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

●动机：

● RNN、LSTM 和 GRU 网络已在序列模型、语言建模、机器翻译等应用中取得不错的效果。循环结构的语言模型和编码器 - 解码器体系结构取得了不错的进展。但是，RNN 固有的顺序属性阻碍了训练样本间的并行化，对于长序列，内存限制将阻碍对训练样本的批量处理。

●本文提出的 Transformer，是一种避免循环的模型结构，完全依赖于注意力机制对输入输出的全局依赖关系进行建模。因为对依赖的建模完全依赖于注意力机制，Transformer 使用的注意力机制被称为自注意力(self-attention)。

●模型

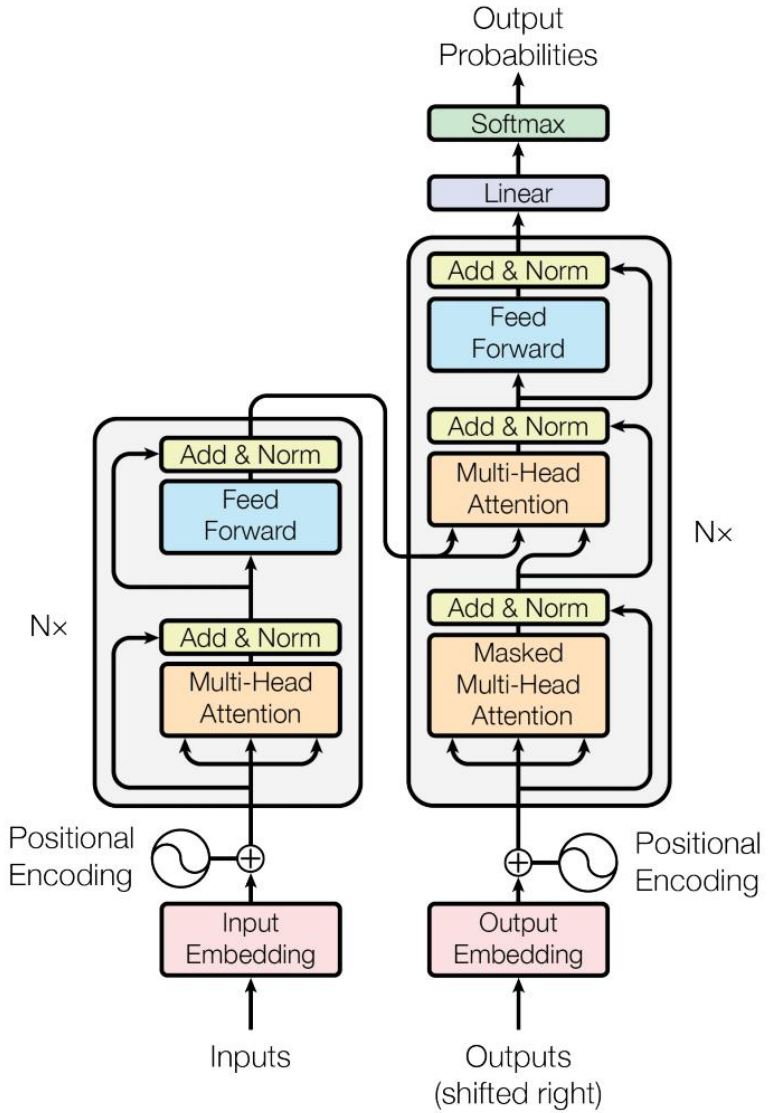
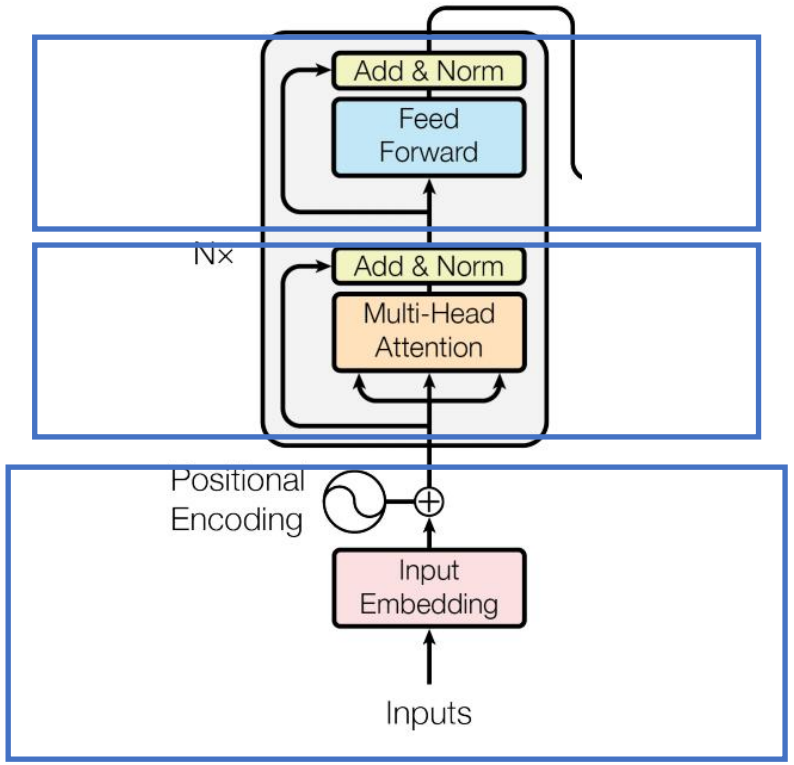


Figure 1: The Transformer - model architecture.

●编码器结构

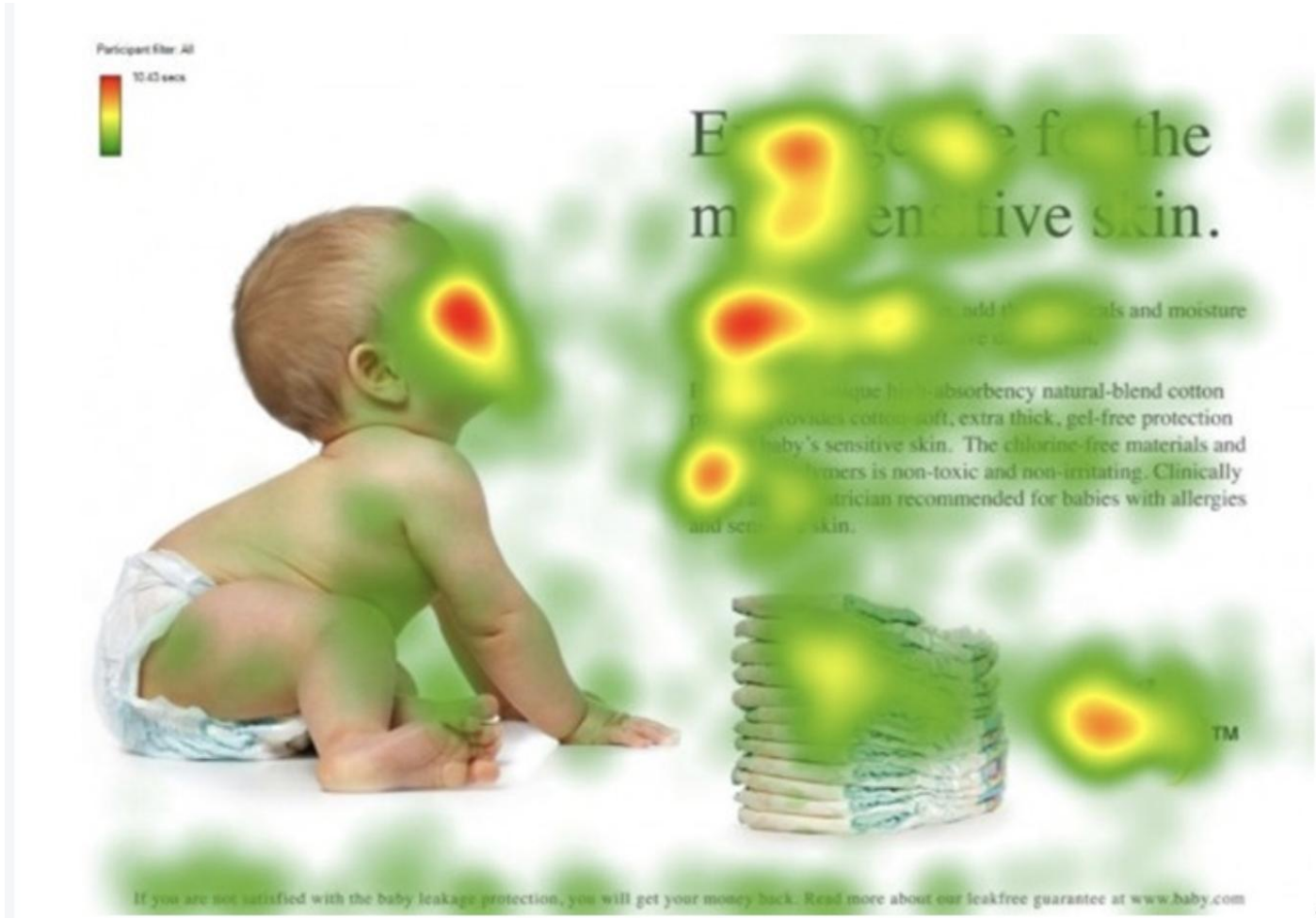


3 前馈神经网络

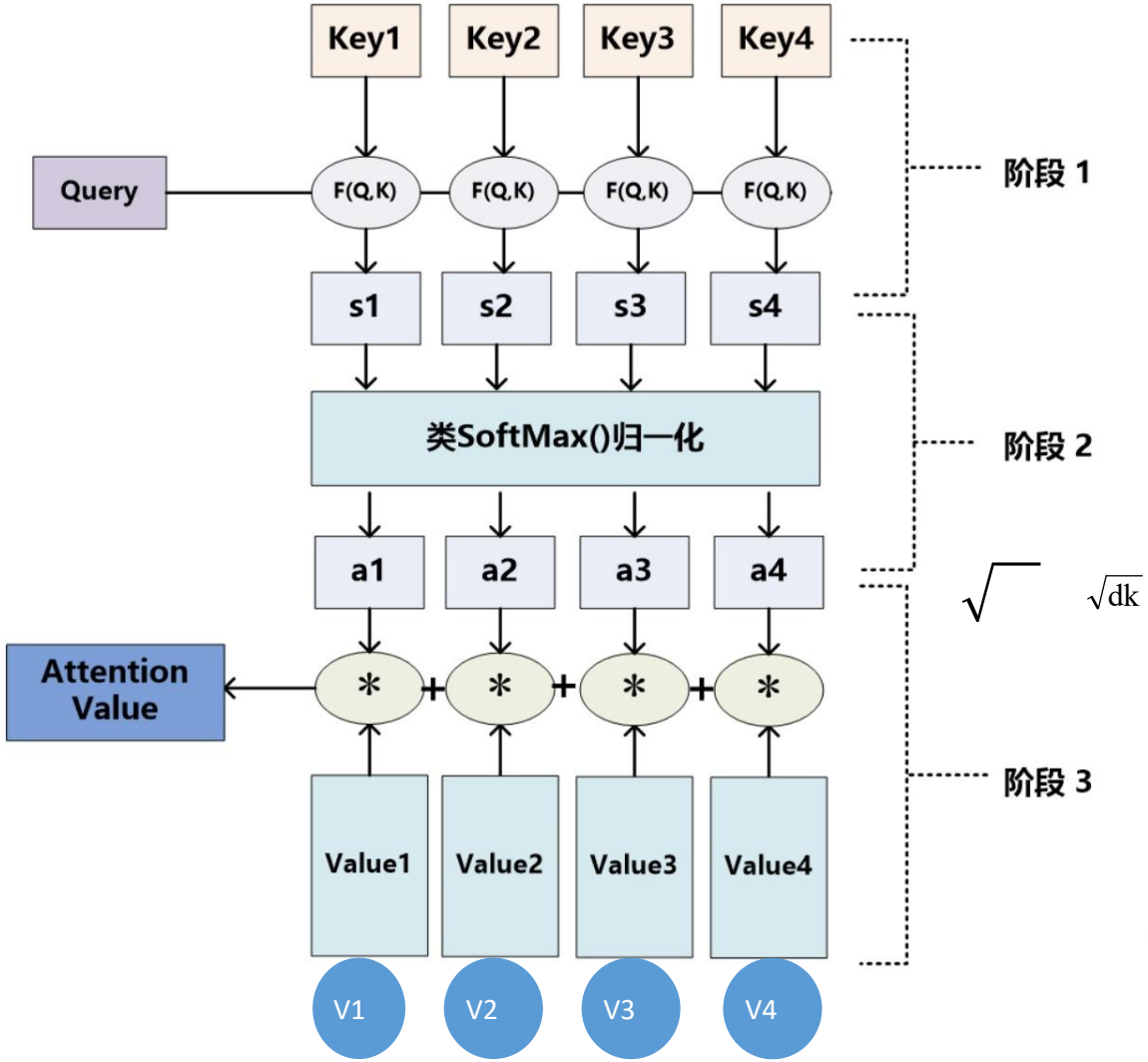
2 注意力机制

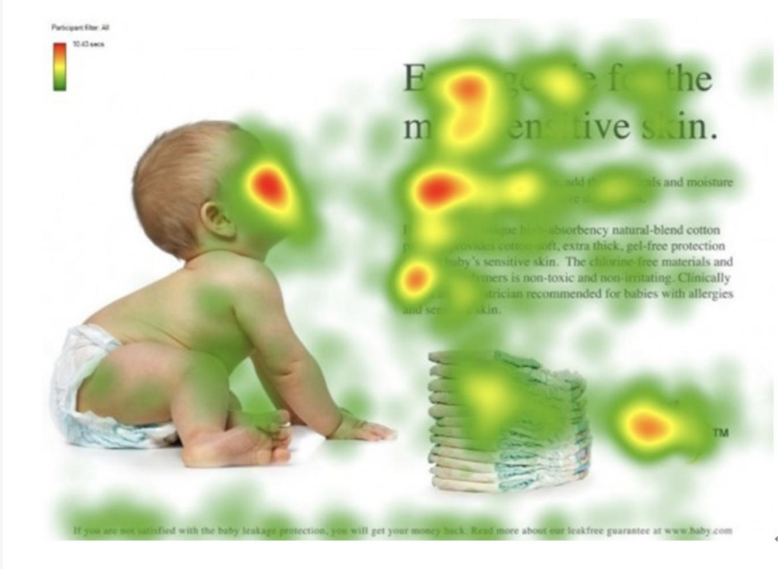
1 输入部分

●注意力机制


Participant Filter: All
10.43 secs
the
and moisture
absorbency natural-blend cotton
extra thick, gel-free protection
baby's sensitive skin. The chlorine-free materials and
is non-toxic and non-irritating. Clinically
recommended for babies with allergies
TM
If you are not satisfied with the baby leakage protection, you will get your money back. Read more about our leakfree guarantee at www.baby.com

●注意力计算公式





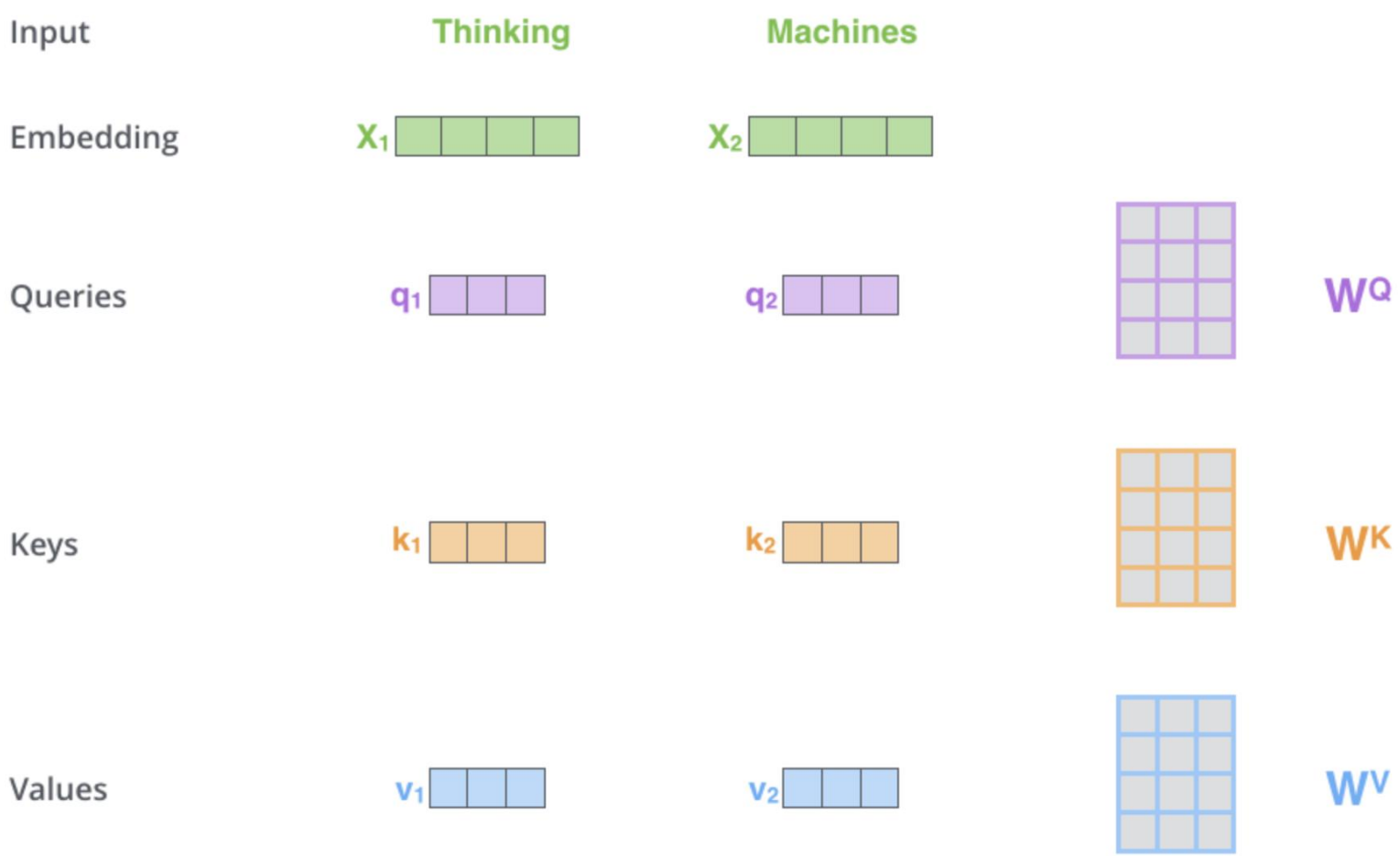
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

为什么除以$\sqrt{d_k}$

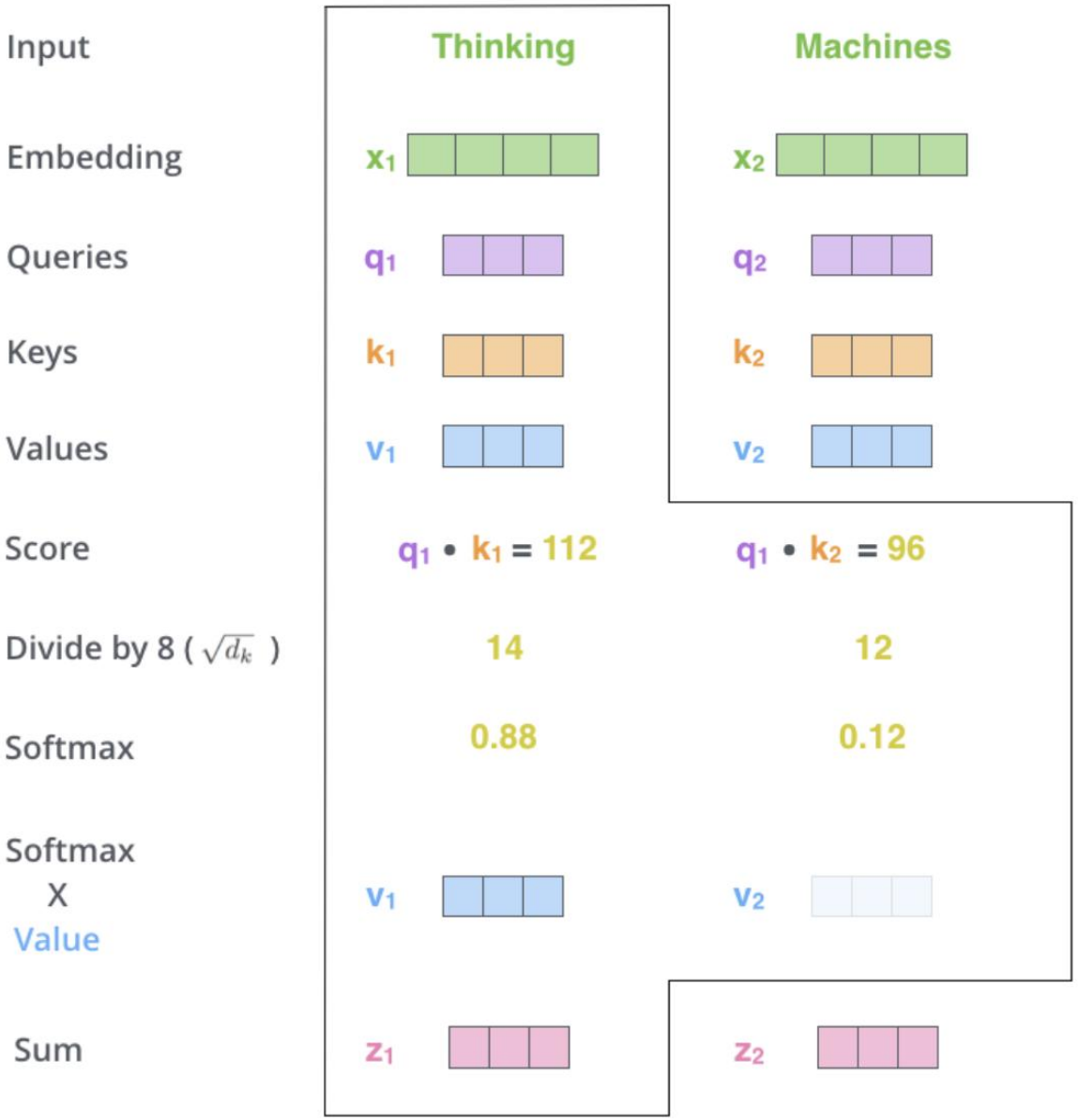
(1)英文介绍

(2)中文介绍

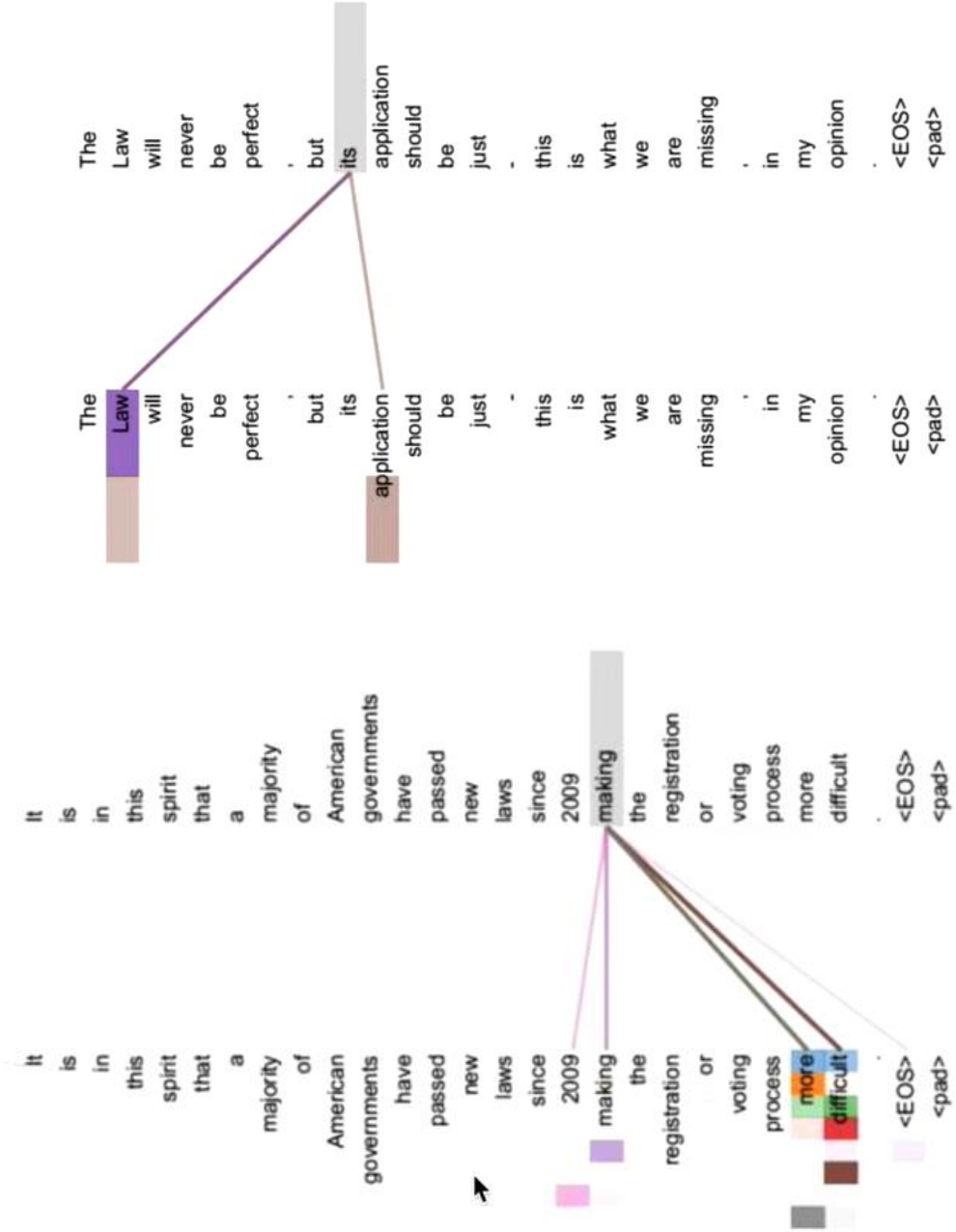
- 自注意力机制


Input
Thinking
Machines
Embedding
X_1
X_2
Queries
q_1
q_2
W^Q
Keys
k_1
k_2
W^K
Values
v_1
v_2
W^V

- 自注意力机制计算


Input
Thinking
Machines
Embedding
x1
x2
Queries
q1
q2
Keys
k1
k2
Values
v1
v2
Score
q1 • k1 = 112
q1 • k2 = 96
Divide by 8 (√dk)
14
12
Softmax
0.88
0.12
Softmax X Value
v1
v2
Sum
z1
z2

●Self-attention得到的词向量具有语法特征和语义特征(表征更完善)



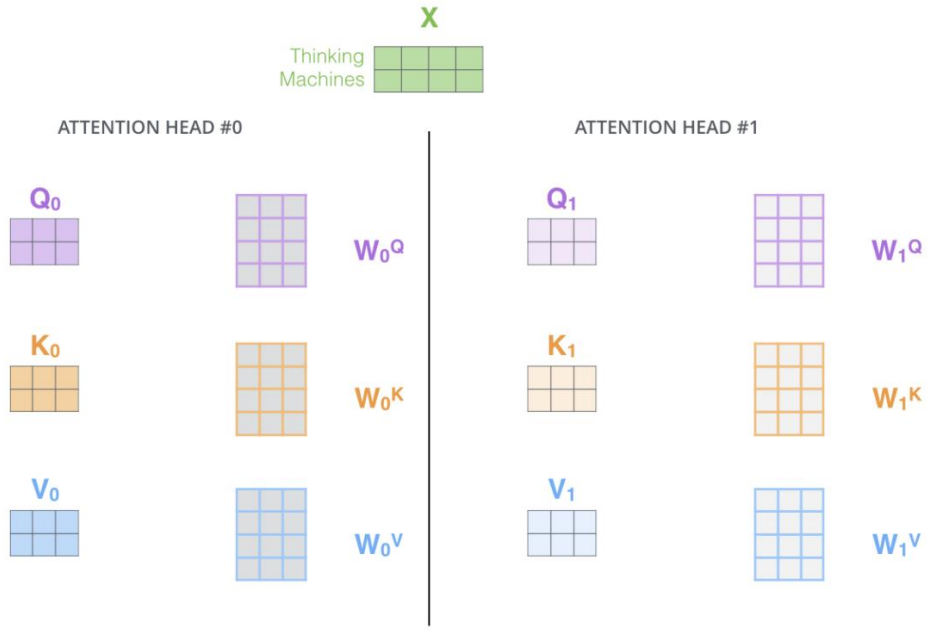
●Attention 与self attention区别

注意力机制是一个很宽泛的概念，QKV相乘就是注意力，但它没有规定QKV是怎么来的。

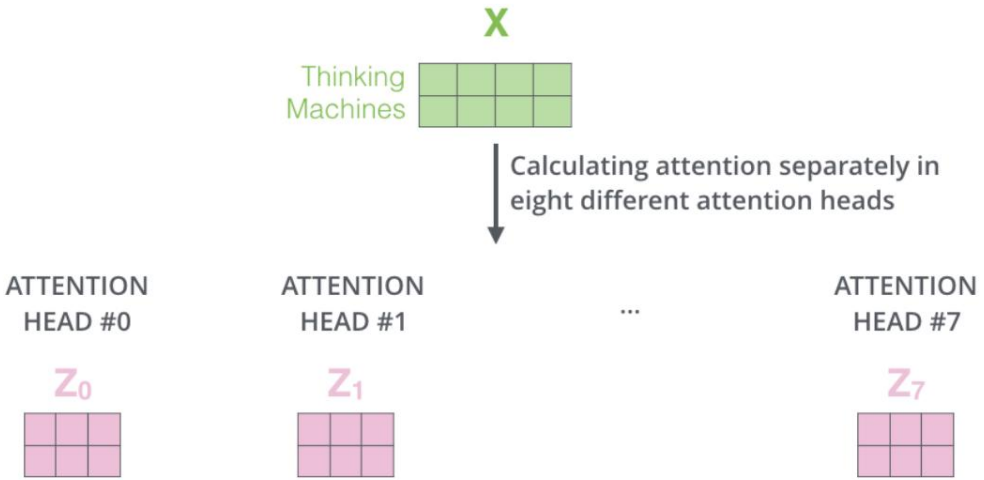
通过一个查询变量Q去找到V里面比较重要的东西。假设K==V，QK相乘得到相似度S，然后SV相乘得到注意力值Z，这个Z就是V的另外一种表现形式，并没有说明QKV怎么来的。

自注意力属于注意力，它要求QKV来源于同一个X。

●多头注意力机制



卷积神经网络比较好的地方在于可以做多个输出通道，每个输出通道可以去识别不一样的模式。多头注意力机制来模拟卷积神经网络多输出通道的结果。



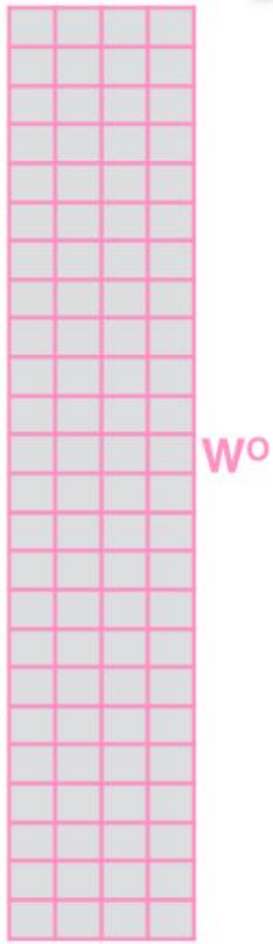
目的：单纯注意力运算可以学到的参数几乎没有，为了识别不一样的模式，希望有不一样的计算相似度的办法。为了增加模型泛化能力，把序列经过线性层投影到低维空间，这个参数W是可以学习的。也就是说，给输入序列H次机会，希望能学到不同投影方法，使得投影到的向量空间里面能够匹配不同视角的相似函数结果(进而不同视角的Attention结果)，类似于卷积神经网络多通道的作用。

● 前馈神经网络

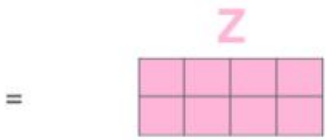
1) Concatenate all the attention heads

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

X

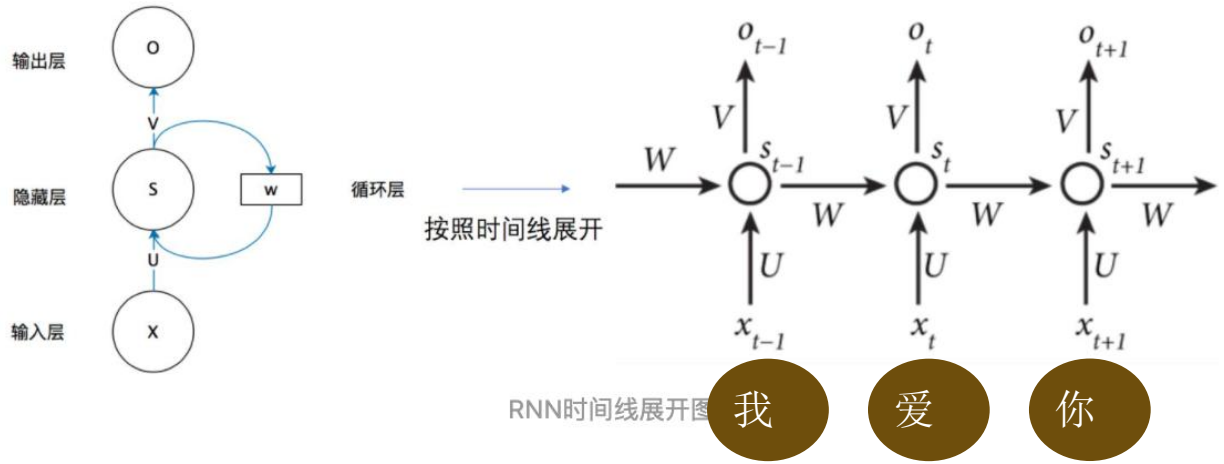


3) The result would be the Z matrix that captures information from all the attention heads. We can send this forward to the FFNN



前馈神经网络没法输入 8 个矩阵。所以我们需要一种方式，把 8 个矩阵降为 1 个，首先，我们把 8 个矩阵连在一起，这样会得到一个大的矩阵，再随机初始化一个矩阵和这个组合好的矩阵相乘，最后得到一个最终的矩阵。

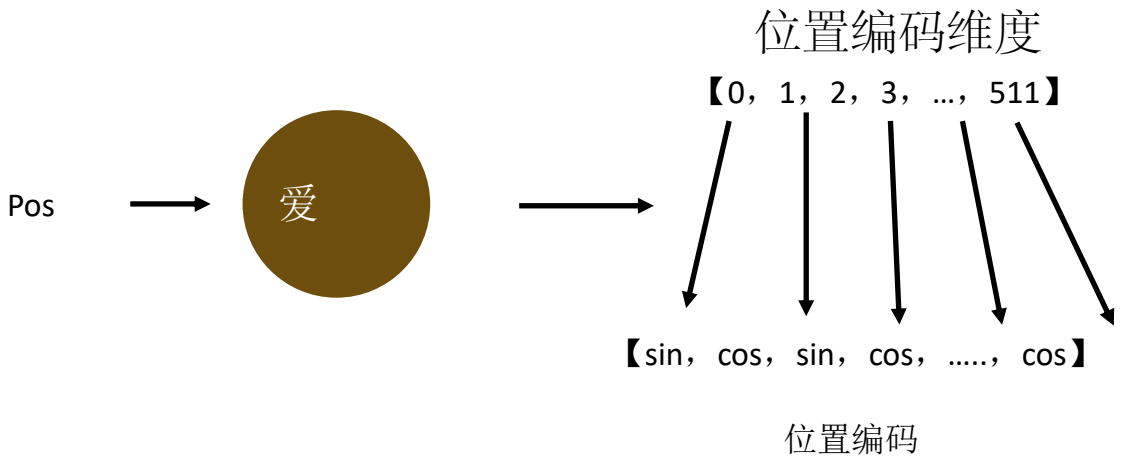
●位置编码



self-attention：对于每个词而言都是没有位置关系的，把每个词的顺序打乱，得到的注意力值依然不变。

●位置编码公式

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

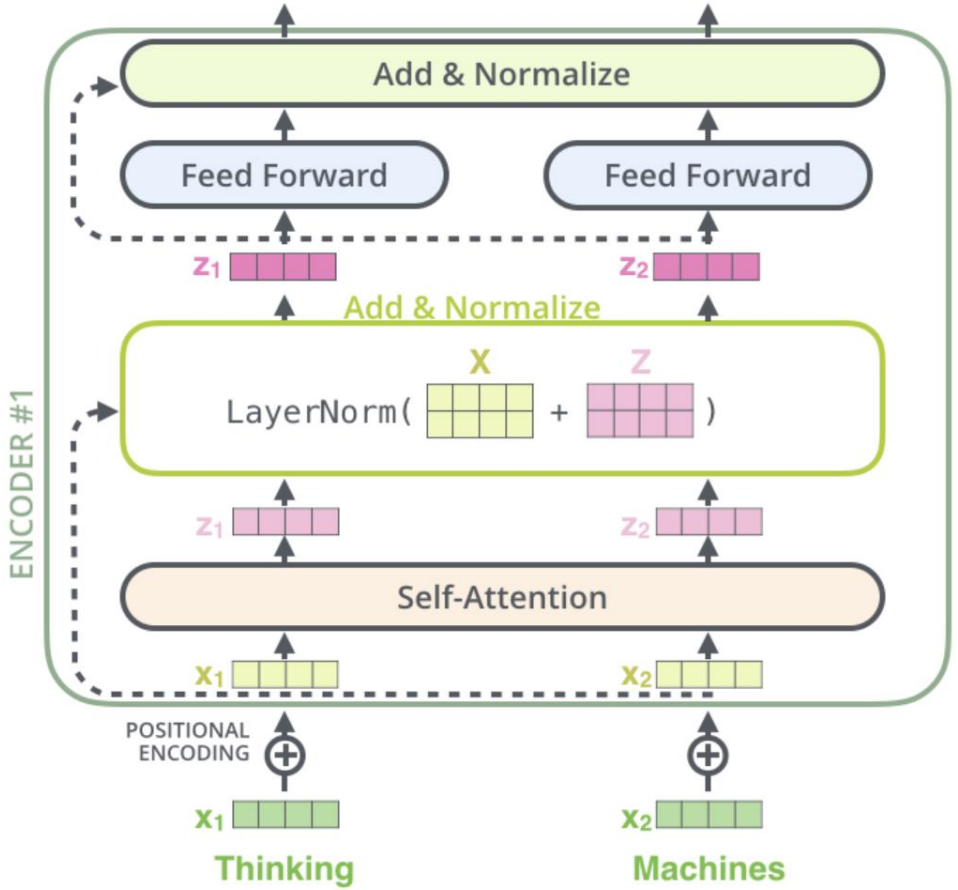


位置编码

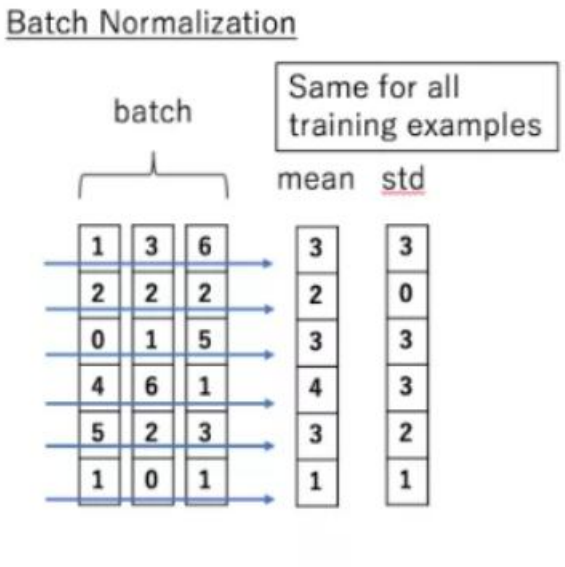
[位置编码追根溯源]

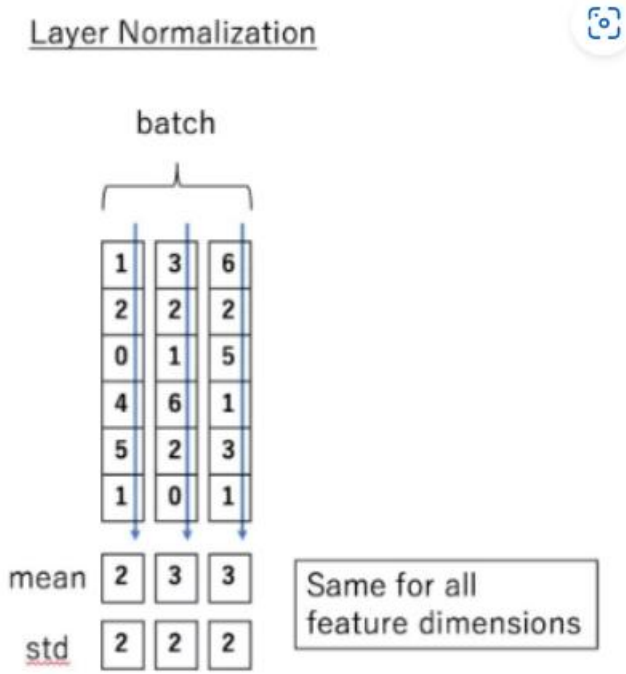
[位置编码介绍]

●残差和LayNorm

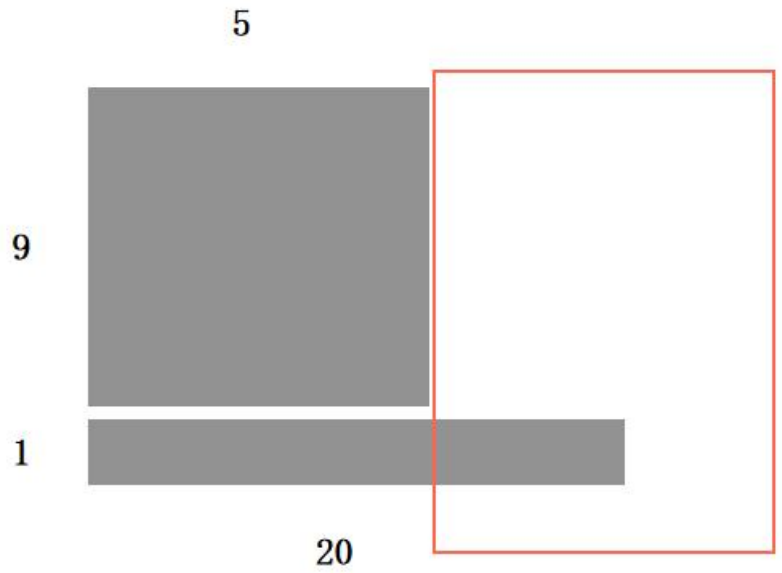

Add & Normalize
Feed Forward
Feed Forward
z1
z2
Add & Normalize
X
Z
LayerNorm(+)
z1
z2
Self-Attention
x1
x2
POSITIONAL ENCODING
x1
x2
Thinking
Machines
ENCODER #1

●标准化


Batch Normalization
batch
Same for all training examples
mean
std
1 3 6
2 2 2
0 1 5
4 6 1
5 2 3
1 0 1
3 2 3 4 3 1
3 0 3 3 2 1


Layer Normalization
batch
1 3 6
2 2 2
0 1 5
4 6 1
5 2 3
1 0 1
mean
2 3 3
std
2 2 2
Same for all feature dimensions

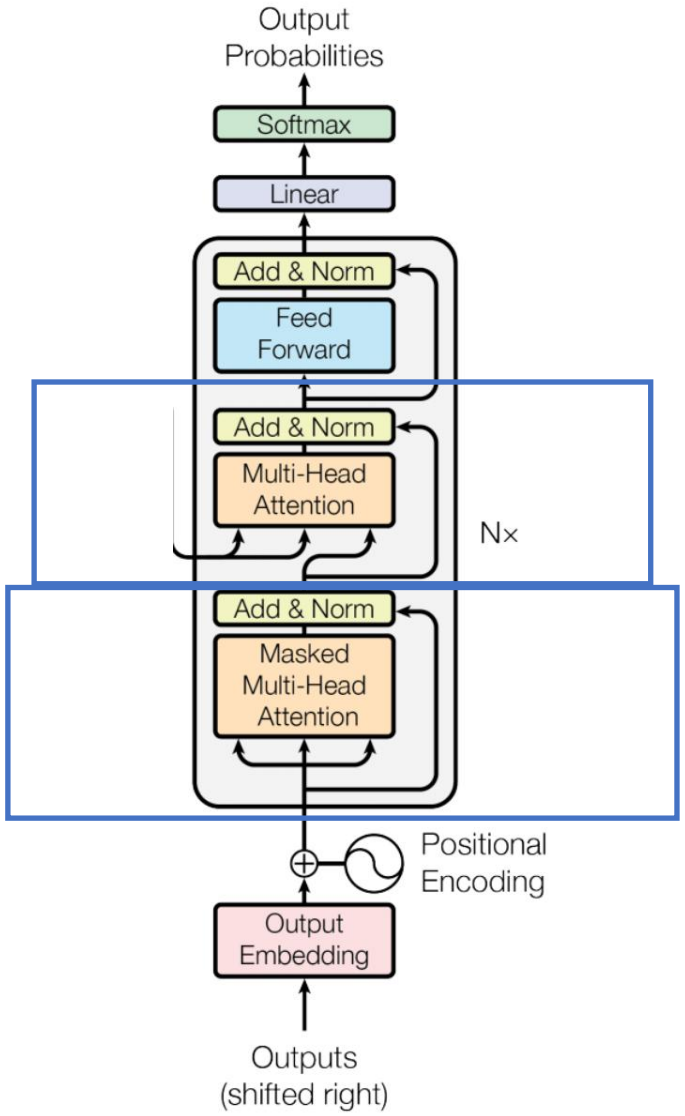
●标准化



我爱中国共产党

今天天气真不错

● 解码器

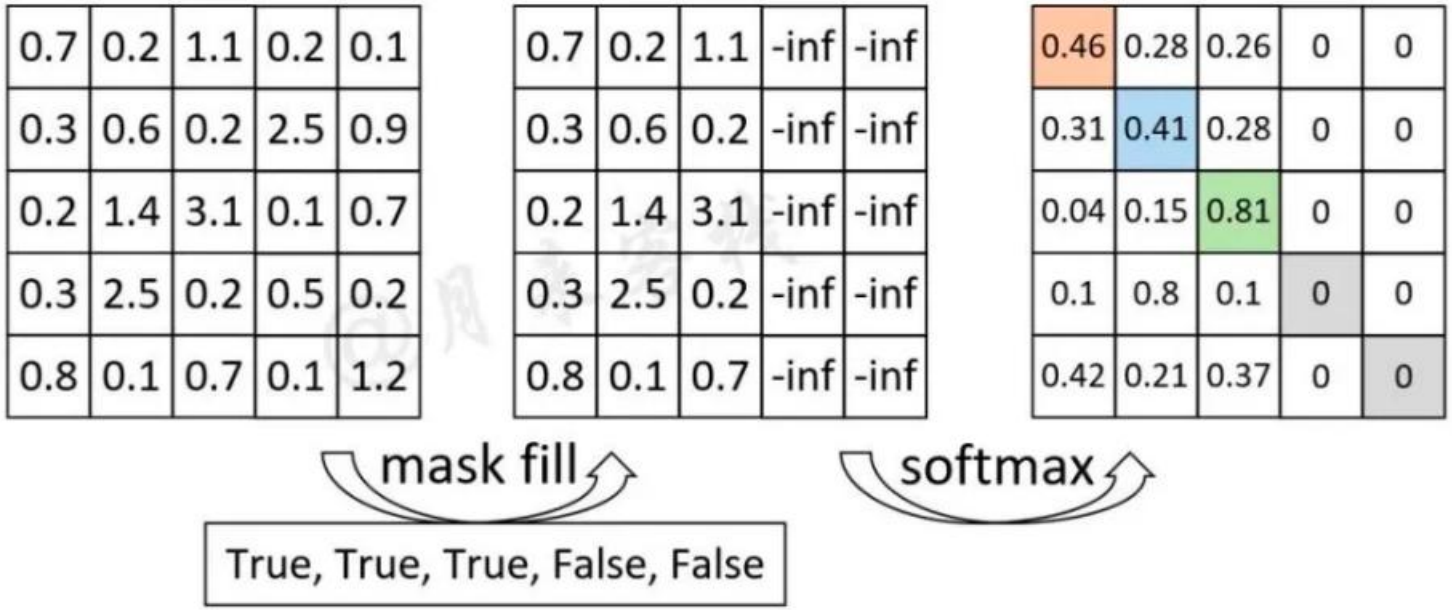

Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
2
1
Positional Encoding
Output Embedding
Outputs (shifted right)

●mask操作

1.padding_mask

由于encoder和decoder两个模块都会有各自相应的输入，但是输入的句子长度是不一样的，计算attention score会出现偏差，为了保证句子的长度一样所以需要进行填充，但是用0填充的位置的信息是完全没有意义的（多余的），经过softmax操作也会有对应的输出，会影响全局概率值，因此我们希望这个位置不参与后期的反向传播过程。以此避免最后影响模型自身的效果，既在训练时将补全的位置给Mask掉，也就是在这些位置上补一些无穷小（负无穷）的值，经过softmax操作，这些值就成了0，就不在影响全局概率的预测。

padding_mask



2.sequence_mask

sequence _mask是只存在decoder的第一个mutil_head_self_attention里，在测试验证阶段，模型并不知道当前时刻的输入和未来时刻的单词信息。也就是对于一个序列中的第i个token解码的时候只能够依靠i时刻之前(包括i)的的输出，而不能依赖于i时刻之后的输出。因此我们要采取一个遮盖的方法(Mask)使得其在计算self-attention的时候只用i个时刻之前的token进行计算。

举例：“我爱中国共产党”，假如要预测“中”这个词，那么当前时刻的输入就是“我”以及“爱”的输入的叠加，一部分来自"我“的信息输出，一部分来自”爱”的信息输出，如果没有mask将后面的单词信息遮住，那么后面的单词对要预测的这个字“中”也会有相应的信息贡献，在训练的时候整个句子的前后字词的位置是已知的，所以不遮挡模型也是可以运行的，因为本身模型输入时就已经知道了句子的整个信息（也就是ground truth embeding）。 但是在进行模型预测（测试新的输入句子）时，输入的句子是未知的，随机的，模型不知道句子的信息，只能通过上一层的输出和原始的输入知道要预测字的前一个信息，进而依次预测后面的字的信息。这就造成了在训练时模型多训练了“中”后面的词，增加了训练时间，消耗了本没必要的空间及时间。在一开始训练时就mask掉，节省时间的同时也降低了过拟合的风险，提高了模型泛化能力。

batch_size	dec_input sen_len				
dec_input sen_len	F	F	F	F	T
	F	F	F	F	T
	F	F	F	F	T
	F	F	F	F	T
	F	F	F	F	T

batch_size	dec_input sen_len				
dec_input sen_len	0	1	1	1	1
		0	1	1	1
			0	1	1
				0	1
					0

batch_size	dec_input sen_len				
dec_input sen_len	0	1	1	1	1
		0	1	1	1
			0	1	1
				0	1
					1

●交互注意力机制

在Transformer的decoder中，K和V均是编码部分的输出Memory经过线性变换后的结果（此时的Memory中包含了原始输入序列每个位置的编码信息），而Q是解码部分多头注意力机制输出的隐含向量经过线性变换后的结果。在Decoder对每一个时刻进行解码时，首先需要做的便是通过Q与 K进行交互（query查询），并计算得到注意力权重矩阵；然后再通过注意力权重与V进行计算得到一个权重向量，该权重向量所表示的含义就是在解码时如何将注意力分配到Memory的各个位置上。
交互注意力层就是根据编码器的输出，解码器从中更加关注自己需要的地方。

●实验结果

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [15]	23.75			
Deep-Att + PosUnk [32]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [31]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [8]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [26]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [32]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [31]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [8]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	**$3.3 \cdot 10^{18}$**	
Transformer (big)	**28.4**	**41.0**	$2.3 \cdot 10^{19}$	

●结论

- ●作者提出的Transformer模型仅仅依靠注意力机制来做序列转录任务用多头注意力机制代替循环神经网络。
- ●在机器翻译任务中，Transformer模型比其他模型训练的要快很多。
- ●对于这种纯注意力机制的模型作者认为也可以用在其他方面(图片、语音等)。

B站视频

B站视频

知乎

代码