

Deep Learning

14 Self-Attention & Transformers

Dr. Konda Reddy Mopuri
Dept. of AI, IIT Hyderabad
Jan-May 2024

Motivation

- ① Why does one need to think beyond LSTMs?

Motivation

- ① Why does one need to think beyond LSTMs?
- ② Sequential processing doesn't allow parallelization
 - Path length = $\mathcal{O}(n)$
 - RNNs need $\mathcal{O}(n)$ steps to process a sentence of length n

Motivation

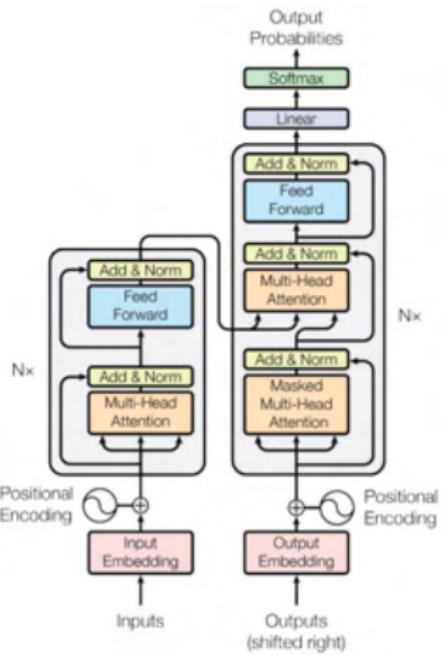
- ① (Despite the LSTM/GRU) RNNs need attention to deal with long-range dependencies

Motivation

- ① (Despite the LSTM/GRU) RNNs need attention to deal with long-range dependencies
- ② Since attention enables accesses to any state, do we need RNNs?

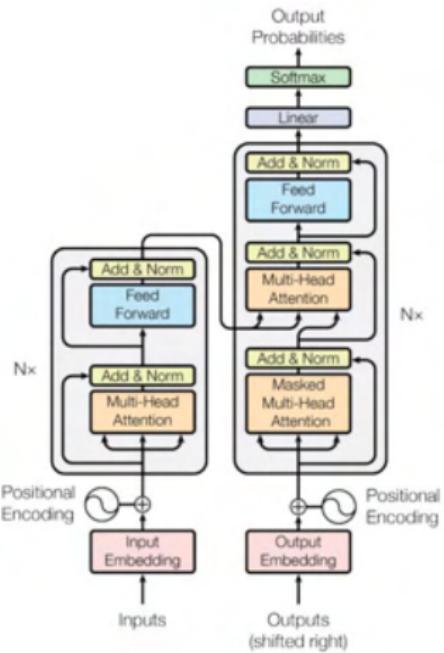
Transformers

- ① Introduced by Vaswani et al.
NeurIPS 2017



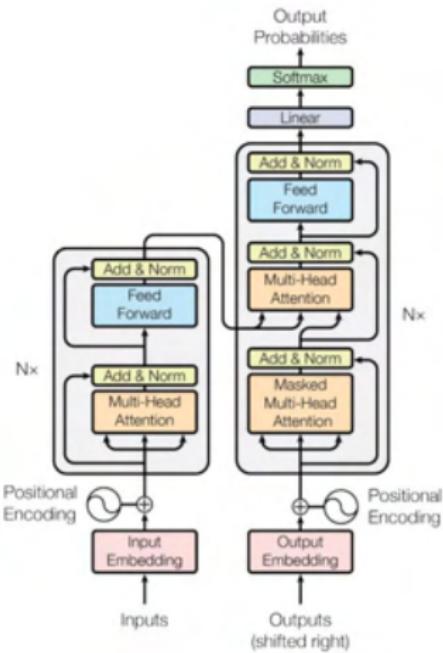
Transformers

- ① Introduced by Vaswani et al.
NeurIPS 2017
- ② Sequence to sequence modeling without RNNs



Transformers

- ① Introduced by Vaswani et al.
NeurIPS 2017
 - ② Sequence to sequence modeling
without RNNs
 - ③ Transformer model is built on
self-attention (no recurrence or
convolutions)



Transformers

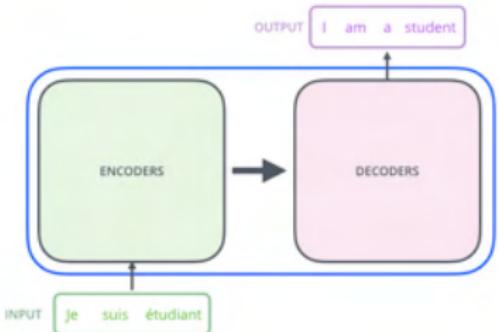


Credits: Jay Alammar

Transformers

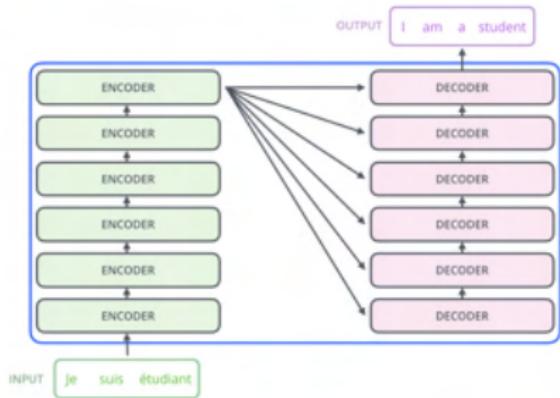


Credits: Jay Alammar



Credits: Jay Alammar

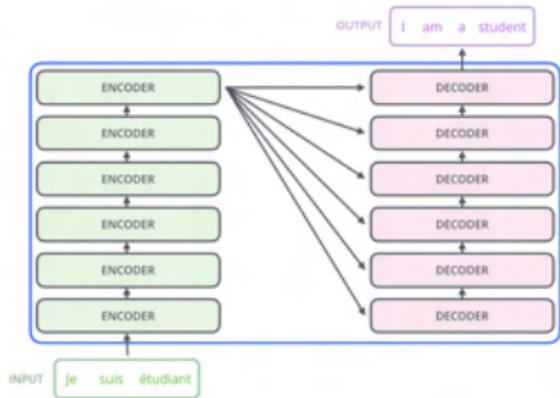
Transformers



- ① Encoding module has a stack of encoders

Credits: Jay Alammar

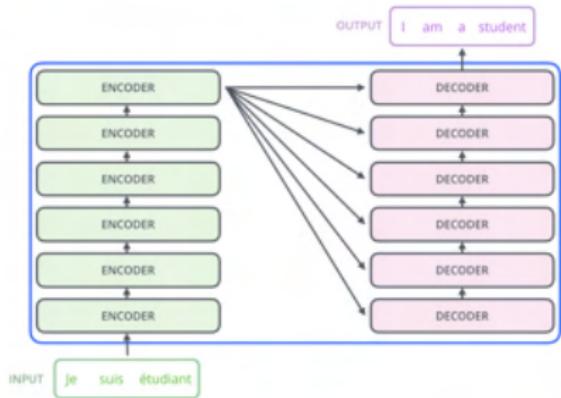
Transformers



- ① Encoding module has a stack of encoders
- ② Same structure different parameters

Credits: Jay Alammar

Transformers

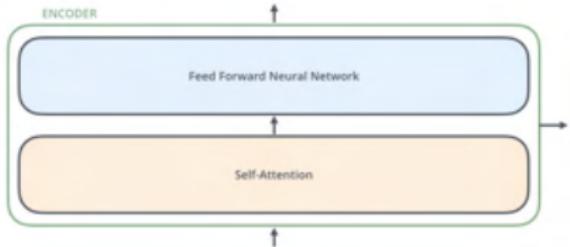


- ① Encoding module has a stack of encoders
- ② Same structure different parameters
- ③ Similarly the decoding module

Credits: Jay Alammar

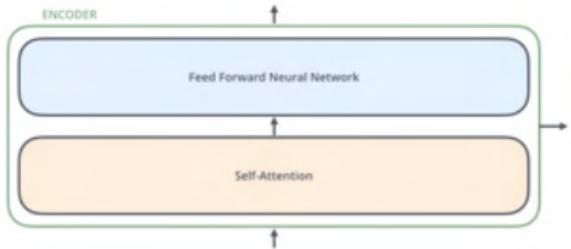
Transformers

- ① Encoder first has a self-attention layer



Credits: [Jay Alammar](#)

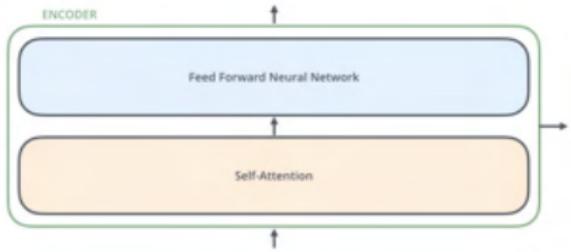
Transformers



- ① Encoder first has a self-attention layer
- ② Looks at the other words while encoding a specific word

Credits: [Jay Alammar](#)

Transformers



- ① Encoder first has a self-attention layer
- ② Looks at the other words while encoding a specific word
- ③ Next a (same) feed-forward NN is applied at all positions

Credits: [Jay Alammar](#)

Attention vs Self-attention

- ① Encoder-Decoder Attention looks

Attention vs Self-attention

- ① Encoder-Decoder Attention looks
- ② **From:** a decoder (current) state

Attention vs Self-attention

- ① Encoder-Decoder Attention looks
- ② **From:** a decoder (current) state
- ③ **To:** all the encoder states

Attention vs Self-attention

- ① Encoder-Decoder Attention looks
 - ② **From:** a decoder (current) state
 - ③ **To:** all the encoder states
- ① Self-Attention looks

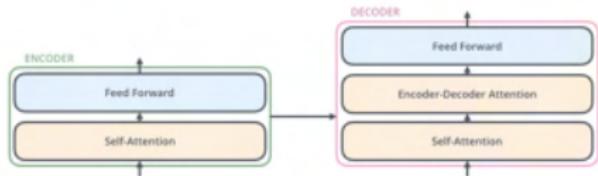
Attention vs Self-attention

- ① Encoder-Decoder Attention looks
- ② **From:** a decoder (current) state
- ③ **To:** all the encoder states
- ① Self-Attention looks
- ② **From:** each state from a set of states

Attention vs Self-attention

- ① Encoder-Decoder Attention looks
- ② **From:** a decoder (current) state
- ③ **To:** all the encoder states
- ① Self-Attention looks
- ② **From:** each state from a set of states
- ③ **To:** all other states in the same set

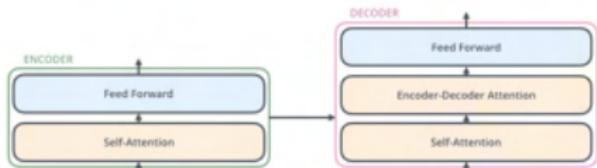
Transformers



- ① Decoder also has both the layers

Credits: Jay Alammar

Transformers



Credits: Jay Alammar

- ① Decoder also has both the layers
- ② But, in the middle it has an encoder-decoder attention layer

Why the name ‘Transformer’?

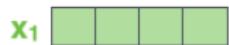
- ① Transforms a set of vectors in some representation space into a corresponding set of vectors (same dimensionality) in some new space

Why the name ‘Transformer’?

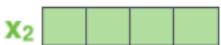
- ① Transforms a set of vectors in some representation space into a corresponding set of vectors (same dimensionality) in some new space
- ② **Goal:** new space will have a **richer internal representation** that is better suited to solve the downstream task

Transformers-Encoding

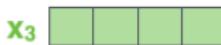
- ① Start with turning each word into a vector at the bottom-most encoder



Je



suis

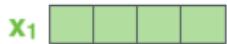


étudiant

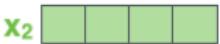
Credits: Jay Alammar

Transformers-Encoding

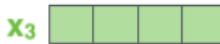
- ① Start with turning each word into a vector at the bottom-most encoder
- ② Others receive a list of vectors from the encoder immediately below



Je



suis



étudiant

Credits: Jay Alammar

Self-attention in Encoder vs. Decoder

- ① Who is doing: all source tokens

Self-attention in Encoder vs. Decoder

- ① Who is doing: all source tokens
- ② What are they doing (**repeat**)
 - look at each other
 - update representations

Self-attention in Encoder vs. Decoder

- ① Who is doing: all source tokens ① Decoder
- ② What are they doing (**repeat**)
 - look at each other
 - update representations

Self-attention in Encoder vs. Decoder

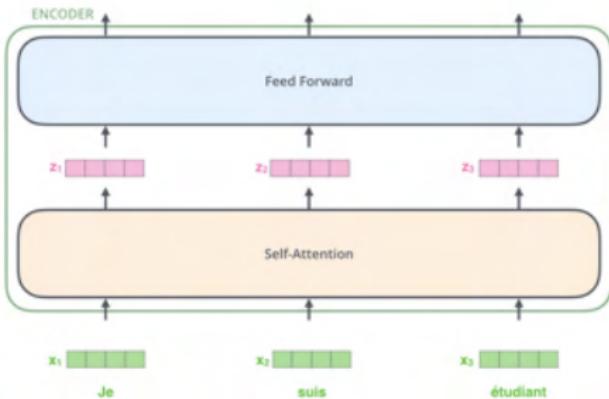
- ① Who is doing: all source tokens
 - ② What are they doing (**repeat**)
 - look at each other
 - update representations
-
- ① Decoder
 - ② Who is doing: target token at each time step

Self-attention in Encoder vs. Decoder

- | | |
|---|---|
| <ul style="list-style-type: none">① Who is doing: all source tokens② What are they doing (repeat)<ul style="list-style-type: none">• look at each other• update representations | <ul style="list-style-type: none">① Decoder② Who is doing: target token at each time step③ What are they doing (repeat)<ul style="list-style-type: none">• looks at previous target tokens (self-attention)• looks at source representations (encoder-decoder attention)• update representation |
|---|---|

Transformers- Encoding

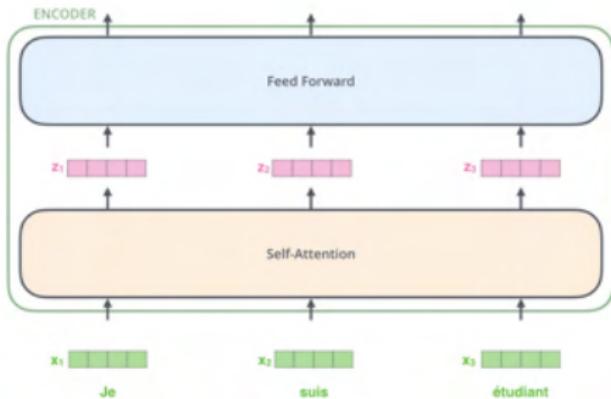
- ① Each word flows through the two layers of the encoder through its own path



Credits: Jay Alammar

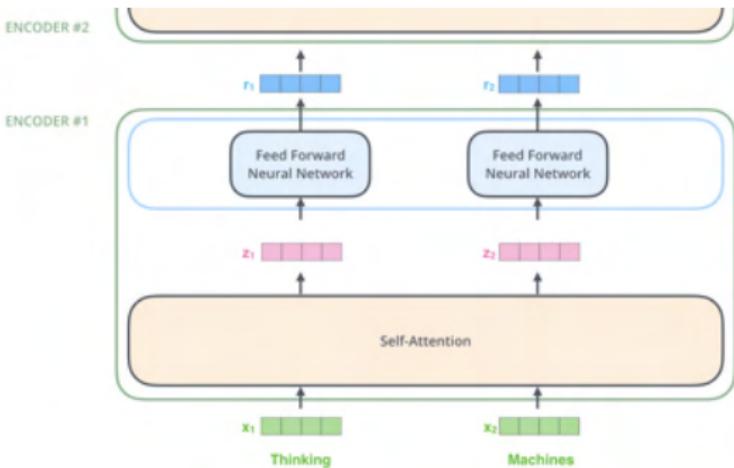
Transformers- Encoding

- ① Each word flows through the two layers of the encoder through its own path
- ② Self-attention layer has dependencies among them. However, the path length is $\mathcal{O}(1)$



Credits: Jay Alammar

Transformers- Encoding



Credits: Jay Alammar

Self-Attention

- ① The animal didn't cross the street because it was too tired
- ② The animal didn't cross the street because it was too wide

Self-Attention

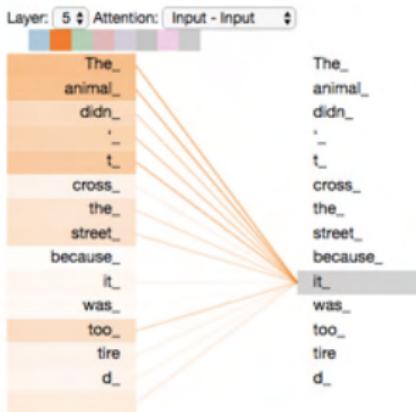
- ① The animal didn't cross the street because it was too tired
- ② The animal didn't cross the street because it was too wide
- ③ What does 'it' refer to?

Self-Attention

- ① The animal didn't cross the street because it was too tired
- ② The animal didn't cross the street because it was too wide
- ③ What does 'it' refer to?
- ④ Easy for humans, but not so much for the traditional Seq2Seq models

Self-Attention

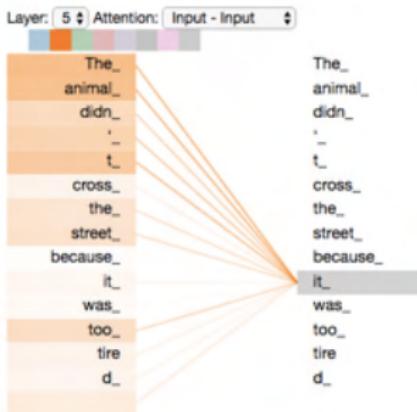
- ① As the model processes each word, self-attention attends other positions in the i/p sequence to encode better



Credits: Jay Alammar

Self-Attention

- ① As the model processes each word, self-attention attends other positions in the i/p sequence to encode better
- ② Unlike RNNs, we don't keep hidden states from previous positions here!



Credits: Jay Alammar

Attention weights

- ① Input tokens x_1, x_2, \dots, x_N

Attention weights

- ① Input tokens x_1, x_2, \dots, x_N
- ② Output tokens y_1, y_2, \dots, y_N

Attention weights

- ① Input tokens x_1, x_2, \dots, x_N
- ② Output tokens y_1, y_2, \dots, y_N
- ③ $y_n = \sum_{m=1}^N a_{nm} \cdot x_m$

Attention weights

- ① Input tokens x_1, x_2, \dots, x_N
- ② Output tokens y_1, y_2, \dots, y_N
- ③ $y_n = \sum_{m=1}^N a_{nm} \cdot x_m$
- ④ $a_{mn} \geq 0$ and $\sum_{m=1}^N a_{mn} = 1$ Why?

How to compute the Attention weights?

- ① A simple way is to use the 'dot product' self-attention

How to compute the Attention weights?

- ① A simple way is to use the ‘dot product’ self-attention

② $a_{nm} = \frac{\exp(\mathbf{x}_n^T \mathbf{x}_m)}{\sum_{m'=1}^N \exp(\mathbf{x}_n^T \mathbf{x}'_m)}$

How to compute the Attention weights?

① A simple way is to use the ‘dot product’ self-attention

$$② a_{nm} = \frac{\exp(\mathbf{x}_n^T \mathbf{x}_m)}{\sum_{m'=1}^N \exp(\mathbf{x}_n^T \mathbf{x}'_{m'})}$$

$$③ \mathbf{Y} = \text{Softmax}[\mathbf{X}\mathbf{X}^T]\mathbf{X}$$

How to compute the Attention weights?

- ① A simple way is to use the ‘dot product’ self-attention
- ② $a_{nm} = \frac{\exp(\mathbf{x}_n^T \mathbf{x}_m)}{\sum_{m'=1}^N \exp(\mathbf{x}_n^T \mathbf{x}'_m)}$
- ③ $\mathbf{Y} = \text{Softmax}[\mathbf{X}\mathbf{X}^T]\mathbf{X}$
- ④ The transformation from \mathbf{X} to \mathbf{Y} is fixed, has no capacity to learn from the data

How to compute the Attention weights?

- ① A simple way is to use the ‘dot product’ self-attention
- ② $a_{nm} = \frac{\exp(\mathbf{x}_n^T \mathbf{x}_m)}{\sum_{m'=1}^N \exp(\mathbf{x}_n^T \mathbf{x}'_m)}$
- ③ $\mathbf{Y} = \text{Softmax}[\mathbf{X}\mathbf{X}^T]\mathbf{X}$
- ④ The transformation from \mathbf{X} to \mathbf{Y} is fixed, has no capacity to learn from the data
- ⑤ Each of the feature values in a token plays an equal role in determining the attention weights

Some terminology from Information Retrieval

- ① Consider a streaming platform and the problem of choosing which movie to watch

Some terminology from Information Retrieval

- ① Consider a streaming platform and the problem of choosing which movie to watch
- ② One approach would be

Some terminology from Information Retrieval

- ① Consider a streaming platform and the problem of choosing which movie to watch
- ② One approach would be
- ③ Associate each movie with a list of attributes (genre, actors, technicians, length, year, etc.) → **Key**

Some terminology from Information Retrieval

- ① Consider a streaming platform and the problem of choosing which movie to watch
- ② One approach would be
- ③ Associate each movie with a list of attributes (genre, actors, technicians, length, year, etc.) → **Key**
- ④ These form a catalog of movies to be searched for

Some terminology from Information Retrieval

- ① Consider a streaming platform and the problem of choosing which movie to watch
- ② One approach would be
- ③ Associate each movie with a list of attributes (genre, actors, technicians, length, year, etc.) → **Key**
- ④ These form a catalog of movies to be searched for
- ⑤ The movie file itself is the **Value**

Some terminology from Information Retrieval

- ① Consider a streaming platform and the problem of choosing which movie to watch
- ② One approach would be
- ③ Associate each movie with a list of attributes (genre, actors, technicians, length, year, etc.) → **Key**
- ④ These form a catalog of movies to be searched for
- ⑤ The movie file itself is the **Value**
- ⑥ User's input of desired attributes → **Query**

Self-Attention

① $\mathbf{Q} = \mathbf{XW}^{(q)}$

Self-Attention

- ① $\mathbf{Q} = \mathbf{XW}^{(q)}$
- ② $\mathbf{K} = \mathbf{XW}^{(k)}$

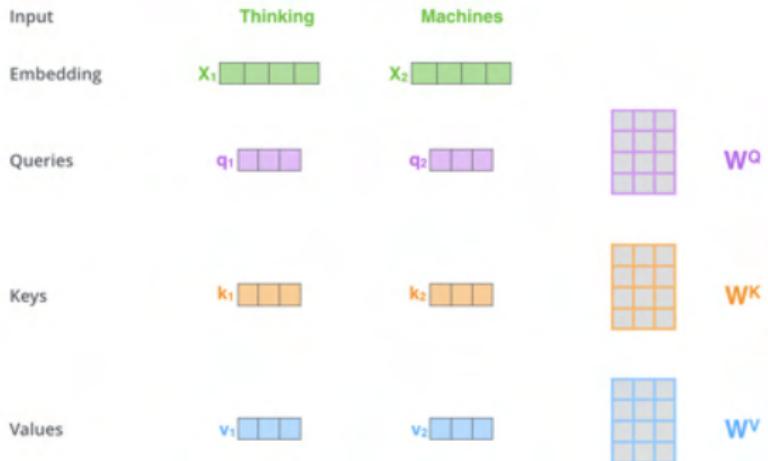
Self-Attention

- ① $\mathbf{Q} = \mathbf{XW}^{(q)}$
- ② $\mathbf{K} = \mathbf{XW}^{(k)}$
- ③ $\mathbf{V} = \mathbf{XW}^{(v)}$

Self-Attention

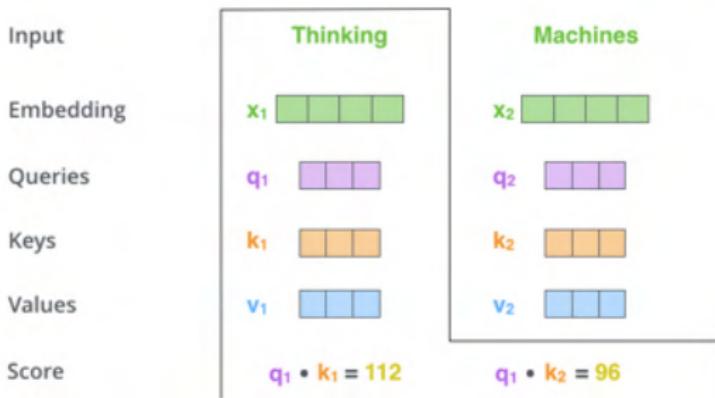
- ① $\mathbf{Q} = \mathbf{XW}^{(q)}$
- ② $\mathbf{K} = \mathbf{XW}^{(k)}$
- ③ $\mathbf{V} = \mathbf{XW}^{(v)}$
- ④ $\mathbf{Y} = \text{Softmax}[\mathbf{QK}^T]\mathbf{V}$

Self-Attention



Credits: Jay Alammar

Self-Attention



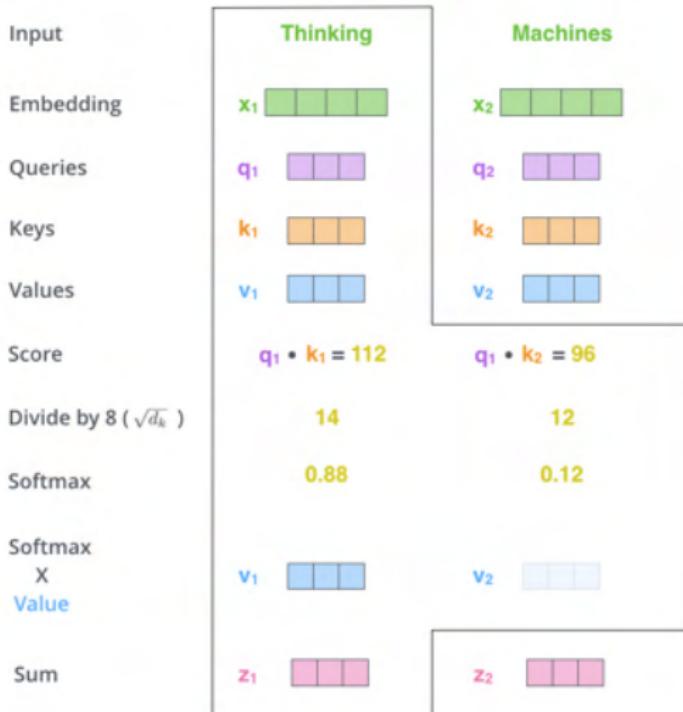
Credits: Jay Alammar

Self-Attention

| Input | Thinking | Machines |
|------------------------------|-----------------------|----------------------|
| Embedding | x_1 | x_2 |
| Queries | q_1 | q_2 |
| Keys | k_1 | k_2 |
| Values | v_1 | v_2 |
| Score | $q_1 \cdot k_1 = 112$ | $q_1 \cdot k_2 = 96$ |
| Divide by 8 ($\sqrt{d_k}$) | 14 | 12 |
| Softmax | 0.88 | 0.12 |

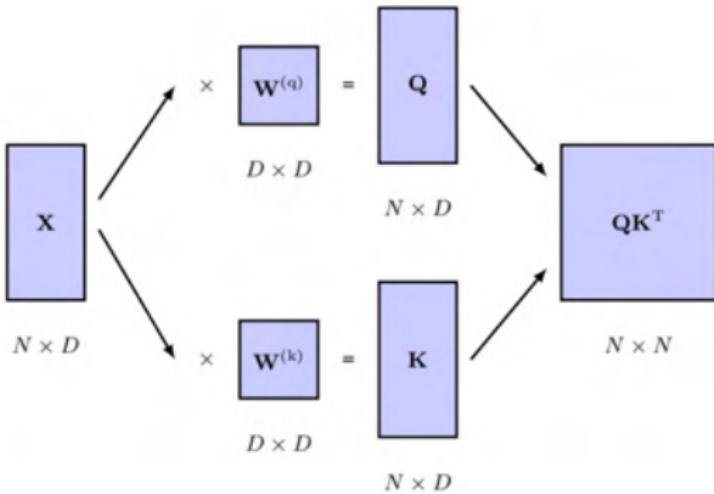
Credits: Jay Alammar

Self-Attention



Credits: Jay Alammar

Self-Attention



Credits: The Bishop's book

Self-Attention

$$\mathbf{Y} = \text{Softmax} \left\{ \begin{matrix} \mathbf{QK}^T \\ N \times N \end{matrix} \right\} \times \mathbf{V} \quad N \times D_v$$

The diagram illustrates the computation of Self-Attention. On the left, a vertical matrix \mathbf{Y} of size $N \times D_v$ is shown. It has a small orange square at its top-left corner. To its right is the formula for Self-Attention. The formula consists of three components: 1) A vertical matrix \mathbf{QK}^T of size $N \times N$, which is the result of applying a query matrix \mathbf{Q} and a key matrix \mathbf{K}^T . This matrix is highlighted with a blue background and an orange border. 2) A brace indicating the dimension $N \times N$. 3) A multiplication sign (\times) followed by a vertical matrix \mathbf{V} of size $N \times D_v$, which is highlighted with a blue background and an orange border.

Credits: The Bishop's book

Scaled Self-Attention

- ① Gradients of the softmax become exponentially small for large input magnitudes

Scaled Self-Attention

- ① Gradients of the softmax become exponentially small for large input magnitudes
- ② To prevent this, the \mathbf{QK}^T is scaled before the softmax

Scaled Self-Attention

- ① Gradients of the softmax become exponentially small for large input magnitudes
- ② To prevent this, the \mathbf{QK}^T is scaled before the softmax
- ③ If the elements of q and v vectors are independent $N(0, 1)$ distributed, the variance of the dot product $\rightarrow D_k$

Scaled Self-Attention

- ① Gradients of the softmax become exponentially small for large input magnitudes
 - ② To prevent this, the \mathbf{QK}^T is scaled before the softmax
 - ③ If the elements of q and v vectors are independent $N(0, 1)$ distributed, the variance of the dot product $\rightarrow D_k$
 - ④ Hence, normalize the product by the standard deviation
- $$\mathbf{Y} = \text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}\left[\frac{\mathbf{QK}^T}{\sqrt{D_k}}\right]\mathbf{V}$$

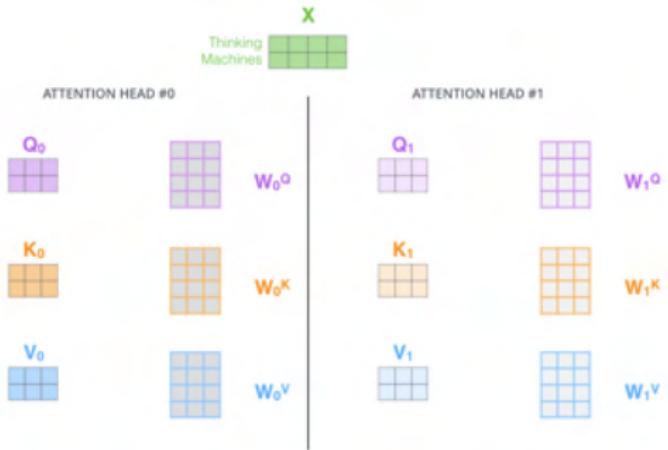
Multi-headed Self-Attention

- ① There may be multiple patterns of attention that are relevant at the same time

Multi-headed Self-Attention

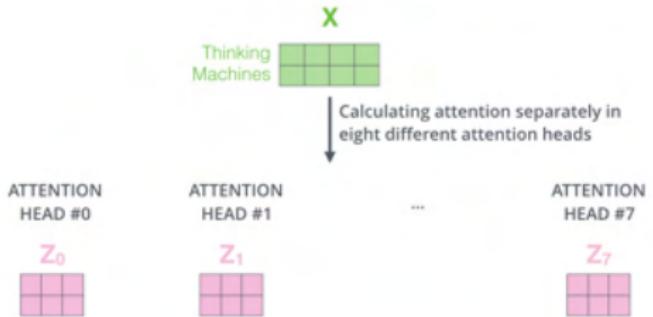
- ① There may be multiple patterns of attention that are relevant at the same time
- ② E.g., some patterns relevant to the 'tense' while others might be associated with the 'vocabulary.'

Multi-headed Self-Attention



Credits: Jay Alammar

Multi-headed Self-Attention



Credits: Jay Alammar

Multi-headed Self-Attention

- ① Expands the model's ability to focus on different relevant positions in the i/p

Multi-headed Self-Attention

- ① Expands the model's ability to focus on different relevant positions in the i/p
- ② Enables different 'representational subspace'

Multi-headed Self-Attention

1) Concatenate all the attention heads



2) Multiply with a weight matrix W^O that was trained jointly with the model

x

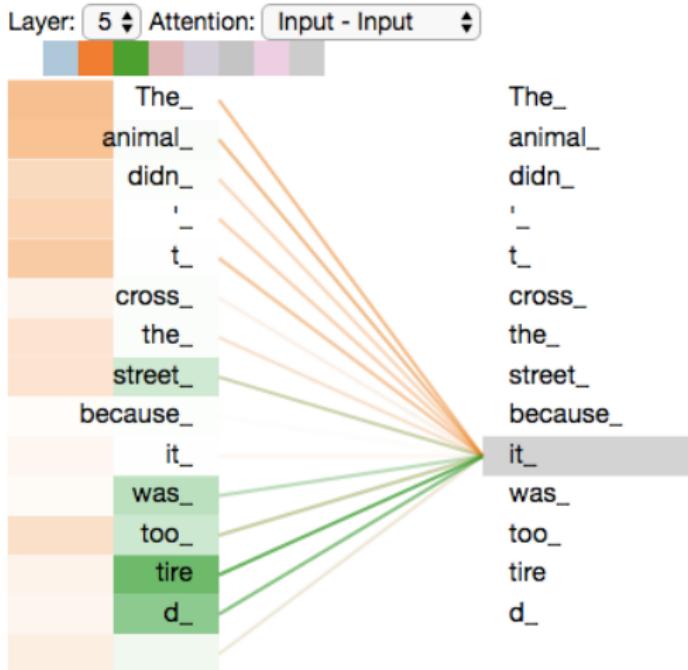


3) The result would be the Z matrix that captures information from all the attention heads. We can send this forward to the FFNN

$$= \begin{matrix} Z \\ \hline \end{matrix}$$

Credits: Jay Alammar

Multi-headed Self-Attention



Credits: Jay Alammar

Transformer Layers

- ① Neural nets benefit greatly from the depth

Transformer Layers

- ① Neural nets benefit greatly from the depth
- ② → stack multiple self-attention layers

Transformer Layers

- ① Neural nets benefit greatly from the depth
- ② → stack multiple self-attention layers
- ③ To improve the training efficiency, introduce residual connections
(requires to maintain the dimensionality)

Transformer Layers

- ① Neural nets benefit greatly from the depth
- ② → stack multiple self-attention layers
- ③ To improve the training efficiency, introduce residual connections
(requires to maintain the dimensionality)
- ④ Followed by Layer normalization

$$\mathbf{Z} = \text{LayerNorm}[\mathbf{Y}(\mathbf{X}) + \mathbf{X}]$$

Transformer Layers

- ① Output vectors are constrained to lie in the subspace spanned by the i/p vectors

Transformer Layers

- ① Output vectors are constrained to lie in the subspace spanned by the i/p vectors
- ② Enhance the expressive capability/flexibility by post-processing using a nonlinear neural net (MLP)

Transformer Layers

- ① Output vectors are constrained to lie in the subspace spanned by the i/p vectors
- ② Enhance the expressive capability/flexibility by post-processing using a nonlinear neural net (MLP)
- ③ This should not affect the transformer's ability to process variable length i/p

Transformer Layers

- ① Output vectors are constrained to lie in the subspace spanned by the i/p vectors
- ② Enhance the expressive capability/flexibility by post-processing using a nonlinear neural net (MLP)
- ③ This should not affect the transformer's ability to process variable length i/p
- ④ Same share net applies to all the o/p tokens (followed by residual connection and normalization)

$$\tilde{\mathbf{X}} = \mathbf{LayerNorm}[\mathbf{MLP}[\mathbf{Z}] + \mathbf{Z}]$$

Positional Encoding

- ① Unlike RNN and CNN encoders, attention encoder outputs don't depend on the order

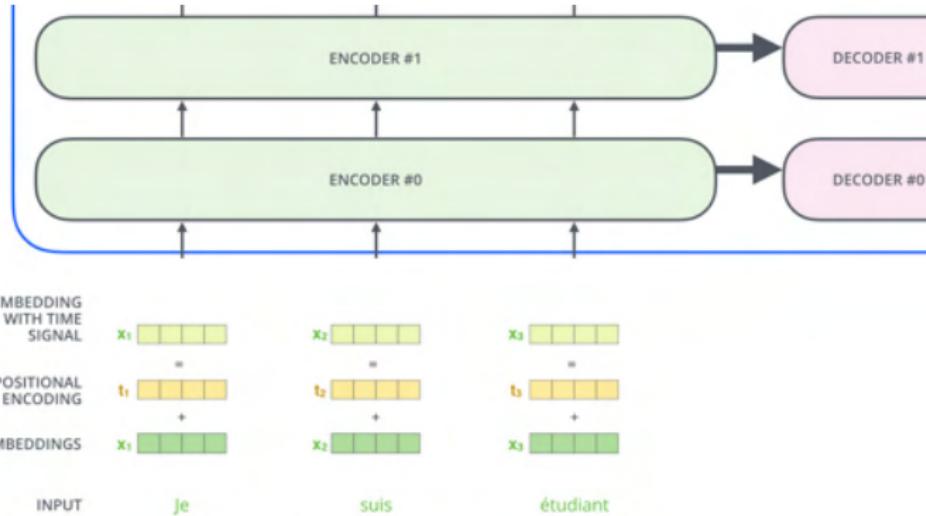
Positional Encoding

- ① Unlike RNN and CNN encoders, attention encoder outputs don't depend on the order
- ② However, order the sequence conveys vital information in some applications

Positional Encoding

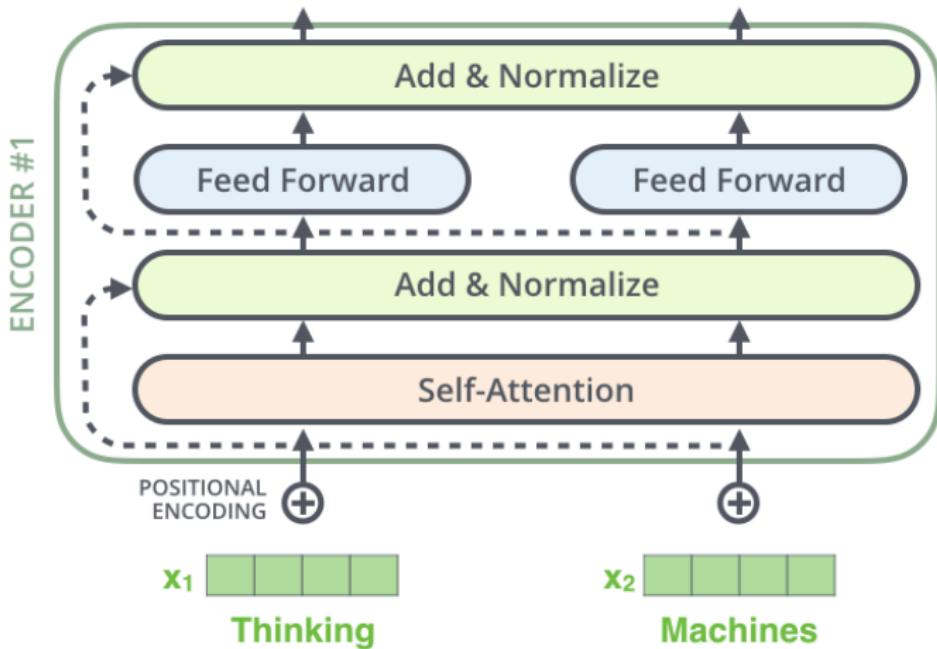
- ① Unlike RNN and CNN encoders, attention encoder outputs don't depend on the order
- ② However, order the sequence conveys vital information in some applications
- ③ Solution: Add positional information of the i/p words into their embedding vectors

Positional Encoding



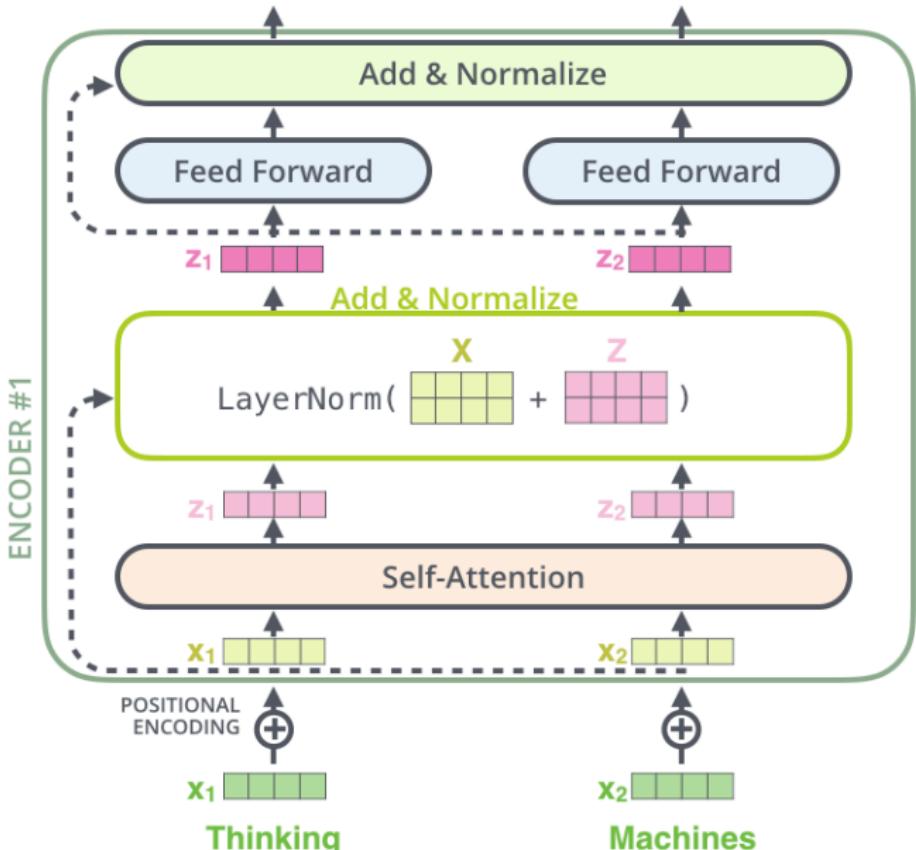
Credits: Jay Alammar

Residuals in the Encoder

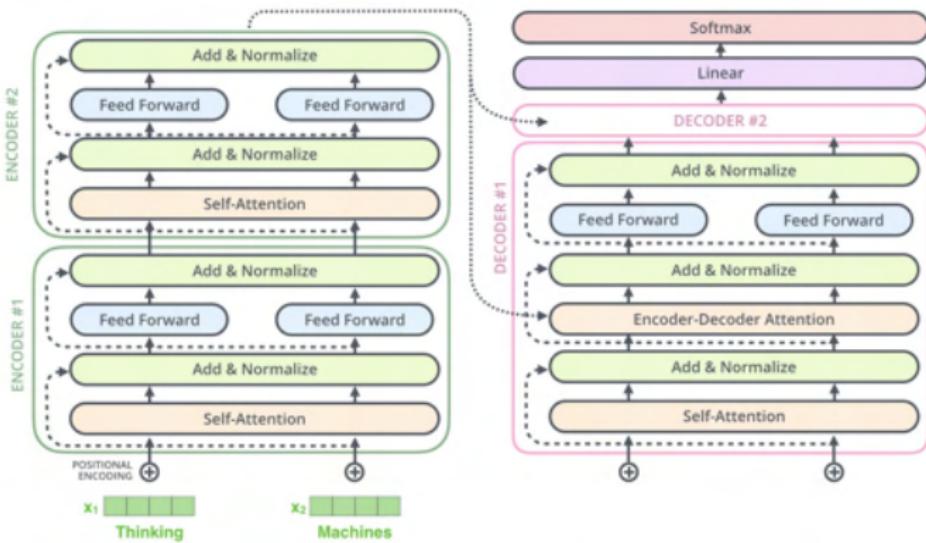


Credits: Jay Alammar

Residuals in the Encoder



Transformer-Decoder



Credits: Jay Alammar

Transformer-Decoder

- ① Self-attention here works in a slightly different way → masks the future positions

Transformer-Decoder

- ① Self-attention here works in a slightly different way → masks the future positions
- ② Uses the top encoder's K and V vectors for its' encoder-decoder attention

Transformer-Decoder

- ① Self-attention here works in a slightly different way → masks the future positions
- ② Uses the top encoder's K and V vectors for its' encoder-decoder attention
- ③ Encoder-decoder attention layer borrows the queries from the layer below it

Transformer-Decoder

Transformer-Decoder

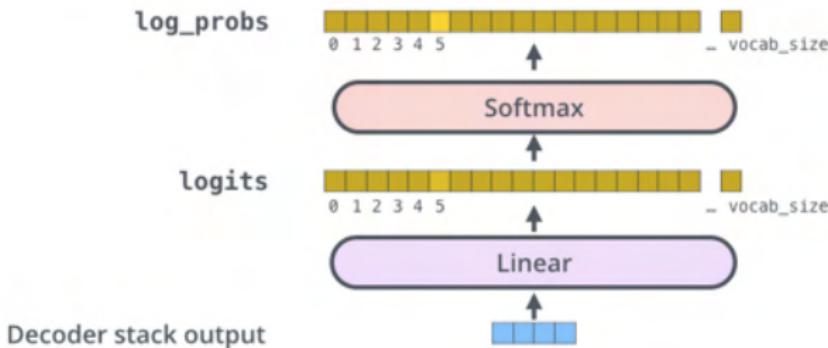
Final o/p

Which word in our vocabulary
is associated with this index?

am

Get the index of the cell
with the highest value
(argmax)

5



Credits: Jay Alammar