

Transformer model

Presenter: Steven Tang

2021.09

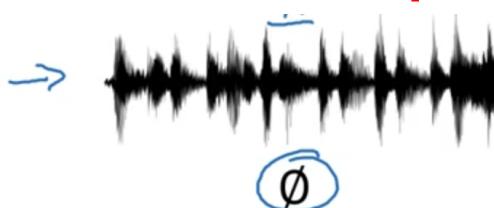
What I will cover

- The motivation behind transformer model
- The architecture of transformer model
- Multi-head self-attention mechanism (What is Q,K,V?)
- Implementation considerations
- Very brief about training

Sequence models use-case

- ML Models that **works on sequence data**

Speech recognition



→ "The quick brown fox jumped
over the lazy dog."

A musical staff with notes represented by vertical stems and horizontal beams. A blue arrow points from the waveform to the staff. A blue circle highlights a note on the staff, with a blue bracket above it indicating a segment.

Music generation

Sentiment classification

"There is nothing to like
in this movie."



DNA sequence analysis → AGCCCCTGTGAGGAAC TAG

→ AGCCCCTGTGAGGAAC **TAG**

Machine translation

Voulez-vous chanter avec
moi?

→ Do you want to sing with
me?

Video activity recognition



→ Running

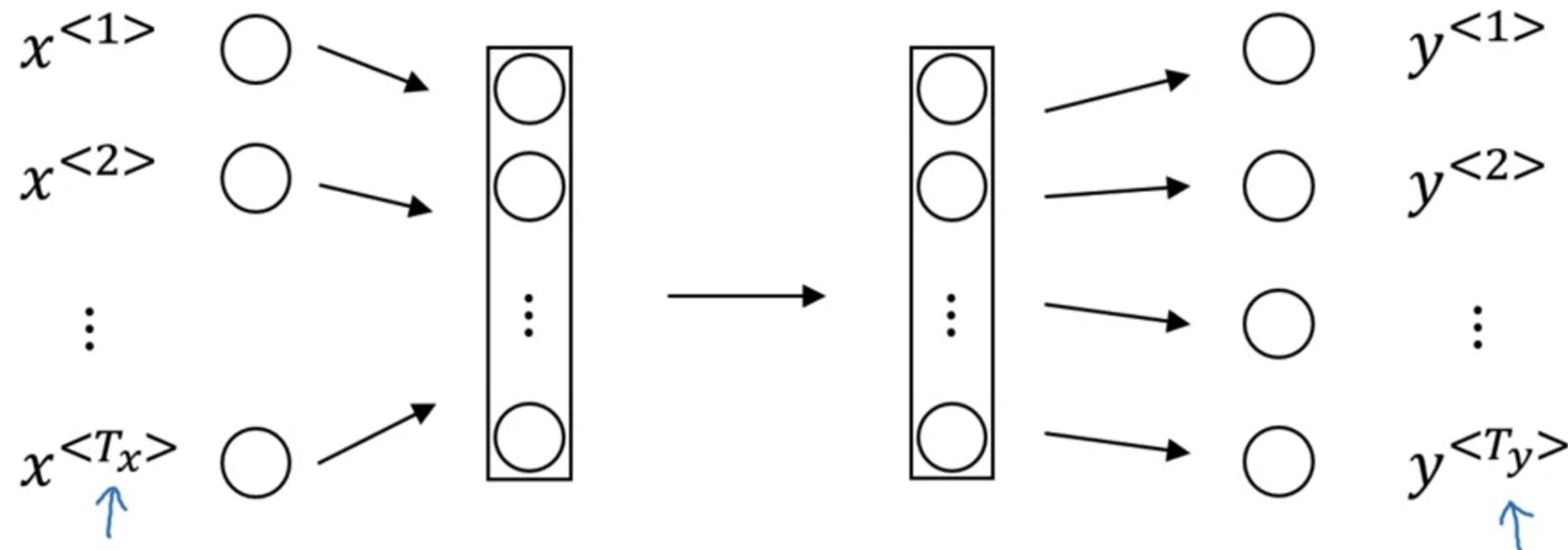
Name entity recognition

Yesterday, Harry Potter
met Hermione Granger.

→ Yesterday, **Harry Potter**
met **Hermione Granger**.

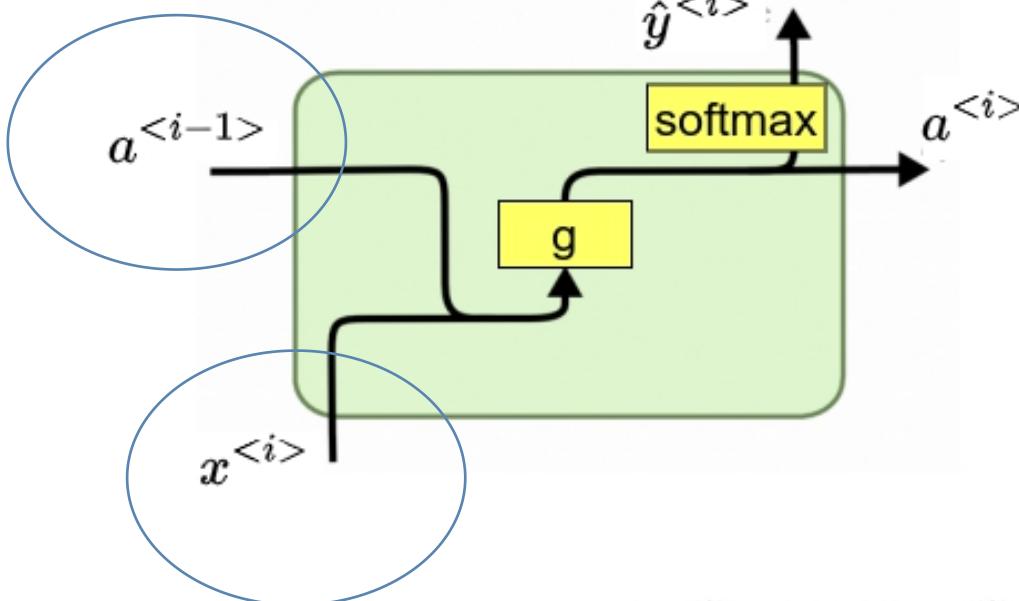
Why not a standard network?

- 1. Variable length inputs
- 2. No learned feature sharing

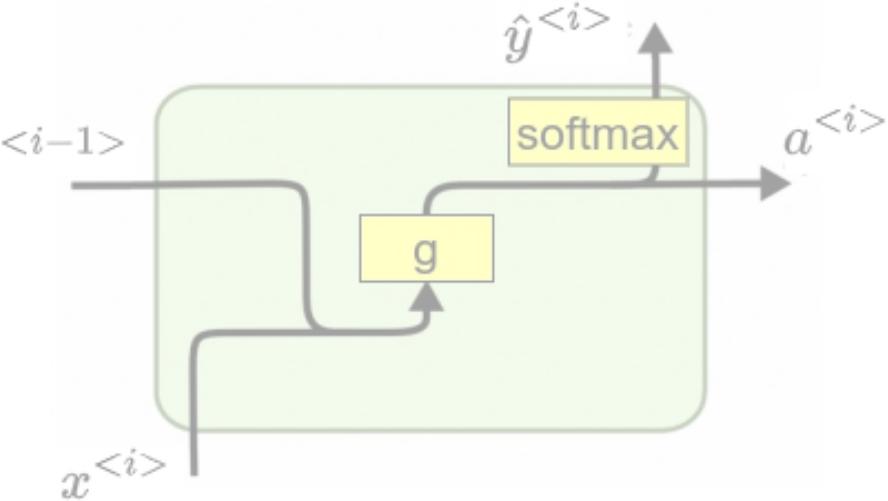


RNN cells

RNN



GRU

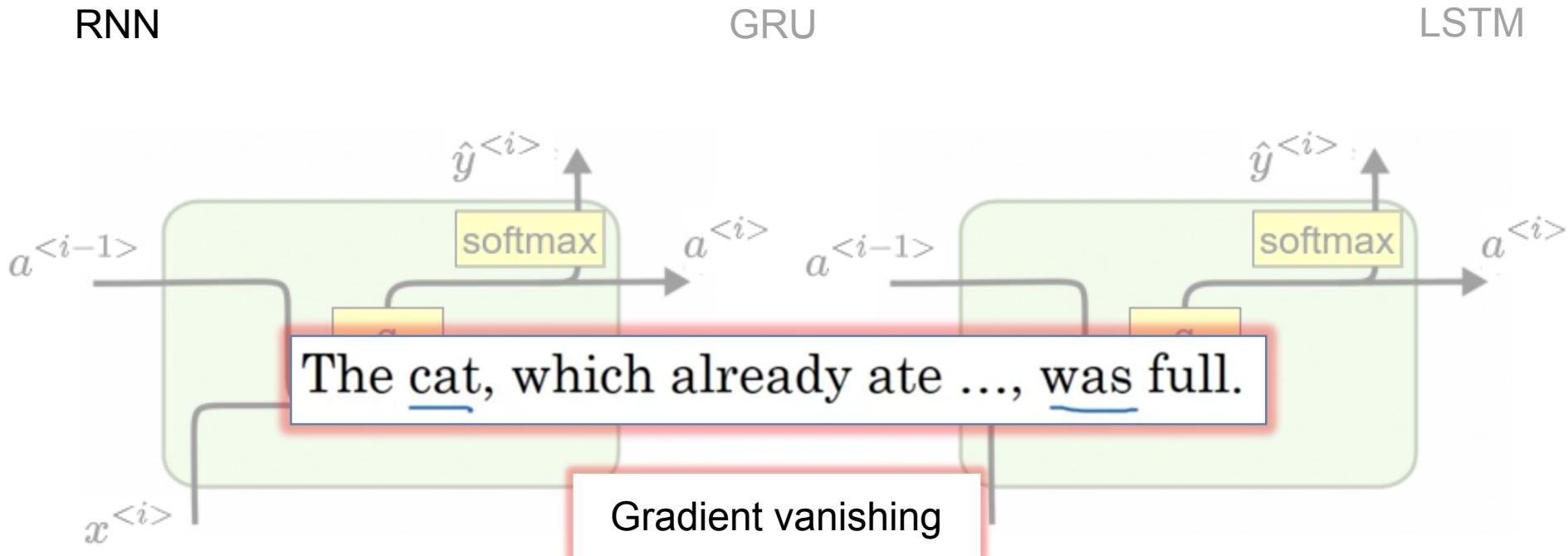


LSTM

$$a^{<t>} = g(W_{aa}a^{<t>} + W_{ax}x^{<t>} + b_a)$$

$$\hat{y}^{<t>} = g(W_{ya}a^{<t>} + b_y)$$

RNN cells

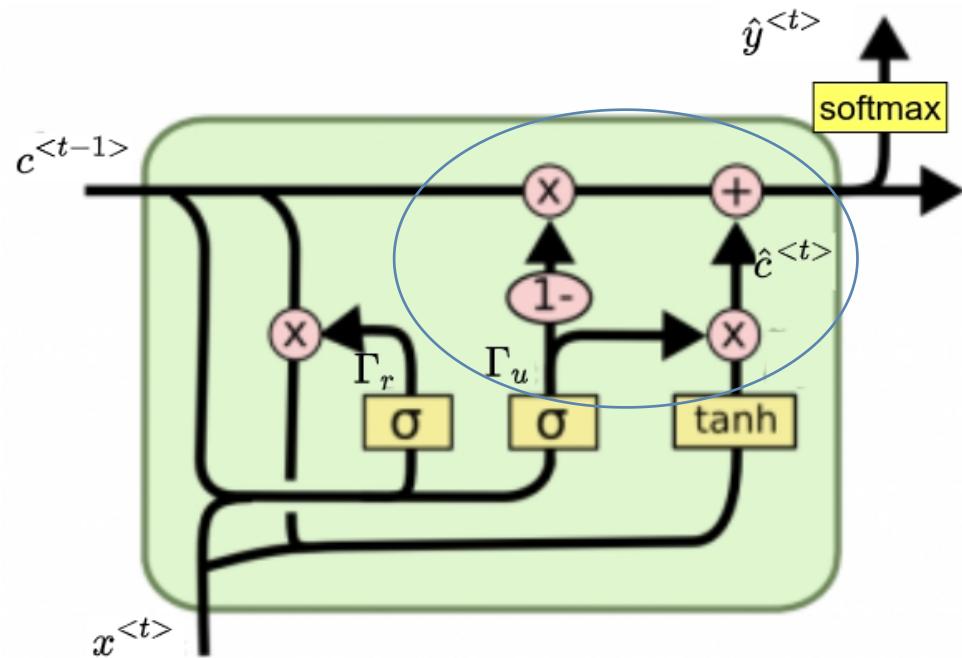


$$a^{<t>} = g(W_{aa}a^{<t>} + W_{ax}x^{<t>} + b_a)$$

$$\hat{y}^{<t>} = g(W_{ya}a^{<t>} + b_y)$$

RNN cells

RNN



GRU

$$\tilde{c}^{<t>} = \tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_u = \sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$$

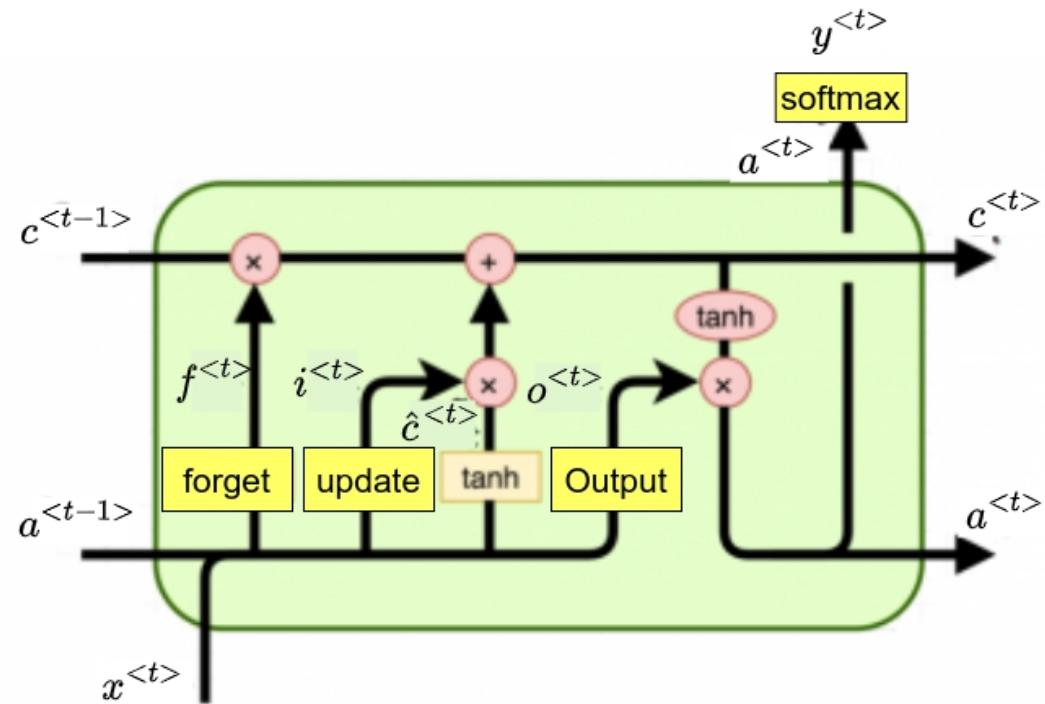
$$\Gamma_r = \sigma(W_r[c^{<t-1>}, x^{<t>}] + b_r)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) + c^{<t-1>}$$

LSTM

RNN cells

RNN



GRU

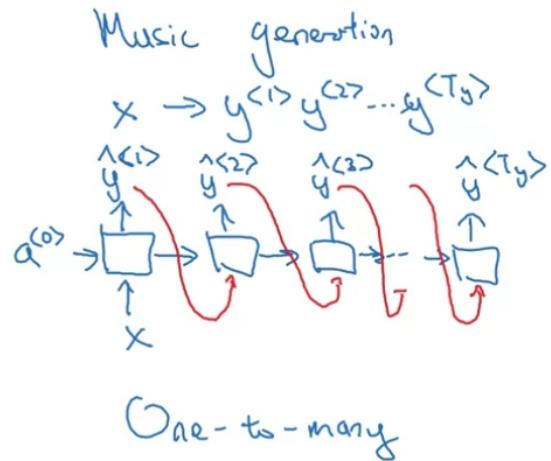
$$\begin{aligned}\tilde{c}^{<t>} &= \tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c) \\ \Gamma_u &= \sigma(W_u[a^{<t-1>}, x^{<t>}] + b_u) \\ \Gamma_f &= \sigma(W_f[a^{<t-1>}, x^{<t>}] + b_f) \\ \Gamma_o &= \sigma(W_o[a^{<t-1>}, x^{<t>}] + b_o) \\ c^{<t>} &= \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>} \\ a^{<t>} &= \Gamma_o * \tanh(c^{<t>})\end{aligned}$$

LSTM

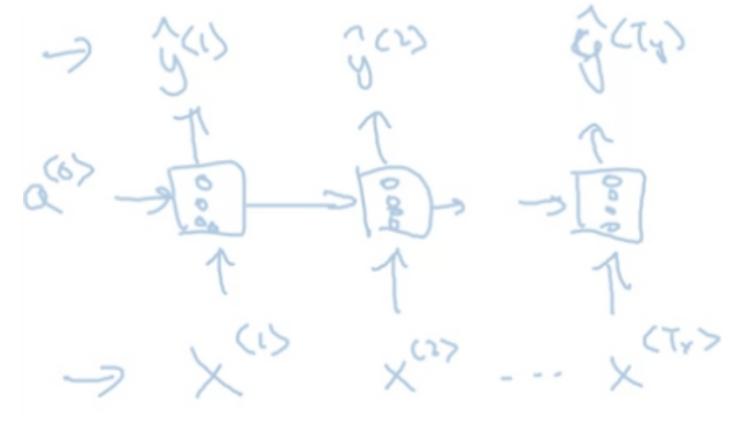
The cat, which already ate ..., was full.

Different types of RNNs

- One-to-many



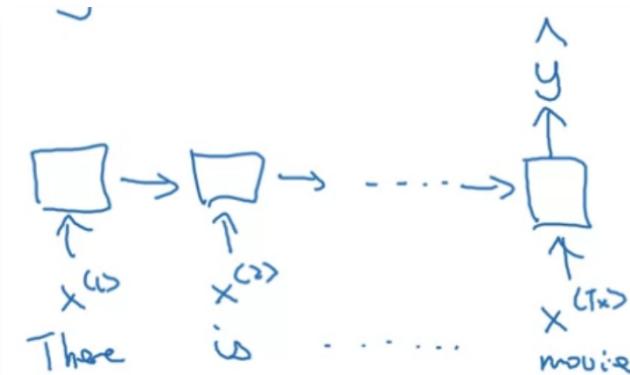
- Many-to-many



- One-to-one

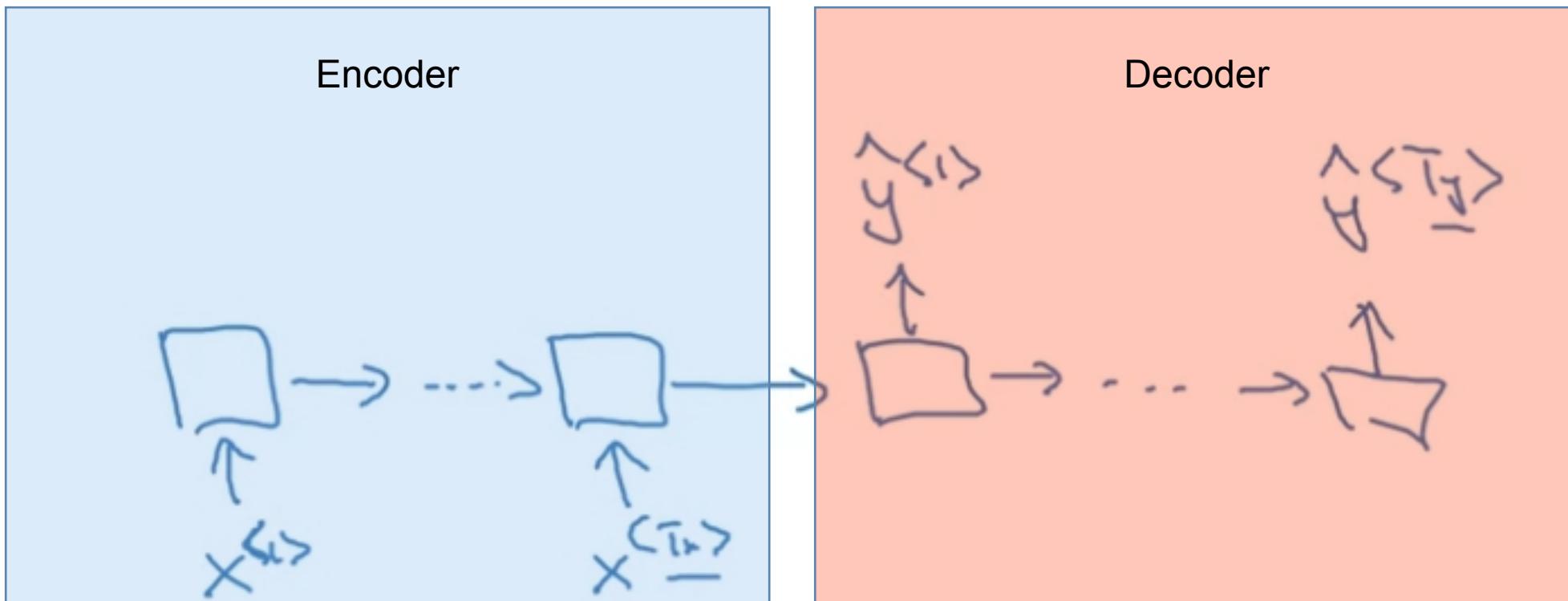


- Many-to-one



Encoder-Decoder architecture

- Many-to-many



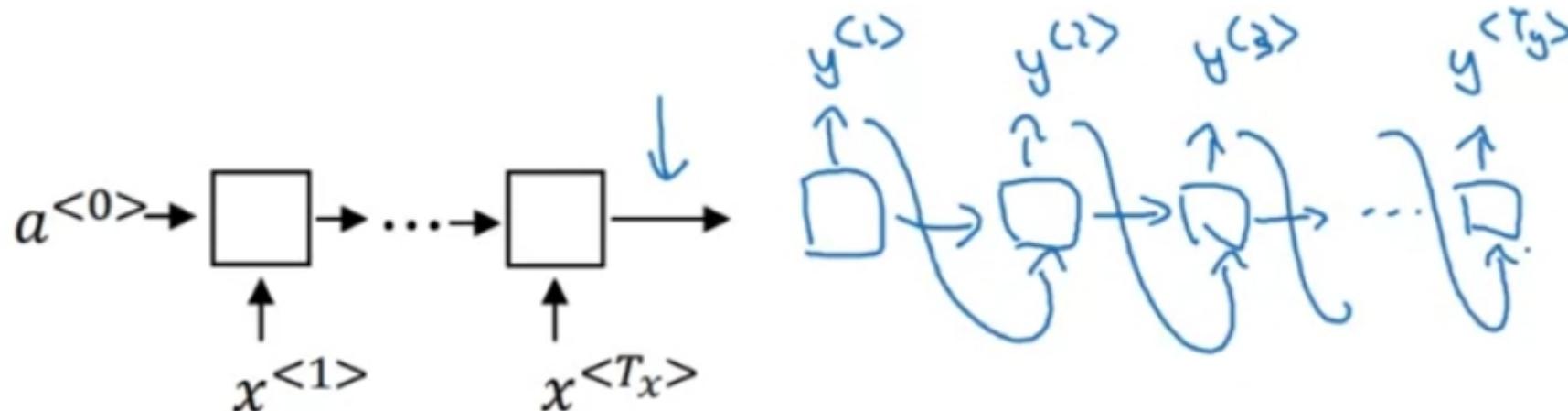
Encoder-Decoder architecture

$x^{<1>} \quad x^{<2>} \quad x^{<3>} \quad x^{<4>} \quad x^{<5>}$
Jane visite L'Afrique en Septembre
-->

Jane is visiting Africa in September

$y^{<1>} \quad y^{<2>} \quad y^{<3>} \quad y^{<4>} \quad y^{<5>} \quad y^{<6>}$

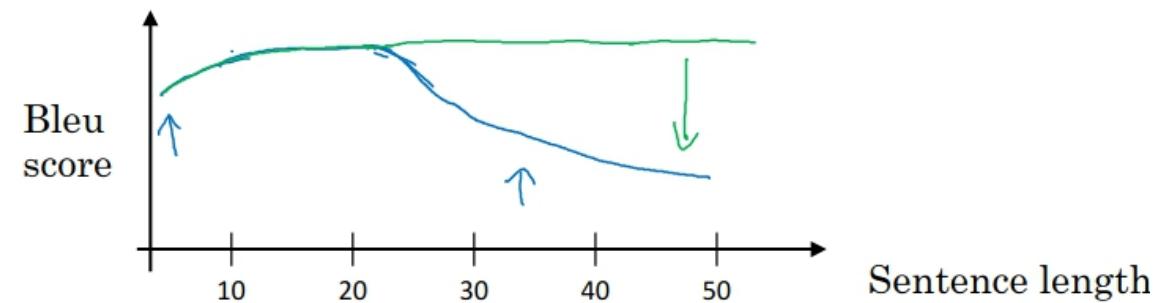
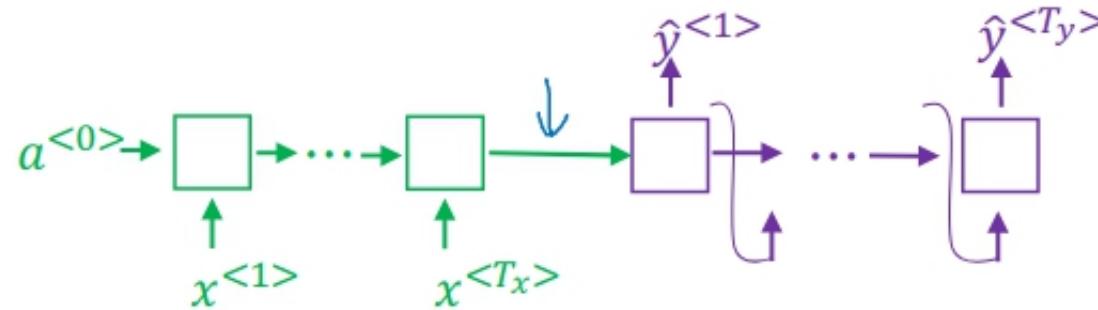
Seq2Seq Problem



Common solution

It is not perfect

- Jane s'est rendue en Afrique en septembre dernier, a apprécié la culture et a rencontré beaucoup degens merveilleux; elle est revenue en parlant comment son voyage était merveilleux, et elle me tented'y aller aussi.
- Jane went to Africa last September, and enjoyed the culture and met many wonderful people; she came back raving about how wonderful her trip was, and is tempting me to go too.

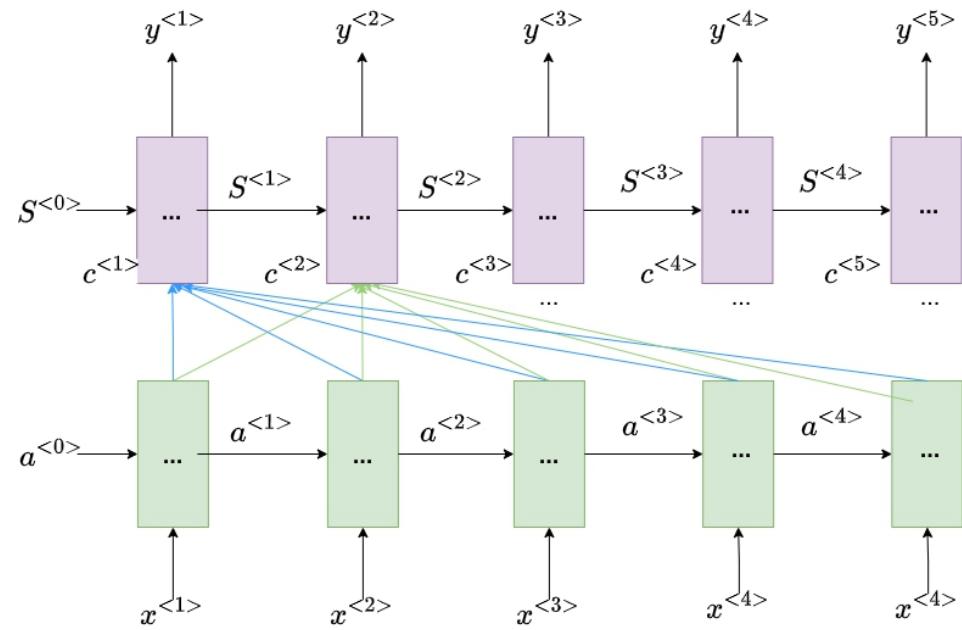


How does human solve the problem?

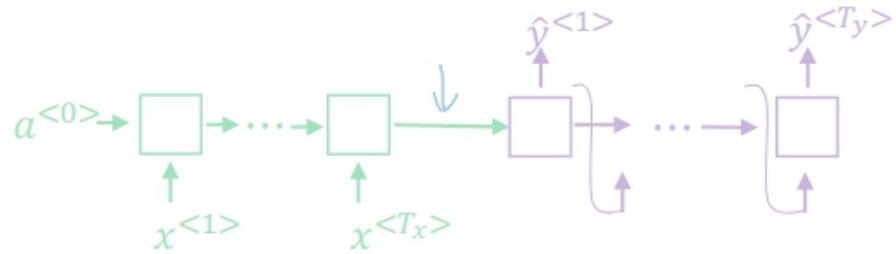
- Jane s'est rendue en Afrique en septembre dernier, a apprécié la culture et a rencontré beaucoup degens merveilleux; elle est revenue en parlant comment son voyage était merveilleux, et elle me tented'y aller aussi.
- Jane went to Africa last September, and enjoyed the culture and met many wonderful people; she came back raving about how wonderful her trip was, and is tempting me to go too.

“Attention” saves brain power

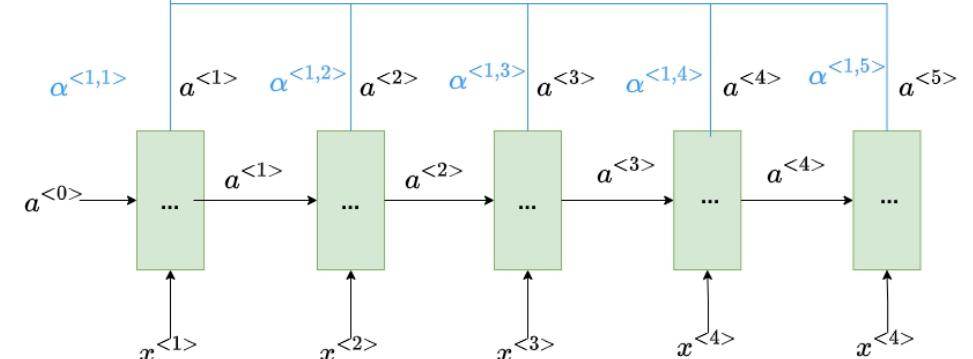
How to represent attention in Encoder-Decoder



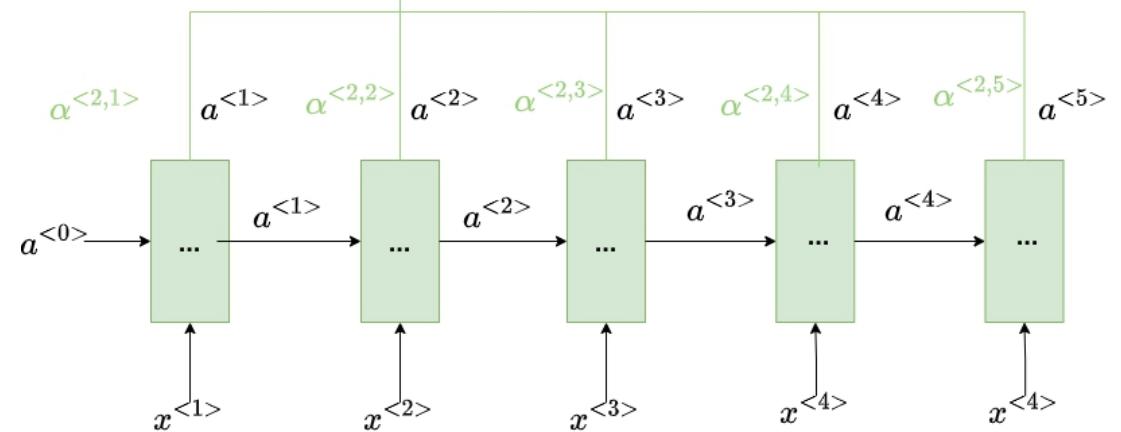
Jane visite L'Afrique en Septembre



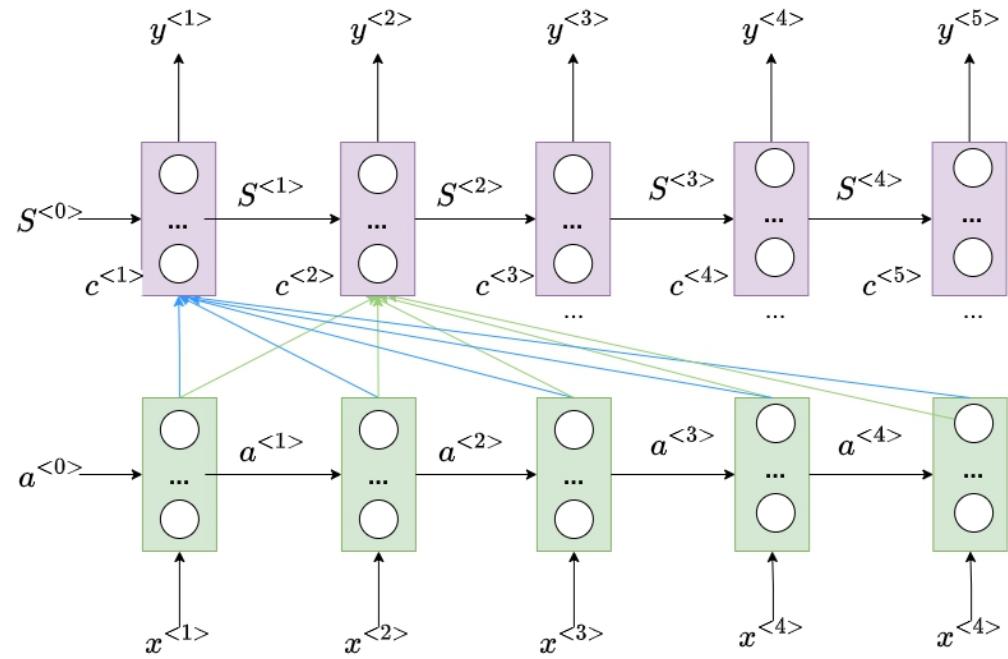
$$c^{<1>} = \alpha^{<1,1>} * a^{<1>} + \dots + \alpha^{<1,5>} * a^{<5>}$$



$$c^{<2>} = \alpha^{<2,1>} * a^{<1>} + \dots + \alpha^{<2,5>} * a^{<5>}$$

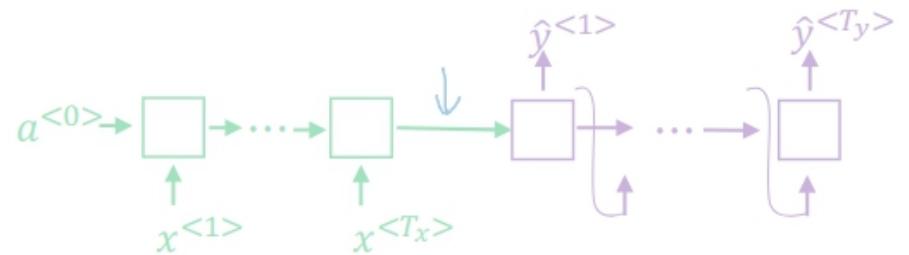


How to represent attention in Encoder-Decoder

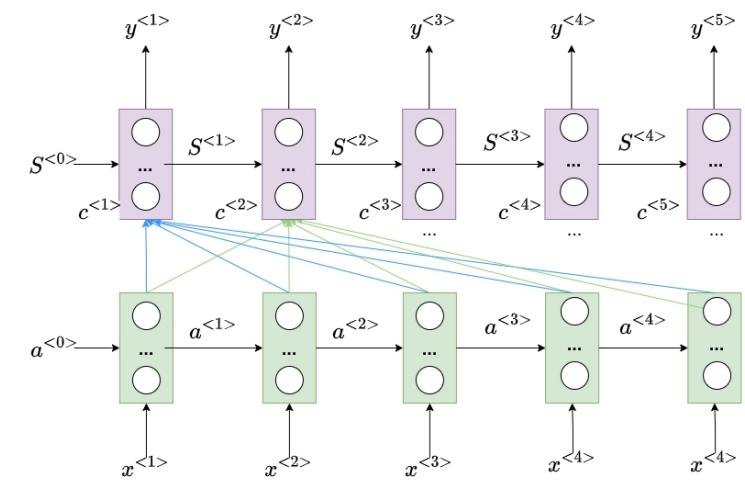


Jane visite L'Afrique en Septembre

$$c^{<t>} = \sum_{t'} \alpha^{<1,t'>} a^{<t'>}$$



How to calculate attention?



①

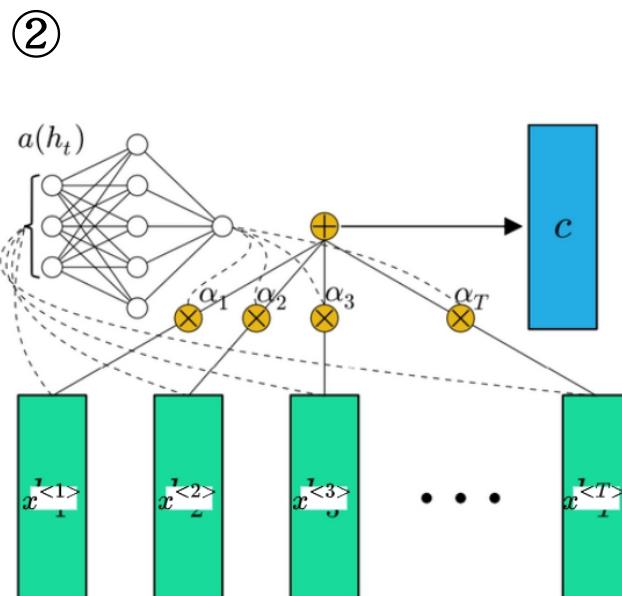
$$\alpha^{<t,t'}> = \frac{\exp(e^{<t,t'}>)}{\sum_{t'=1}^{T_x} \exp(e^{<t,t'}>)}$$

$s^{<t-1>}$

$a^{<t'>}$

$e^{<t,t'>}$

$\alpha^{<t,t'>}$



$$c^{<t>} = \sum_{t'} \alpha^{<1,t'}> a^{<1,t'>}$$

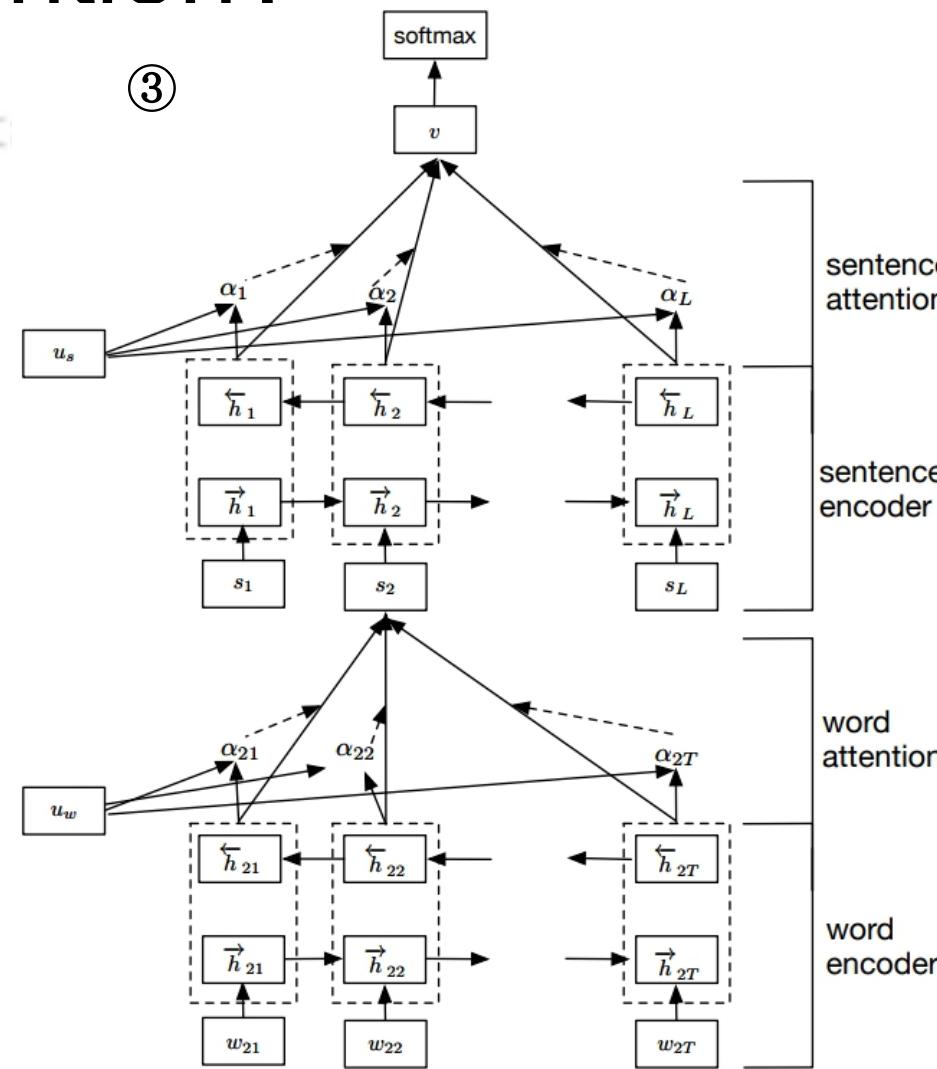
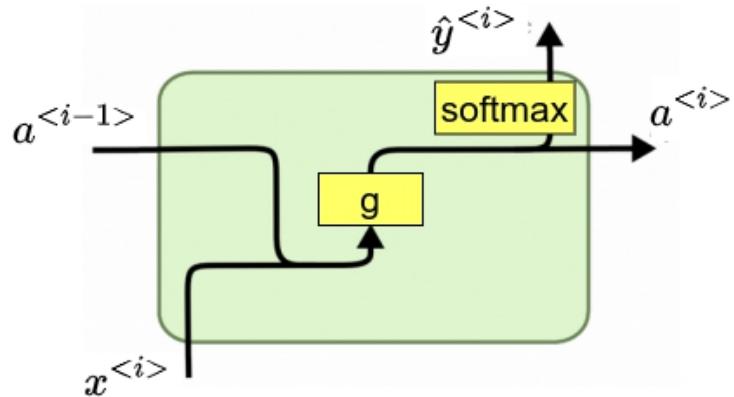


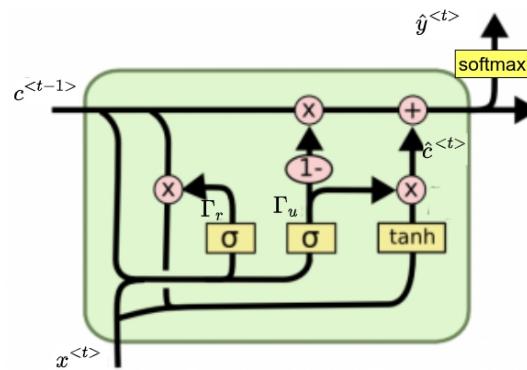
Figure 2: Hierarchical Attention Network.

Limitation?

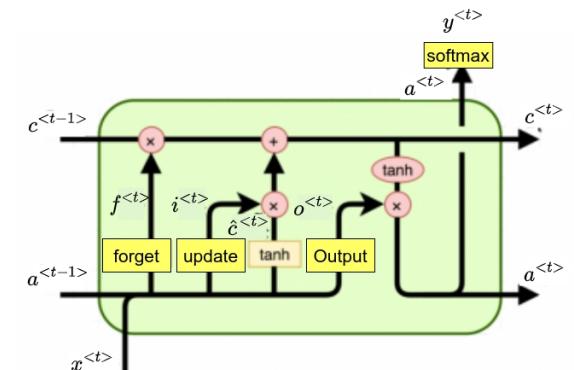
RNN



GRU



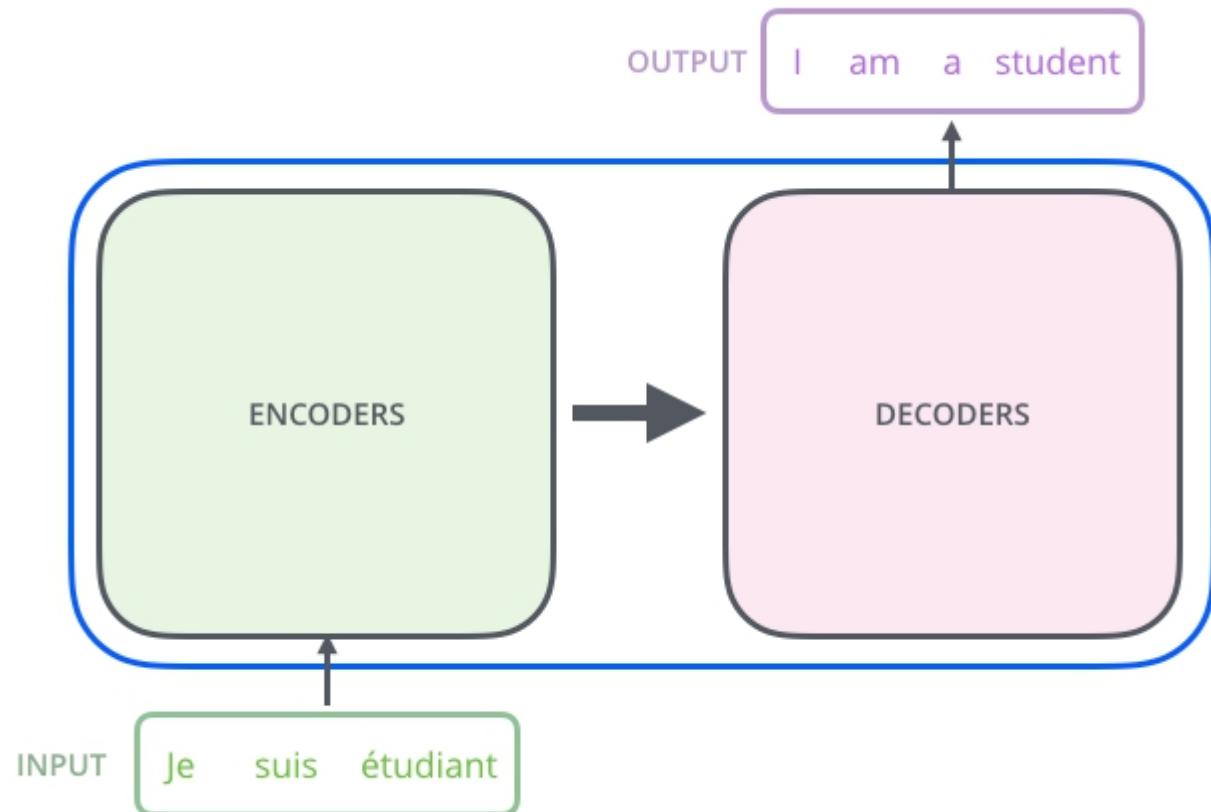
LSTM

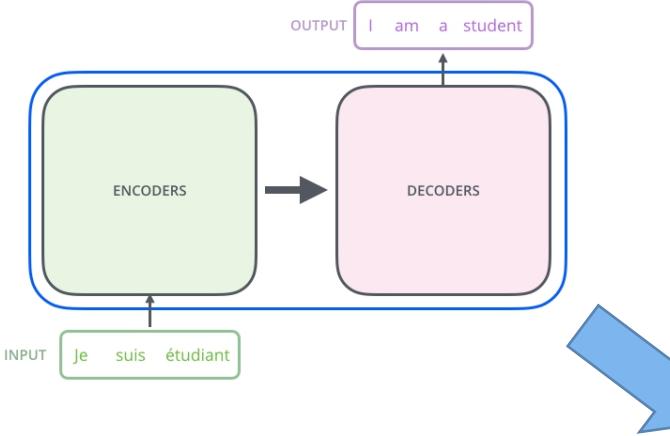


More and more complex

+
Sequential

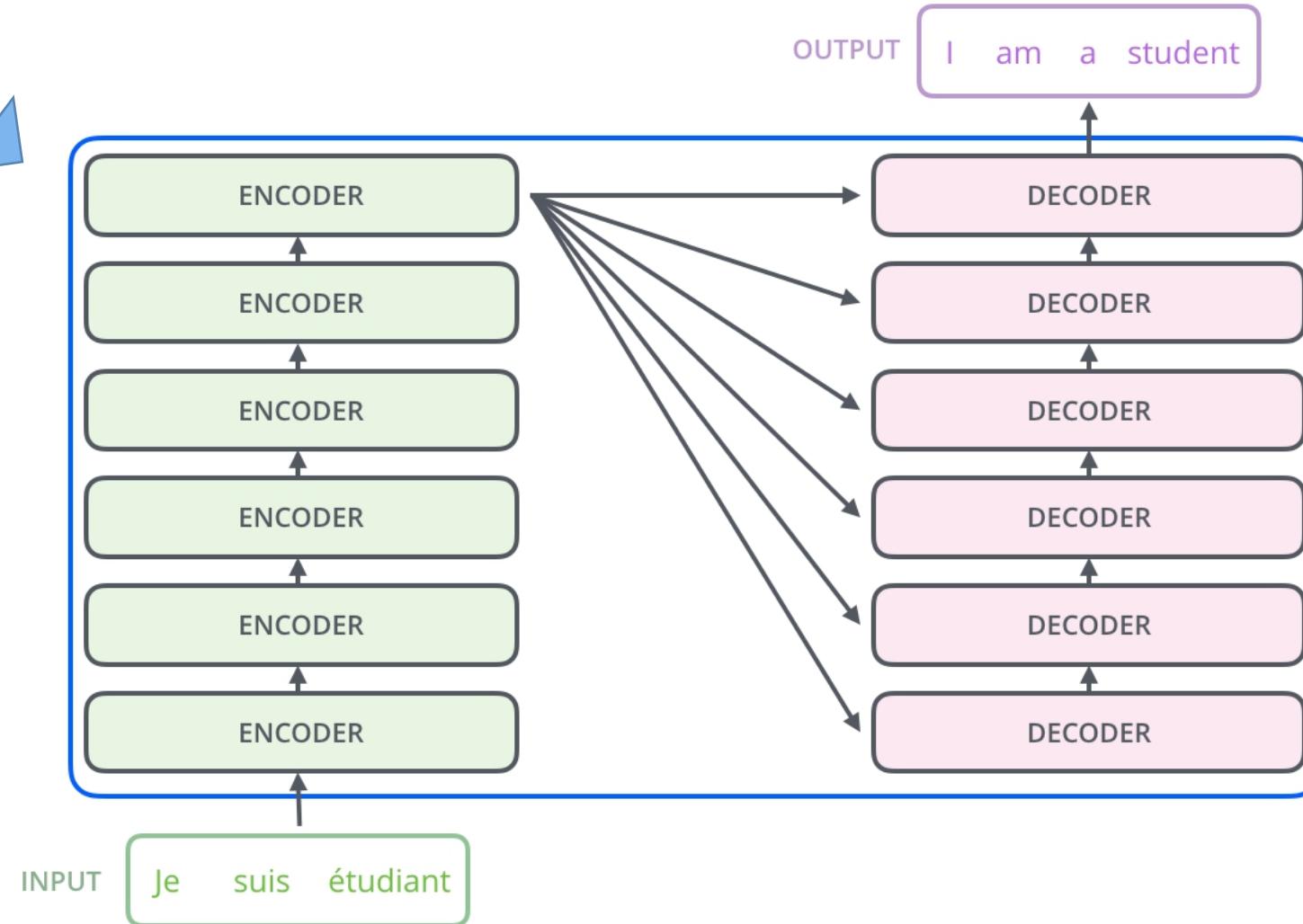
Overall architecture

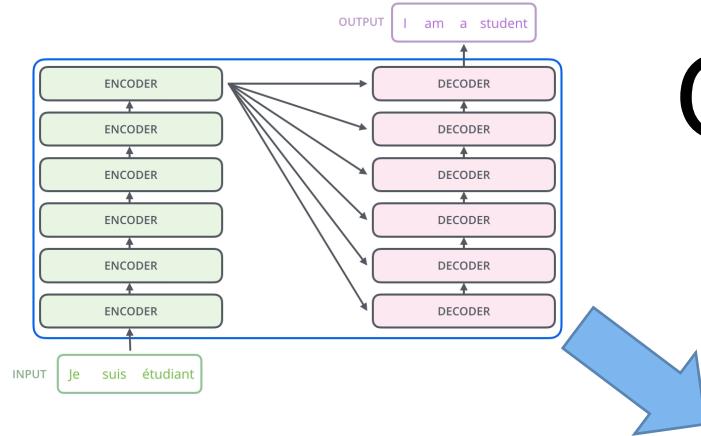




Encoder:
Same structure,
no weight sharing

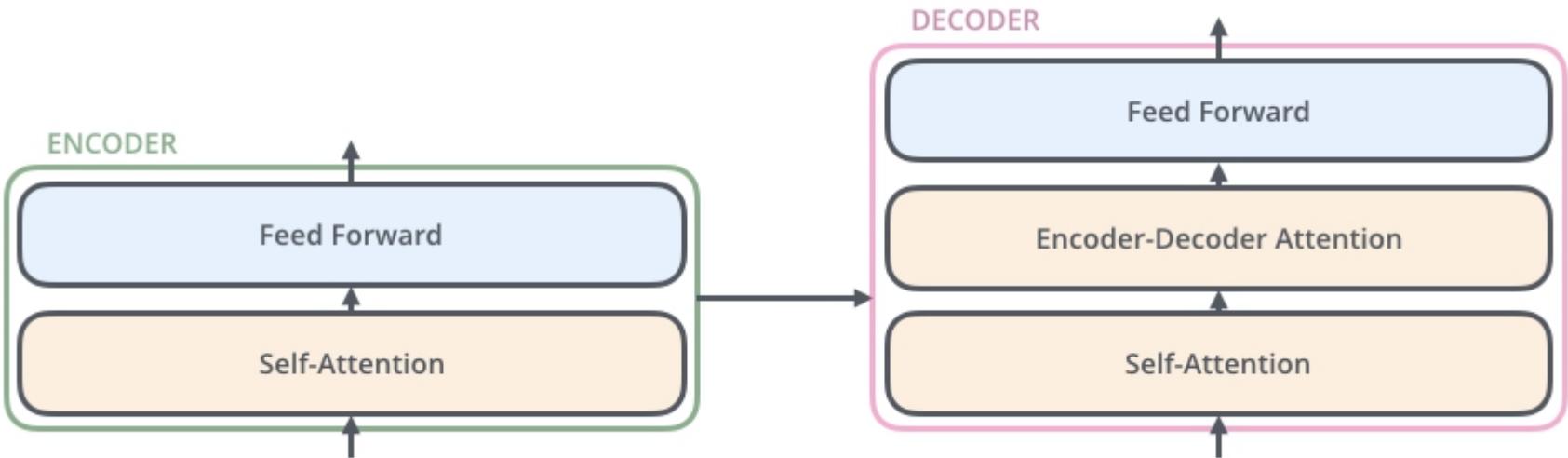
Overall architecture





Self-Attention: Focus on other words in the input sentence

Encoder-Decoder Attention: Focus on relevant parts of the input sentence



Overall architecture

Bringing tensors into the picture

- Word embedding

$$V = [a, \text{aaron}, \dots, \text{zulu}, \text{<UNK>}]$$

$$|V| = 10,000$$

1-hot representation

Man (5391)	Woman (9853)	King (4914)	Queen (7157)	Apple (456)	Orange (6257)
$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}$
\hookrightarrow	\hookrightarrow				
O_{5391}	O_{9853}				

I want a glass of orange juice.
I want a glass of apple ?.

Bringing tensors into the picture

- Word embedding

$$V = [a, \text{aaron}, \dots, \text{zulu}, \text{<UNK>}]$$

$$|V| = 10,000$$

1-hot representation

Man (5391)	Woman (9853)	King (4914)	Queen (7157)	Apple (456)	Orange (6257)
$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}$
\hookrightarrow	\hookrightarrow				
O_{5391}	O_{9853}				

I want a glass of orange juice.
I want a glass of apple ?.

Bringing tensors into the picture

	Man (5391)	Woman (9853)	King (4914)	Queen (7157)	Apple (456)	Orange (6257)
Gender	-1	1	-0.95	0.97	0.00	0.01
300 Royal	0.01	0.62	0.93	0.95	-0.01	0.00
Age	0.03	0.02	0.7	0.69	0.03	-0.02
Food	0.04	0.01	0.02	0.01	0.95	0.97
size	⋮	⋮				
cost						
alt						
verb						

↑ Gender

300 Royal

Age

Food

⋮

size

cost

alt

verb

e_{5391}

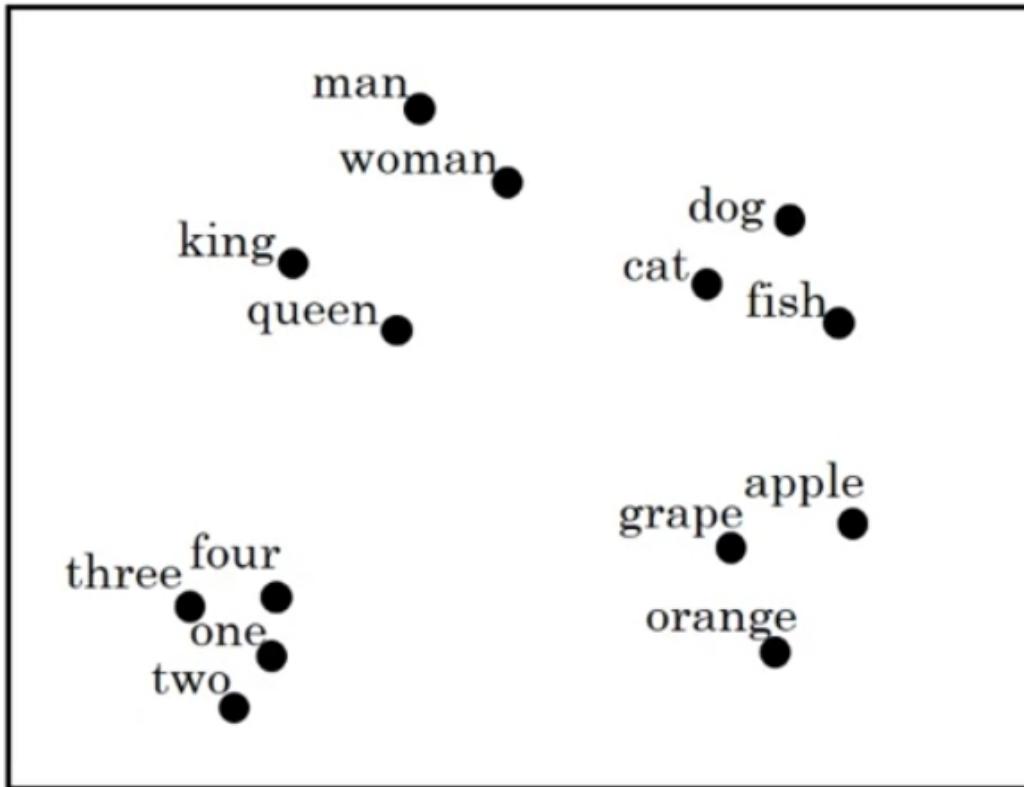
e_{9853}

I want a glass of orange juice.

I want a glass of apple juice.

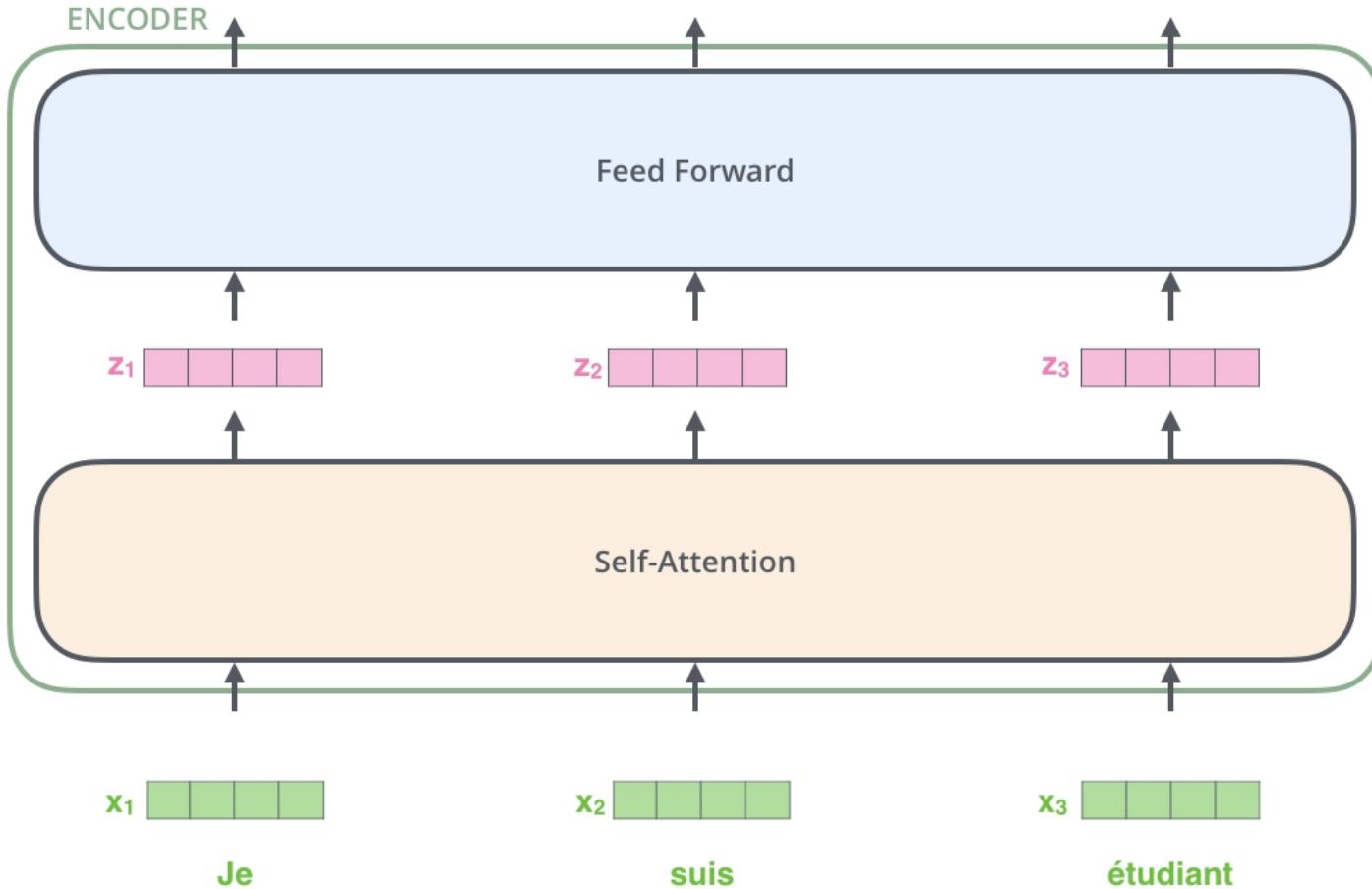
Andrew Ng

Bringing tensors into the picture

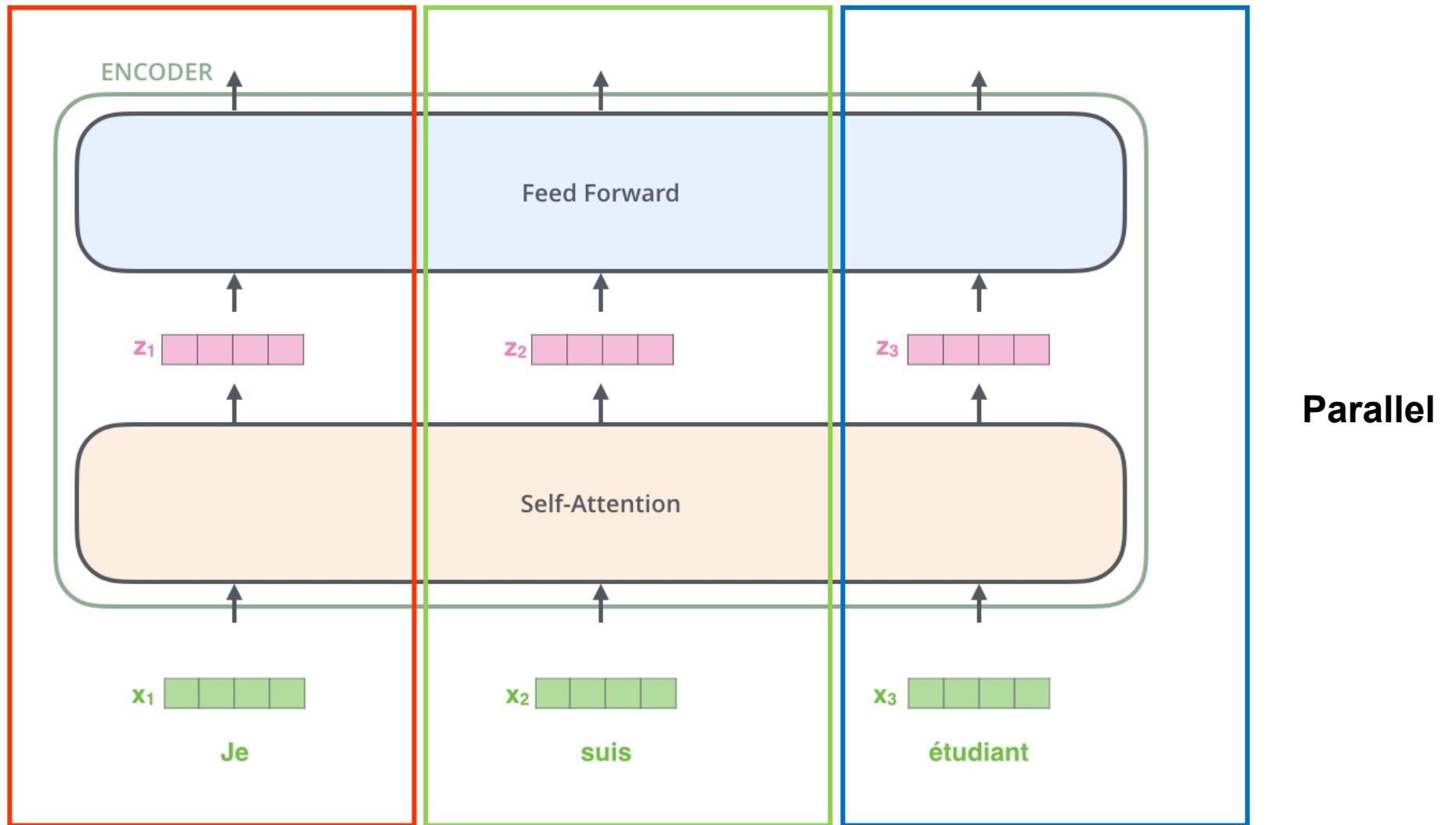


$t\text{-SNE}$

Bringing tensors into the picture

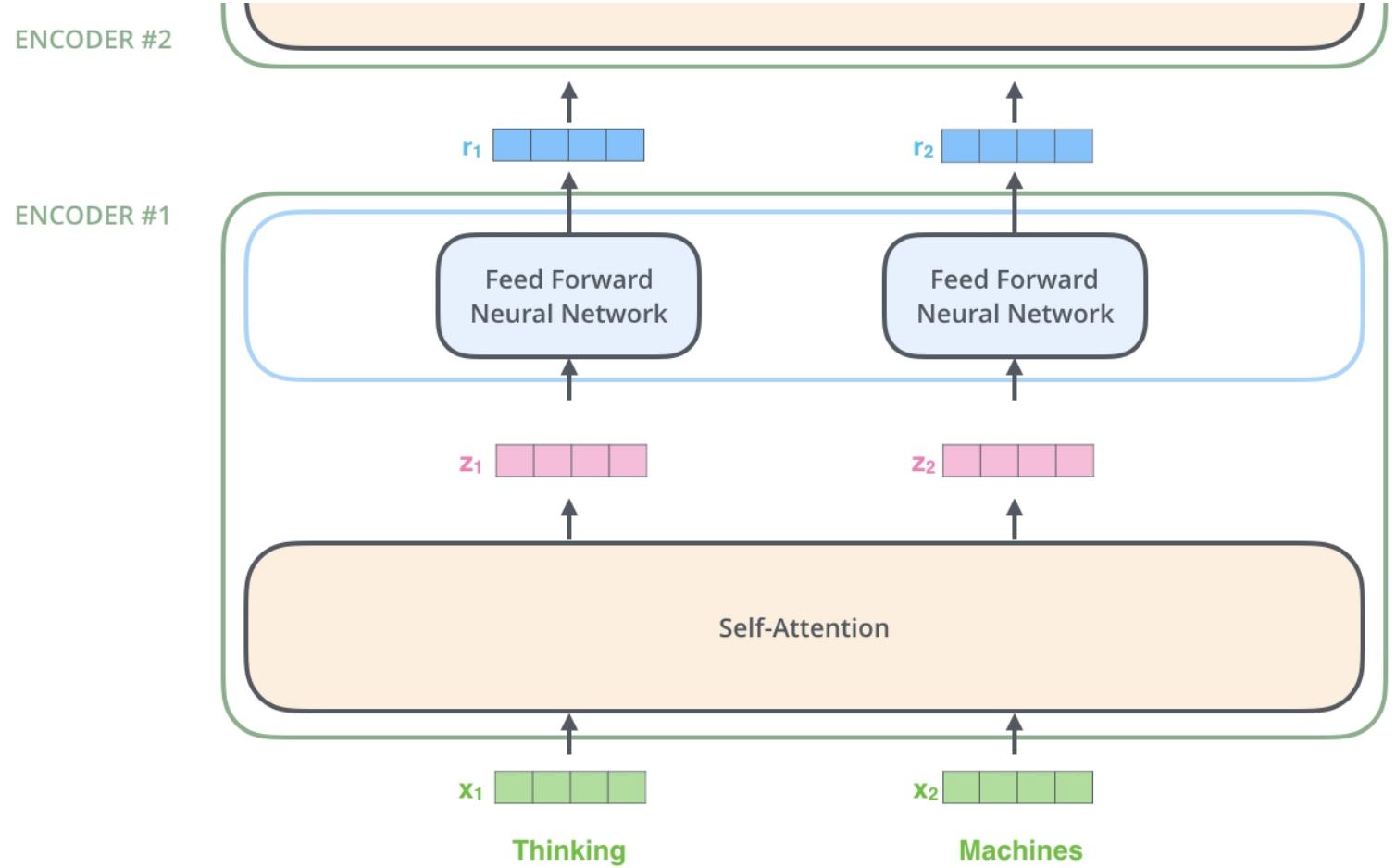
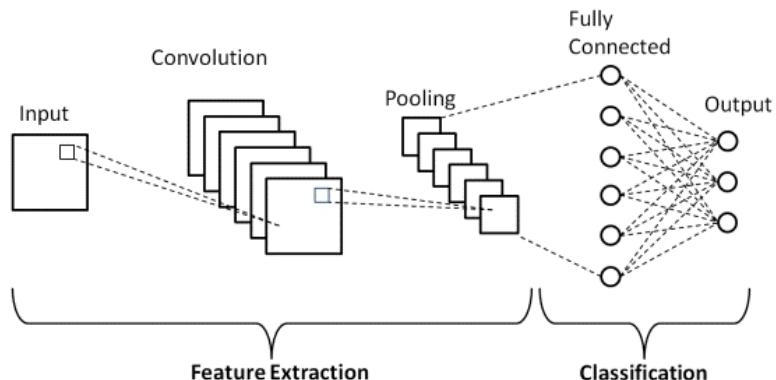


Bringing tensors into the picture

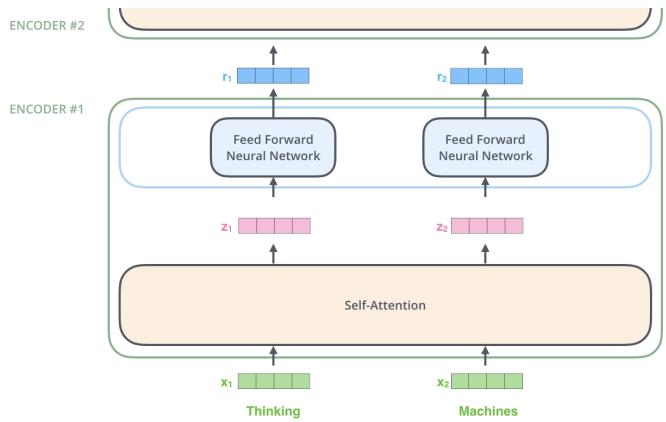


Let's encoding!

Q: Why feed forward NN?

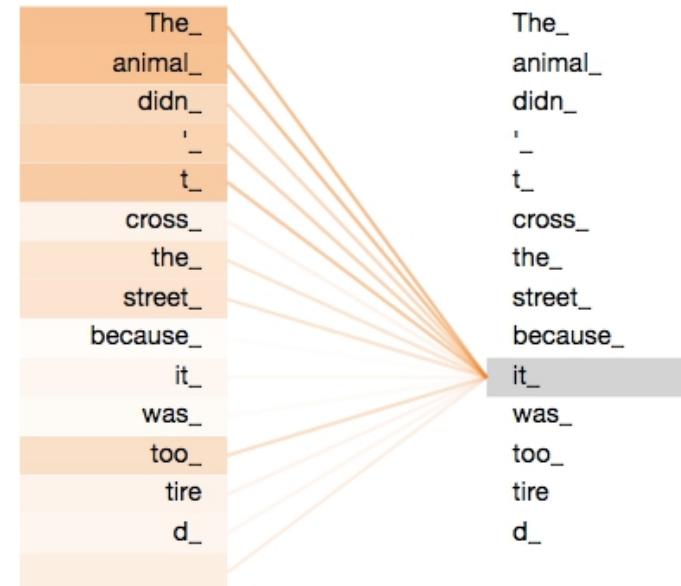


Wait.. What is self-attention?



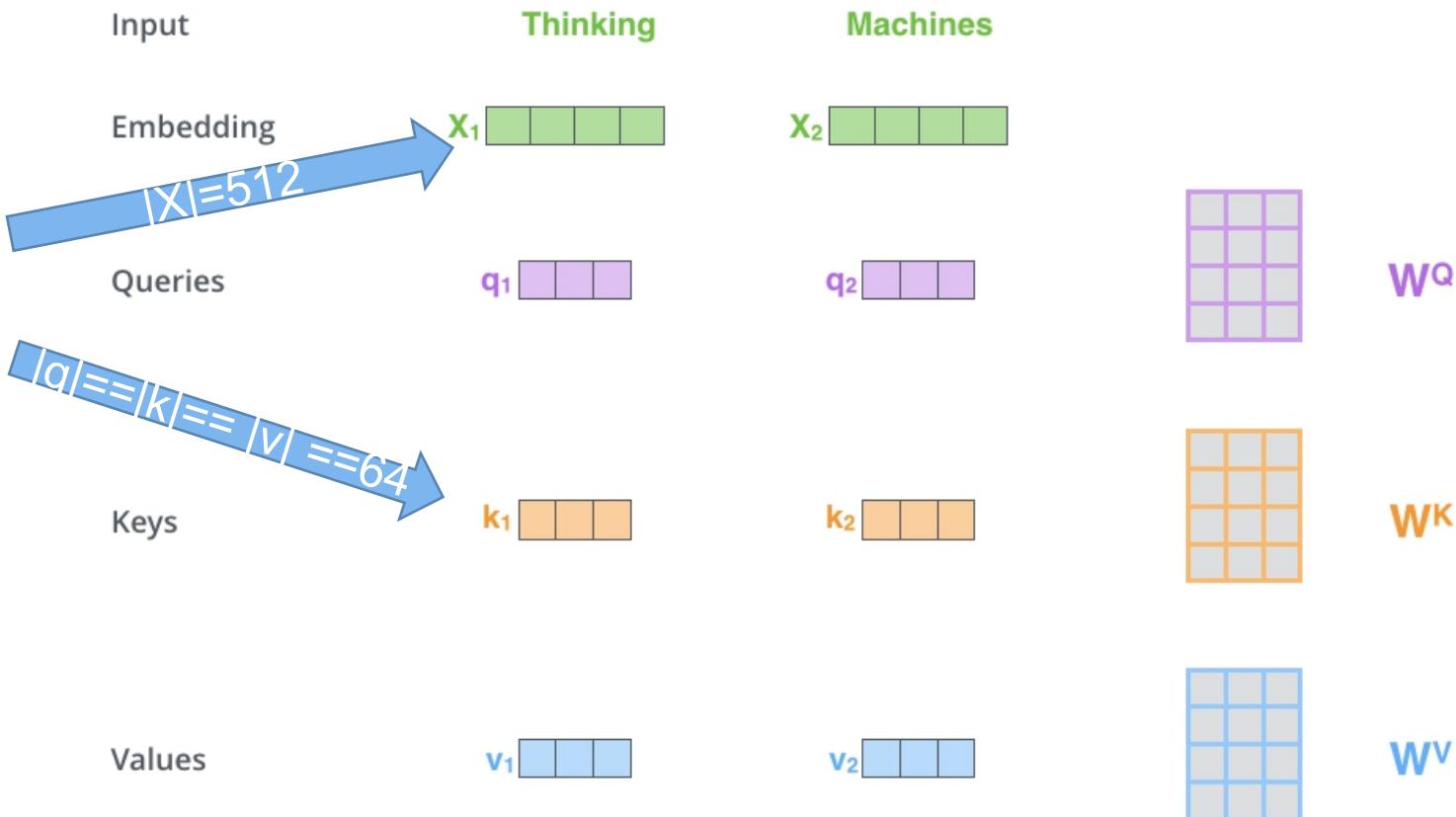
"The animal didn't cross the street because **it** was too tired"

Animal? Street?



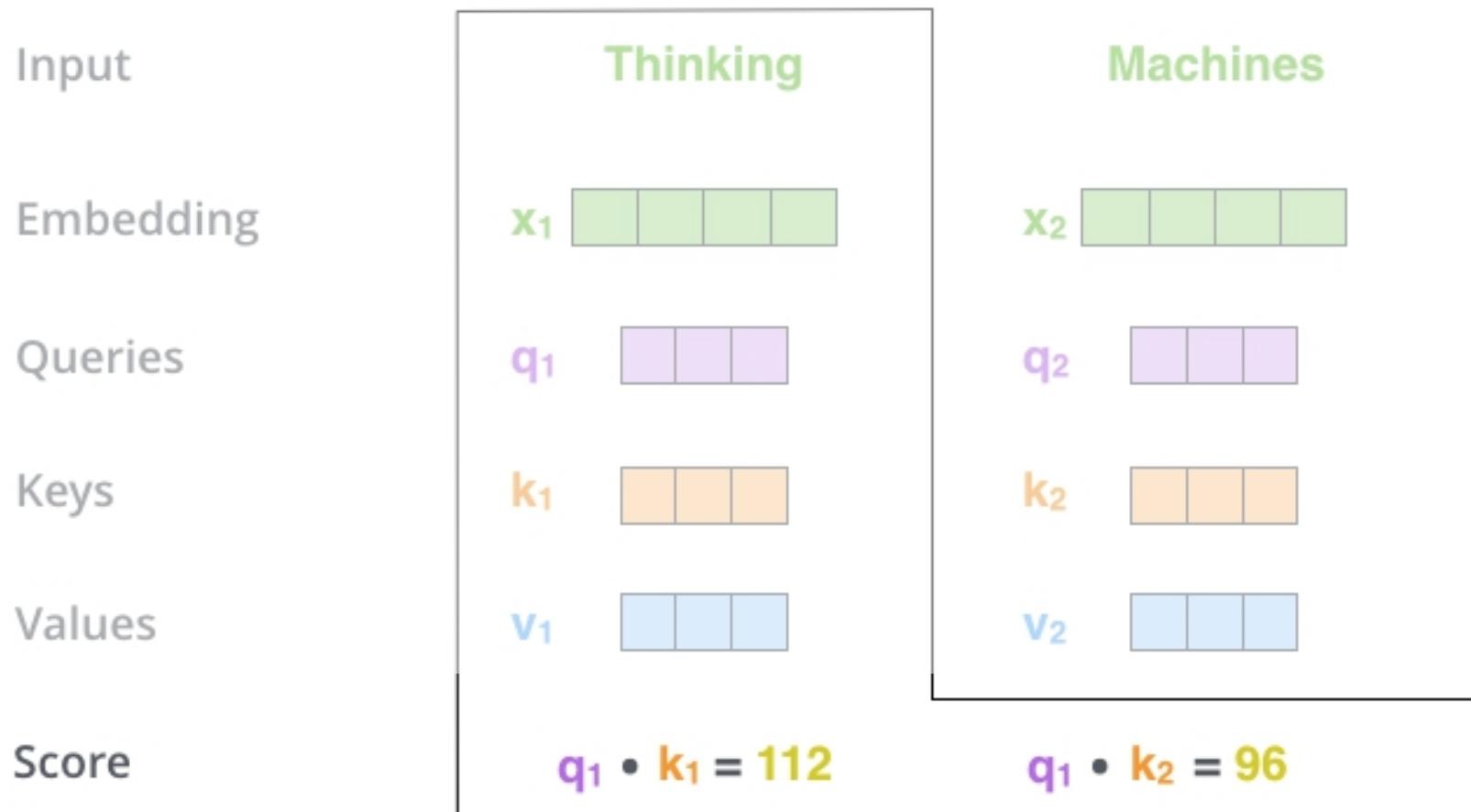
Okay.. How to calculate self-attention?

Intuition: make the computation of multiheaded attention (mostly) constant.



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.

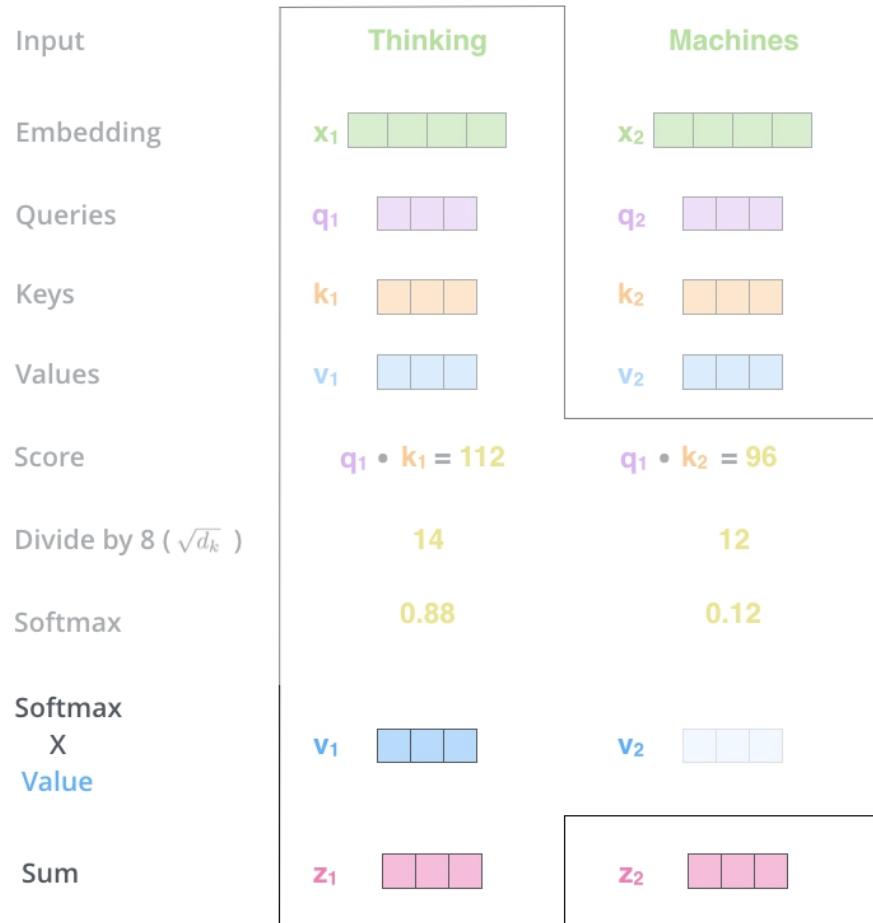
Okay.. How to calculate self-attention?



Okay.. How to calculate 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	

Okay.. How to calculate self-attention?



Okay.. How to calculate self-attention?

Input	Thinking		Machines	
Embedding	x_1	[green]	x_2	[green]
Queries	q_1	[purple]	q_2	[purple]
Keys	k_1	[orange]	k_2	[orange]
Values	v_1	[blue]	v_2	[blue]
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	
Softmax X Value	v_1	[blue]	v_2	[blue]
Sum	z_1 [pink]		z_2 [pink]	

$$\begin{array}{ccc} \mathbf{X} & \times & \mathbf{WQ} \\ \begin{matrix} \text{[green]} \\ \text{[green]} \\ \text{[green]} \\ \text{[green]} \end{matrix} & \times & \begin{matrix} \text{[purple]} \\ \text{[purple]} \\ \text{[purple]} \\ \text{[purple]} \end{matrix} \\ = & \mathbf{Q} & \begin{matrix} \text{[purple]} \\ \text{[purple]} \\ \text{[purple]} \\ \text{[purple]} \end{matrix} \end{array}$$
$$\begin{array}{ccc} \mathbf{X} & \times & \mathbf{WK} \\ \begin{matrix} \text{[green]} \\ \text{[green]} \\ \text{[green]} \\ \text{[green]} \end{matrix} & \times & \begin{matrix} \text{[orange]} \\ \text{[orange]} \\ \text{[orange]} \\ \text{[orange]} \end{matrix} \\ = & \mathbf{K} & \begin{matrix} \text{[orange]} \\ \text{[orange]} \\ \text{[orange]} \\ \text{[orange]} \end{matrix} \end{array}$$
$$\begin{array}{ccc} \mathbf{X} & \times & \mathbf{WV} \\ \begin{matrix} \text{[green]} \\ \text{[green]} \\ \text{[green]} \\ \text{[green]} \end{matrix} & \times & \begin{matrix} \text{[blue]} \\ \text{[blue]} \\ \text{[blue]} \\ \text{[blue]} \end{matrix} \\ = & \mathbf{V} & \begin{matrix} \text{[blue]} \\ \text{[blue]} \\ \text{[blue]} \\ \text{[blue]} \end{matrix} \end{array}$$

Okay.. How to calculate self-attention?

$$\begin{matrix} \mathbf{x} \\ \begin{array}{|c|c|c|c|}\hline & & & \\ \hline \end{array} \end{matrix} \times \begin{matrix} \mathbf{W}^Q \\ \begin{array}{|c|c|c|c|}\hline & & & \\ \hline \end{array} \end{matrix} = \begin{matrix} \mathbf{Q} \\ \begin{array}{|c|c|c|}\hline & & \\ \hline & & \\ \hline & & \\ \hline \end{array} \end{matrix}$$

$$\begin{matrix} \mathbf{x} \\ \begin{array}{|c|c|c|c|}\hline & & & \\ \hline \end{array} \end{matrix} \times \begin{matrix} \mathbf{W}^K \\ \begin{array}{|c|c|c|c|}\hline & & & \\ \hline \end{array} \end{matrix} = \begin{matrix} \mathbf{K} \\ \begin{array}{|c|c|c|}\hline & & \\ \hline & & \\ \hline & & \\ \hline \end{array} \end{matrix}$$

$$\begin{matrix} \mathbf{x} \\ \begin{array}{|c|c|c|c|}\hline & & & \\ \hline \end{array} \end{matrix} \times \begin{matrix} \mathbf{W}^V \\ \begin{array}{|c|c|c|c|}\hline & & & \\ \hline \end{array} \end{matrix} = \begin{matrix} \mathbf{V} \\ \begin{array}{|c|c|c|}\hline & & \\ \hline & & \\ \hline & & \\ \hline \end{array} \end{matrix}$$

$$\text{softmax}\left(\frac{\mathbf{Q} \times \mathbf{K}^T}{\sqrt{d_k}}\right) \mathbf{V} = \mathbf{Z}$$

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Okay.. How to calculate self-attention?

Input	Thinking		Machines	
Embedding	x_1	[green]	x_2	[green]
Queries	q_1	[purple]	q_2	[purple]
Keys	k_1	[orange]	k_2	[orange]
Values	v_1	[blue]	v_2	[blue]
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	
Softmax X Value	v_1	[blue]	v_2	[blue]
Sum	z_1 [pink]		z_2 [pink]	

$$\begin{array}{ccc} \mathbf{X} & \times & \mathbf{WQ} \\ \begin{matrix} \text{[green]} \\ \text{[green]} \\ \text{[green]} \\ \text{[green]} \end{matrix} & \times & \begin{matrix} \text{[purple]} \\ \text{[purple]} \\ \text{[purple]} \\ \text{[purple]} \end{matrix} \\ = & \mathbf{Q} & \begin{matrix} \text{[purple]} \\ \text{[purple]} \\ \text{[purple]} \\ \text{[purple]} \end{matrix} \end{array}$$
$$\begin{array}{ccc} \mathbf{X} & \times & \mathbf{WK} \\ \begin{matrix} \text{[green]} \\ \text{[green]} \\ \text{[green]} \\ \text{[green]} \end{matrix} & \times & \begin{matrix} \text{[orange]} \\ \text{[orange]} \\ \text{[orange]} \\ \text{[orange]} \end{matrix} \\ = & \mathbf{K} & \begin{matrix} \text{[orange]} \\ \text{[orange]} \\ \text{[orange]} \\ \text{[orange]} \end{matrix} \end{array}$$
$$\begin{array}{ccc} \mathbf{X} & \times & \mathbf{WV} \\ \begin{matrix} \text{[green]} \\ \text{[green]} \\ \text{[green]} \\ \text{[green]} \end{matrix} & \times & \begin{matrix} \text{[blue]} \\ \text{[blue]} \\ \text{[blue]} \\ \text{[blue]} \end{matrix} \\ = & \mathbf{V} & \begin{matrix} \text{[blue]} \\ \text{[blue]} \\ \text{[blue]} \\ \text{[blue]} \end{matrix} \end{array}$$

Okay.. But why? What is Q,K,V?

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

- In fact, self-attention calculation is this, why?

$$\text{Softmax}(XX^T)X$$

- Dot product of two vectors a means the projection

$$\begin{aligned}[1, 3, -5] \cdot [4, -2, -1] &= (1)(4) + (3)(-2) + (-5)(-1) \\ &= 4 - 6 + 5 \\ &= 3\end{aligned}$$

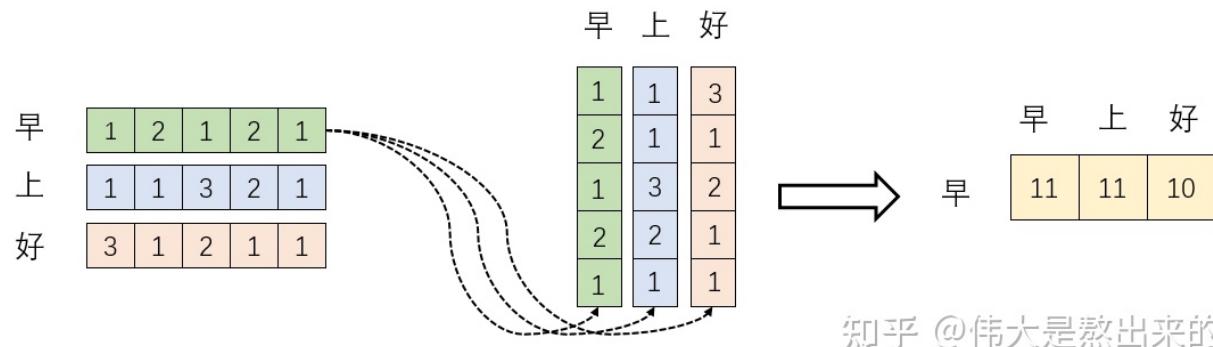
- The projection of vector a into vector b/The dot product of measures} the **similarity** of two vectors.

Okay.. But why? What is Q,K,V?

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

- In fact, self-attention calculation is this, why?

$$\text{Softmax}(XX^T)X$$



知乎 @伟大是熬出来的

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{c=1}^C e^{z_c}}$$

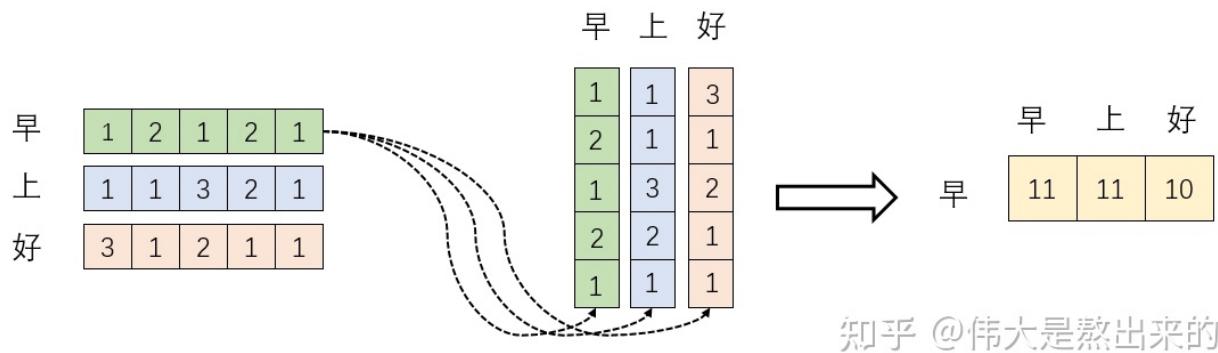
知乎 @伟大是熬出来的

Okay.. But why? What is Q,K,V?

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

- In fact, self-attention calculation is this, why?

$$\text{Softmax}(XX^T)X$$

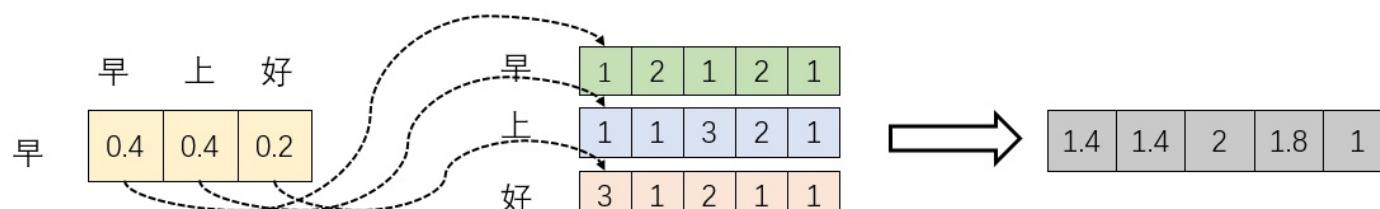


Okay.. But why? What is Q,K,V?

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

- In fact, self-attention calculation is this, why?

$$\text{Softmax}(XX^T)X$$

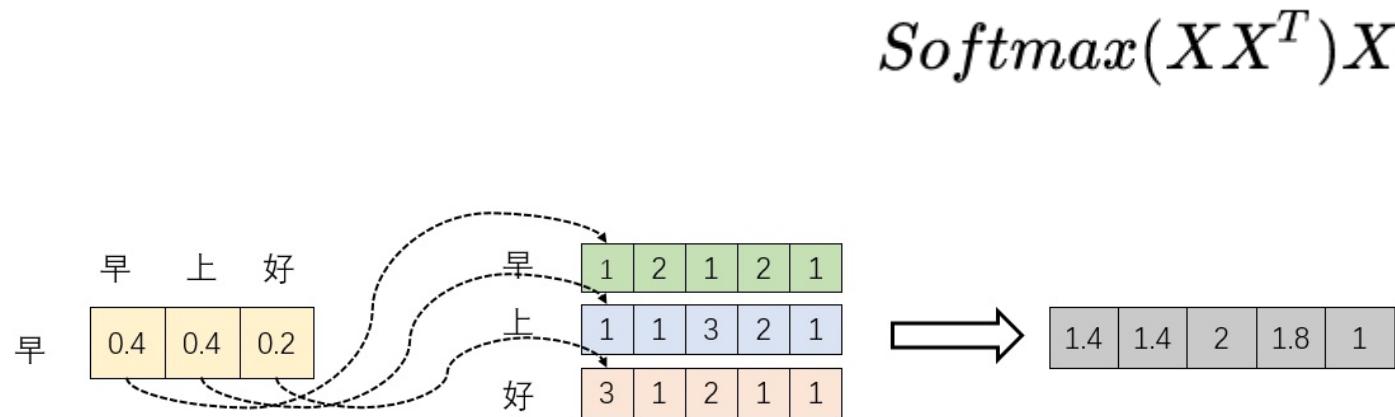


知乎 @伟大是熬出来的

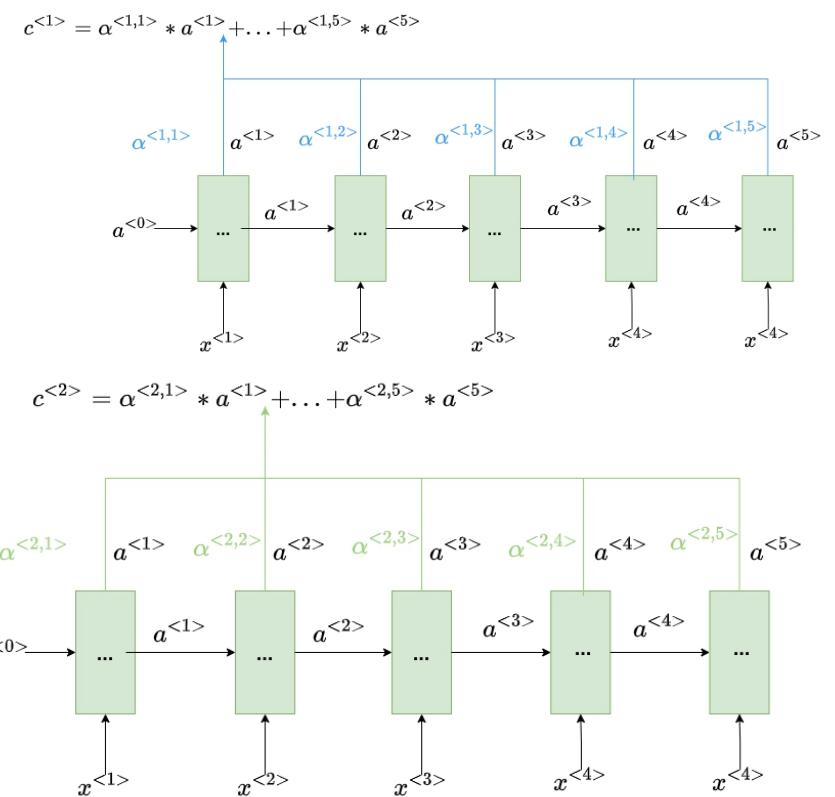
Okay.. But why? What is Q,K,V?

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

- In fact, self-attention calculation is this, why?



知乎 @伟大是熬出来的



How does Q,K,V correlates with X?

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

$$Softmax(XX^T)X$$

$$\begin{matrix} X & \times & W^Q & = & Q \end{matrix}$$

The diagram shows three matrices: X (green, 2x5), W^Q (purple, 5x5), and Q (purple, 2x4). The multiplication operation \times is placed between X and W^Q , and the equals sign $=$ is placed after Q .

$$\begin{matrix} X & \times & W^K & = & K \end{matrix}$$

The diagram shows three matrices: X (green, 2x5), W^K (orange, 5x5), and K (orange, 2x2). The multiplication $X \times W^K$ results in matrix K .

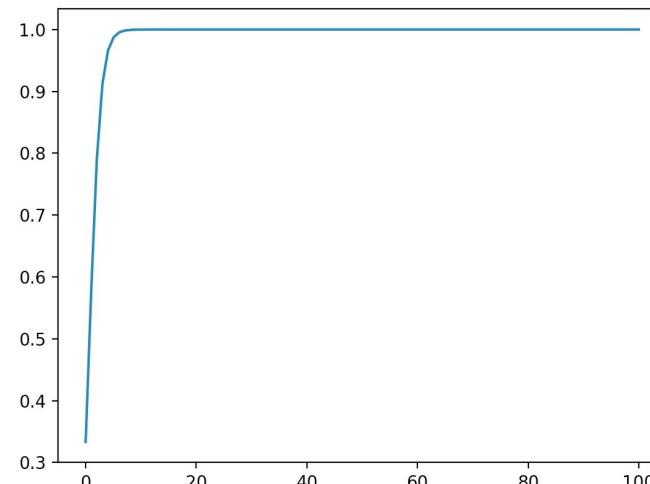
$$\begin{matrix} \text{X} & \times & \text{WV} & = & \text{V} \\ \begin{matrix} \text{---} \\ | \\ \text{---} \\ | \\ \text{---} \\ | \\ \text{---} \\ | \\ \text{---} \end{matrix} & & \begin{matrix} \text{---} \\ | \\ \text{---} \\ | \\ \text{---} \\ | \\ \text{---} \\ | \\ \text{---} \end{matrix} & & \begin{matrix} \text{---} \\ | \\ \text{---} \\ | \\ \text{---} \\ | \\ \text{---} \\ | \\ \text{---} \end{matrix} \end{matrix}$$

What about $\sqrt{d_k}$?

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Additive attention computes the compatibility function using a feed-forward network with a single hidden layer. Dot product attention is much faster and space-efficient.

For large values of d_k will also make dot-product attention large.

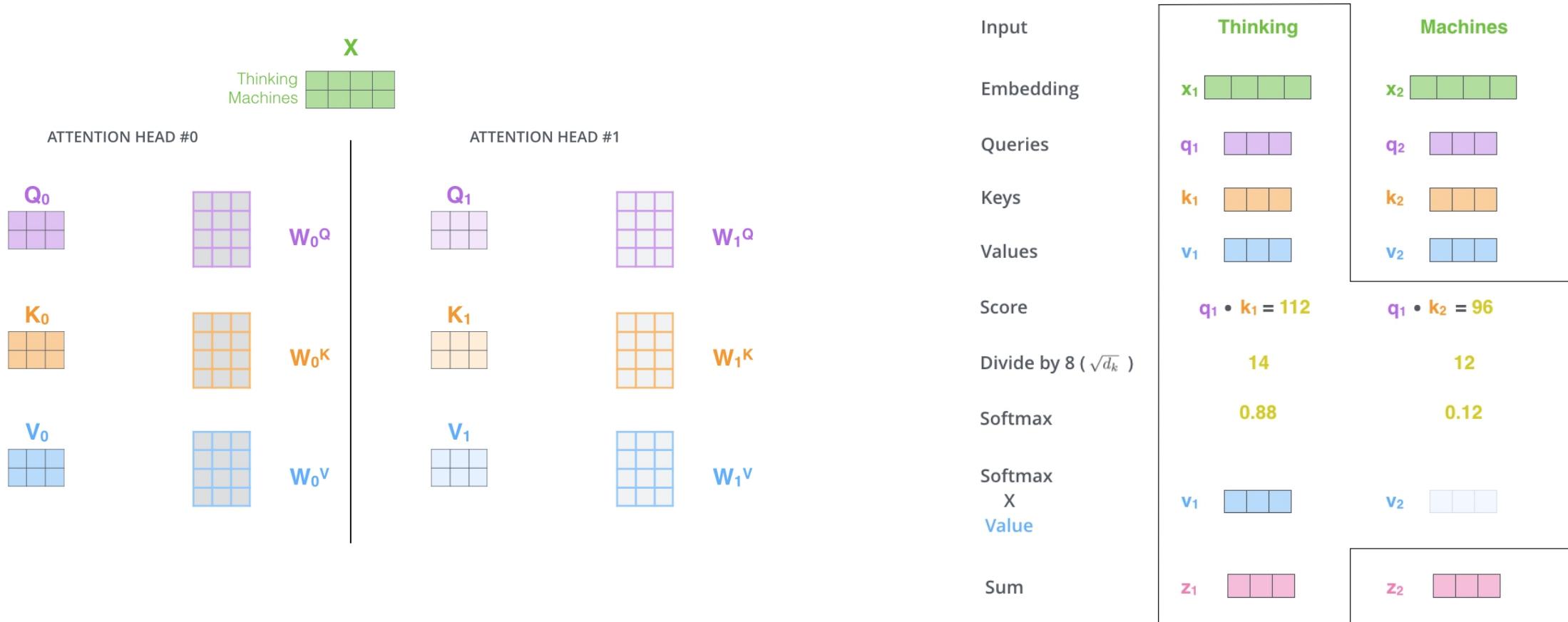


$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{c=1}^C e^{z_c}}$$

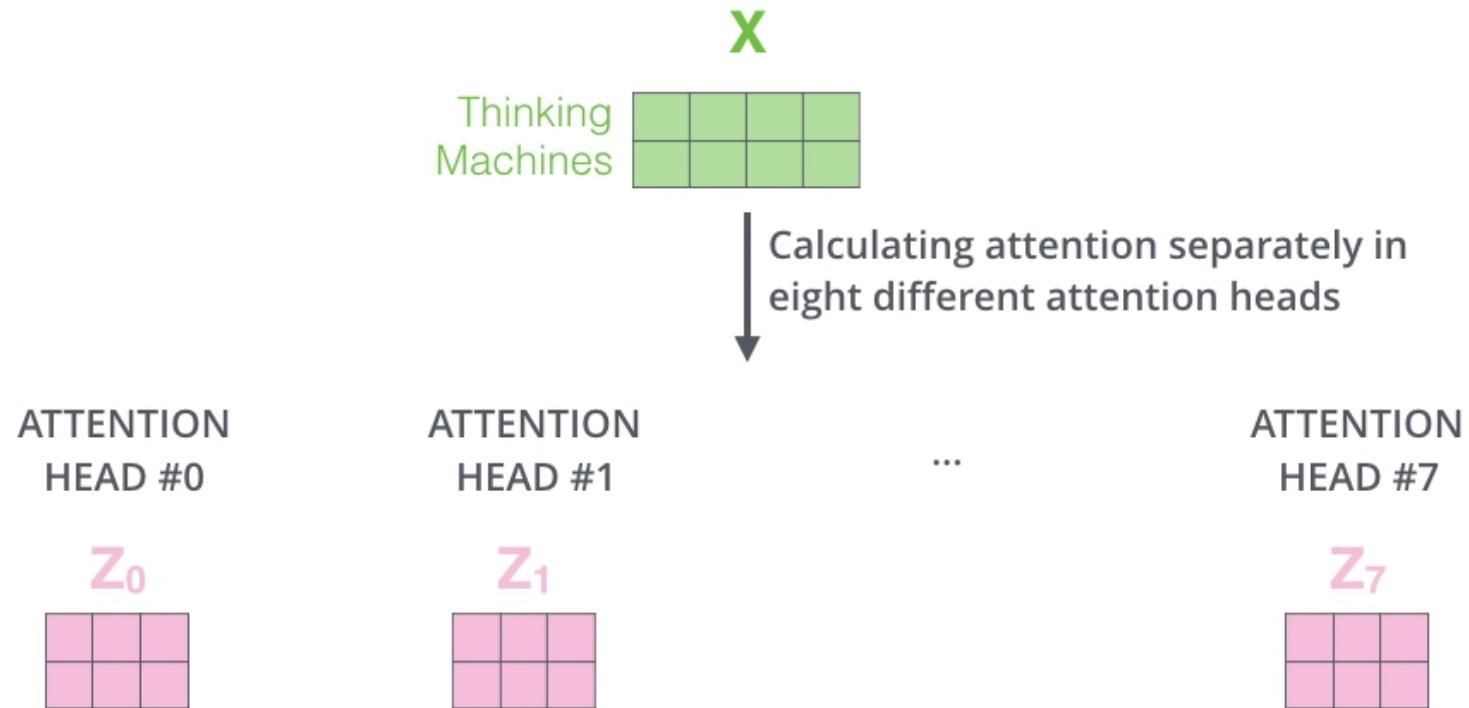
知乎 @伟大是熬出来的

Proof: <https://www.zhihu.com/question/339723385> (If you need further explanation, email me.)

What is multi-head self-attention



What is multi-head self-attention



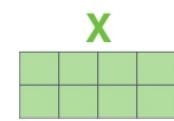
What is multi-head self-attention

1) This is our input sentence*

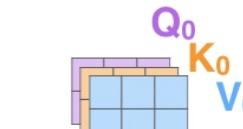
2) We embed each word*

Thinking
Machines

3) Split into 8 heads.
We multiply X or R with weight matrices



4) Calculate attention using the resulting $Q/K/V$ matrices



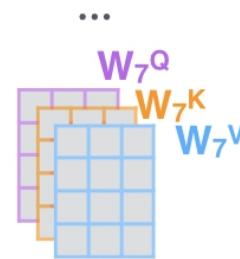
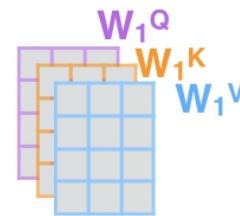
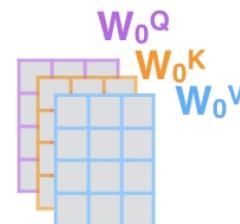
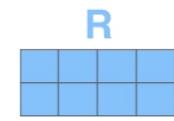
5) Concatenate the resulting Z matrices, then multiply with weight matrix W^o to produce the output of the layer



W^o

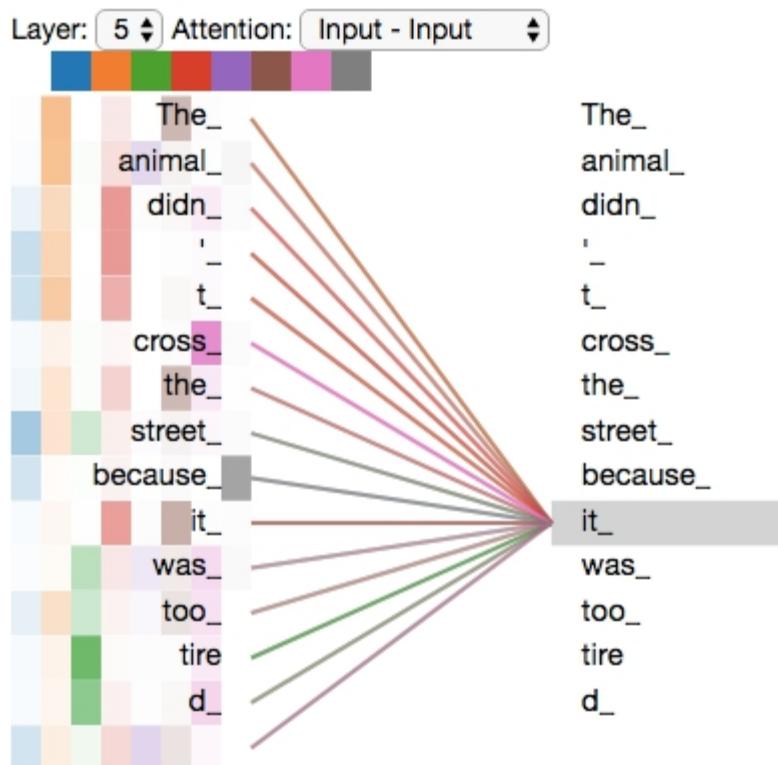


* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one

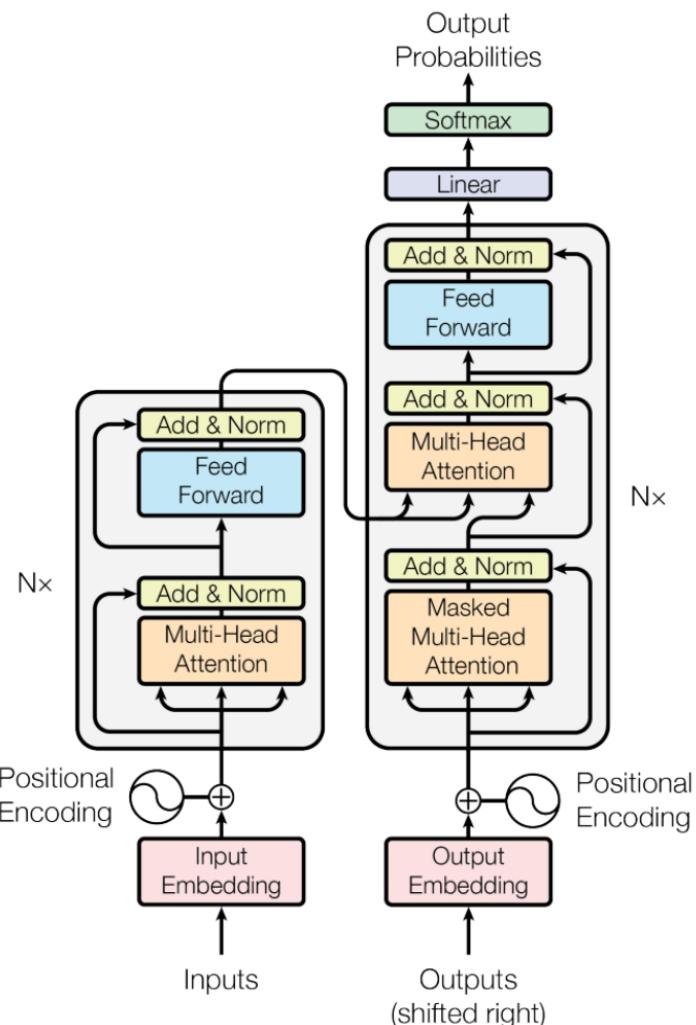


Why multi-head self-attention

- It expands the model's ability to focus on different positions.
- It gives the attention layer multiple “representation subspaces”. After training, each Q,K,V is used to project the input embeddings (or vectors from lower encoders/decoders) into a different representation subspace.



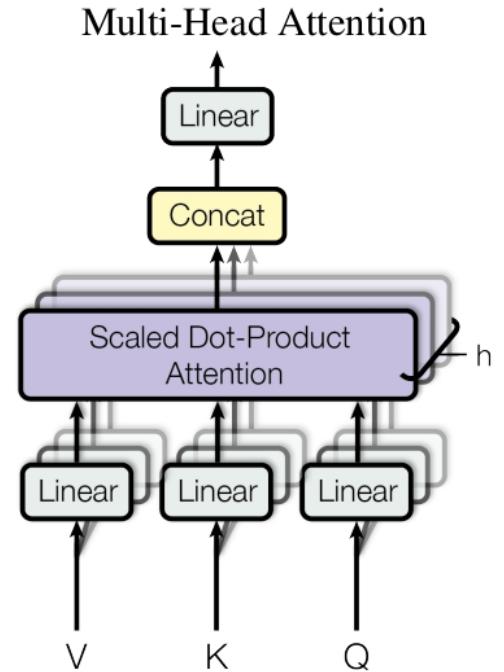
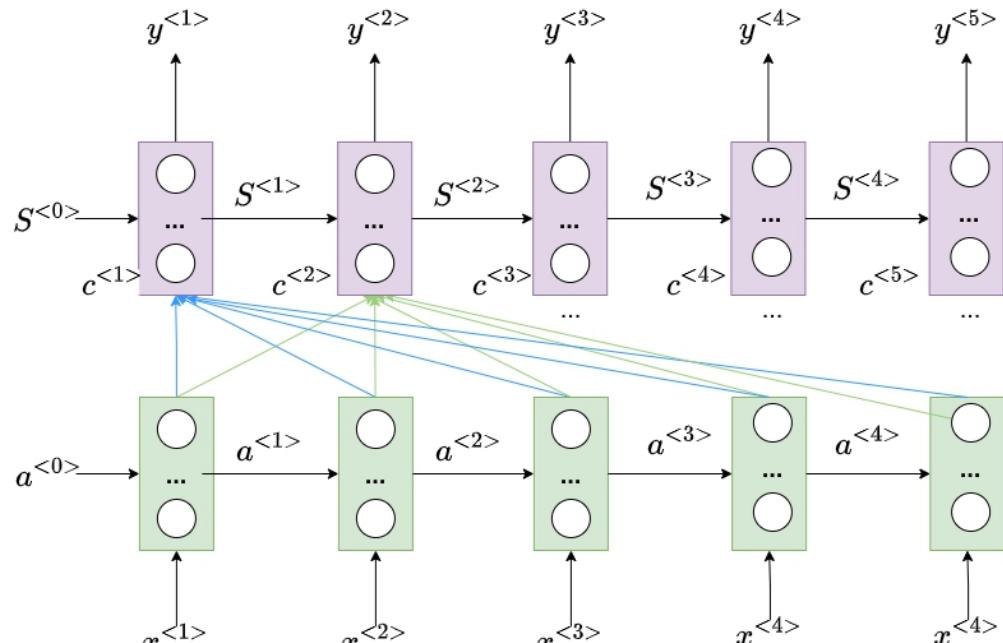
Decoder side



The “Encoder-Decoder Attention” layer works just like multiheaded self-attention, except it **creates its Queries matrix from the layer below it, and takes the Keys and Values matrix from the output of the encoder stack.**

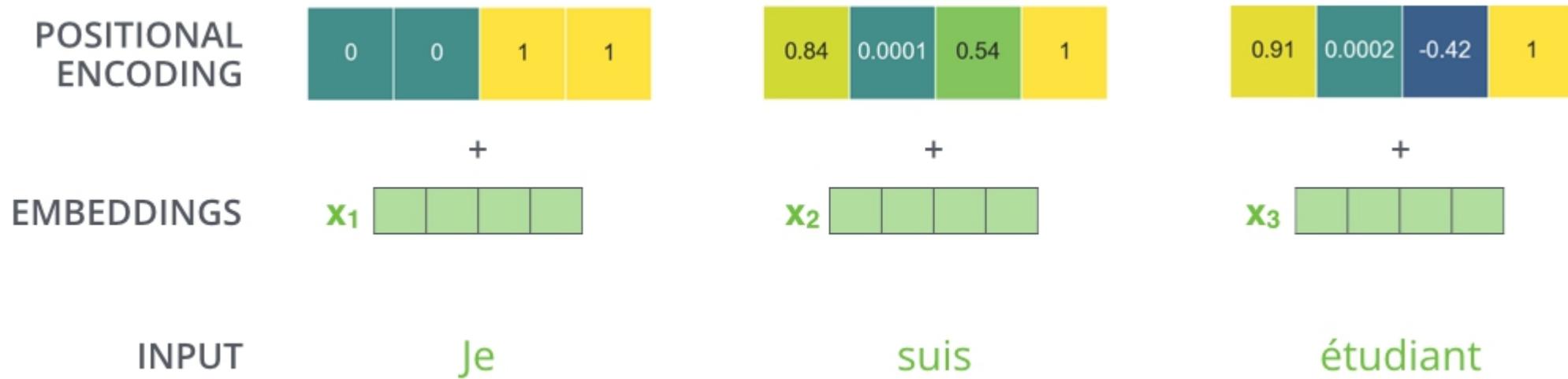
Figure 1: The Transformer - model architecture.

Positional encoding



- Position information for words in sequence is lost.
- But position information is necessary to help translation.
- **Idea: Try to encode position information inside the input embedding.**

Positional encoding



$$PE_{(pos,2i)} = \sin(pos/10000^{2i/d_{model}})$$

$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/d_{model}})$$

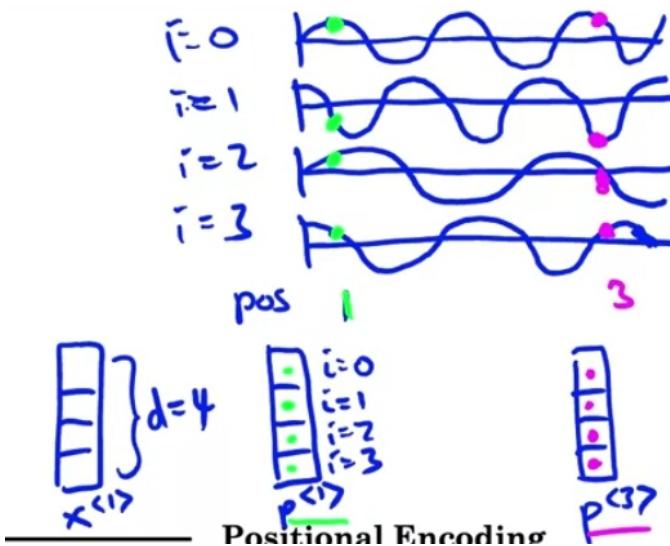
Why choose this encoding?

- For any fixed offset k , $\text{PE}_{\{\text{pos}+k\}}$ can be represented as a linear function of $\text{PE}_{\{\text{pos}\}}$

$$\text{PE}_{(\text{pos}, 2i)} = \sin(\text{pos}/10000^{2i/d_{\text{model}}})$$

$$\text{PE}_{(\text{pos}, 2i+1)} = \cos(\text{pos}/10000^{2i/d_{\text{model}}})$$

- i : encoding dim
- pos : word posi



Masking

- **Intuition: Let the model only focus on what it should focus on.**
- Padding mask: (Set “padded element attention to -inf”)
- Sequence mask: Bans the decode from seeing future information. (Set “future attention to -inf”)

Residual

- **Intuition: When the model goes deeper, gradient vanishing, gradient explosion, and saturation may occur.**

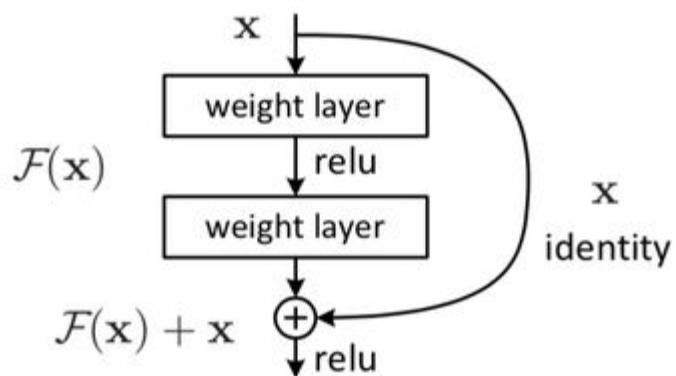


Figure 2. Residual learning: a building block.

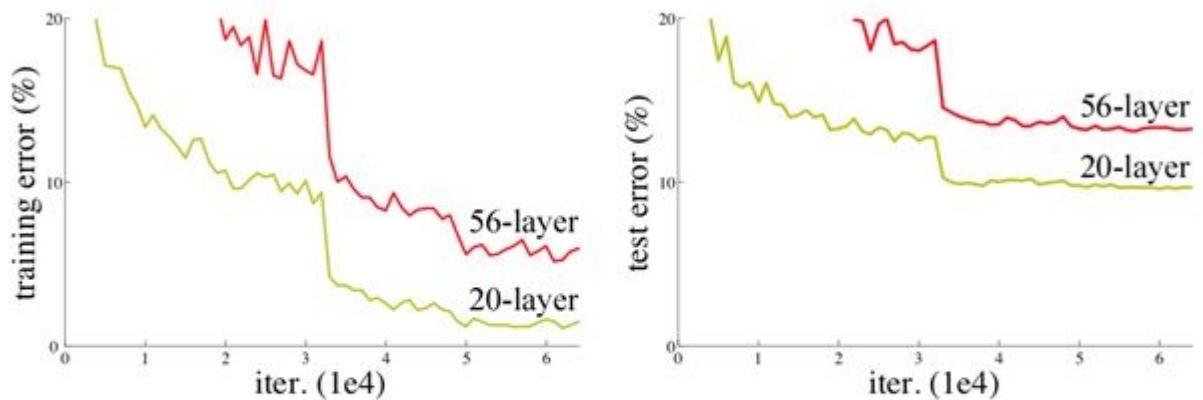


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer “plain” networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

Training

- Encoder-decoder is trained by forward-backward propagation (Just like RNN).
- The result is $P(x=\text{current word}|\text{previous sentences})$.
Beam search is used to generate meaningful sentences.