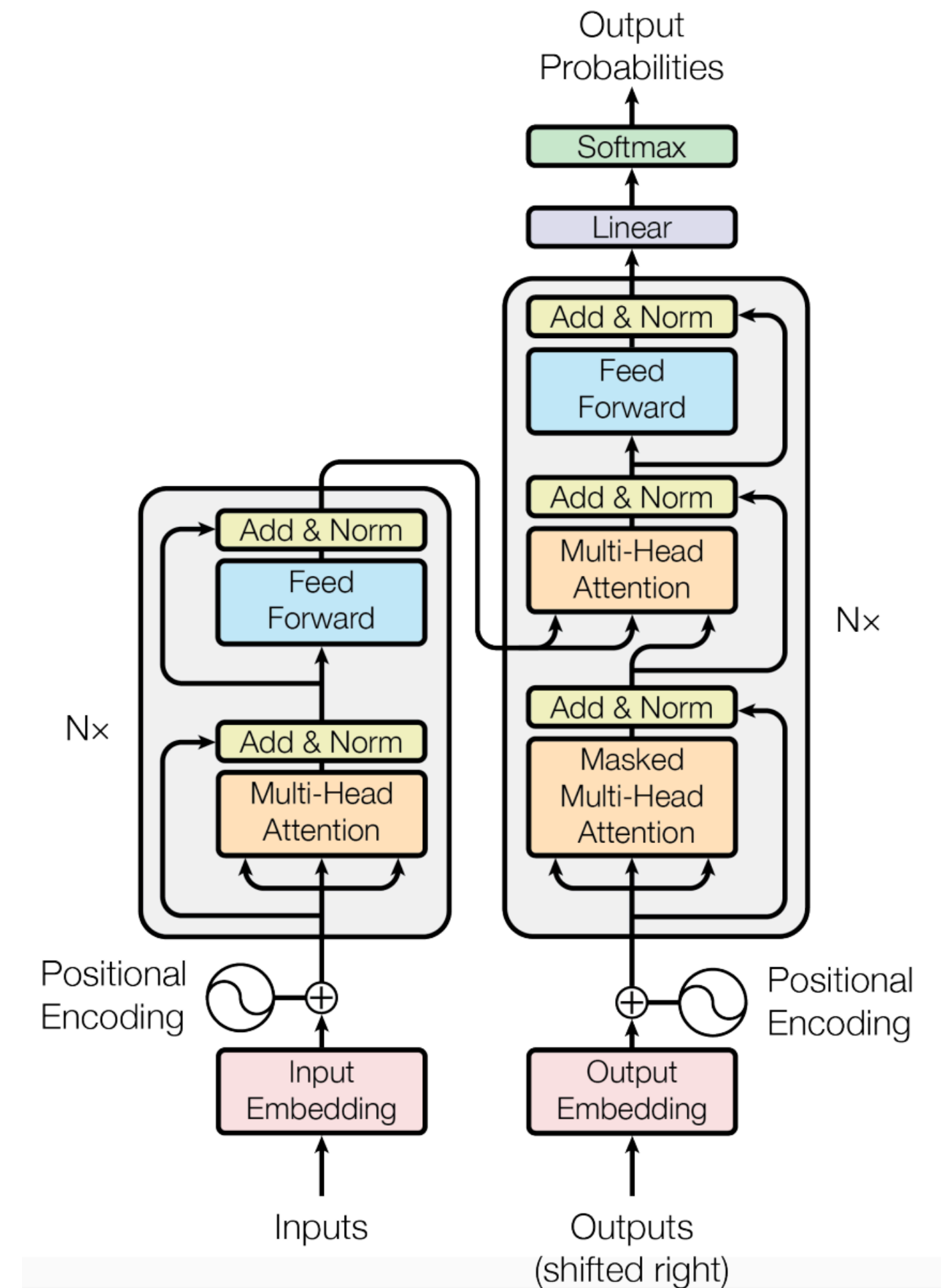


# The Transformer Model

- The Transformer (Vaswani et al. 2017) model is the standard neural model used in NLP today
  - Although it does have several important variants
- It is often used as a sequence-to-sequence model, but is not an RNN
  - This means that like the RNN encoder-decoder, it is essentially a model for predicting the next word, given the input sentence and prefix

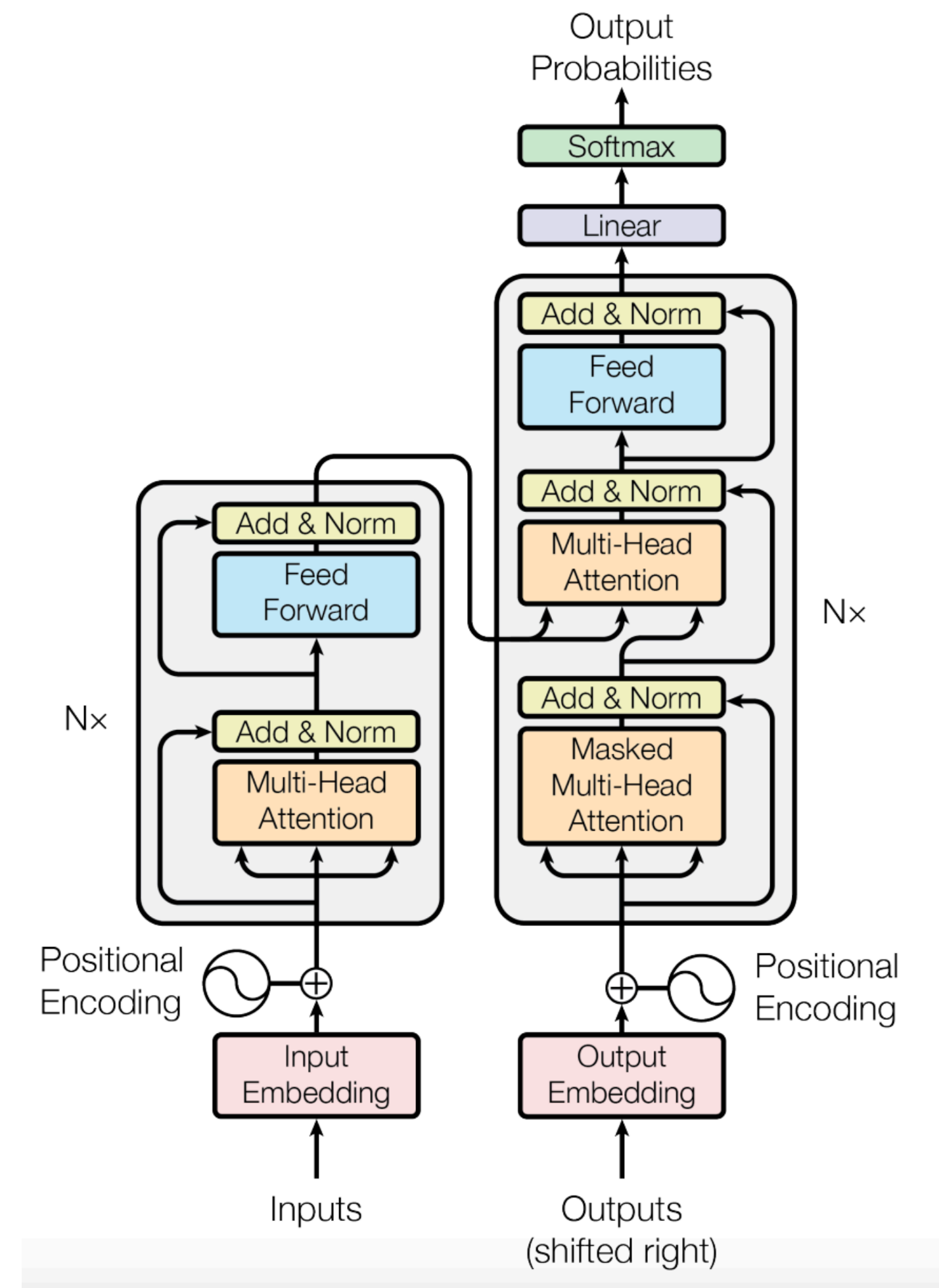
$$p_{\theta}(y_i | y_{i-1}, \dots, y_1, \mathbf{x})$$

- The input size is fixed, and special padding tokens are used to reach uniform input length



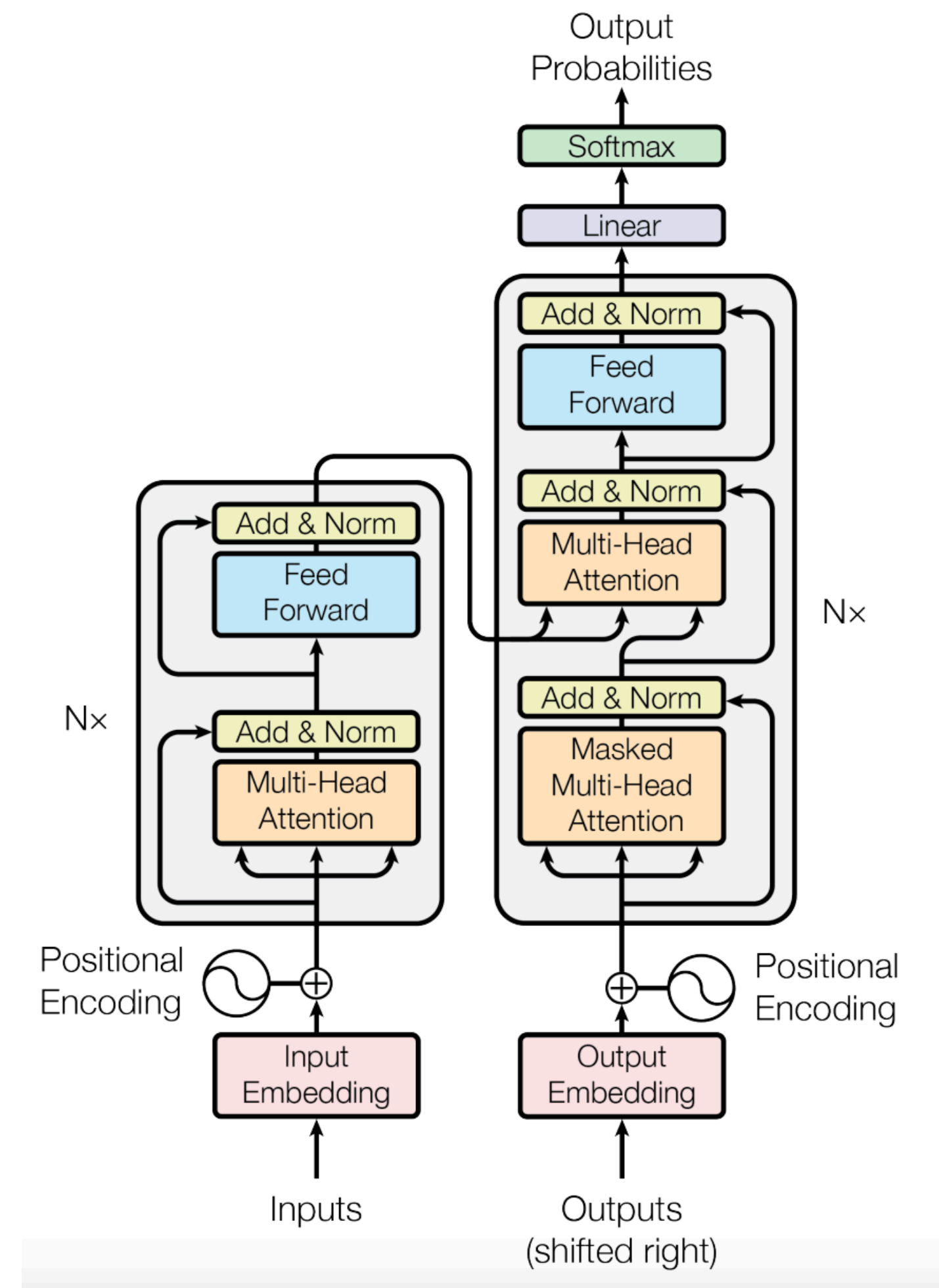
# The Transformer Model

- The Transformer model is divided into blocks. Each block receives a representation of the tokens in the previous layer as vectors in  $\mathcal{R}^d$  and computes a representation of the next layer
- The network is both deep (often double-digit number of layers in the encoder and the decoder) and wide
  - The network uses several attention mechanisms at each layer, and concatenates their output



# The Transformer Model

- The decoder for the model is similar in architecture to the encoder, but with several important differences:
  - Each decoder block includes a block of (multi-head) self-attention, followed by a block of attention over both the encoder input and the output of the self-attention
  - The input to the decoder is the prefix
  - The decoder outputs a probability over the possible next tokens
  - Several other low-level differences

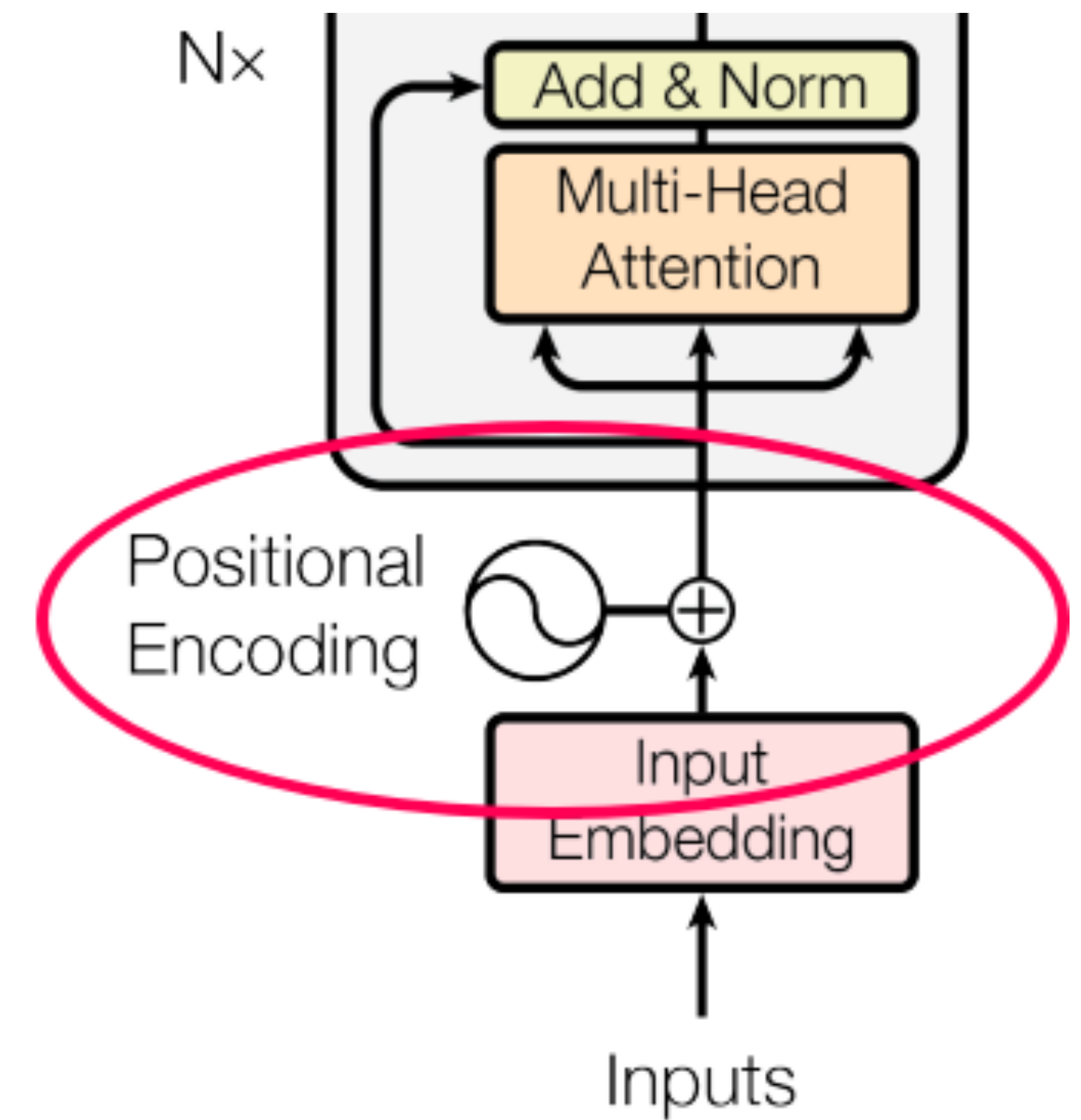




# The Encoding of Positions in Transformers

---

- Tokens are represented in the Transformers as a concatenation of an input embedding (non-contextualized) and a positional embedding (different embedding for every position index)
- The positional embeddings are the only elements that break the symmetry between tokens
- Positional embeddings are randomly initialized and learned during training
- The model can therefore learn to take into consideration any patterns in the relative positions, as long as it is within its context window



# Training and Decoding with the Transformer

---

- The Transformer is trained with standard log probability loss, similar to the ones used for neural language models

$$L(\theta) = \frac{1}{N} \sum_{i=1}^N \log[p_{\theta}(y_i | y_{i-1}, \dots, y_1, \mathbf{x})]$$

- Unlike RNNs, the Transformer does not require backpropogating through time
  - It is essentially a feed-forward network
  - Allows more parallelization in training
- The text is divided into segments using a sliding window (often with overlaps)
- As with RNNs, decoding can be done greedily, through beam search or using more advanced decoding techniques

# Advantages of Transformers

---

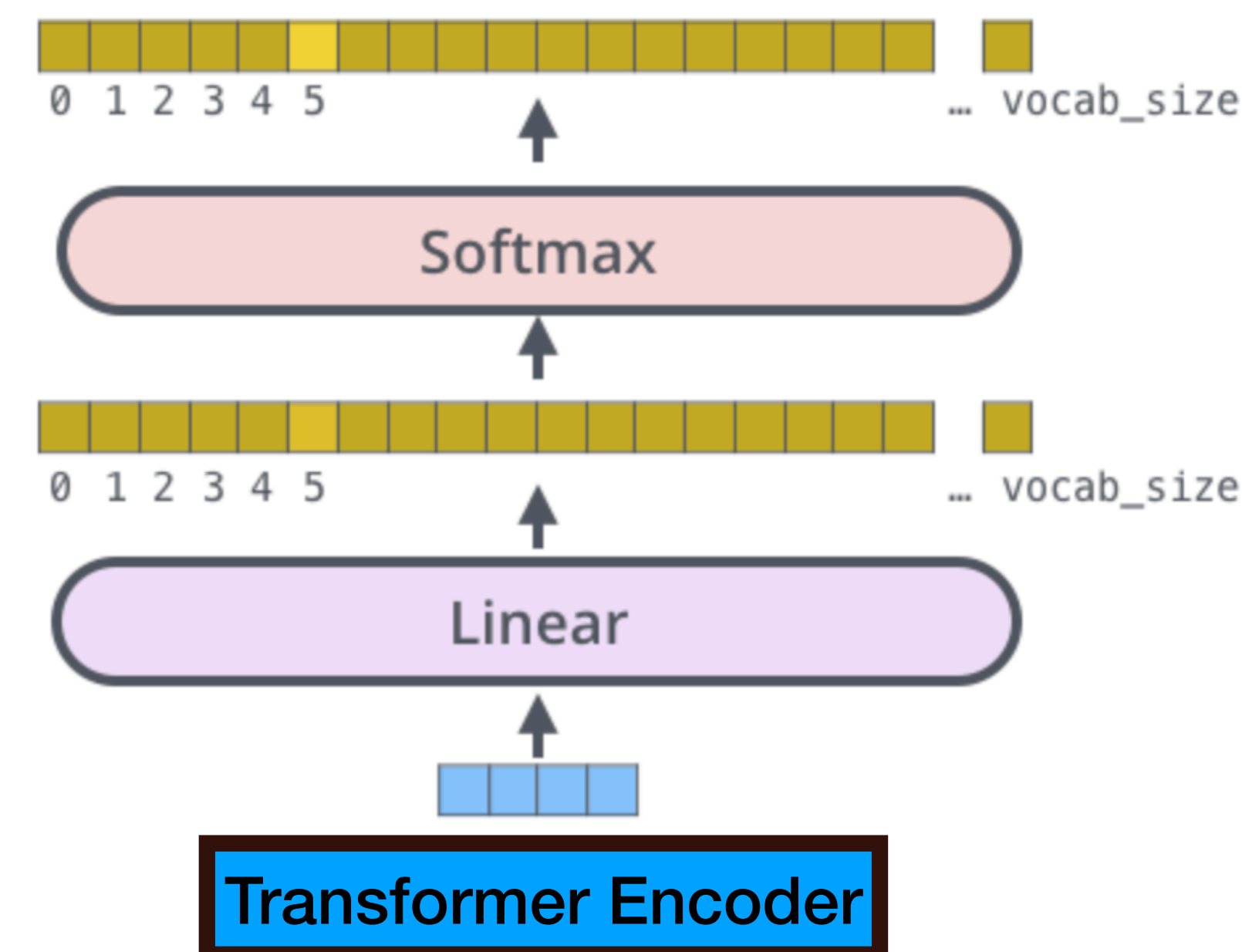
- Transformers have shown superior results over RNNs in many NLP tasks, including Machine Translation
- Much easier to train than RNNs
- Less sensitivity to position
  - The input is given in parallel and not sequentially
  - Positional embeddings are learned



# Back to BERT

- We are now ready to give a more technical presentation of BERT
- Recall: BERT is a masked language model, often used to produce contextualized word embeddings of words in a given document
  - Loss function:

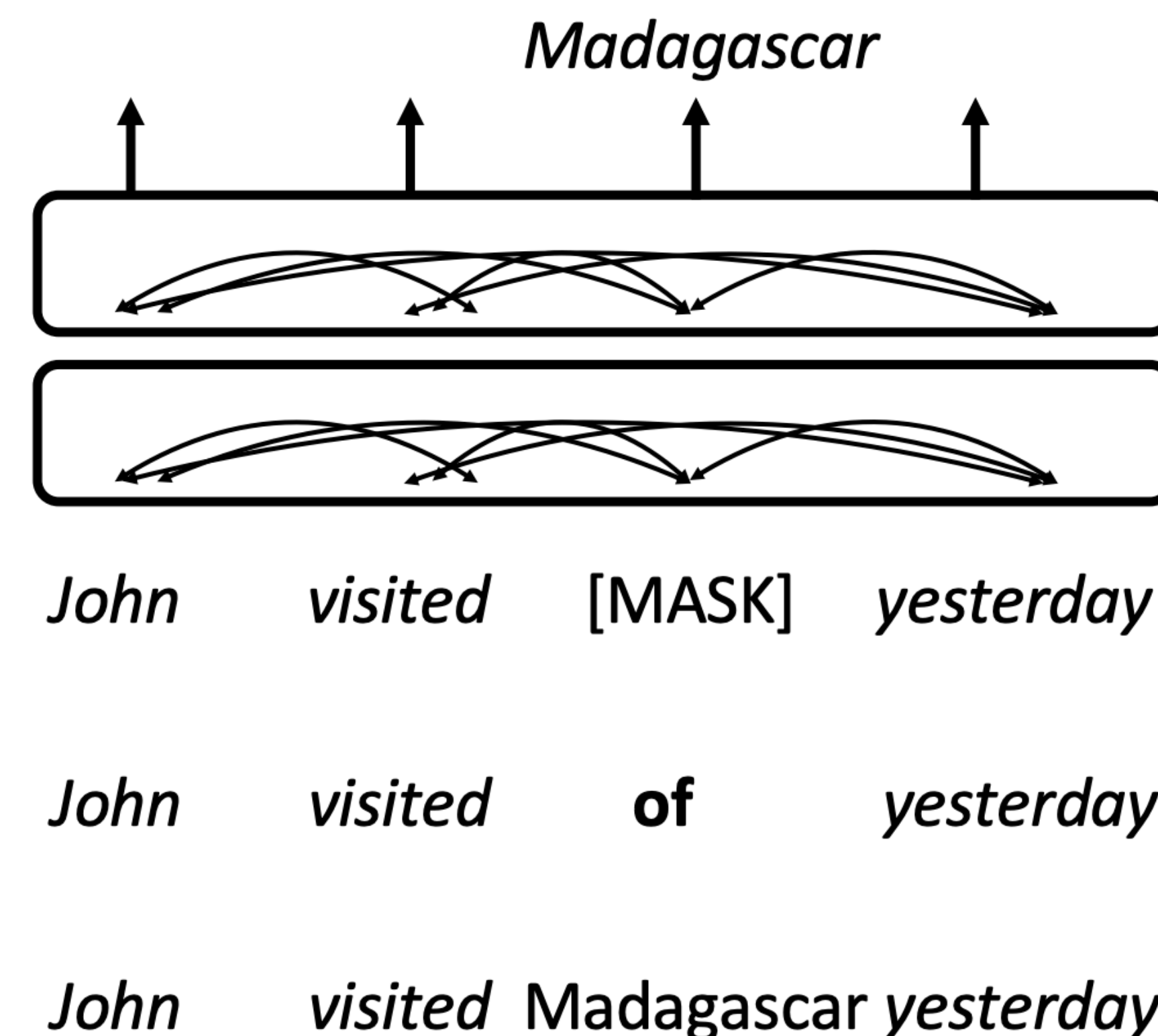
$$L(\theta) = \frac{1}{|S|} \sum_{i \in S} \log(p_{\theta}(x_i | \mathbf{x}'))$$



# Back to BERT

---

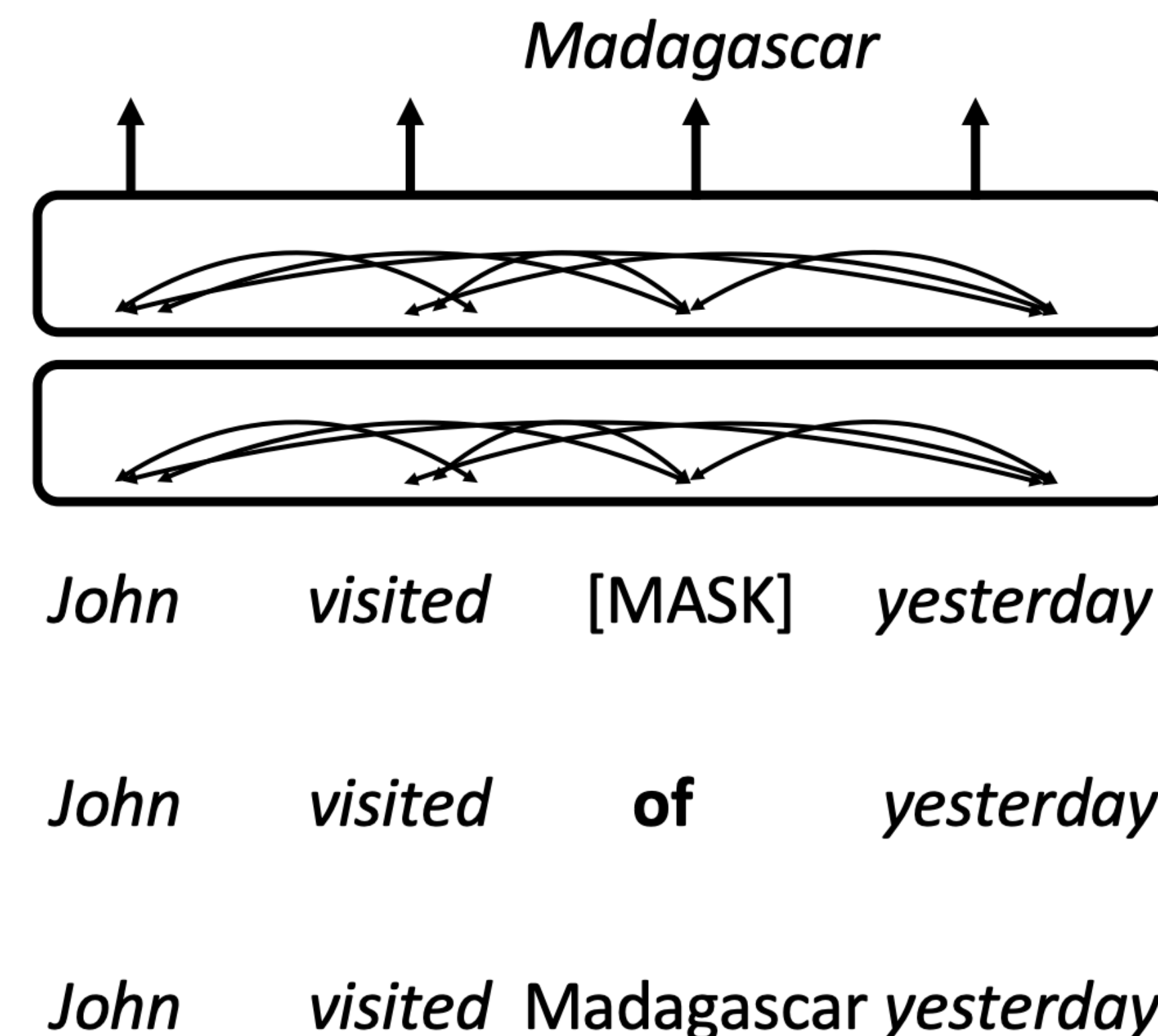
- For 15% of the tokens, predict them (and compute the loss according to the prediction):
  - For 80%, replace the input token with [MASK]
  - For 10%, replace with a random token
  - For 10%, keep the same



# Back to BERT

---

- During decoding, the BERT embeddings of words in a sentence are defined as the activation in the penultimate layer when inputting the unmasked sentence
- BERT's training encourages it to encode information about:
  - the input word itself (to predict it when it is unmasked)
  - the context (to predict the masked tokens, and the ones replaced with random tokens)

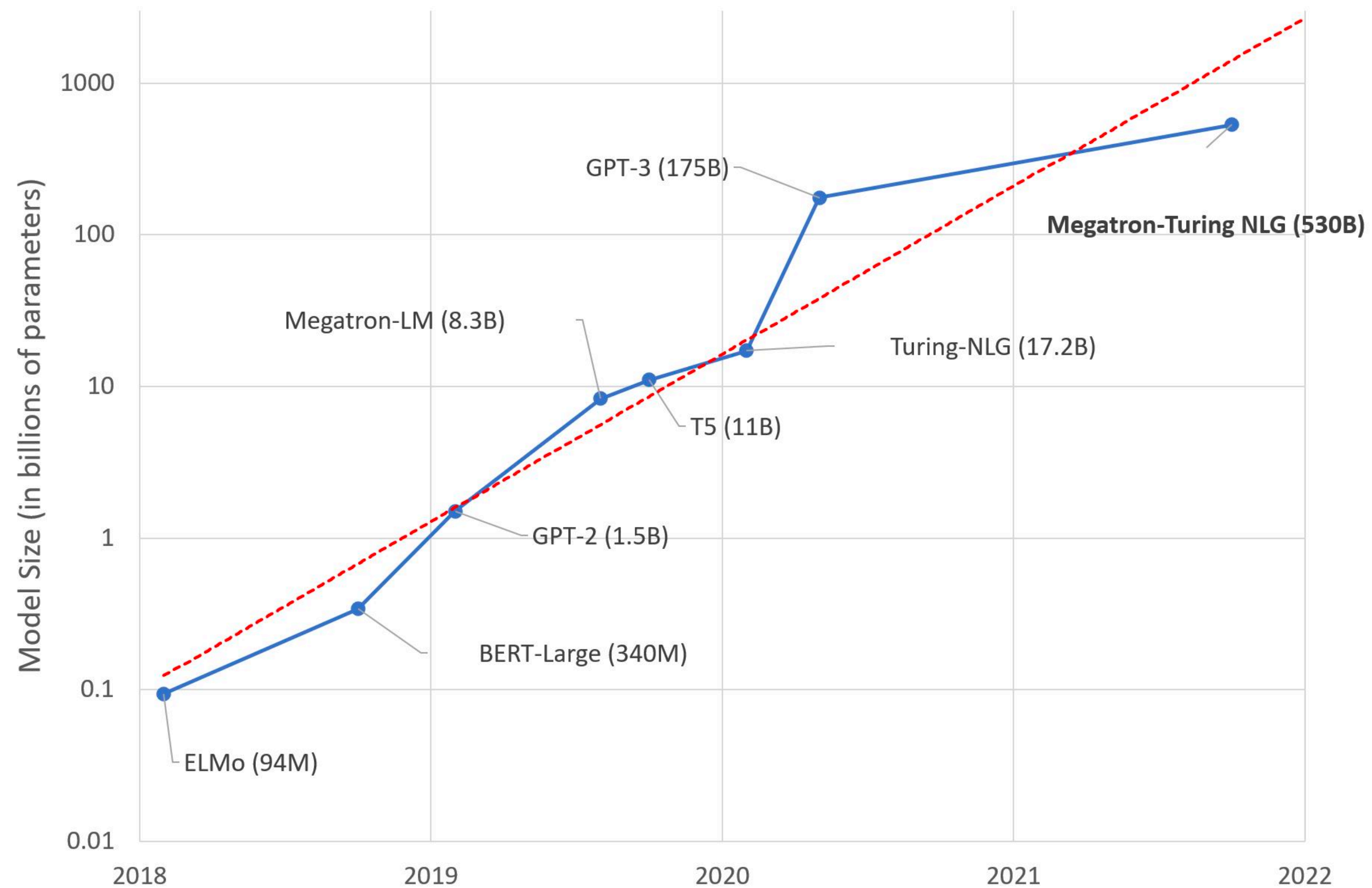


# Transformers: Frontiers

---

- Very large language models (GPT3 and others)
- Few-shot capabilities
- Zero-shot capabilities (domain-general models)

# Very Large Language Models



# Emergent Properties

---

A "whatpu" is a small, furry animal native to Tanzania. An example of a sentence that uses the word whatpu is:

We were traveling in Africa and we saw these very cute whatpus.

---

To do a "farduddle" means to jump up and down really fast. An example of a sentence that uses the word farduddle is:

One day when I was playing tag with my little sister, she got really excited and she started doing these crazy farduddles.

A "yalubalu" is a type of vegetable that looks like a big pumpkin. An example of a sentence that uses the word yalubalu is:

I was on a trip to Africa and I tried this yalubalu vegetable that was grown in a garden there. It was delicious.

---

A "Burringo" is a car with very fast acceleration. An example of a sentence that uses the word Burringo is:

In our garage we have a Burringo that my father drives to work every day.

---

A "Gigamuru" is a type of Japanese musical instrument. An example of a sentence that uses the word Gigamuru is:

I have a Gigamuru that my uncle gave me as a gift. I love to play it at home.

---

To "screeg" something is to swing a sword at it. An example of a sentence that uses the word screeg is:

We screeghed at each other for several minutes and then we went outside and ate ice cream.

Few-shot  
learning: using a  
new word in a  
sentence



Language Models are Few-Shot Learners  
Brown et al., 2020  
<https://arxiv.org/pdf/2005.14165.pdf>






# Emergent Properties

---

- Zero-shot capabilities: a domain-general language model (can do anything if asked to)
- More of an ideal than an accomplished goal
- ChatGPT can be seen as the latest such attempt

 write a biblical verse in the style of the king james bible explaining how to remove a peanut butter sandwich from a VCR 

 And it came to pass that a man was troubled by a peanut butter sandwich, for it had been placed within his VCR, and he knew not how to remove it.  

And he cried out to the Lord, saying, "Oh, Lord, how can I remove this sandwich from my VCR, for it is stuck fast and will not budge?"

And the Lord spoke unto him, saying, "Fear not, my child, for I shall guide thy hand and show thee the way. Take thy butter knife, and carefully insert it between the sandwich and the VCR, and gently pry them apart. And with patience and perseverance, the sandwich shall be removed, and thy VCR shall be saved."

And the man did as the Lord commanded, and lo and behold, the sandwich was removed from the VCR, and the man was saved.

And the Lord said, "Verily I say unto thee, seek not to put thy peanut butter sandwiches in thy VCR, for it is not a suitable place for such things. Rather, keep thy sandwiches in thy refrigerator or on thy plate, where they belong."

And the man heeded the Lord's words, and from that day forth, he kept his sandwiches in their proper place, and was saved from trouble and woe. Amen.