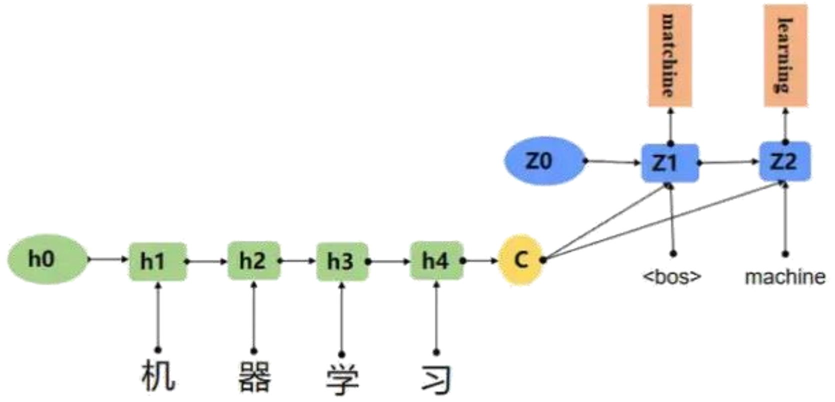
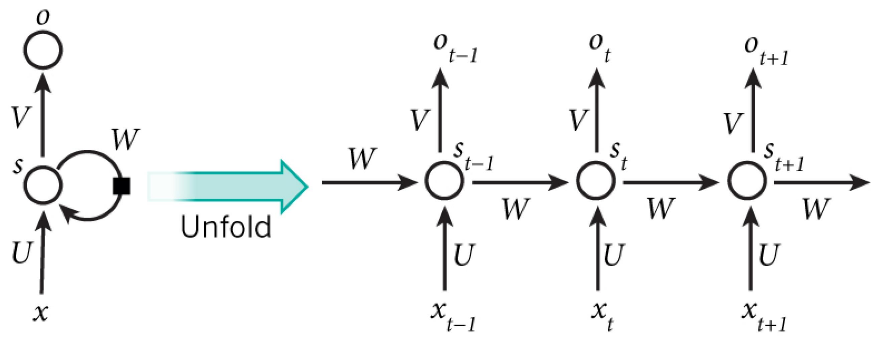
Attention Is All You Need 论文导读 & Transformer详解

Why Transformer ?

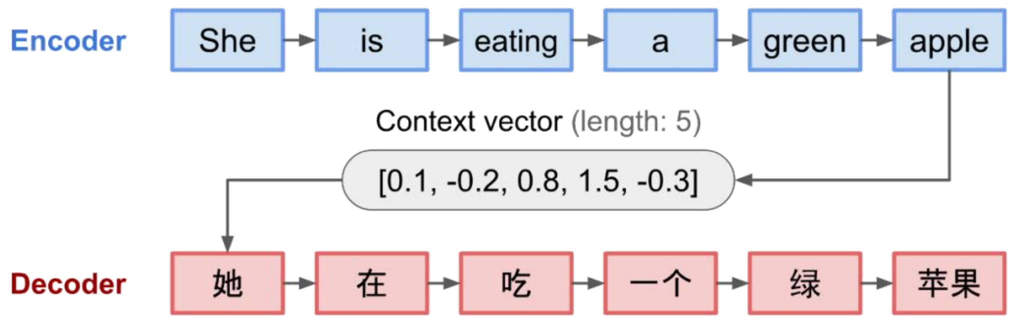
Seq2Seq 任务示例——NMT (Neural Machine Translation)

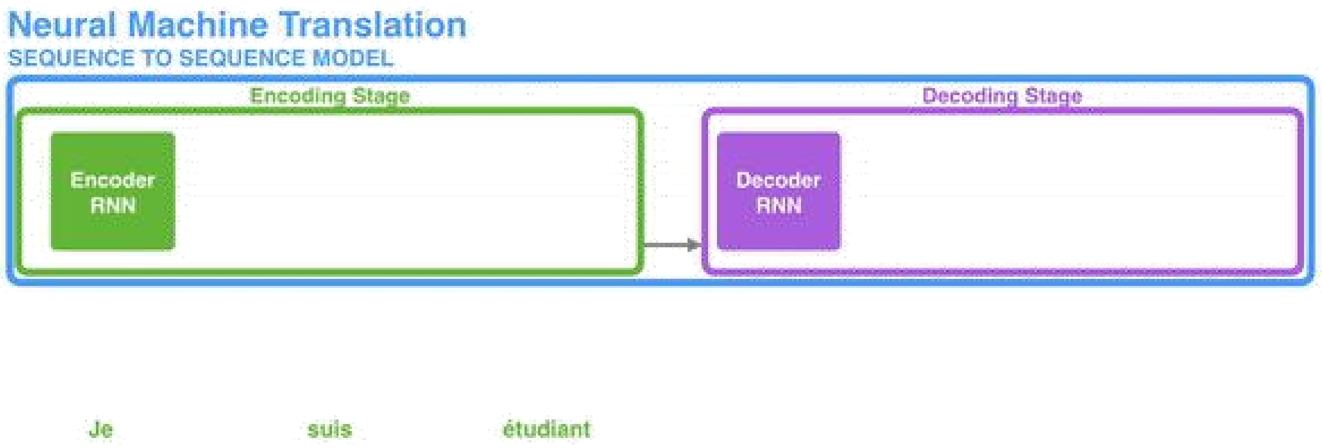


Seq2Seq模型示例——RNN (Recurrent Neural Network)



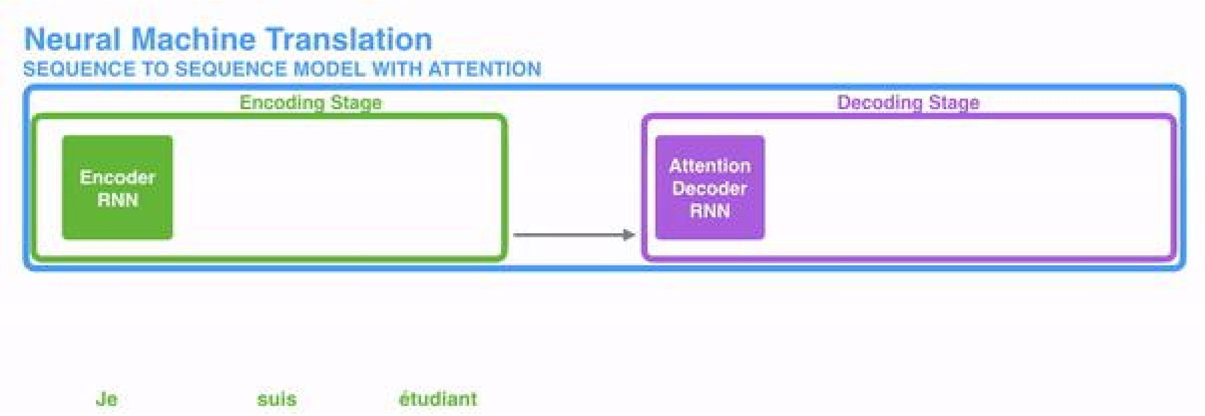
Encoder-Decoder 框架（编码器+解码器）

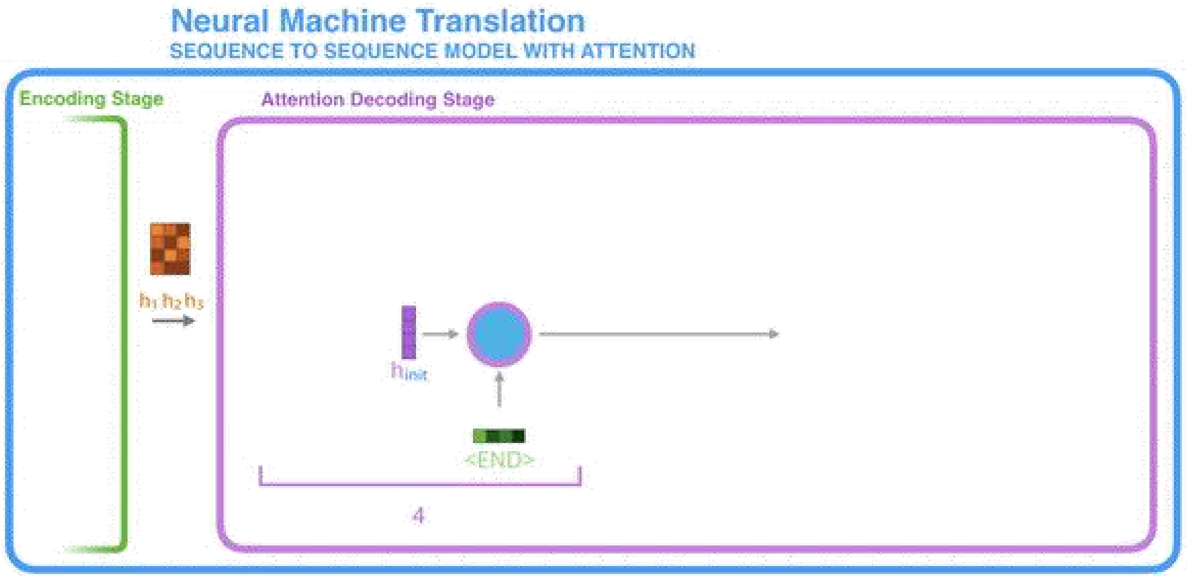




Why Transformer ?

Seq2Seq model with Attention





RNN及其变体模型无法并行计算，模型效率低下！

论文信息

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

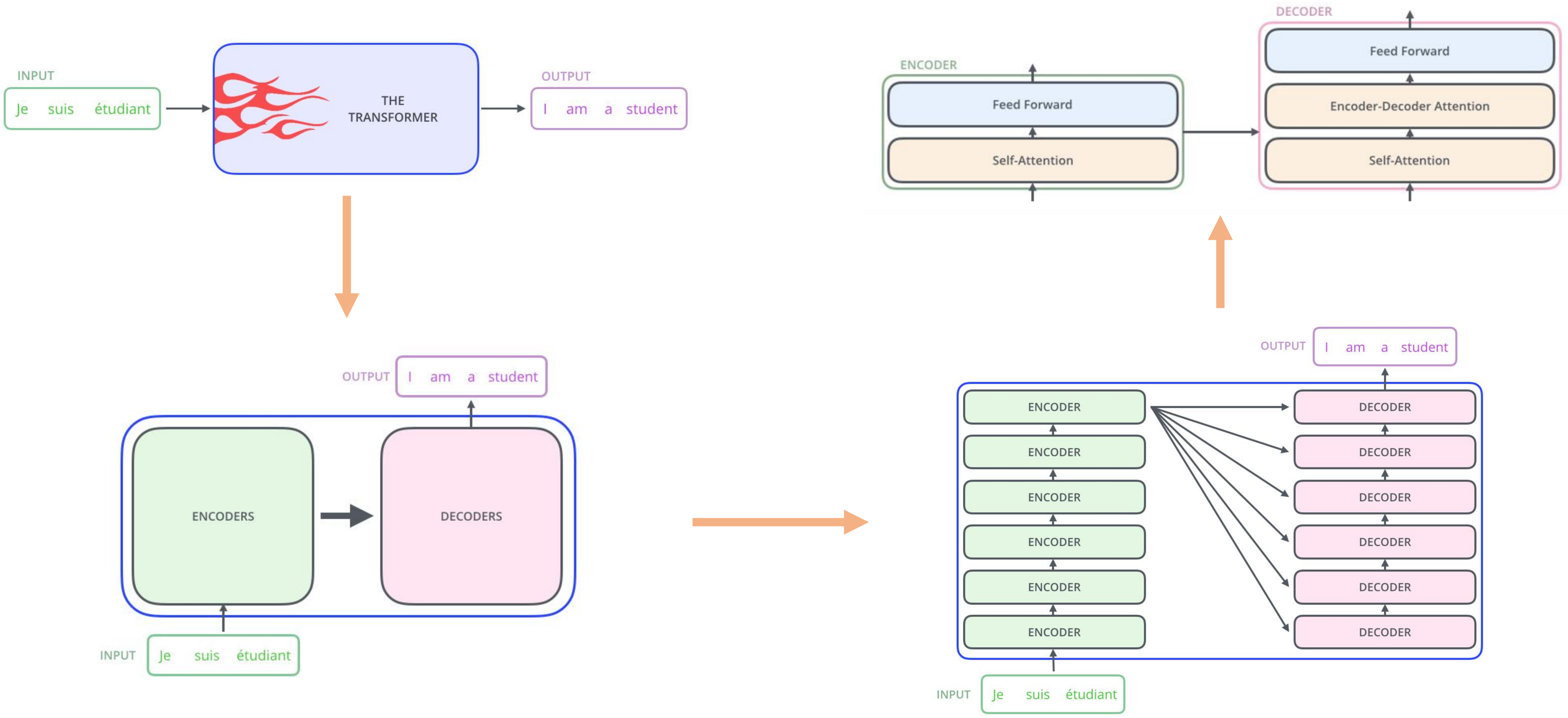
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. 2017. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). Curran Associates Inc., Red Hook, NY, USA, 6000–6010.

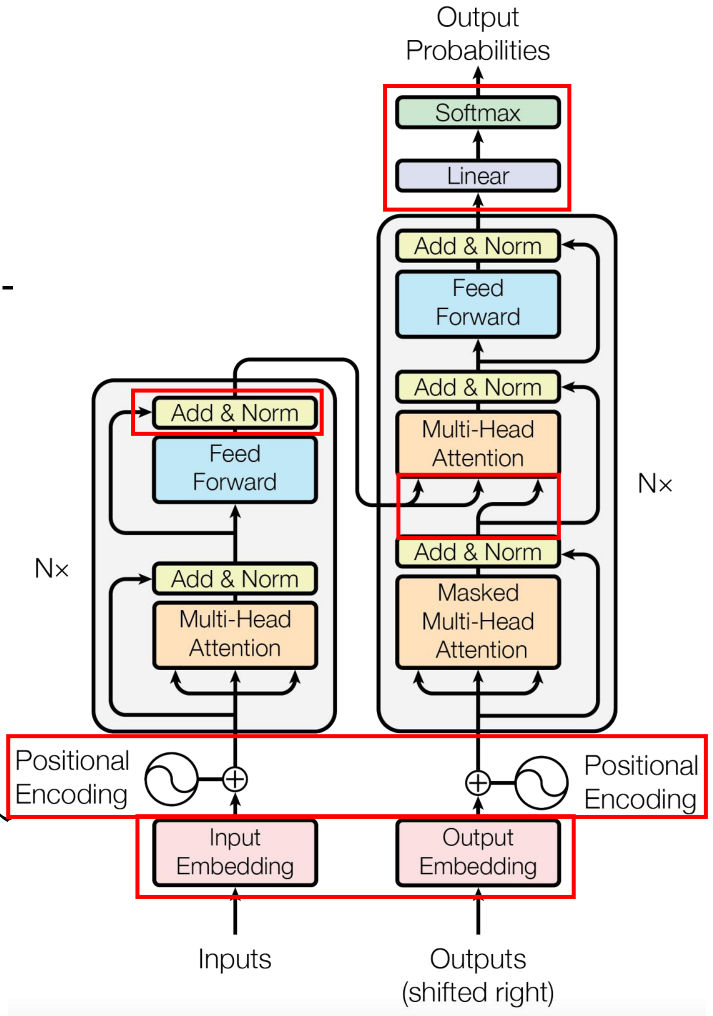
Transformer 架构



Transformer 架构

编码器

- 由N个block堆叠而成；
- 每个block有两层：
  - Multi-Head Attention (Self-Attention)
    + Add (Residual Connection)
    + Norm (LayerNorm)；
  - Feed Forward
    + Add (Residual Connection)
    + Norm (LayerNorm)；
- $\text{Block}_1 \sim \text{Block}_{N-1}$的输出：输入到下个Block；
- Block_N的输出：输入到解码器的各层中。

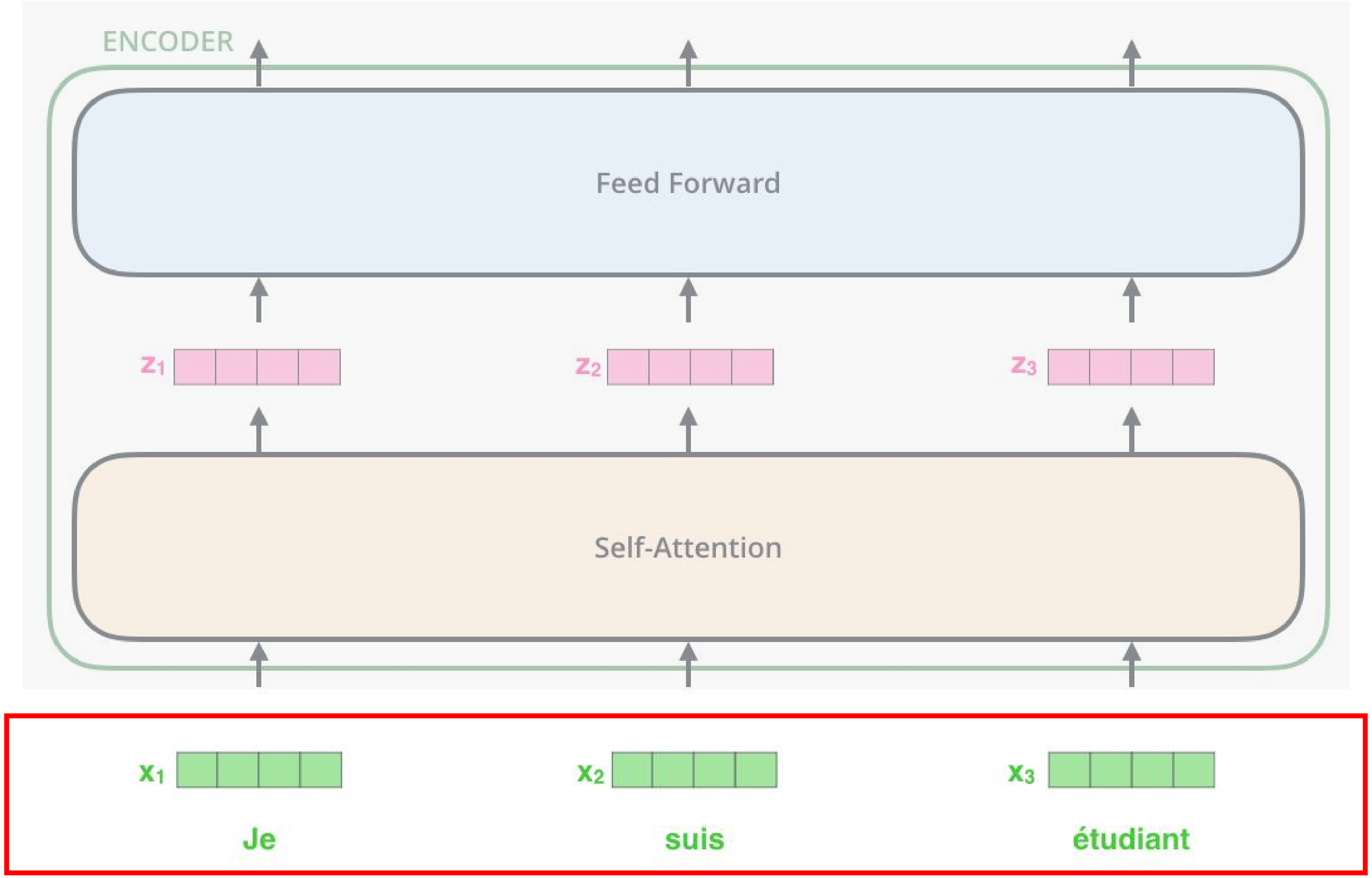


解码器

- 由N个block堆叠而成；
- 每个block有三层：
  - Masked Multi-Head Attention (Self-Attention)
    + Add (Residual Connection)
    + Norm (LayerNorm)；
  - Multi-Head Attention (Co-Attention)
    + Add (Residual Connection)
    + Norm (LayerNorm)；
  - Feed Forward
    + Add (Residual Connection)
    + Norm (LayerNorm)；
- $\text{Block}_1 \sim \text{Block}_{N-1}$的输出：输入到下个Block；
- Block_N的输出：输入到后续的Linear层中。

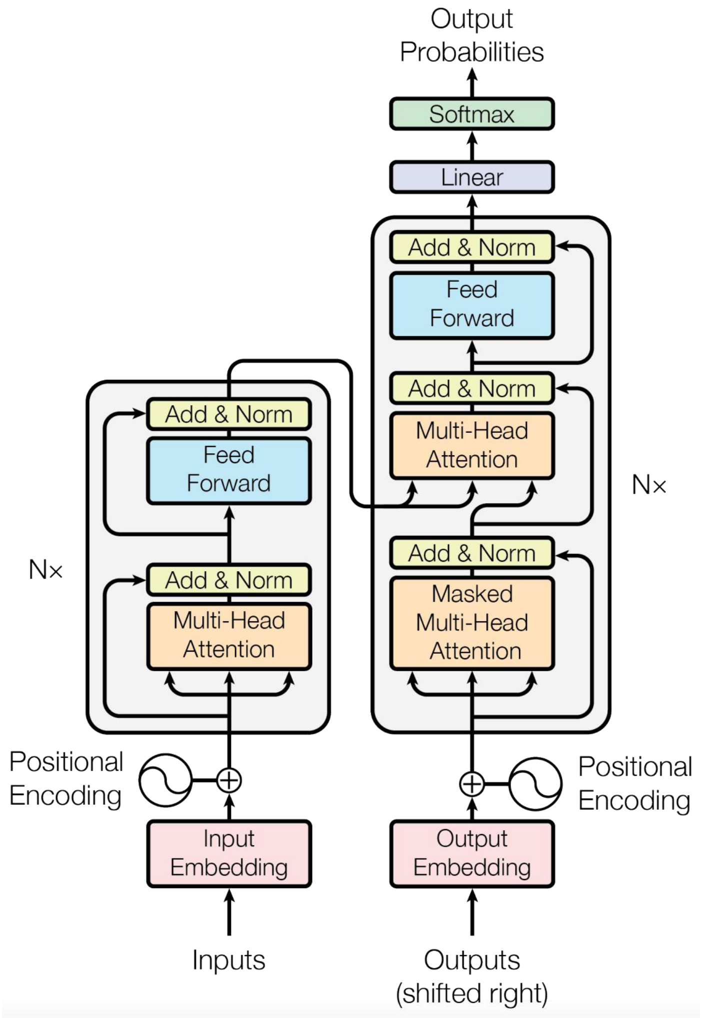
Transformer 工作流程

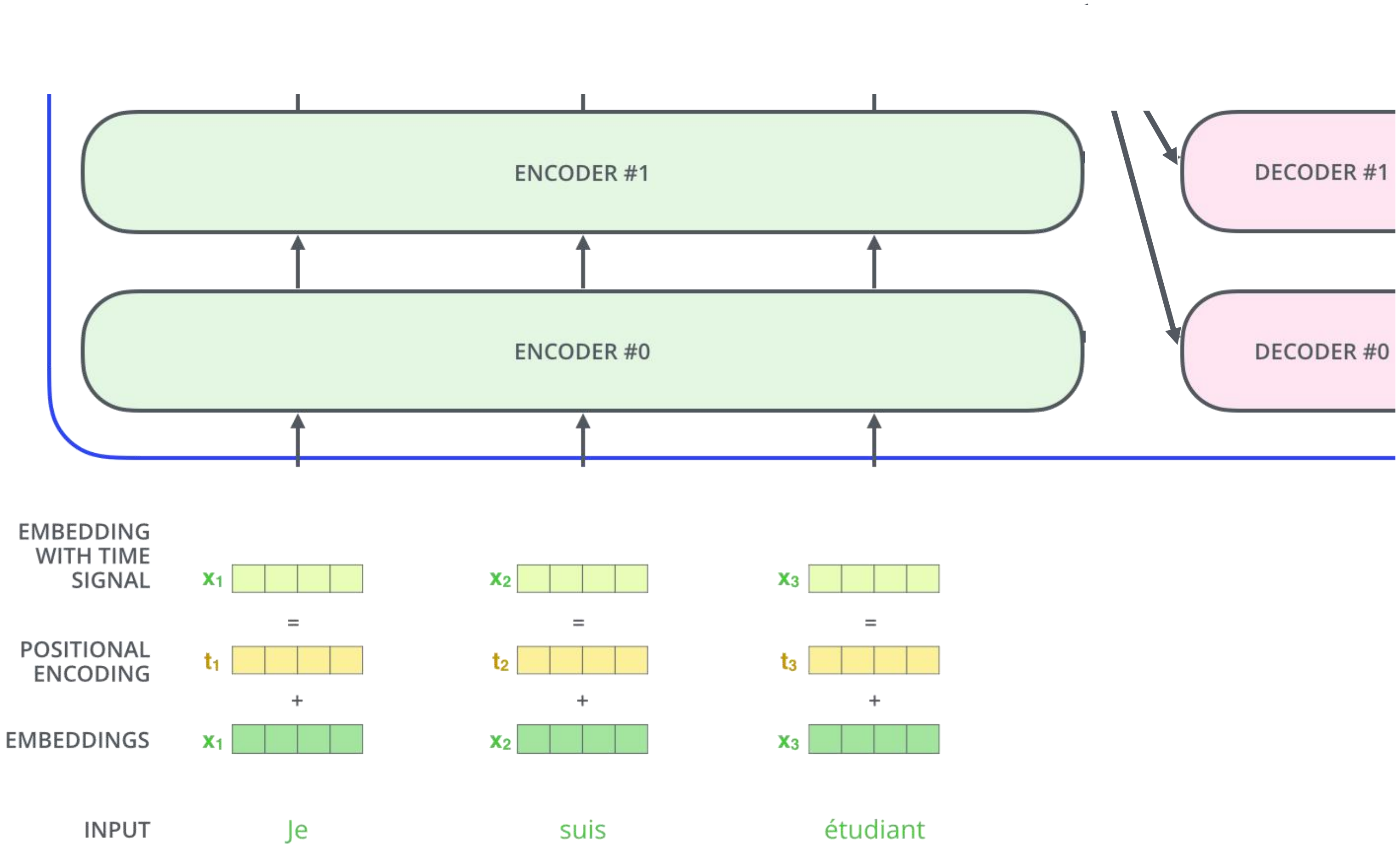
Word Embedding



Transformer 工作流程

位置编码（Positional Encoding）





Transformer 工作流程

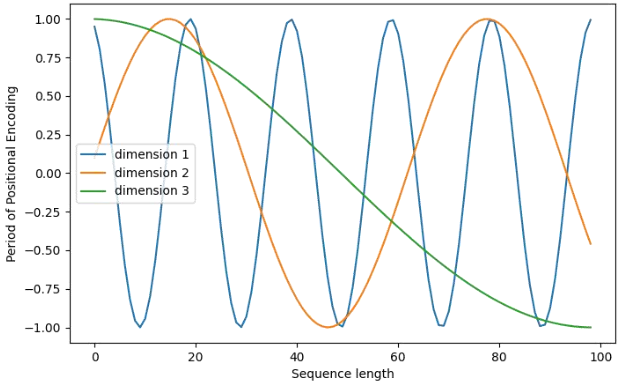
位置编码（Positional Encoding）

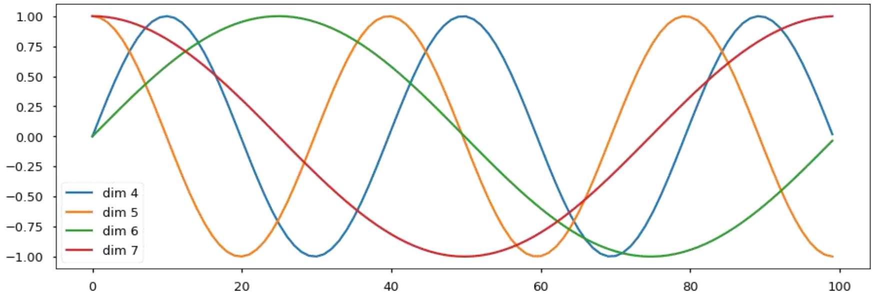
$$PE(pos, 2i) = \sin(pos/10000^{2i/d_{model}})$$
$$PE(pos, 2i+1) = \cos(pos/10000^{2i/d_{model}})$$

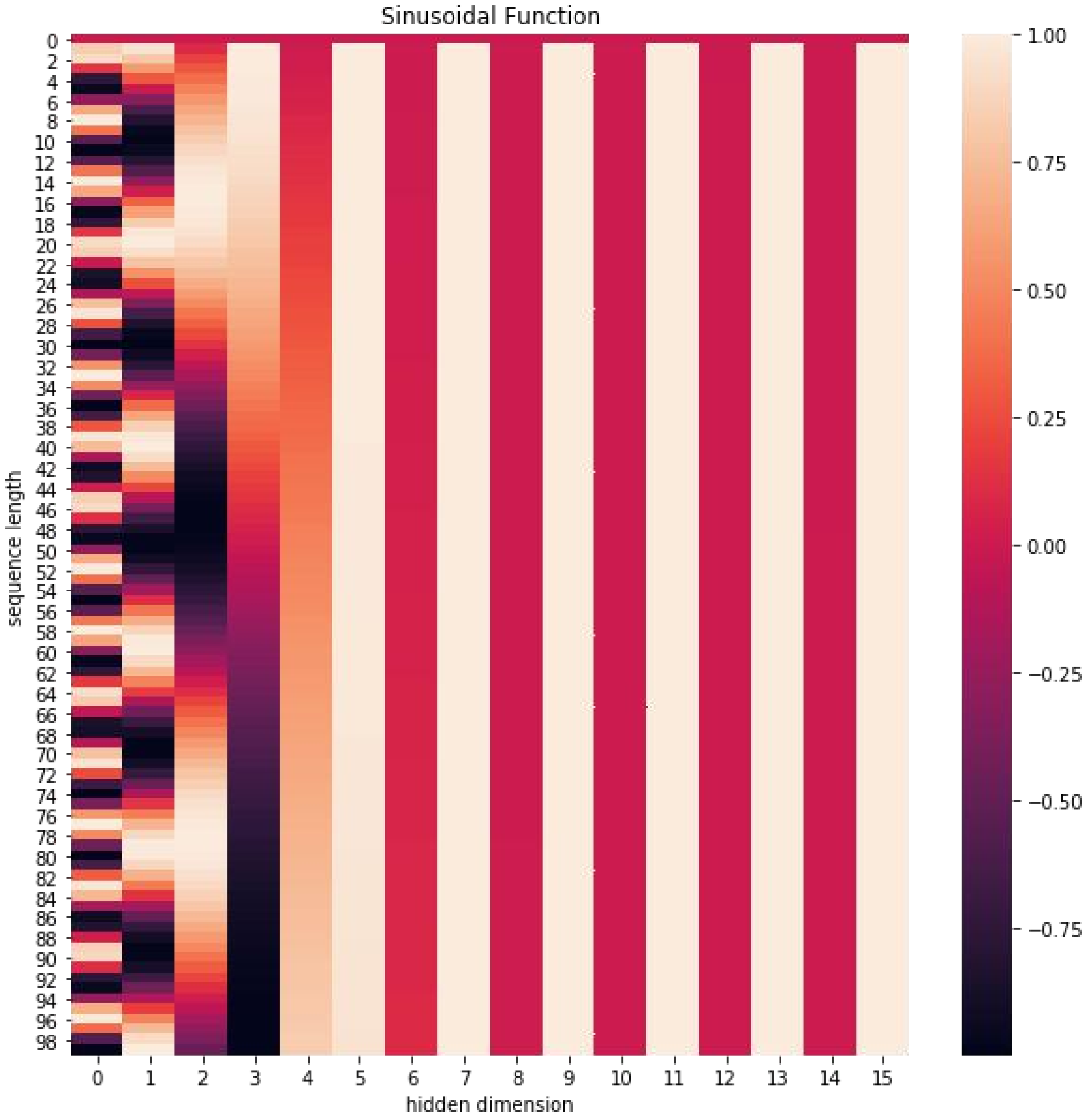
pos $\in [0, \text{max\_sequence\_length})$

$T = \frac{2\pi}{\omega} = 2\pi * 10000^{\frac{2i}{d_{model}}}$

I $\in [0, \frac{d_{model}}{2})$

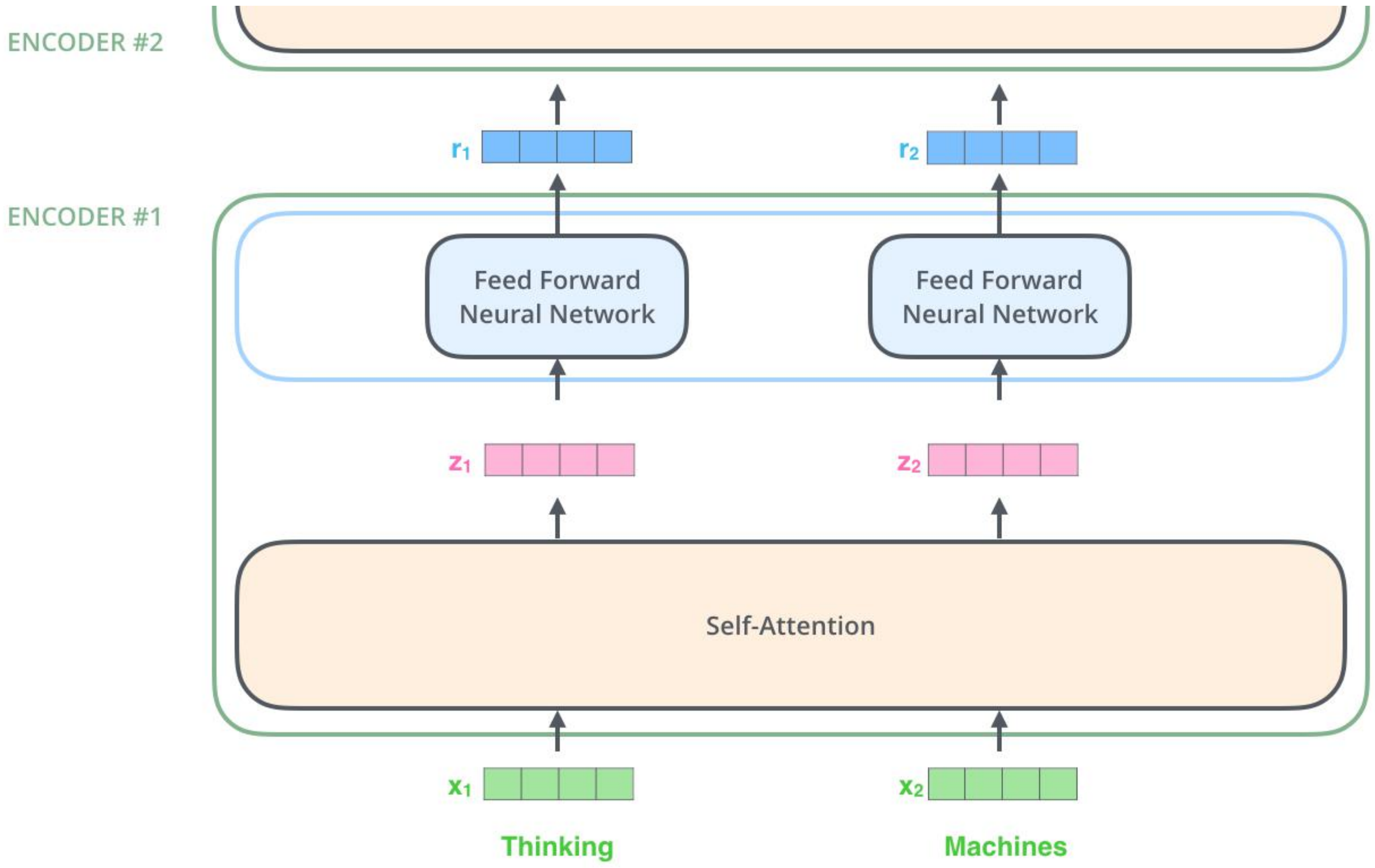






Transformer 工作流程

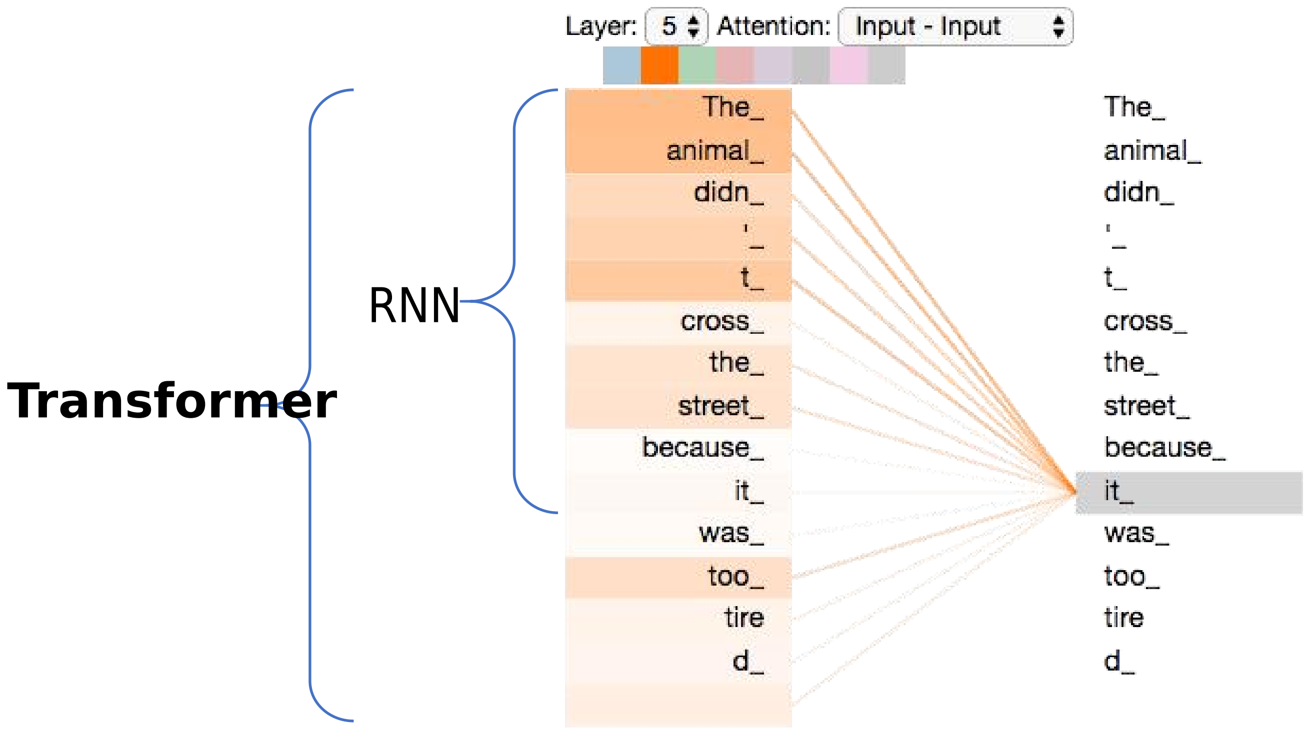
编码过程



Transformer 工作流程

编码过程 —— Self-Attention（宏观）

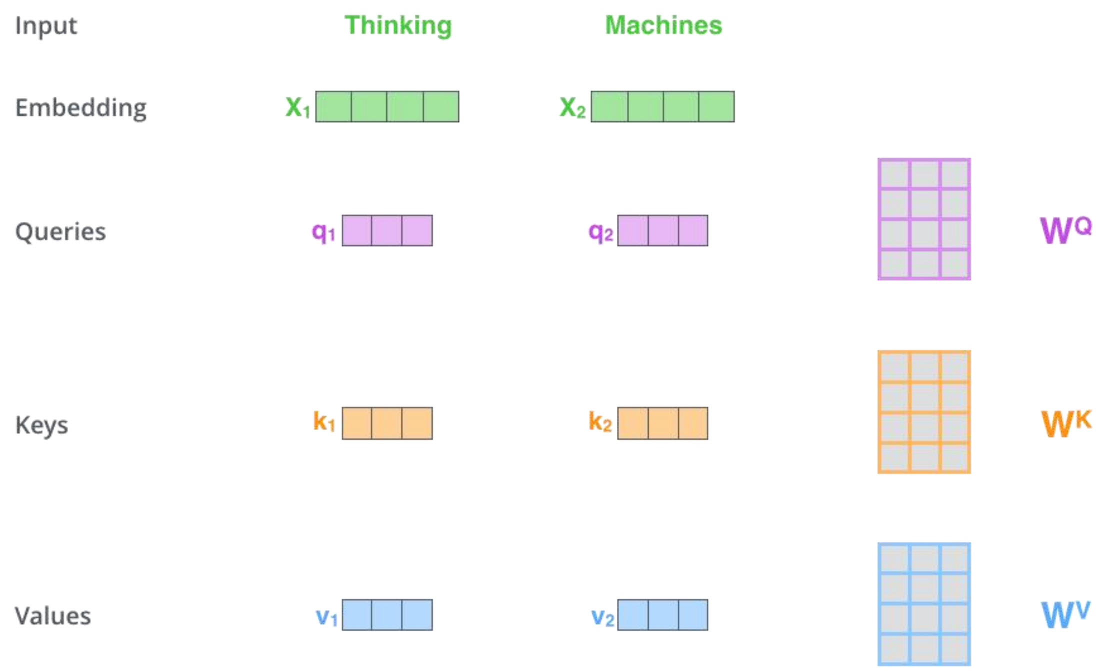
The animal didn’t cross the street because it was too tired.



Transformer 工作流程

编码过程 —— Self-Attention（微观）

第一步：生成 Q、K、V，辅助计算注意力机制



X_1与W^Q权重矩阵相乘得到q_1 ,就是与这个单词相关的查询向量。通过这种方式，为输入序列的每个单词都创建一个查询向量Q、一个键向量K和一个值向量V。

Transformer 工作流程

编码过程 —— Self-Attention（微观）

核心公式：$Attention(Q,K,V)$

$$= softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

第二步：计算当前单词的Q与候选单词的K的点积；

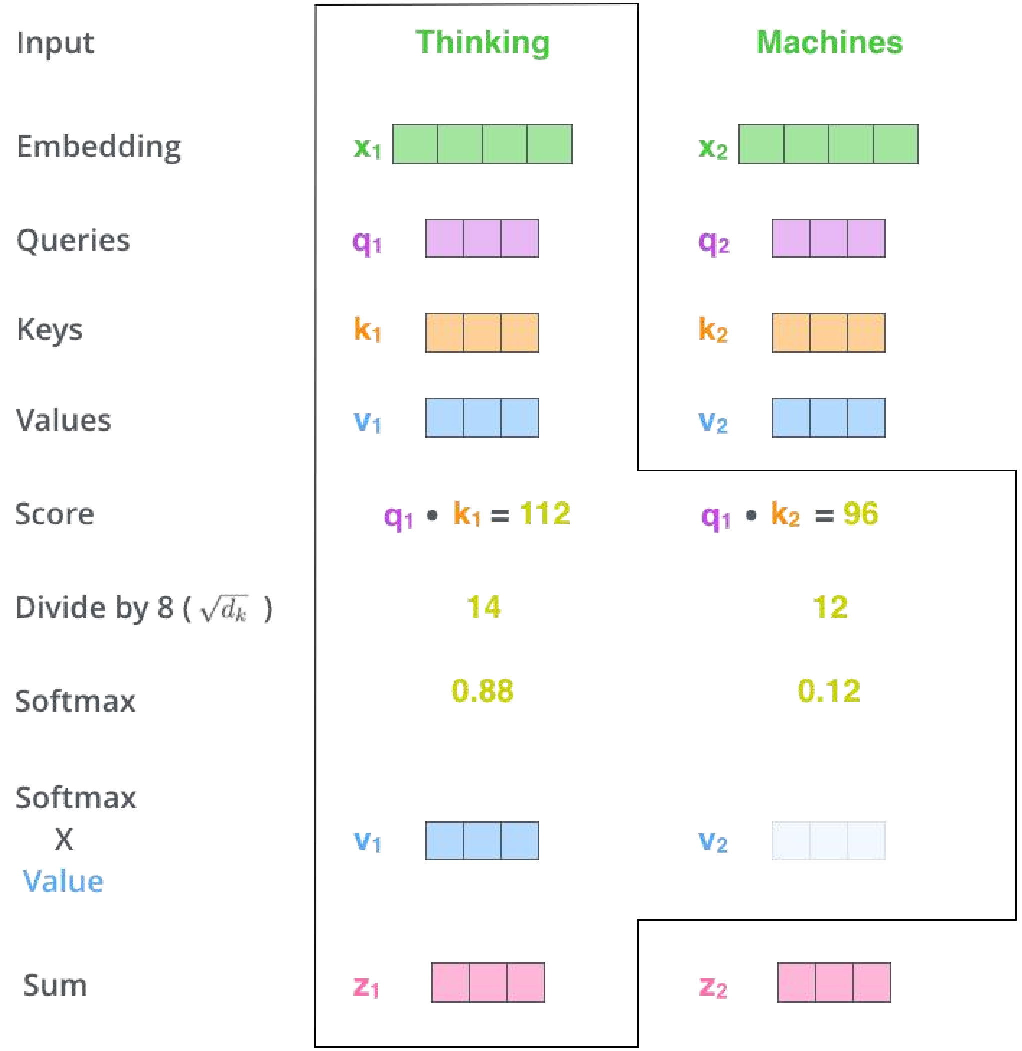
第三步：将上一步结果除以维度的平方根；

第四步：将上一步结果通过softmax函数转换；

第五步：将候选单词的每个值向量乘以softmax分数；

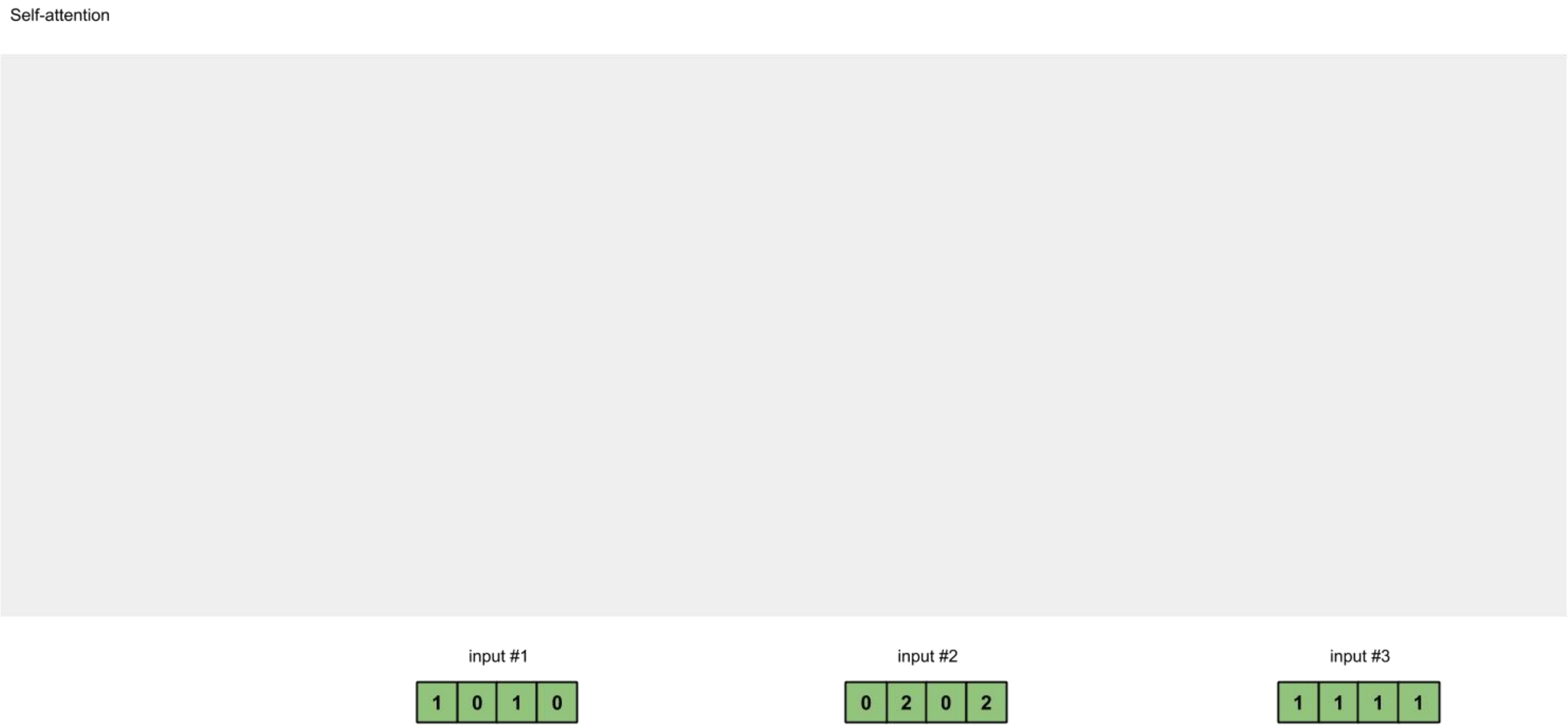
第六步：对加权后的值向量求和，

即得到自注意力层在该位置的输出；



Transformer 工作流程

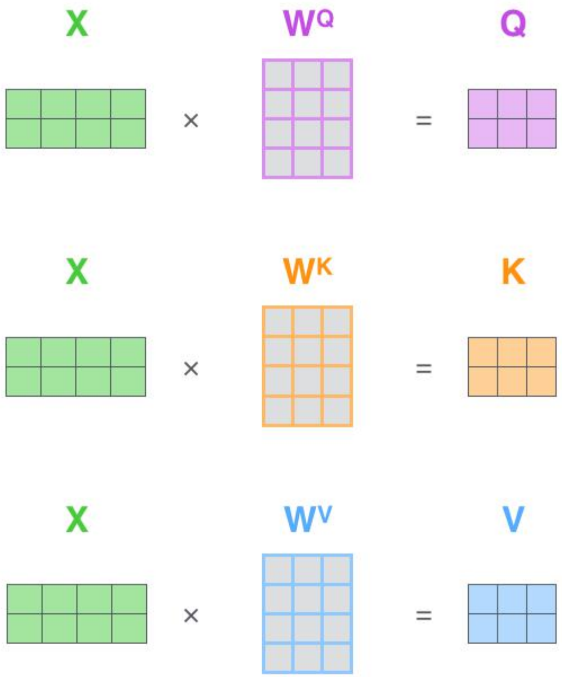
编码过程 —— Self-Attention（微观）

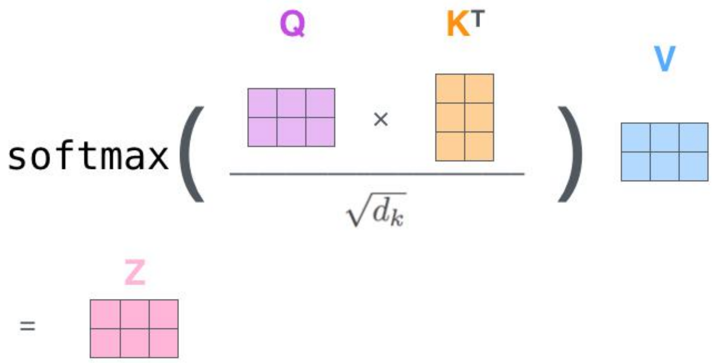


Transformer 工作流程

编码过程 —— Self-Attention（微观）

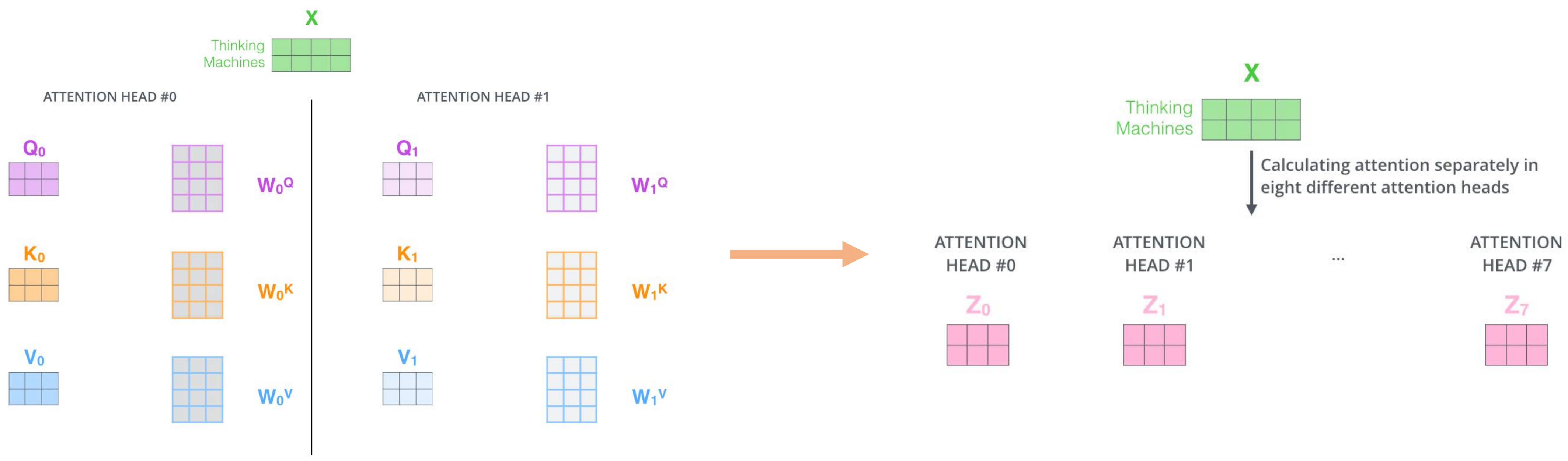
通过矩阵运算实现自注意力机制





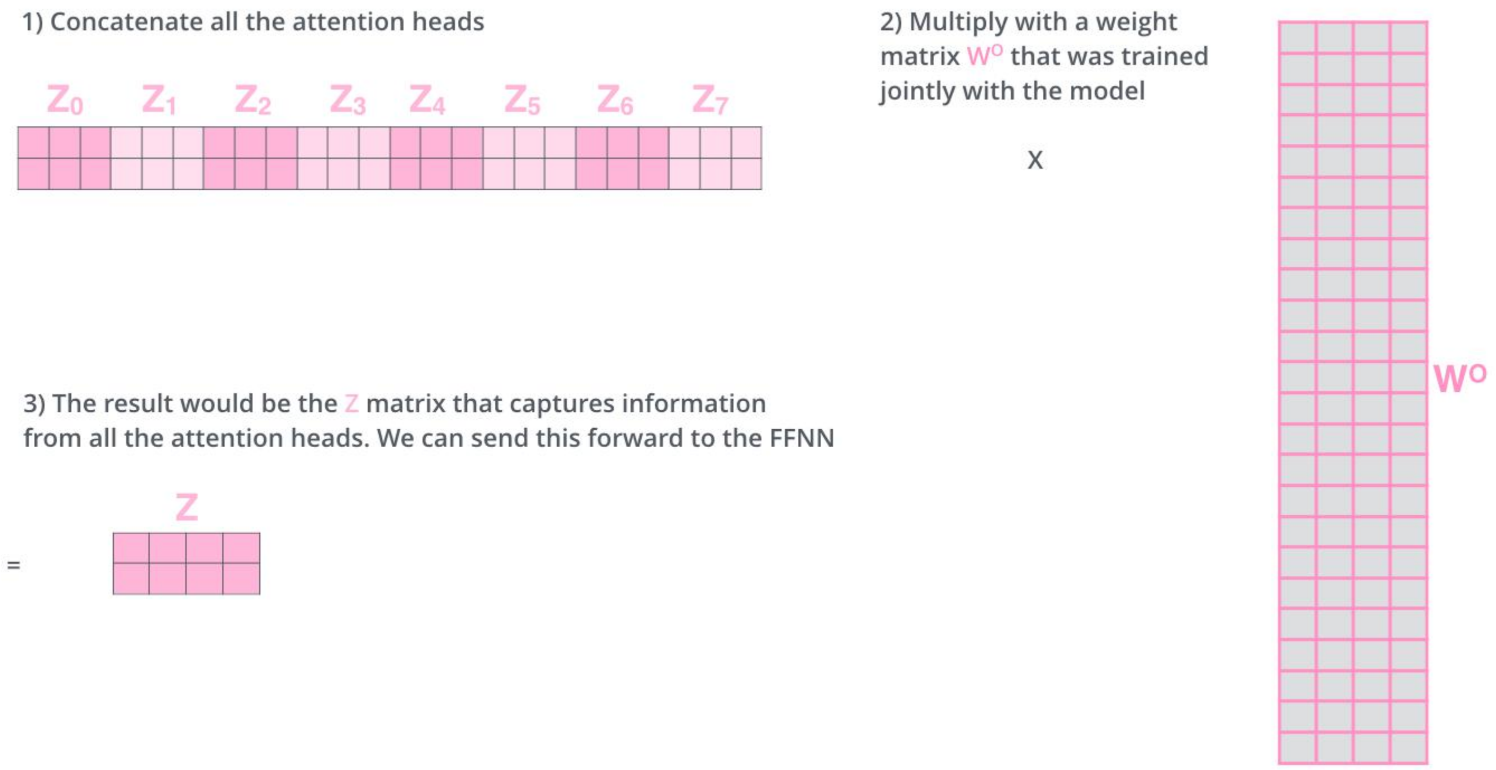
Transformer 工作流程

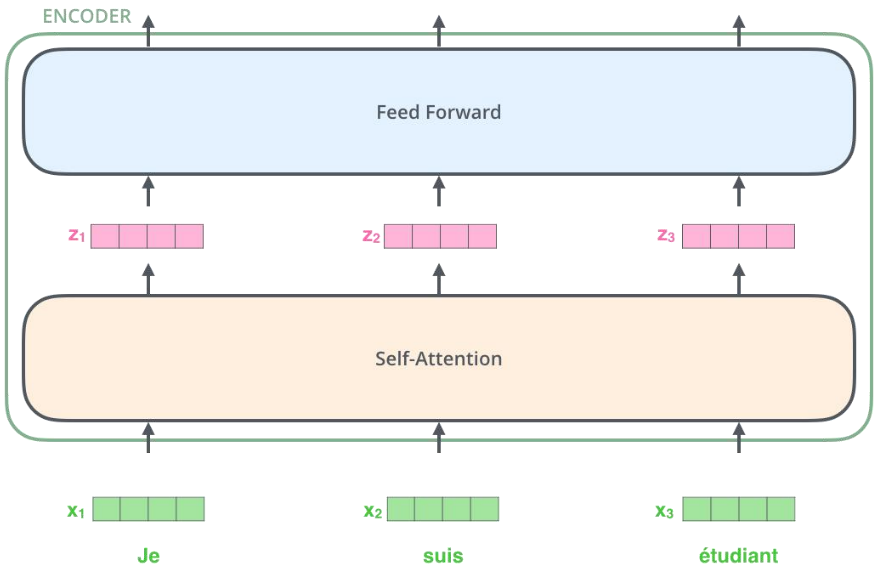
编码过程 —— Multi-Head Attention



Transformer 工作流程

编码过程 —— Multi-Head Attention





Transformer 工作流程

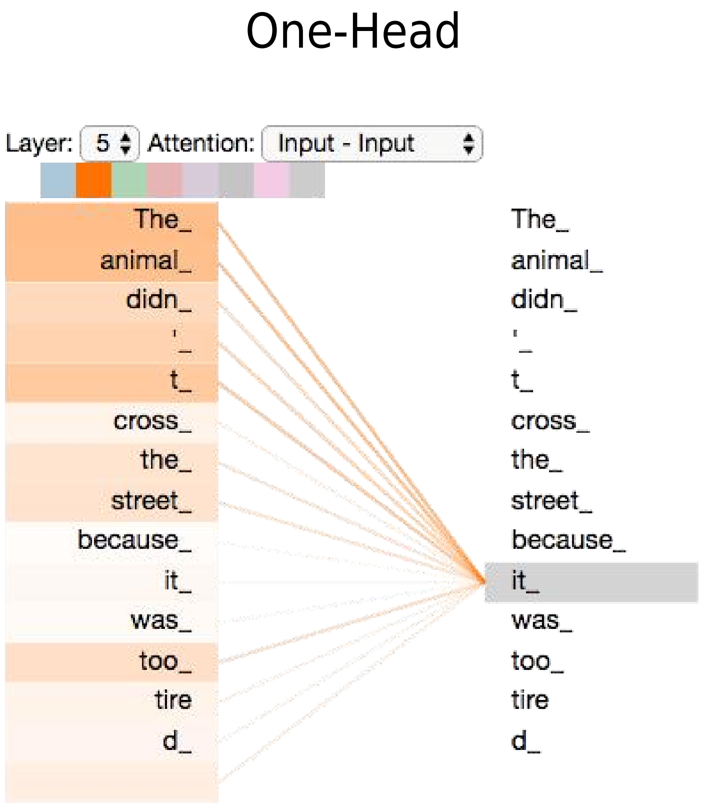
编码过程 —— Multi-Head Attention

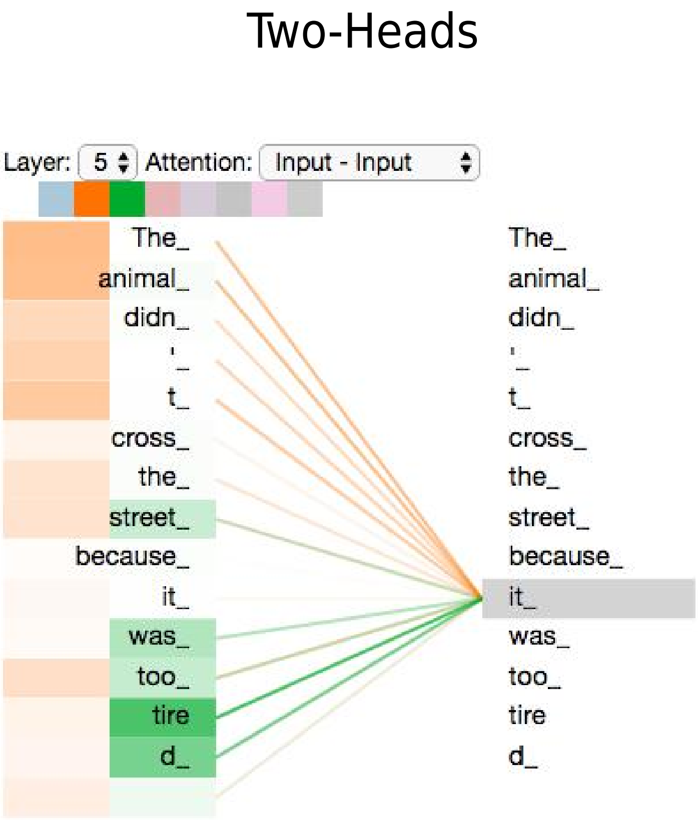
1) This is our input sentence*

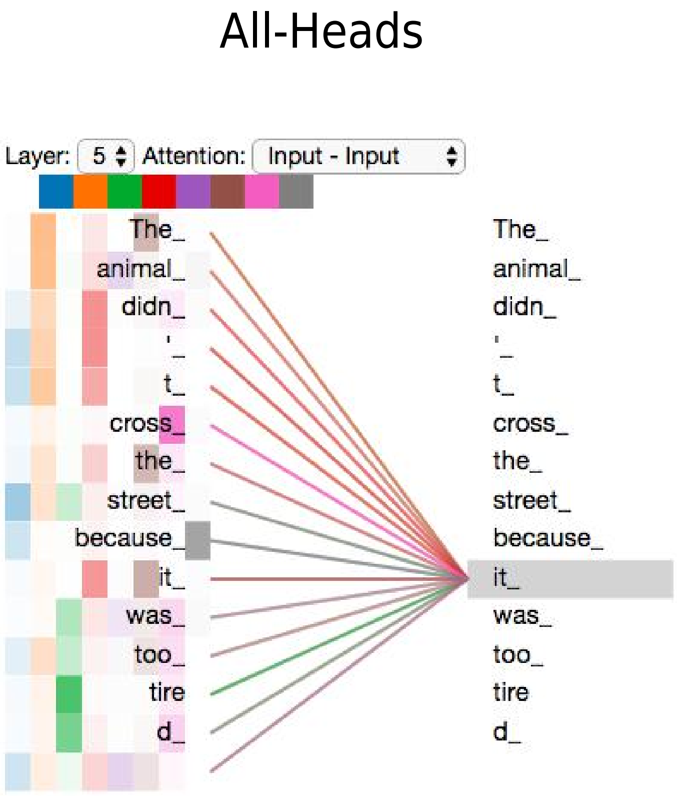
Thinking
Machines

Transformer 工作流程

编码过程 —— Multi-Head Attention



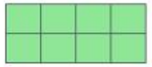




Transformer 工作流程

Padding 操作

X



X：Thinking Machines

X 的维度：[sequence_length, embedding_dimension]

Actually...

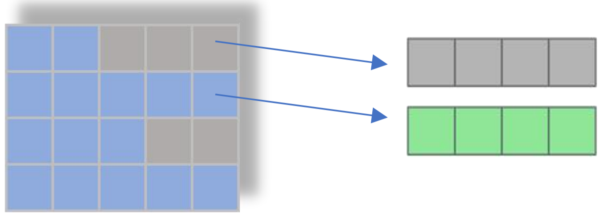
X：Thinking Machines (seq_len: 2)
A Tale of Two Cities (seq_len: 5)
Science and Art (seq_len: 3)
the Art of Motorcycle Maintenance (seq_len: 5)

X 的维度：[batch_size, **max_sequence_length**, embedding_dimension]

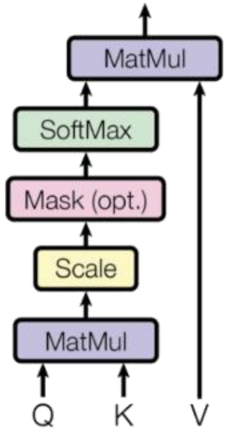
max_sequence_length

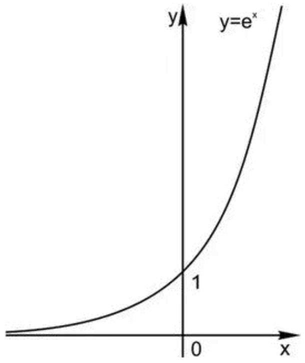
X:

batch_size



Scaled dot-product attention





Softmax函数: $\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$

$e^0 = 1$

$e^{-\infty} \to 0$

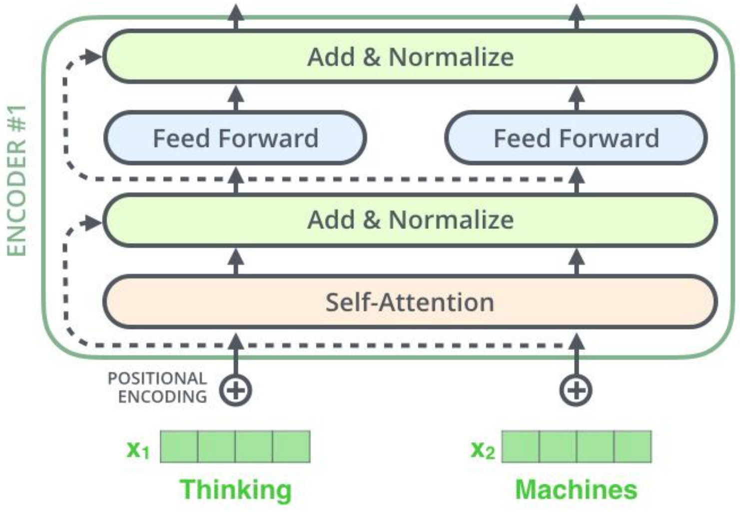
Padding Mask

0	0	1	1	1
0	0	0	0	0
0	0	0	1	1
0	0	0	0	0

0	0	-inf	-inf	-inf
0	0	0	0	0
0	0	0	-inf	-inf
0	0	0	0	0

Transformer 工作流程

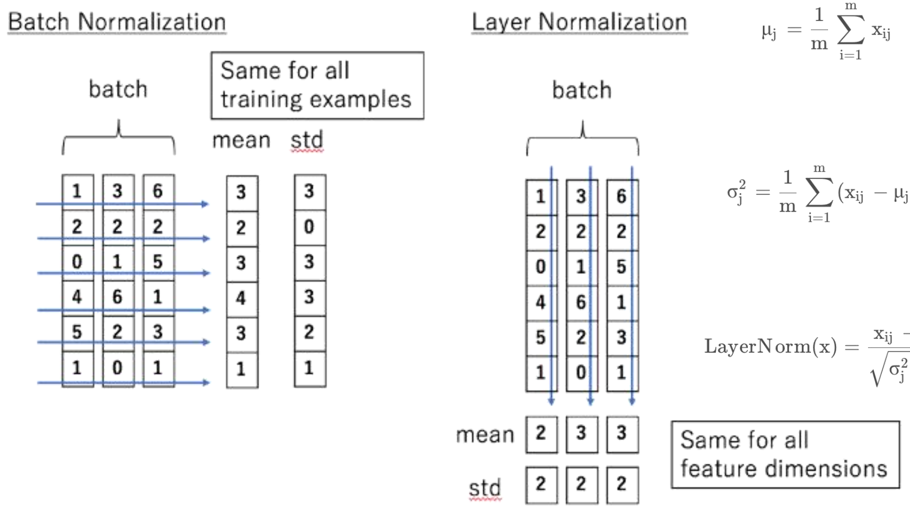
Add & Norm



Add

$$X_{embedding} + \text{Self Attention}(Q, K, V)$$

Layer Norm

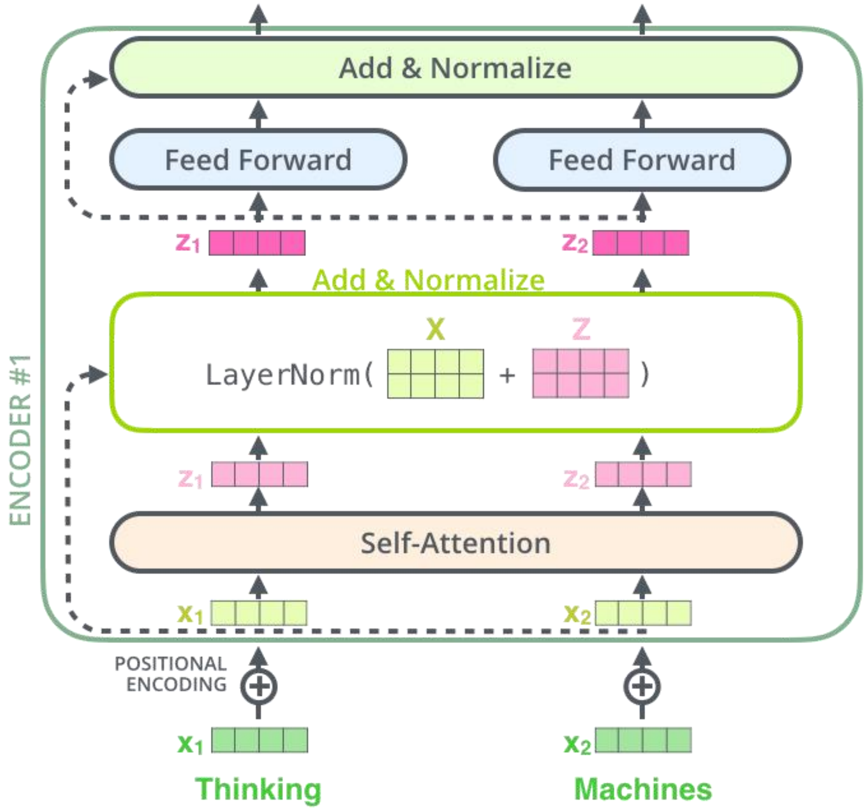


$$\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_{ij}$$

$$\sigma_j^2 = \frac{1}{m}\sum_{i=1}^{m}(x_{ij} - \mu_j)^2$$

$$\text{LayerNorm}(x) = \frac{x_{ij} - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}}$$

Visualization



Transformer 工作流程

编码过程

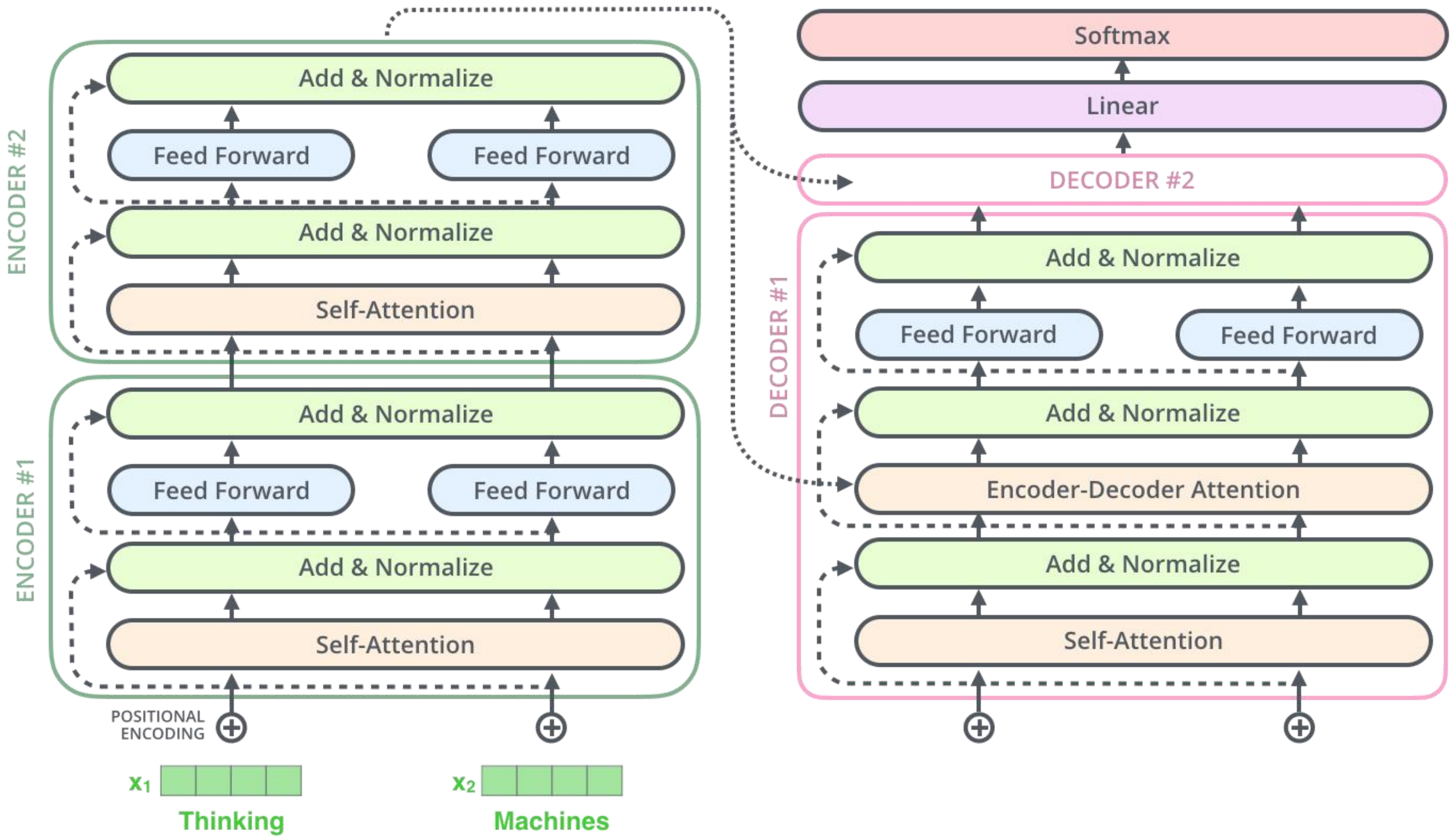
$$X_{hidden} = X_{attention} + X_{hidden}$$
$$X_{hidden} = LayerNorm(X_{hidden})$$

$$X_{hidden} = Linear(ReLU(Linear(X_{attention})))$$

$$X_{attention} = X + X_{attention}$$
$$X_{attention} = LayerNorm(X_{attention})$$

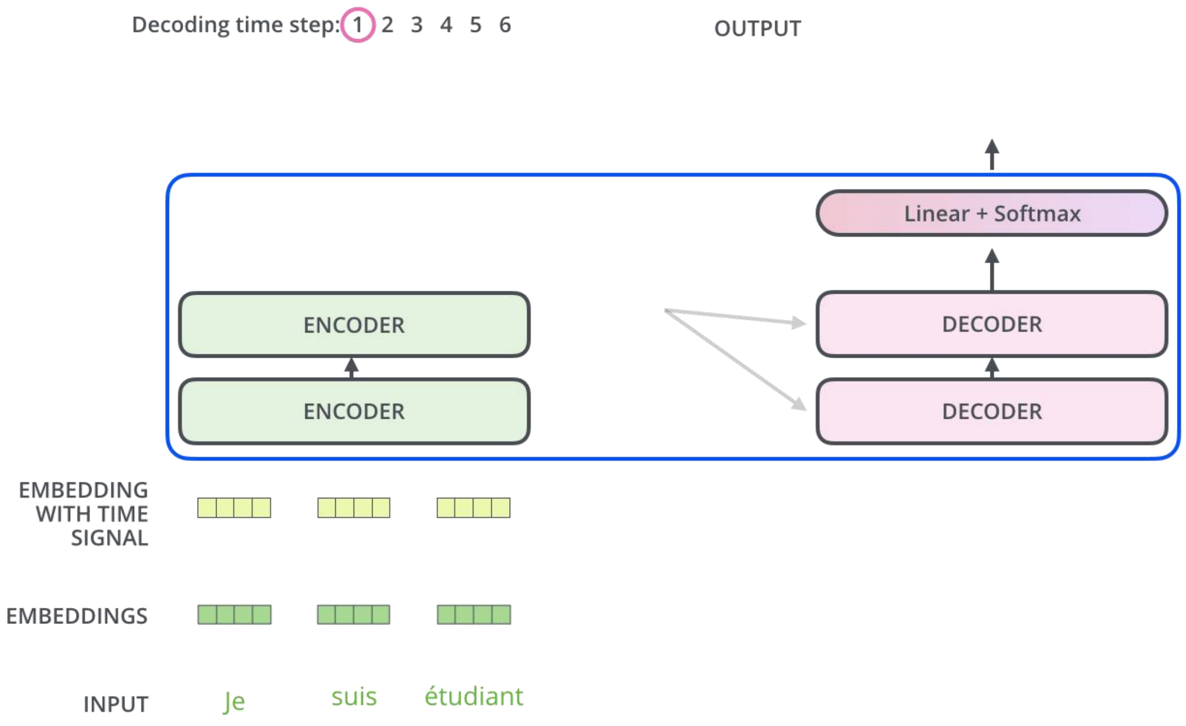
$$Q = Linear(X) = XW_Q$$
$$K = Linear(X) = XW_K$$
$$V = Linear(X) = XW_V$$
$$X_{attention} = SelfAttention(Q, K, V)$$

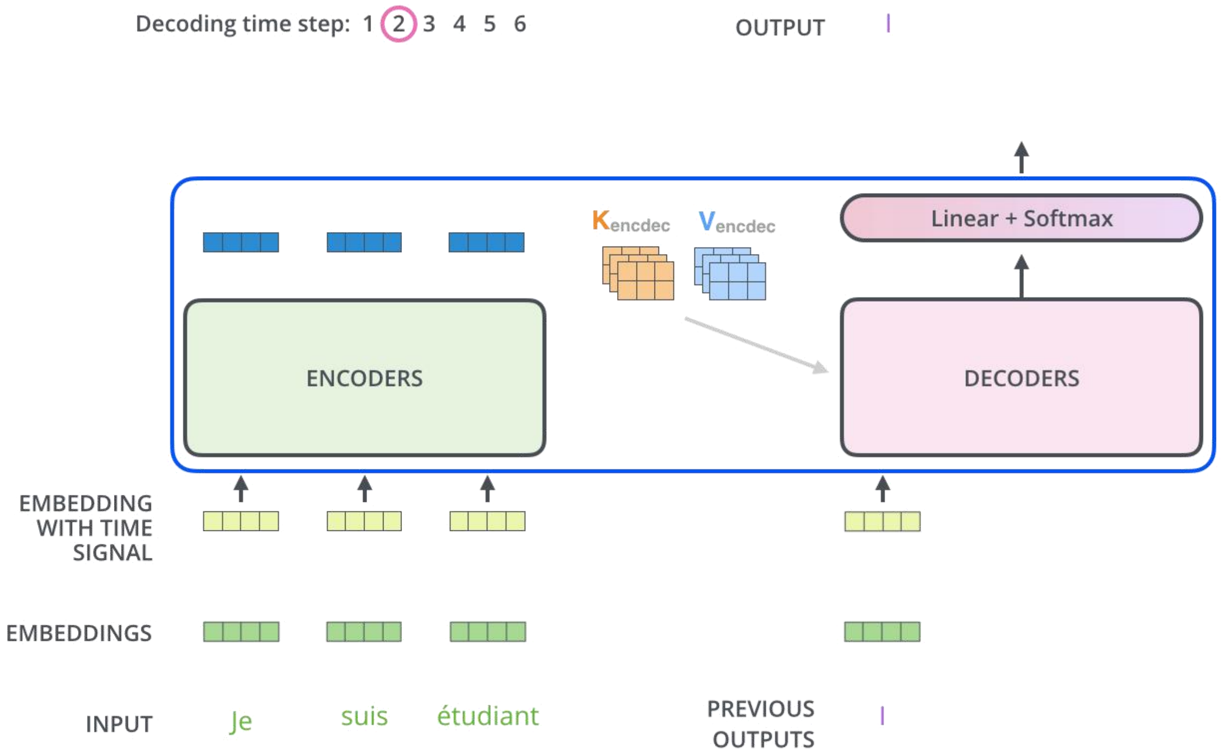
$$X = Embedding\ Lookup(X) + Positional\ Encoding$$



Transformer 工作流程

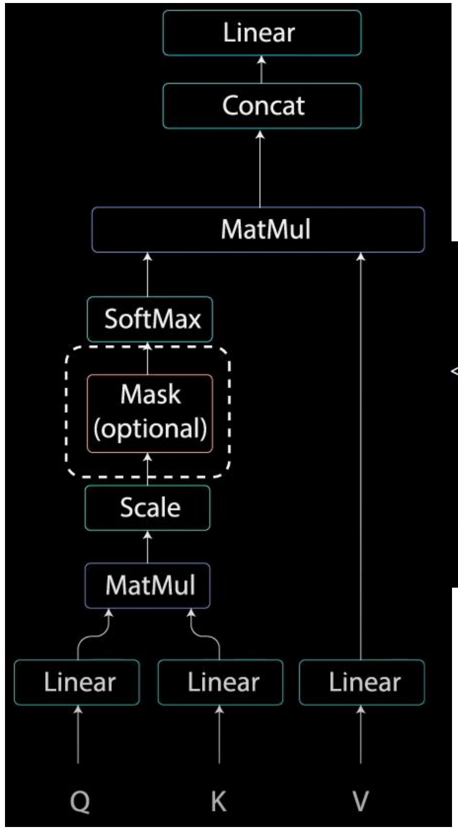
编码过程&解码过程

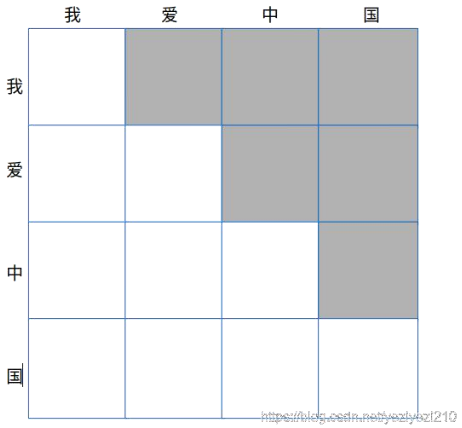




Transformer 工作流程

解码过程 —— Masked Self-Attention

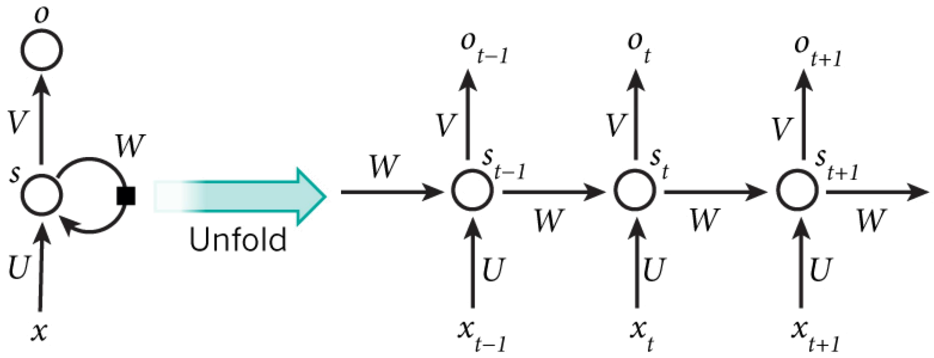


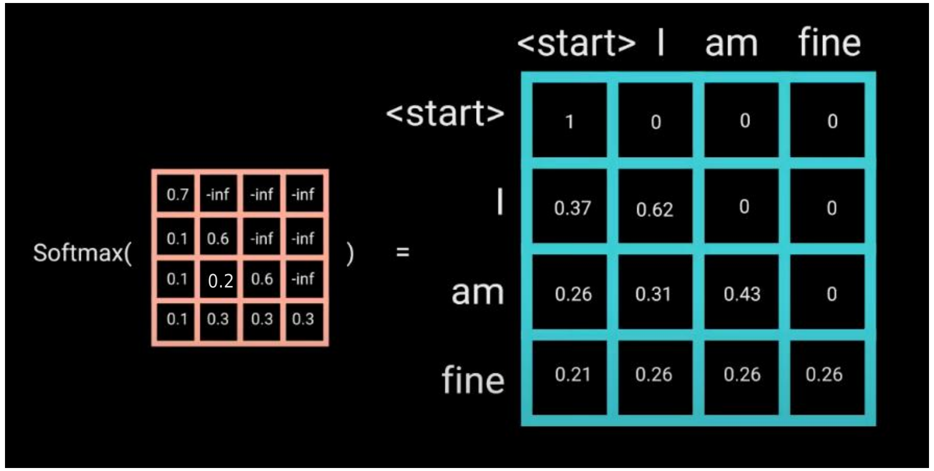



Scaled Scores

	<start>	I	am	fine
<start>	0.7	0.1	0.1	0.1
I	0.1	0.6	0.2	0.1
am	0.1	0.2	0.6	0.1
fine	0.1	0.3	0.3	0.3

RNN模型：




Softmax(

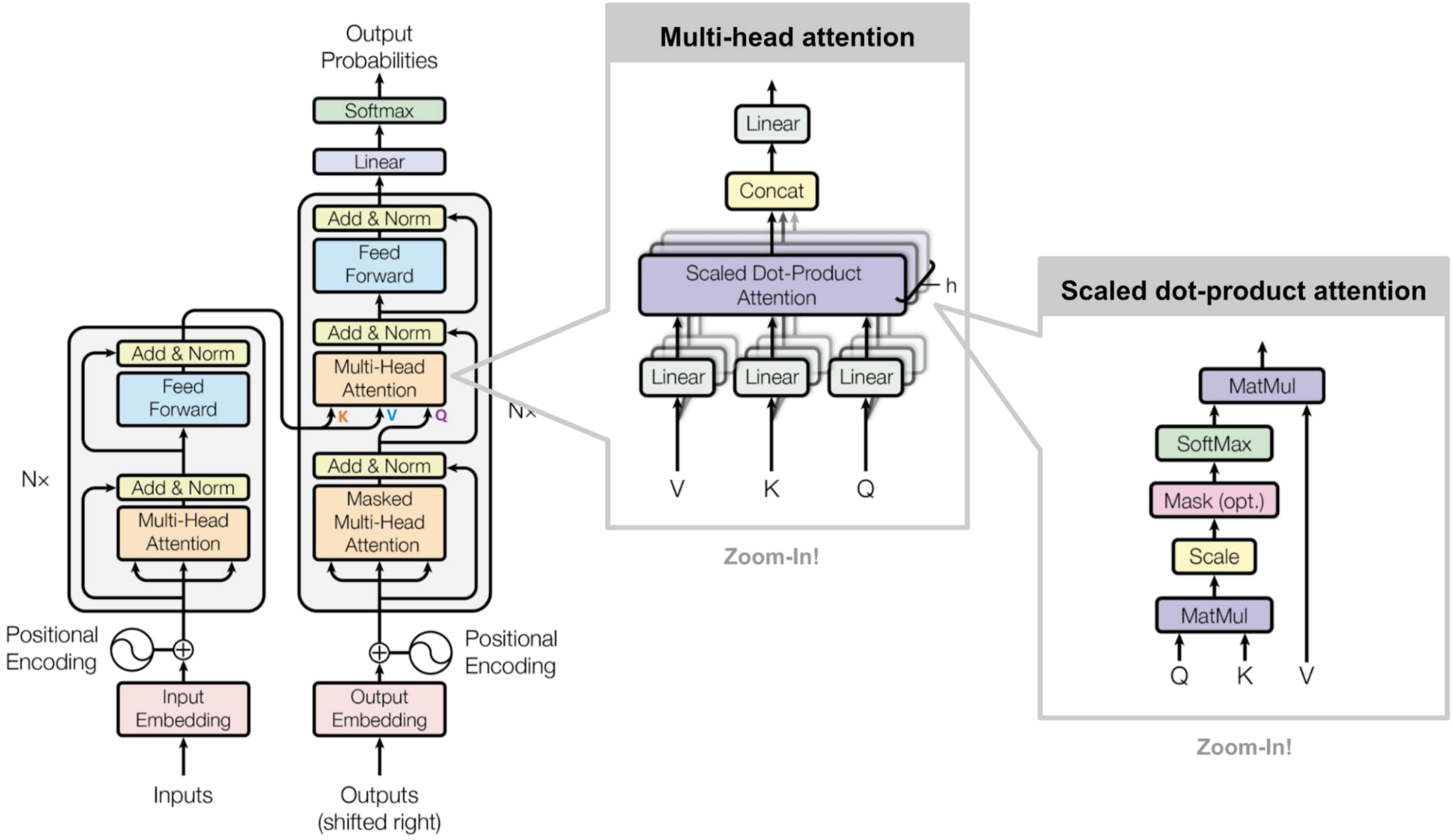
0.7	-inf	-inf	-inf
0.1	0.6	-inf	-inf
0.1	0.2	0.6	-inf
0.1	0.3	0.3	0.3

) =

	<start>	I	am	fine
<start>	1	0	0	0
I	0.37	0.62	0	0
am	0.26	0.31	0.43	0
fine	0.21	0.26	0.26	0.26

Transformer 工作流程

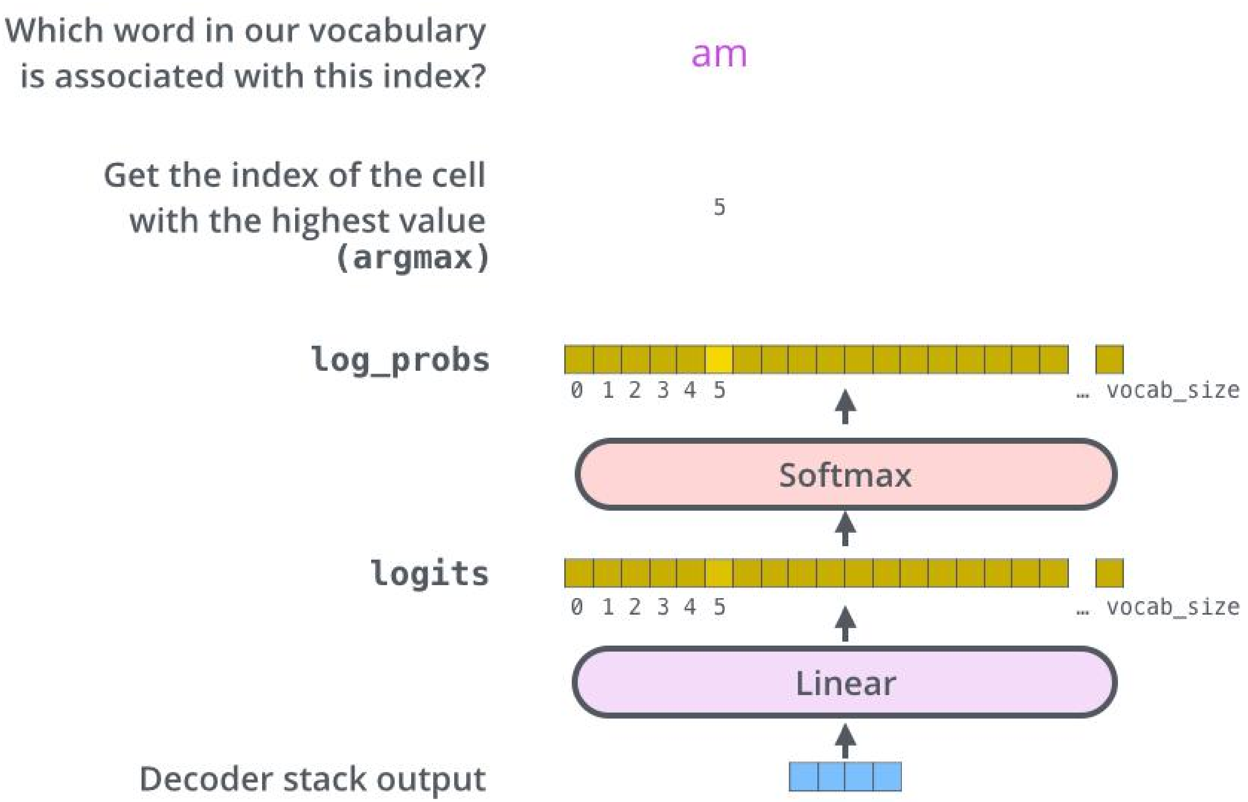
解码过程 —— Masked Encoder-Decoder Attention



- **Encoder Multi-Head Attention:**
  - Q, K, V all from Encoders
- **Decoder Masked Multi-Head Atte**
  - Q, K, V all from Decoders
- **Decoder Multi-Head Attention:**
  - Q from Decoders
  - K, V from Encoders

Transformer 工作流程

解码过程 —— Linear & Softmax



Transformer 训练

Output Vocabulary

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

eos：end of sentence的缩写形式

输入："je suis étudiant"，期望输出："i am a student"

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>

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>

Transformer 预测

解码/采样策略

0.01	0.8	0.1	0.05	0.01	0.03

概率降序排列

0.8	0.1	0.05	0.03	0.01	0.01

缩小采样范围

Top-k (k=3): | 0.8 | 0.1 | 0.05 |

Top-p (p=0.95): | 0.8 | 0.1 | 0.05 | 0.03 |

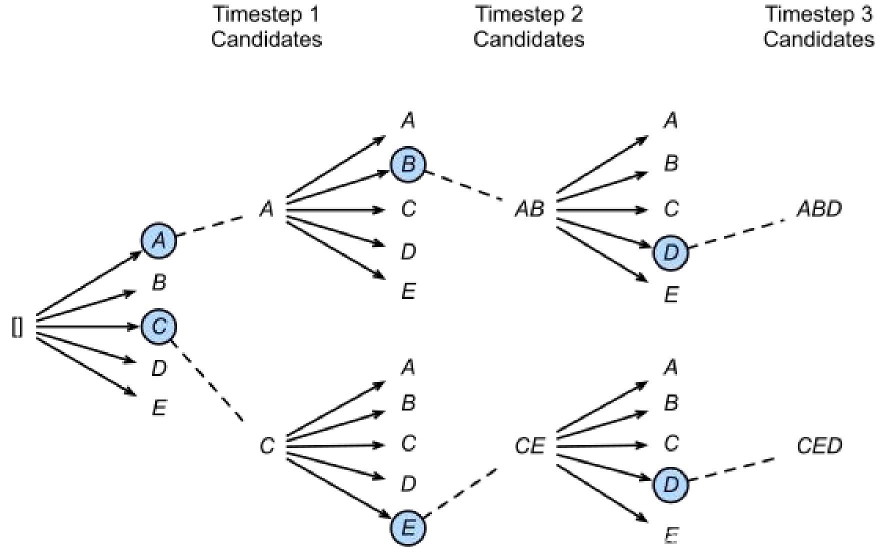
随机采样

Top-k (k=3): | 0.8 |

Top-p (p=0.95): | 0.1 |

Timestep	1	2	3	4
A	0.5	0.1	0.2	0.0
B	0.2	0.4	0.2	0.2
C	0.2	0.3	0.4	0.2
<eos>	0.1	0.2	0.2	0.6

- Argmax Decoding（贪婪采样）
  - Greedy Search（贪心搜索）
  - Beam Search（集束搜索）
- Stochastic Decoding（随机采样）
  - Temperature-controlled Stochastic Sampling
  - Top-k Sampling
  - Top-p Sampling（Nucleus Sampling）
- 混合采样



$$P(x|x_{1:t-1}) = \frac{exp(u_t/t)}{\sum_{t'} exp(u_{t'}/t)}$$ ，其中 $t \in [0,1)$

CTRL: $$p_i = \frac{\exp(x_i/(T \cdot I(i \in g))}{\sum_j \exp(x_j/(T \cdot I(j \in g))} \quad I(c) = \theta \text{ if } c \text{ is True else } 1$$

作业

实现 Transformer 架构，通过单步调试理解数据流动过程及维度变化。

参考代码：https://github.com/wmathor/nlp-tutorial

5. Model based on Transformer

- 5-1. The Transformer - **Translate**
  - Paper - Attention Is All You Need(2017)
  - Colab - Transformer_Torch.ipynb
  - bilibili - https://www.bilibili.com/video/BV1mk4y1q7eK

Model	Example
NNLM	Predict Next Word
Word2Vec(Softmax)	Embedding Words and Show Graph
TextCNN	Sentence Classification
TextRNN	Predict Next Step
TextLSTM	Autocomplete
Bi-LSTM	Predict Next Word in Long Sentence
Seq2Seq	Change Word
Seq2Seq with Attention	Translate
Bi-LSTM with Attention	Binary Sentiment Classification
Transformer	Translate
Greedy Decoder Transformer	Translate
BERT	how to train

参考资料

- Attention is All you Need (acm.org)
- The Illustrated Transformer – Jay Alammar – Visualizing machine learning one concept at a time. (jalammar.github.io)
- The Annotated Transformer (harvard.edu)
- wmathor.com
- Transformer详解_数学家是我理想的博客-CSDN博客
- 图解Transformer（完整版）_龙心尘的博客-CSDN博客
- 如何优雅地编码文本中的位置信息？三种positioanl encoding方法简述 (qq.com)
- NLP基础模型和注意力机制_开始King的博客-CSDN博客
- 文本生成自回归解码策略总结_自回归解码器_Meilinger_的博客-CSDN博客

Q & A