Transformer

李安贞

目 录

Contents

1 Encoder-Decoder

Q

如何将input转换成特征？

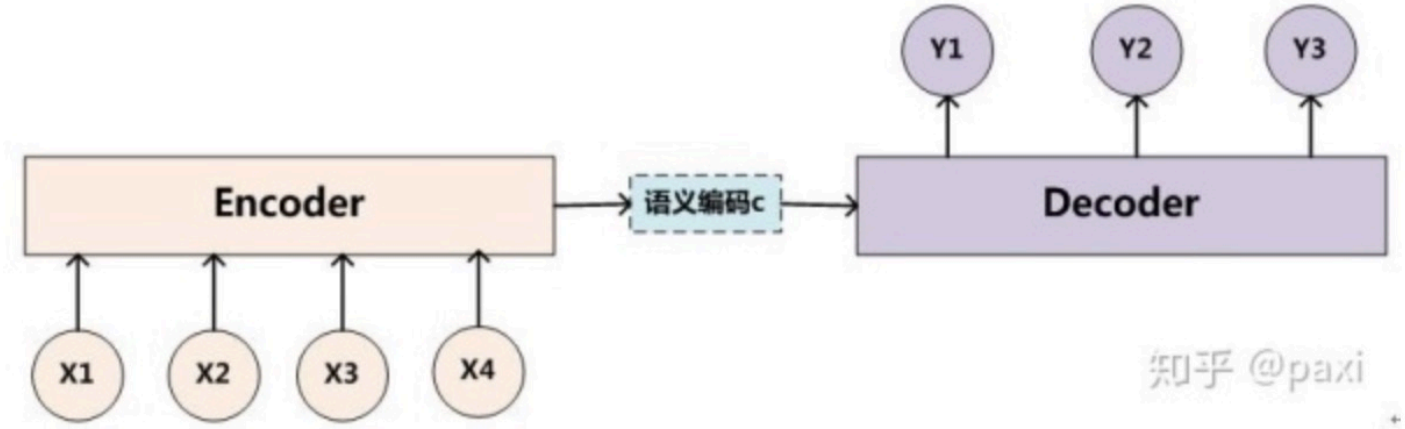
A

Feature engineering

数据预处理 特征选择 降维 ……

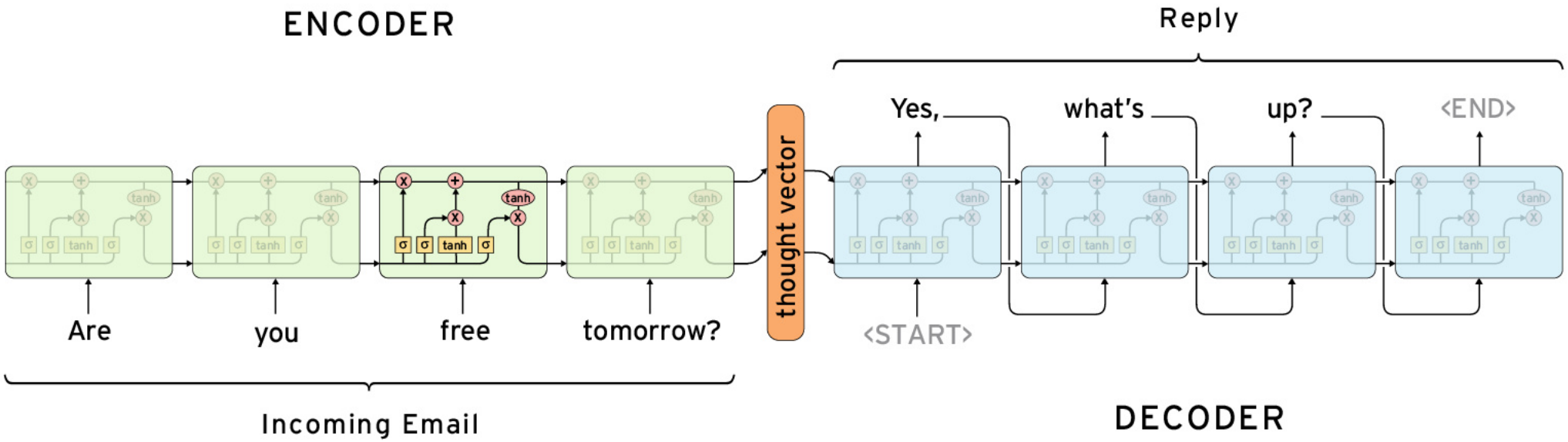
Representation Learning(Feature Learning)

Encoder-Decoder结构：是一种想要提升神经网络特征表示能力的架构。

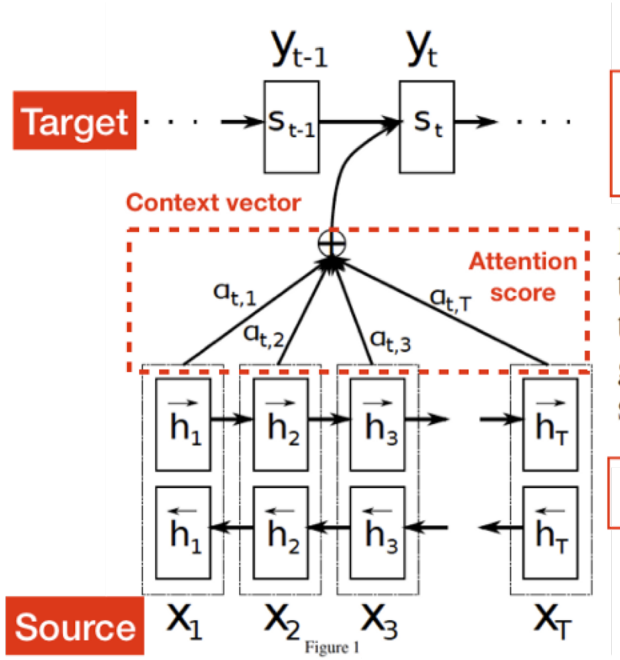


1 Encoder-Decoder

Sequence to Sequence



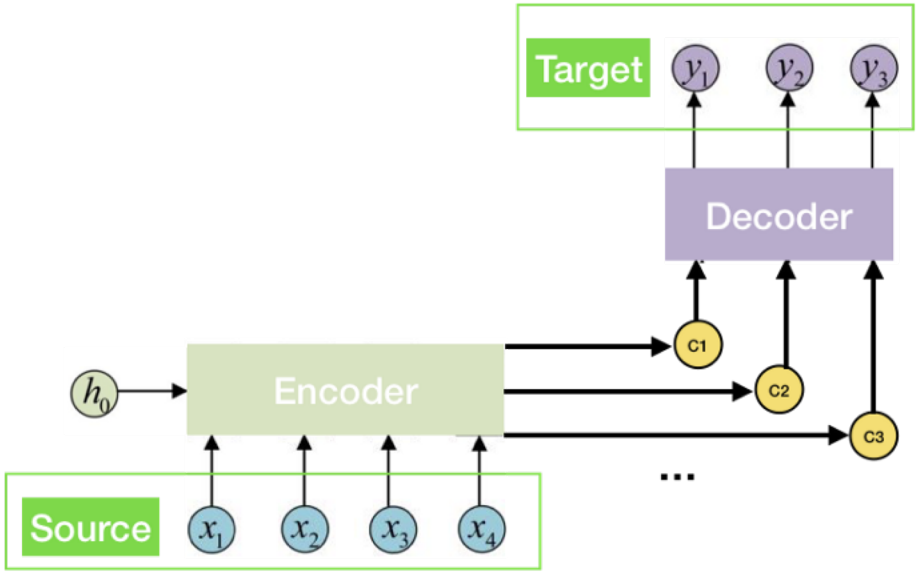
2 Attention



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

Figure 1: The graphical illustration of the proposed model trying to generate the t-th target word y_t given a source sentence $(x_1, x_2, \ldots, x_T)$.

Encoder: Bi-direcntional RNN



$$\underline{score}\ e_{ij} = a(s_{i-1},\ h_j)$$

2 Attention

$$\underline{score}\ e_{ij} = a(s_{i-1},\ h_j)$$

3.1 Global Attention

The idea of a global attentional model is to consider all the hidden states of the encoder when deriving the context vector c_t. In this model type, a variable-length alignment vector $\boldsymbol{a}_t$, whose size equals the number of time steps on the source side, is derived by comparing the current target hidden state $\boldsymbol{h}_t$ with each source hidden state $\bar{\boldsymbol{h}}_s$:

$$\begin{aligned} \boldsymbol{a}_t(s) &= \text{align}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) \\ &= \frac{\exp\left(\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s)\right)}{\sum_{s'} \exp\left(\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_{s'})\right)} \end{aligned} \tag{7}$$

Here, score is referred as a *content-based* function for which we consider three different alternatives:

$$\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) = \begin{cases} \boldsymbol{h}_t^\top \bar{\boldsymbol{h}}_s & \textit{dot} \\ \boldsymbol{h}_t^\top \boldsymbol{W}_{\boldsymbol{a}} \bar{\boldsymbol{h}}_s & \textit{general} \\ \boldsymbol{W}_{\boldsymbol{a}}[\boldsymbol{h}_t; \bar{\boldsymbol{h}}_s] & \textit{concat} \end{cases} \tag{8}$$

Multiplicative attention：*Luong, M.-T., Pham, H., & Manning, C. D. (2015). Effective Approaches to Attention-based Neural Machine Translation. EMNLP 2015.*

Additive attention：*Bahdanau, D., Cho, K., & Bengio, Y.. Neural Machine Translation by Jointly Learning to Align and Translate. ICLR 2015.*

2 Attention

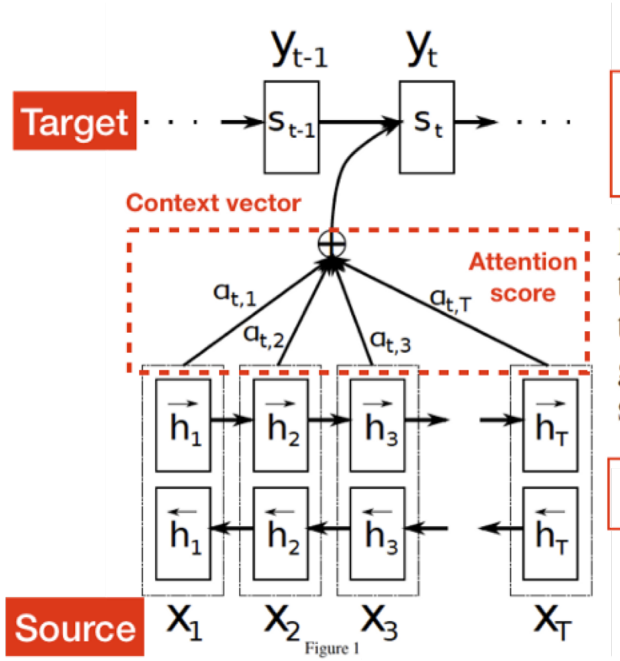
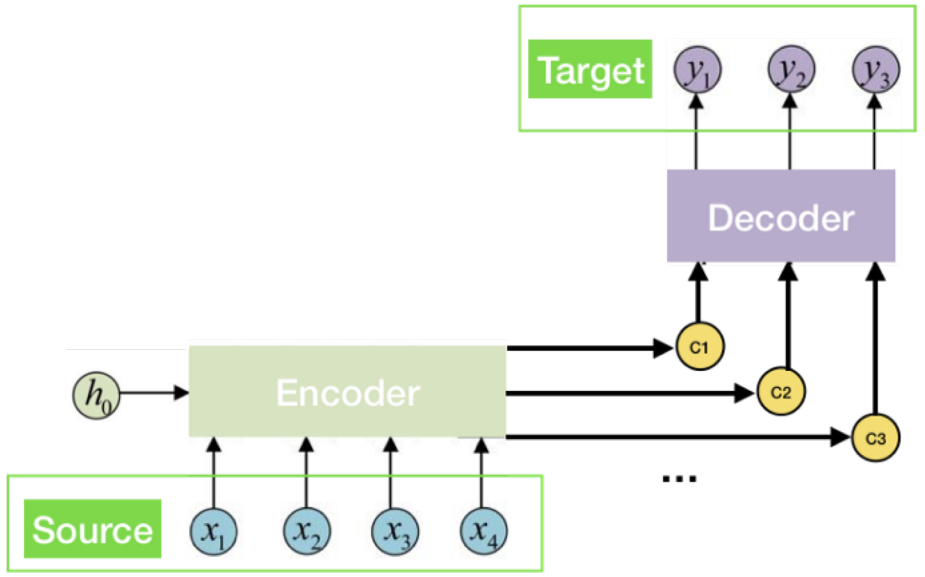
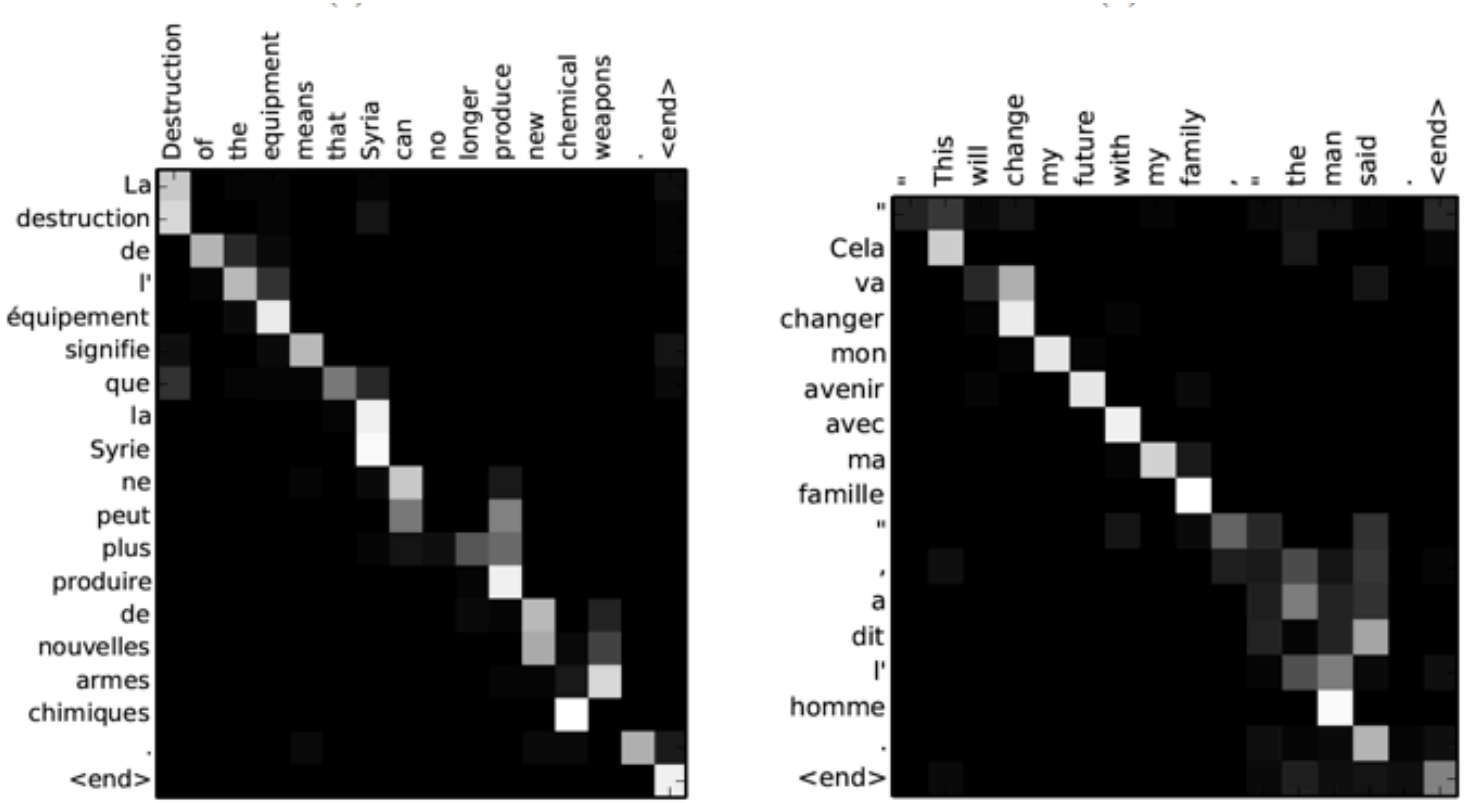


Figure 1: The graphical illustration of the proposed model trying to generate the t-th target word y_t given a source sentence $(x_1, x_2, \ldots, x_T)$.



$$\underline{Attention\ score}\ \alpha_{ij} = e^{e_{ij}} / \sum_{k=1}^{T_x} e^{e_{ik}}$$

2 Attention



x轴和y轴分别对应输入、输出句，每个像素表示输入句中的第j个文字对应目标句中第i个文字的权重大小，亦即attention score(0 : 黑，1 : 白)。

2 Attention

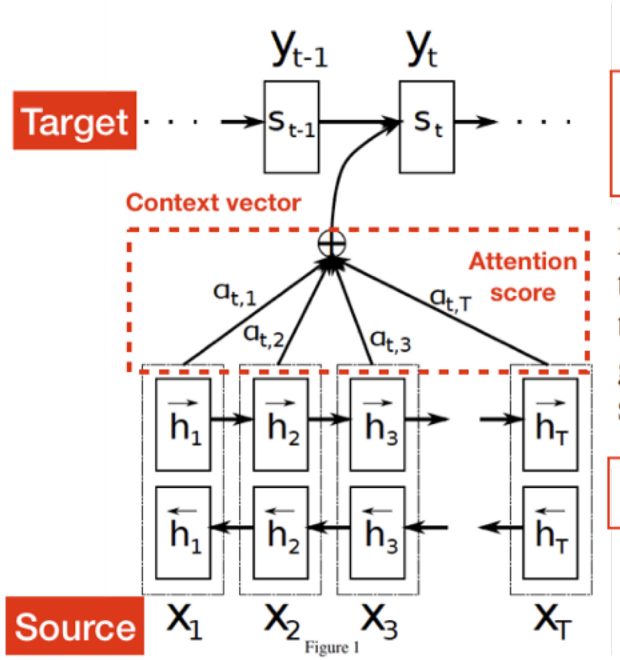
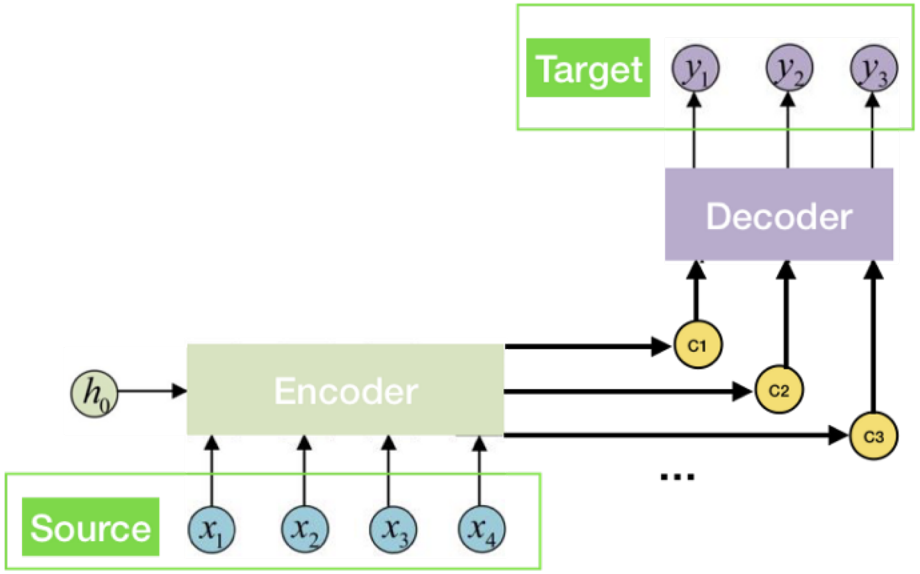


Figure 1: The graphical illustration of the proposed model trying to generate the t-th target word y_t given a source sentence $(x_1, x_2, \ldots, x_T)$.



$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$$

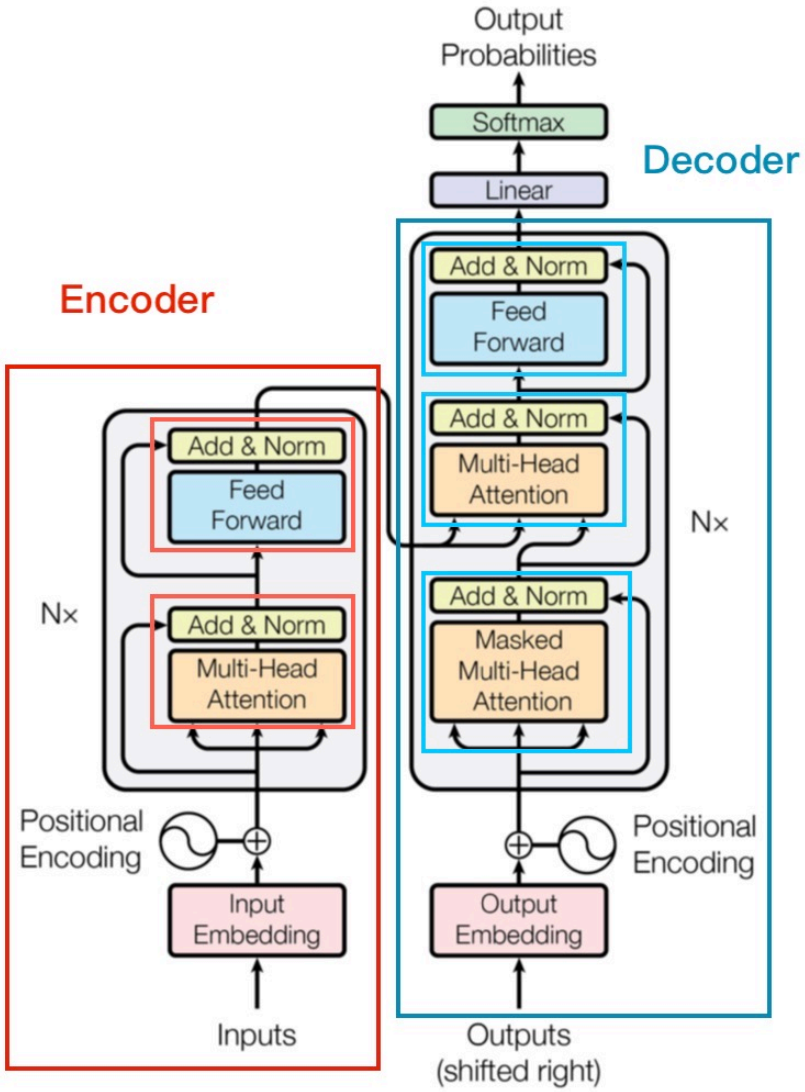
3 Attention Is All You Need

Attention model的缺陷：

- context vector计算的是输入句与目标句之间的关联，而忽略了输入句的文字之间的关联，同样也忽略了输出句的文字之间的关联。
- RNN的缺点是无法并行化处理，导致模型训练时间很长。

解决方法：

- Convolutional Seq2seq learning
- Attention is all your need——Transformer



3 Attention Is All You Need

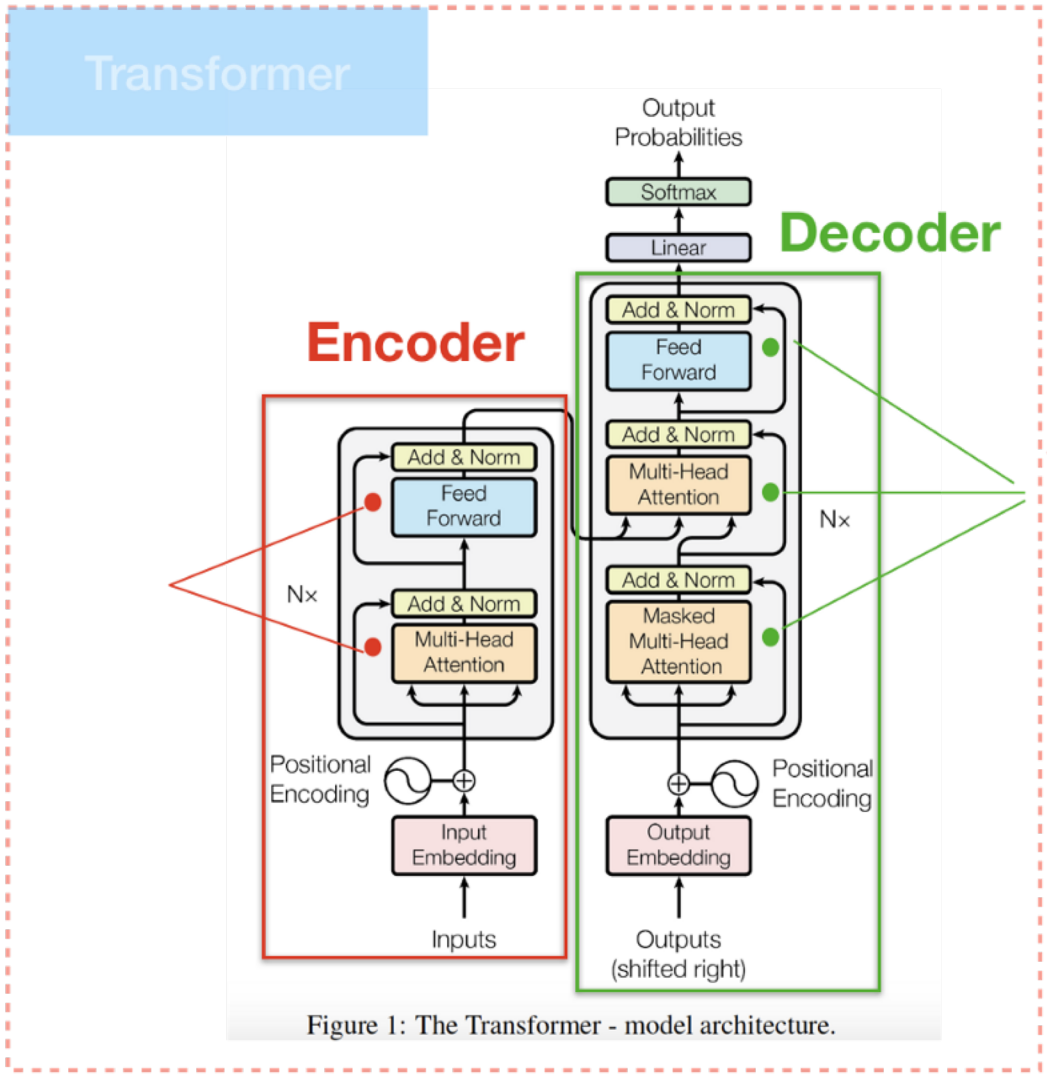
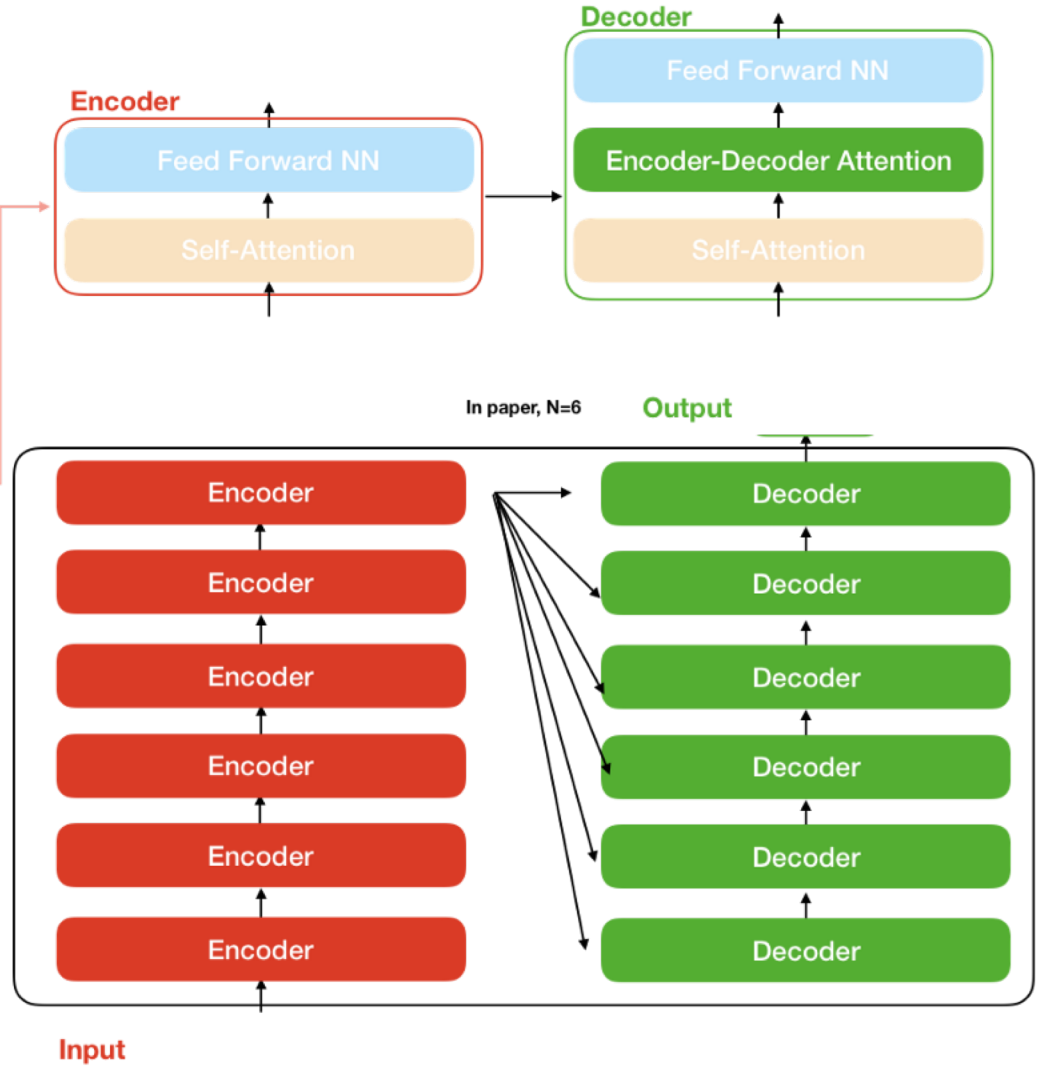


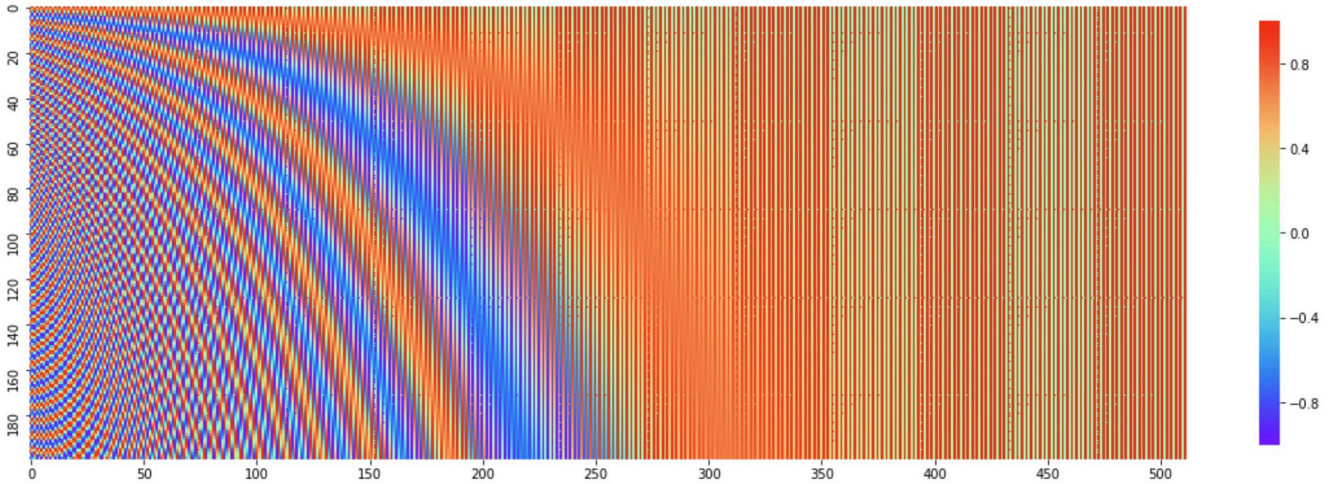
Figure 1: The Transformer - model architecture.



3.1 Positional Encoding

問題

Multi-head不能捕捉序列順序，
如果打亂句子的詞序，
Attention結果一樣



解法 Positional Encoding

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

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

$if\ shape\ of\ enc, dec\ = [T, d_{model}]$ pos: 詞的位置

$\rightarrow then\ pos \in [0, T), i \in [0, d_{model})$ i: d_{model} 的第i個元素

優點： 位置相隔的詞，其位置關係是**線性關係**

→ **位置 i+k 可表示為 i 向量的線性變換**

$$sin(\alpha+\beta) = sin\alpha cos\beta + cos\alpha sin\beta$$

$$cos(\alpha+\beta) = cos\alpha cos\beta - sin\alpha sin\beta$$

∵ 三角函數週期性

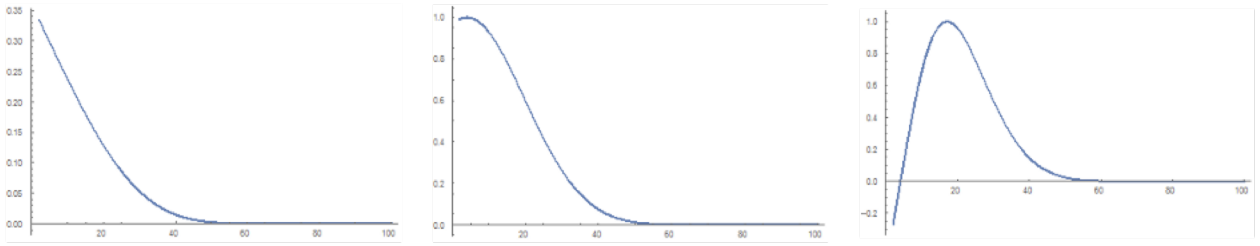
∴ **學到絕對、相對位置關係**

最後 **Word embedding + positional encoding**

(sum、concat皆可)

$if\ d_{model} = 512:$

pos=1 pos=5 pos=10

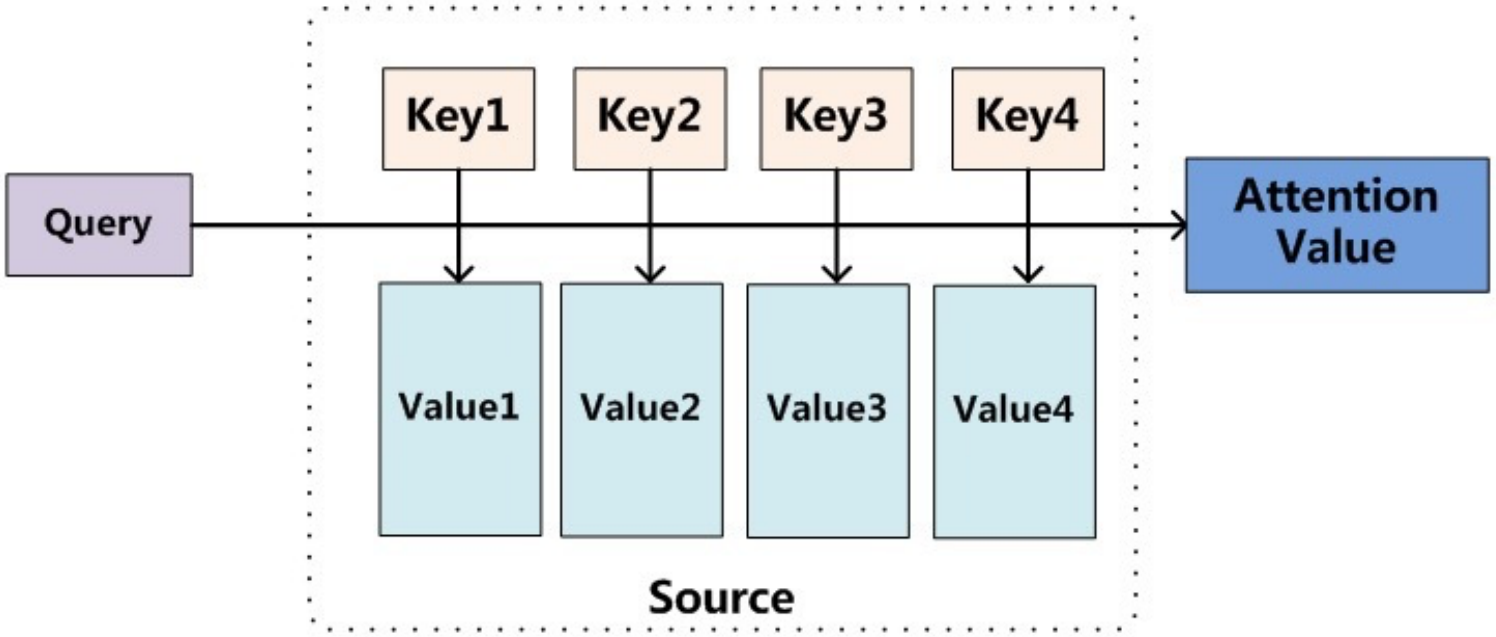


→ 三張圖很像進行平移縮放

learned positional embedding： Convolutional Sequence to Sequence Learning

3.2 Attention

Key, Value, Query



An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.

Alexander H. Miller. et al. (2016) Key-Value Memory Networks for Directly Reading Documents

3.2 Attention

Dot-Product Attention

Decoder Formula

The context vector c_i is, then, computed as a weighted sum of these annotations h_i:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \quad \leftrightarrow \quad \text{Attention}(\text{Query}, \text{Source}) = \sum_{i=1}^{L_x} a_i \cdot Value_i$$

Attention vector; Attention score; Hidden state — 階段 3

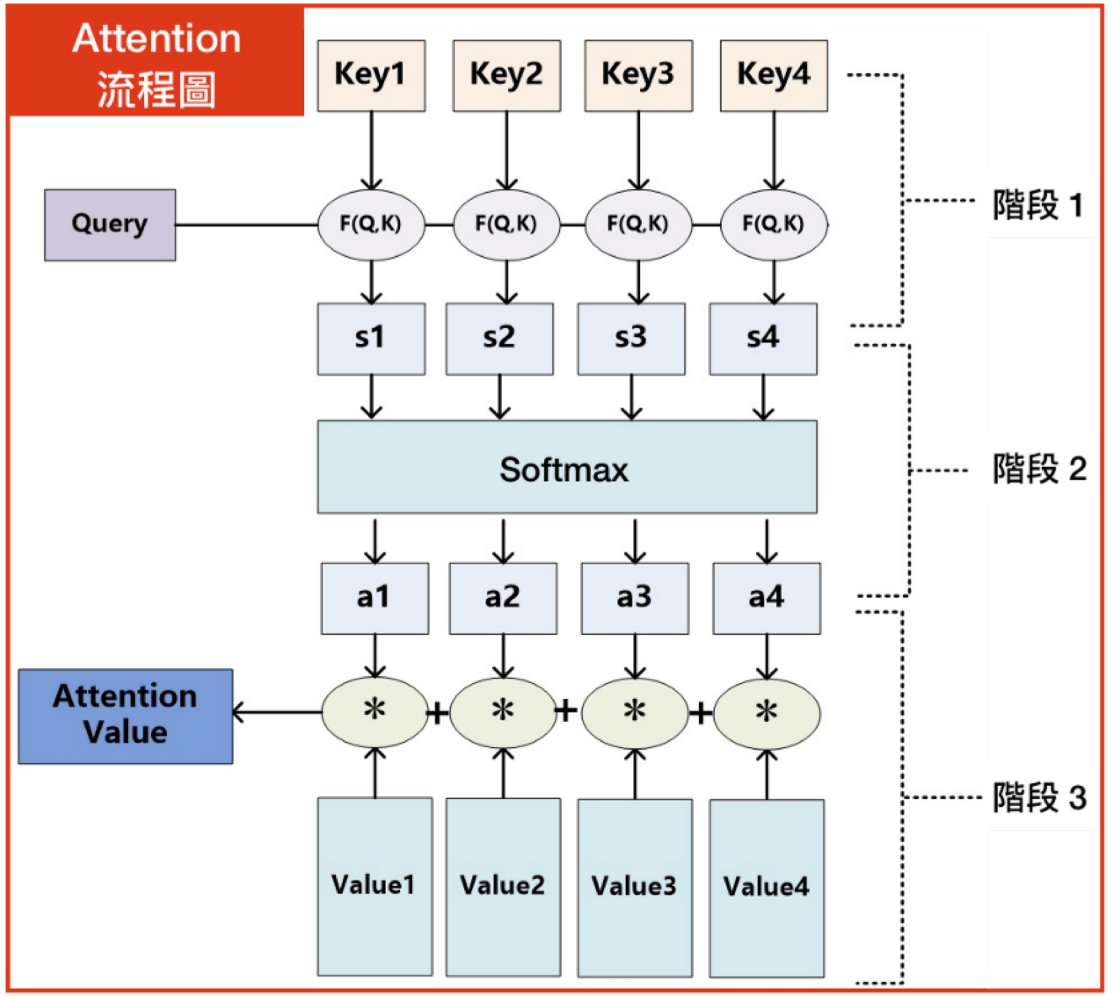
The weight α_{ij} of each annotation h_j is computed by

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}, \quad \leftrightarrow \quad a_i = Softmax(Sim_i) = \frac{e^{Sim_i}}{\sum_{j=1}^{L_x} e^{Sim_j}}$$

Attention score — 階段 2

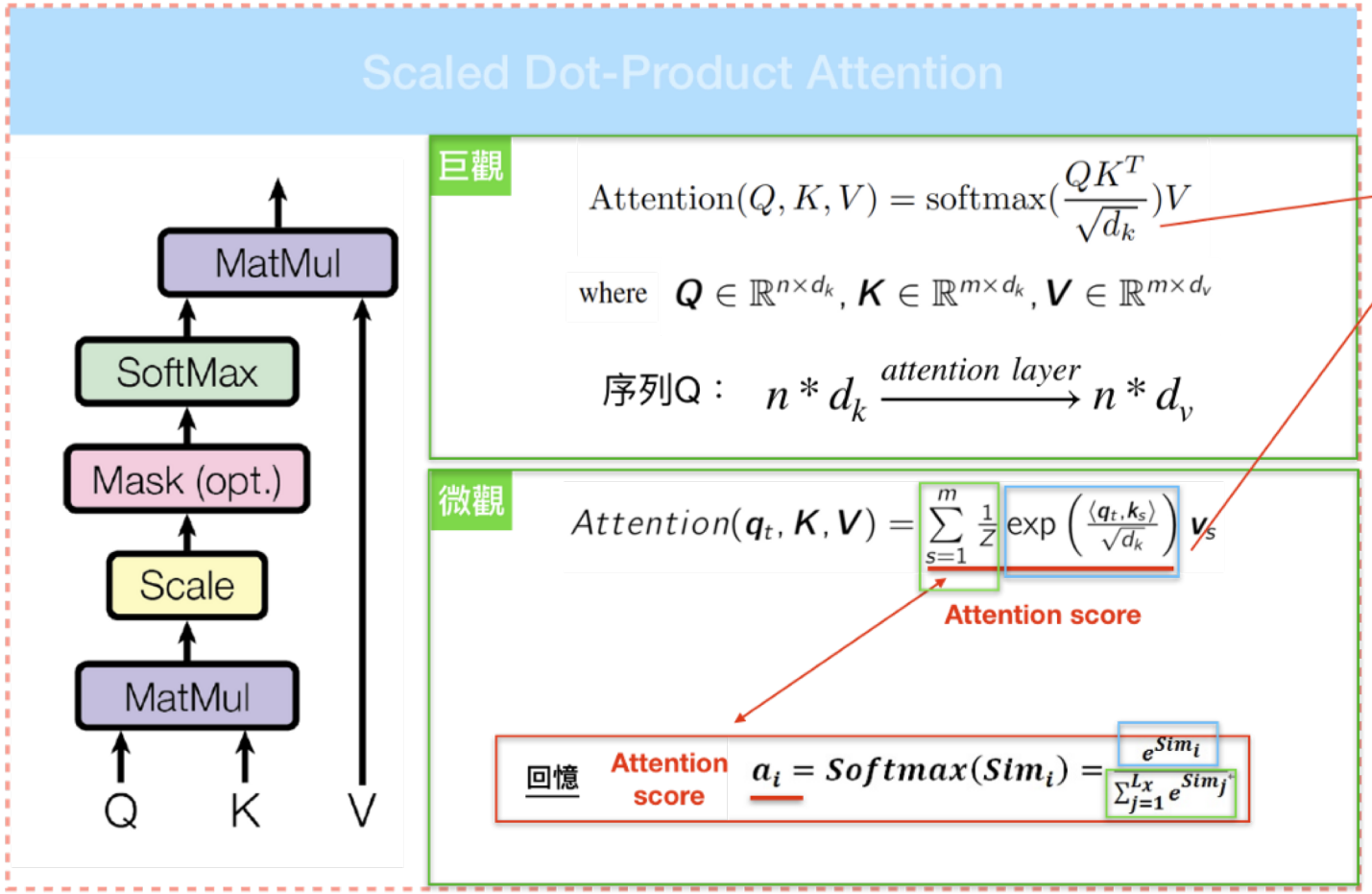
where $e_{ij} = a(s_{i-1}, h_j)$ $\leftrightarrow$ $\text{Similarity}(\text{Query}, Key_i) = \text{Query} \cdot Key_i$ *dot* — 階段 1

is an *alignment model* which scores how well the inputs around position j and the output at position i match. The score is based on the RNN hidden state s_{i-1} (just before emitting y_i, Eq. (4)) and the j-th annotation h_j of the input sentence.



3.2 Attention

Scaled Dot-Product Attention



Hint

Q: 為什麼要除以根號 $\sqrt{d_k}$ ？

A:

$$\because \quad q\cdot k=\sum_{i=1}^{d_k} q_i\,k_i$$

假定 q_i , k_i 為 標準常態分配，

$$\because \quad Var\left(\sum_{i=1}^{m}X_i\right)=\sum_{i=1}^{m}Var(X_i)$$

$q\cdot k=\sum_{i=1}^{d_k}q_ik_i$, has mean 0 and variance d_k.

如果 d_k 很大，內積總和會很大，

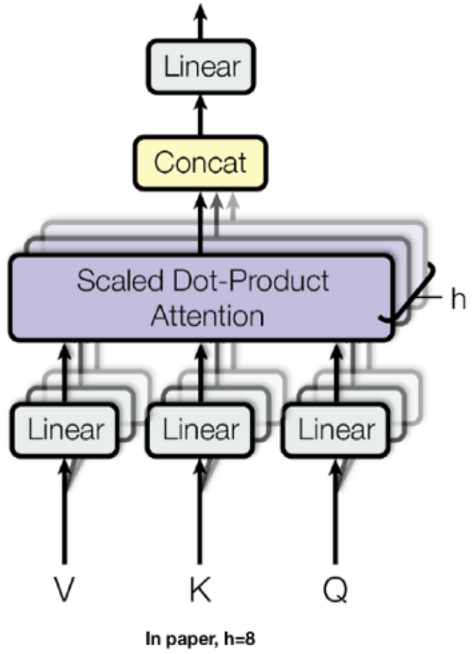
使得softmax非0即1，

因此透過 $\sqrt{d_k}$ 調節規模

3.2 Attention

Multi-head Attention

Multi-head Attention



In paper, h=8

公式

$$\text{MultiHead}(Q,K,V) = \text{Concat}(\text{head}_1,...,\text{head}_\text{h})W^O$$

$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

$$W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}, W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}, W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v} \text{ and } W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}.$$

$$d_k = d_v = d_{model}/h = 64\ ,\ h = 8,\ d_{model} = 512$$

解釋

d_{model}線性變換成h個d_k/d_v部分，

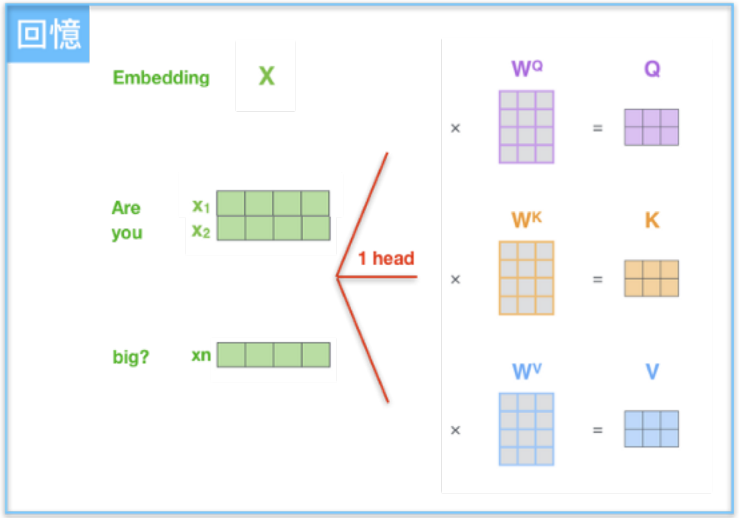
針對這h個部分，各自做attention，

concat再一起進行線性變換= d_{model}，

類似cnn進行h次計算，

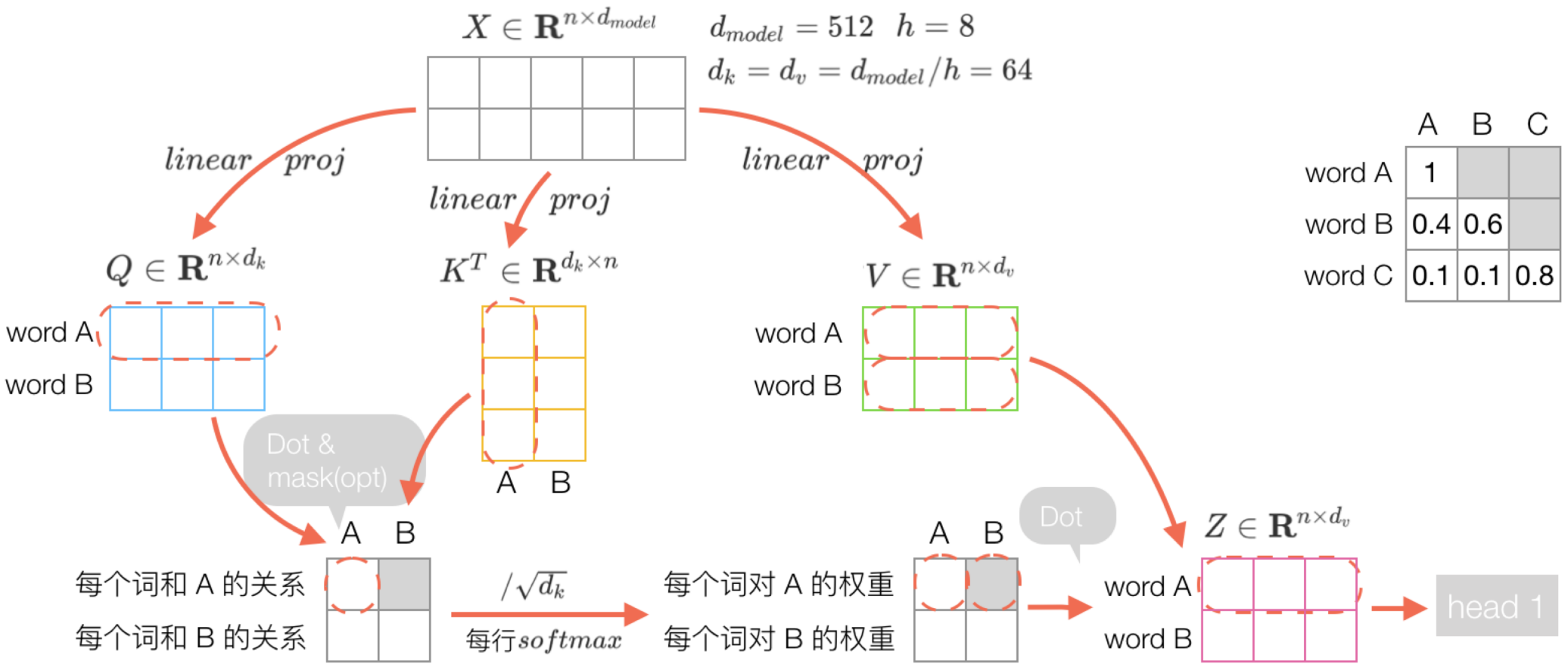
可在不同空間學到更多訊息。

回憶



3.2 Attention

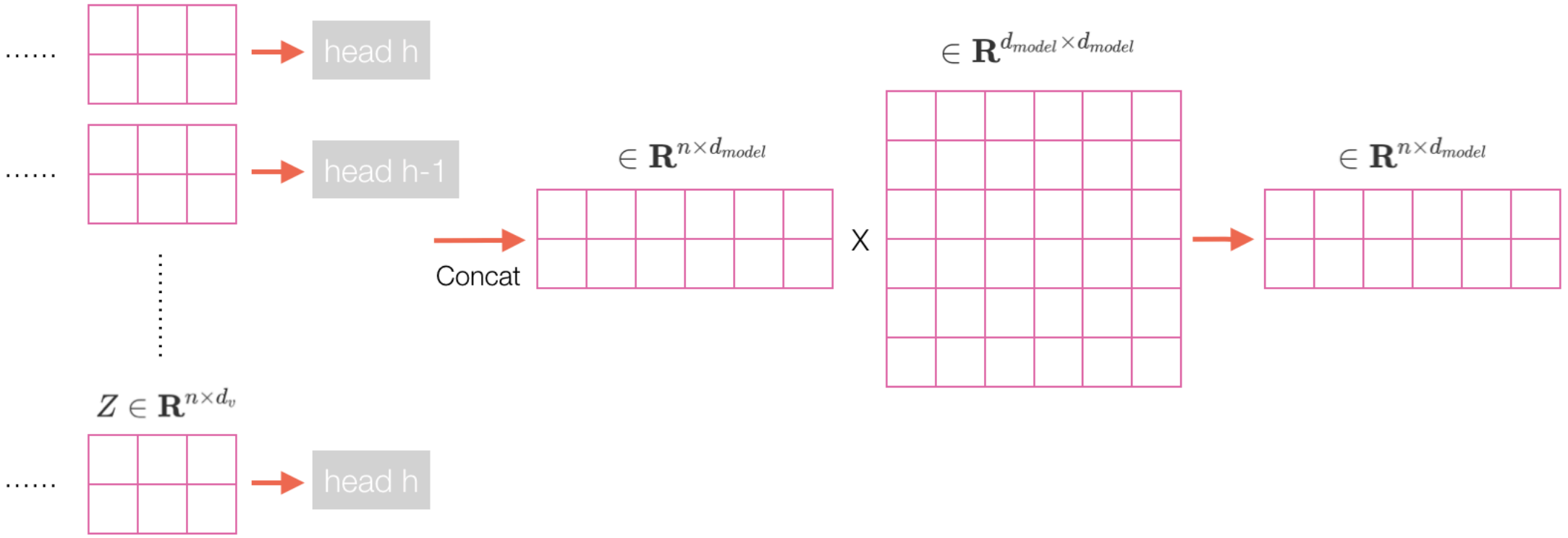
Multi-head Attention 1



	A	B	C
word A	1		
word B	0.4	0.6	
word C	0.1	0.1	0.8

3.2 Attention

Multi-head Attention 2



3.2 Attention

Applications of Attention in Transformer

Transformer

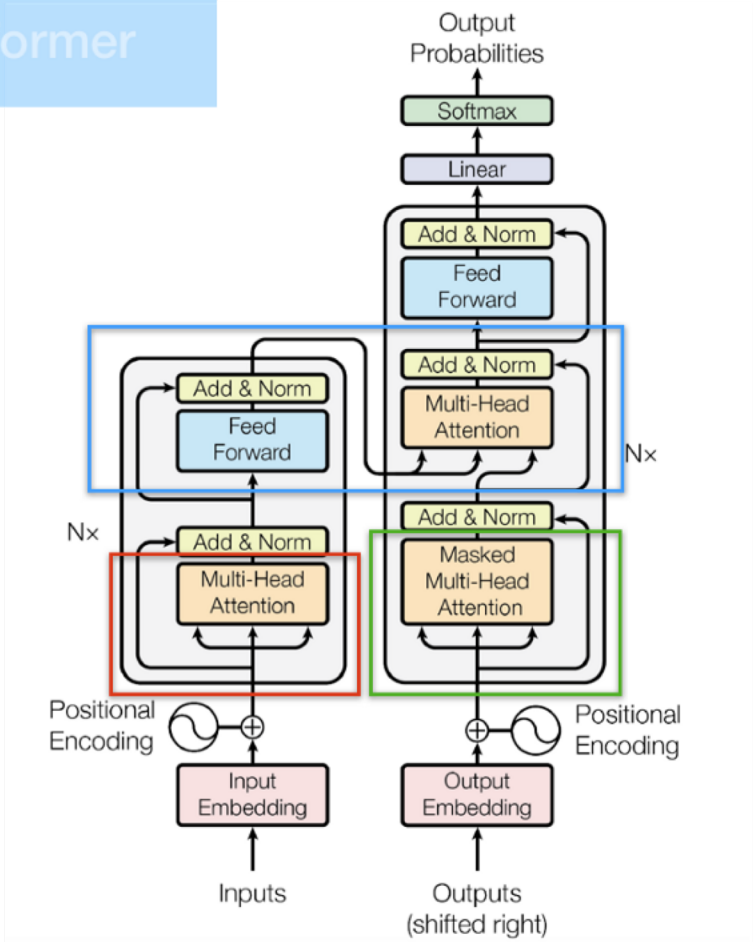


Figure 1: The Transformer - model architecture.

encoder self attention

1. Multi-head Attention
2. Query=Key=Value

decoder self attention

1. Masked Multi-head Attention
2. Query=Key=Value

encoder-decoder attention

1. Multi-head Attention
2. Encoder Self attention=Key=Value
3. Decoder Self attention=Query

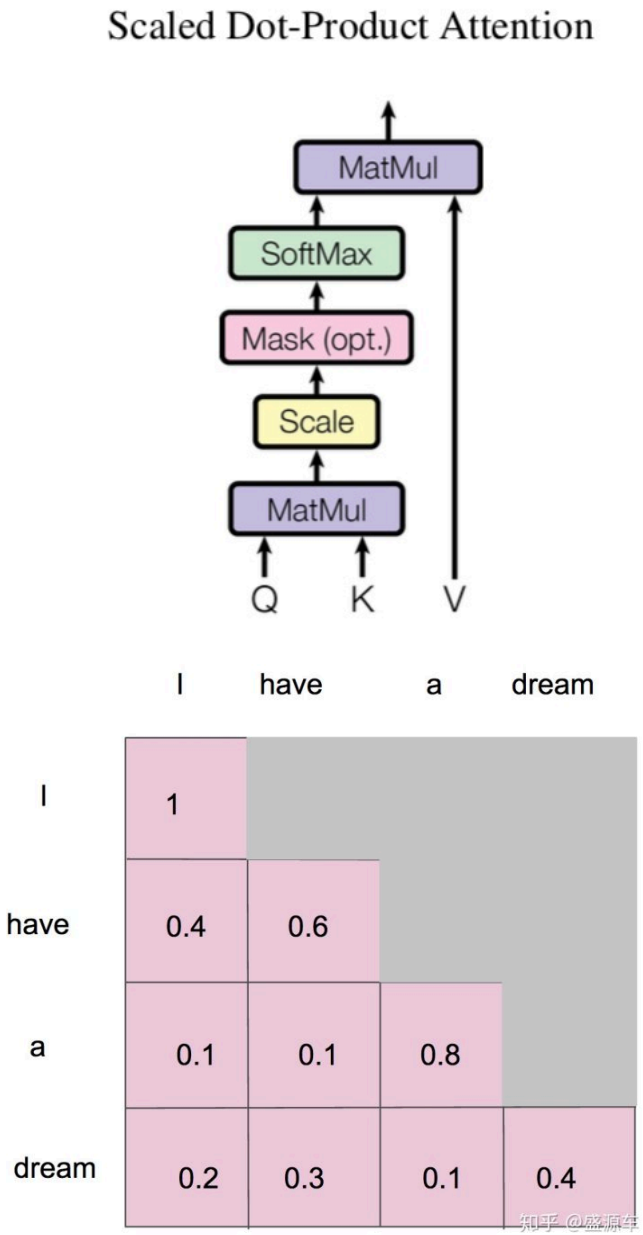
3.2 Attention

Masked multi-head attention

与encoder中的self attention不同的地方是，decoder中用的是**Masked** multi-head attention 。

- 目的：为了避免在训练时，decoder还在翻译前半段时，突然翻译到后半段的句子。
- 方法：在计算softmax前先mask掉未來的位置(設定成-∞)。

这个步骤确保在预测位置i时，只能根据i之前位置的输出。这个是根据encoder-decoder的特性而做出的措施，因为encoder-decode可以看到整个句子。



3.3 Residual Connections & Layer normalization

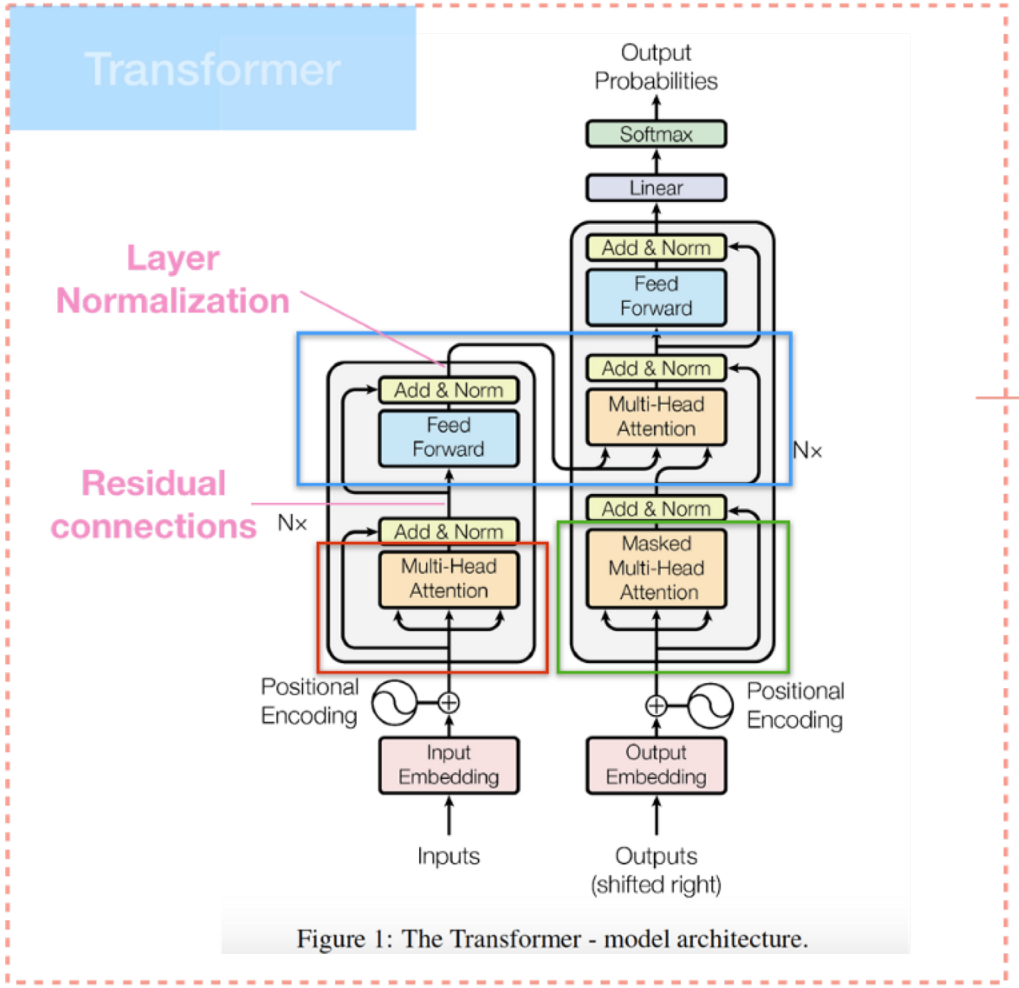
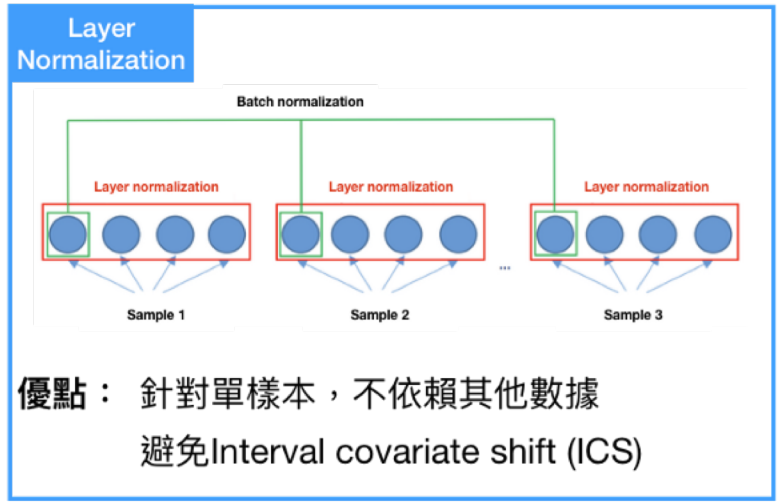
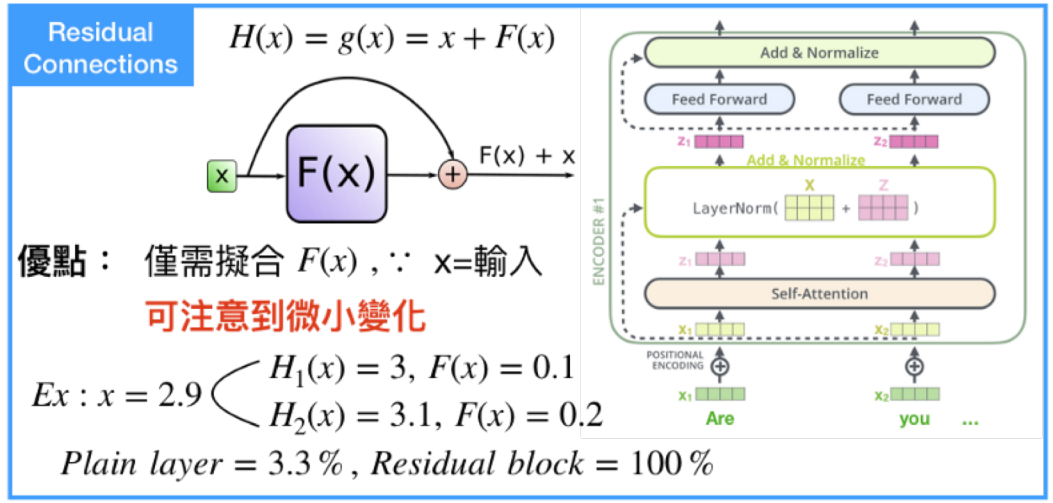


Figure 1: The Transformer - model architecture.





Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun; The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778

Jimmy Lei Ba, Jamie Ryan Kiros, Geoffrey E. Hinton; 2016,Layer Normalization,arXiv:1607.06450

3.3 Residual Connections & Layer normalization

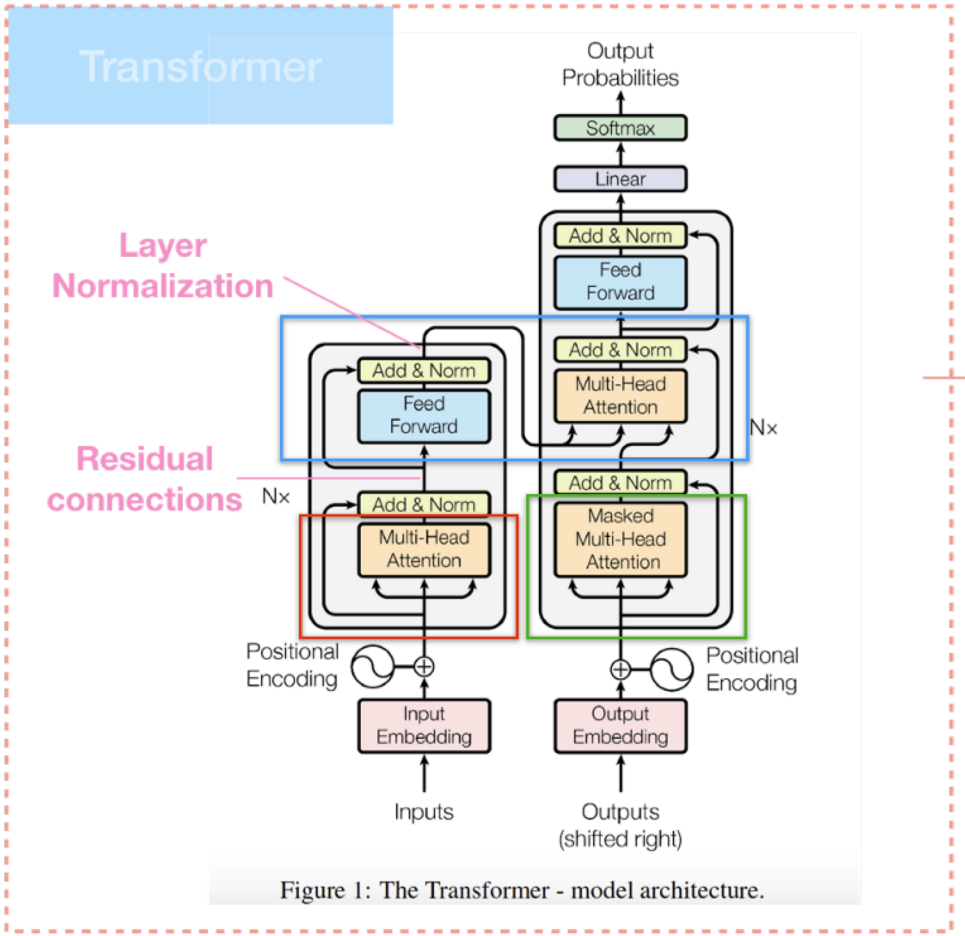
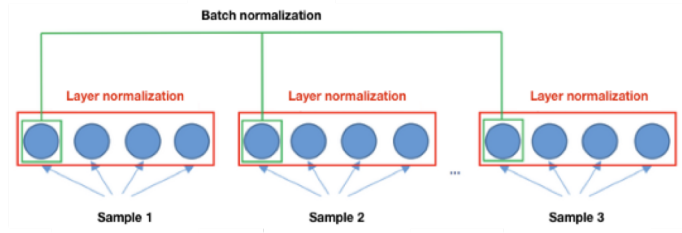


Figure 1: The Transformer - model architecture.

Layer Normalization

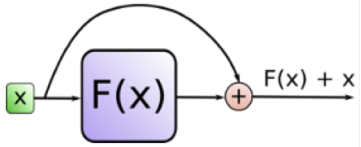


優點： 針對單樣本，不依賴其他數據

避免Interval covariate shift (ICS)

Residual Connections

$$H(x) = g(x) = x + F(x)$$

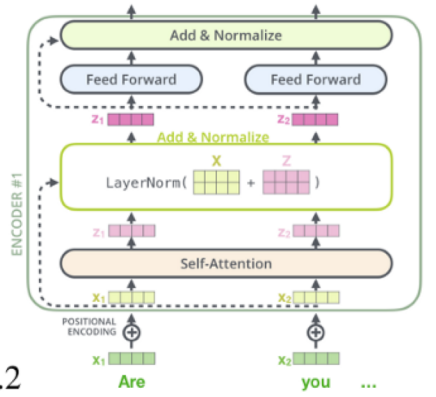


優點： 僅需擬合 $F(x)$, ∵ x=輸入

可注意到微小變化

$$Ex: x = 2.9 \begin{cases} H_1(x) = 3,\ F(x) = 0.1 \\ H_2(x) = 3.1,\ F(x) = 0.2 \end{cases}$$

$$Plain\ layer = 3.3\,\%\ ,\ Residual\ block = 100\,\%$$



The output of each sub-layer is LayerNorm(x + Sublayer(x)), where Sublayer(x) is the function implemented by the sub-layer itself.

3.4 Position-wise Feed-Forward Networks

Transformer

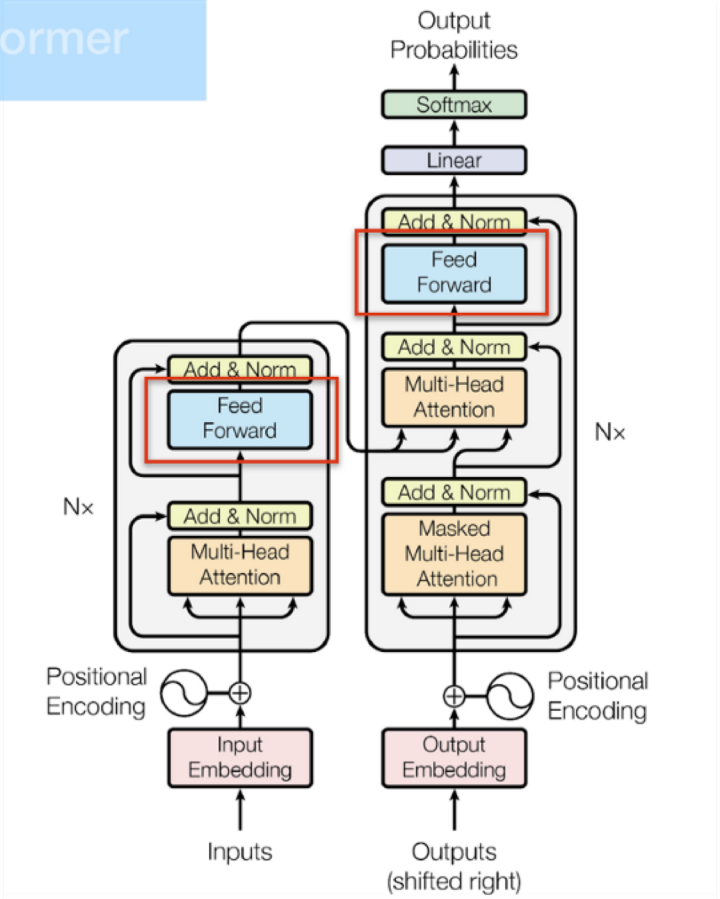


Figure 1: The Transformer - model architecture.

公式

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

Two linear Transformations + Relu

= Two convolution with kernel size 1 $(512 \rightarrow 2048 \rightarrow 512)$

解釋

1. 每個位置獨立進行FFN ⟶ position-wise
2. kernel size =1 ⟶ 每個位置獨立運算

   (if kernel size ≠ 1, then position dependency)
3. 為什麼 $(512 \rightarrow 2048 \rightarrow 512)$?

減少計算量 (可參考Kaiming He. Resnet bottleneck)

3.5 The Final Linear and Softmax Layer

Linear + Softmax

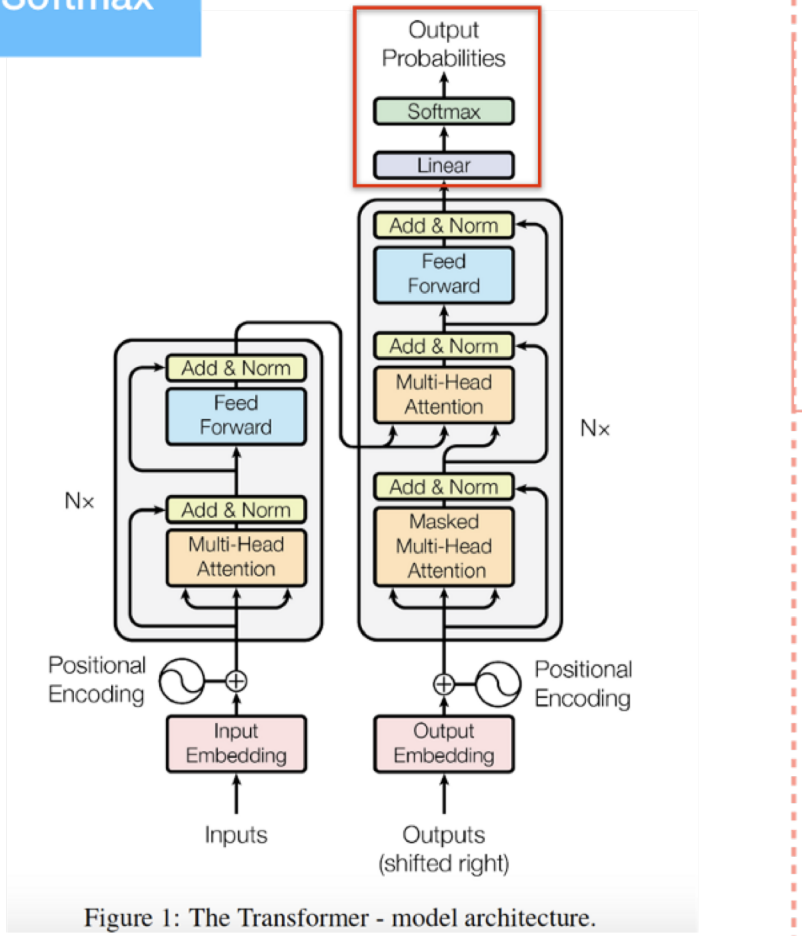
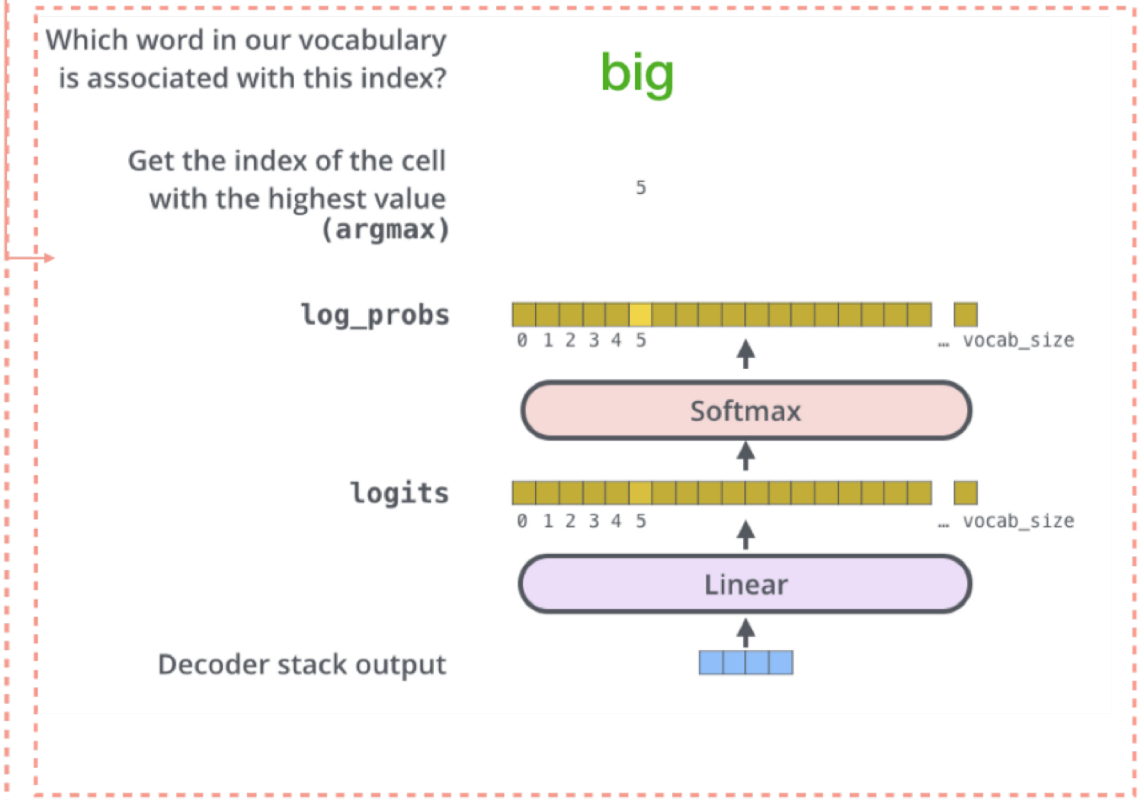
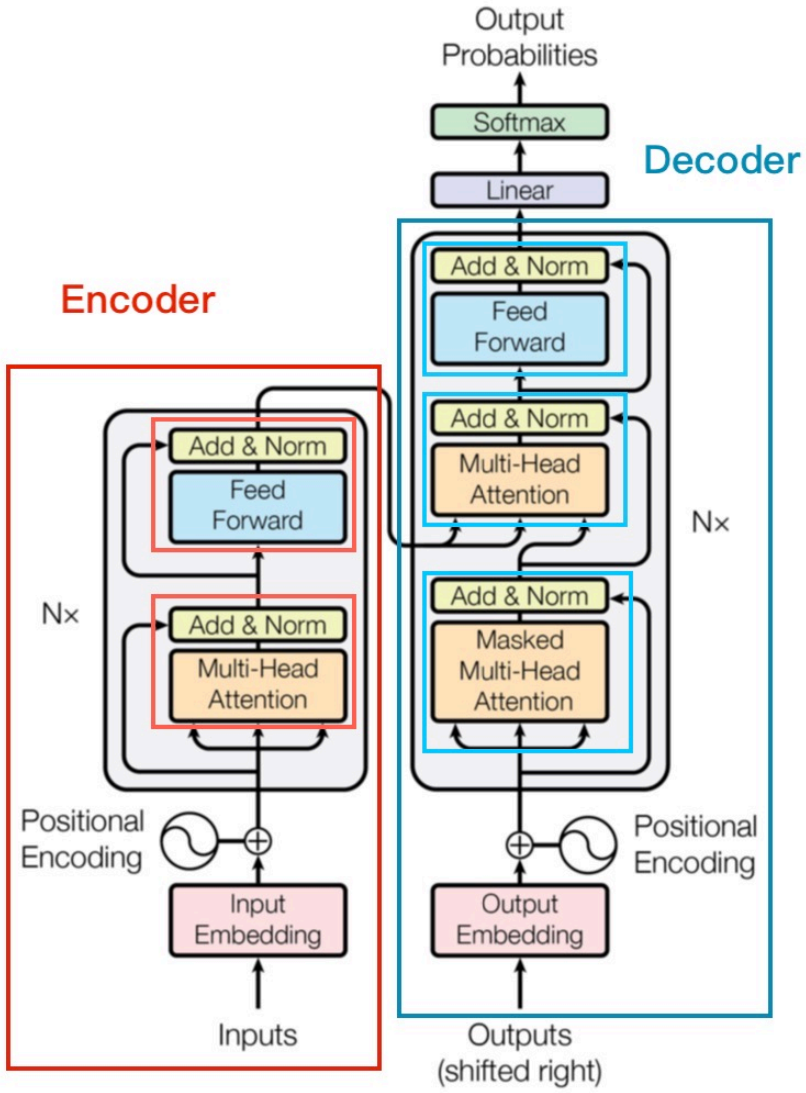
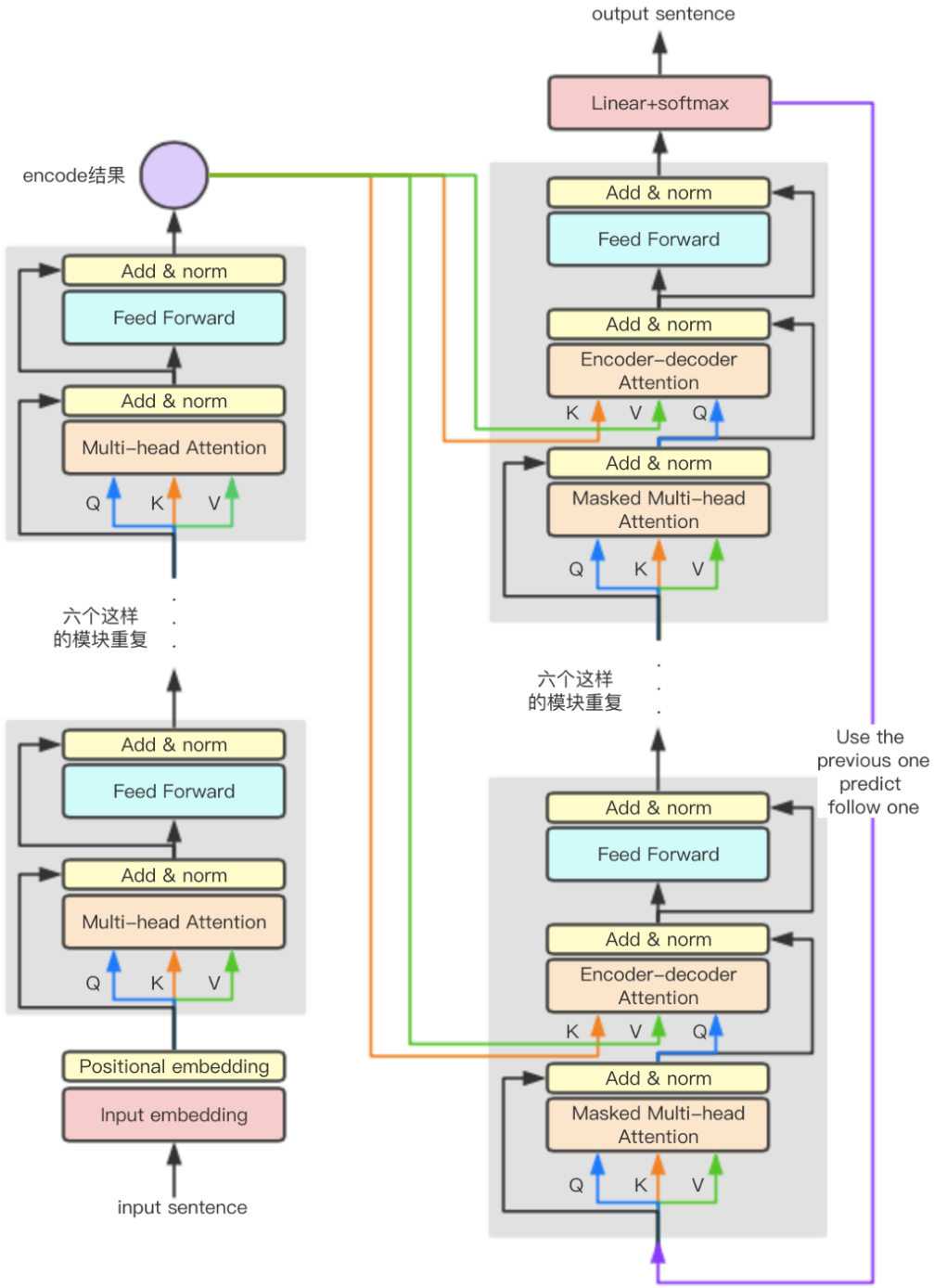


Figure 1: The Transformer - model architecture.

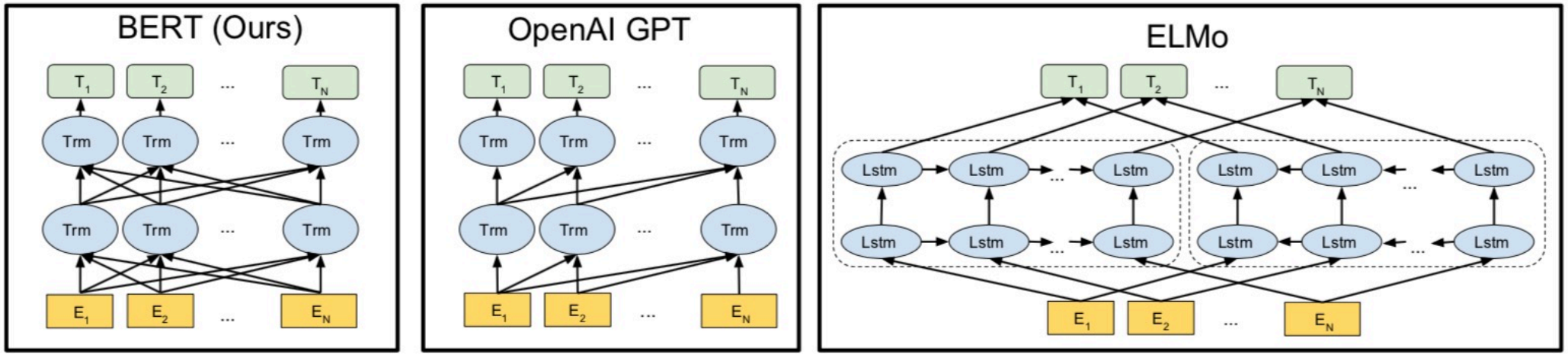


3.6 Model Architecture





4 BERT



L：the number of layers (i.e., Transformer blocks)

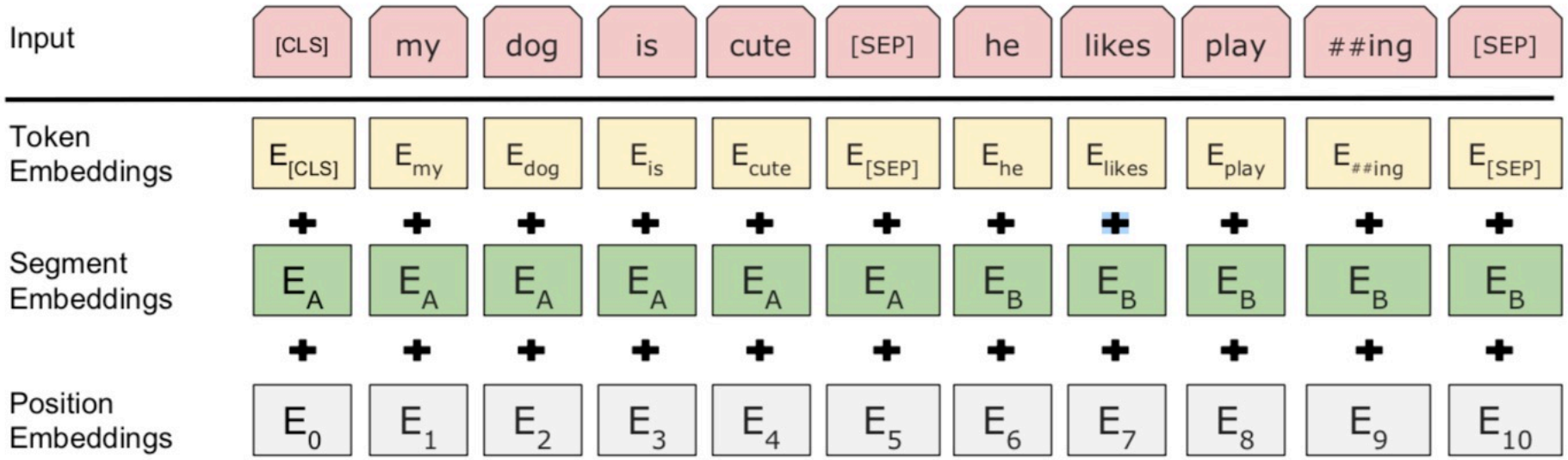
H： the hidden size

A：the number of self-attention heads

- BERTBASE: L=12, H=768, A=12, feed-forward/filter size= 3072, Total Parameters=110M
- BERTLARGE: L=24, H=1024, A=16, feed-forward/filter size= 4096, Total Parameters=340M

4 BERT

Input Representation



4 BERT

Pre-training Task

用两个无监督任务对网络进行预训练

- Masked LM：随机地屏蔽了每个序列中15%的词
- Next Sentence Prediction

Fine-tuning Procedure

Reference

BERT：

Attention Is All You Need

Semi-supervised sequence tagging with bidirectional language models

Deep contextualized word representations（ELMO）

Improving Language Understanding by Generative Pre-Training

Code：

https://github.com/tensorflow/tensor2tensor

tensorflow: https://github.com/Kyubyong/transformer

pytorch: http://nlp.seas.harvard.edu/2018/04/03/attention.html