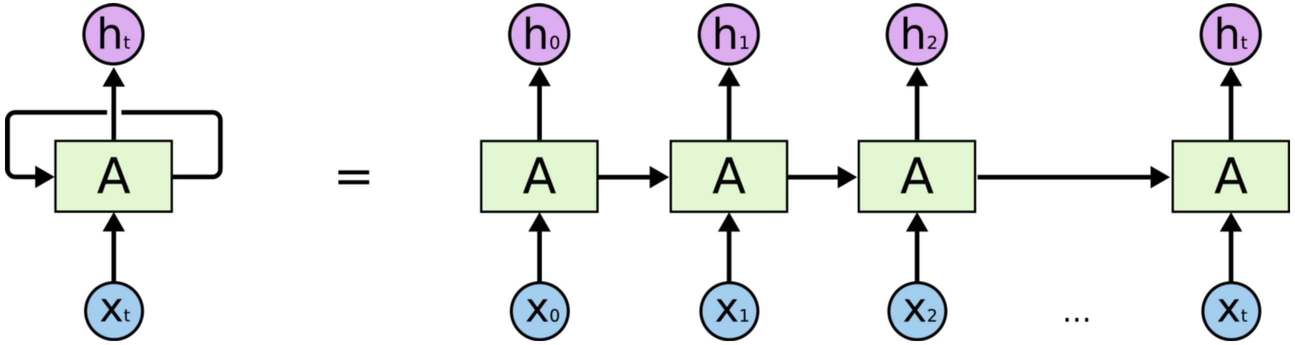
Transformer Models

김윤식

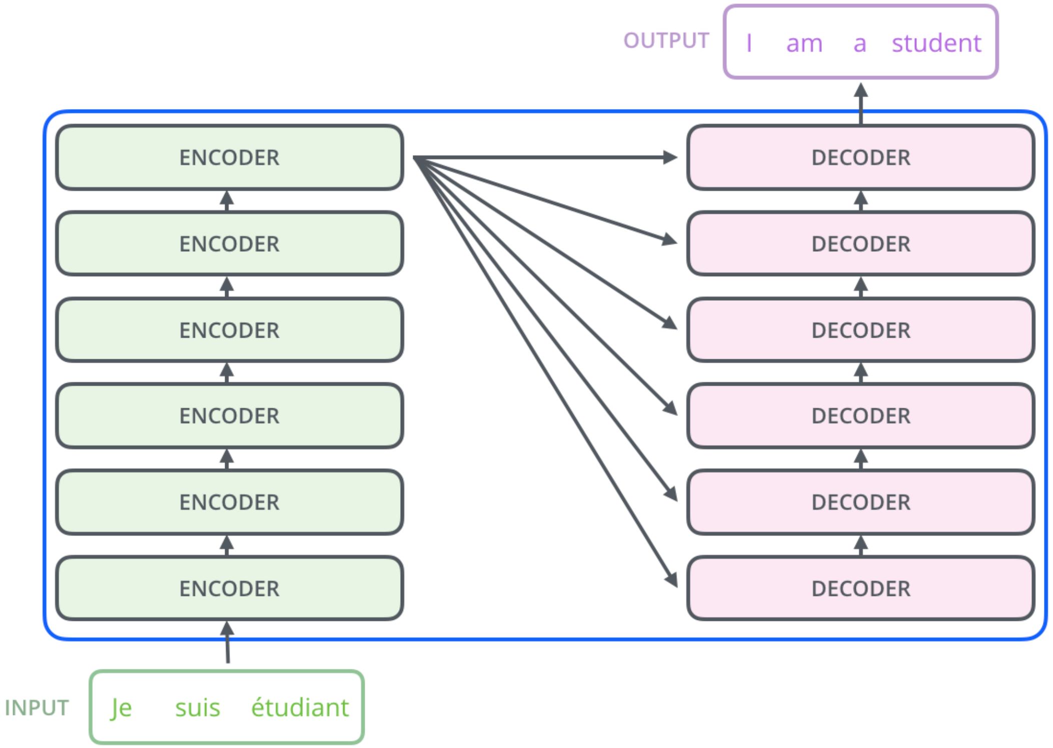
Introduction

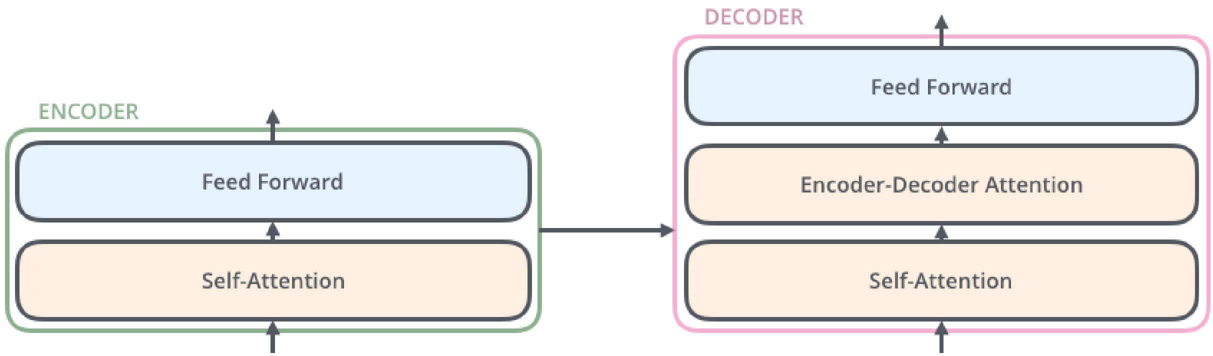
- Machine translation, Machine text generation, etc.: Seq2Seq
- Previously used RNN, LSTM, ⋯



- New approach: “Attention”

Overview





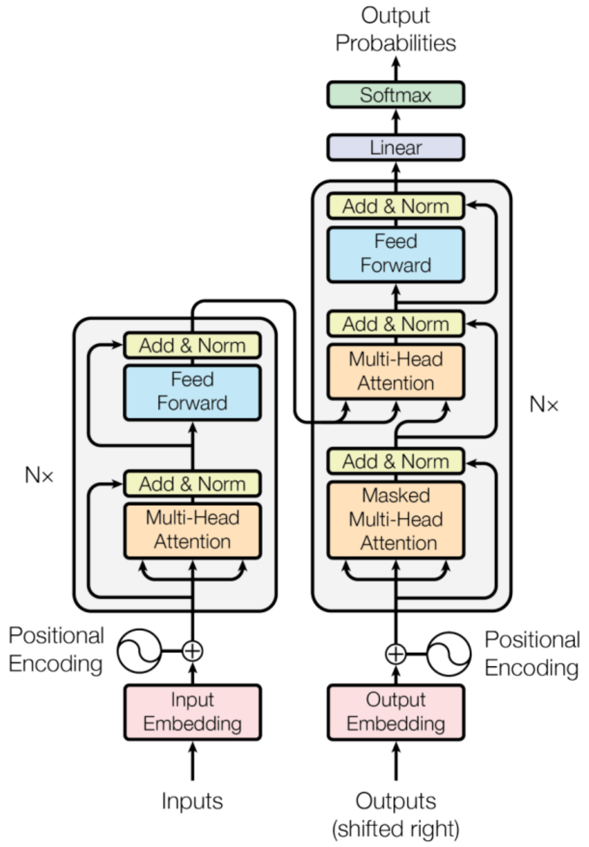
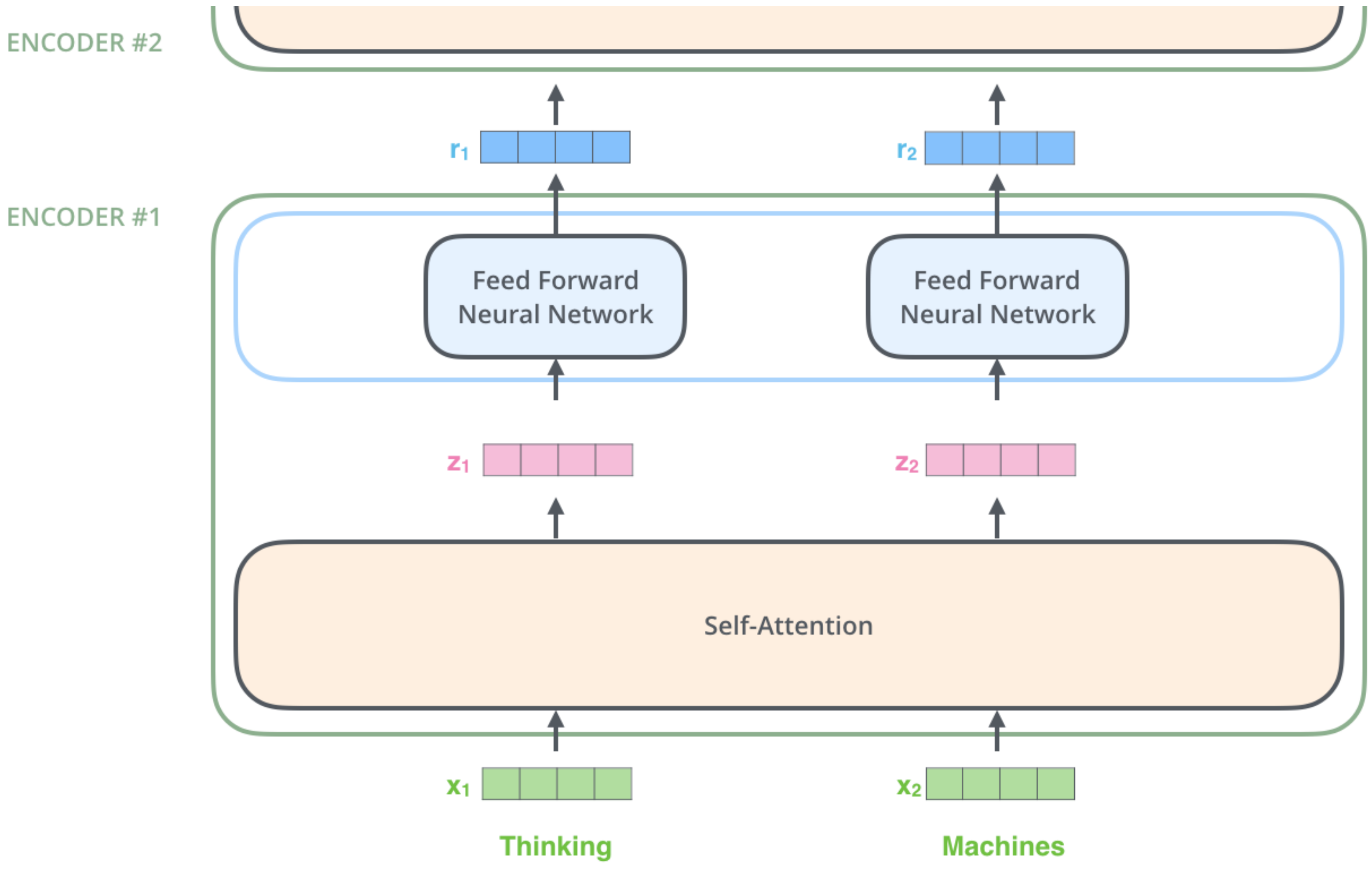
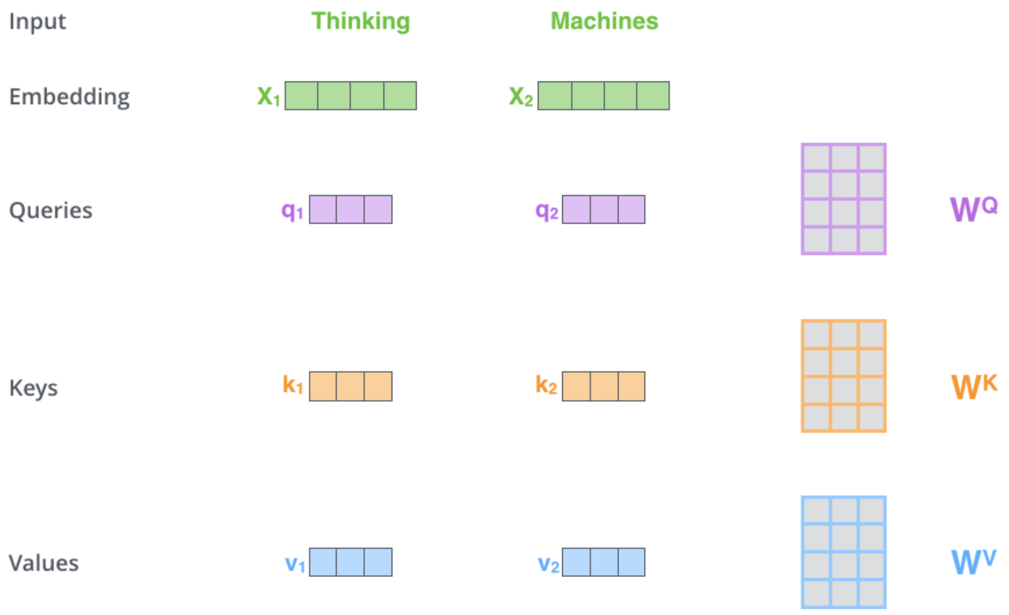


Figure 1: The Transformer - model architecture.

Encoder


ENCODER #2
r_1
r_2
ENCODER #1
Feed Forward Neural Network
Feed Forward Neural Network
z_1
z_2
Self-Attention
x_1
x_2
Thinking
Machines

Self-attention



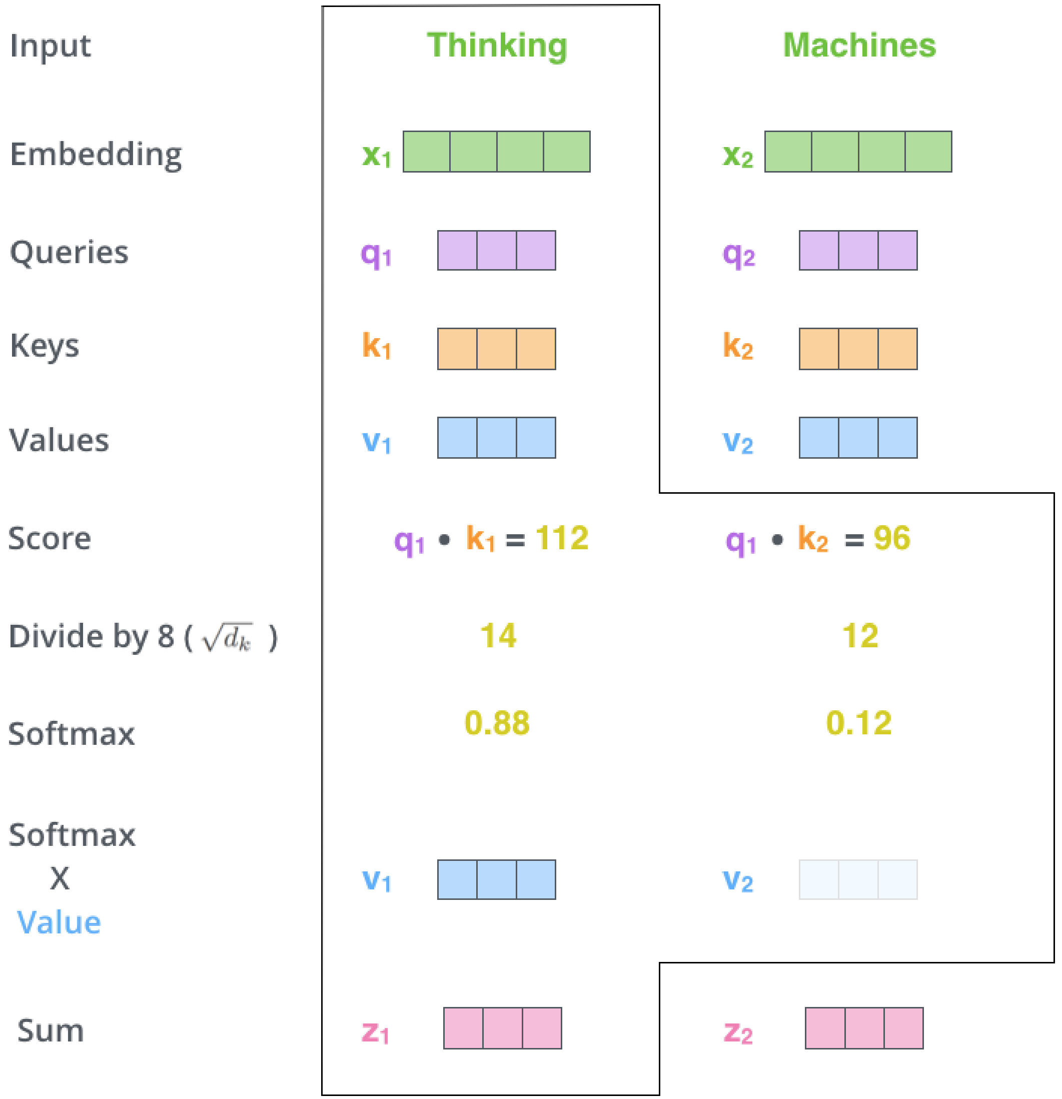
Input | Thinking | Machines

Embedding | x_1 | x_2

Queries | q_1 | q_2 | W^Q

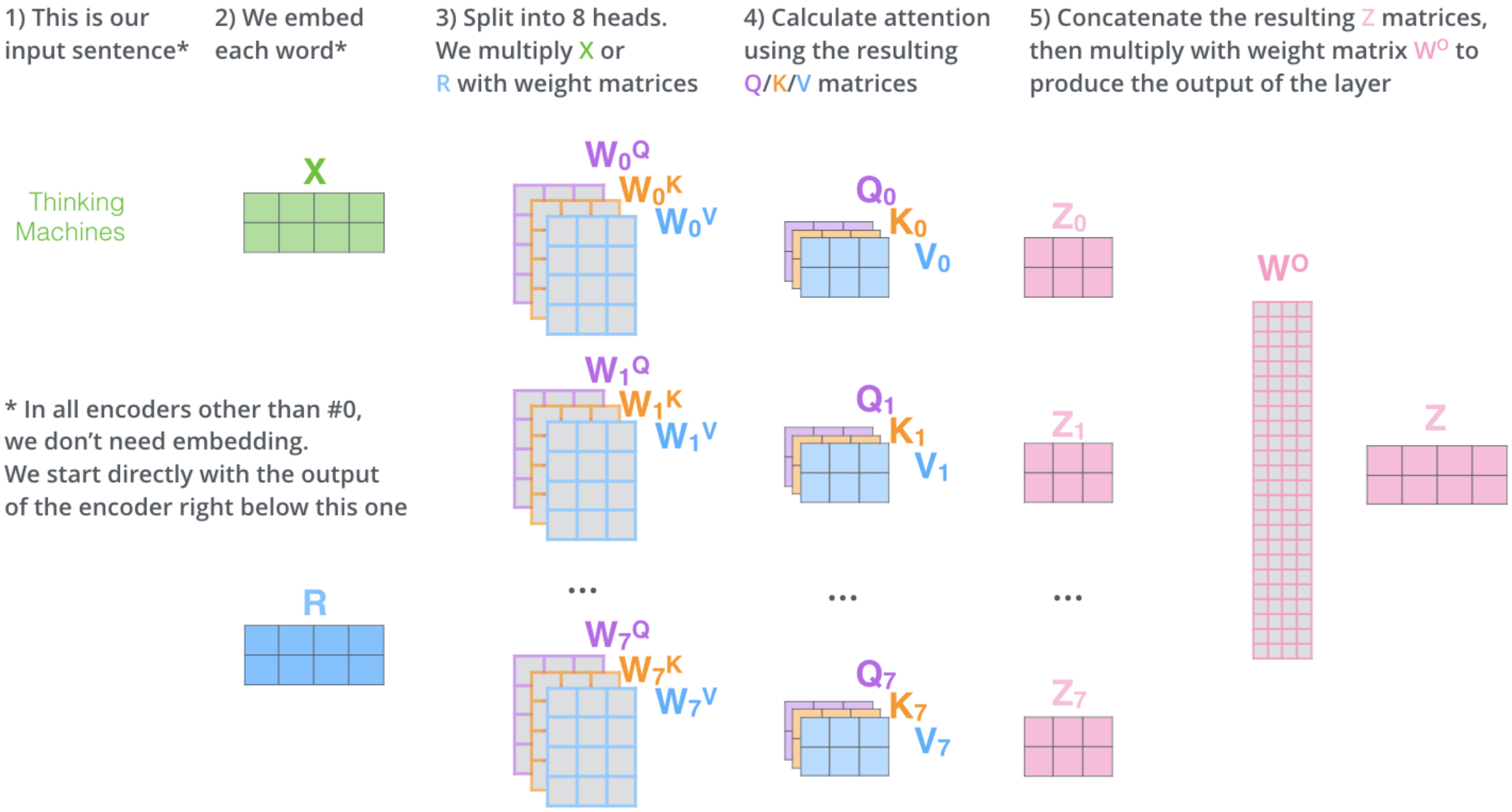
Keys | k_1 | k_2 | W^K

Values | v_1 | v_2 | W^V



	Thinking	Machines
Input	Thinking	Machines
Embedding	x_1	x_2
Queries	q_1	q_2
Keys	k_1	k_2
Values	v_1	v_2
Score	$q_1 \bullet k_1 = 112$	$q_1 \bullet k_2 = 96$
Divide by 8 ($\sqrt{d_k}$)	14	12
Softmax	0.88	0.12
Softmax X Value	v_1	v_2
Sum	z_1	z_2

Multi-headed Self-attention



1) This is our input sentence*

2) We embed each word*

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

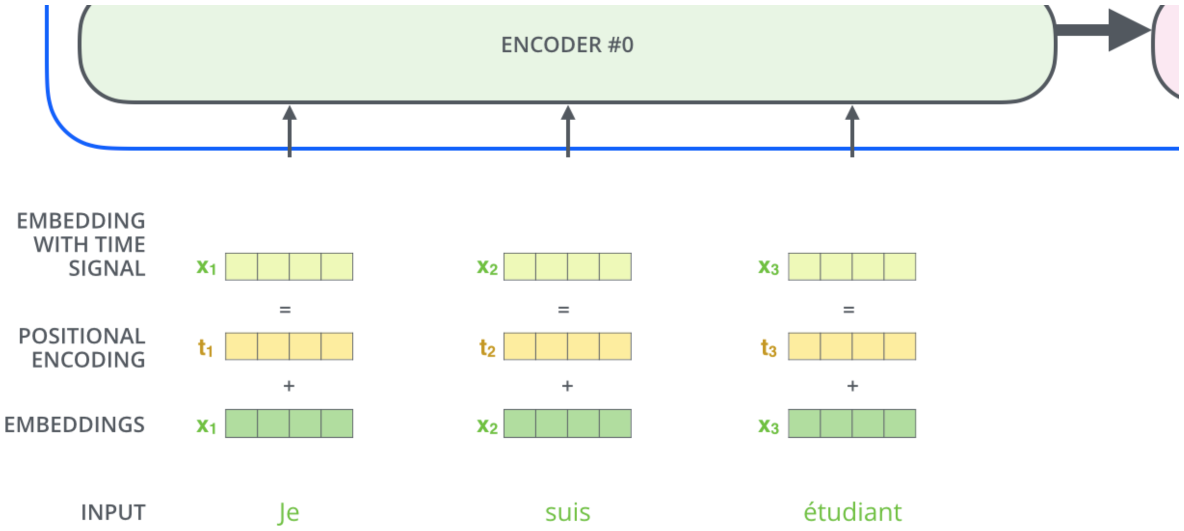
4) Calculate attention using the resulting Q/K/V matrices

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

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

Positional Encoding

- Injecting information about the positions of tokens

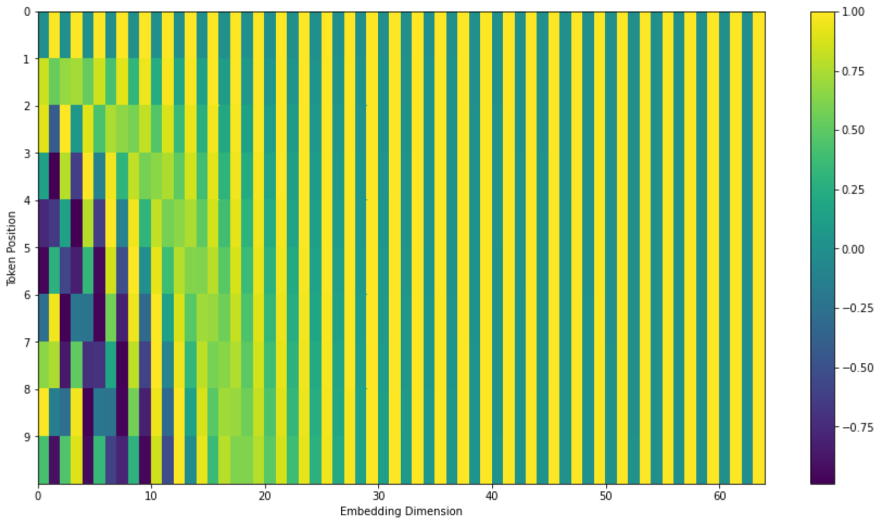


ENCODER #0

EMBEDDING WITH TIME SIGNAL	x_1	x_2	x_3
	=	=	=
POSITIONAL ENCODING	t_1	t_2	t_3
	+	+	+
EMBEDDINGS	x_1	x_2	x_3
INPUT	Je	suis	étudiant

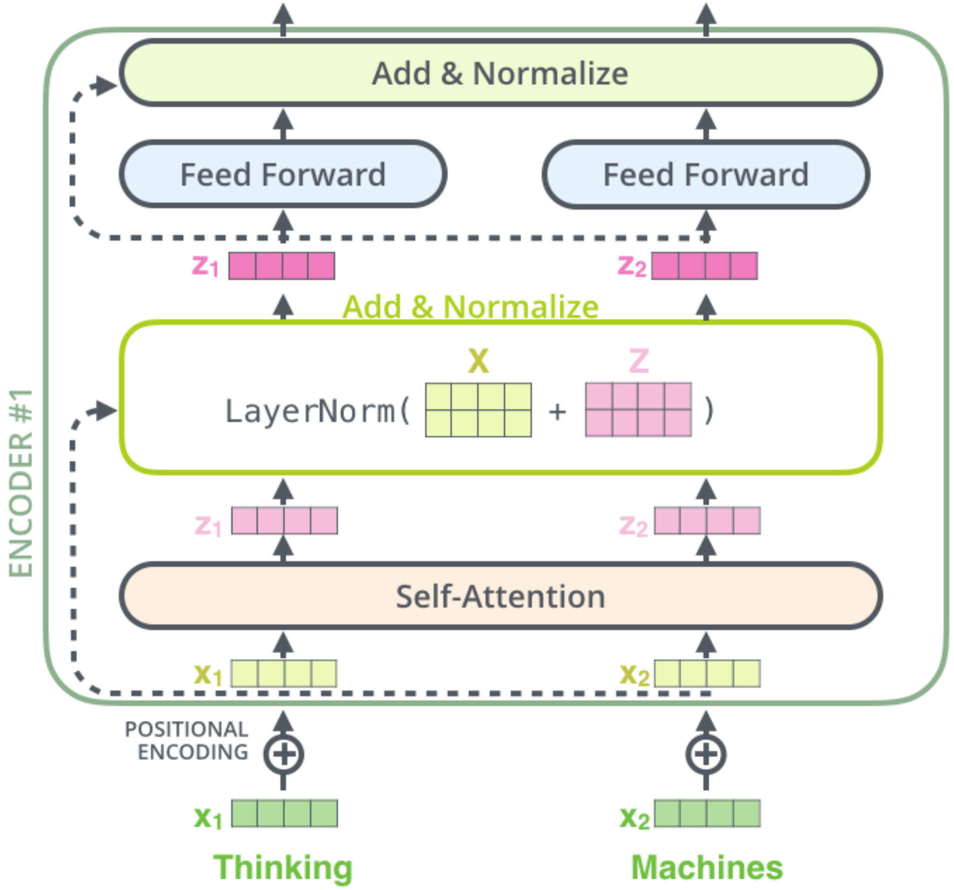
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$

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

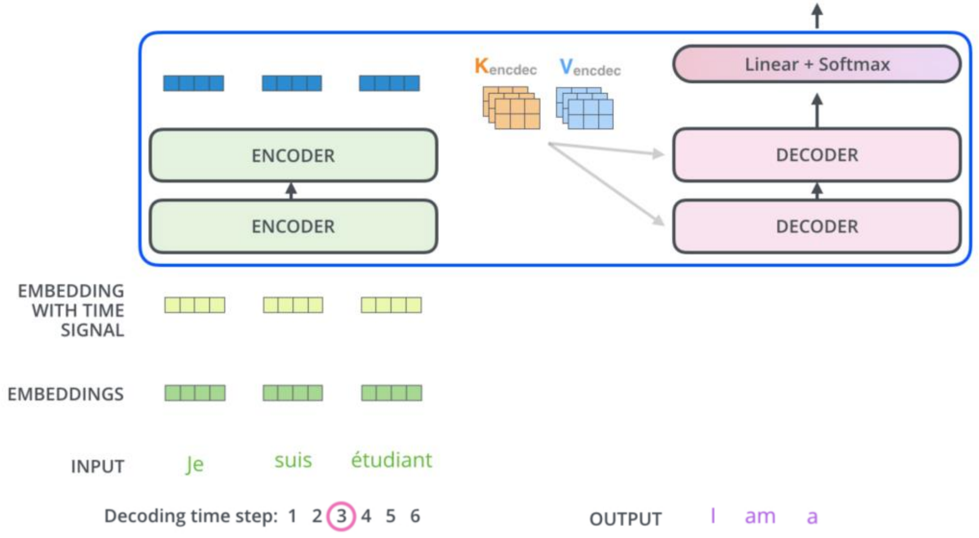


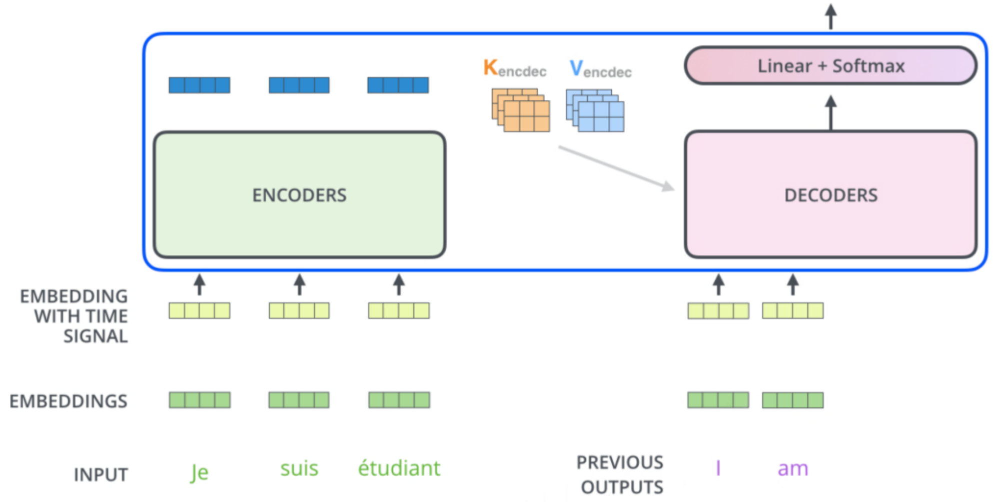
Overall structure of an Encoder

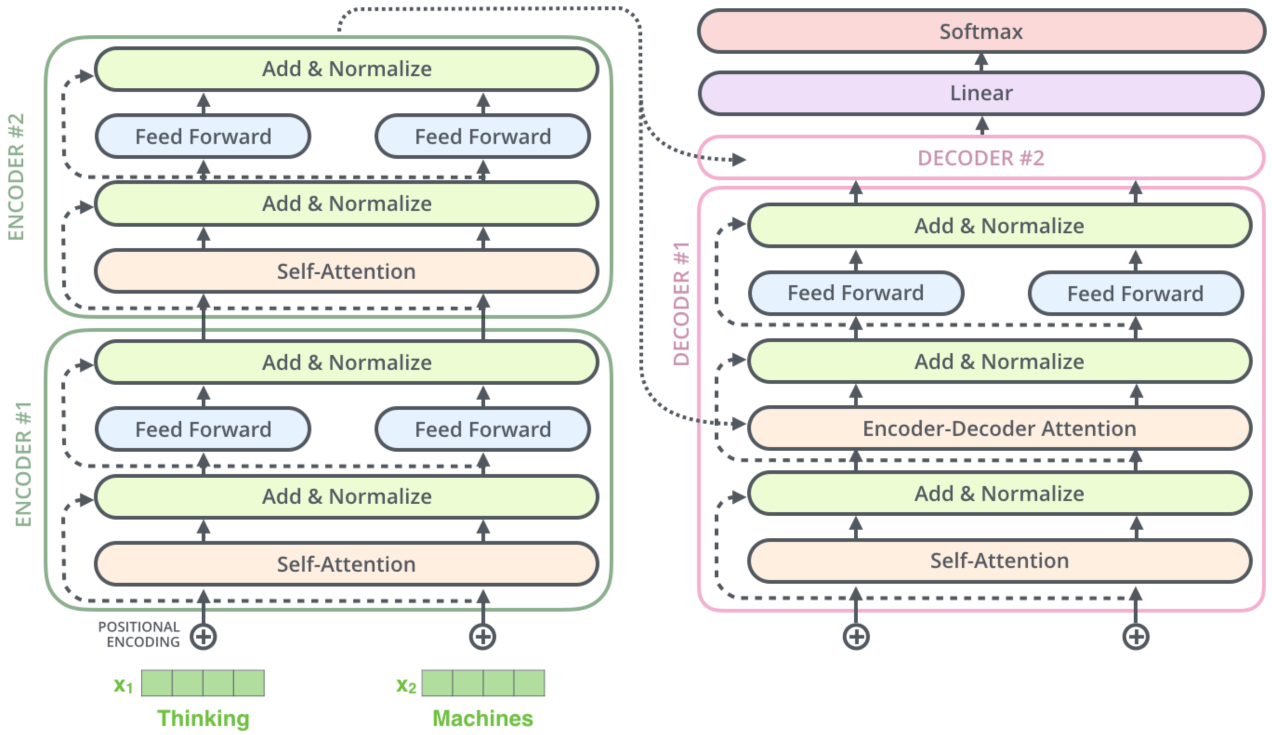
- Residual connections
- Layer normalization



Decoder



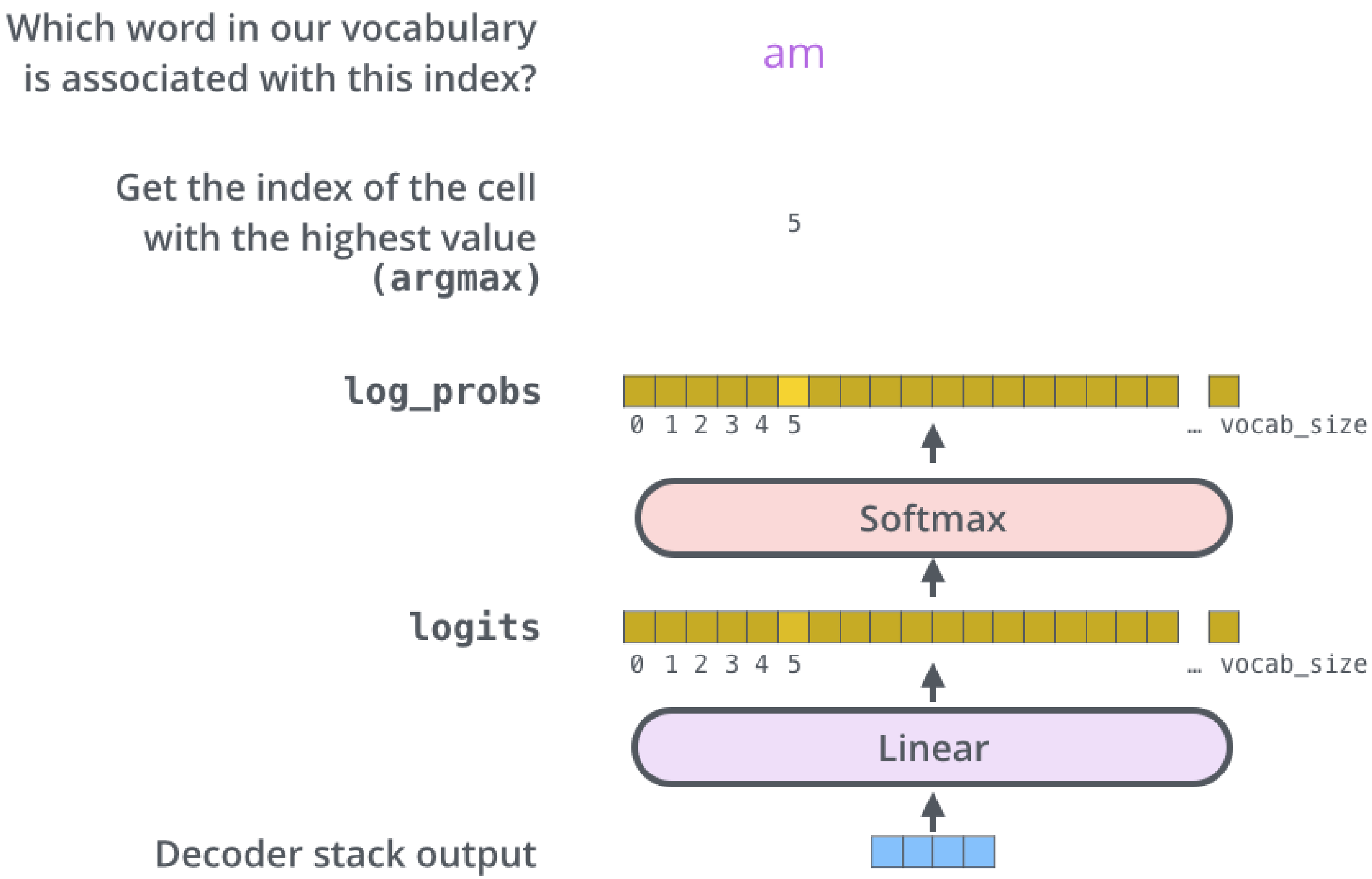




Encoder-Decoder Attention:
Same as Self-attention, but K and V vectors are from the encoder outputs

“Masked” Attention:
Since decoders should only consider earlier positions in the sequence, future positions are “masked” by setting them to −∞ before the softmax step

Linear and Softmax layers



Which word in our vocabulary is associated with this index?

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

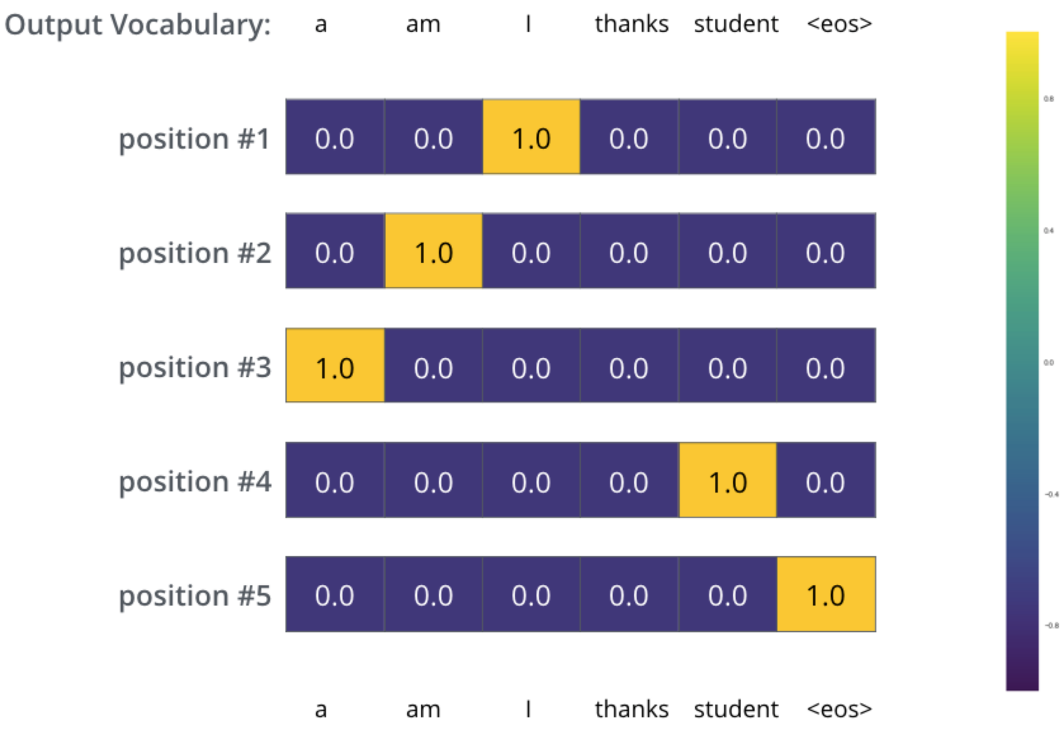
log_probs

logits

Decoder stack output

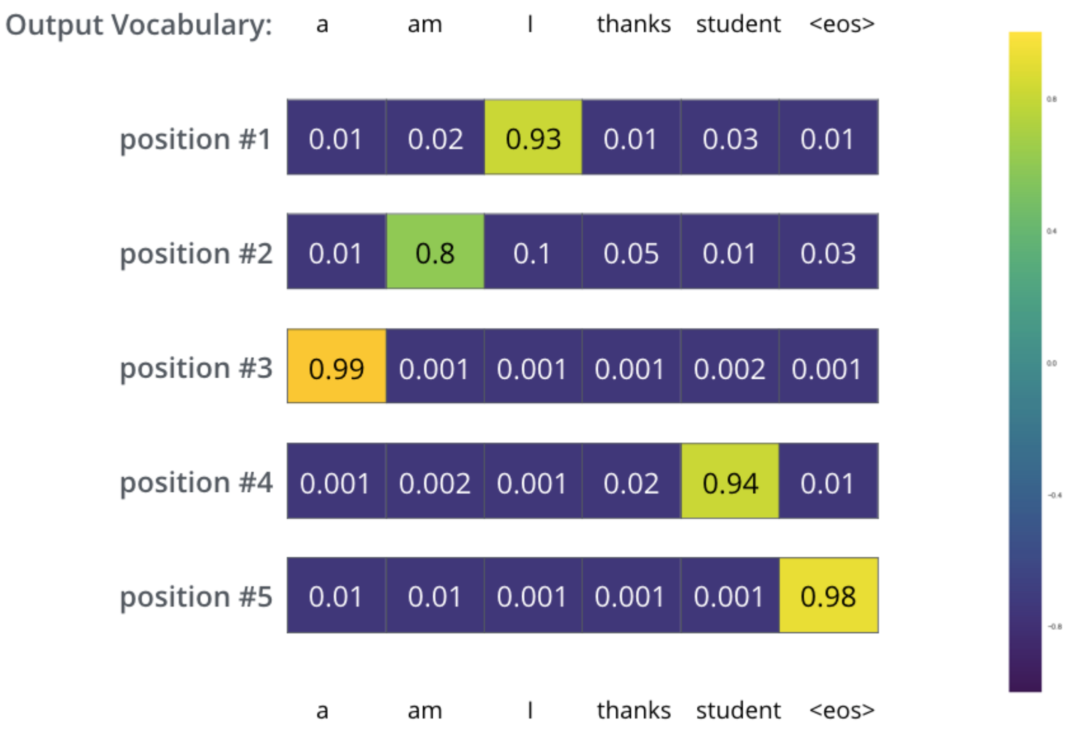
Training the model

Target Model Outputs



Output Vocabulary:	a	am	I	thanks	student	<eos>
position #1	0.0	0.0	1.0	0.0	0.0	0.0
position #2	0.0	1.0	0.0	0.0	0.0	0.0
position #3	1.0	0.0	0.0	0.0	0.0	0.0
position #4	0.0	0.0	0.0	0.0	1.0	0.0
position #5	0.0	0.0	0.0	0.0	0.0	1.0

Trained Model Outputs

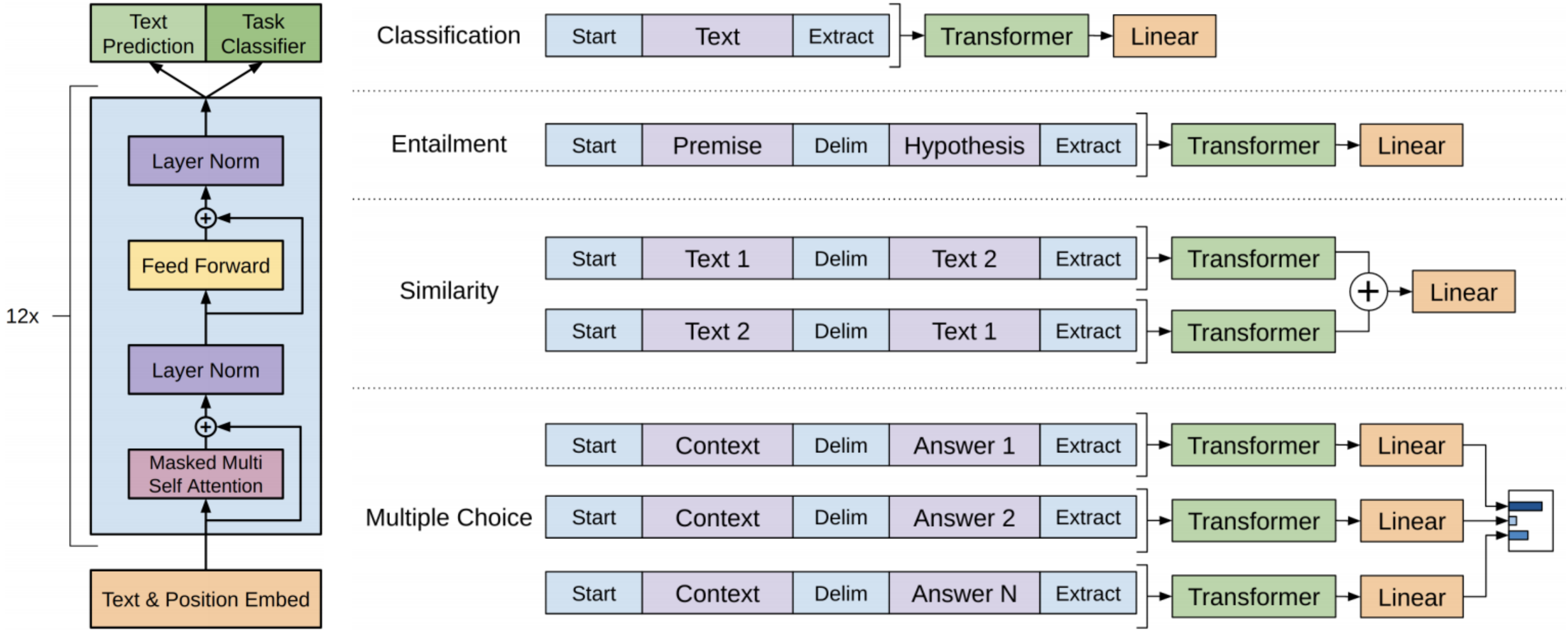


Output Vocabulary:	a	am	I	thanks	student	<eos>
position #1	0.01	0.02	0.93	0.01	0.03	0.01
position #2	0.01	0.8	0.1	0.05	0.01	0.03
position #3	0.99	0.001	0.001	0.001	0.002	0.001
position #4	0.001	0.002	0.001	0.02	0.94	0.01
position #5	0.01	0.01	0.001	0.001	0.001	0.98

ex. Cross-entropy loss

※ GPT (Generative Pre-Training)

- General model for many NLP tasks
- Only uses Decoder part of the Transformer architecture



References

- Attention Is All You Need (paper): https://arxiv.org/abs/1706.03762
- The Illustrated Transformer (blog post): https://jalammar.github.io/illustrated-transformer/
- Improving Language Understanding by Generative Pre-Training (paper): https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf