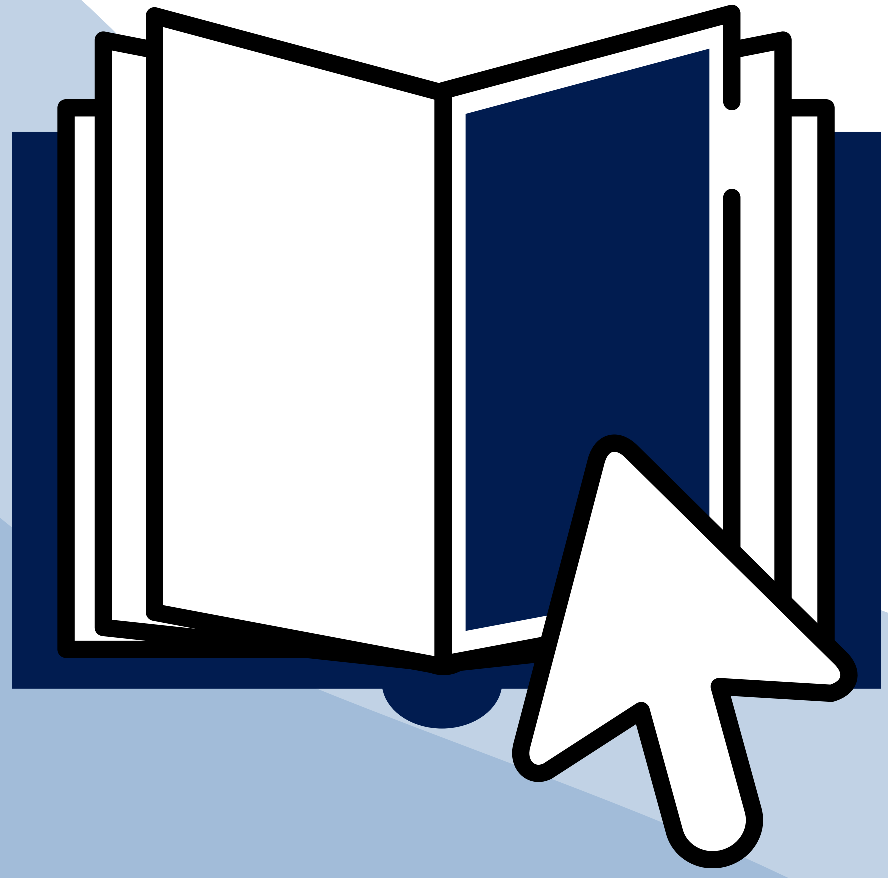
Transformers

Attention is All you need

By
Shivam Mishra
Anup Joseph
Akansha Goyal



Agenda

Background and Motivation

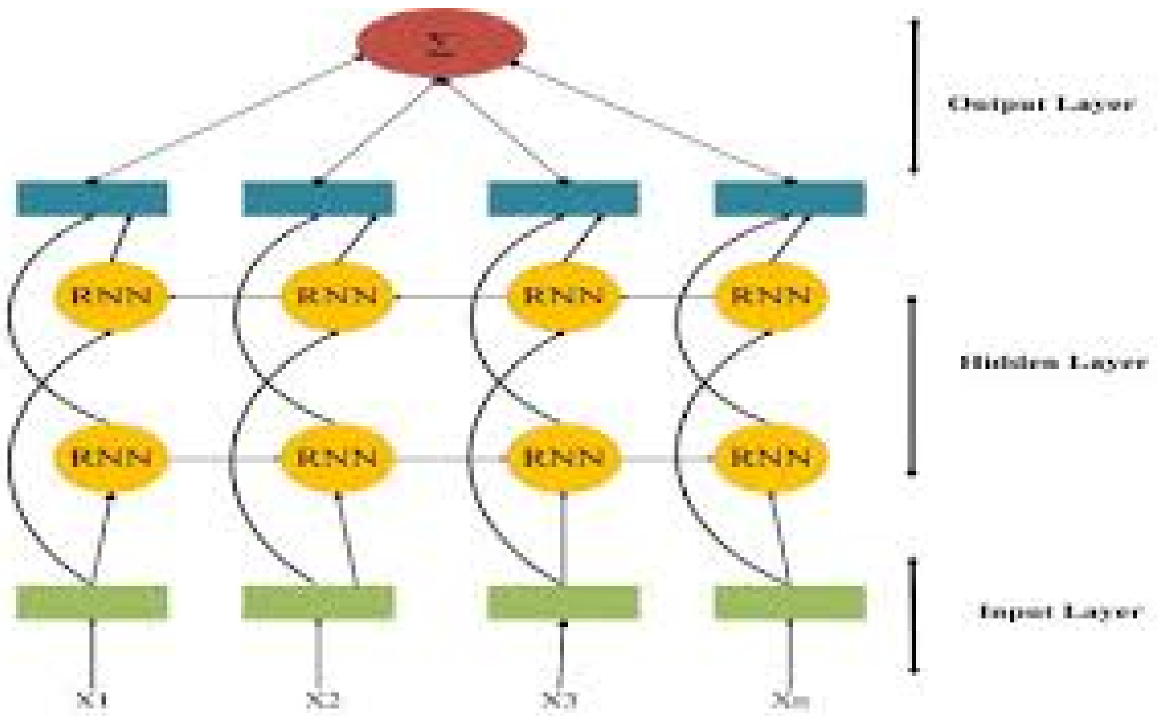
Architecture

Main ideas

Training

Results

Background



Former techniques are not good at parallelization

- RNN,LSTM techniques have been SOTA in sequence modeling.
- The sequential nature prevents parallelization especially when sequence length is long

Challenge of Computational Cost

- Difficulty in learning dependencies between distant positions

Motivation

- The Transformer architecture is aimed at the problem of sequence transduction where the goal is to design a single framework to handle as many sequences as possible.
- It reduces the number of sequential operations to relate two symbols from input/output sequences to a constant number of operations.

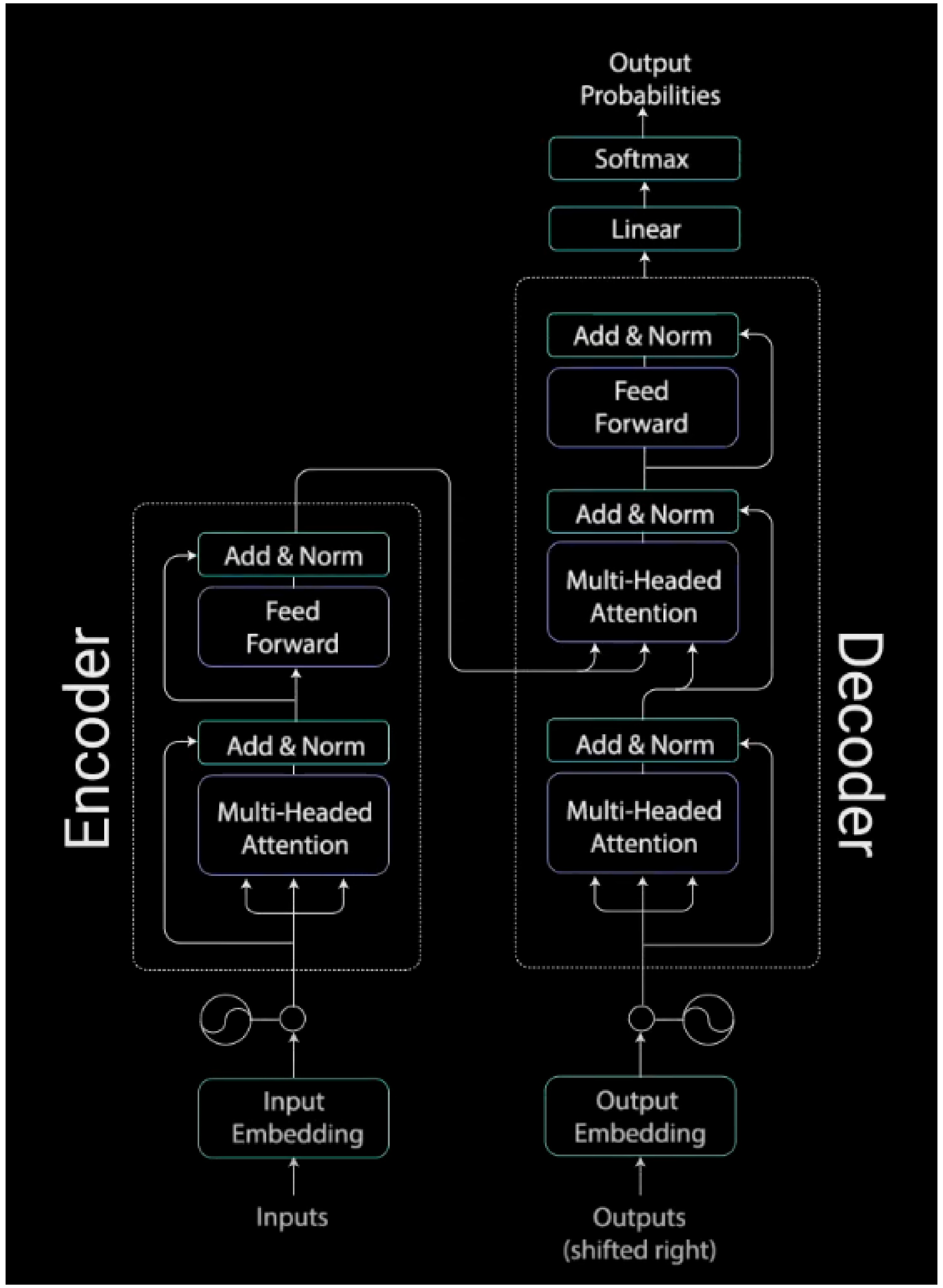
Motivation

- Transformer achieve parallelization by replacing recurrence with attention and encoding the symbol position in sequence,which in turn leads to significantly shorter training time.

- It eliminates not only recurrence but also convolution in favor of applying self-attention , additionally providing more space for parallelization

Transformers Architecture

It's complicated!



Self-Attention

Multi-headed Self-Attention

Positional Encoding

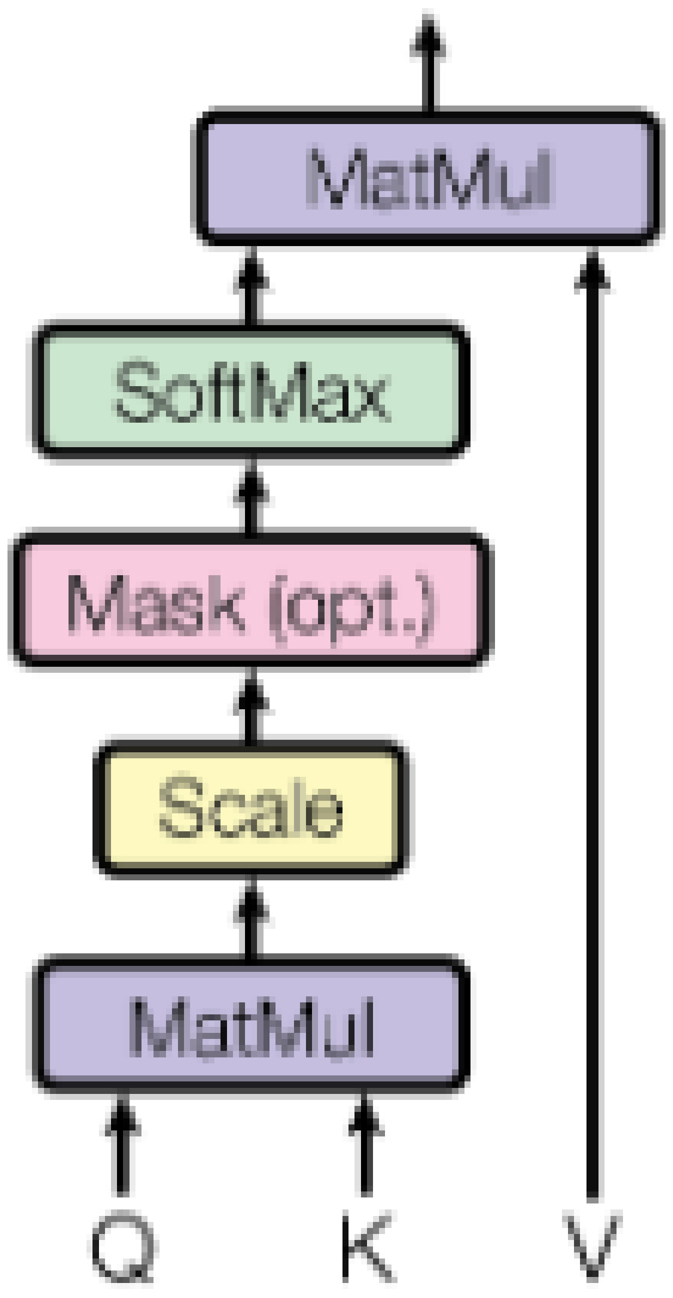
Residuals

Self-Attention

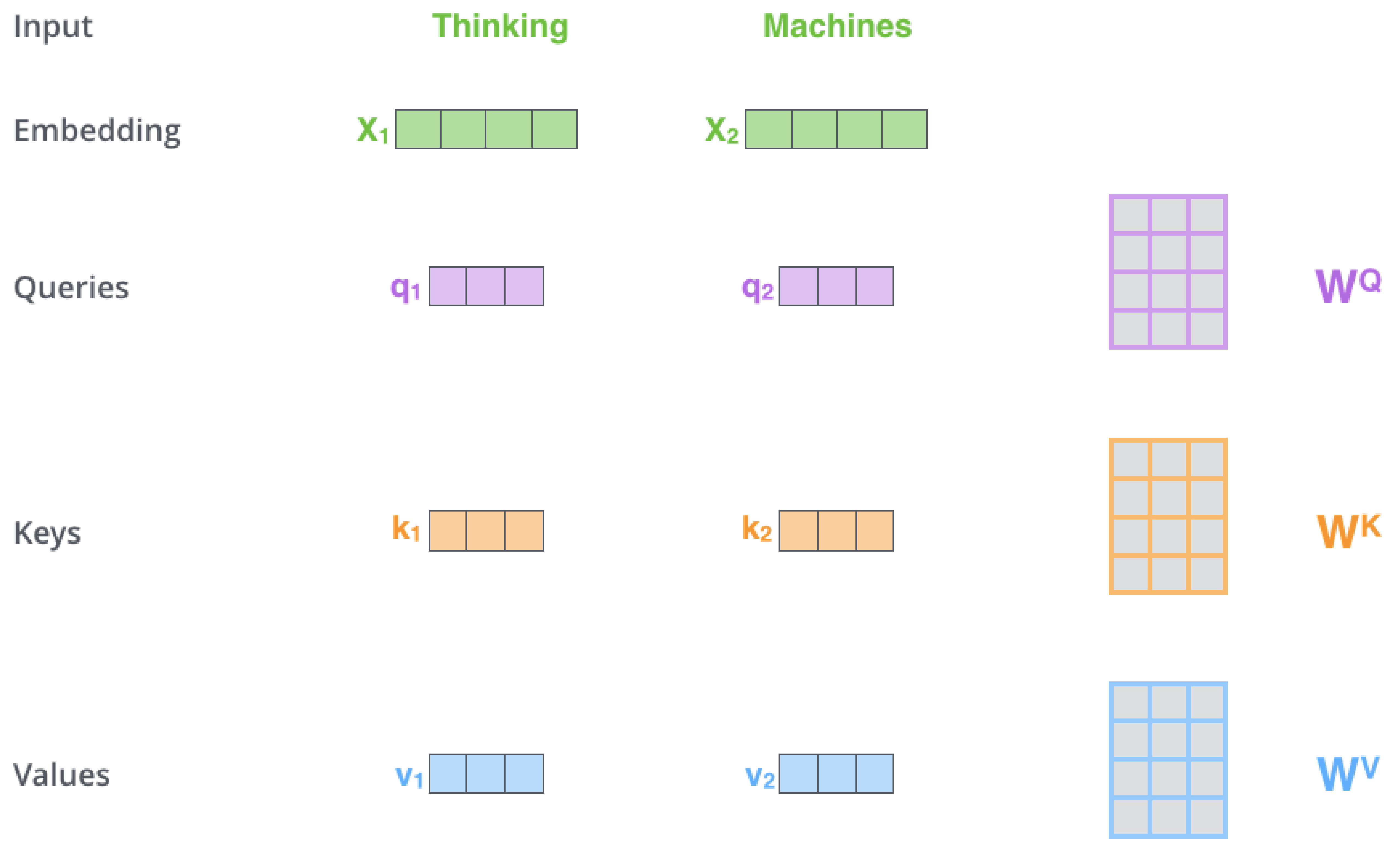
Self- Attention is the mechanism used by Transformers to associate one word with another

I am a student

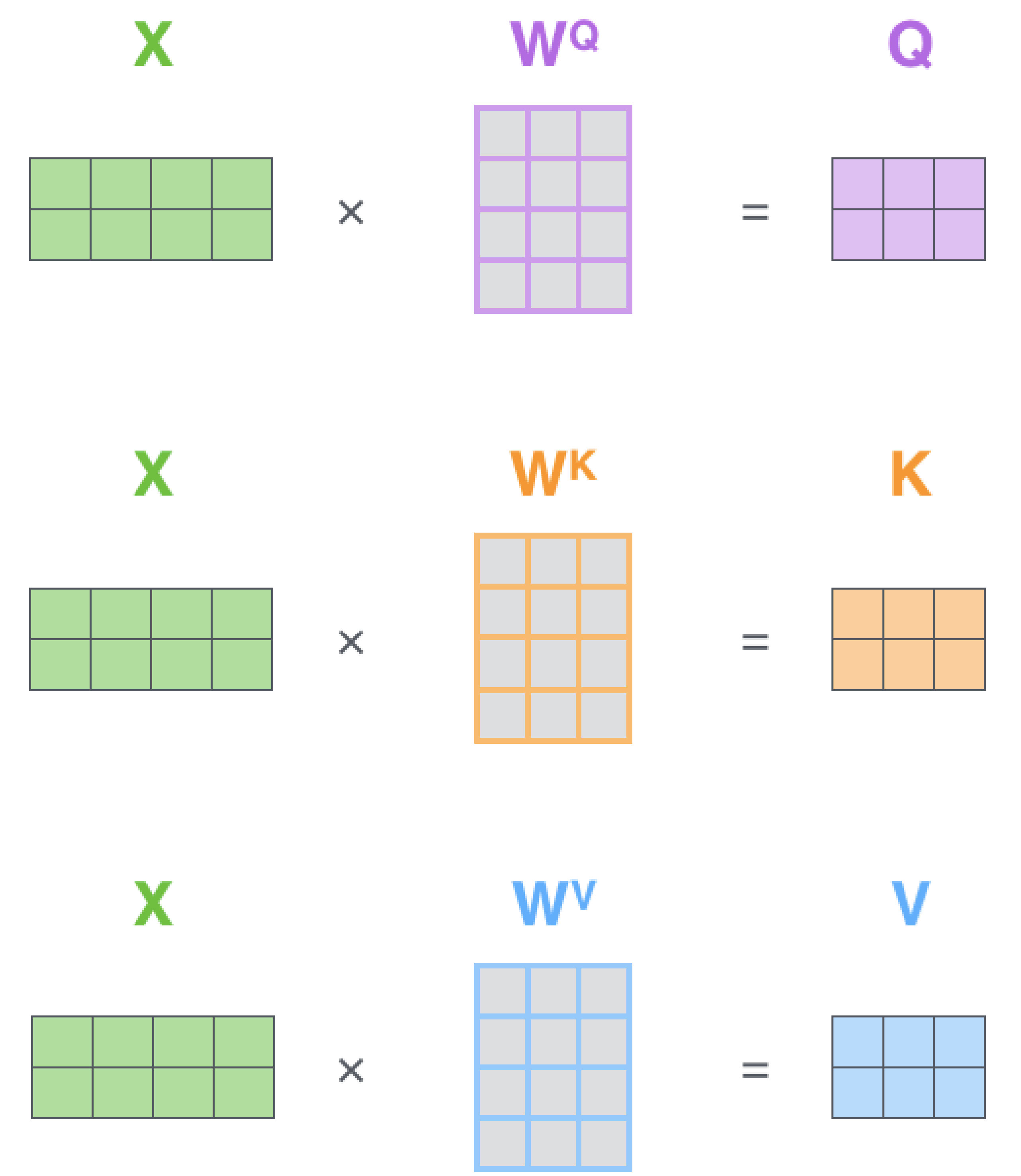
In this sentence self-attention allows the model to relate I to student

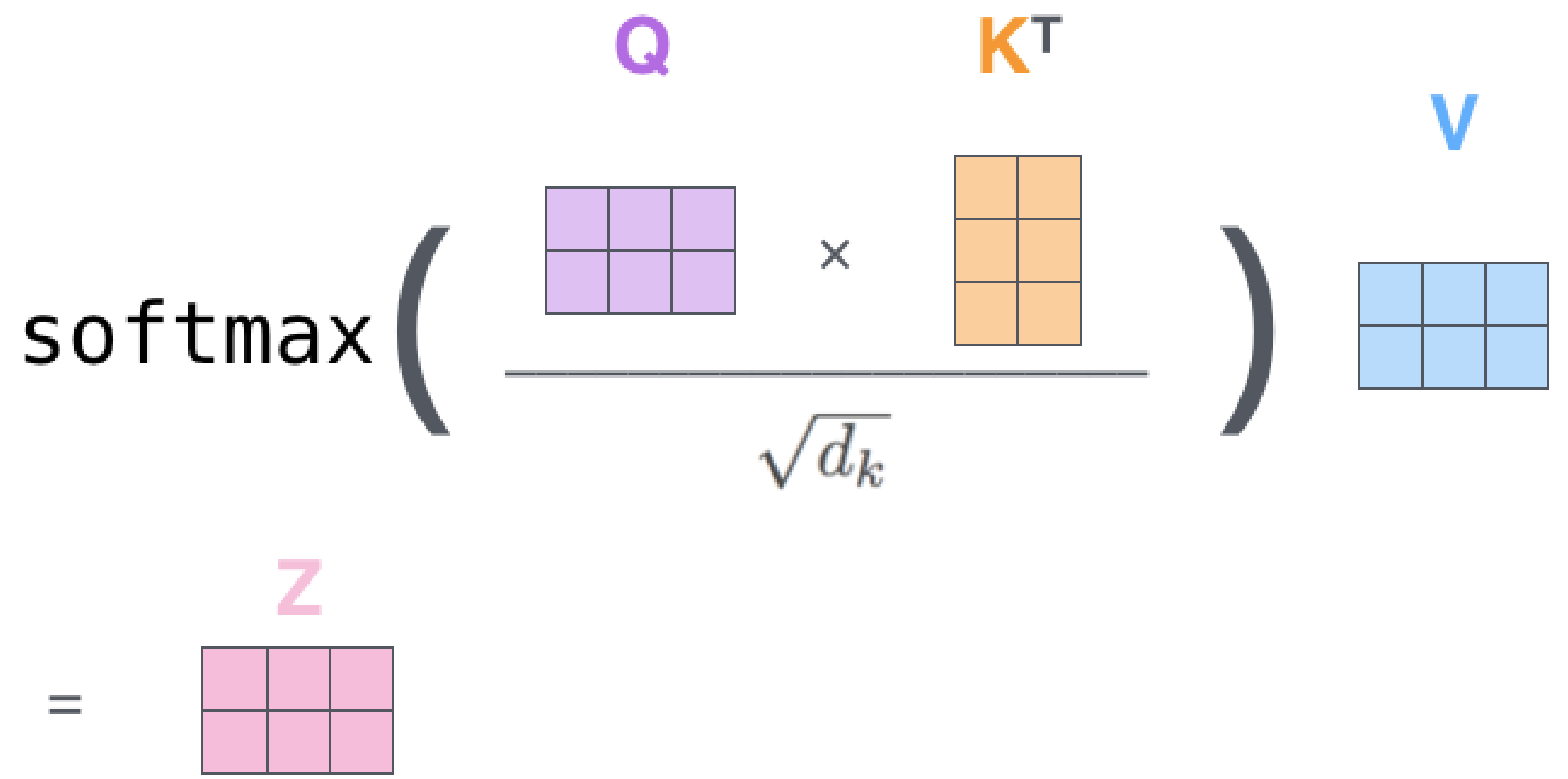


Self-Attention Calculation

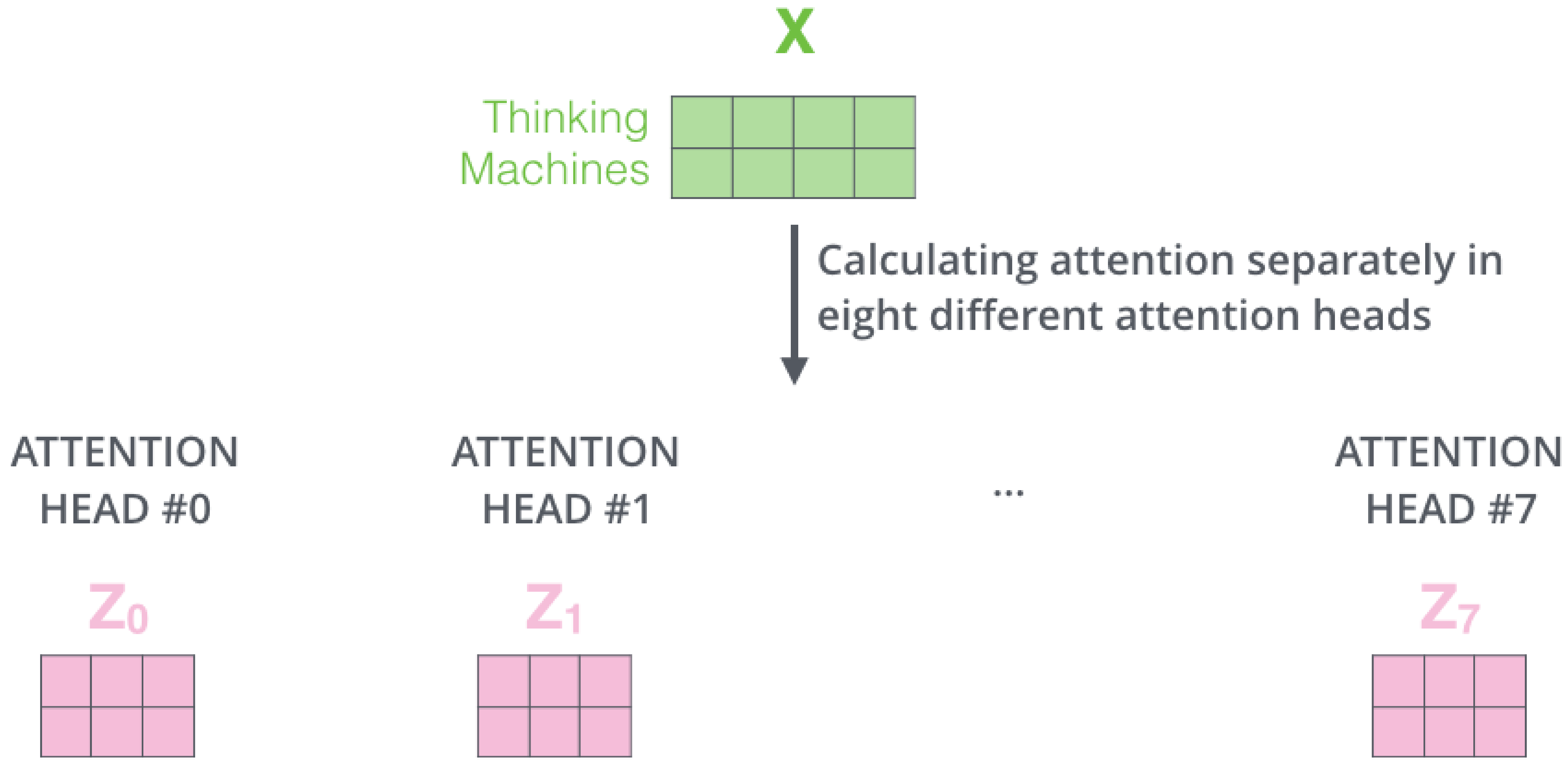


Self-Attention Calculation



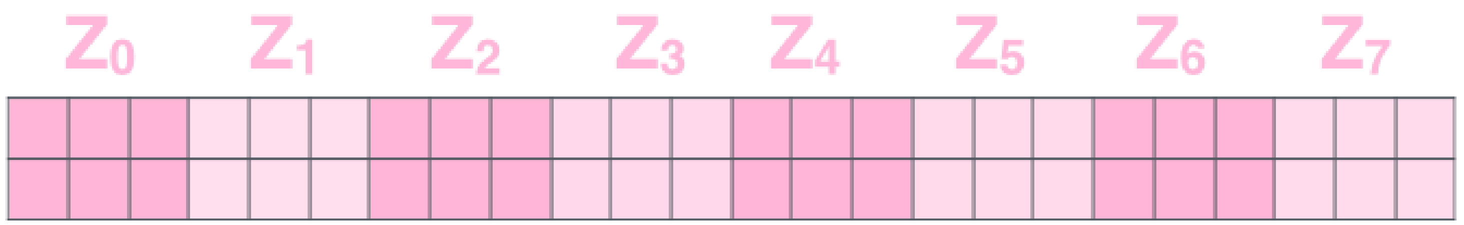


Multi-headed Self-Attention



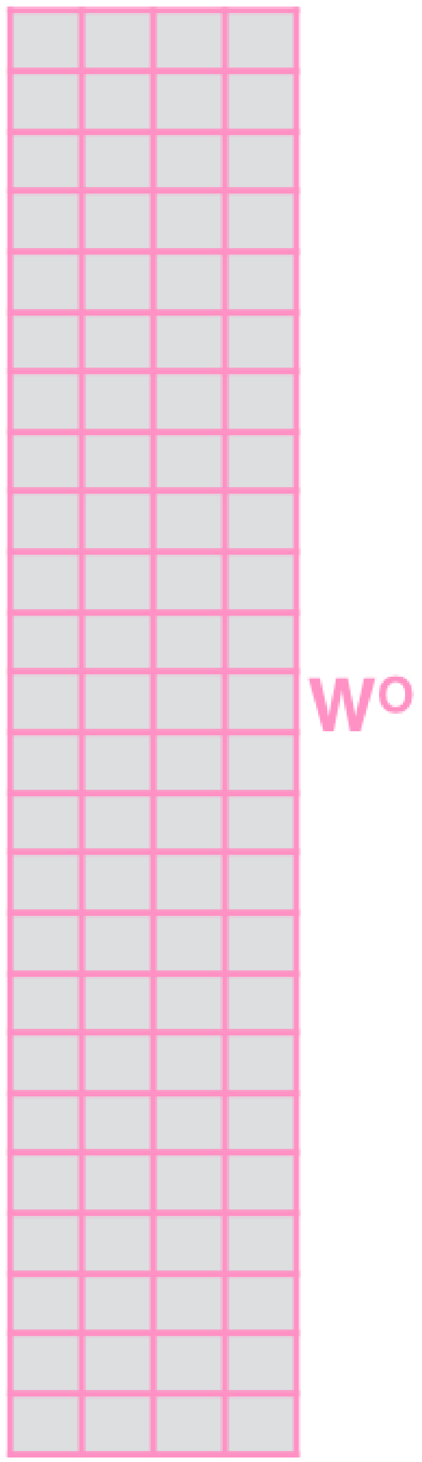
Multi-headed Self-Attention

1) Concatenate all the attention heads



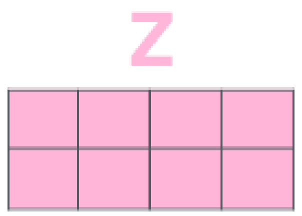
2) Multiply with a weight matrix W^O that was trained jointly with the model

X

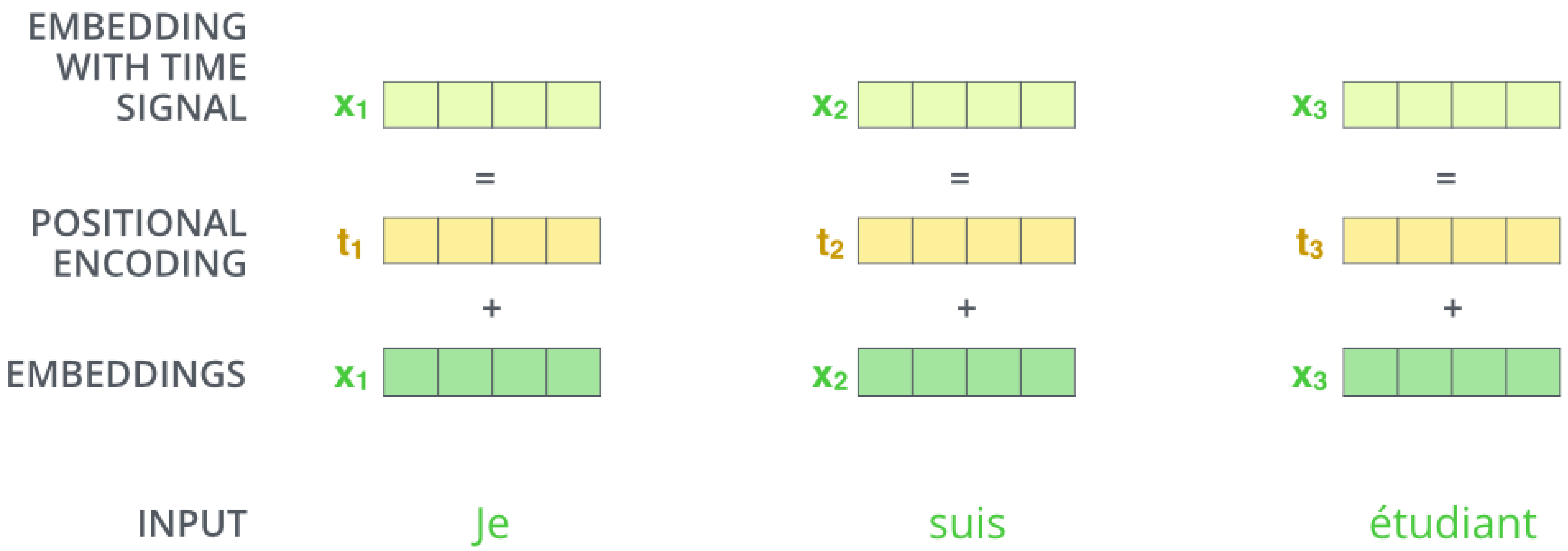


3) The result would be the Z matrix that captures information from all the attention heads. We can send this forward to the FFNN

=

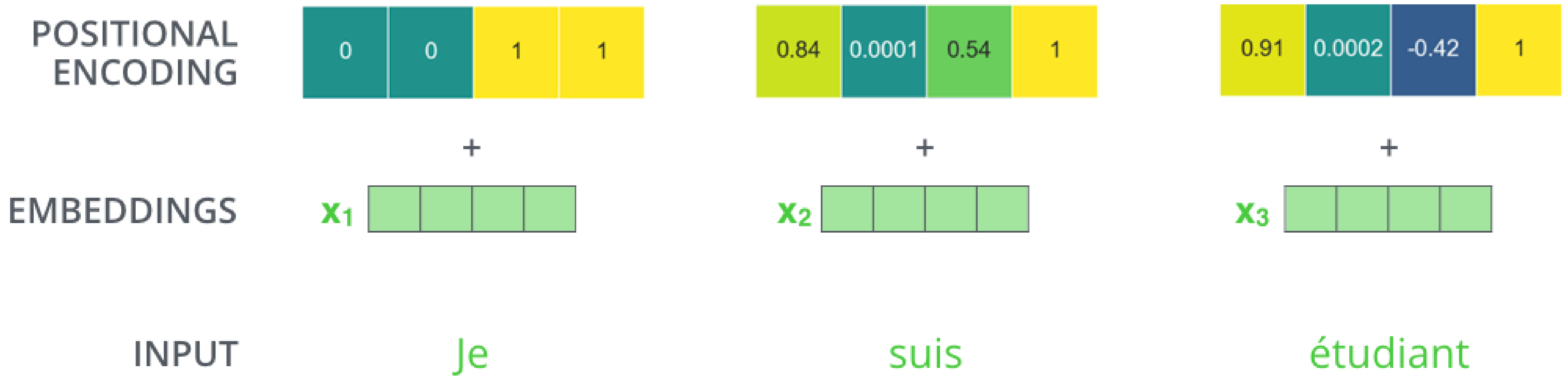


Positional encoding is the Transformers technique to account for the order of words in the sequence

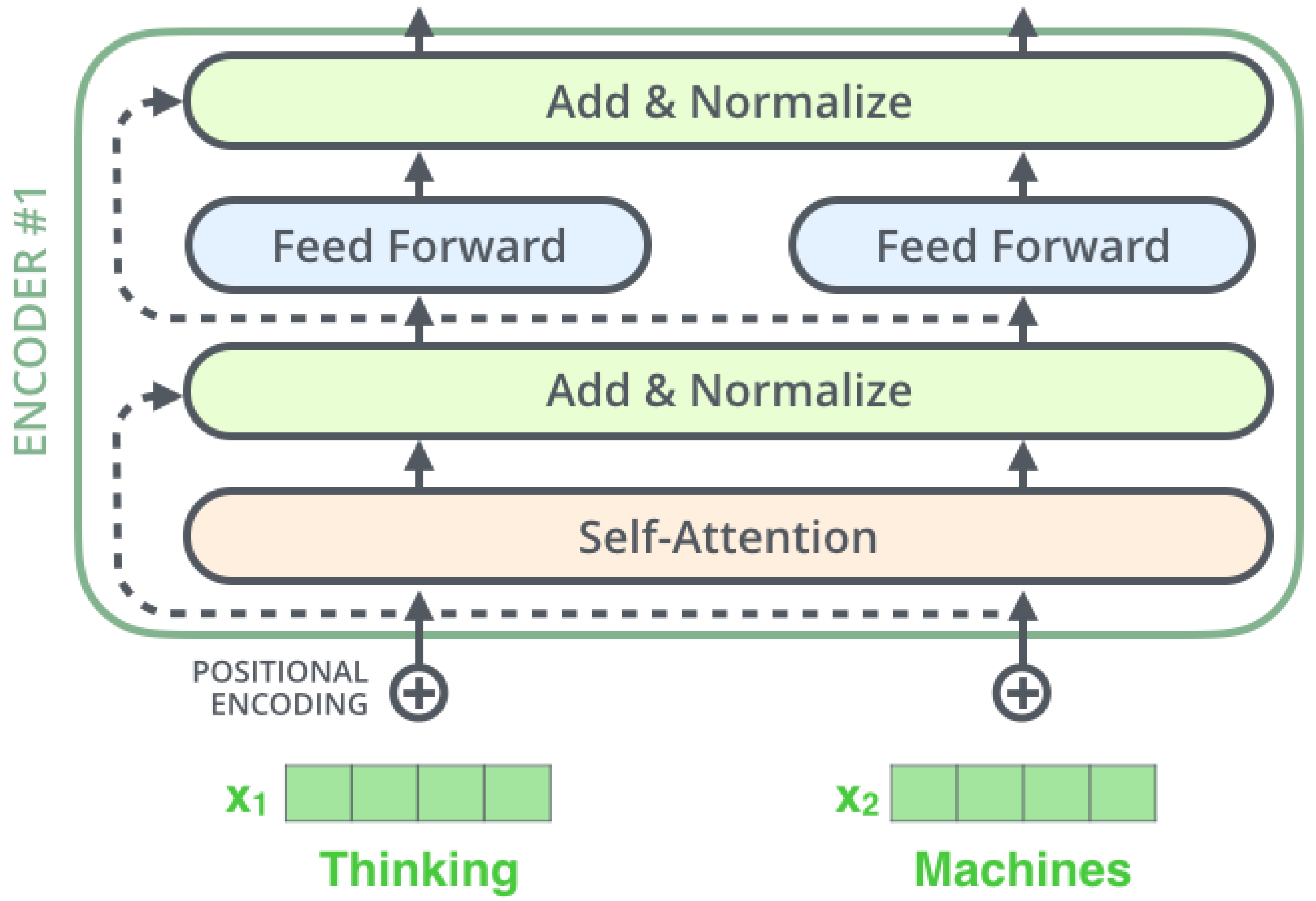


Positional Encoding

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

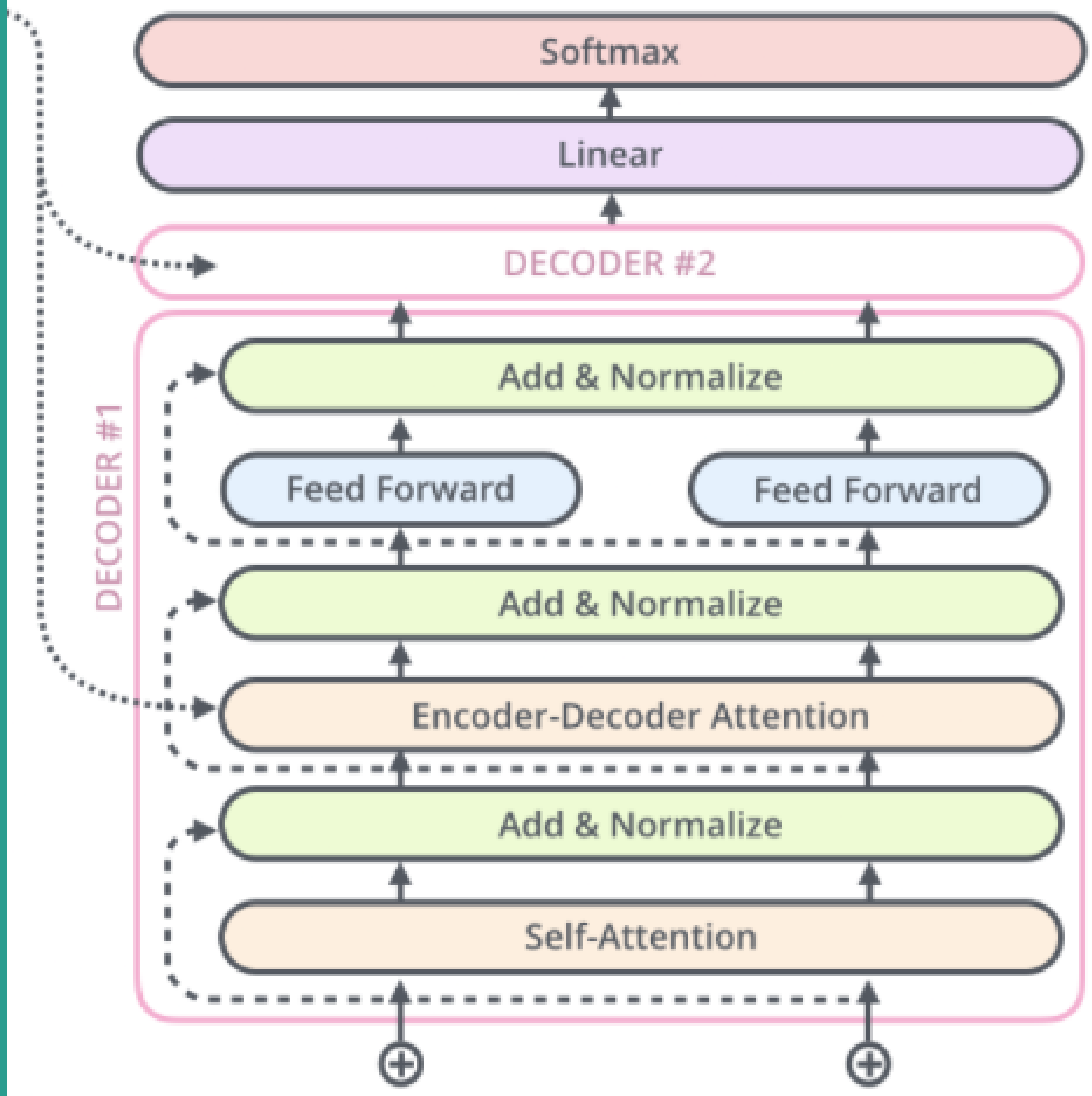


Residual Connections

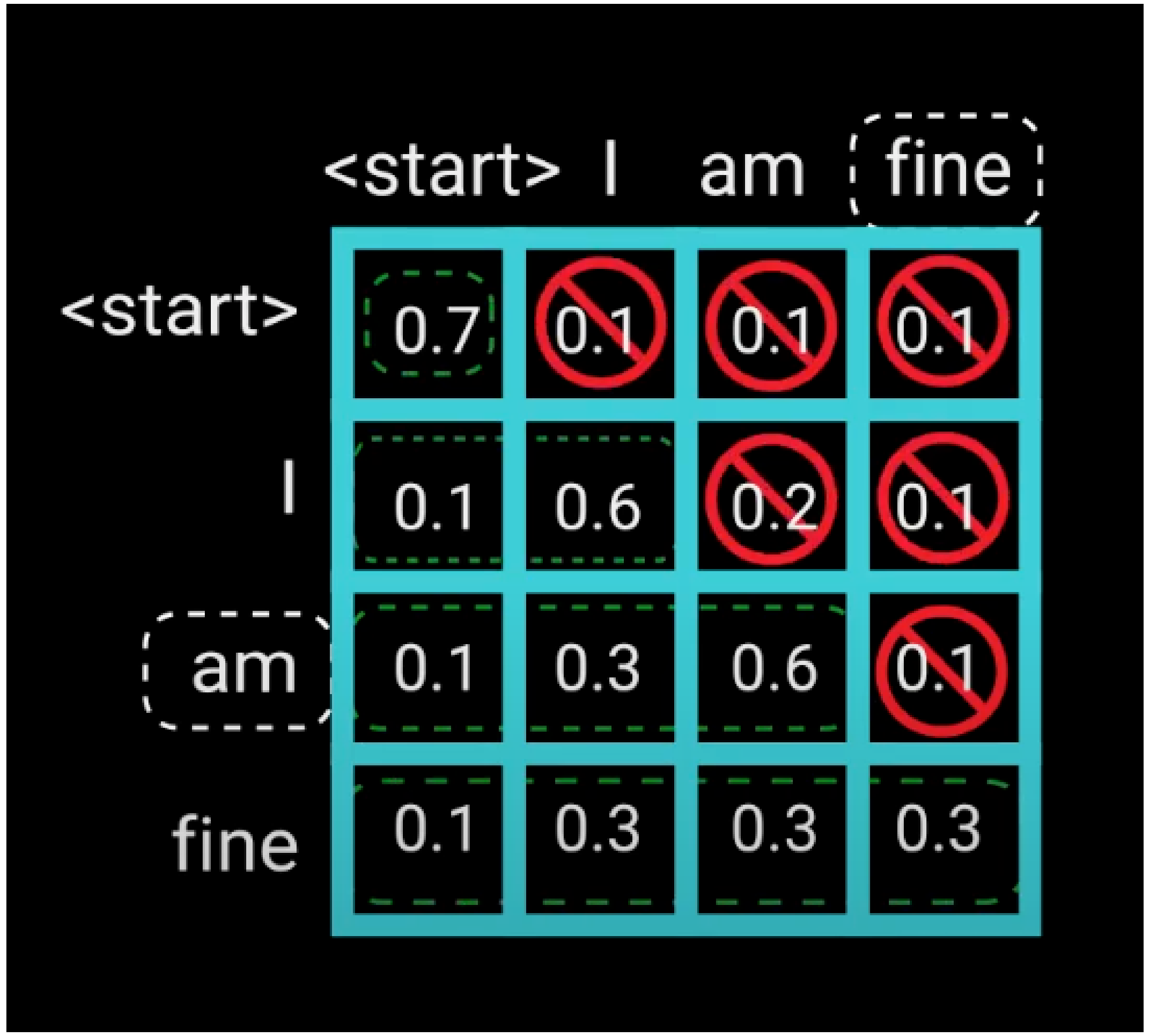


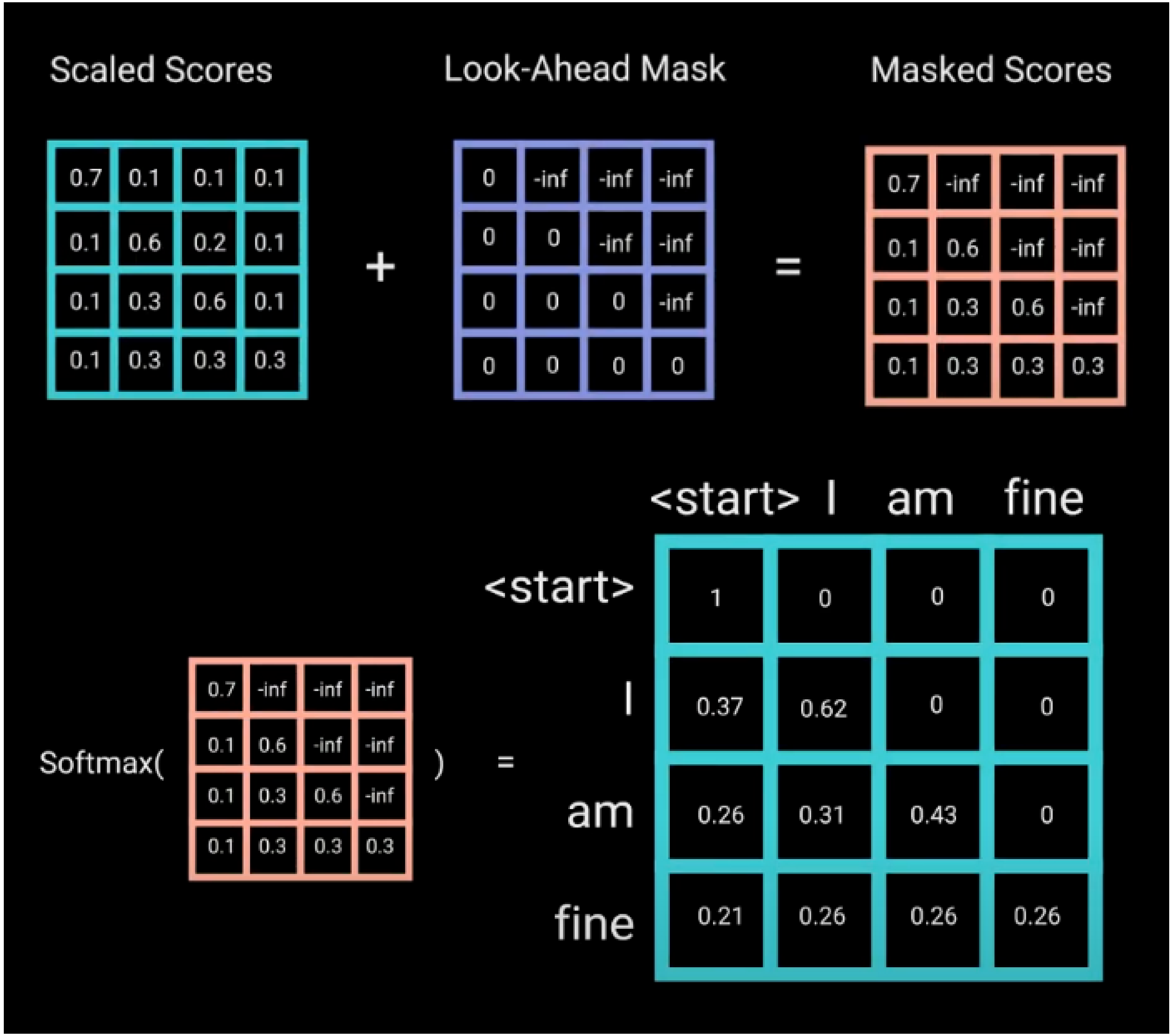
Decoders

Almost the same as encoders with a twist



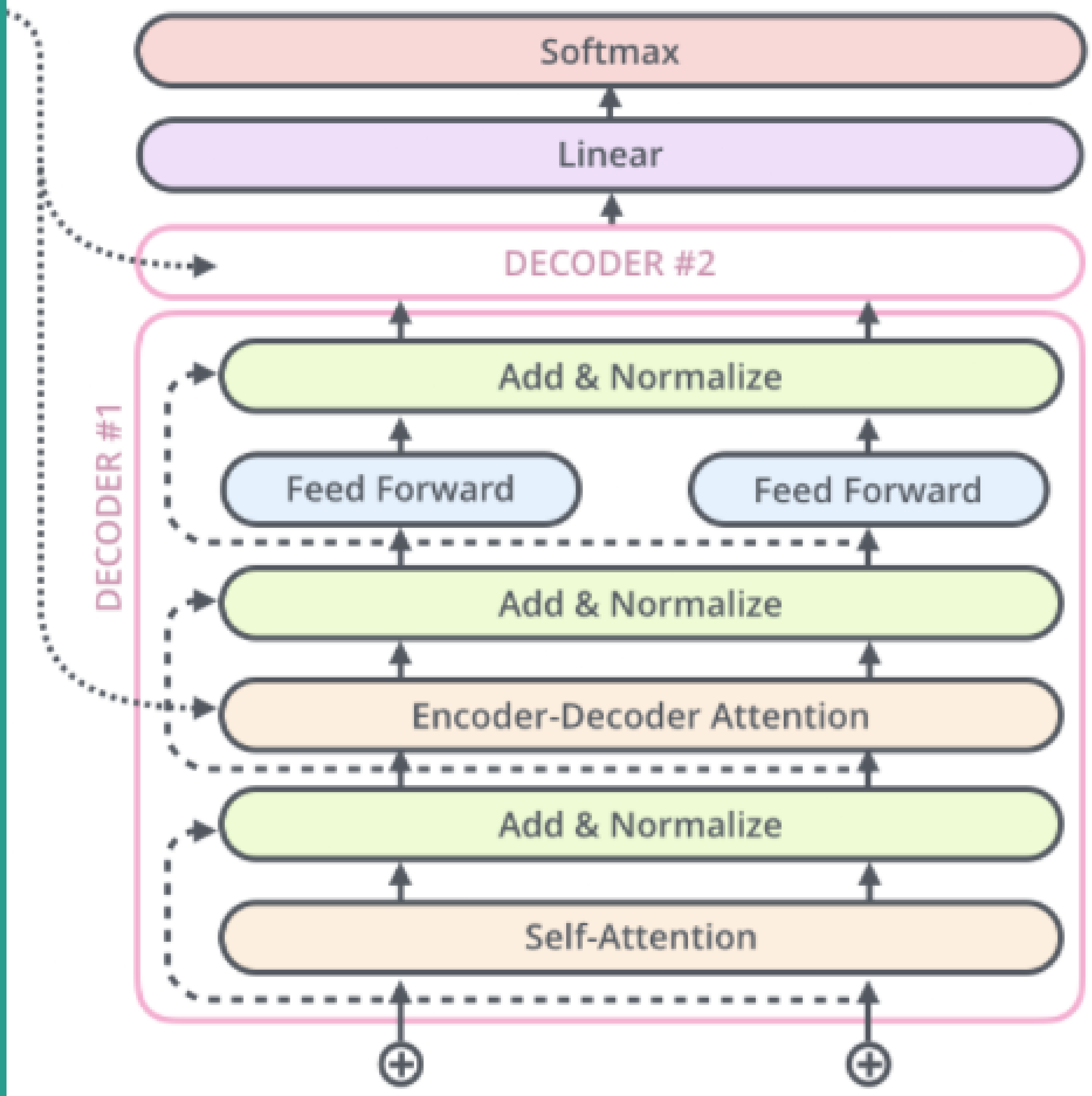
Lookahead Mask



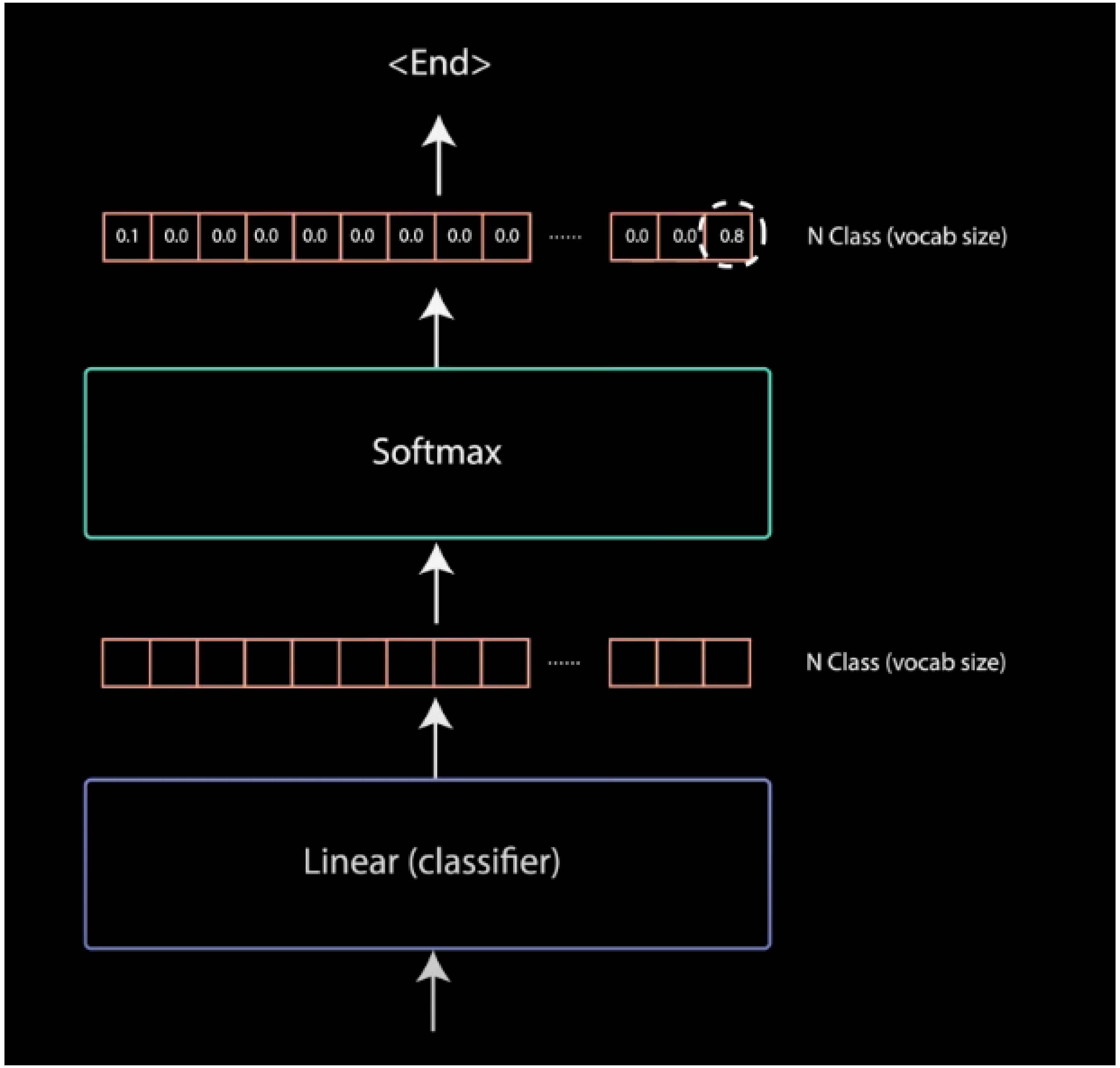


Decoders

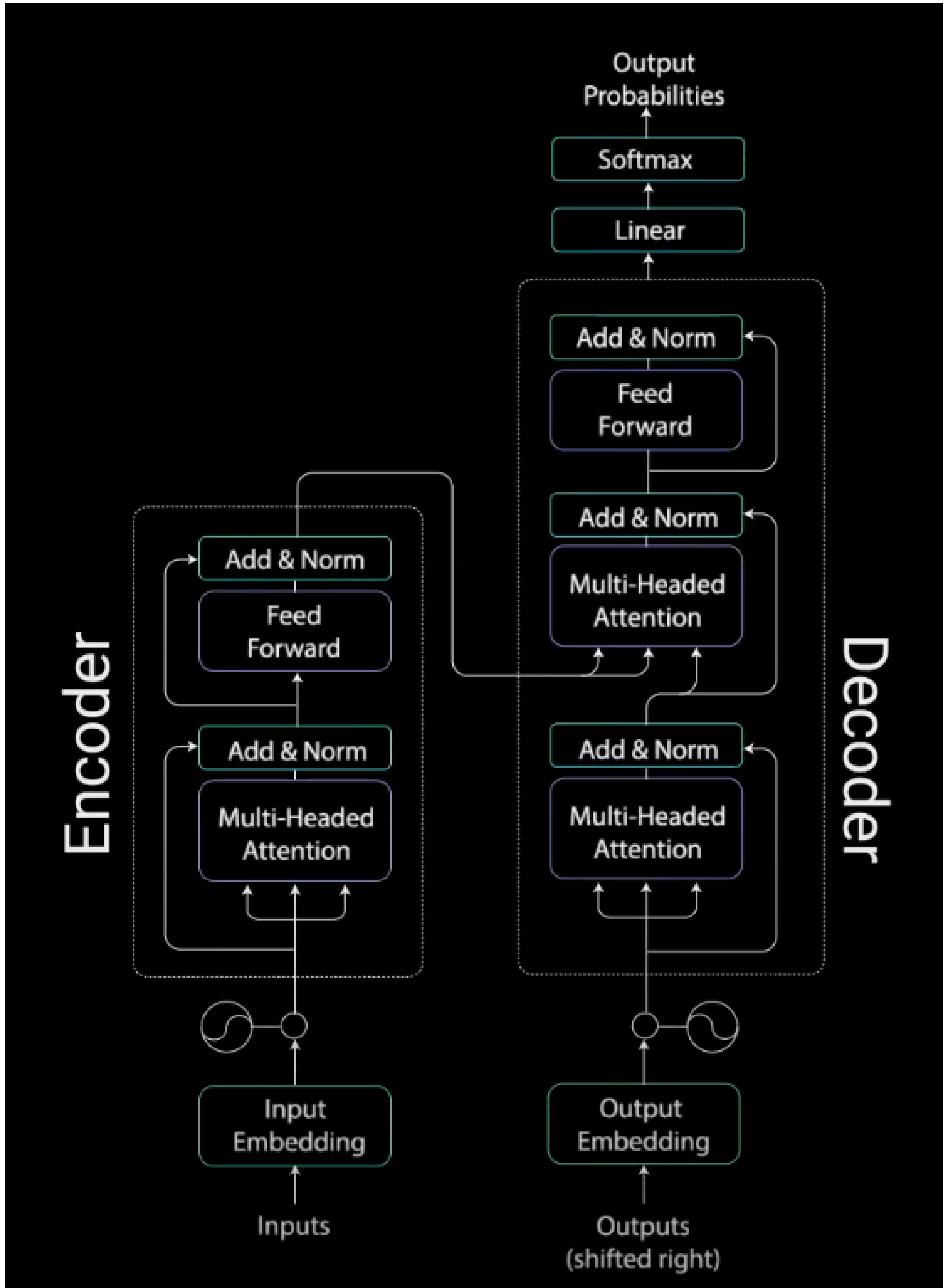
Coming back to decoders



Final Classification



Final Architecture



Training

- The objective was the translation task, performed for translations from English to French and German.
- Sentence Pairs - 4.5M, 36M ; Tokens - 37 K, 32 K
- Sentence pairs batched together by approximate sequence length
- Base model trained for 100,00 steps and big models trained for 300,000 steps.
- Varied learning rate; optimizer parameters are nearly the same as default ones .
- Two residual dropouts, and label smoothing

Results

- For the English - German translation task, the big transformer model outperformed every other model , establishing a new BLEU score.
- The base models surpassed other previous models and ensembles with relatively less training cost.
- For the English - French translation task, the big model outperformed previous models at 1/4th the relative training cost, with dropout 0.1 (rather than 0.3).
- Hyperparameters such as beam size (4) , penalty length (0.6) are chosen after experimenting on the development set.
- Maximum output length during inference is ip_length + 50, but terminate early when possible.

Model Variations, Performance

Varied the base model to evaluate importance of different components.

Single head attention is 0.9 BLEU worst than best setting, quality drops off with too many heads.

	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	

Parser	Training	WSJ 23 F1
Vinyals & Kaiser el al. (2014) [37]	WSJ only, discriminative	88.3
Petrov et al. (2006) [29]	WSJ only, discriminative	90.4
Zhu et al. (2013) [40]	WSJ only, discriminative	90.4
Dyer et al. (2016) [8]	WSJ only, discriminative	91.7
Transformer (4 layers)	WSJ only, discriminative	91.3
Zhu et al. (2013) [40]	semi-supervised	91.3
Huang & Harper (2009) [14]	semi-supervised	91.3
McClosky et al. (2006) [26]	semi-supervised	92.1
Vinyals & Kaiser el al. (2014) [37]	semi-supervised	92.1
Transformer (4 layers)	semi-supervised	92.7
Luong et al. (2015) [23]	multi-task	93.0
Dyer et al. (2016) [8]	generative	93.3

References

[1]. Attention is All you need

https://arxiv.org/abs/1706.03762

[2]. The Illustrated Transformer – Jay Alammar

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

[3]. Illustrated Guide to Transformers– Step by Step Explanation – Micheal Phi

https://towardsdatascience.com/illustrated-guide-to-transformers-step-by-step-explanation-f74876522bc0


T1 H4 A1 N1 K5 S1