Knowledge Discovery & Data Mining
— Transformers —

Instructor: Yong Zhuang

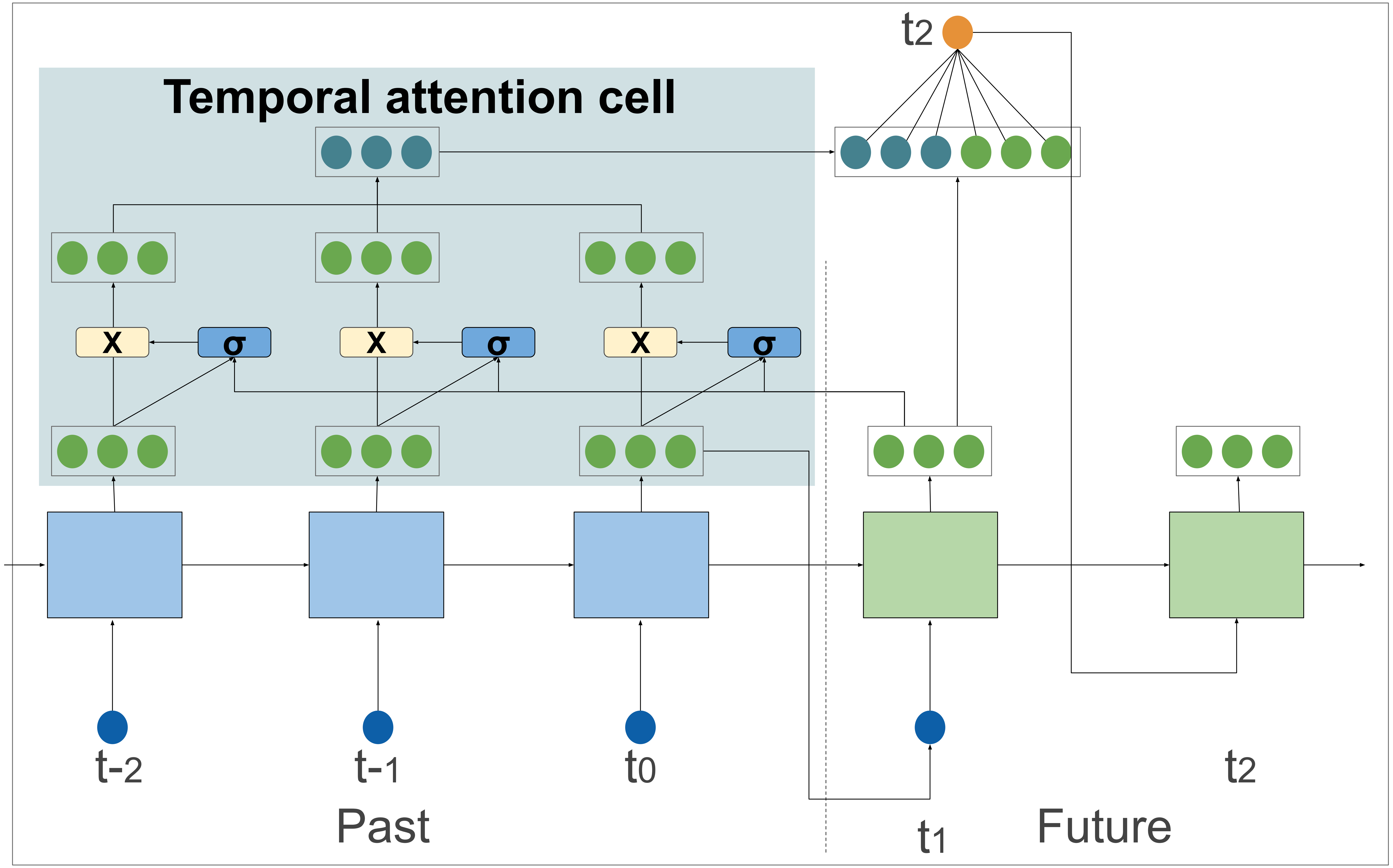
yong.zhuang@gvsu.edu

Based on the original version at https://jalammar.github.io/illustrated-transformer/

Outline

- Transformers
  - Self-Attention, Softmax,
  - Multi-Headed
  - Self-Attention,
  - Positional Encoding,
  - Residual Connection,
  - Layer Normalization,
  - Encoder-Decoder

Temporal attention injection

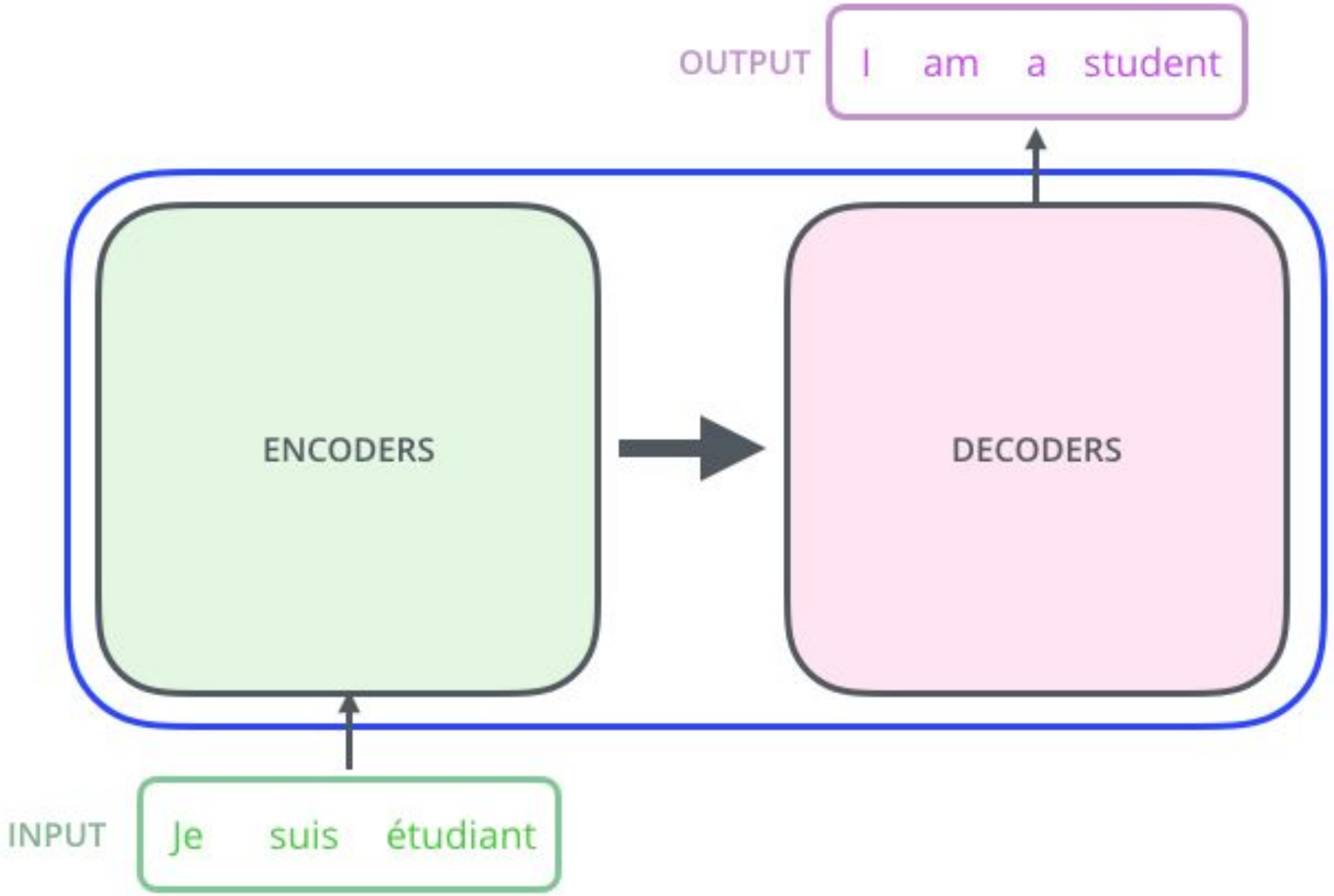


In light of the performance enhancements brought by attention mechanisms, is the RNN component still necessary in our models, or can attention mechanisms effectively replace the traditional role of RNNs?

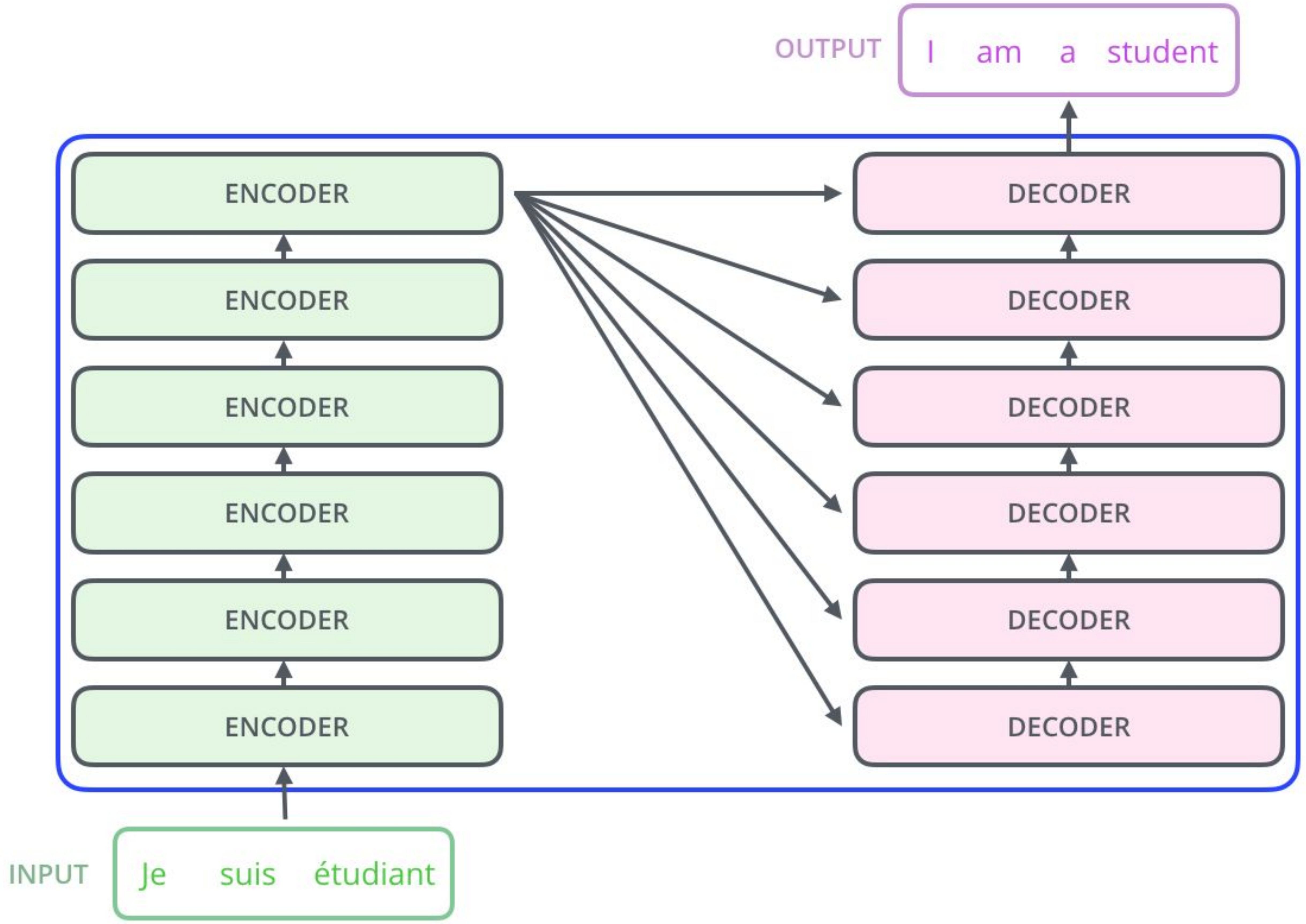
Transformer

Attention is all you need

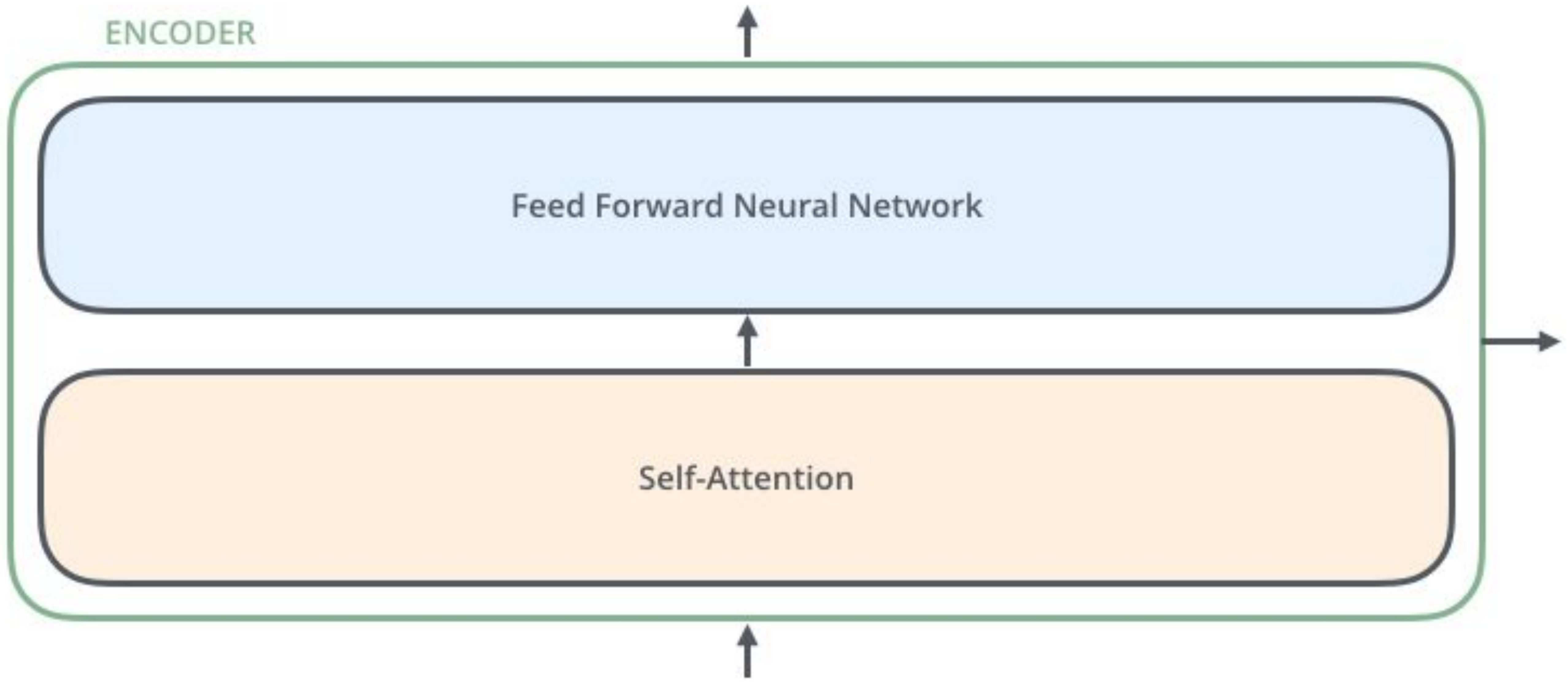
Encoder-Decoder Architecture



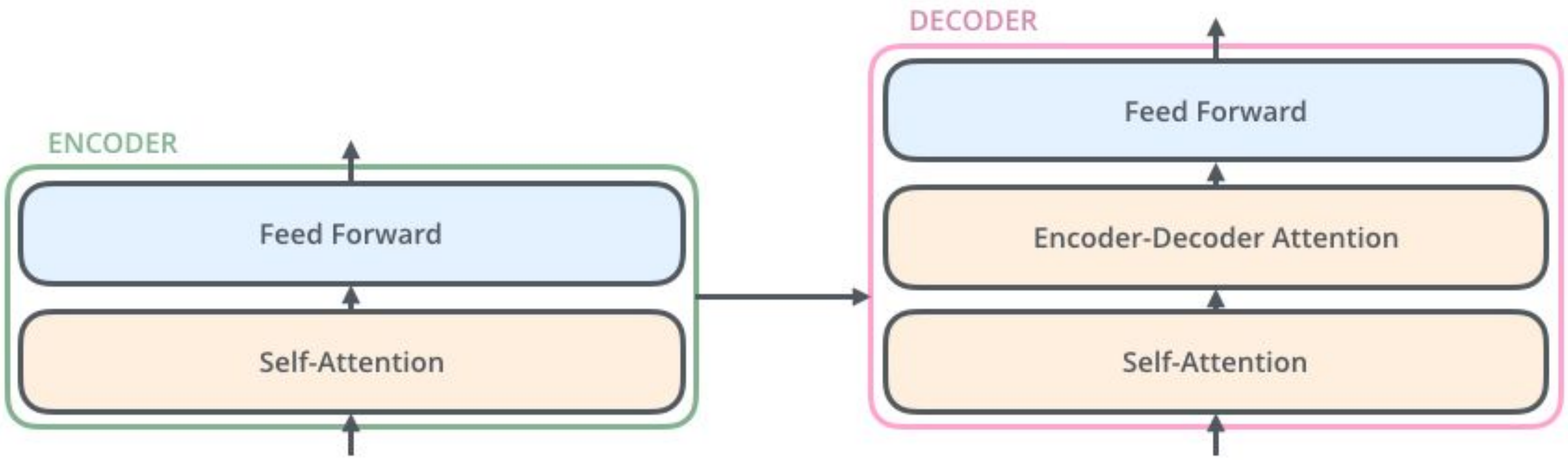
Encoder-Decoder Architecture



Encoder

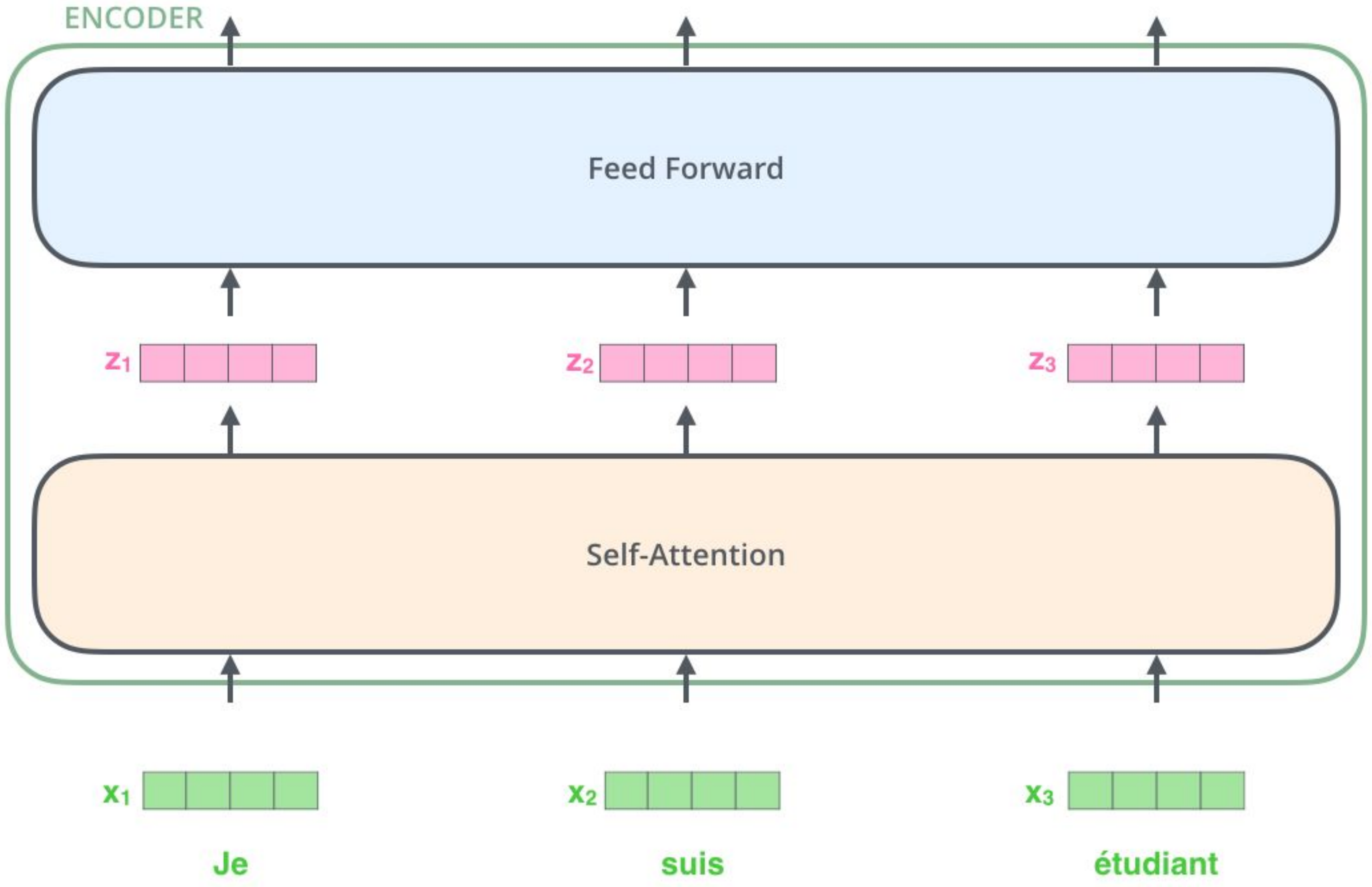


Decoder

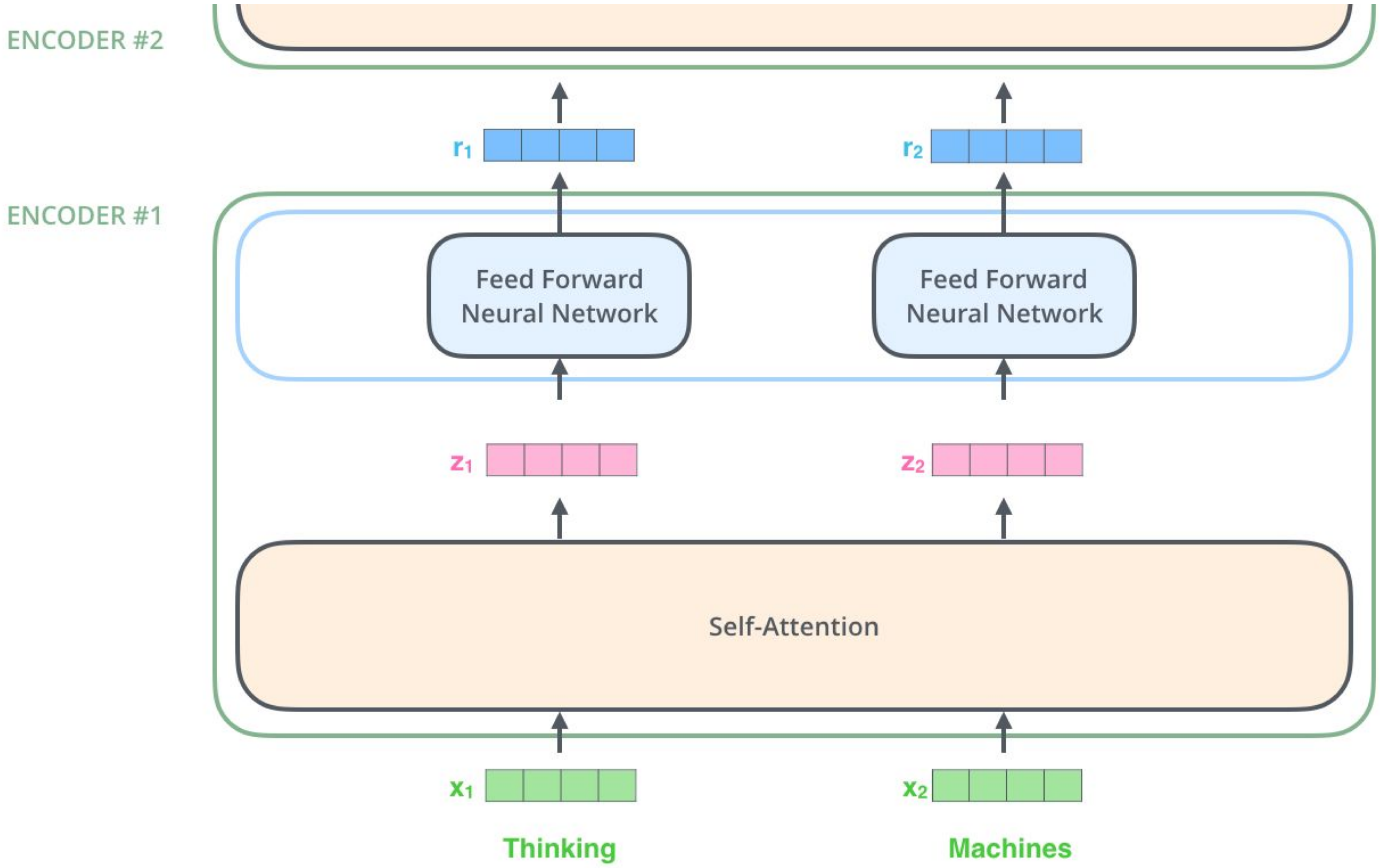

DECODER
Feed Forward
Encoder-Decoder Attention
Self-Attention
ENCODER
Feed Forward
Self-Attention

Word Embedding

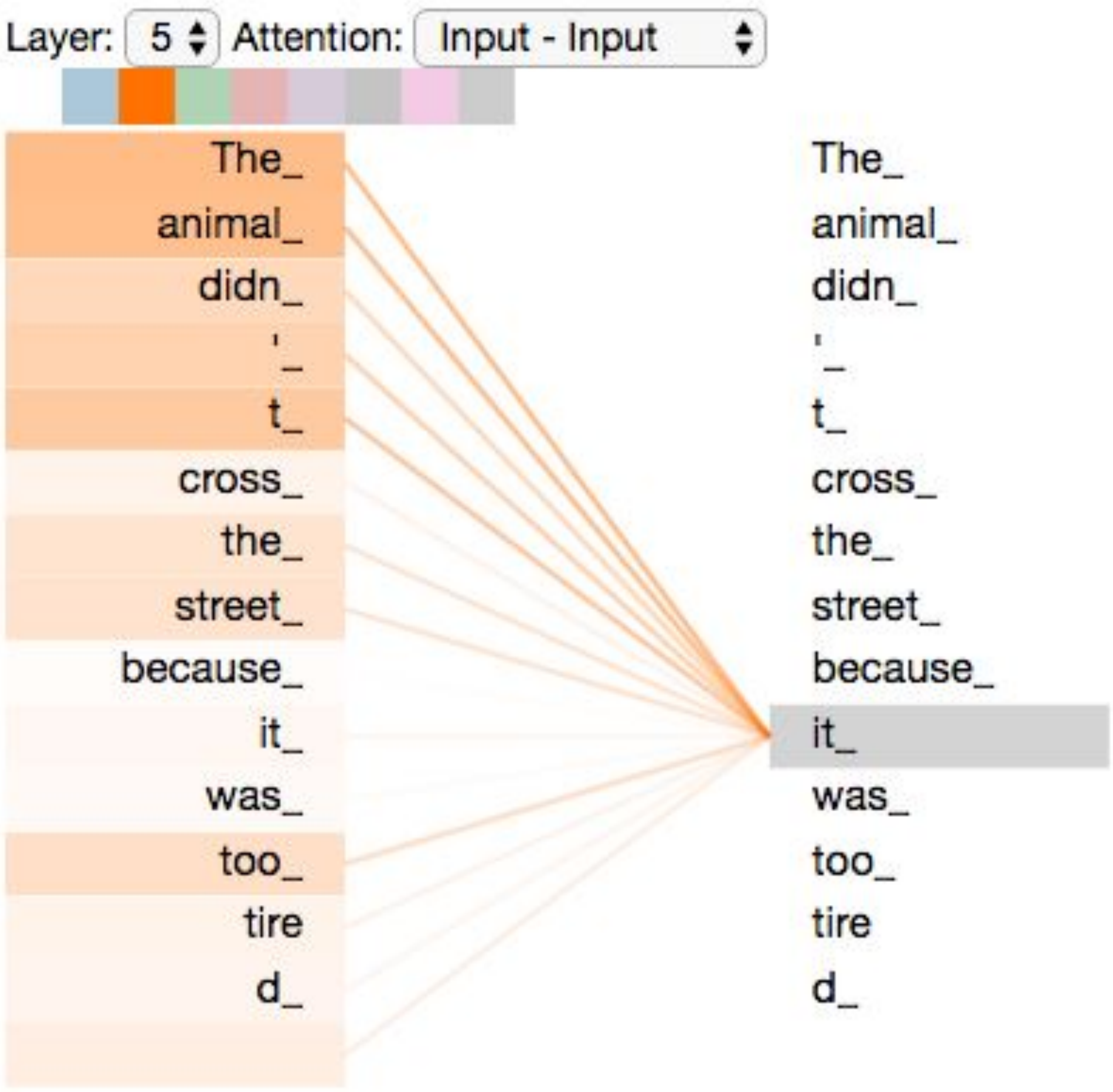
Feeding Word Embeddings into the Encoder



Encoding

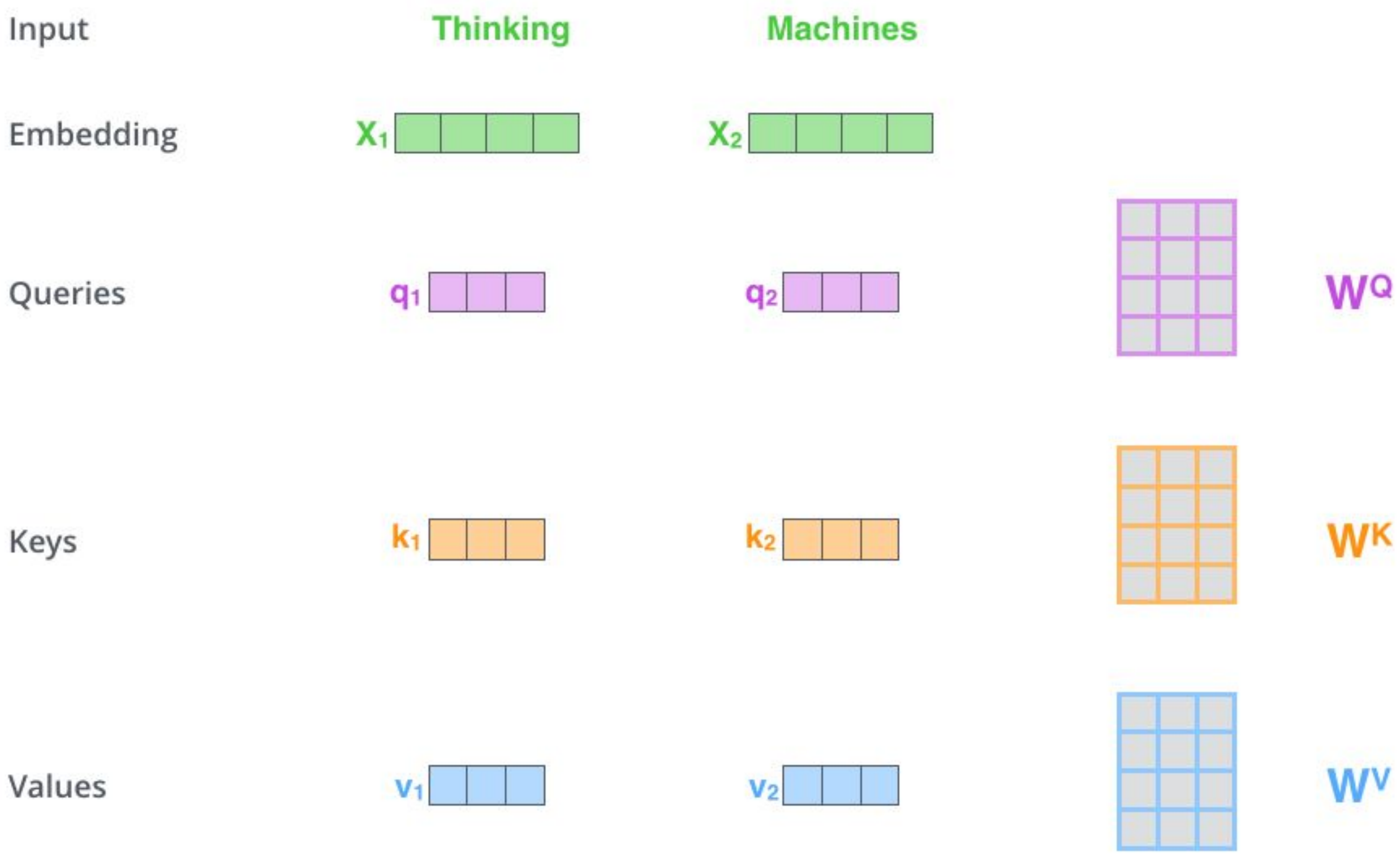

ENCODER #2
ENCODER #1
r_1
r_2
Feed Forward Neural Network
Feed Forward Neural Network
z_1
z_2
Self-Attention
x_1
x_2
Thinking
Machines

Self-Attention

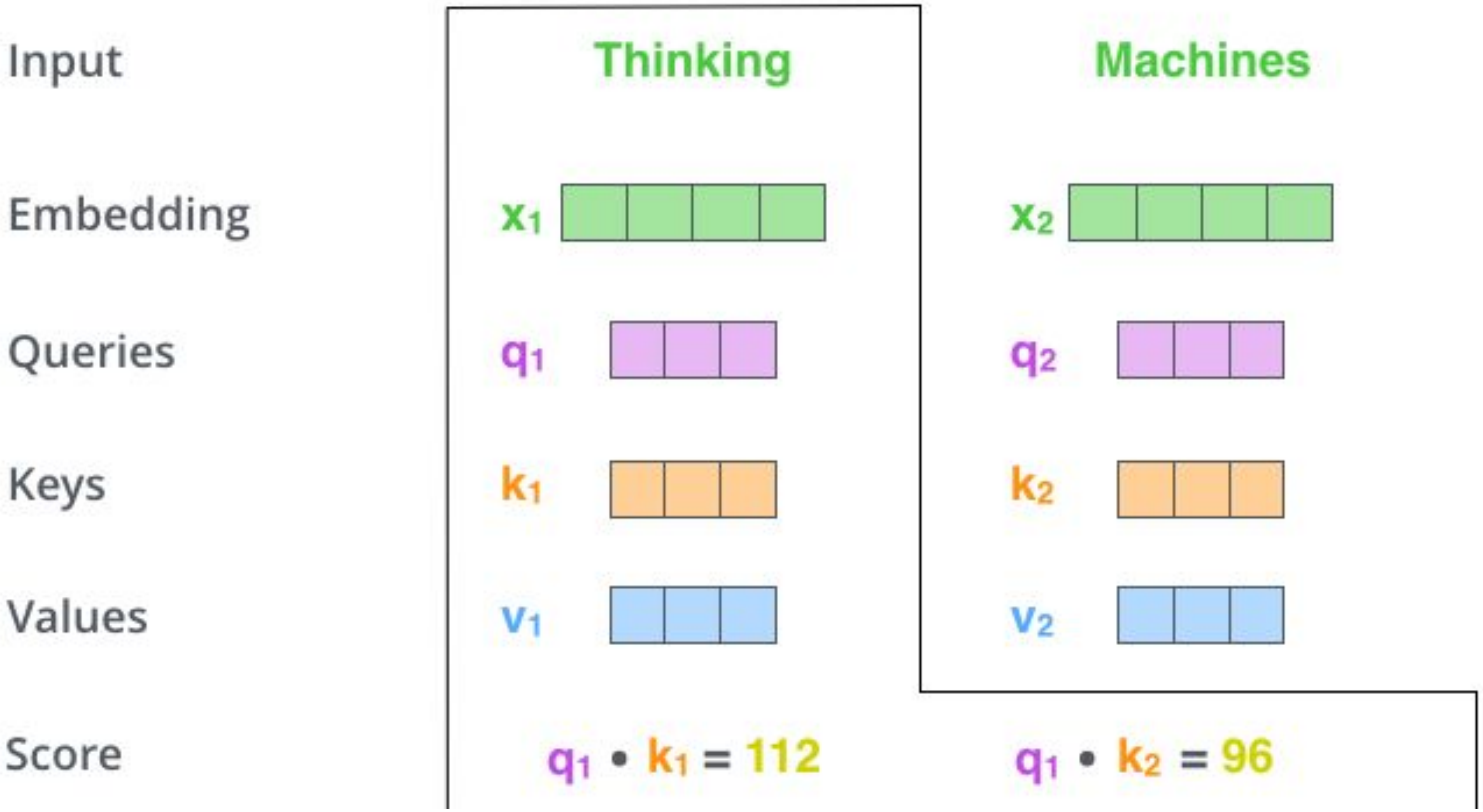


As we are encoding the word "it" in encoder #5 (the top encoder in the stack), part of the attention mechanism was focusing on "The Animal", and baked a part of its representation into the encoding of "it".

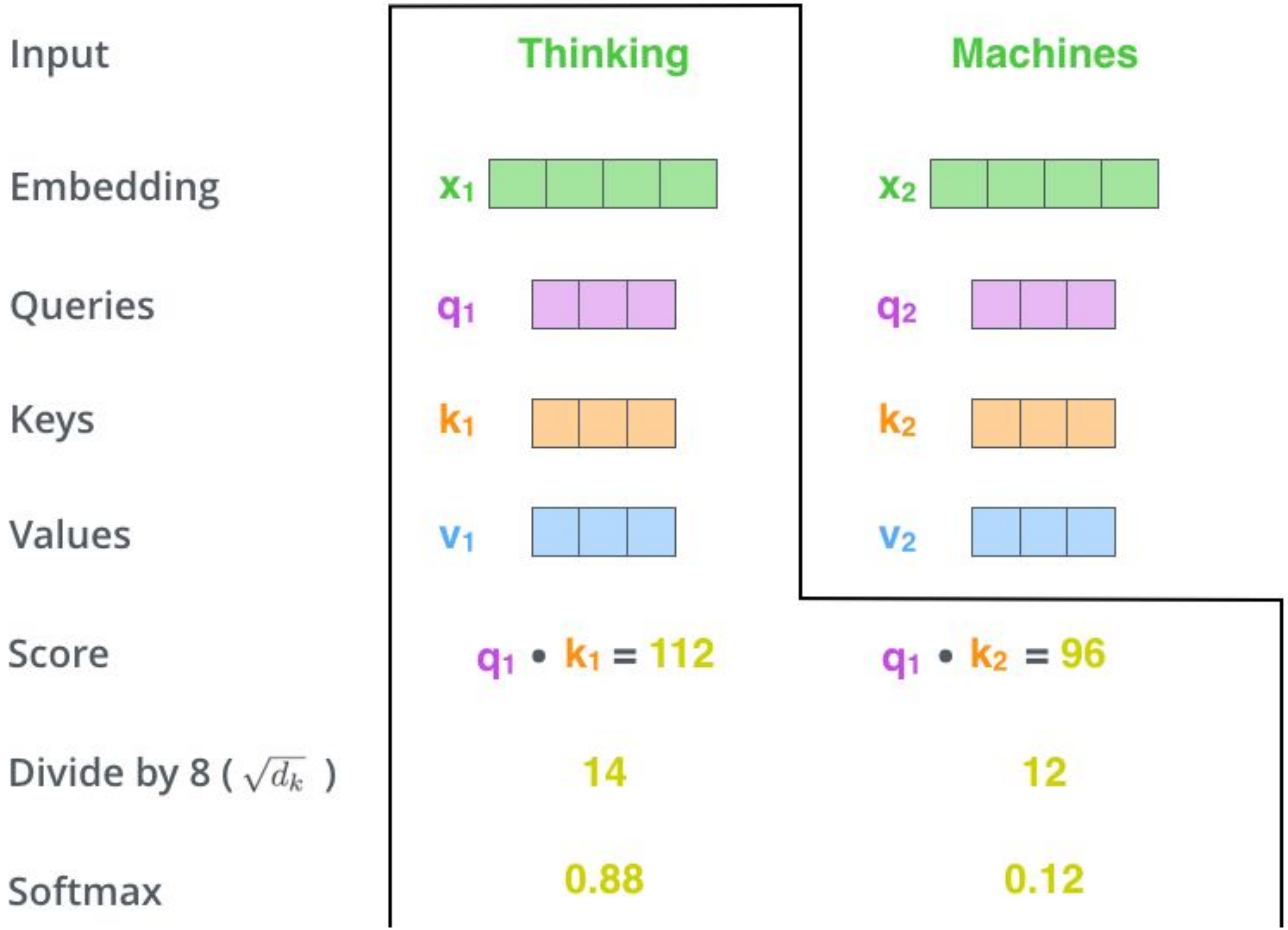
Self-Attention in Detail



Self-Attention in Detail

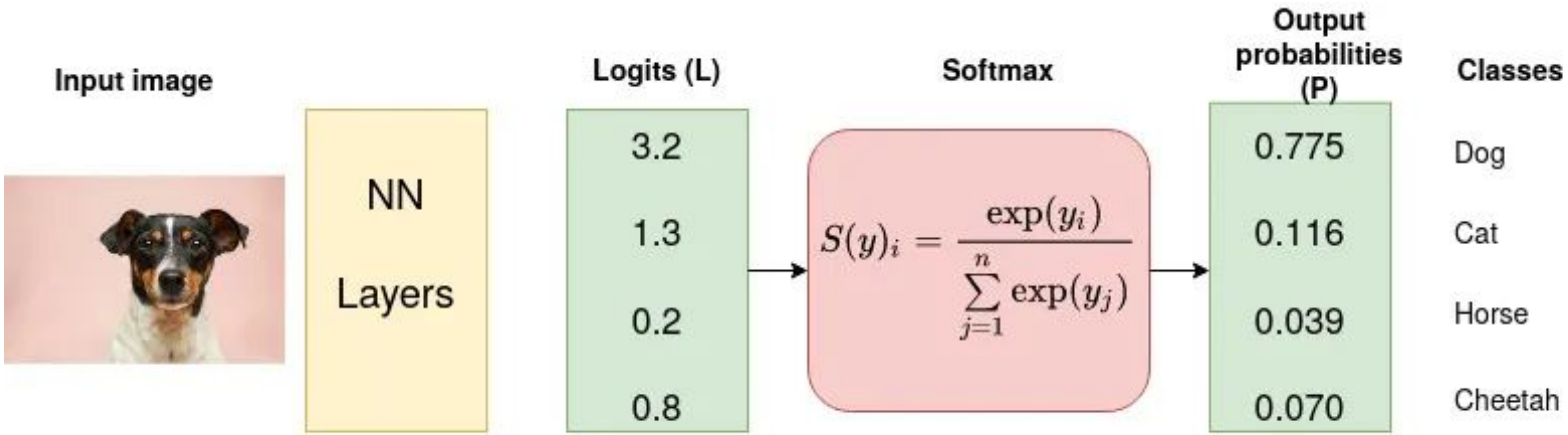


Self-Attention in Detail



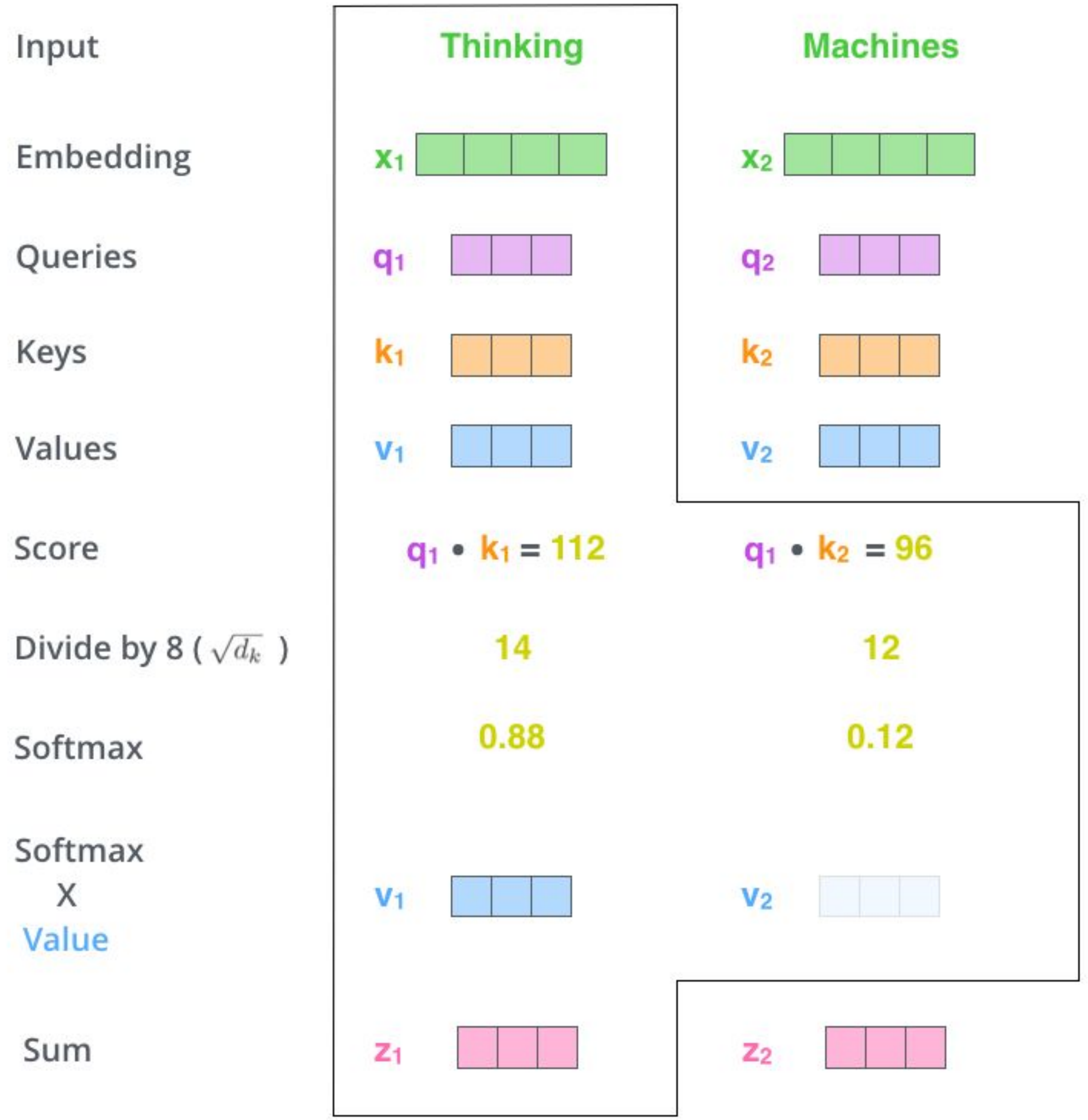
Divide the scores by 8 (the square root of the dimension of the key vectors used in the paper – 64. This leads to having more stable gradients.

Softmax

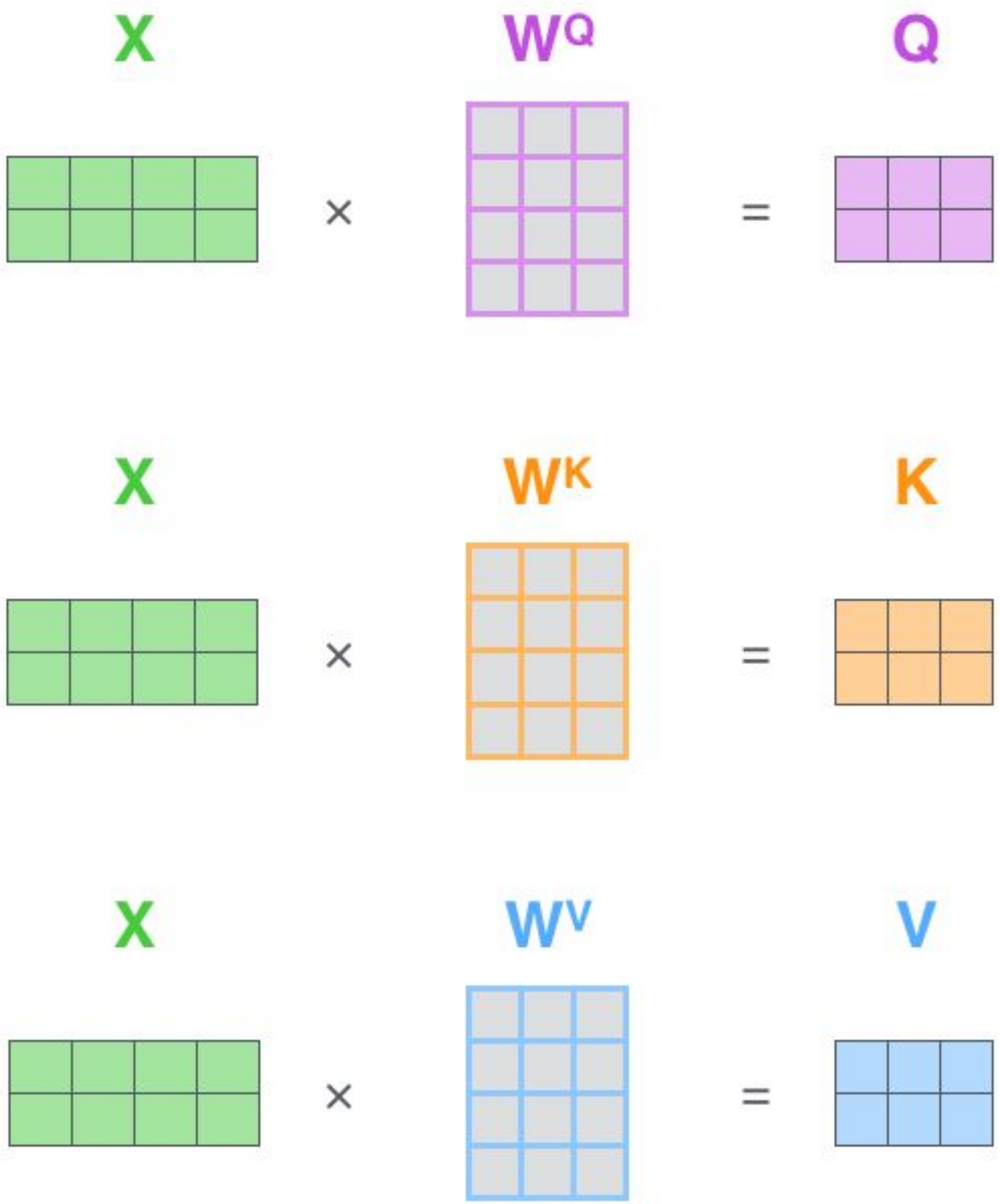


Input image source: Photo by Victor Grabarczyk on Unsplash . Diagram by author.

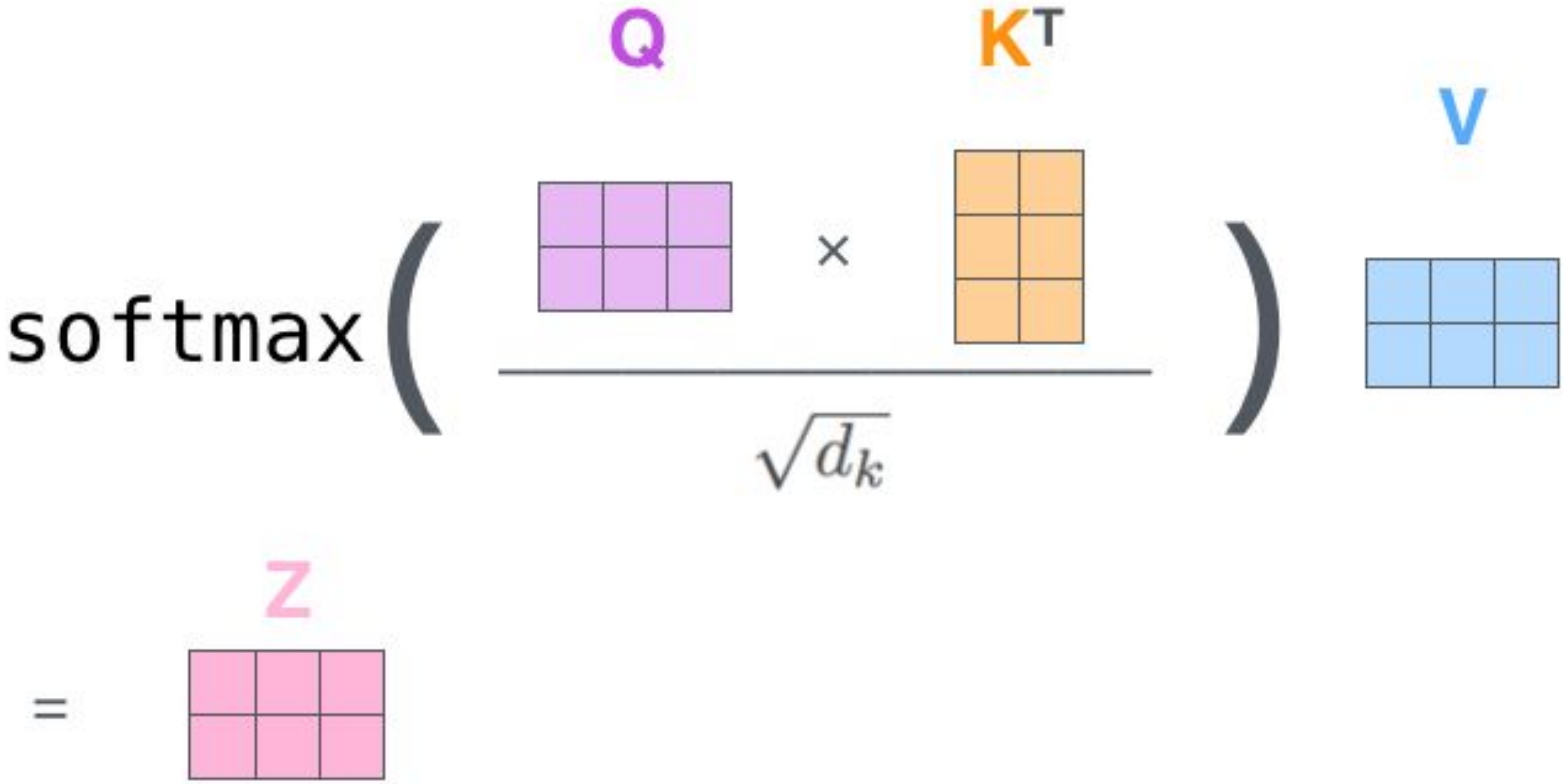
Self-Attention



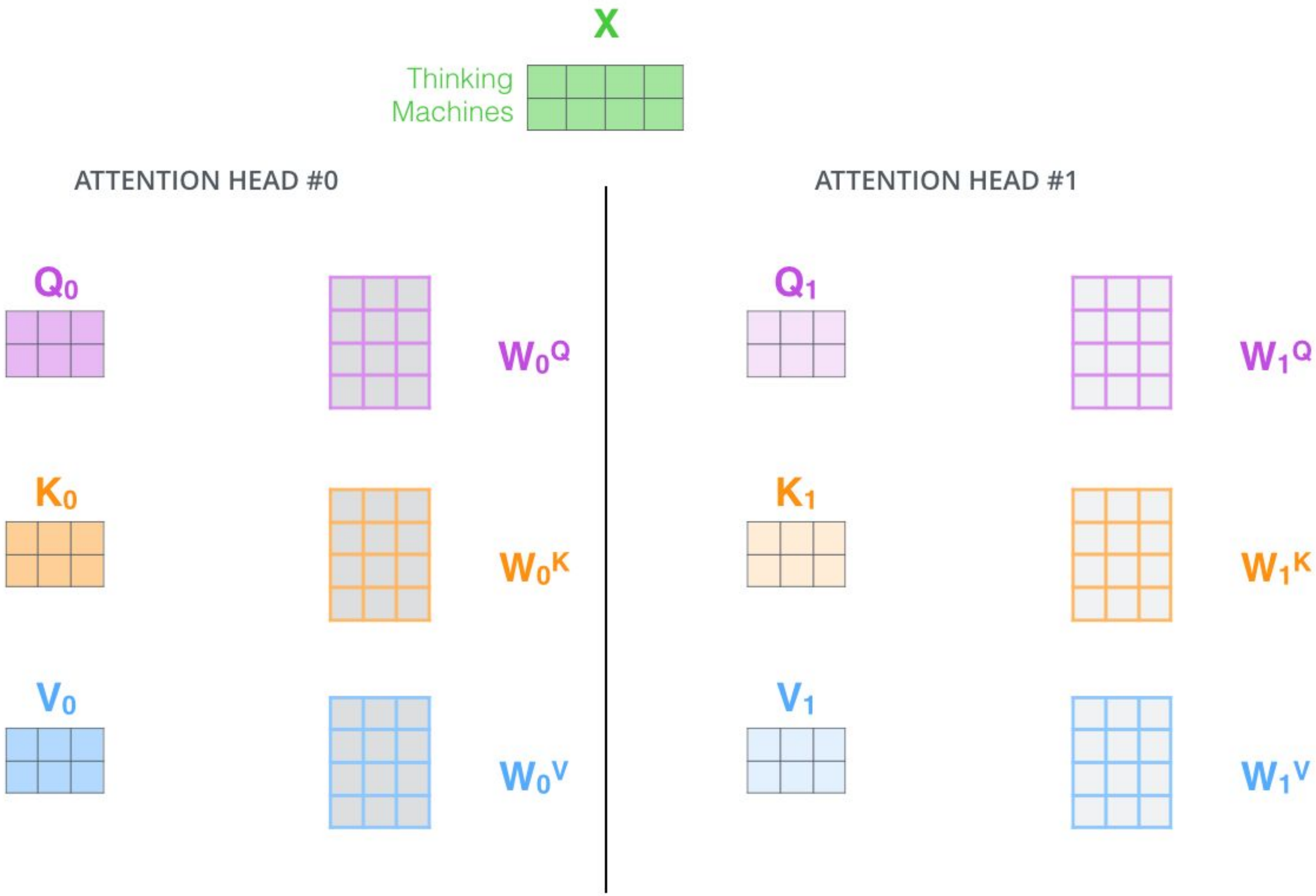
Matrix Calculation of Self-Attention



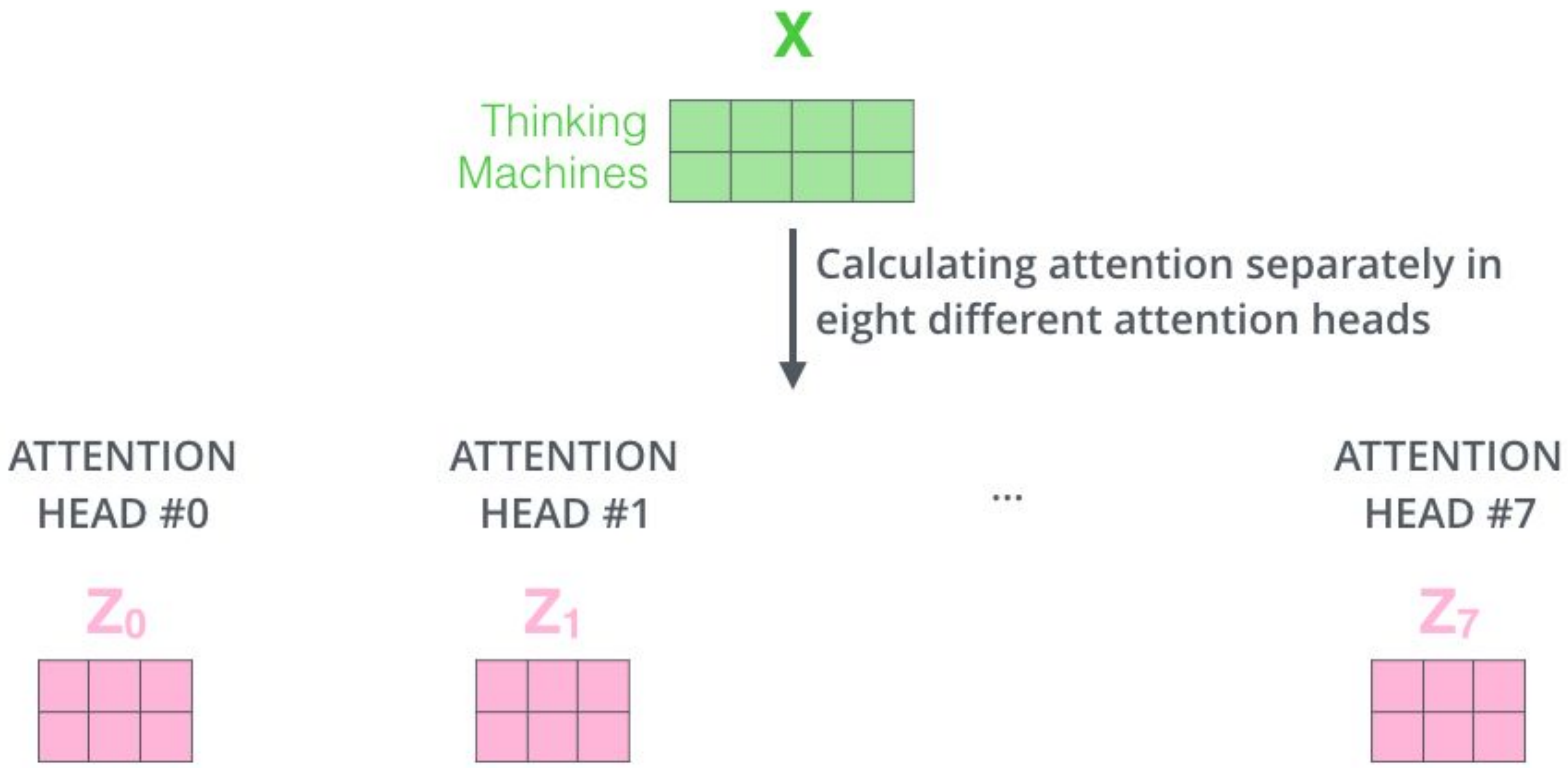
Matrix Calculation of Self-Attention



Multi-Headed Self-Attention



Multi-Headed Self-Attention

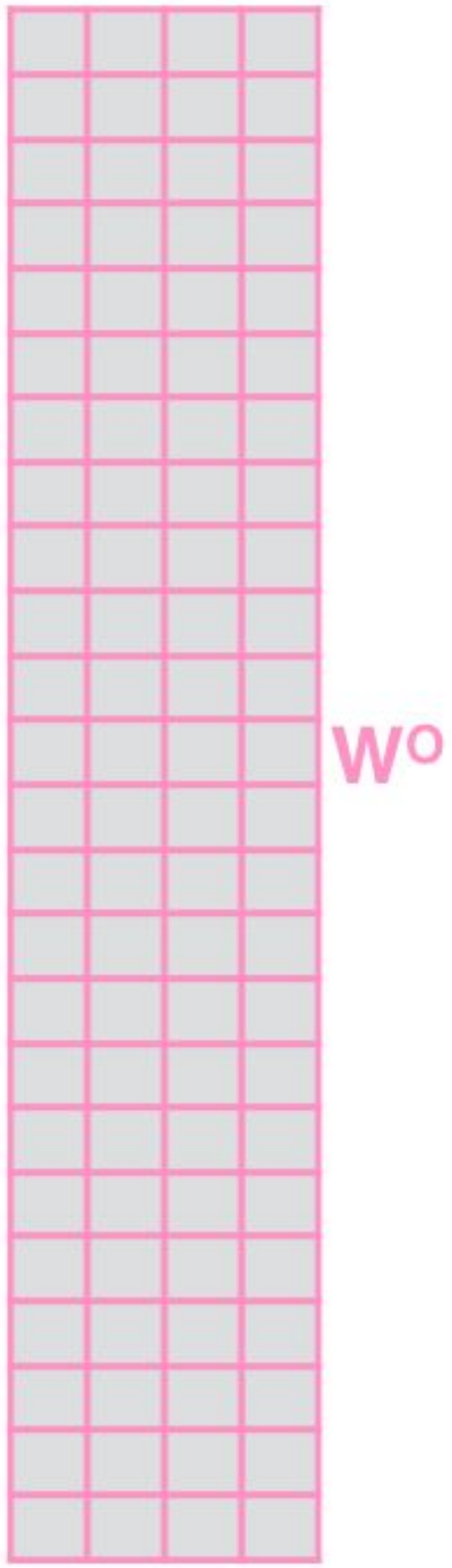


Multi-Headed Self-Attention

1) Concatenate all the attention heads

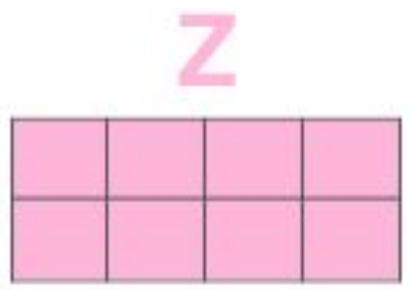
2) Multiply with a weight matrix W^O that was trained jointly with the model

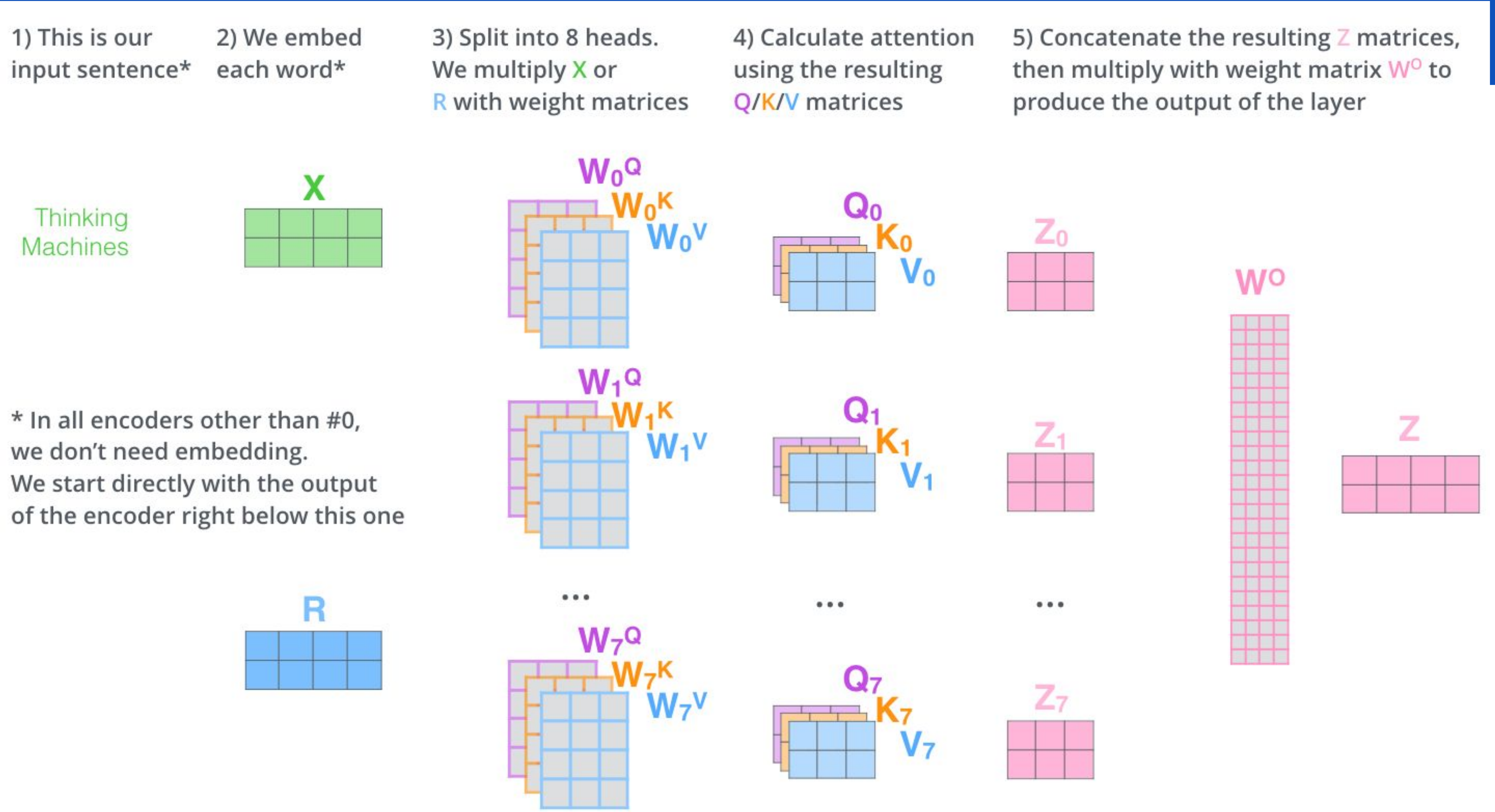
X



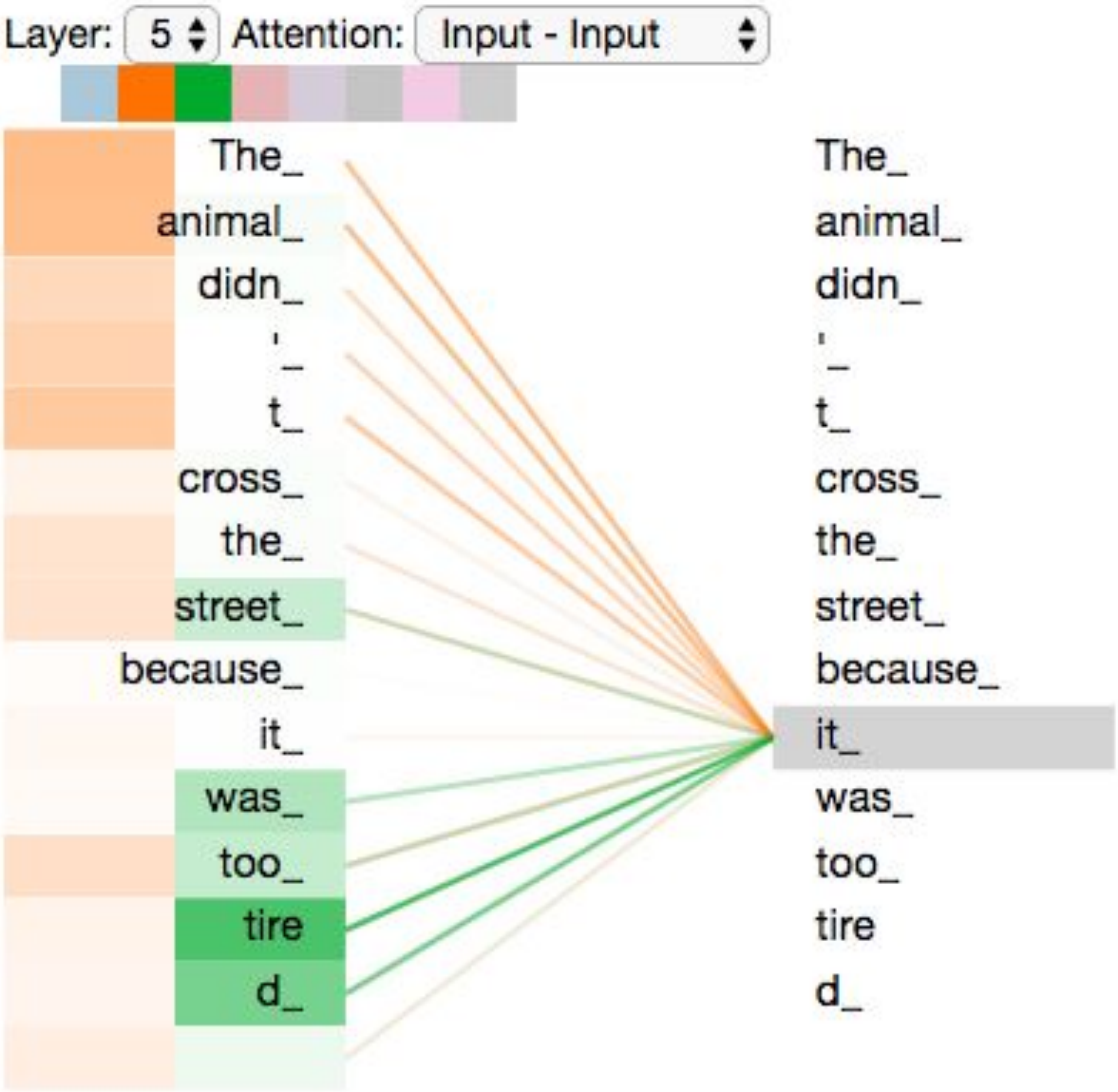
3) The result would be the Z matrix that captures information from all the attention heads. We can send this forward to the FFNN

=



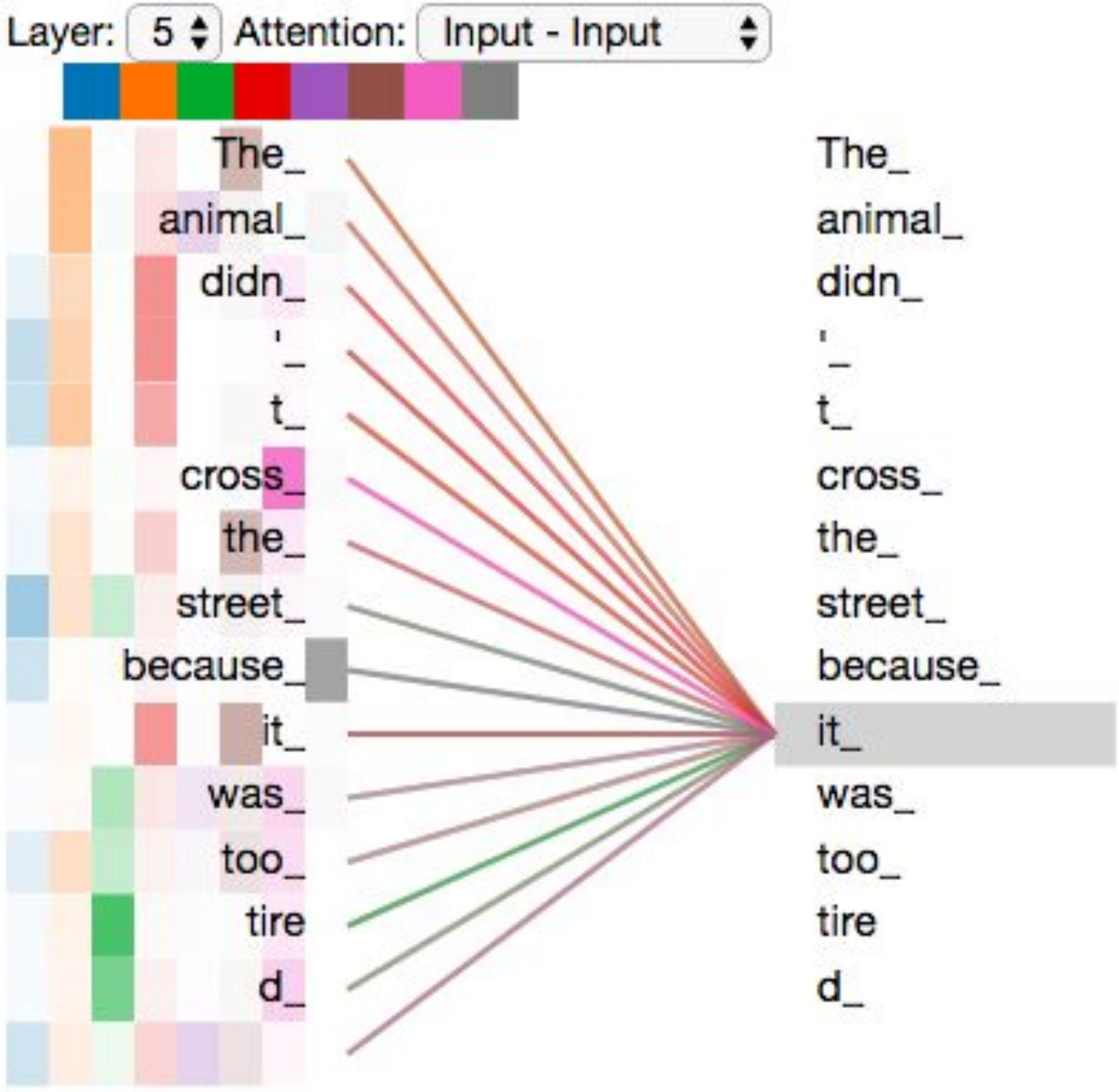

1) This is our input sentence*
2) We embed each word*
3) Split into 8 heads. We multiply X or R with weight matrices
4) Calculate attention using the resulting Q/K/V matrices
5) Concatenate the resulting Z matrices, then multiply with weight matrix W^O to produce the output of the layer
Thinking Machines
X
R
* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one
W0Q
W0K
W0V
W1Q
W1K
W1V
W7Q
W7K
W7V
Q0
K0
V0
Q1
K1
V1
Q7
K7
V7
Z0
Z1
Z7
WO
Z

Multi-Headed Self-Attention



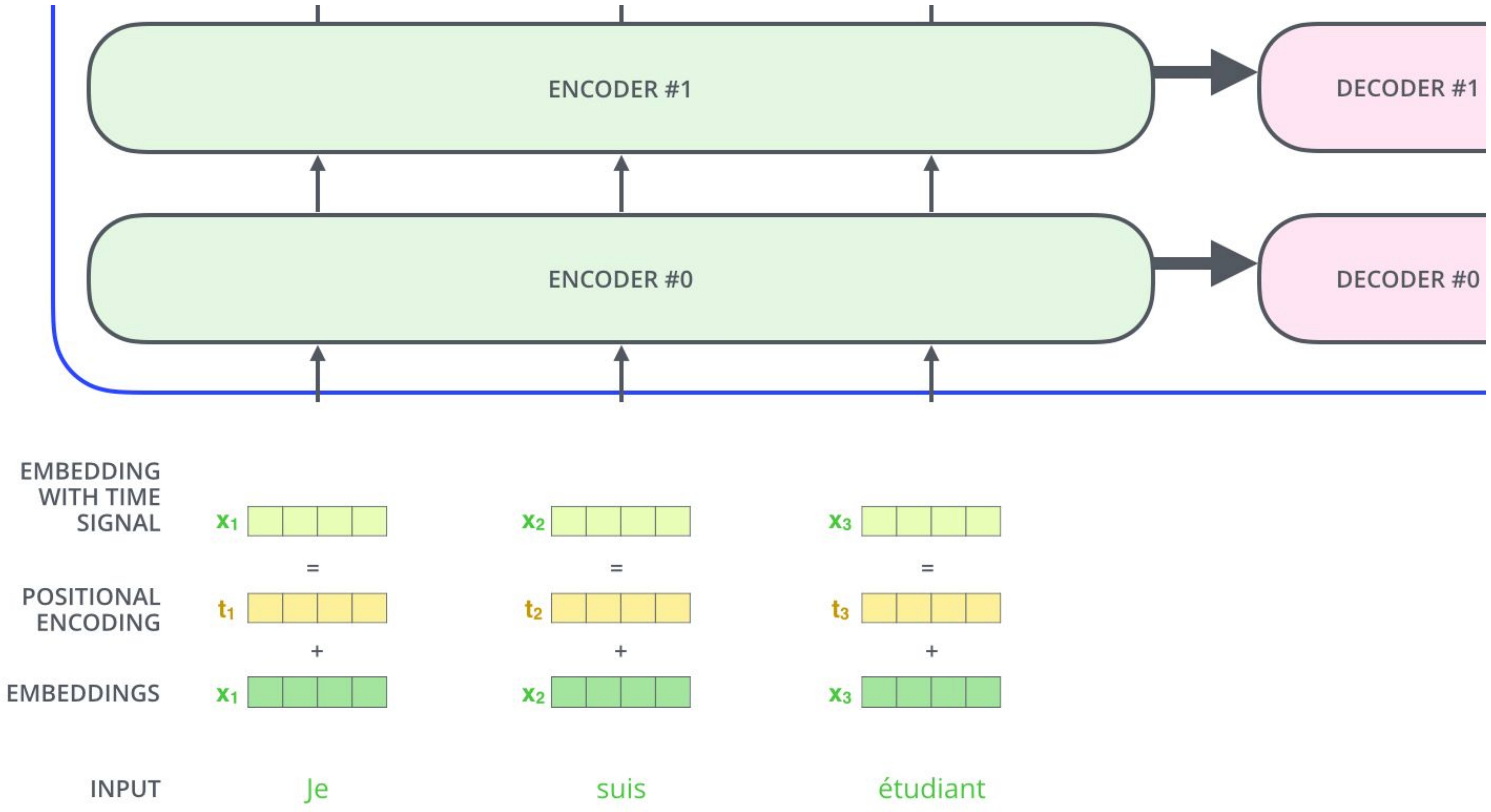
As we encode the word "it", one attention head is focusing most on "the animal", while another is focusing on "tired" -- in a sense, the model's representation of the word "it" bakes in some of the representation of both "animal" and "tired".

Multi-Headed Self-Attention

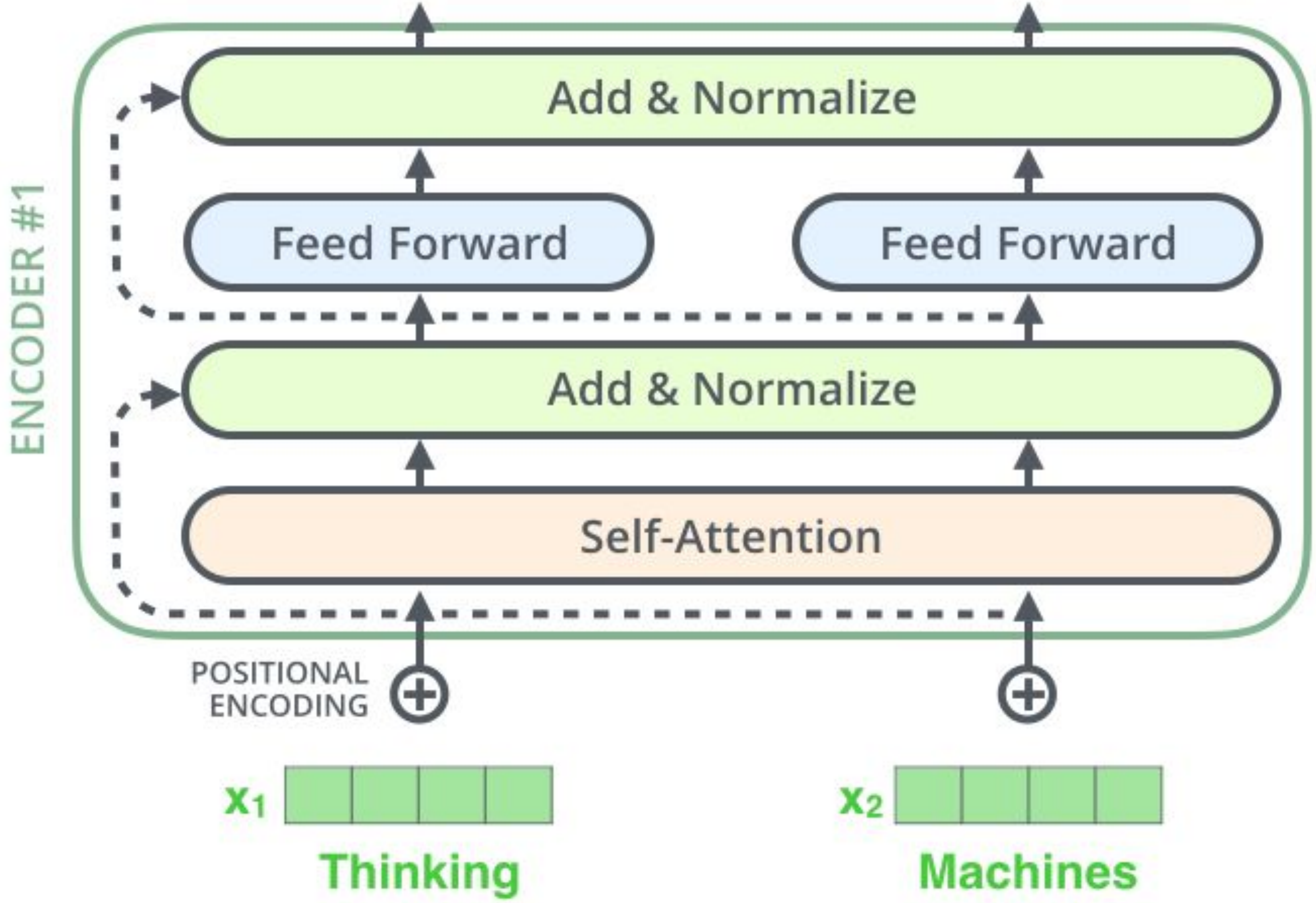


In considering the importance of the positional information of elements in sequence data, how can the Transformer model effectively utilize this positional information in its processing?

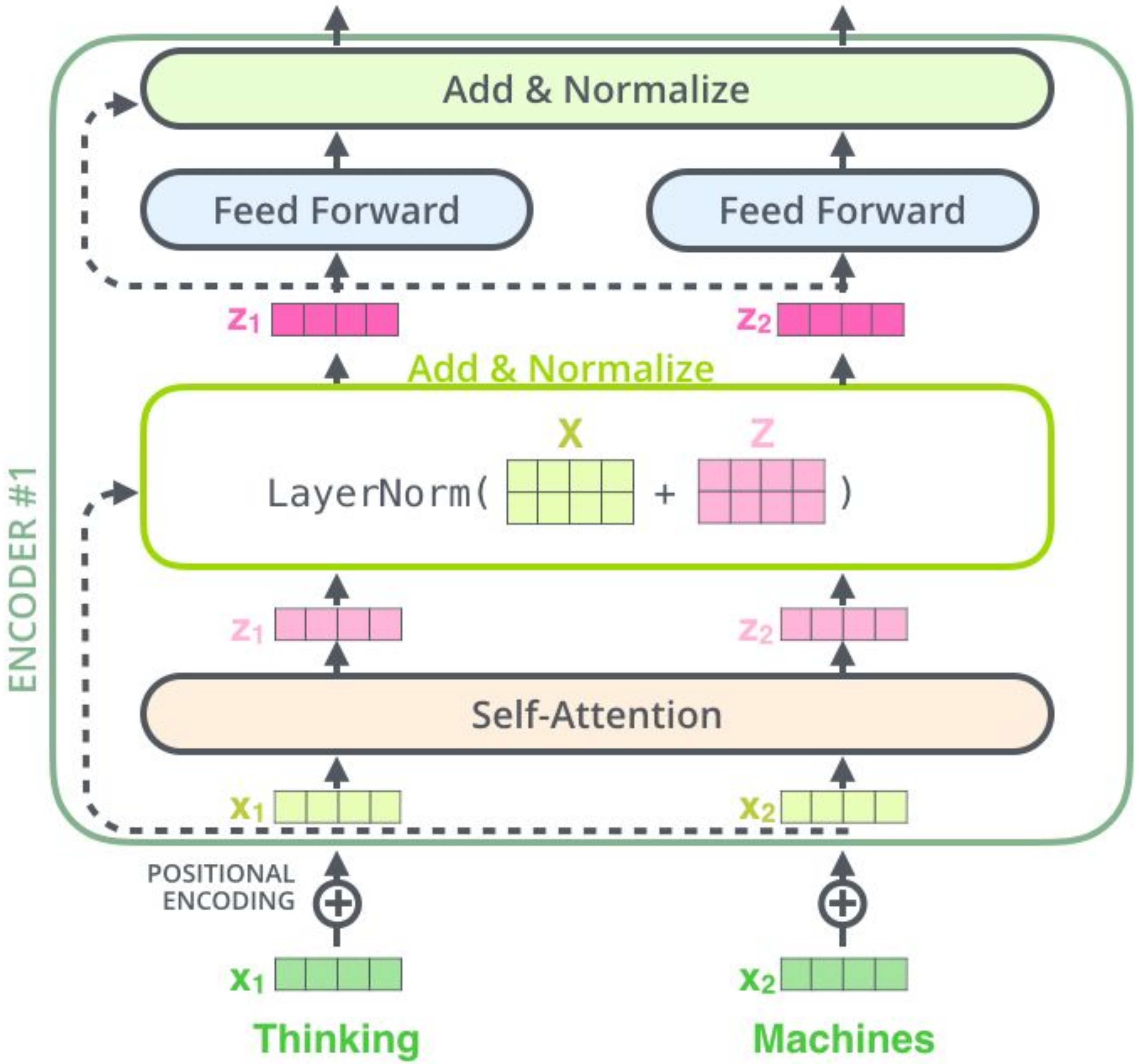
Positional Encoding



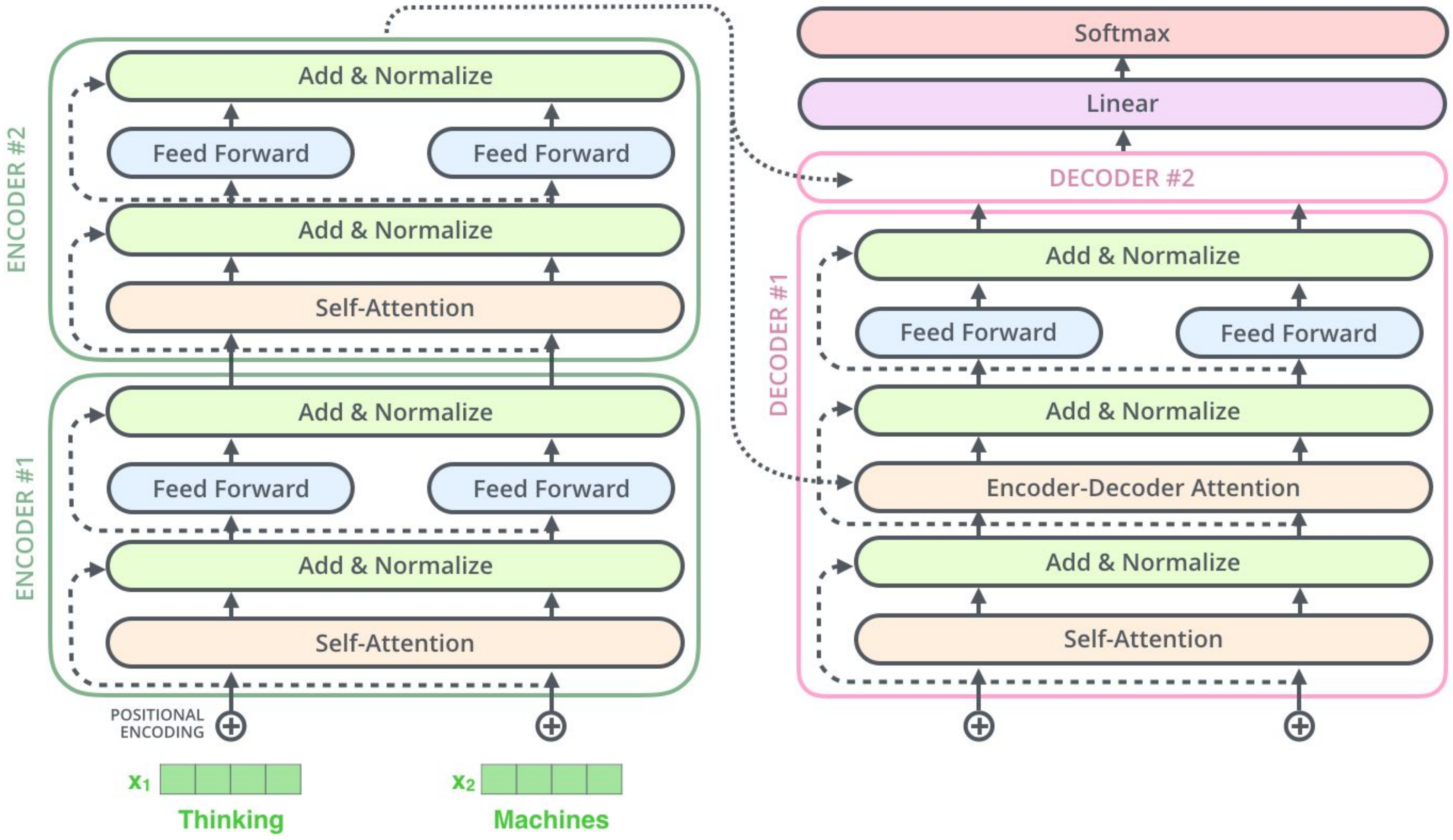
Add Residual Connections



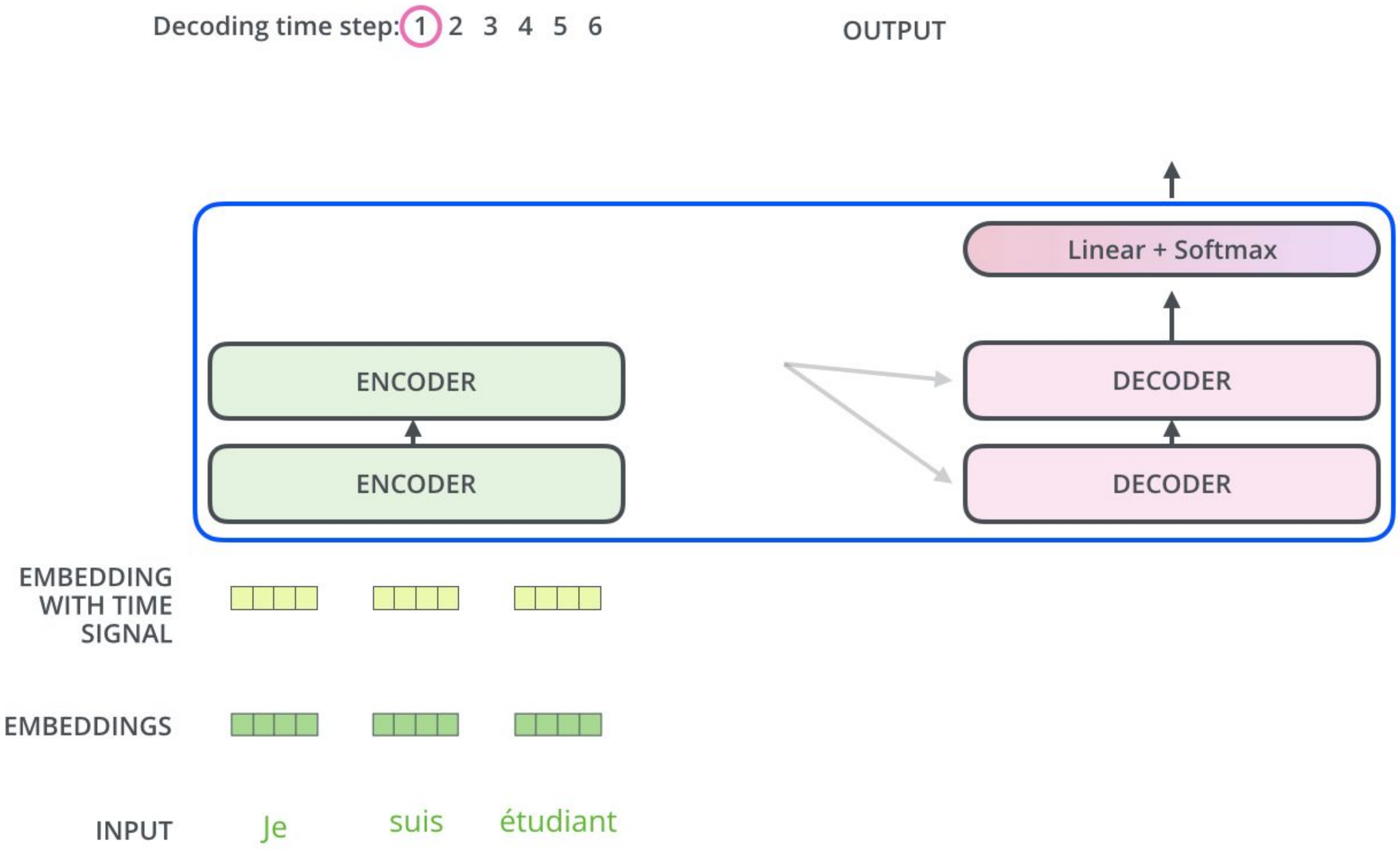
Layer Normalization



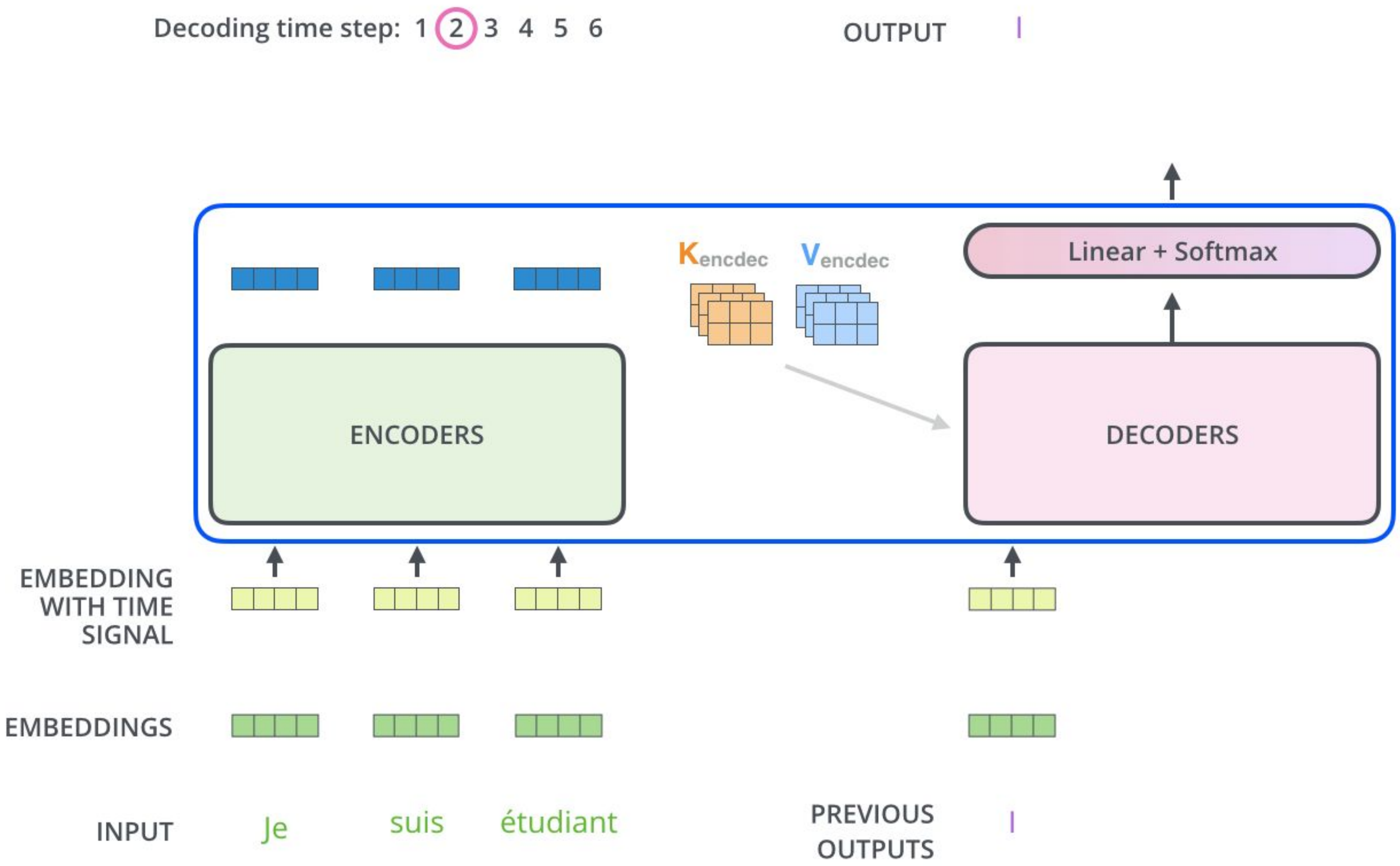
Encoder-Decoder



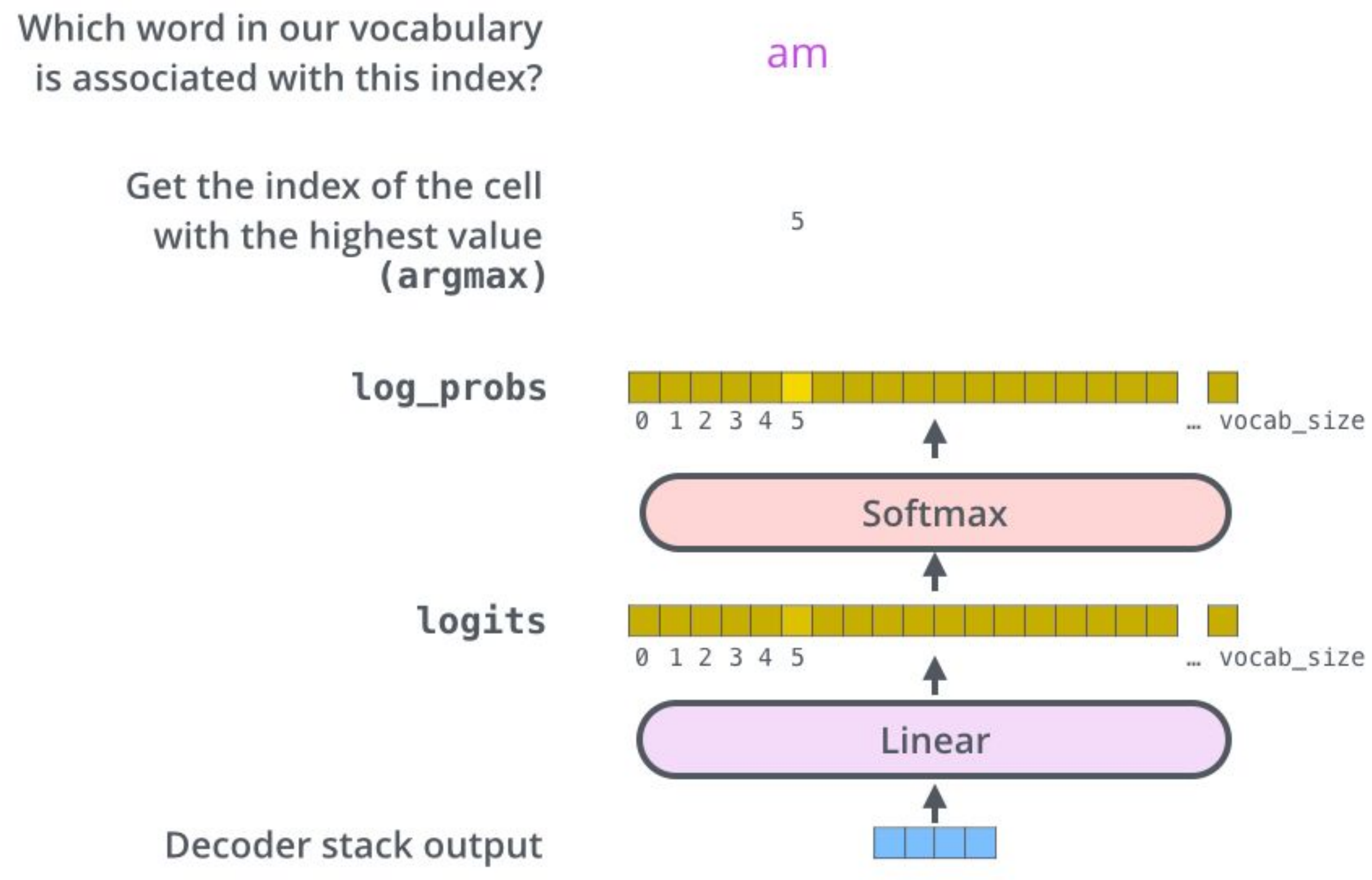
Encoder-Decoder



Encoder-Decoder



The Final Linear and Softmax Layer



Example

Output Vocabulary

WORD	a	am	I	thanks	student	<eos>
INDEX	0	1	2	3	4	5

Example

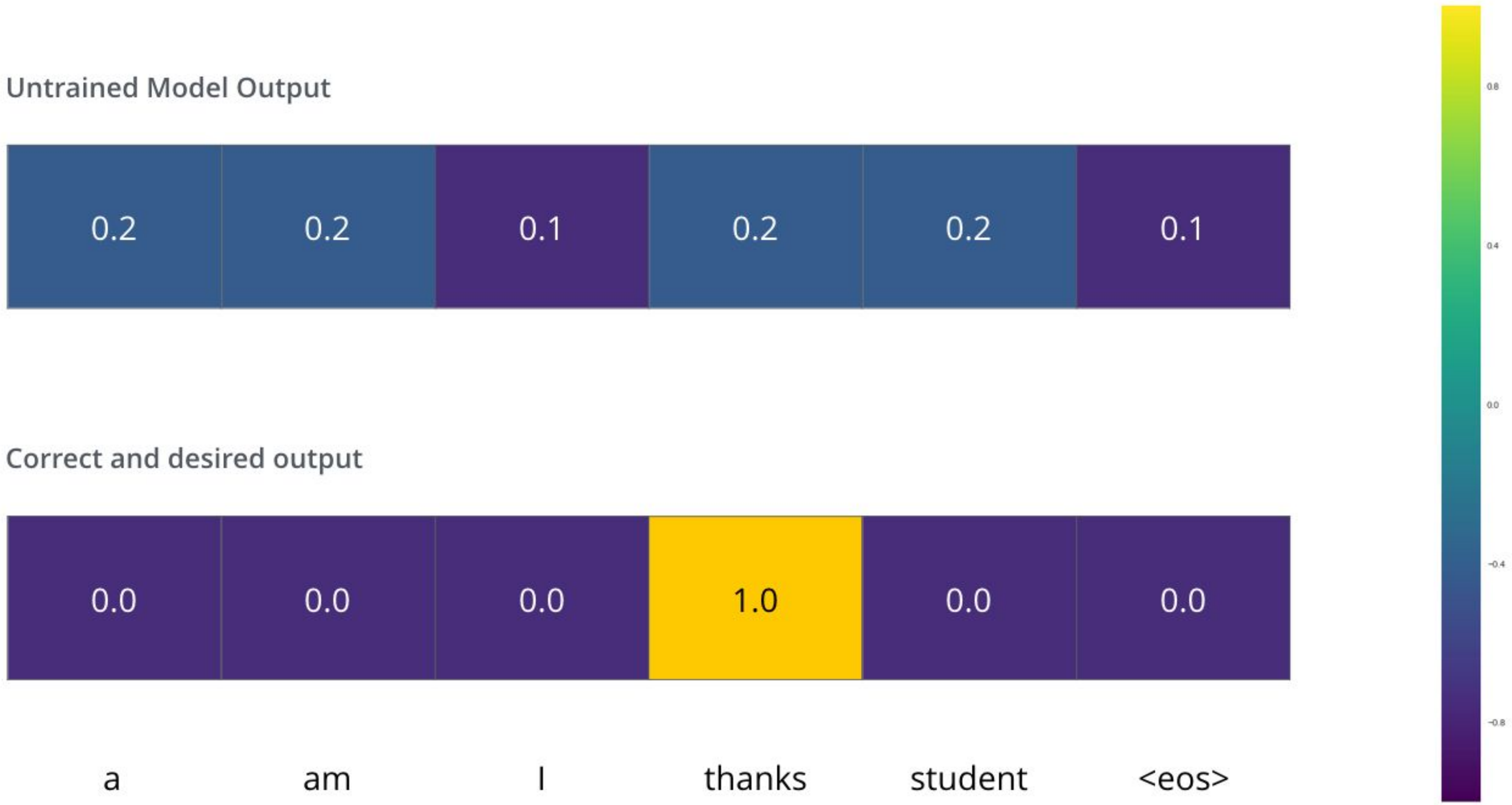
Output Vocabulary

WORD	a	am	I	thanks	student	<eos>
INDEX	0	1	2	3	4	5

One-hot encoding of the word "am"

0.0	1.0	0.0	0.0	0.0	0.0

Example


Untrained Model Output
0.2
0.2
0.1
0.2
0.2
0.1
Correct and desired output
0.0
0.0
0.0
1.0
0.0
0.0
a
am
I
thanks
student
<eos>

Example

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
	a	am	I	thanks	student	<eos>

Example

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

a am I thanks student <eos>

References

- [Understanding Long Short-Term Memory Networks (LSTMs) - Chris Olah]
- [The Illustrated Transformer - Jay Alammar]
- [Attention is all you need]

Summary

- Transformers
  - Self-Attention, Softmax,
  - Multi-Headed
  - Self-Attention,
  - Positional Encoding,
  - Residual Connection,
  - Layer Normalization,
  - Encoder-Decoder