Attention, Transformer and BERT

Prof. Kuan-Ting Lai

2020/6/16

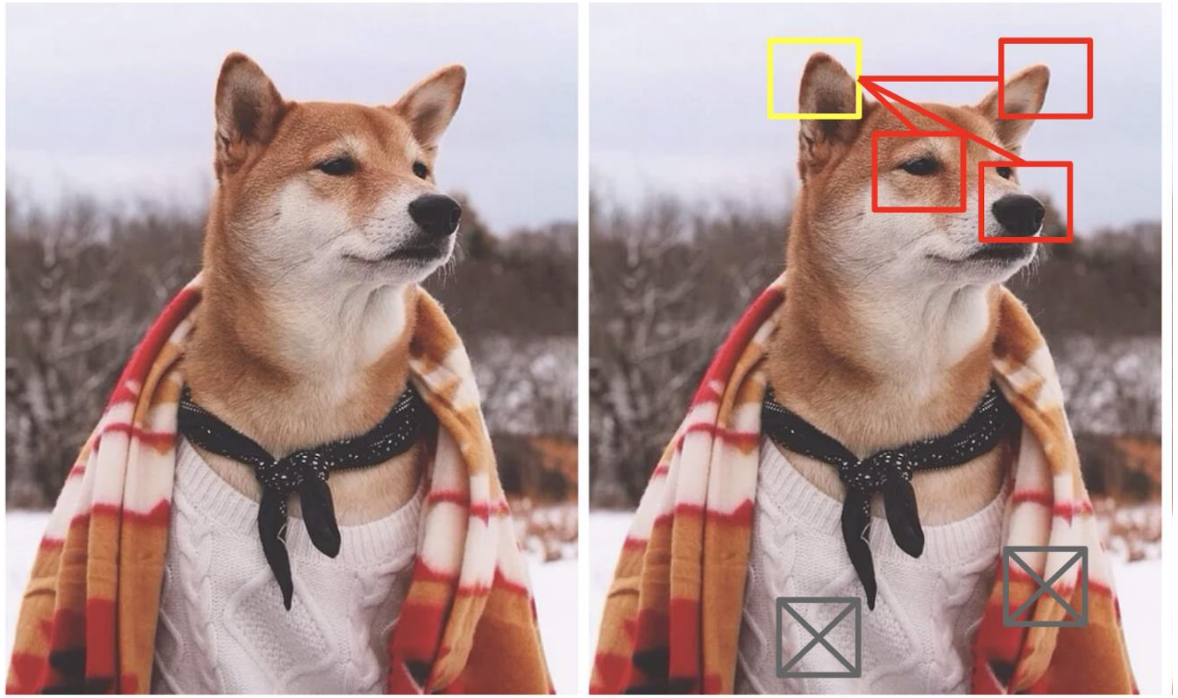
Attention is All You Need!

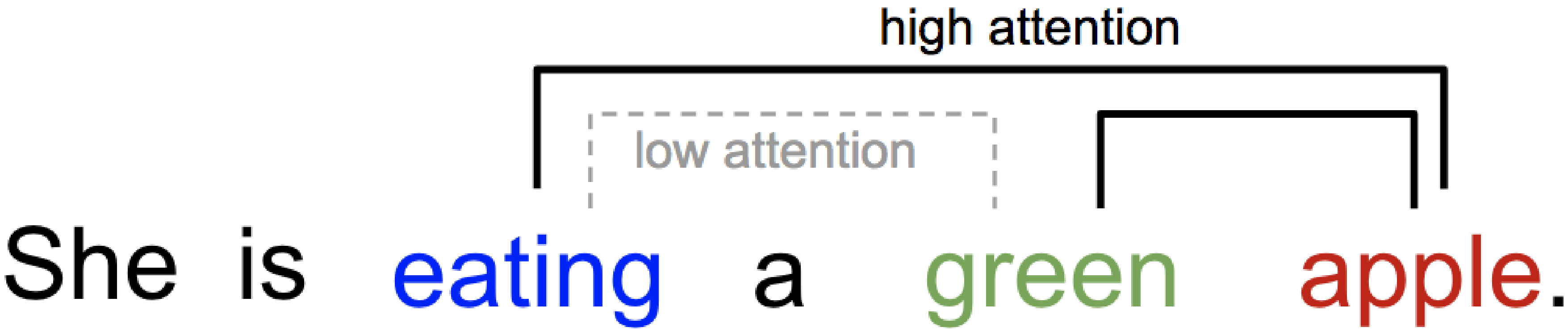
A. Waswani et al., *NIPS*, 2017

Google Brain & University of Toronto

Attention

- Visual attention and textual attention





high attention

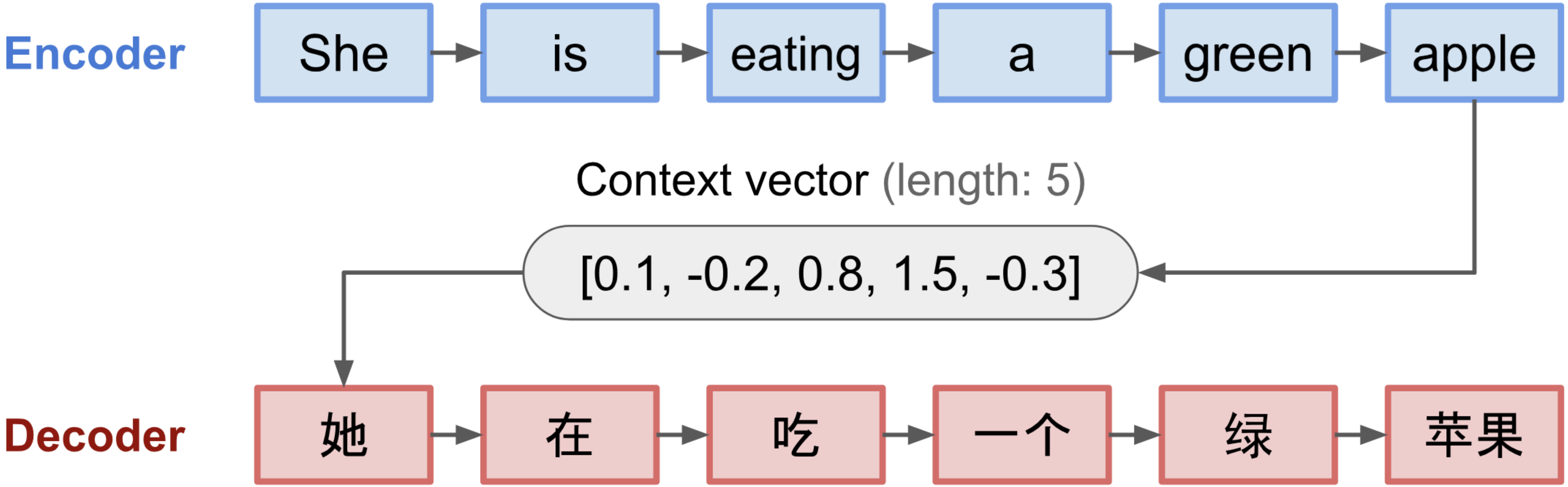
low attention

She is eating a green apple.

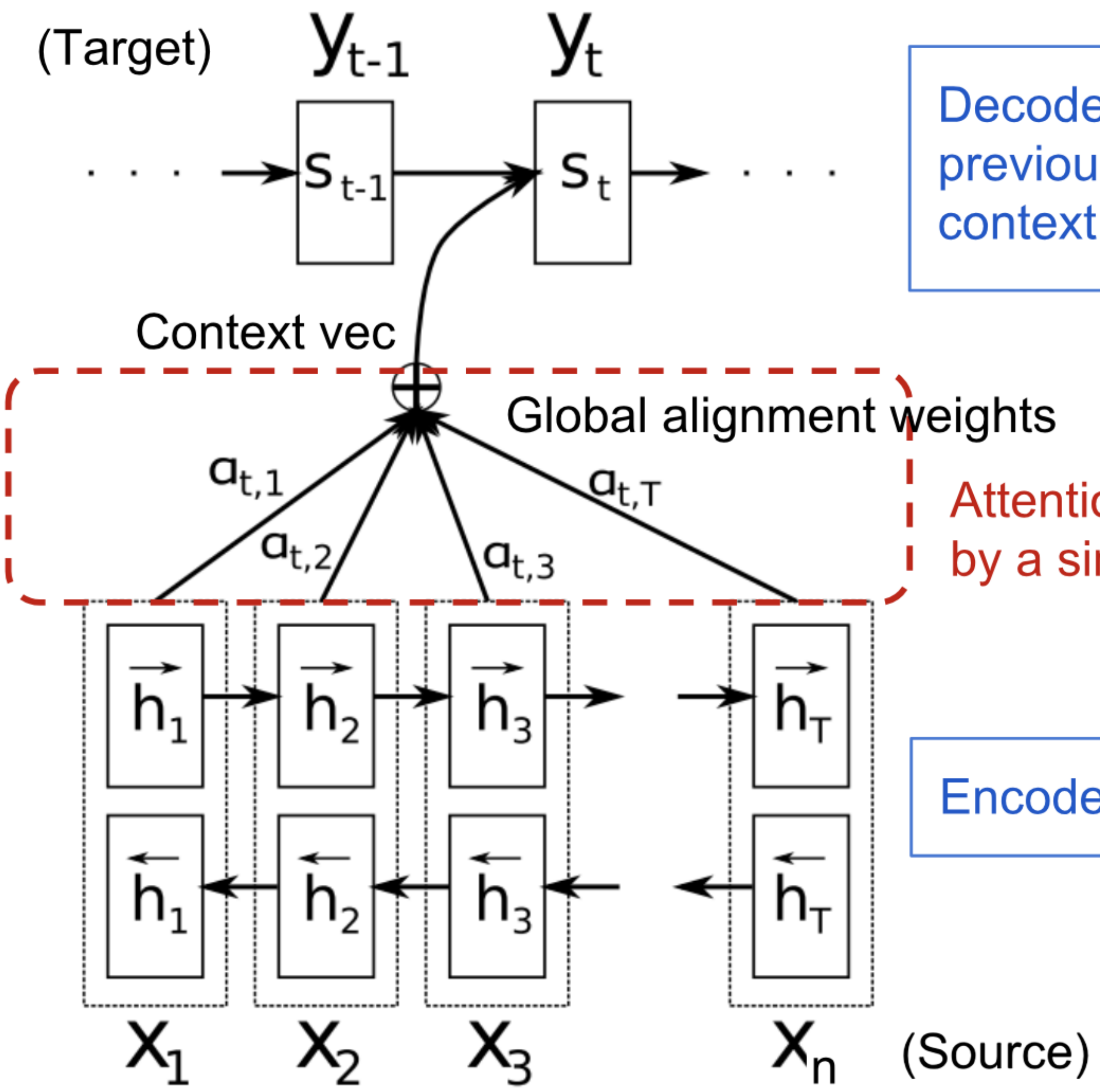
https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html

Seq2seq model

- Language translation



Attention = Vector of Importance Weights



Decoder: RNN with input from previous state + dynamic context vector.

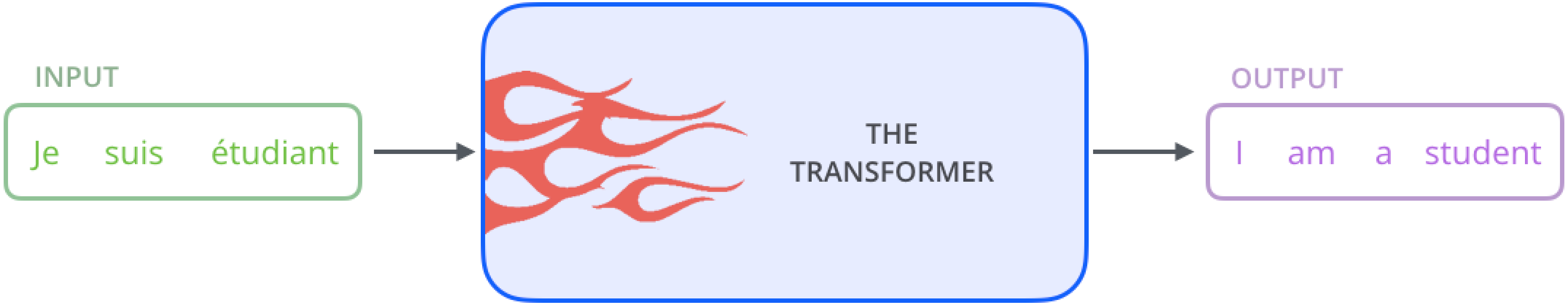
Attention layer: parameterized by a simple feed-forward network

Additive Attention

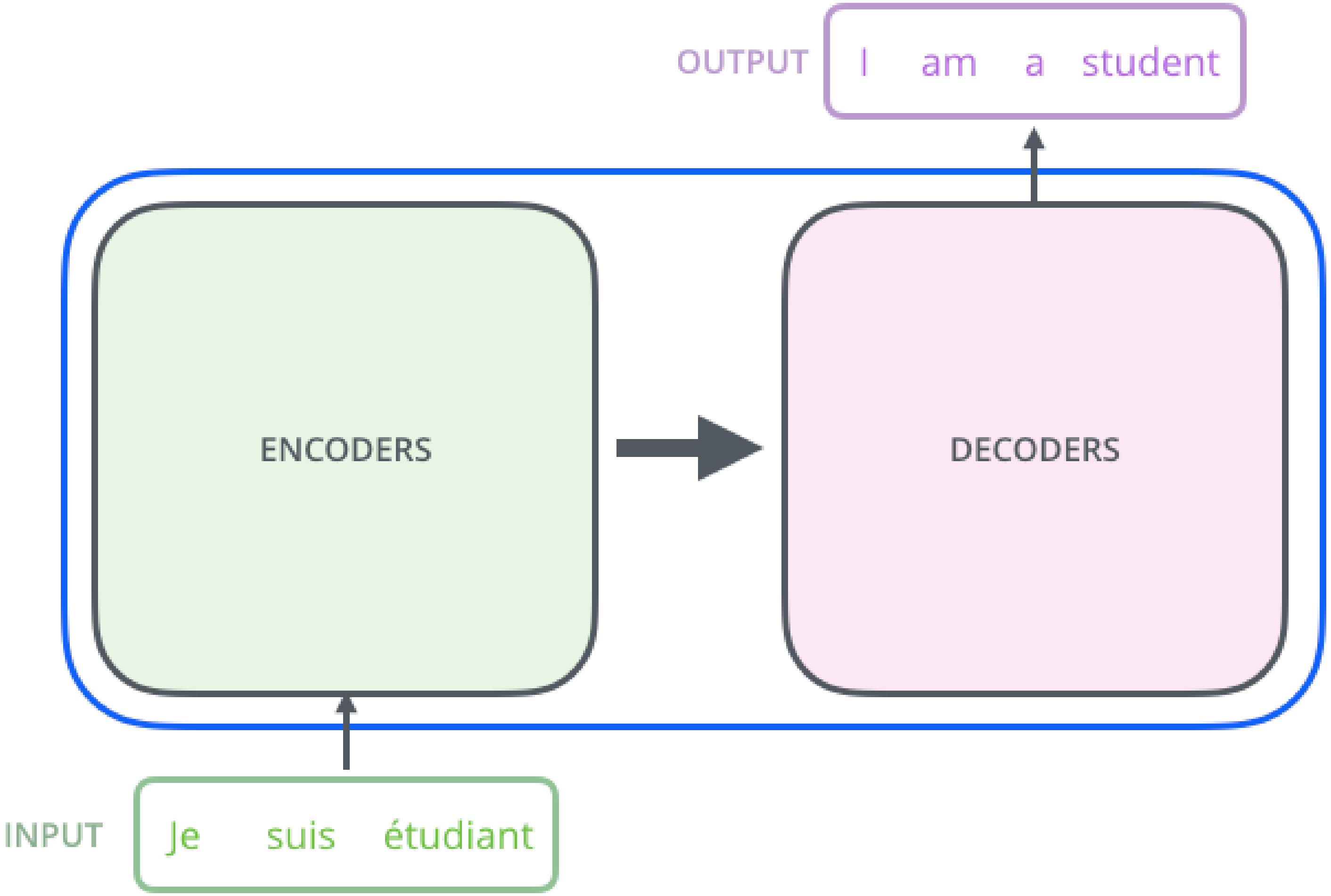
Encoder: bidirectional RNN

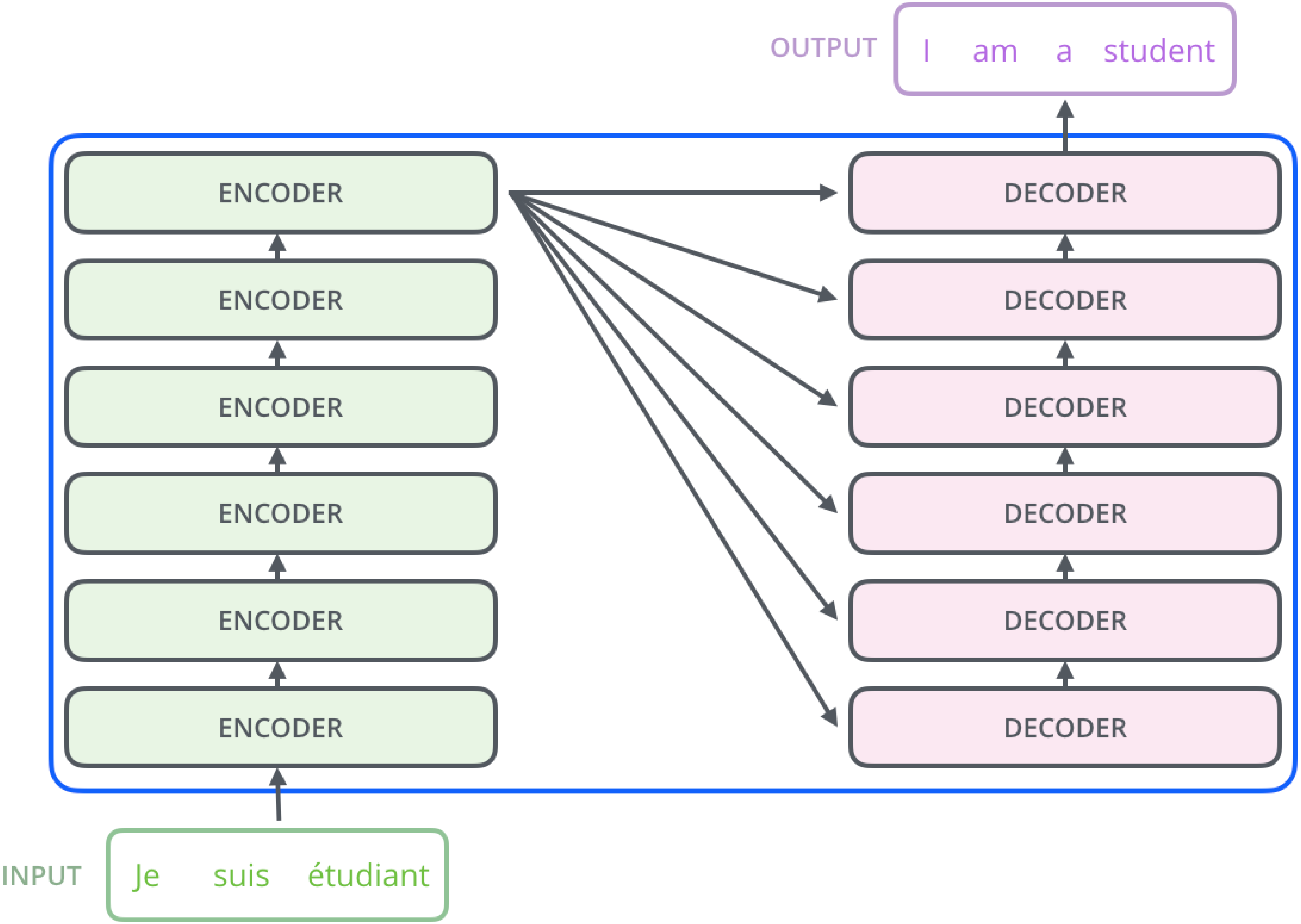
Transformer

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



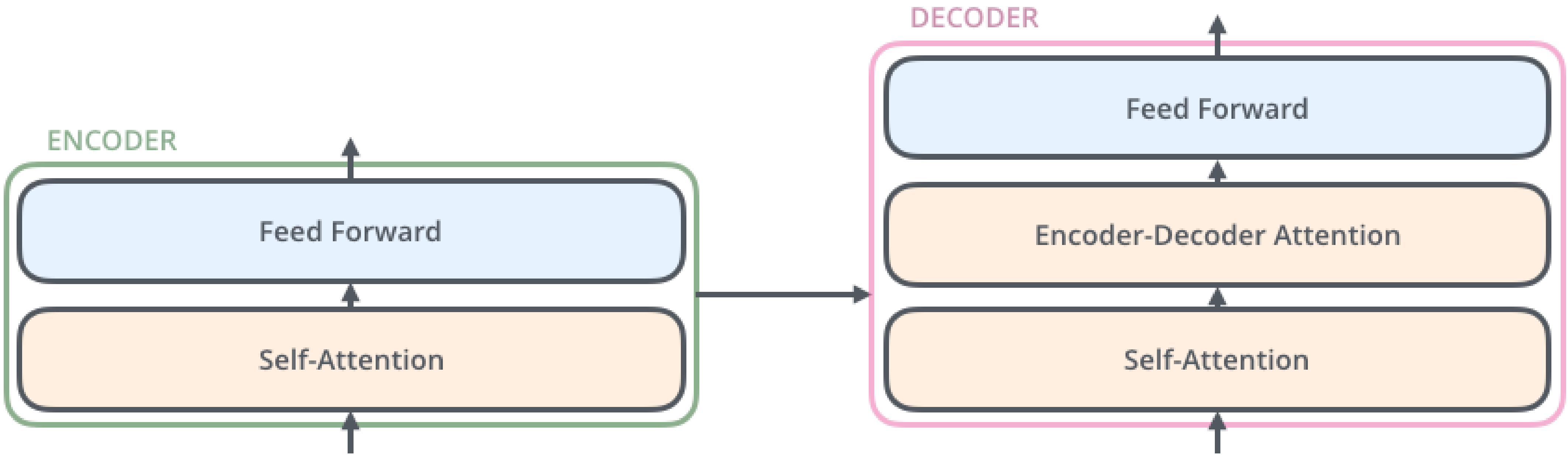
Encoder and Decoder

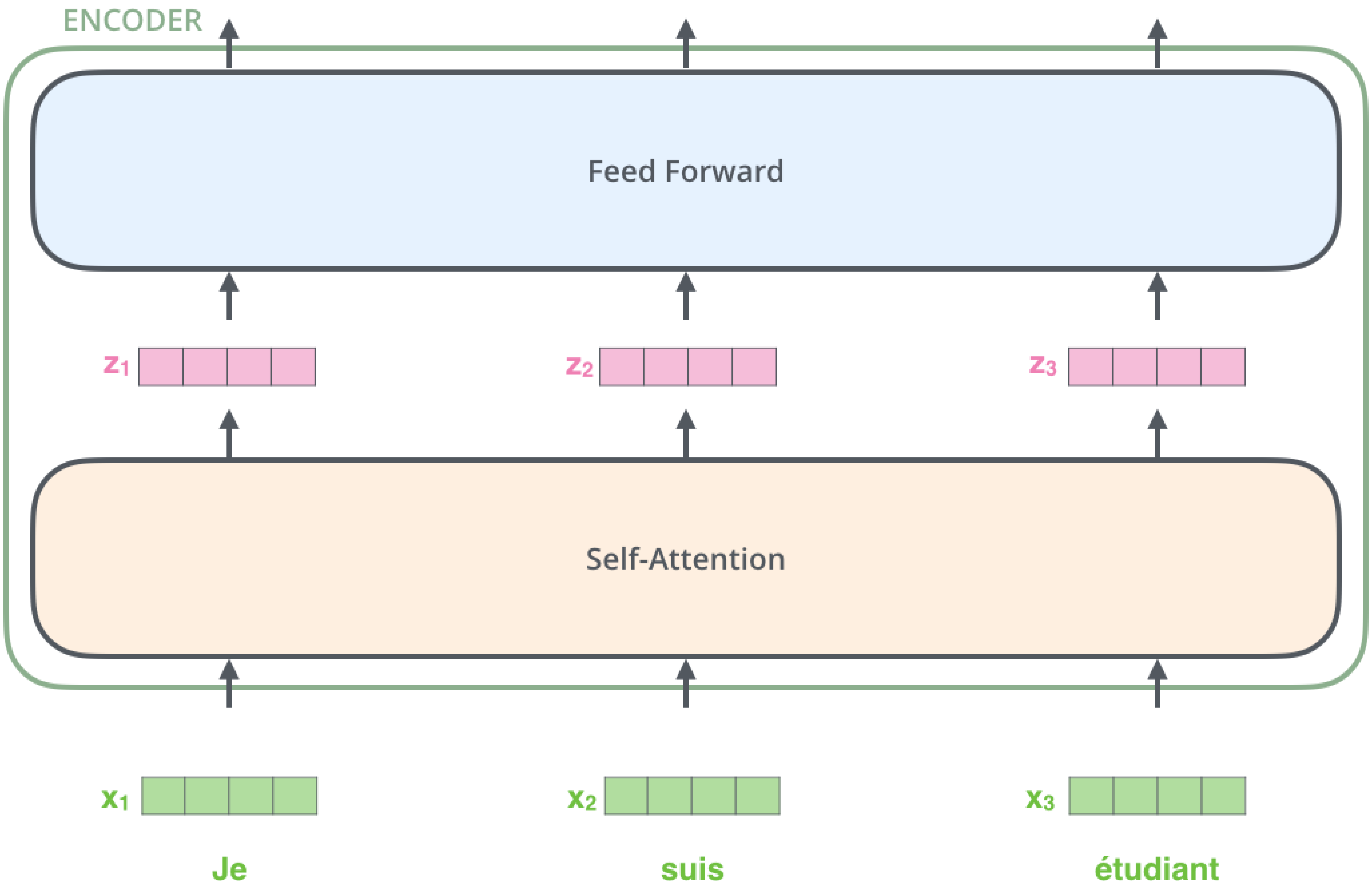



OUTPUT
I am a student
ENCODER
ENCODER
ENCODER
ENCODER
ENCODER
ENCODER
DECODER
DECODER
DECODER
DECODER
DECODER
DECODER
INPUT
Je suis étudiant

Structure of the Encoder and Decoder

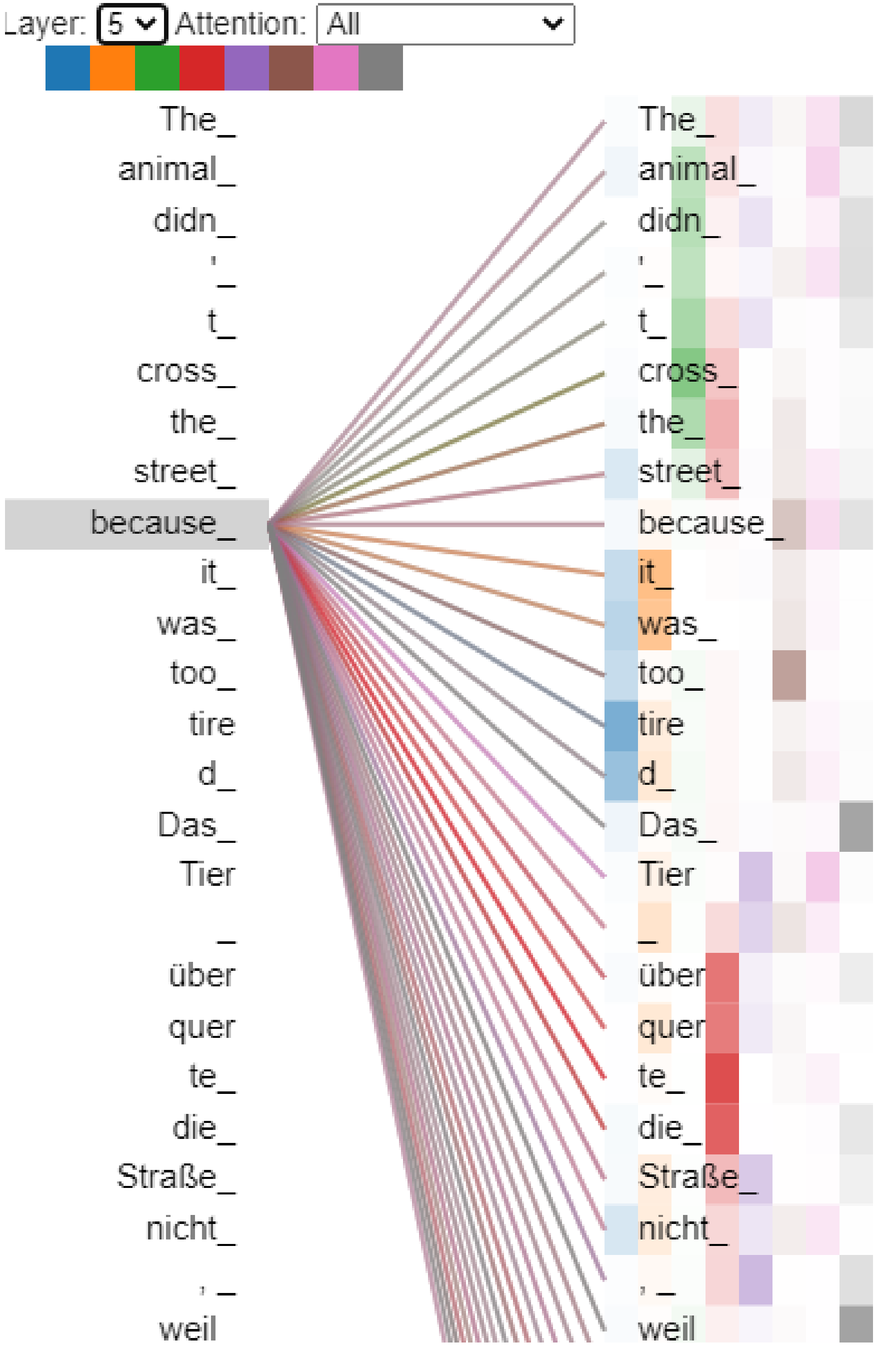
- Self-attention
- Encoder-decoder attention




ENCODER
Feed Forward
z_1
z_2
z_3
Self-Attention
x_1
x_2
x_3
Je
suis
étudiant

Tensor2Tensor Notebook

- https://colab.research.google.com/github/tensorflow/tensor2tensor/blob/master/tensor2tensor/notebooks/hello_t2t.ipynb



Layer: 5 Attention: All

Self-attention (query, key, value)

q: query (to match others)

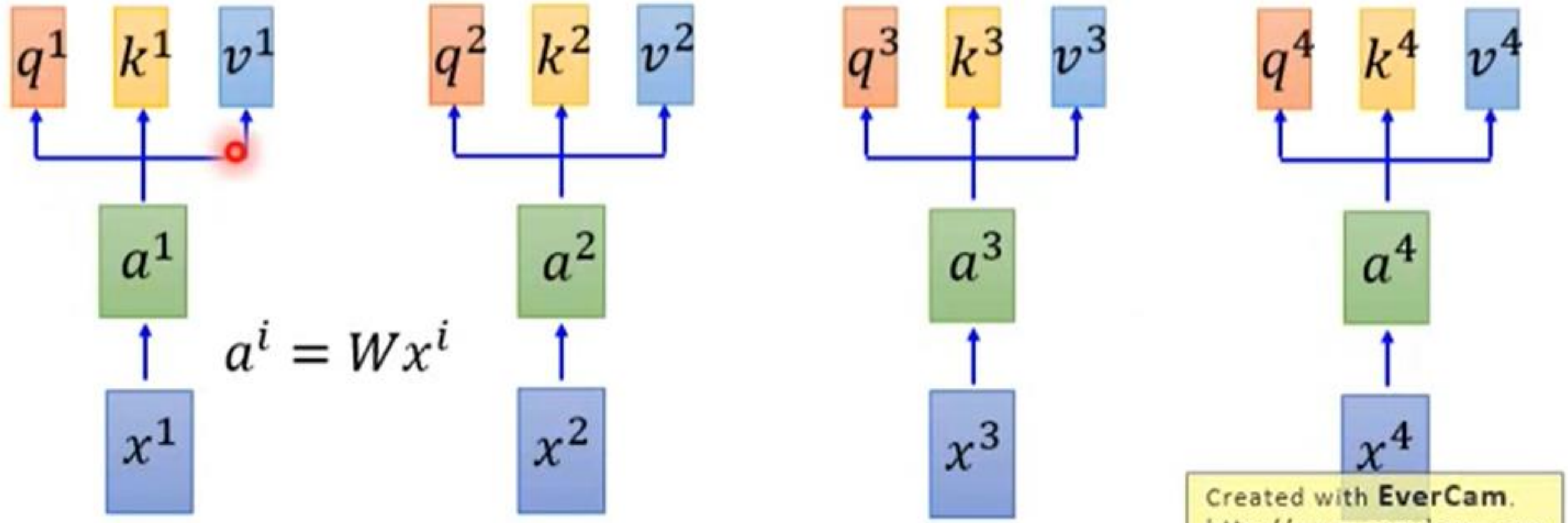
$$q^i = W^q a^i$$

k: key (to be matched)

$$k^i = W^k a^i$$

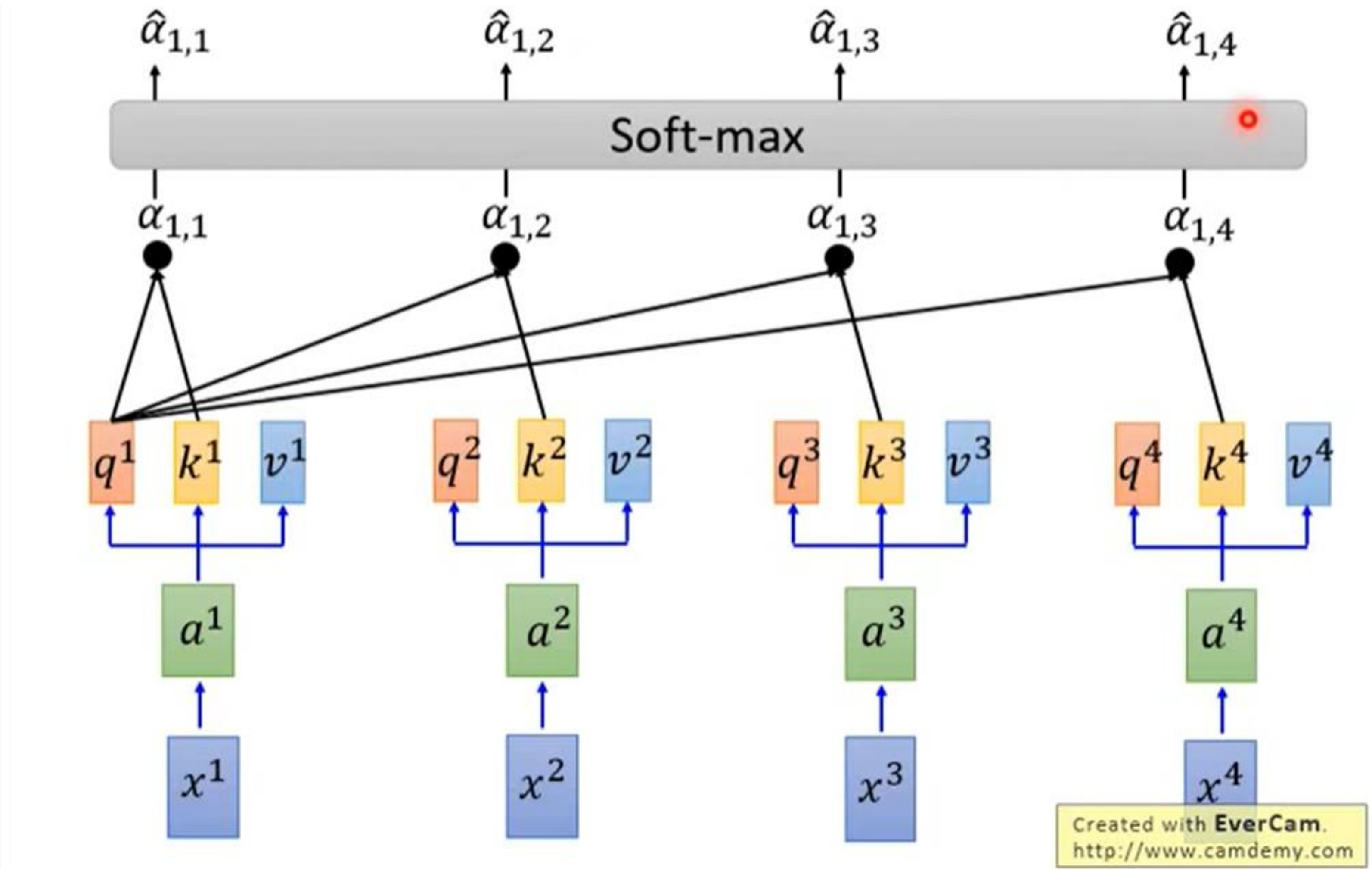
v: information to be extracted

$$v^i = W^v a^i$$



https://www.youtube.com/watch?v=ugWDIIOHtPA&t=1089s

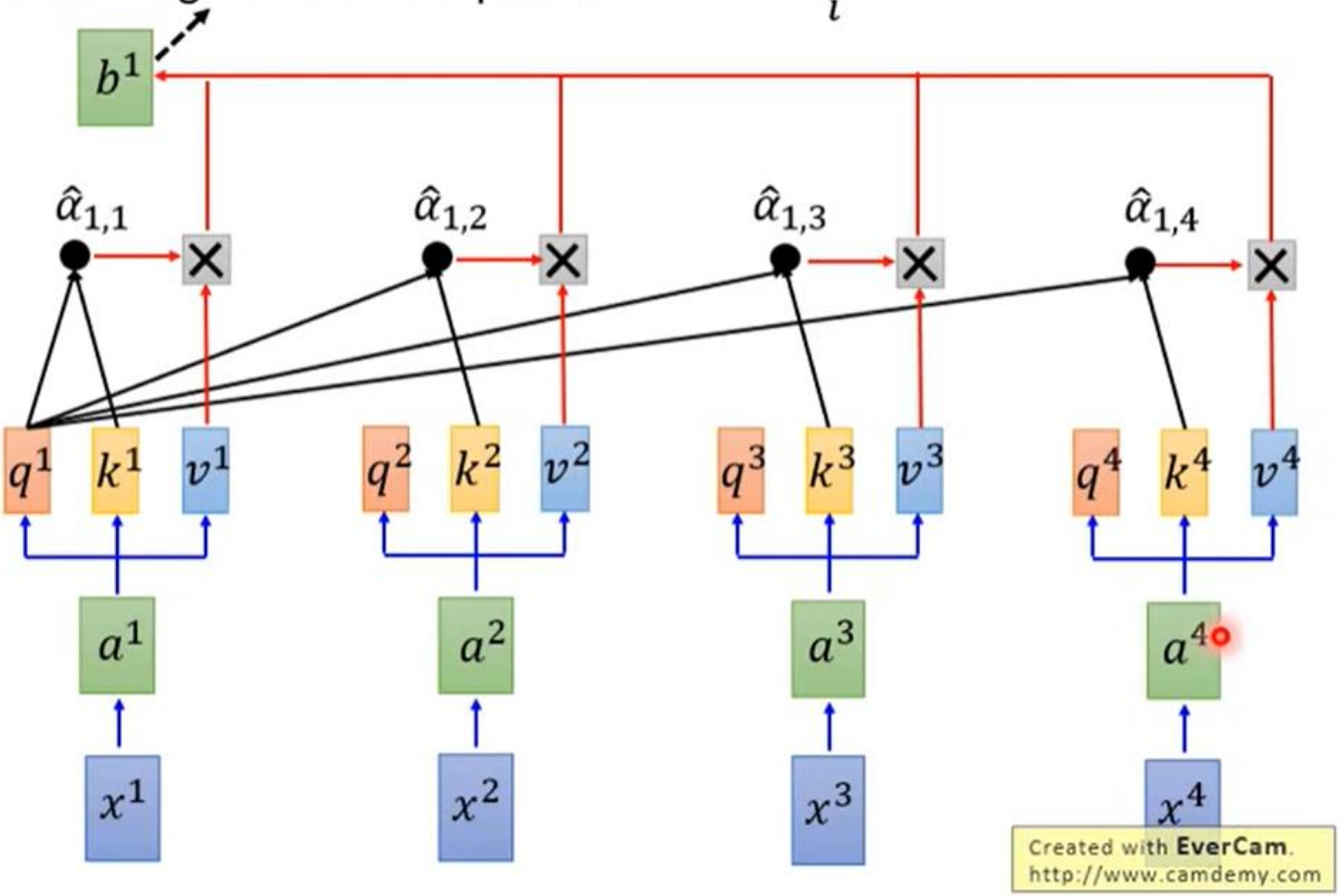
Self-attention



Self-attention

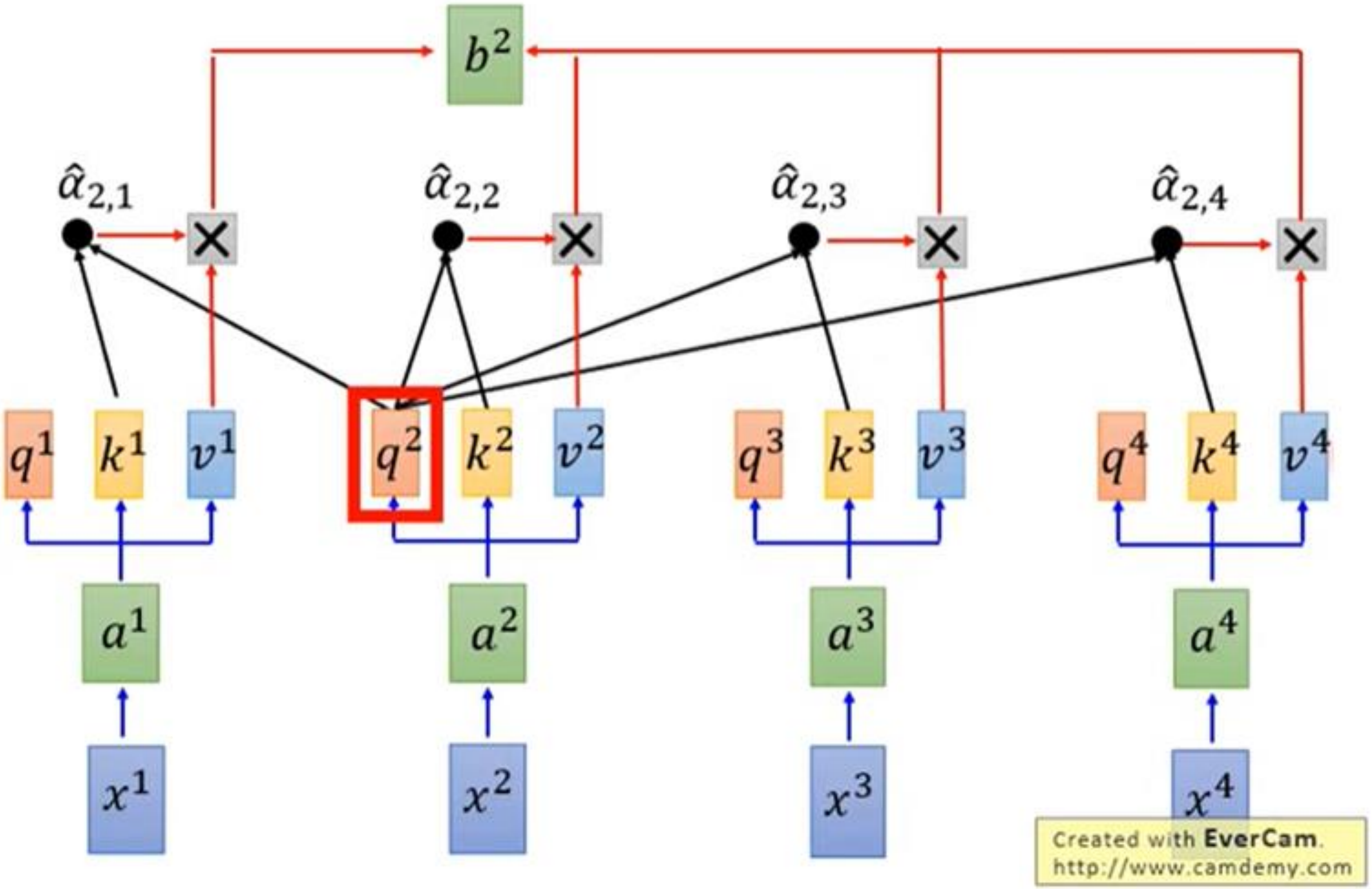
Considering the whole sequence

$$b^1 = \sum_i \hat{\alpha}_{1,i} v^i$$

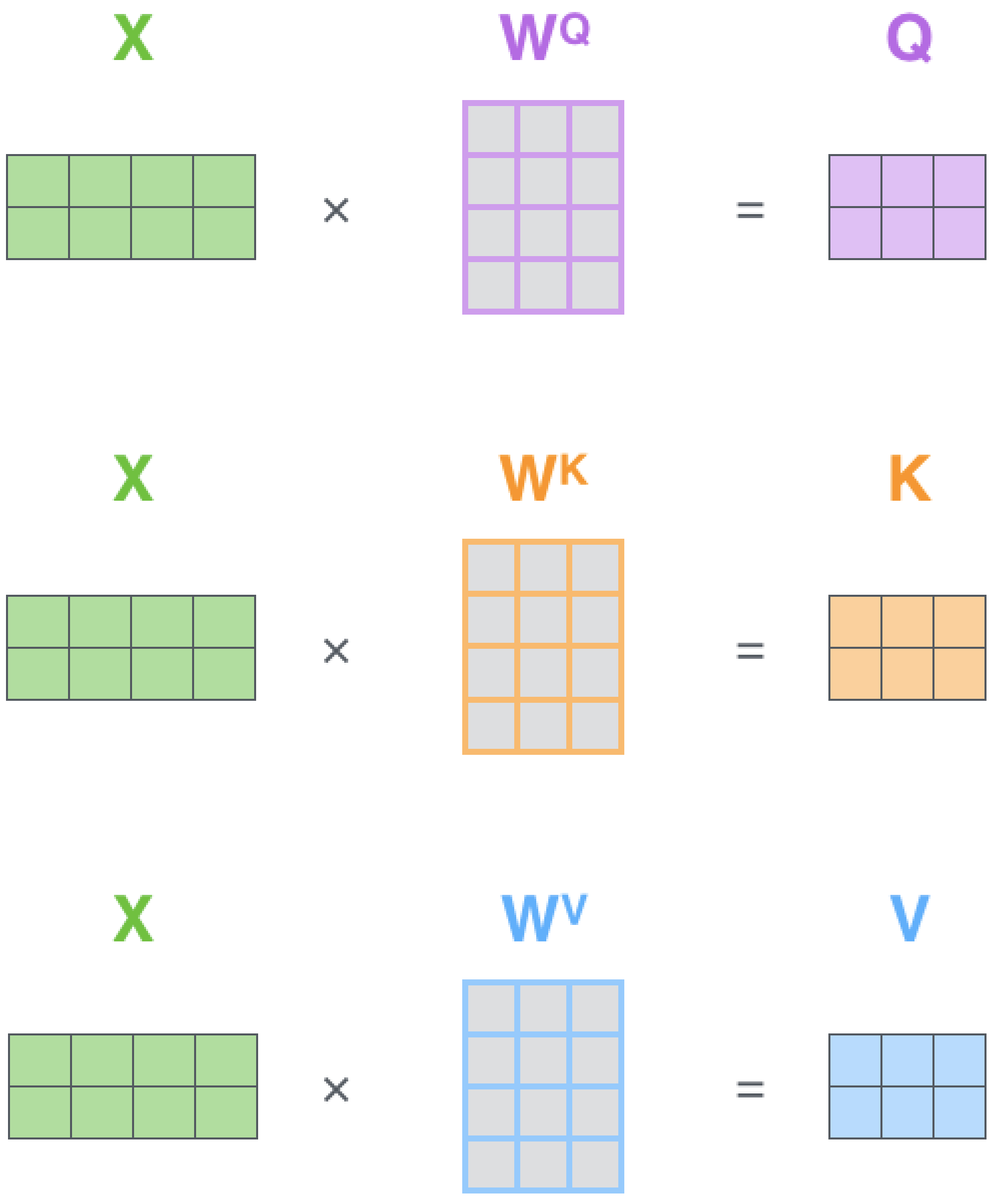


Calculating b^2

$$b^2 = \sum_i \hat{\alpha}_{2,i} v^i$$



Matrix Mutiplication



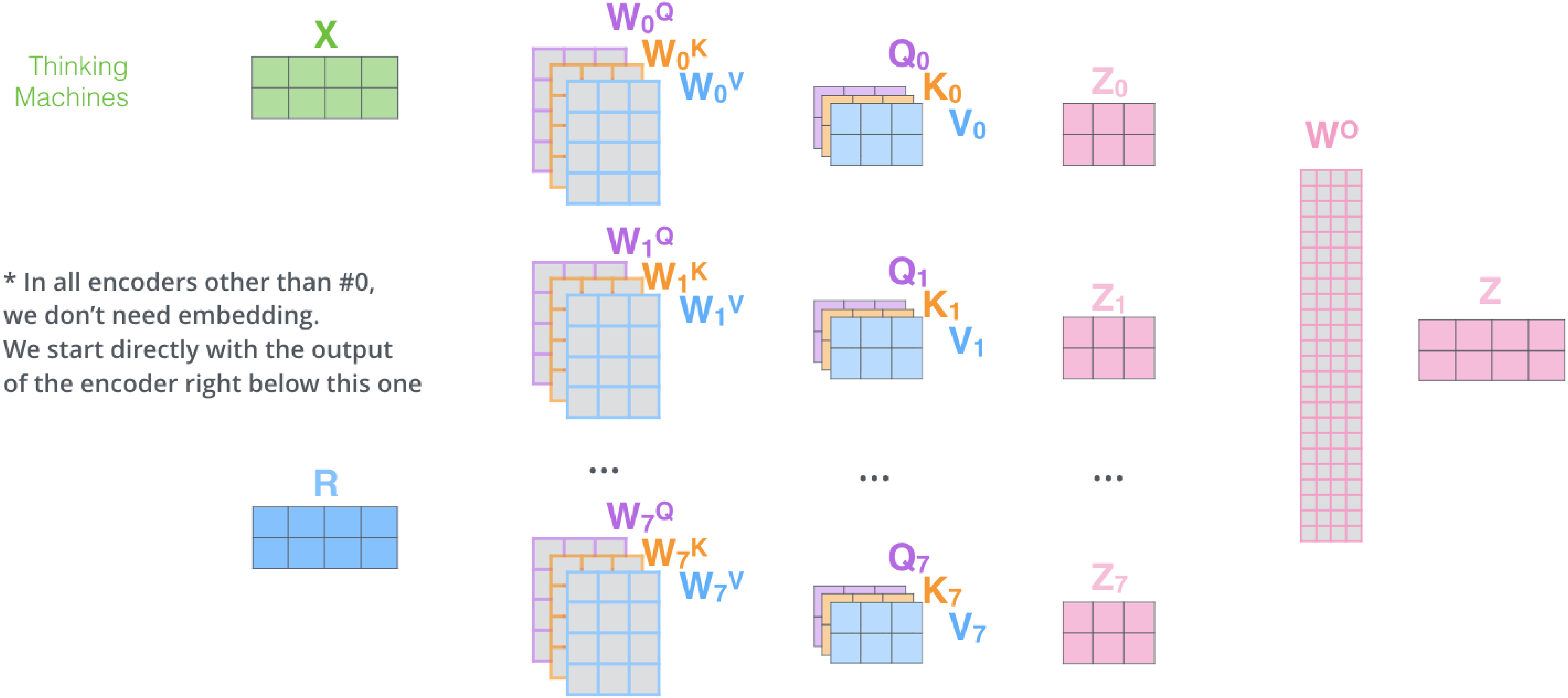
1) This is our input sentence*

2) We embed each word*

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

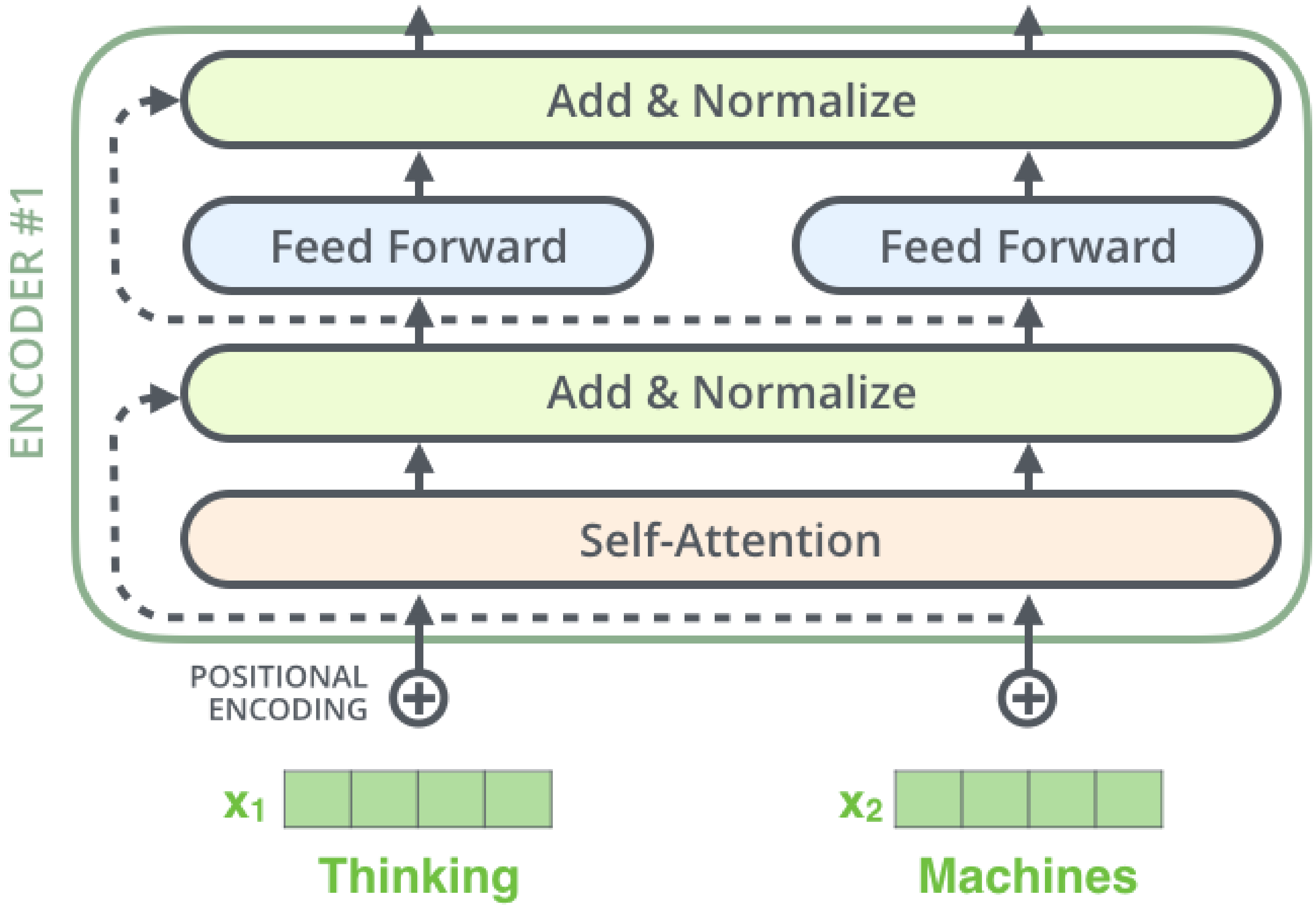
4) Calculate attention using the resulting Q/K/V matrices

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

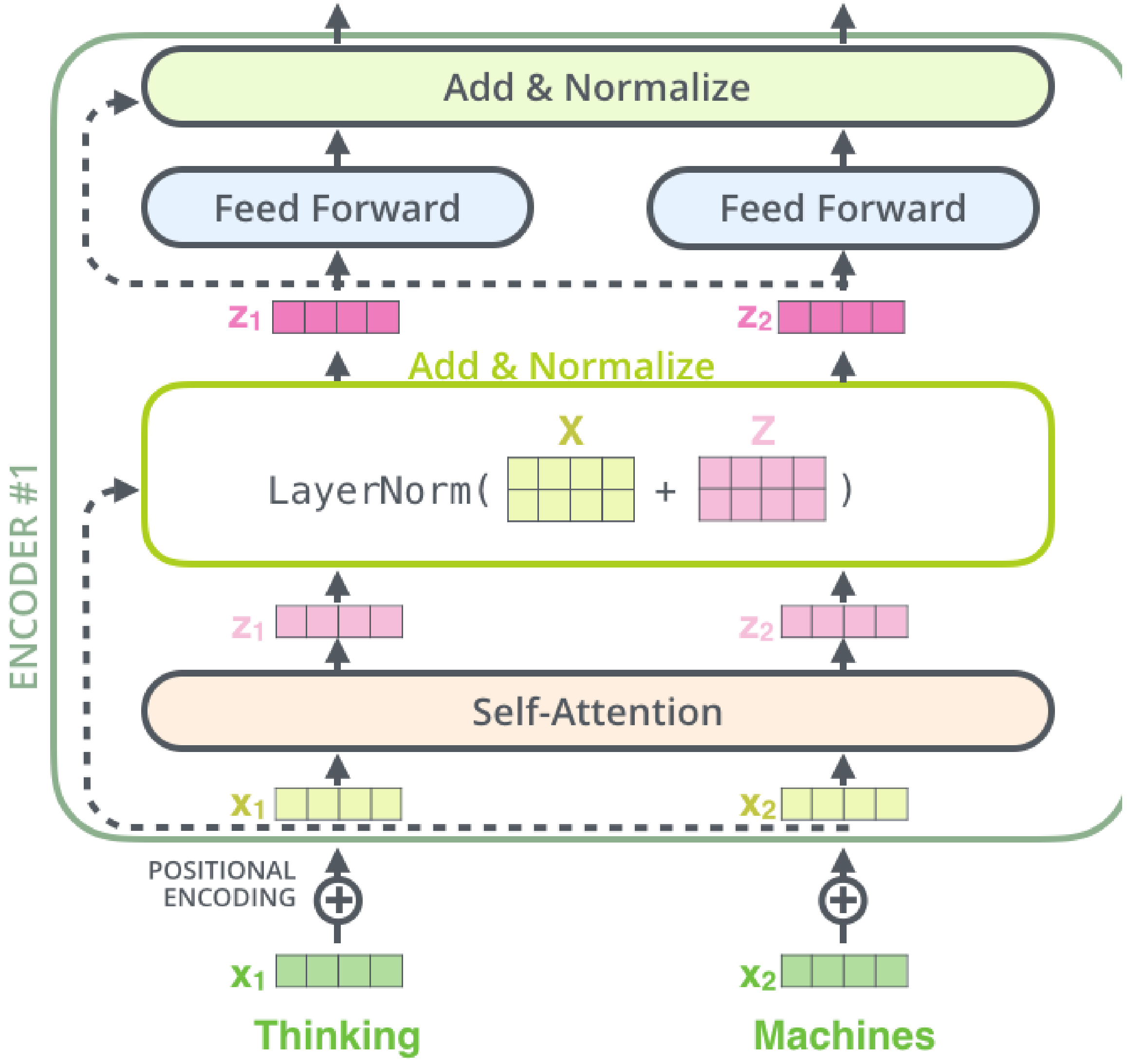


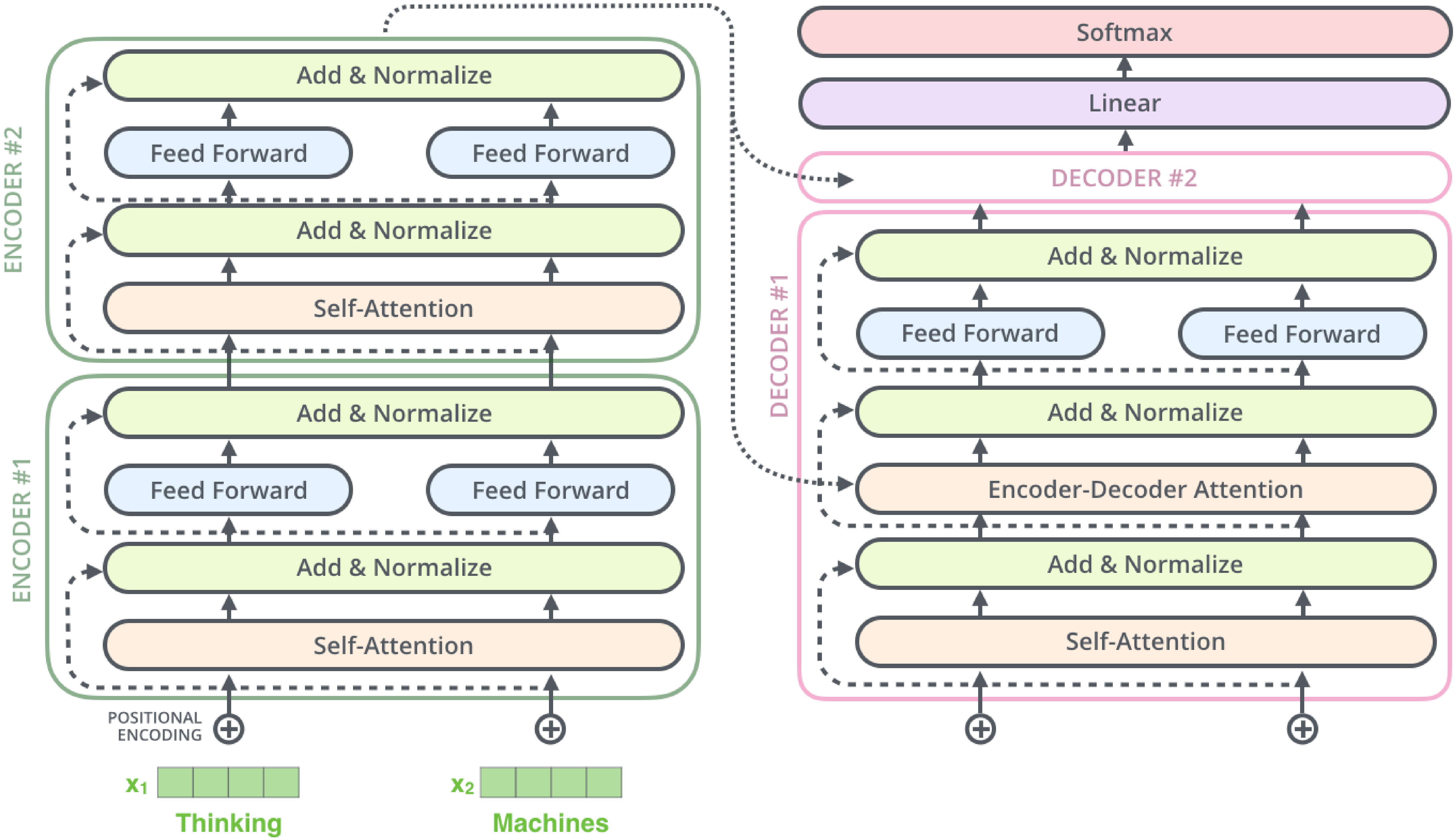
* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one

Adding Residual Connections



Layer Normalization




Softmax
Linear
DECODER #2
ENCODER #2
Add & Normalize
Feed Forward
Feed Forward
Add & Normalize
Self-Attention
DECODER #1
Add & Normalize
Feed Forward
Feed Forward
Add & Normalize
Encoder-Decoder Attention
Add & Normalize
Self-Attention
ENCODER #1
Add & Normalize
Feed Forward
Feed Forward
Add & Normalize
Self-Attention
POSITIONAL ENCODING
x_1
x_2
Thinking
Machines

References

1. https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html
2. http://jalammar.github.io/illustrated-transformer/
3. Hong-Yi Lee, Transformer, 2019
   https://www.youtube.com/watch?v=ugWDIIOHtPA