Attention Is All You Need

Vaswani et al. NeurIPS 2017

Presented by Luke Song

Abstract

- Presents a new neural architecture named the Transformer
- Based solely on the attention mechanism widely used in SEQ2SEQ models
- More parallelizable compared to existing state-of-the-art (SOTA) models
- Achieves SOTA in 2 machine translation datasets

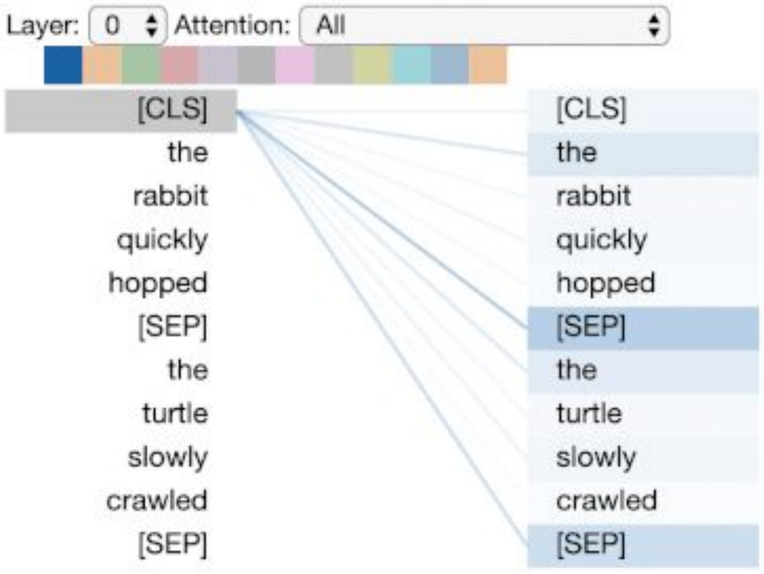
Outline

1. Important Background
2. Model Architecture
3. Experimental Results
4. Model Variation Study
5. Conclusion & Limitation
6. Discussion Time :)

Important Background

What is Attention Mechanism?

- Mechanism used to let individual tokens "attend" to other tokens regardless of the distance between them
- Transformer uses only self-attention which is attention onto the same sentence
- Think of self-attention as recalculating the representation of each token based on how its meaning is influenced by other tokens in the same sentence

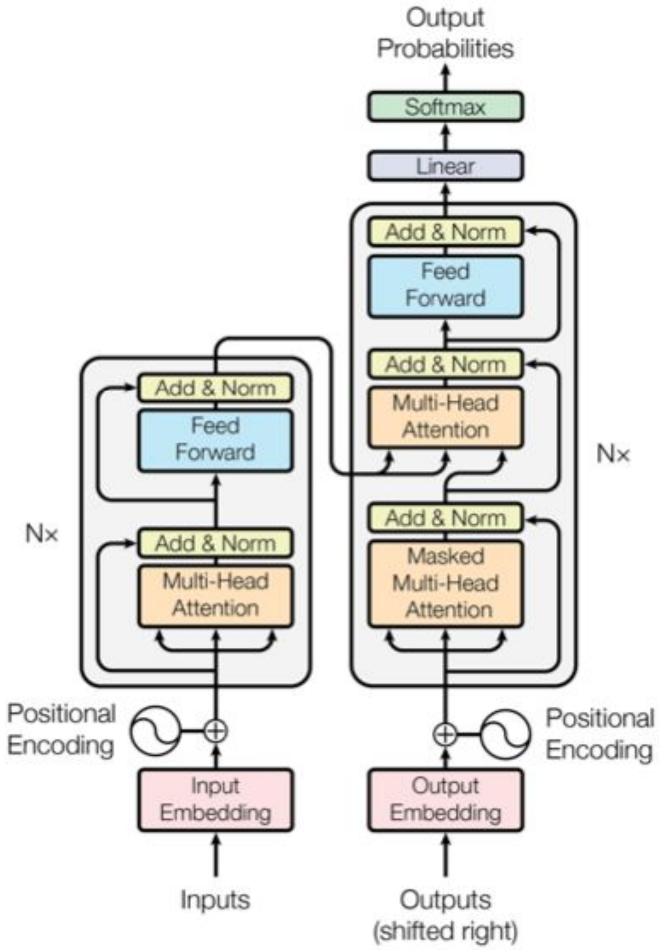


Source: https://github.com/jessevig/bertviz

Model Architecture

High Level

- Input embedding is first added with Positional Encoding
- 3 components in each encoder/decoder: (Masked) Multi-Head Attention, Addition & Normalization, Feed Forward Network



Source: Attention Is All You Need

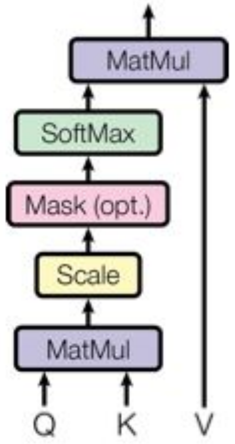
Model Architecture

Attention Function

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

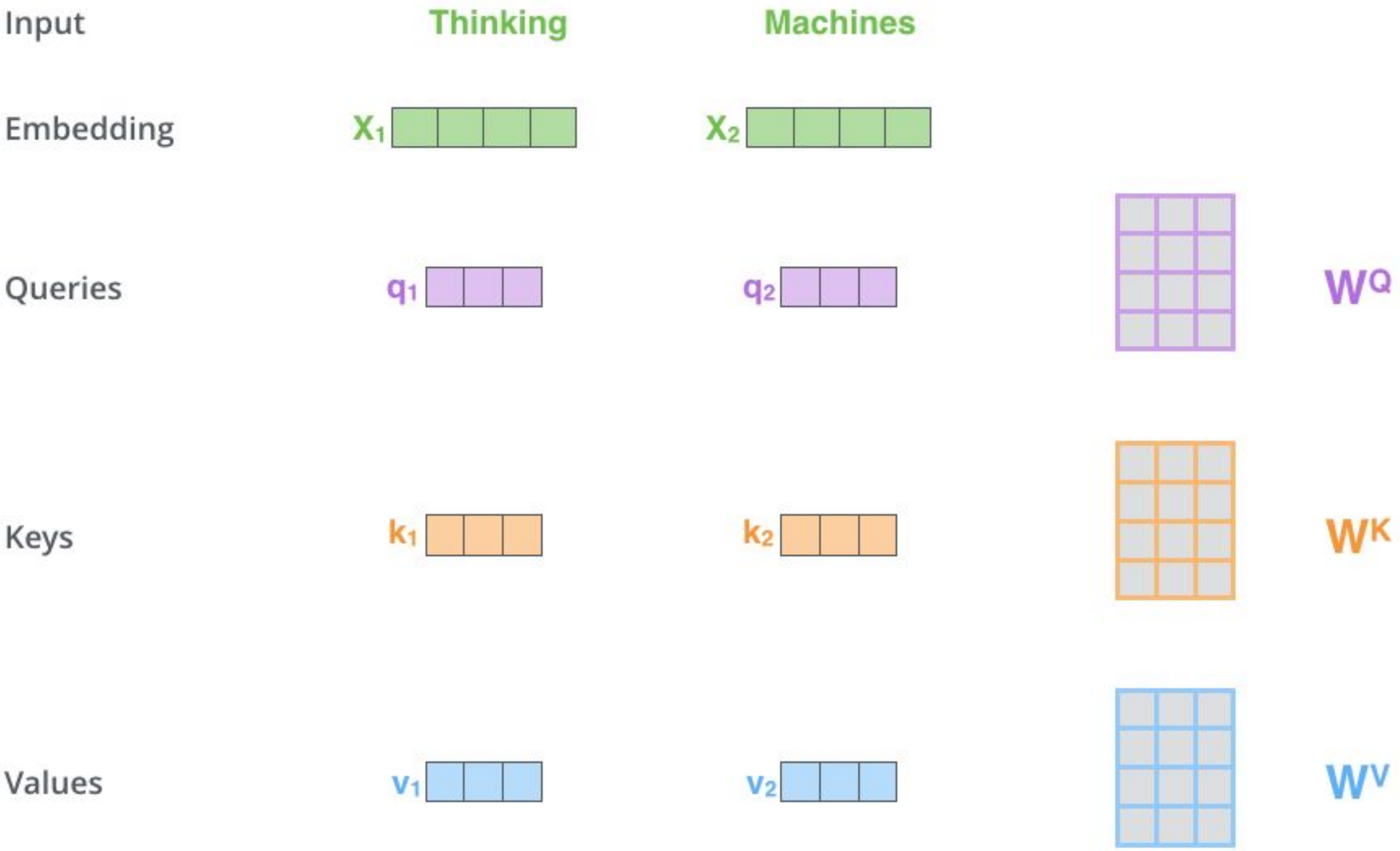
- Mapping a query and set of key-value pairs to an output, where the query, keys, values, and output are all vectors
- Q: Queries
  K: Keys
  V: Values
  d_k: dimension of k (64 in the paper)
- Uses a dot-product attention due to its empirical speed/space advantage
- Scale dot product by 1/sqrt(d_k) b/c large values of d_k may push softmax function to region where it has extremely small gradients

Scaled Dot-Product Attention

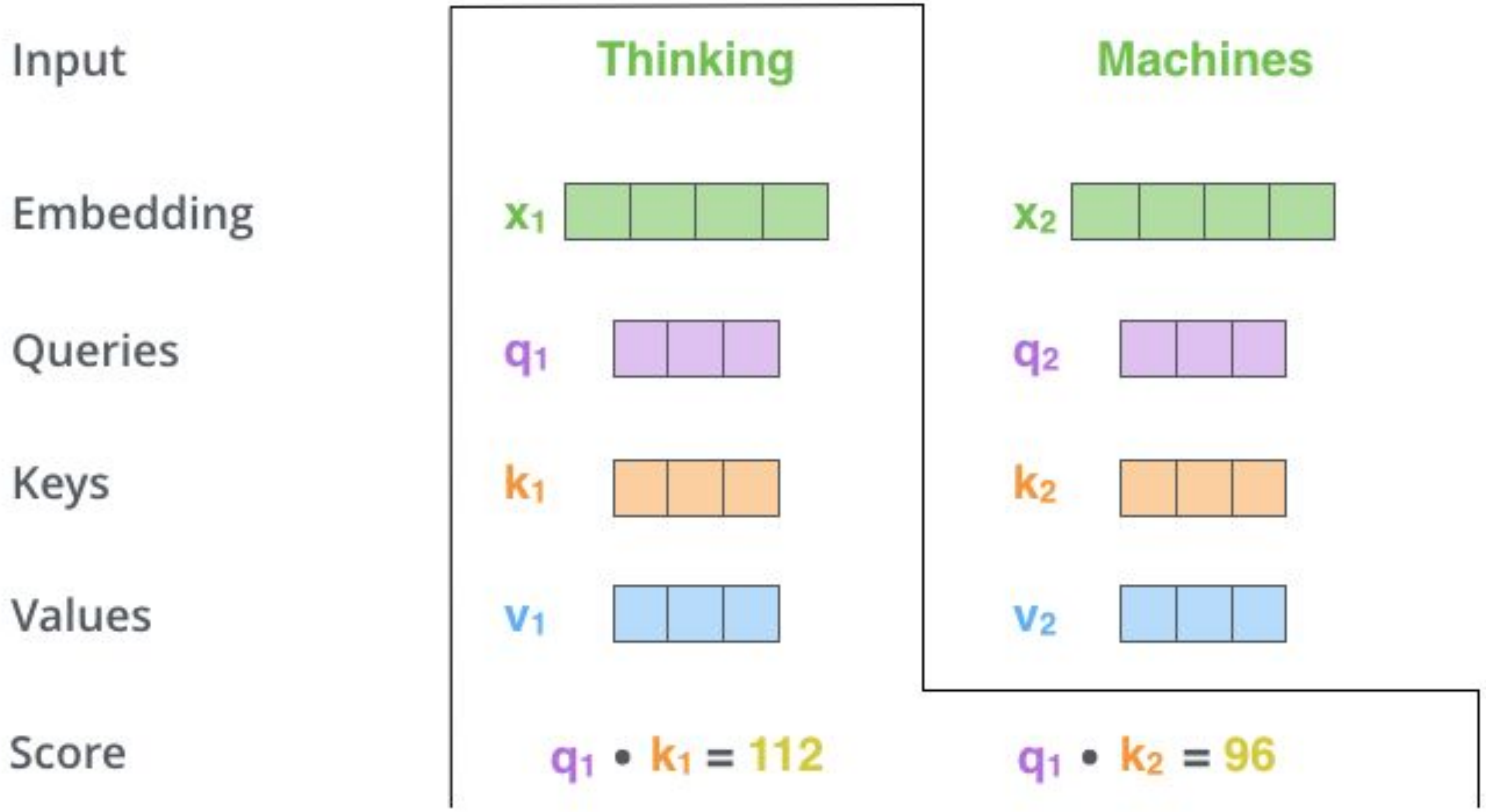


Source: Attention Is All You Need

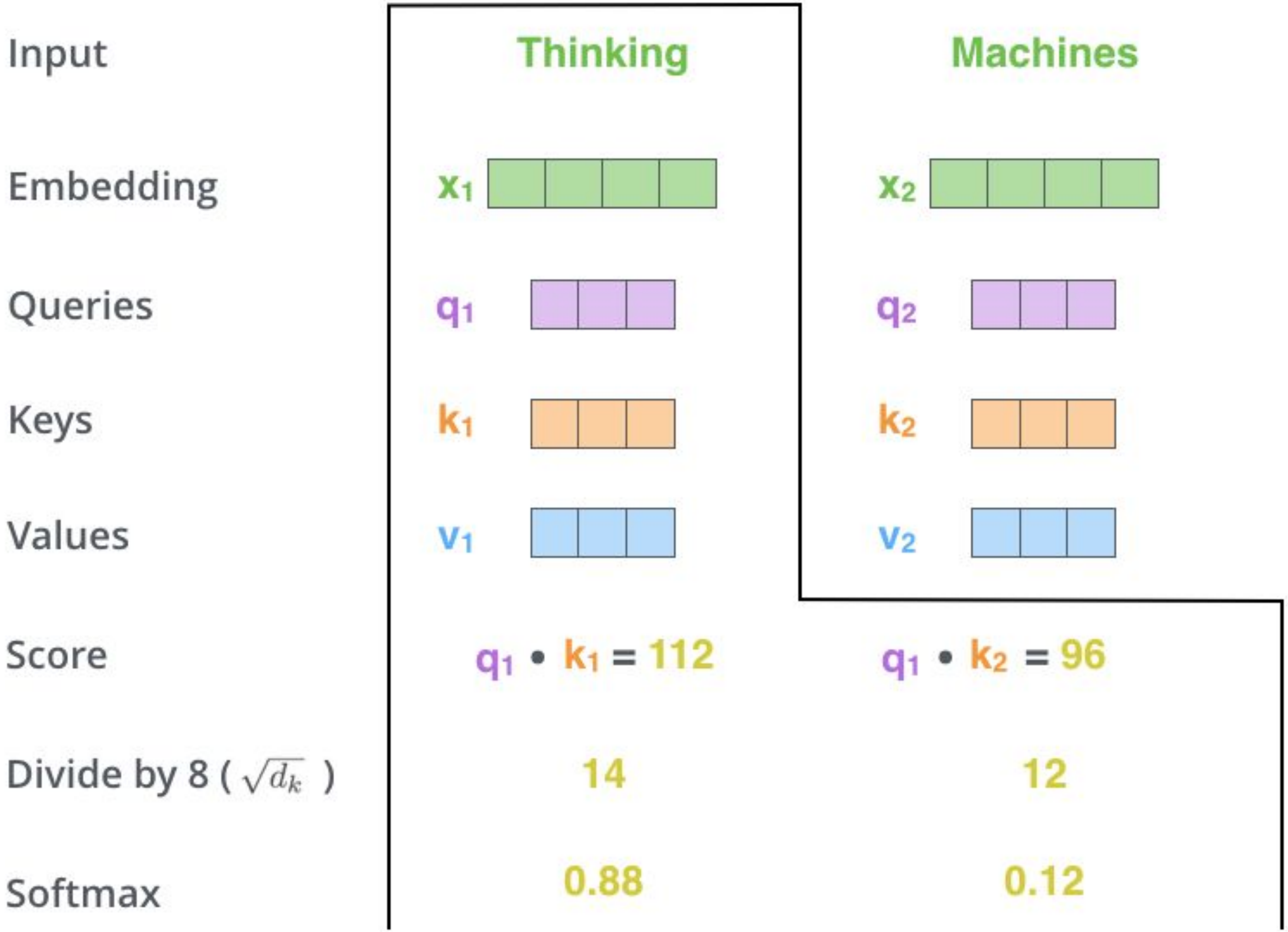
Source: Illustrated Transformer



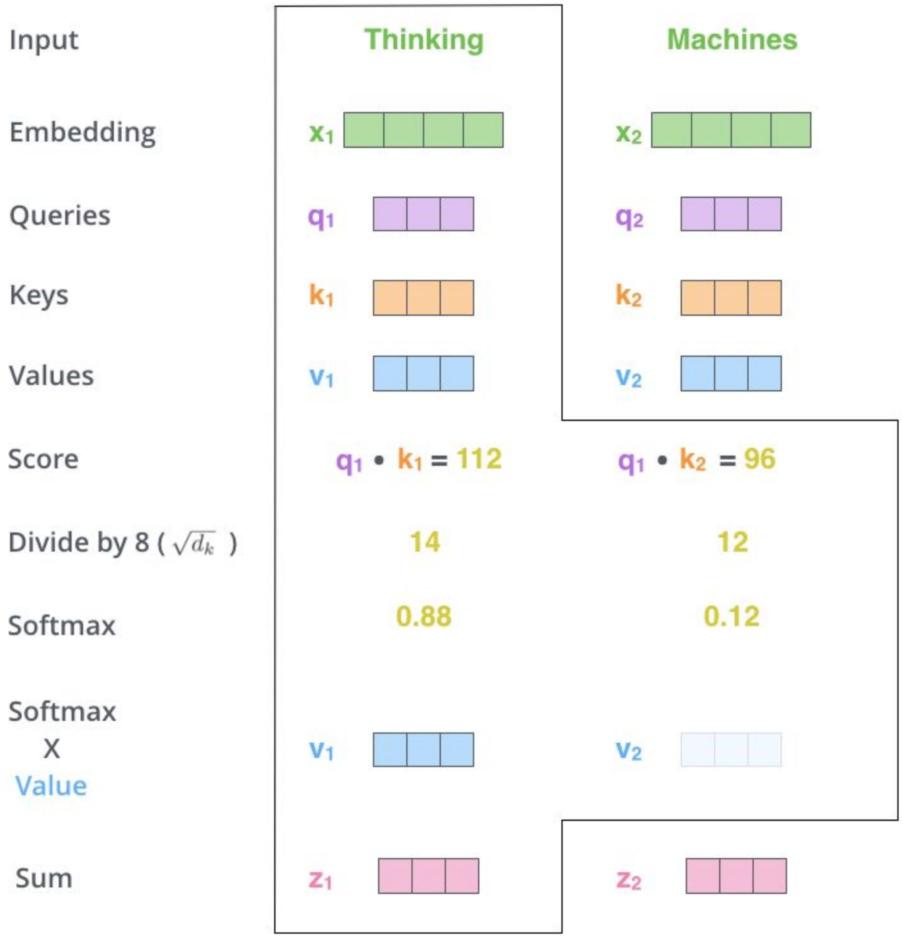
Source: Illustrated Transformer



Source: Illustrated Transformer

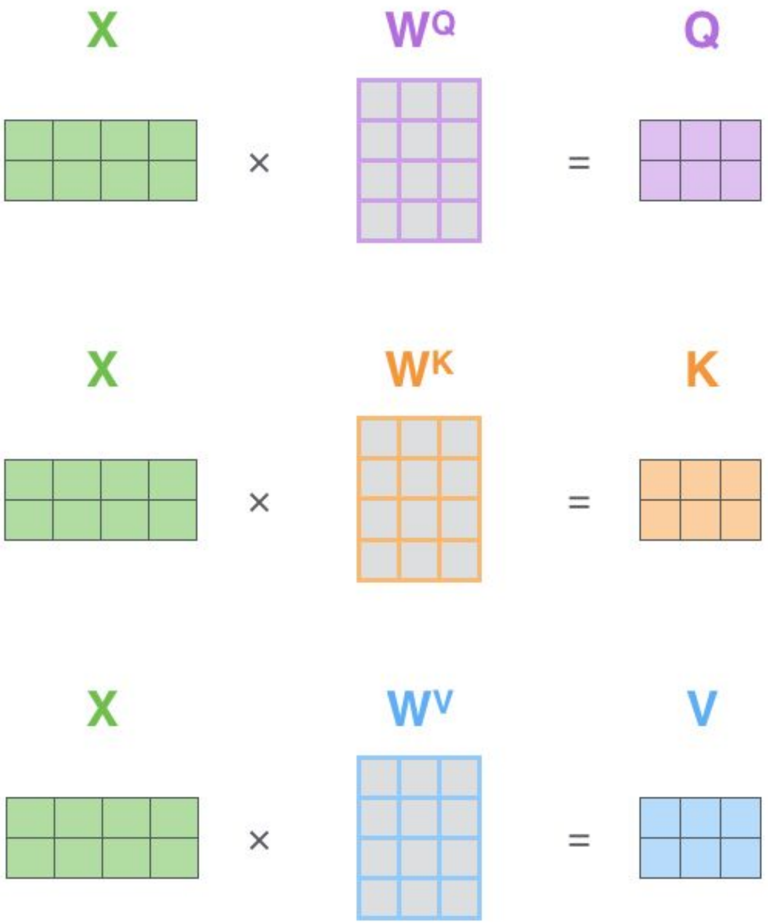


Source: Illustrated Transformer

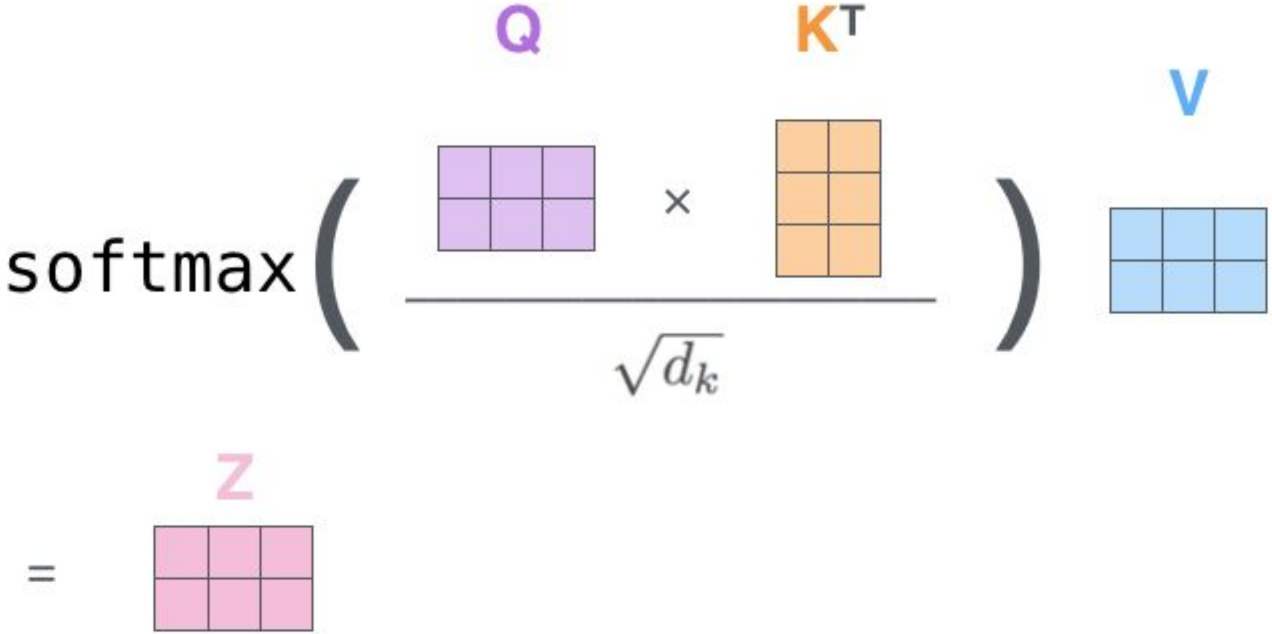


Adding it all together...

Source: Illustrated Transformer



Source: Illustrated Transformer

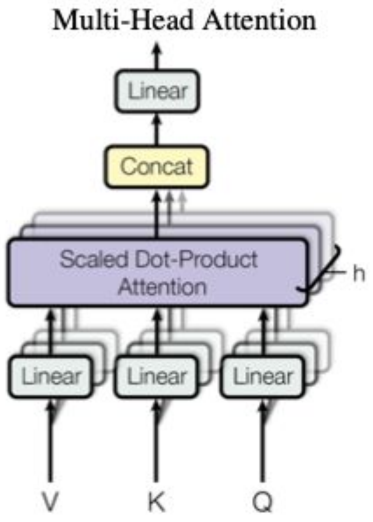


Model Architecture

Multi-Head Attention

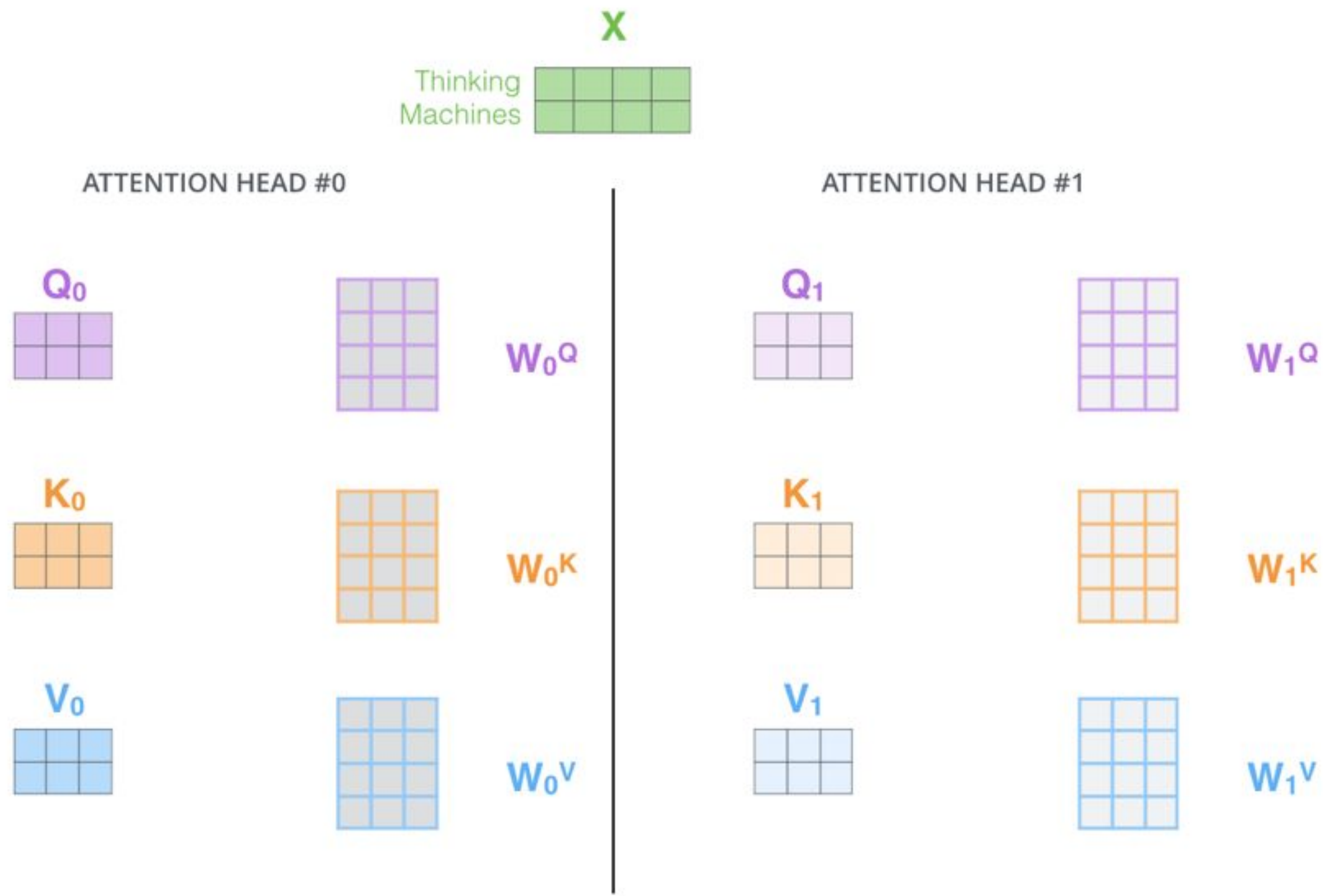
$$\mathrm{MultiHead}(Q, K, V) = \mathrm{Concat}(\mathrm{head_1}, ..., \mathrm{head_h})W^O$$
$$\text{where } \mathrm{head_i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

- Apply attention to different versions of Q, K, V
- Expands model's ability to focus on different positions
- Generates a multiple "representation subspaces" in order to give the model better representation of the input
- Uses 8 attention heads which are concatenated and fed into a linear layer at the end

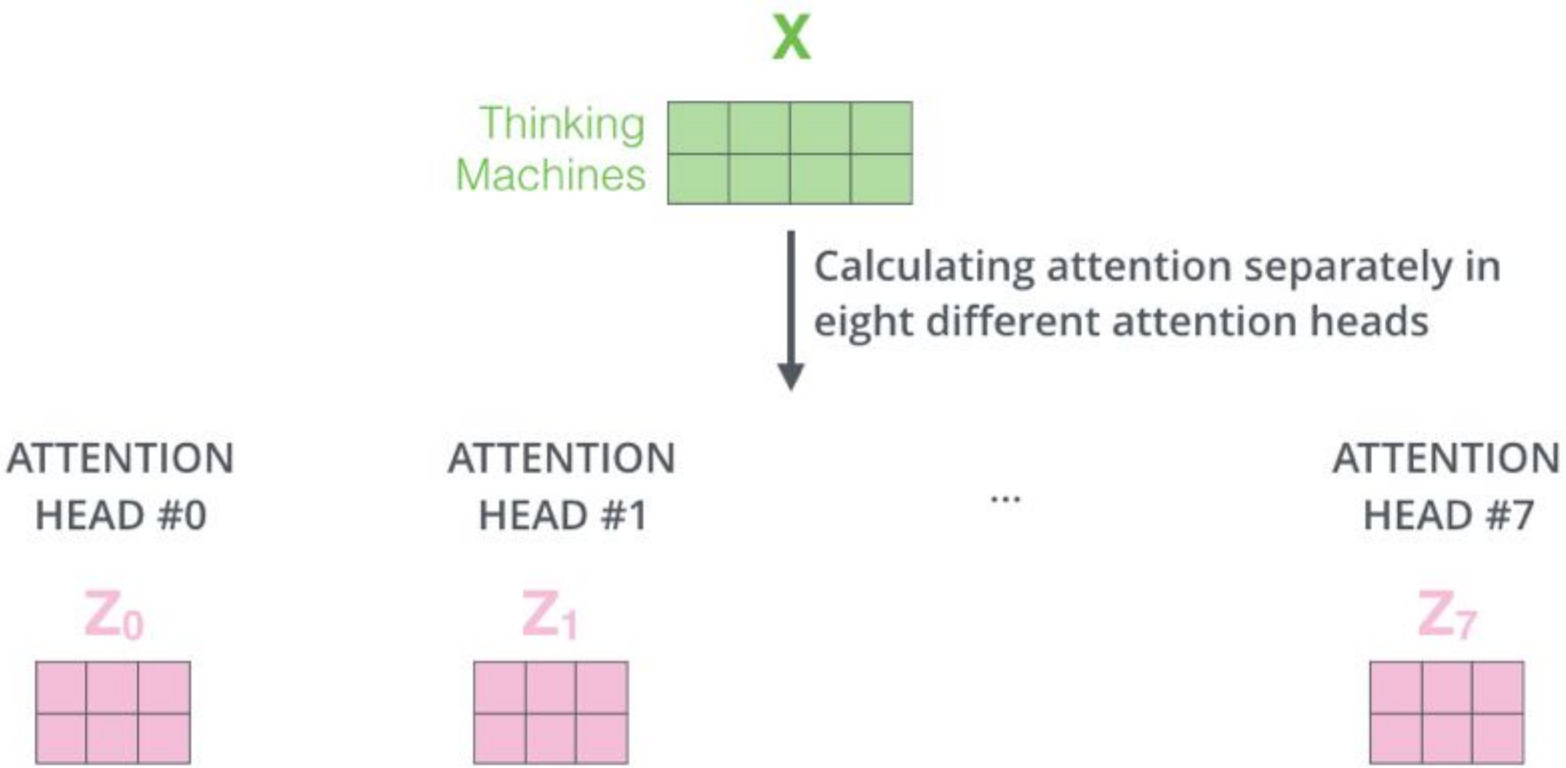


Source: Attention Is All You Need

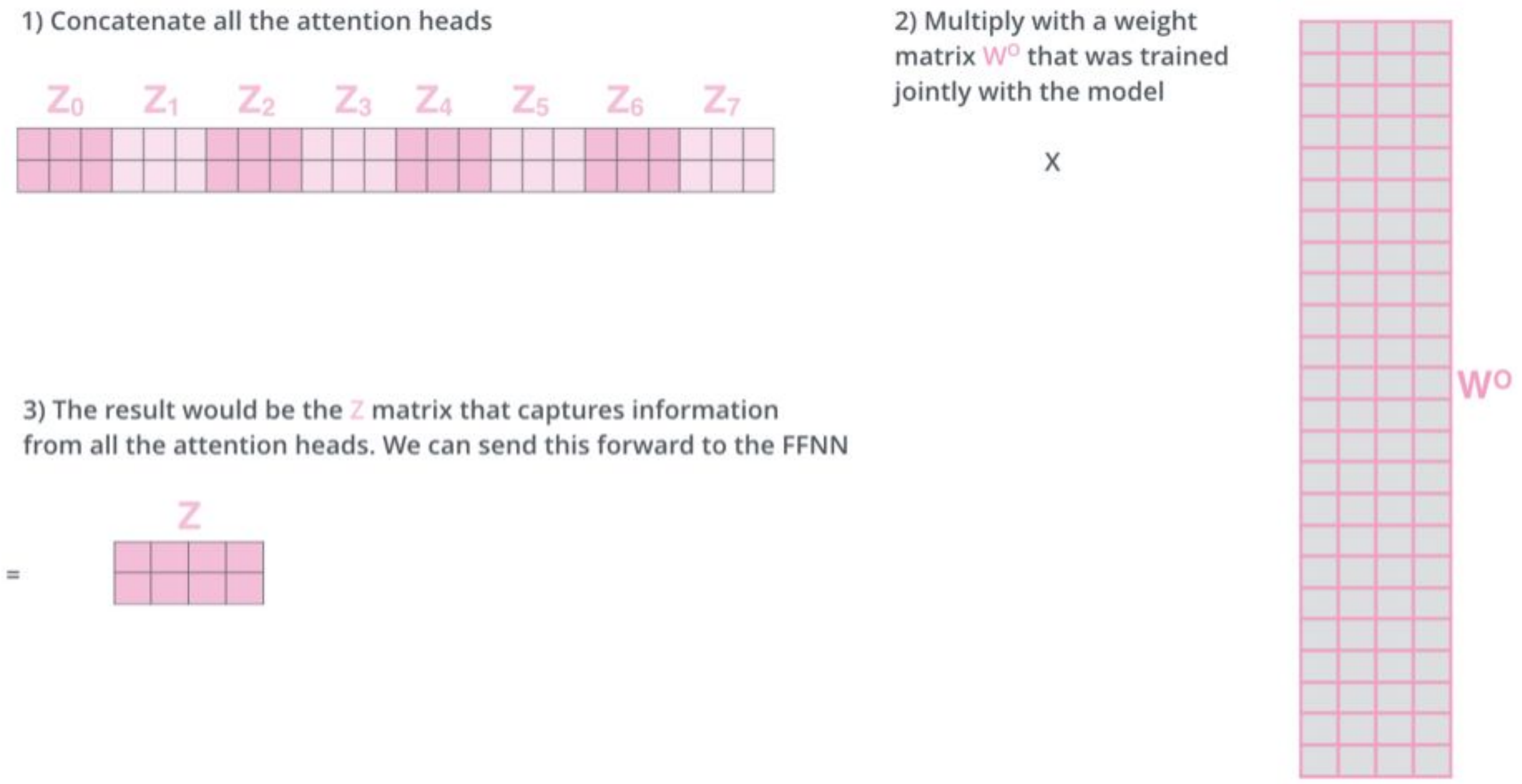
Source: Illustrated Transformer



Source: Illustrated Transformer

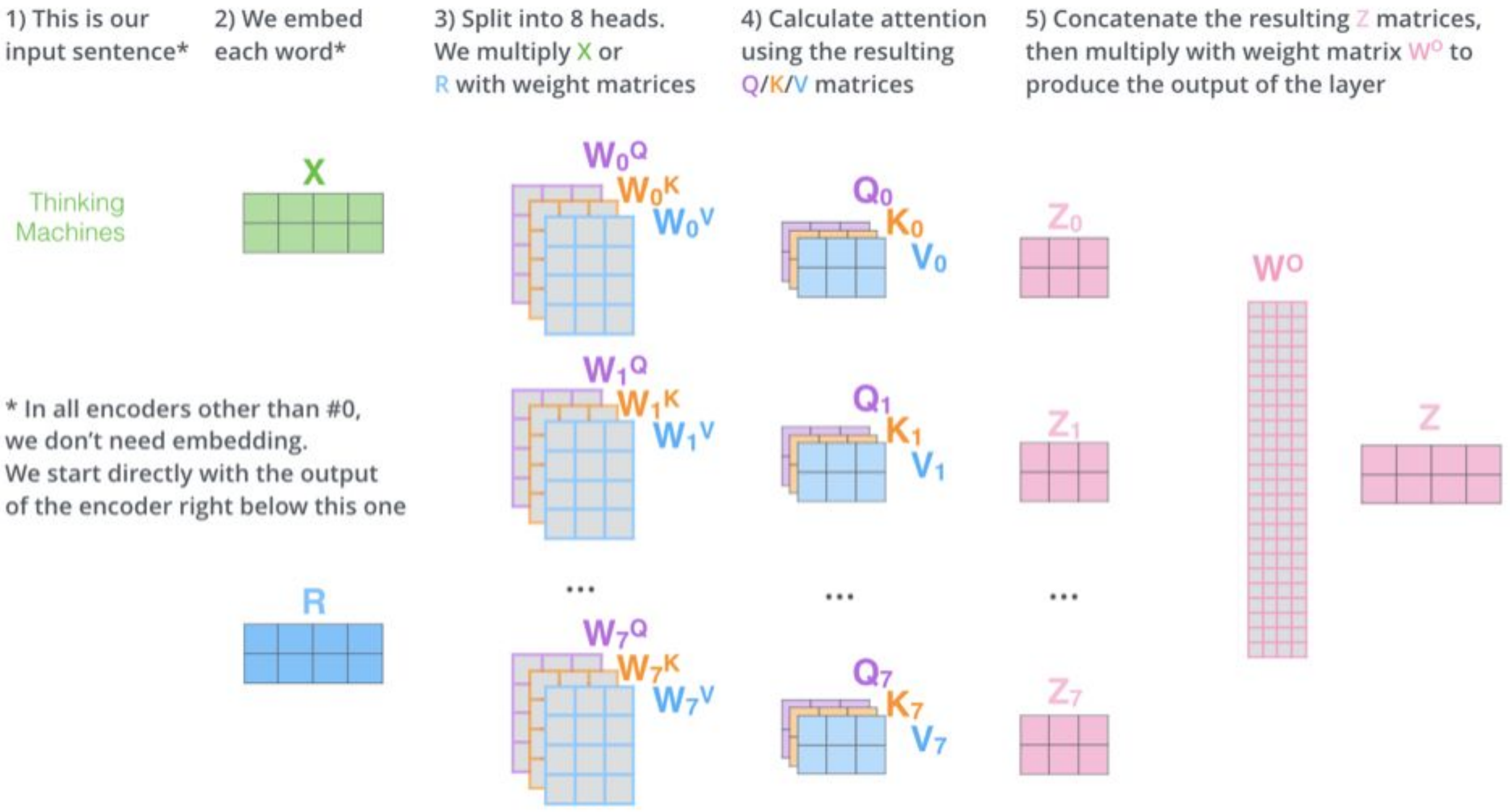


Source: Illustrated Transformer



Combining everything attention-wise...

Source: Illustrated Transformer



Before moving on..

- In encoder, all queries, keys, and values come from the same place
- In encoder-decoder attention layer, queries come from the previous decoder layer and keys and values come from the output of the encoder
- This mimics the typical encoder-decoder attention mechanism
- In decoder to ensure auto-regressive property, the model masks everything right to the current token being attended

Source: Illustrated Transformer, Positional Embedding

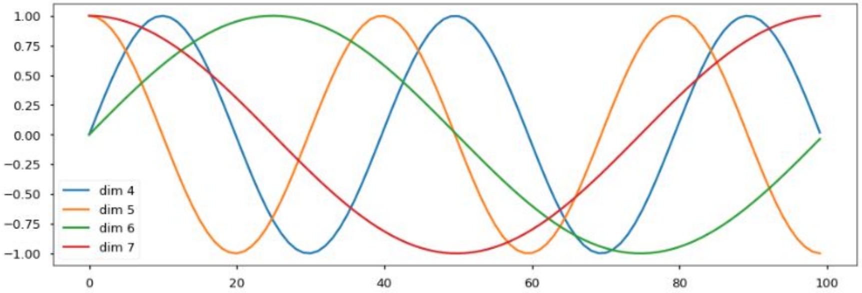
Model Architecture

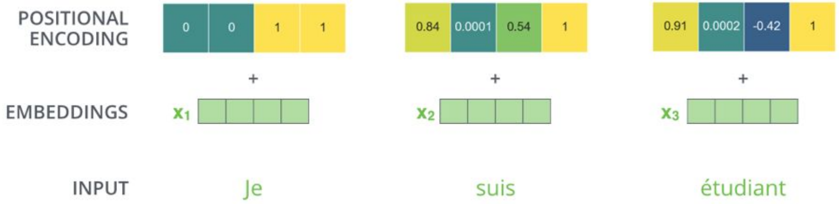
Positional Encoding

- Since attention mechanism in the Transformer does not attend each word auto-regressively (no recurrence nor convolution), model needs something to let it know the relative position of tokens in the sentence
- Positional Encoding is the combination of sine and cosine functions of different frequencies
- Advantages include distance between tokens being symmetrical and being easier to calculate distance between tokens

$$\vec{p_t} = \begin{bmatrix} \sin(\omega_1 . t) \\ \cos(\omega_1 . t) \\ \\ \sin(\omega_2 . t) \\ \cos(\omega_2 . t) \\ \\ \vdots \\ \\ \sin(\omega_{d/2} . t) \\ \cos(\omega_{d/2} . t) \end{bmatrix}_{d\times 1}$$

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





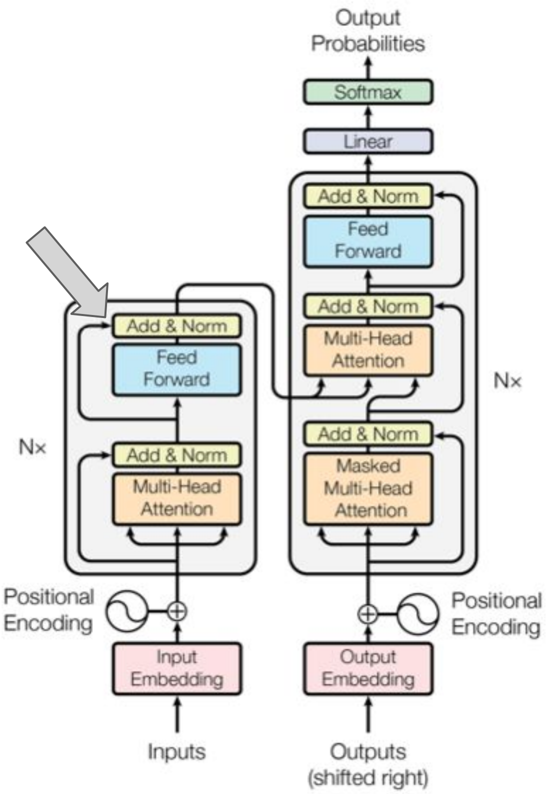
Source: Illustrated Transformer

Model Architecture

$$\mathrm{LayerNorm}(x + \mathrm{Sublayer}(x))$$

Layer Normalization & Residual Connection

- Layer normalization (Ba et al. 2016) is applied to output of sub-layer + input to sub-layer
- Layer normalization normalizes the input across the features
- Empirically shown to reduce training time
- Residual connection means there is a connection that skips few layers (in here 1)

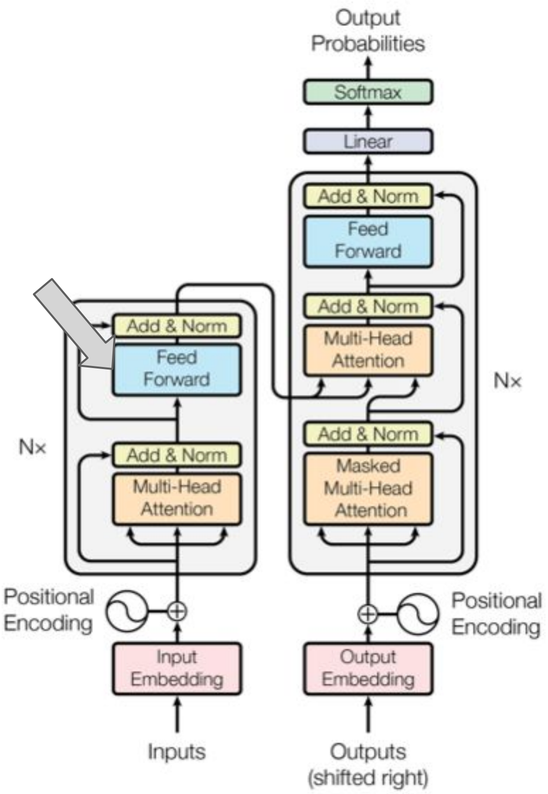


Model Architecture

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

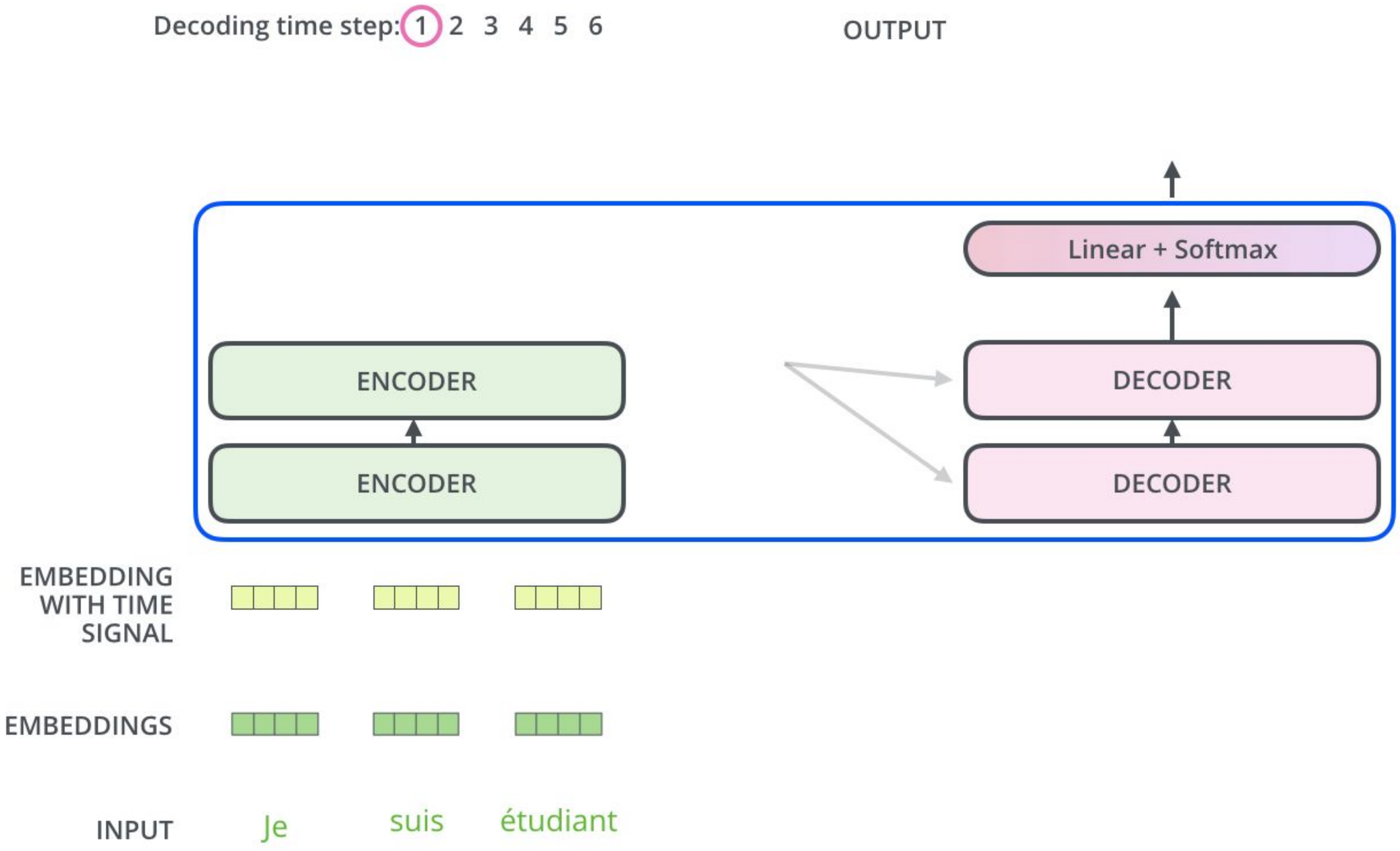
Position-wise Feed Forward Networks

- Fully connected feed-forward network
- Two linear transformations with a ReLU activation in between
- Inner layer has dimensionality of 2048
- Applied to each position separately and identically

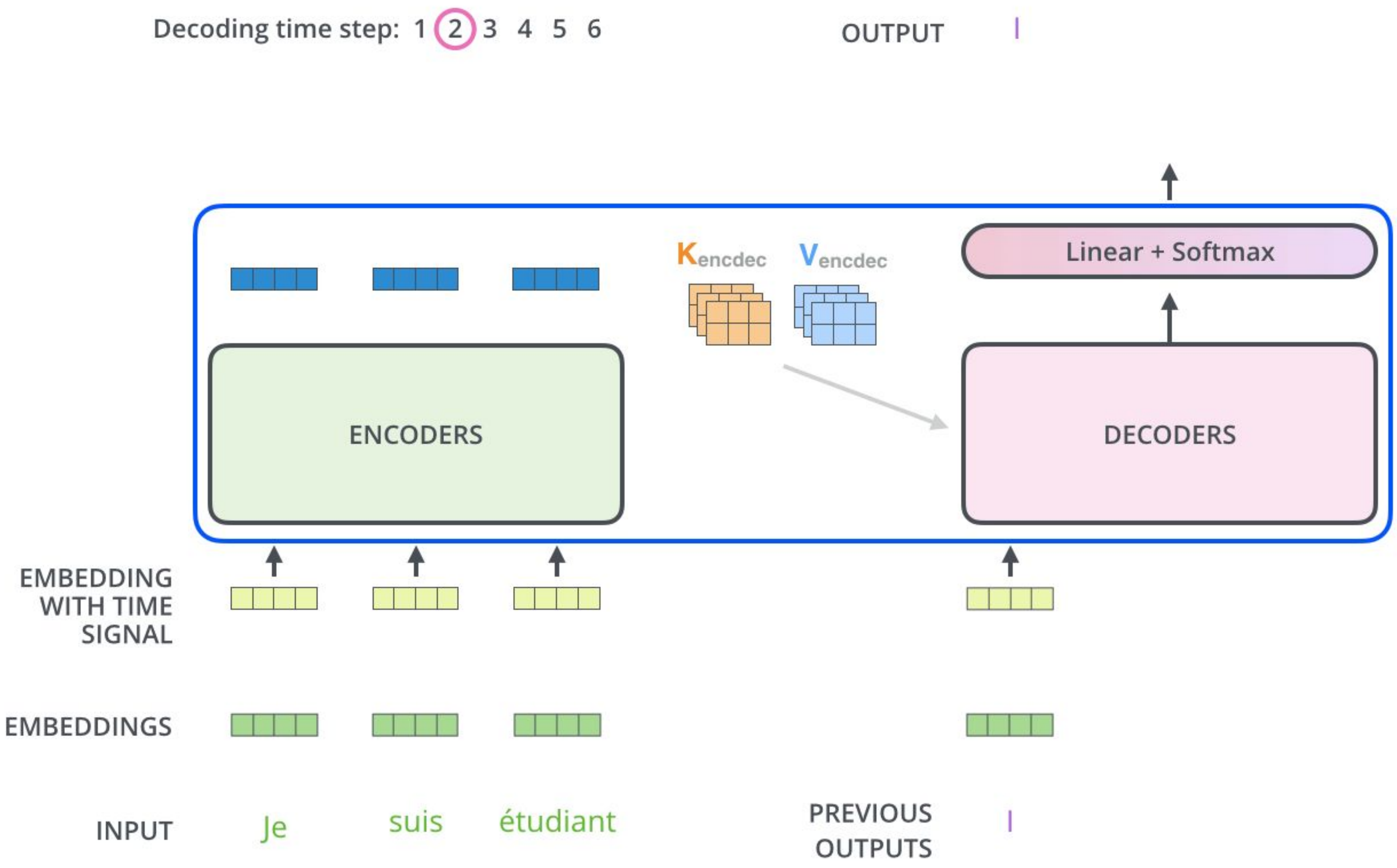


Combining all elements...

Source: Illustrated Transformer

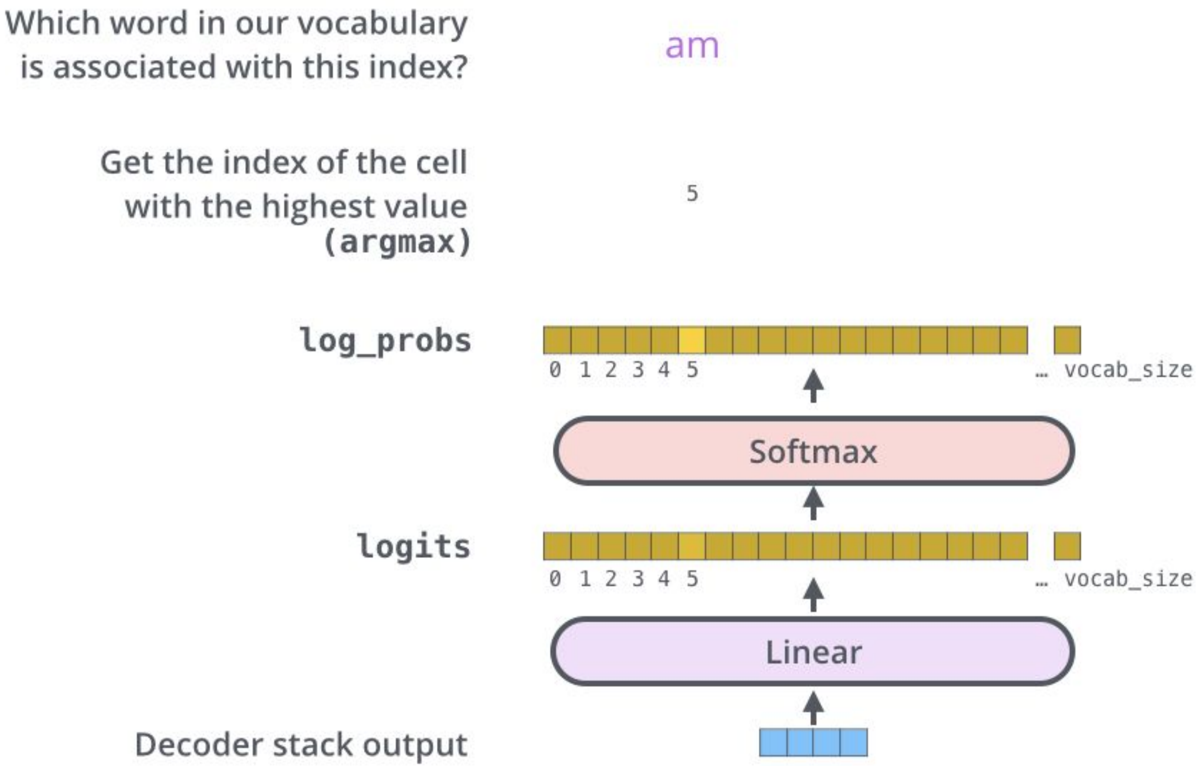


Source: Illustrated Transformer



Source: Illustrated Transformer

In case you are curious



Why Self-Attention?

Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

- Less total computational complexity per layer
- More parallelizable than existing fully autoregressive models
- Shorten the path between tokens to enable model to learn long-term dependency better
- Tang et al. (EMNLP 2018) claims that self-attention outperforms RNN/CNN as a semantic feature extractor and empirically show that it excels on word sense disambiguation task (but not subject-verb agreement over long distance!)

Experimental Results

- Achevies SOTA on 2 machine translation dataset
- Less training cost than existing SOTA models

Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [15]	23.75			
Deep-Att + PosUnk [32]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [31]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [8]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [26]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [32]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [31]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [8]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.0**	$2.3 \cdot 10^{19}$	

Model Variation Study

- Attention key size is important
- More heads doesn't necessary mean better performance
- Learned positional embedding is not better than sinusoidal positional encoding

Table 3: Variations on the Transformer architecture. Unlisted values are identical to those of the base model. All metrics are on the English-to-German translation development set, newstest2013. Listed perplexities are per-wordpiece, according to our byte-pair encoding, and should not be compared to per-word perplexities.

	N	d_{model}	d_{ff}	h	d_k	d_v	P_{drop}	ϵ_{ls}	train steps	PPL (dev)	BLEU (dev)	params $\times 10^6$
base	6	512	2048	8	64	64	0.1	0.1	100K	4.92	25.8	65
(A)				1	512	512				5.29	24.9	
				4	128	128				5.00	25.5	
				16	32	32				4.91	25.8	
				32	16	16				5.01	25.4	
(B)					16					5.16	25.1	58
					32					5.01	25.4	60
(C)	2									6.11	23.7	36
	4									5.19	25.3	50
	8									4.88	25.5	80
		256			32	32				5.75	24.5	28
		1024			128	128				4.66	26.0	168
			1024							5.12	25.4	53
			4096							4.75	26.2	90
(D)							0.0			5.77	24.6	
							0.2			4.95	25.5	
								0.0		4.67	25.3	
								0.2		5.47	25.7	
(E)	positional embedding instead of sinusoids									4.92	25.7	
big	6	1024	4096	16			0.3		300K	**4.33**	**26.4**	213

Conclusion & Limitation

- Introduces a groundbreaking new model that is solely based on attention
- Faster and better than existing models
- Still not fully parallelized due to decoder being auto-regressive
- Context is fixed length and cannot attend long-term dependency
- Stacking more encoders/decoders might lead to vanishing gradients

References

- “*Attention Is All You Need*,” Vaswani et al. NeurIPS 2017
- “*Why Self-Attention? A Targeted Evaluation of Neural Machine Translation Architectures*,” Tang et al. EMNLP 2018
- “*The Annotated Transformer*,” https://nlp.seas.harvard.edu/2018/04/03/attention.html
- “*The Illustrated Transformer*,” http://jalammar.github.io/illustrated-transformer/
- “*Positional Embedding*,” https://kazemnejad.com/blog/transformer_architecture_positional_encoding/
- “*BertViz*,” https://github.com/jessevig/bertviz

Thank you! &
Discussion Time :)