Week 8 – attention and BERT

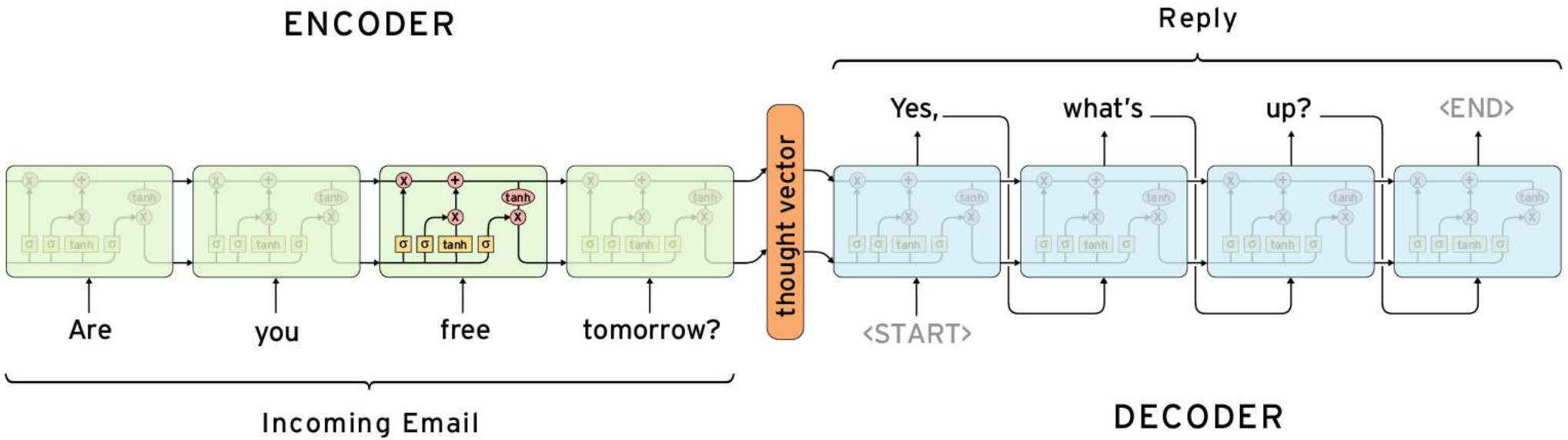
EGCO467 Natural Language and Speech Processing

Sources

- https://arxiv.org/abs/1706.03762 Vaswani, Ashish, et al. "**Attention is all you need.**" Advances in neural information processing systems. 2017.
- http://jalammar.github.io/illustrated-transformer/
- https://www.youtube.com/watch?v=iDulhoQ2pro
- http://nlp.seas.harvard.edu/2018/04/03/attention.html

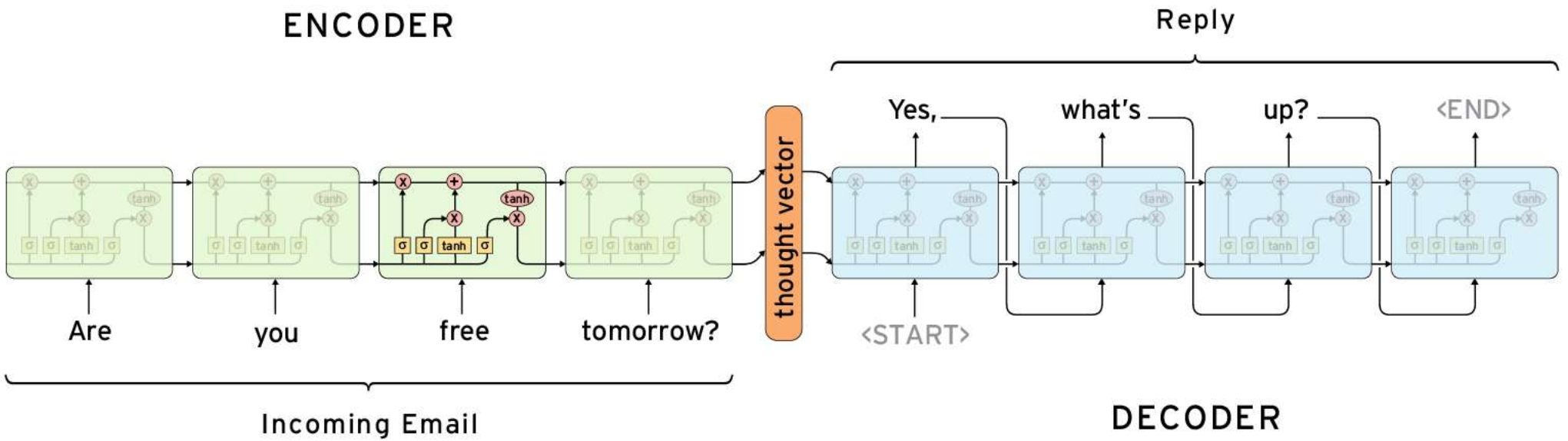
seq2seq

- sequence to sequence model



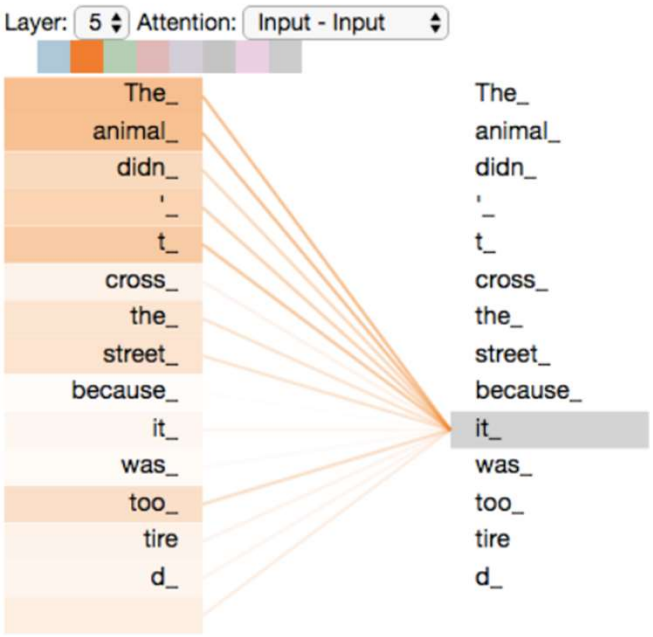
disadvantage of LSTM

- sequential mode – computation can't be parallelized
- all texts that comes before position t already absorbed into state vector h



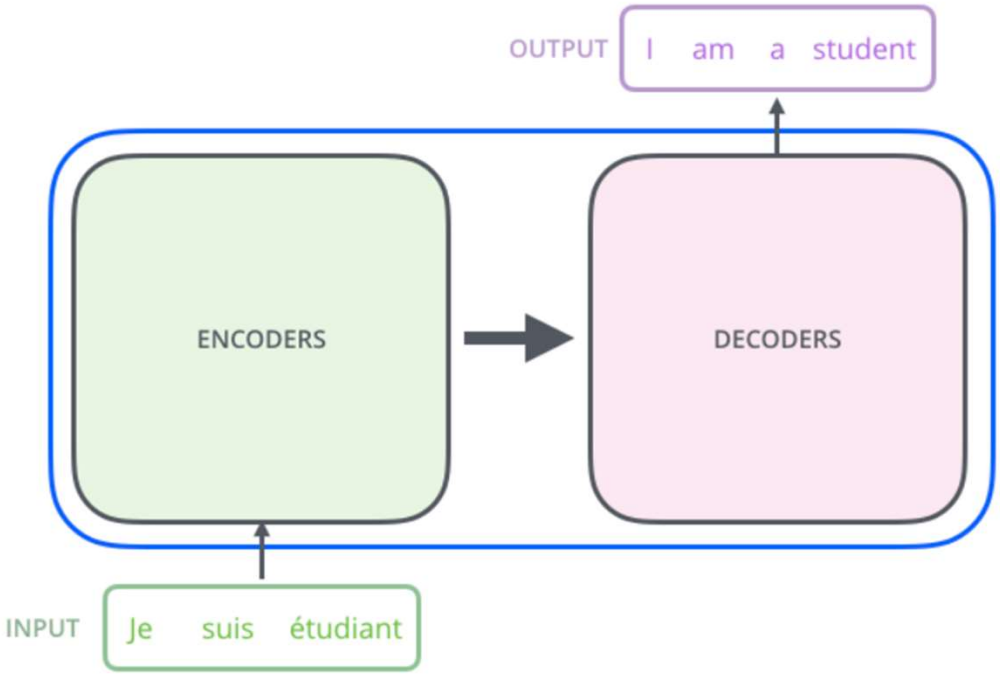
Attention Mechanism

- Consider every pair of token
- If n tokens, there are n^2 pairs

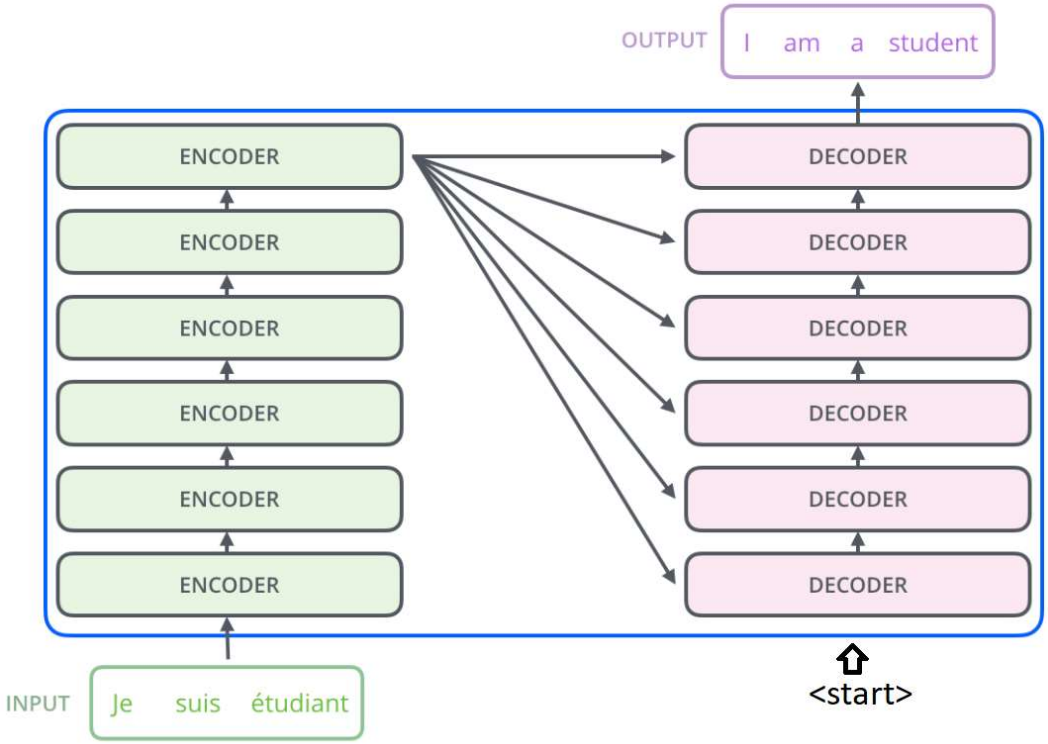


encoder-decoder (seq2seq)

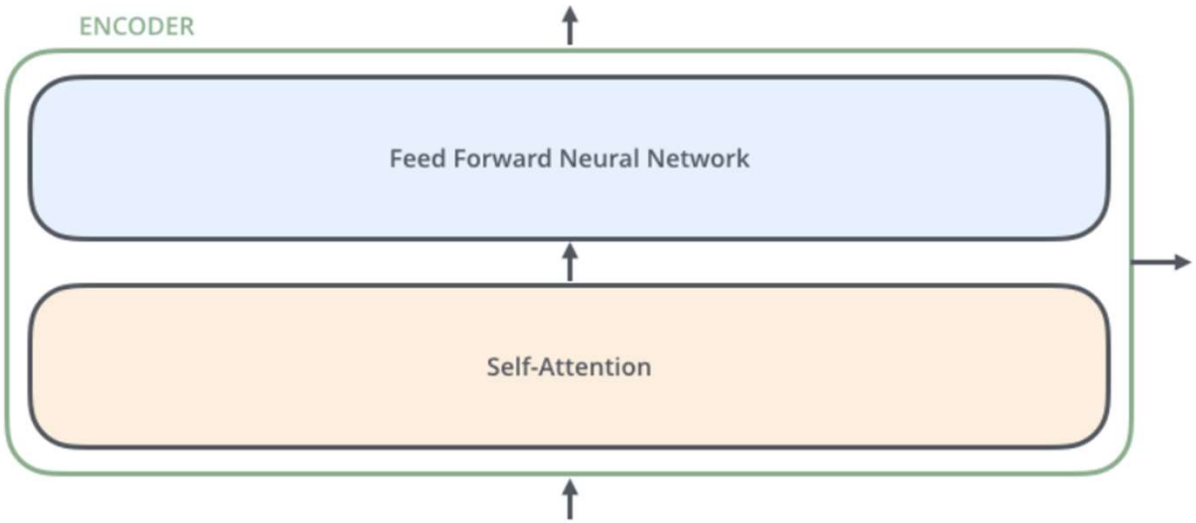
encoder-decoder



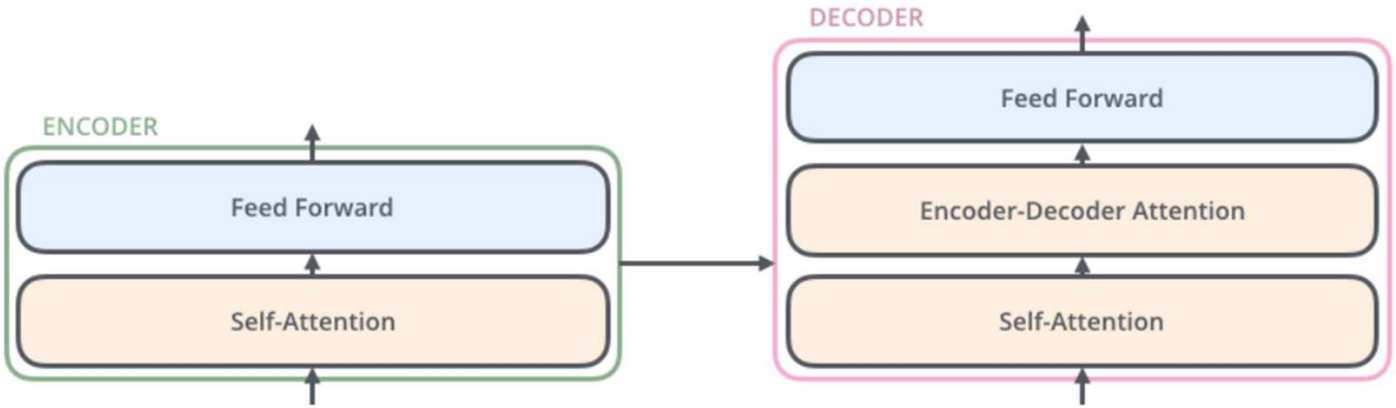
encoder-decoder



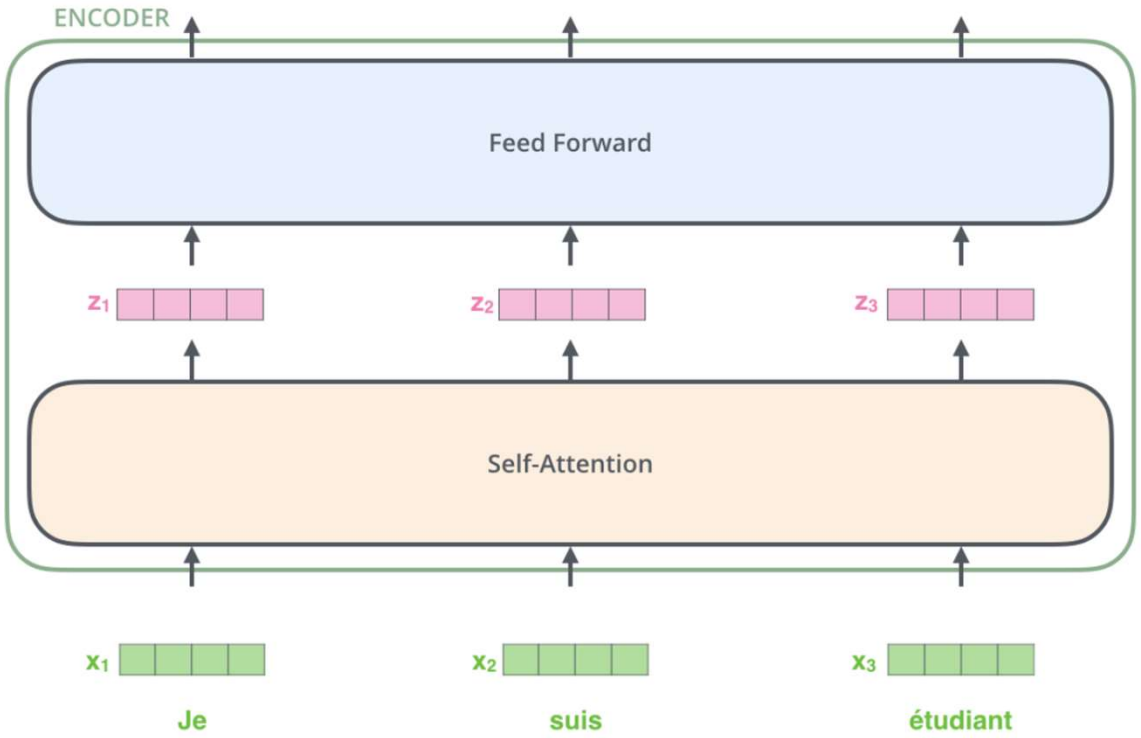
encoder



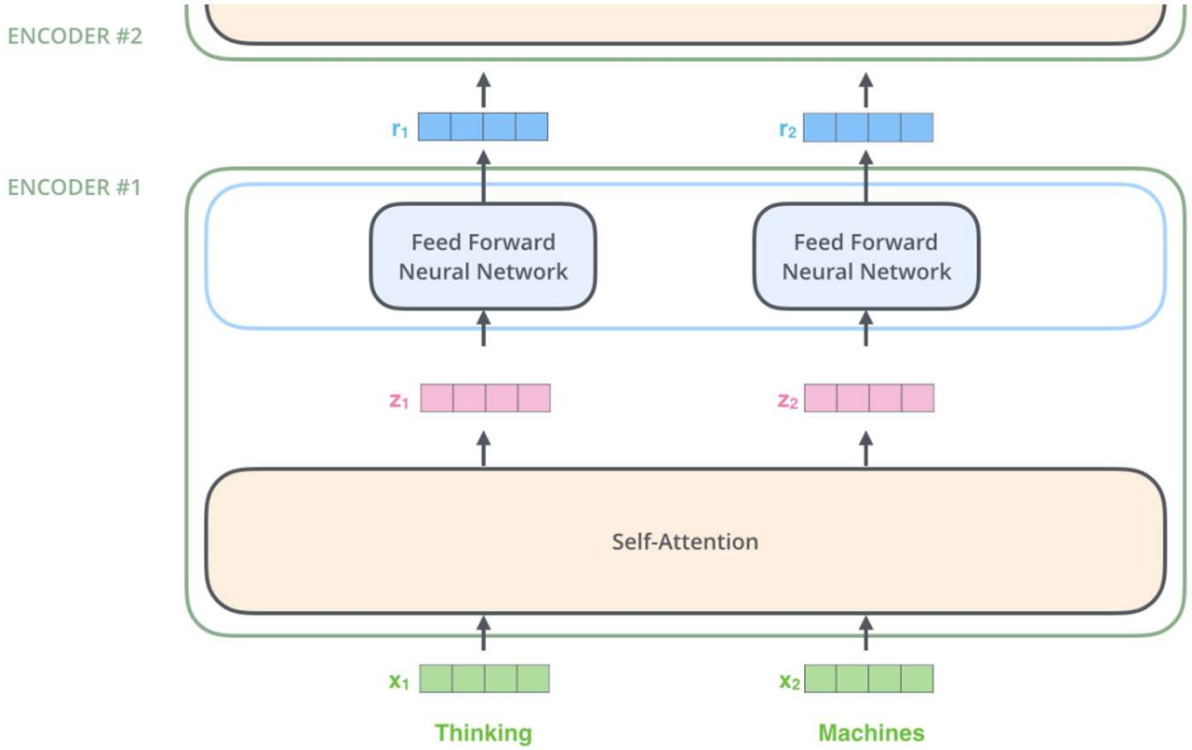
self-attention vs. “normal” attention



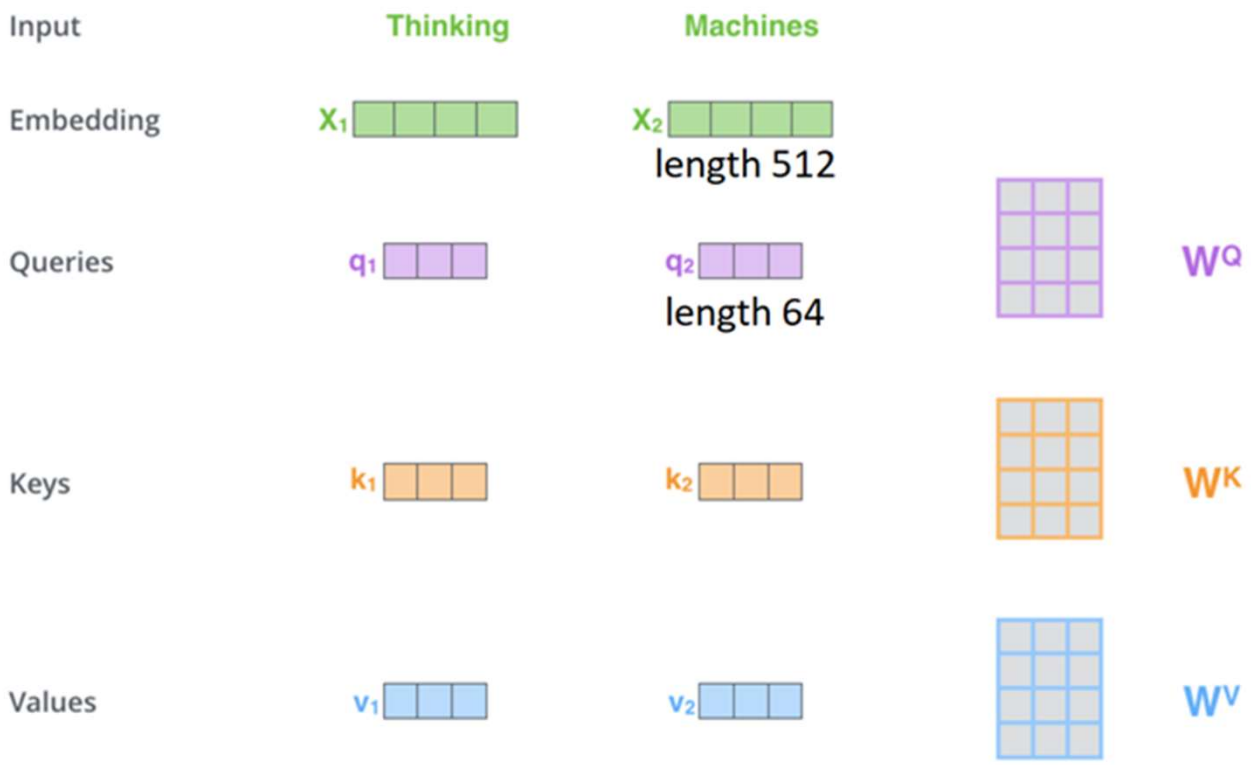
First Layer


ENCODER
Feed Forward
z1
z2
z3
Self-Attention
x1
x2
x3
Je
suis
étudiant

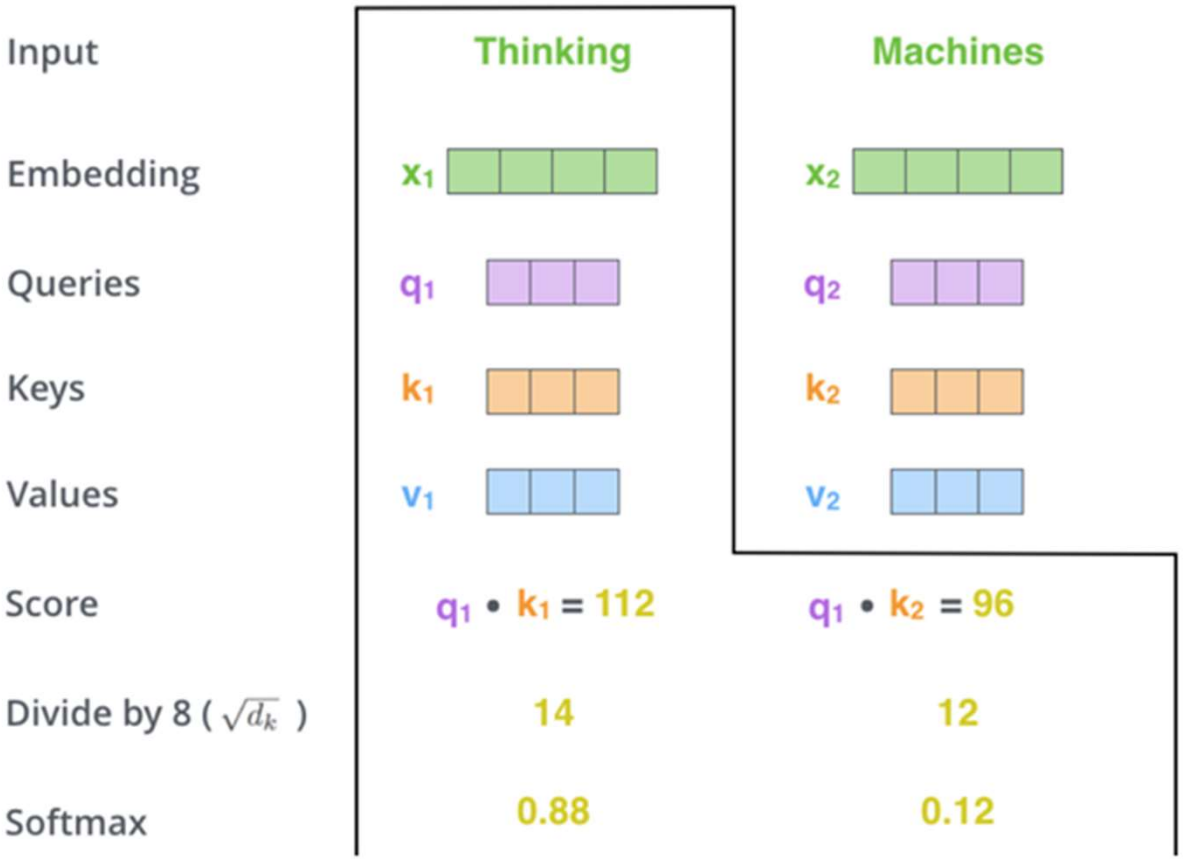
Inside First Layer



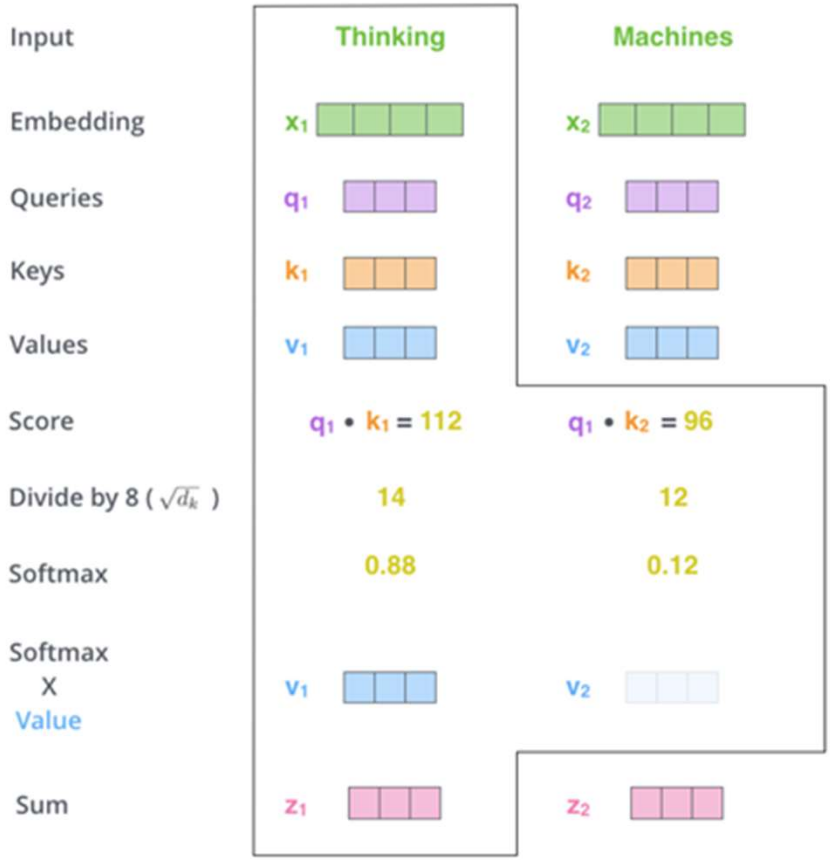
Self attention



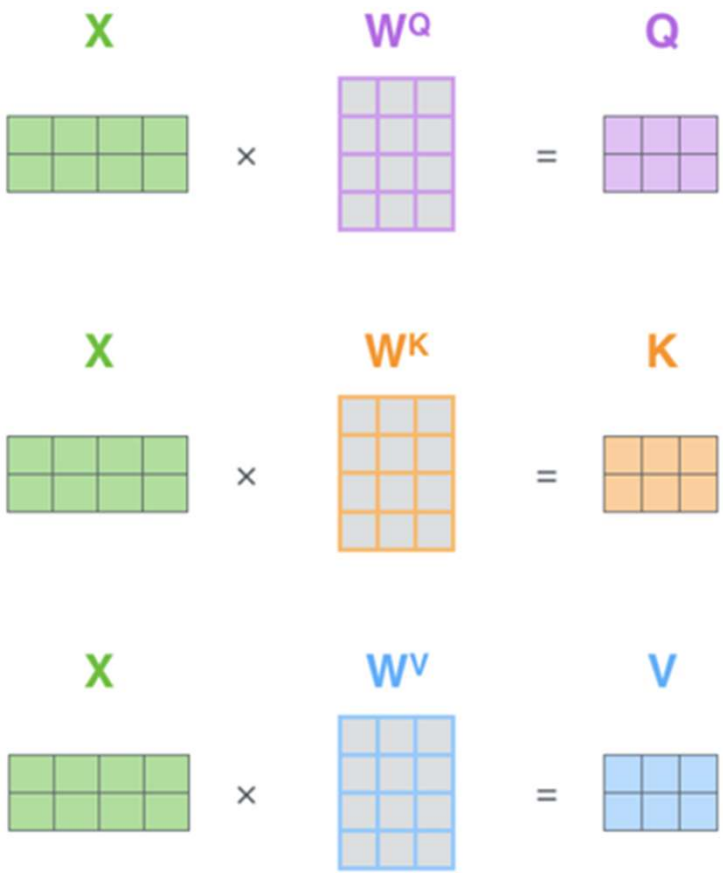
Self attention



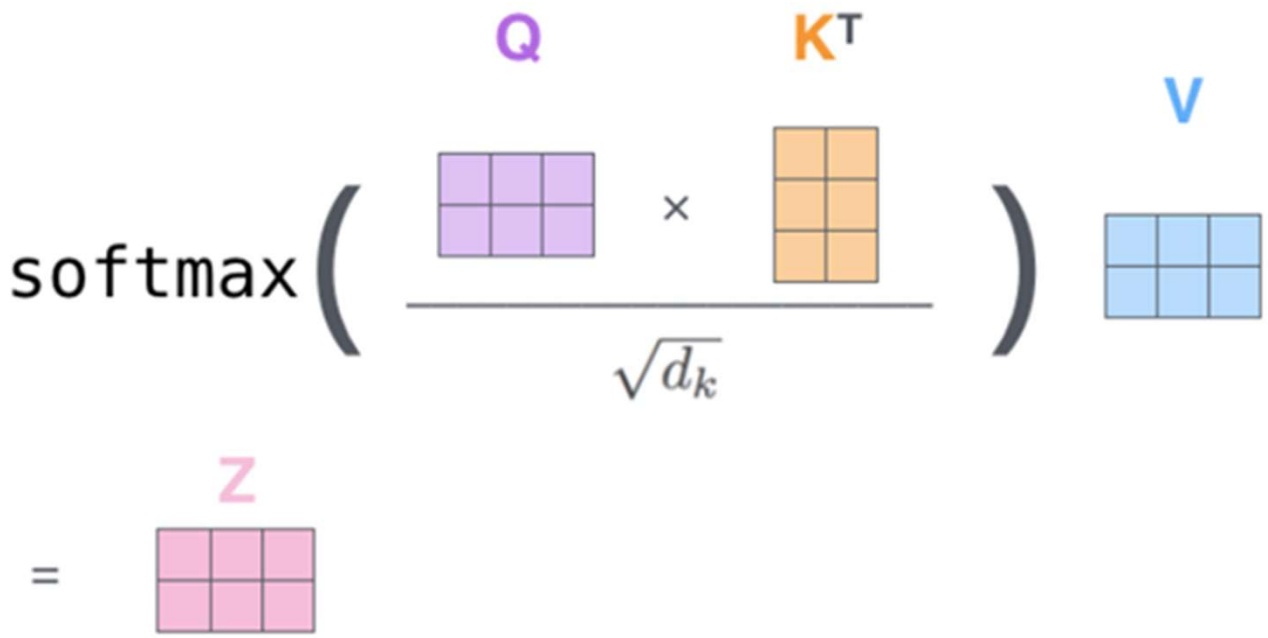
Self attention



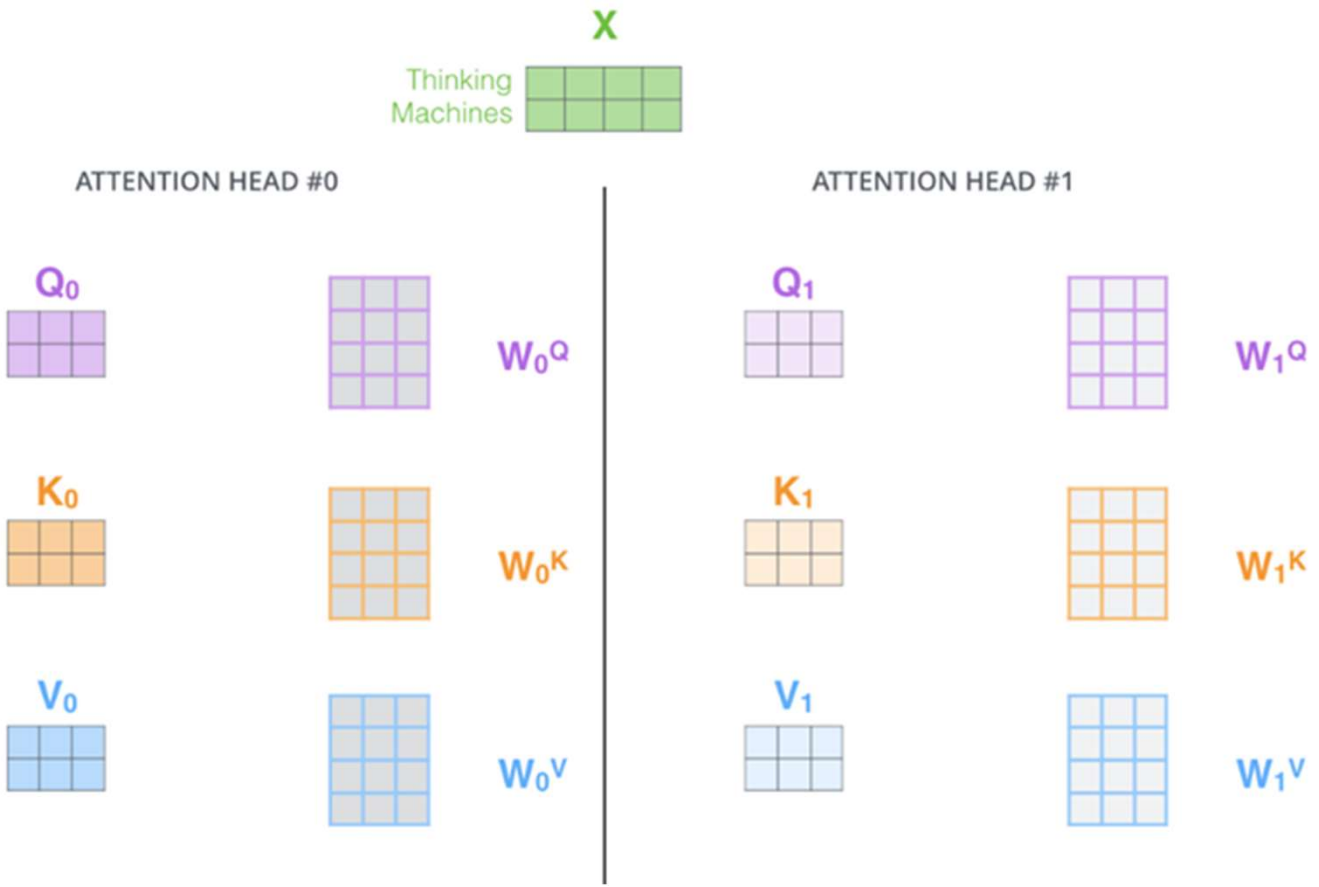
Matrix form



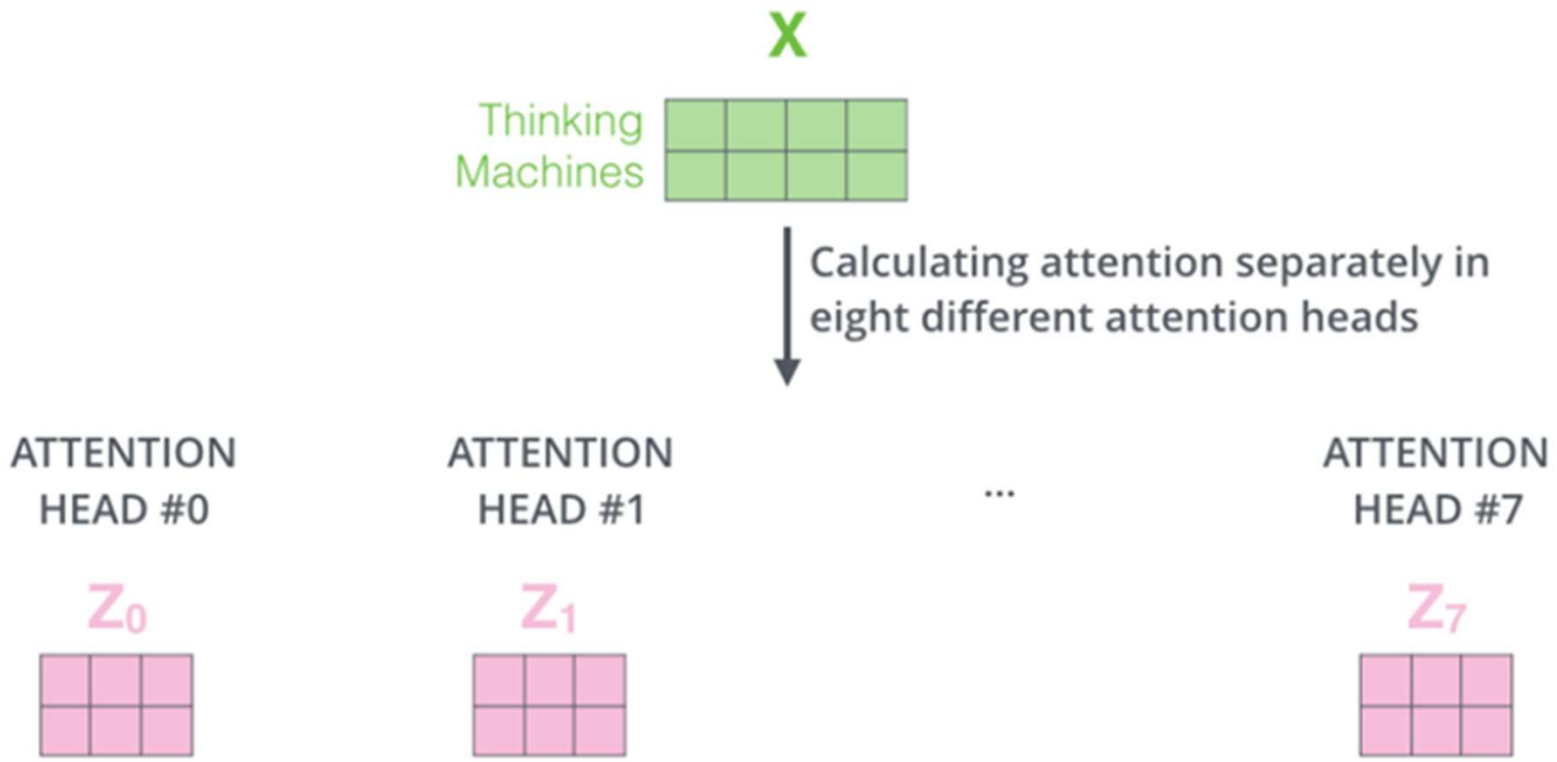
Matrix form



Multi-head


X
Thinking
Machines
ATTENTION HEAD #0
ATTENTION HEAD #1
Q0
K0
V0
W0Q
W0K
W0V
Q1
K1
V1
W1Q
W1K
W1V

Multi-head

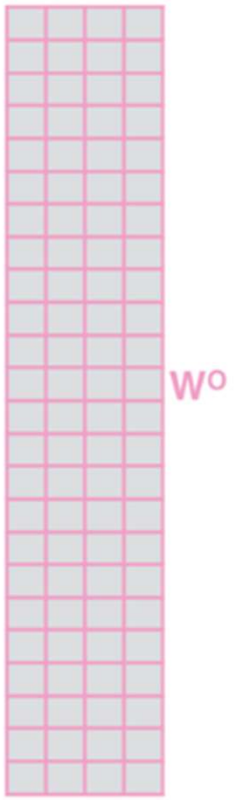

X
Thinking
Machines
Calculating attention separately in
eight different attention heads
ATTENTION
HEAD #0
ATTENTION
HEAD #1
...
ATTENTION
HEAD #7
Z0
Z1
Z7

Multi-head

1) Concatenate all the attention heads

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

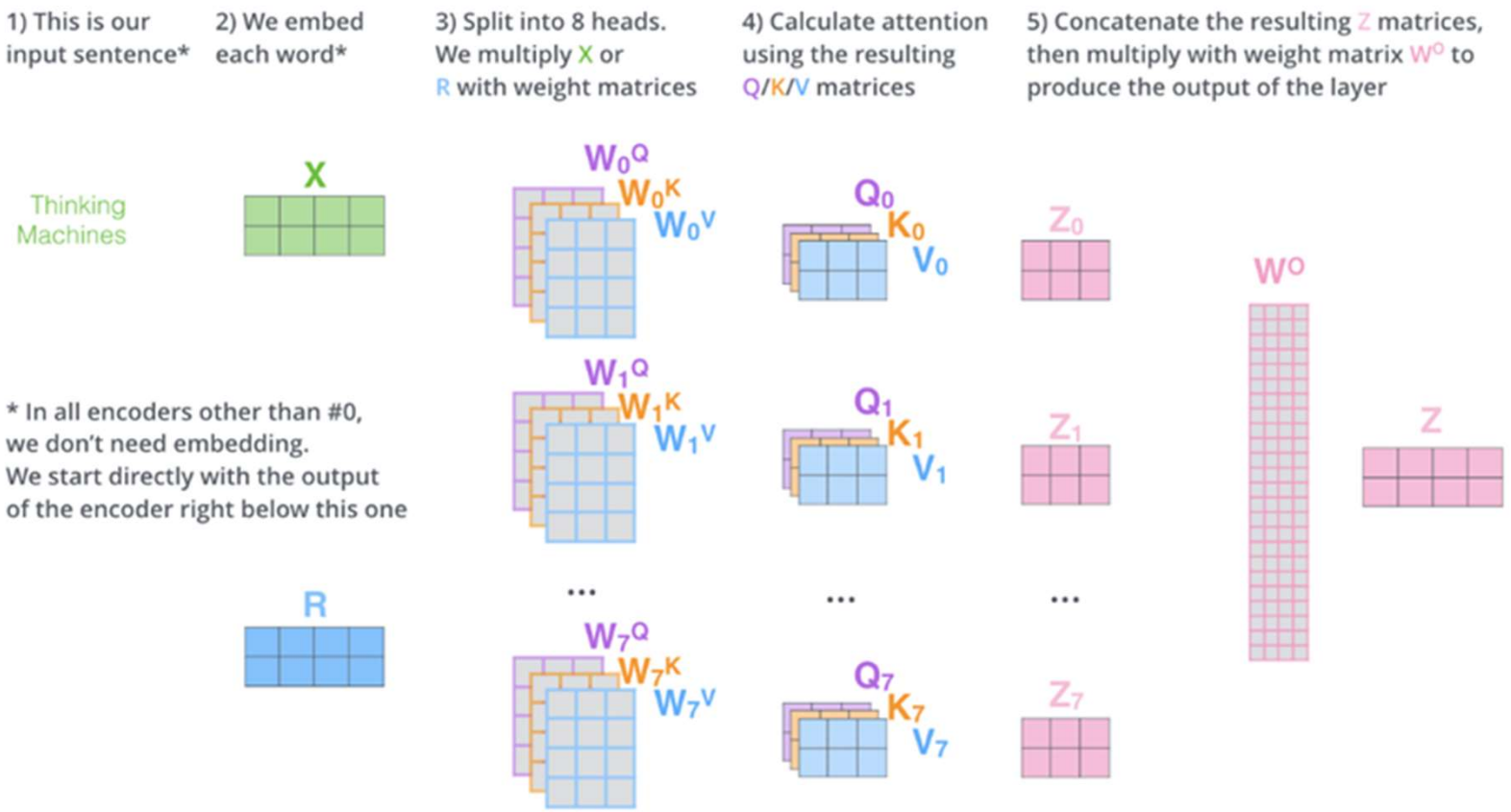
X



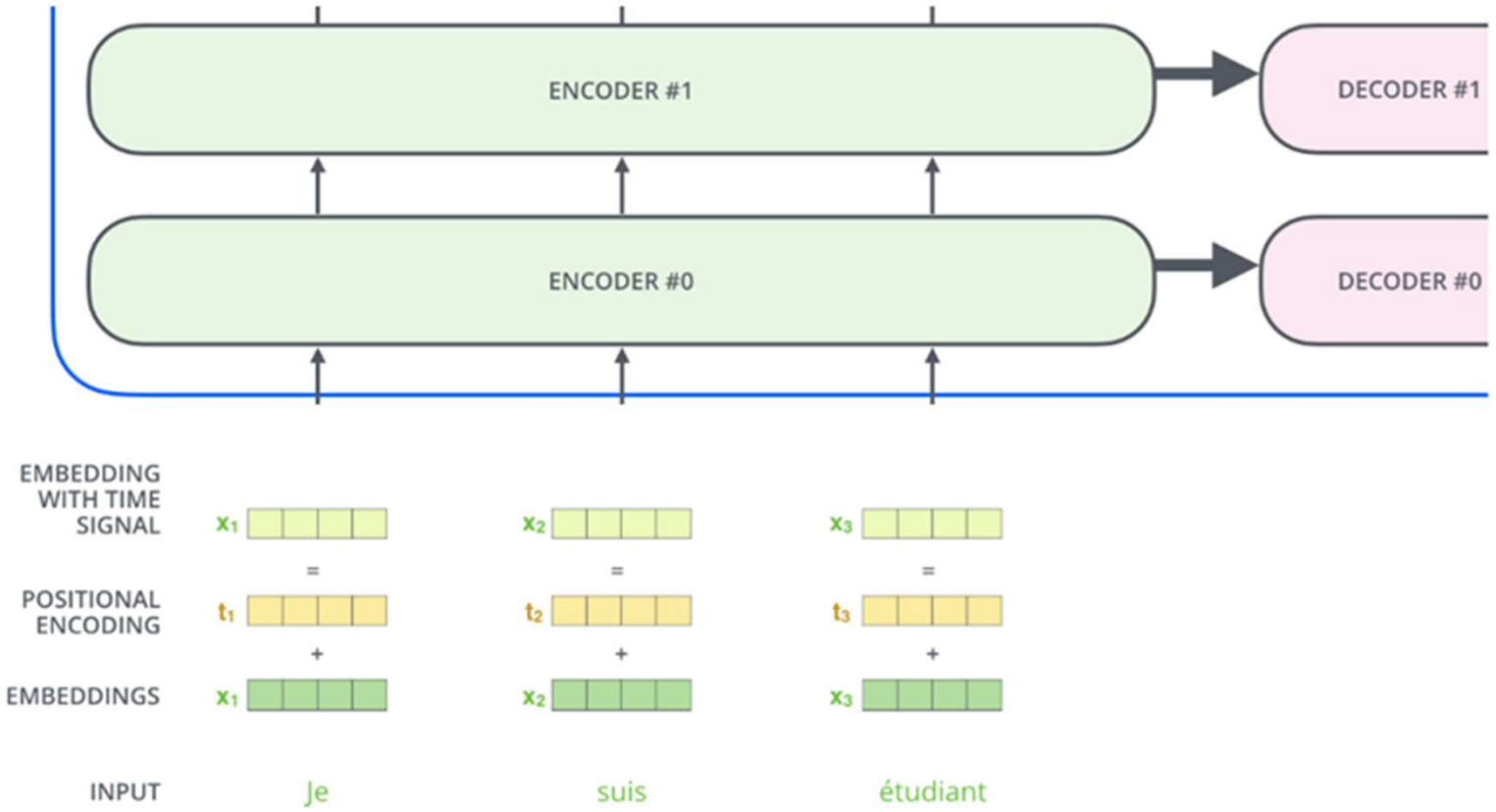
3) The result would be the Z matrix that captures information from all the attention heads. We can send this forward to the FFNN

=

Summary of self-attention



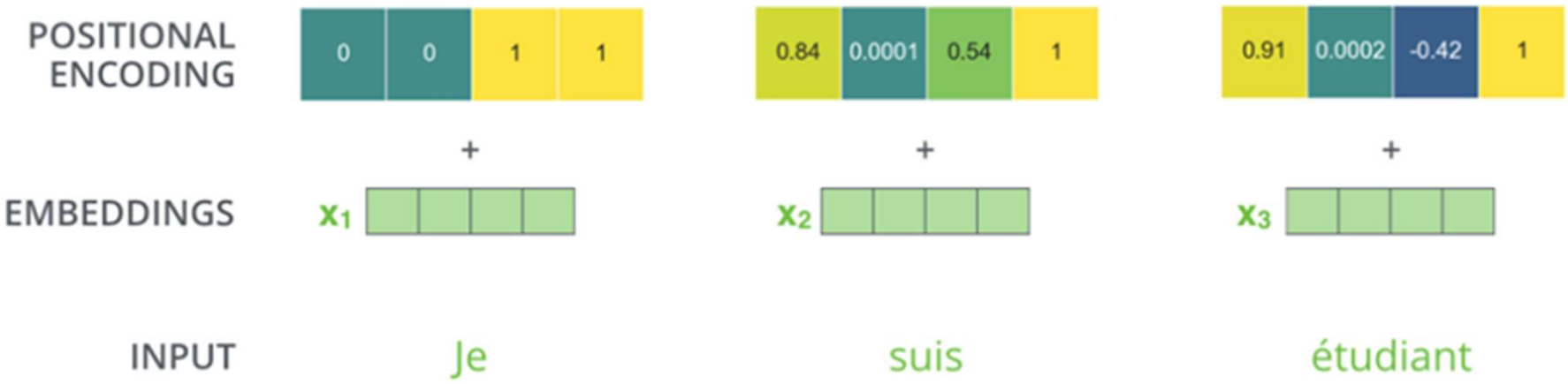
Order of sequence

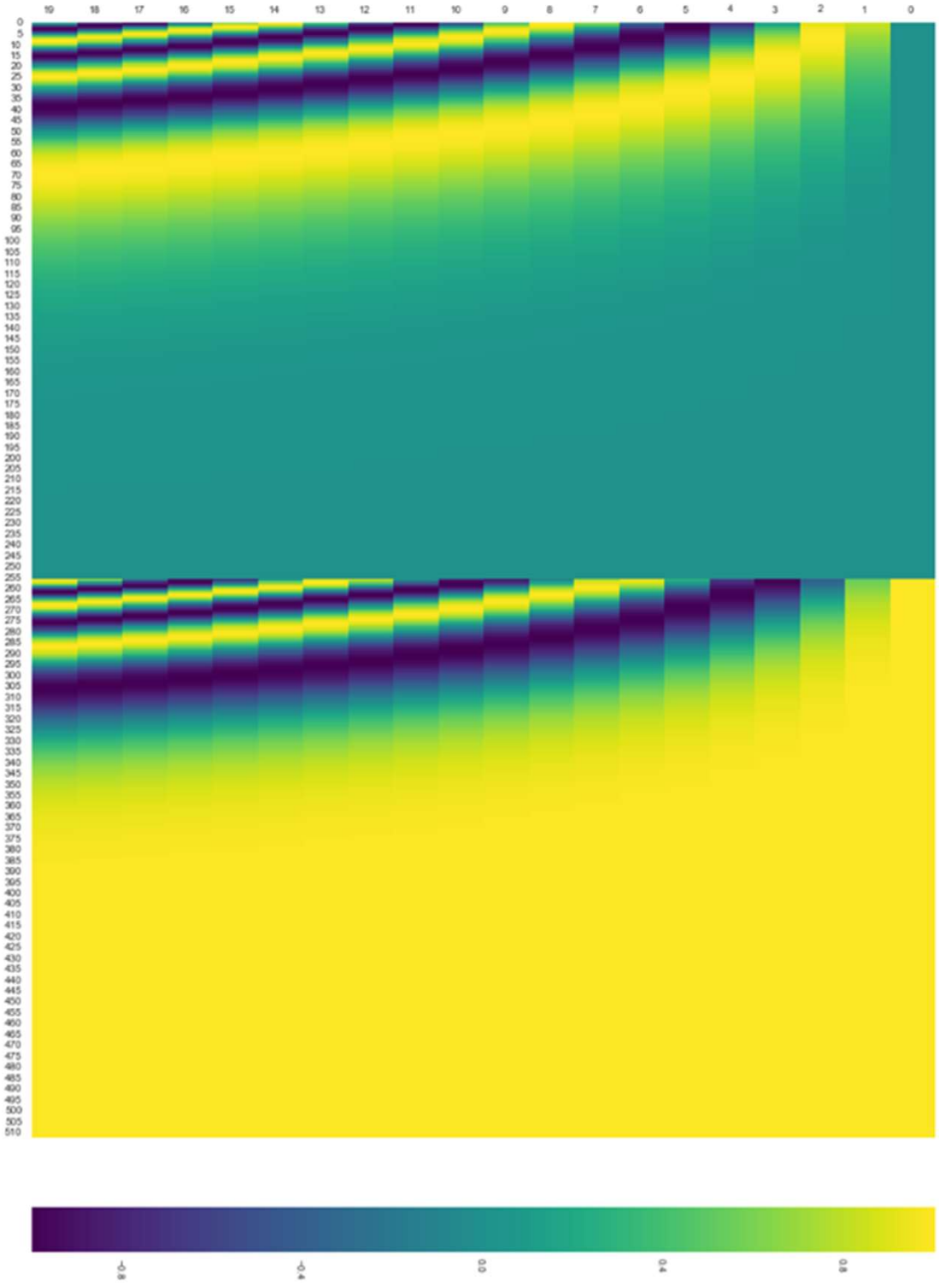


Positional encoding

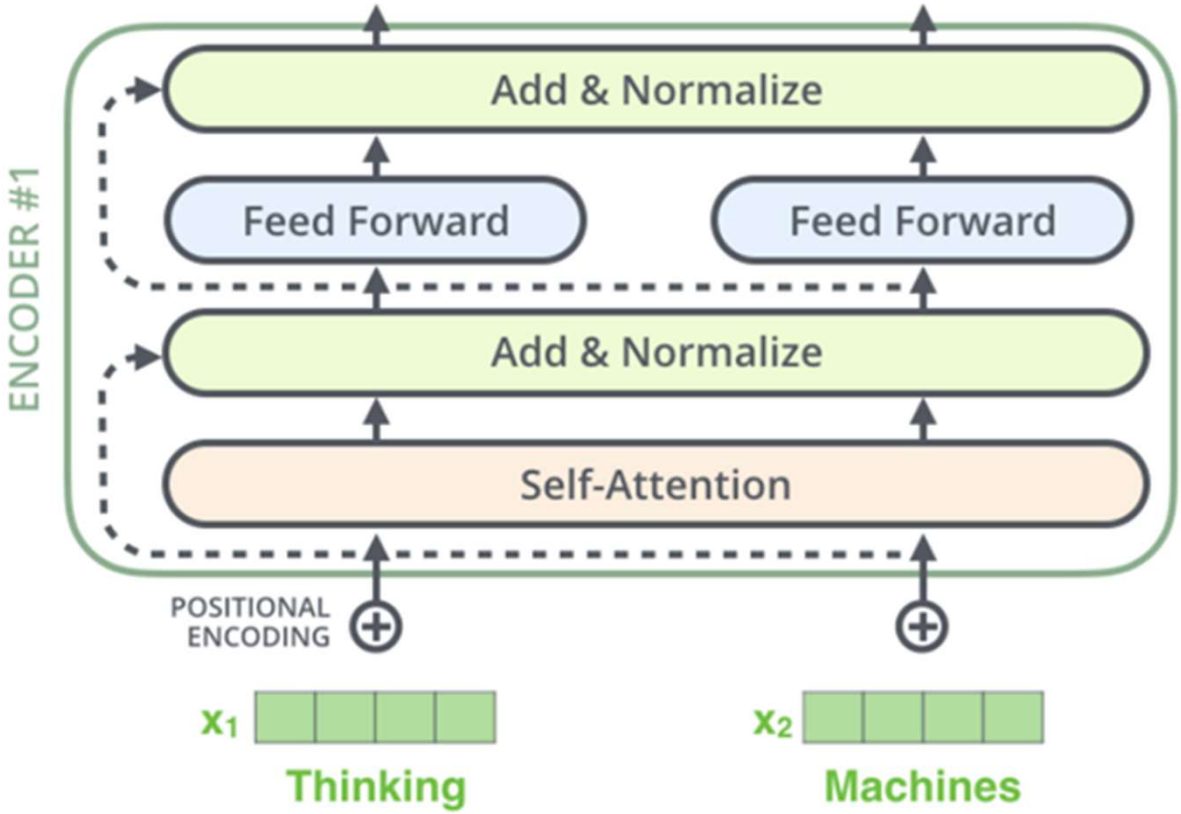
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$

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

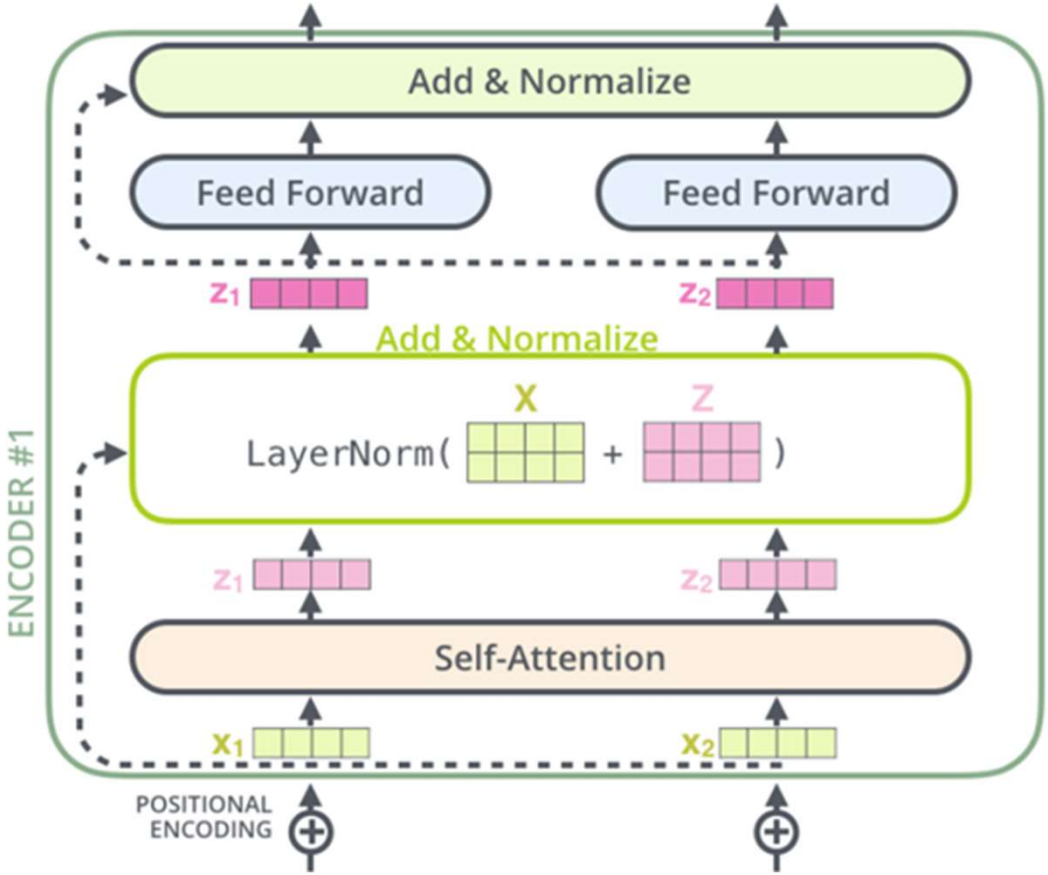



19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100 105 110 115 120 125 130 135 140 145 150 155 160 165 170 175 180 185 190 195 200 205 210 215 220 225 230 235 240 245 250 255 260 265 270 275 280 285 290 295 300 305 310 315 320 325 330 335 340 345 350 355 360 365 370 375 380 385 390 395 400 405 410 415 420 425 430 435 440 445 450 455 460 465 470 475 480 485 490 495 500 505 510

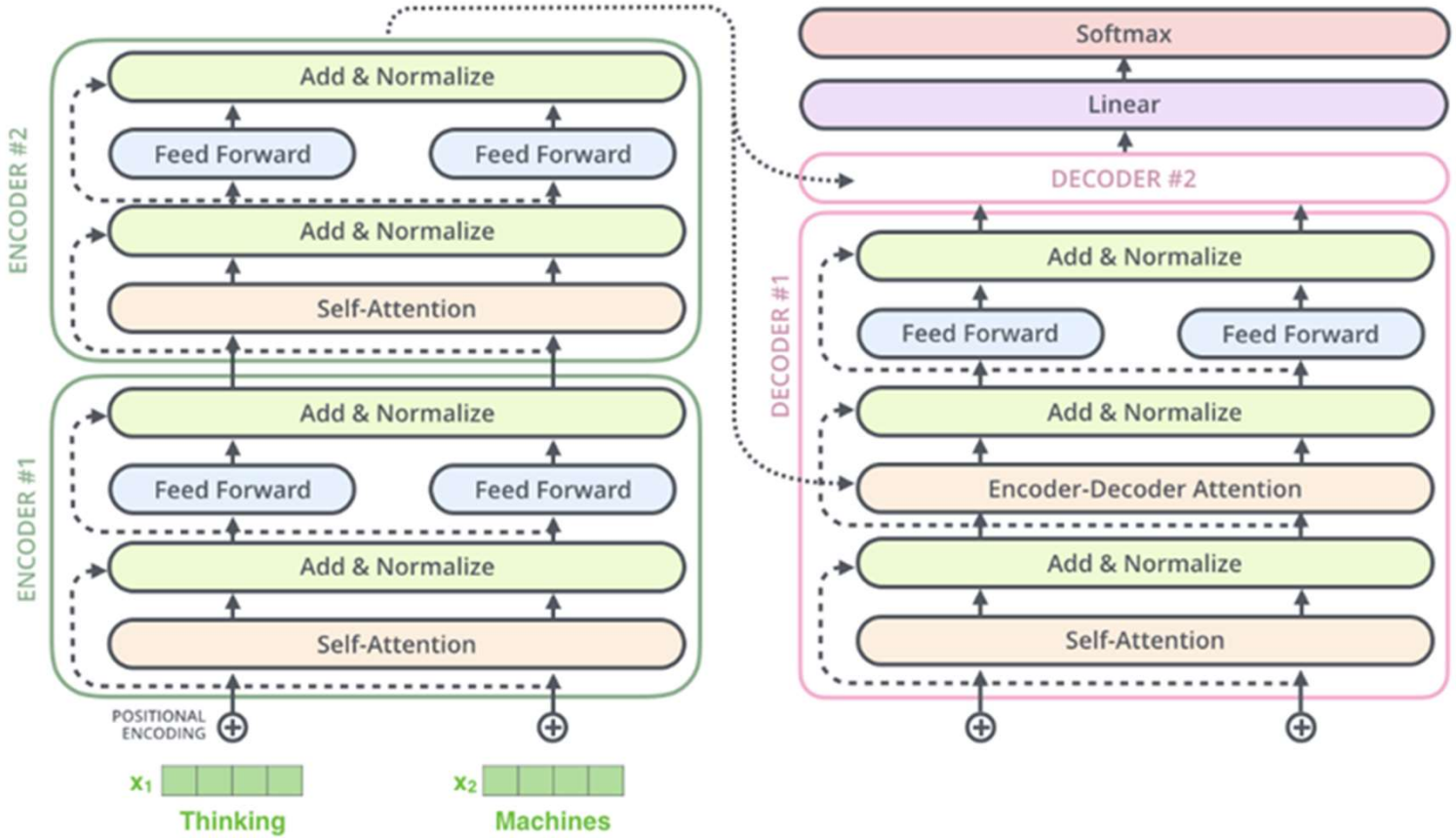
Residue connection

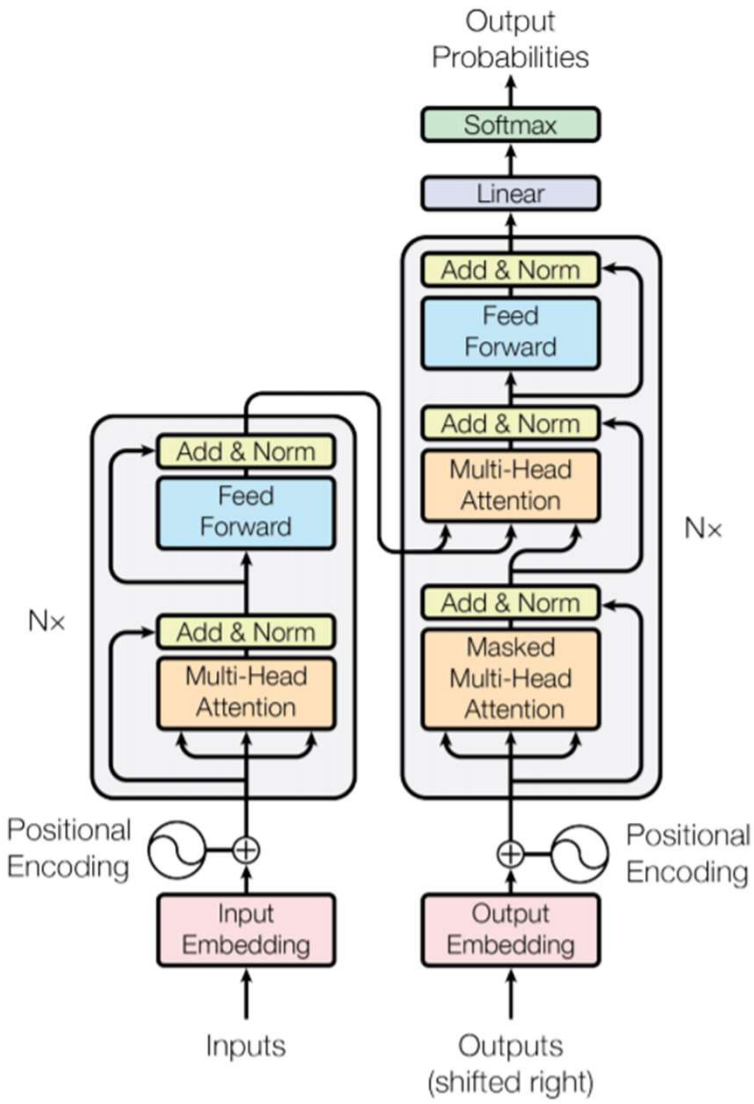


Complete encoder

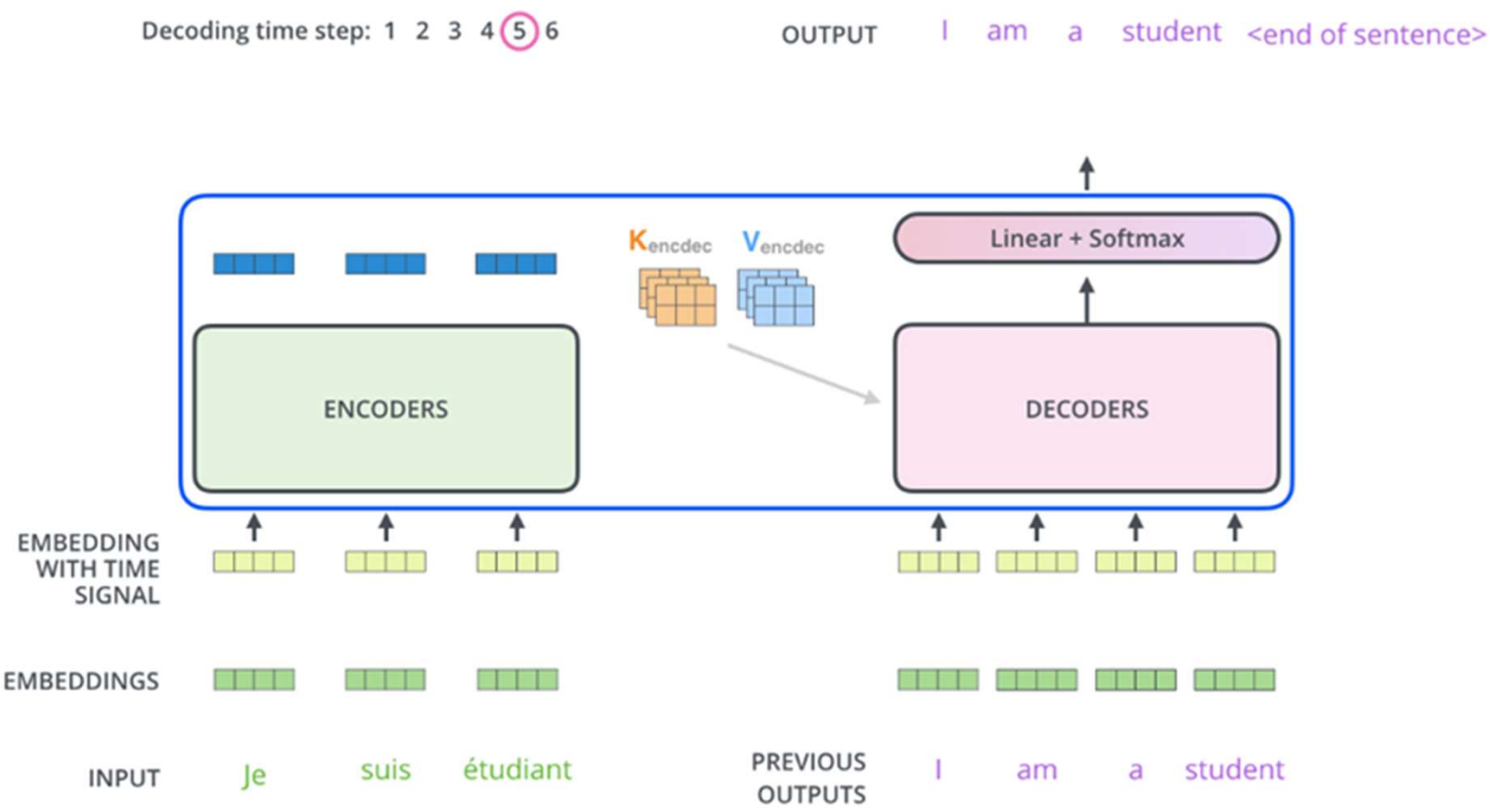


Decoder side




Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
N×
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
N×
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

Decoding


Decoding time step: 1 2 3 4 5 6
OUTPUT
I am a student <end of sentence>
Linear + Softmax
K encdec
V encdec
ENCODERS
DECODERS
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je suis étudiant
PREVIOUS OUTPUTS
I am a student

Implementation

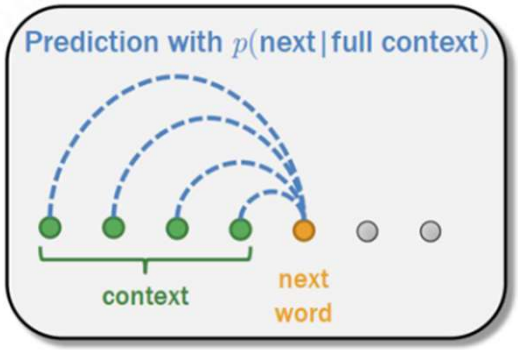
- https://nlp.seas.harvard.edu/2018/04/03/attention.html

BERT

https://arxiv.org/pdf/1810.04805.pdf

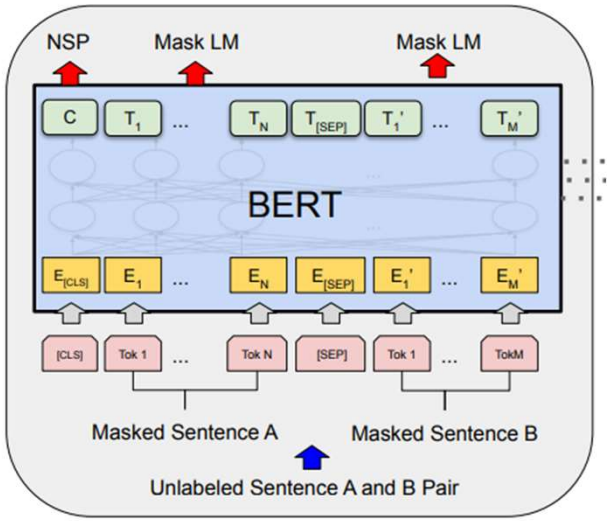
https://arxiv.org/pdf/1907.11692.pdf

LM vs. Masked LM



$$\max_{\theta} \quad \log p_\theta(\mathbf{x}) = \sum_{t=1}^{T} \log p_\theta(x_t \mid \mathbf{x}_{<t}) = \sum_{t=1}^{T} \log \frac{\exp\left(h_\theta(\mathbf{x}_{1:t-1})^\top e(x_t)\right)}{\sum_{x'} \exp\left(h_\theta(\mathbf{x}_{1:t-1})^\top e(x')\right)},$$

Masked LM



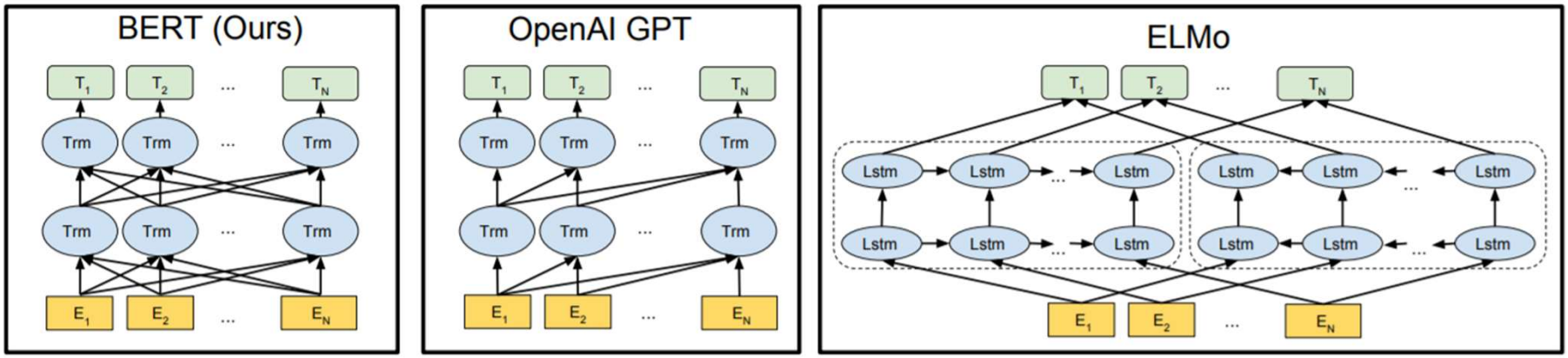
Special Tokens:
[CLS]
[SEP]
[MASK]
[PAD]

NSP = next sentence prediction

$$\max_{\theta} \quad \log p_\theta(\bar{\mathbf{x}} \mid \hat{\mathbf{x}}) \approx \sum_{t=1}^{T} m_t \log p_\theta(x_t \mid \hat{\mathbf{x}}) = \sum_{t=1}^{T} m_t \log \frac{\exp\left(H_\theta(\hat{\mathbf{x}})_t^\top e(x_t)\right)}{\sum_{x'} \exp\left(H_\theta(\hat{\mathbf{x}})_t^\top e(x')\right)},$$

masking: 15% of tokens is replaced with [MASK], mt is 0 if [MASK] and 1 if not

Pre-training model architectures

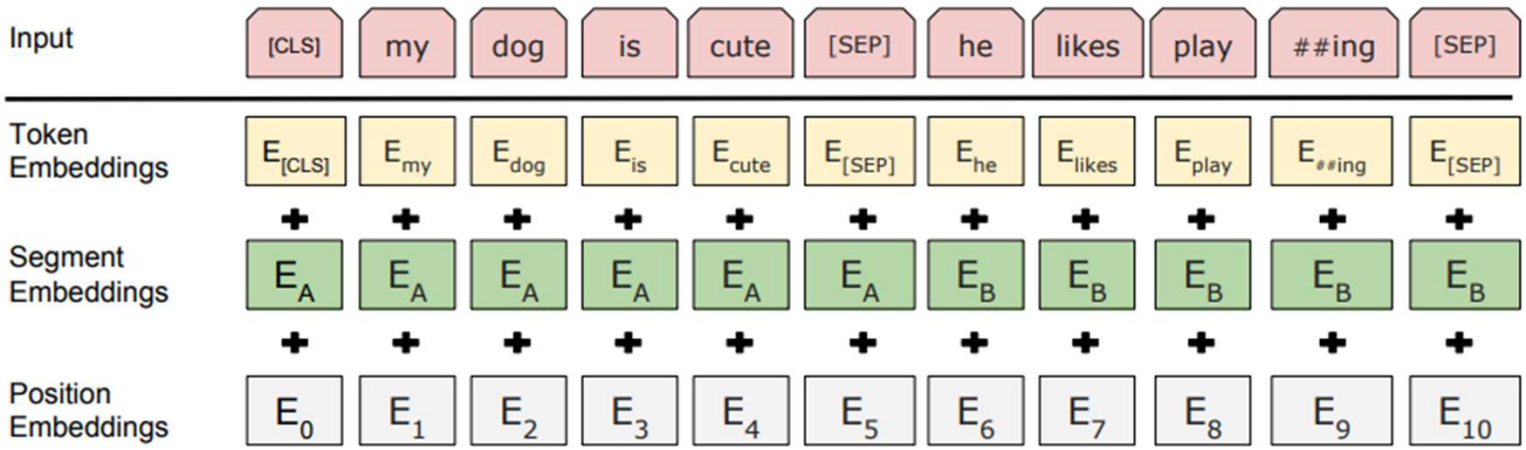


Pretraining Tasks

- LM task: predict the masked token.
- NSP: next sentence prediction – predict if sentence B is actually a sentence that follows A in corpus.

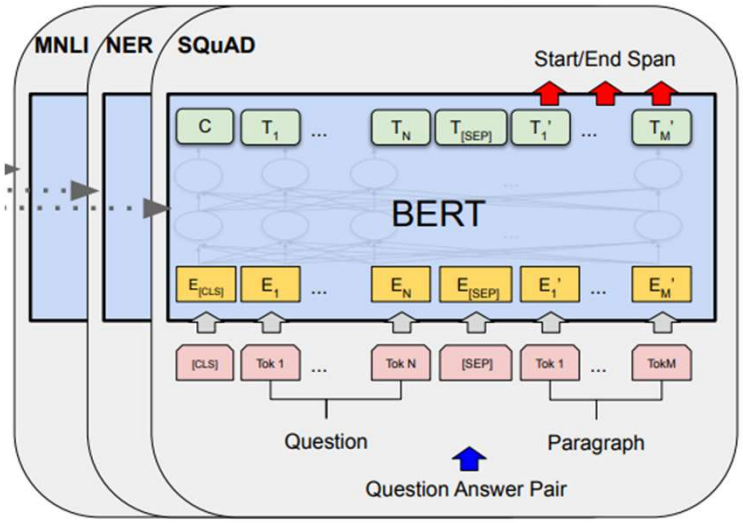
- For LM task: token are masked with 15% probability.
- NSP: 50-50 split between actual A followed by B, and replace B with a random sentence from corpus.

Encoding the input



Fine-tuning

- Change the head according to task
- E.g.
  - text recognition – softmax head
  - squad (question answering) two softmax heads, one for start token position and for end token position



Roberta

- A variation on BERT
- remove NSP loss
- dynamic masking (masks are randomized for each batch)
- large batch size
- byte-level BPE tokenizer (BERT uses Wordpiece)
- more data and train for longer

Roberta vs. Bert

Model	data	bsz	steps	SQuAD (v1.1/2.0)	MNLI-m	SST-2
RoBERTa						
with BOOKS + WIKI	16GB	8K	100K	93.6/87.3	89.0	95.3
+ additional data (§3.2)	160GB	8K	100K	94.0/87.7	89.3	95.6
+ pretrain longer	160GB	8K	300K	94.4/88.7	90.0	96.1
+ pretrain even longer	160GB	8K	500K	**94.6/89.4**	**90.2**	**96.4**
$\text{BERT}_{\text{LARGE}}$						
with BOOKS + WIKI	13GB	256	1M	90.9/81.8	86.6	93.7
$\text{XLNet}_{\text{LARGE}}$						
with BOOKS + WIKI	13GB	256	1M	94.0/87.8	88.4	94.4
+ additional data	126GB	2K	500K	94.5/88.8	89.8	95.6

Example - Huggingface

- https://github.com/huggingface/transformers/tree/v4.23-release/examples/pytorch