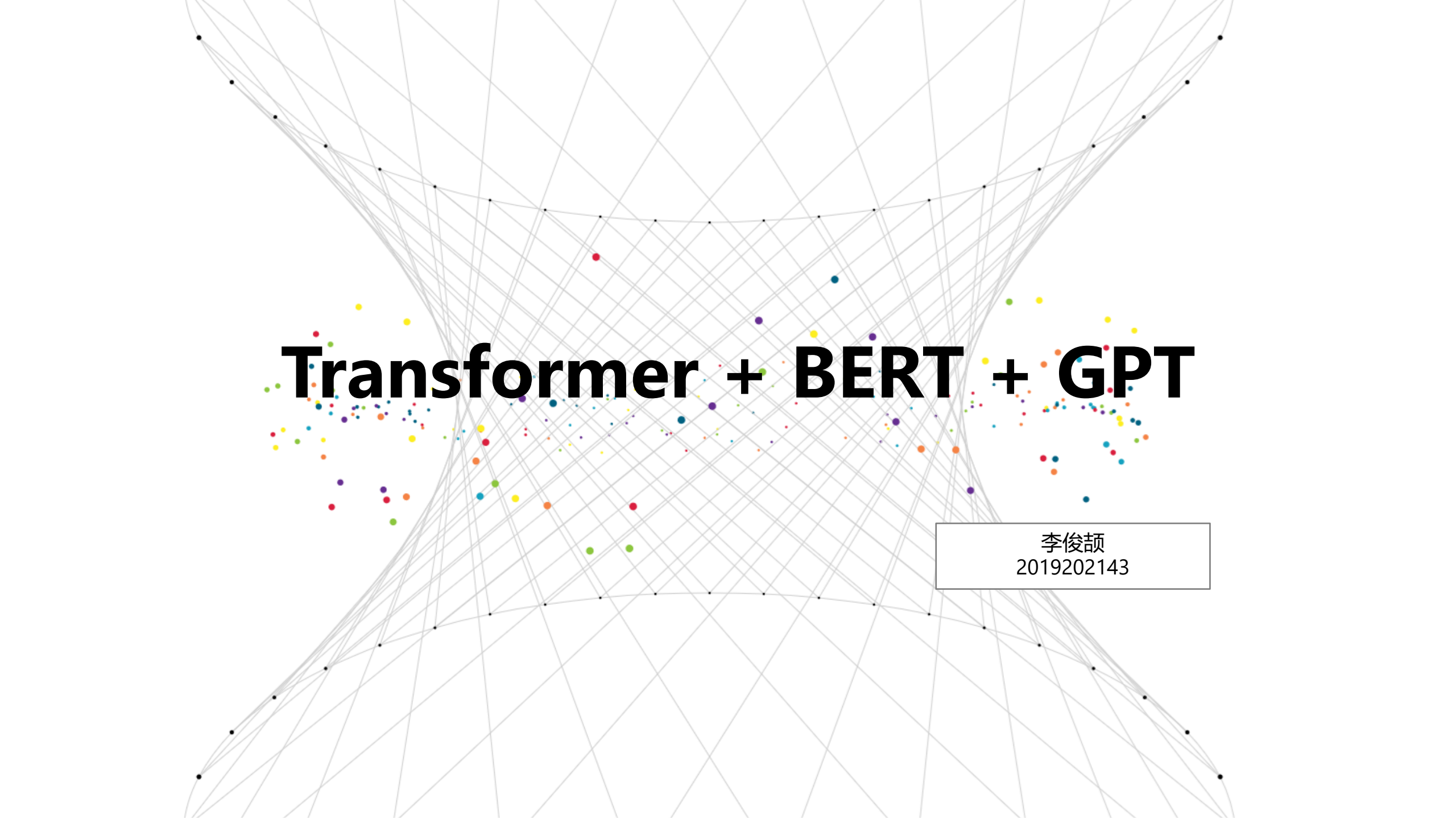
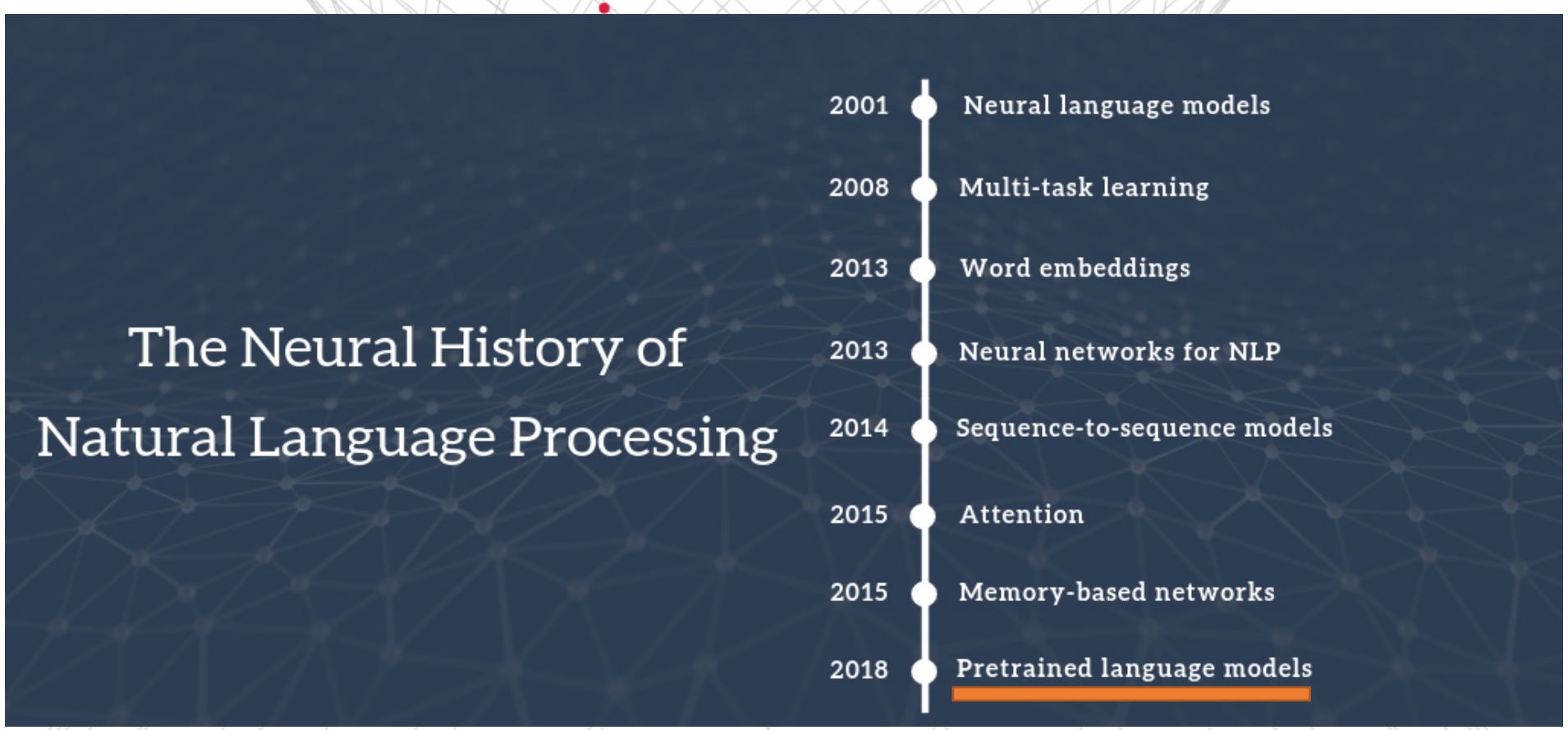


Transformer + BERT + GPT

李俊颉
2019202143

BERT：Pre-training of Deep Bidirectional Transformers for Language Understanding

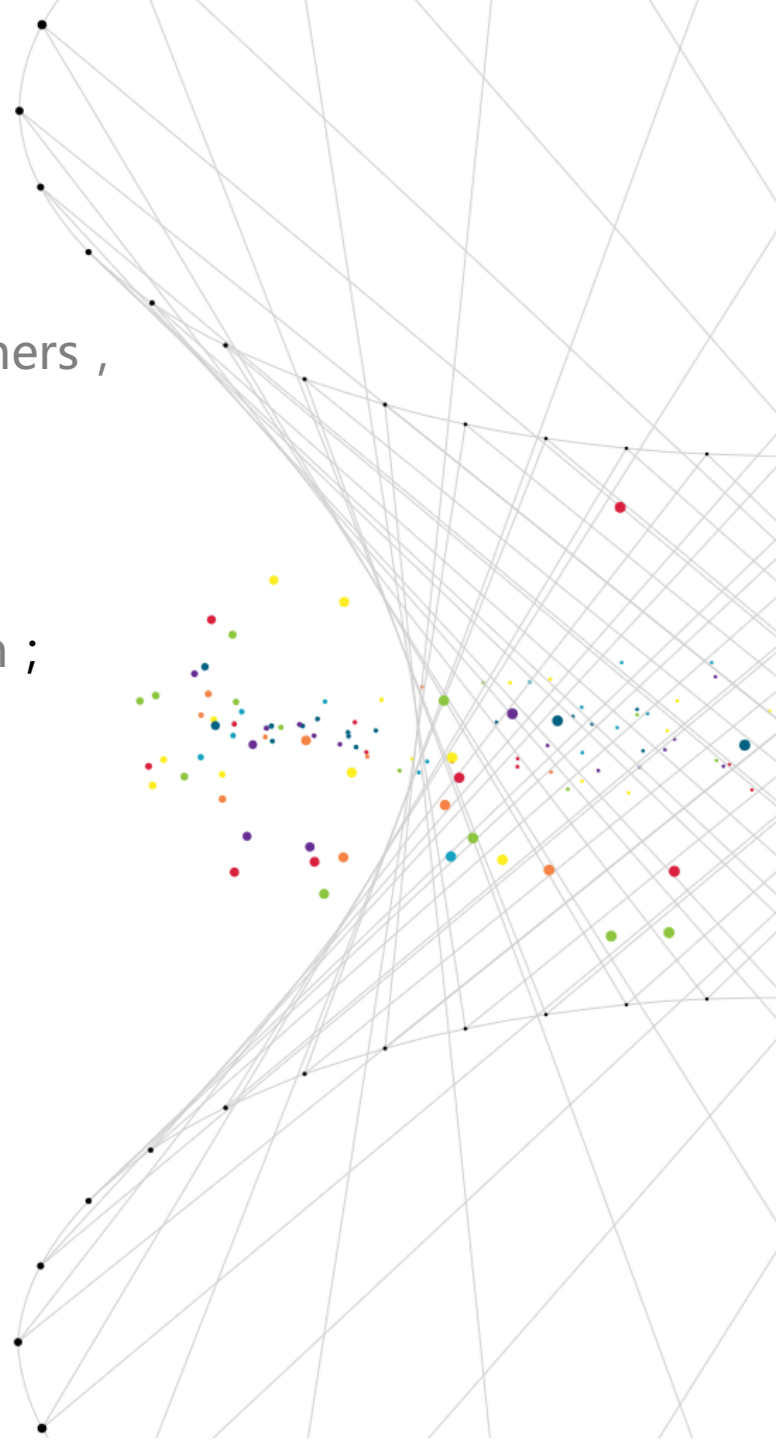


BERT

1.BERT的全称是Bidirectional Encoder Representation from Transformers，即双向Transformer的Encoder；

2.模型的主要创新点都在pre-train方法上，即用了Masked LM和Next Sentence Prediction两种方法分别捕捉词语和句子级别的representation；

3.Learned from a large amount of text without annotation；



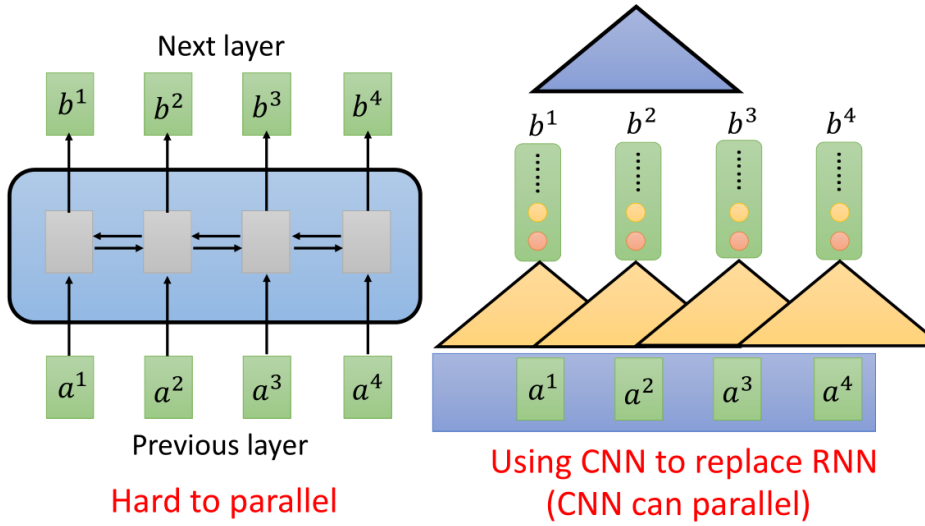
Transformer（Attention Is All You Need）

1.Seq2seq model with “Self-attention”；

2.Transformer由且仅由self-Attenion和Feed Forward Neural Network组成；

[1] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need [C]//Advances in Neural Information Processing Systems. 2017: 5998-6008.
[2]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

Transformer （Attention Is All You Need）



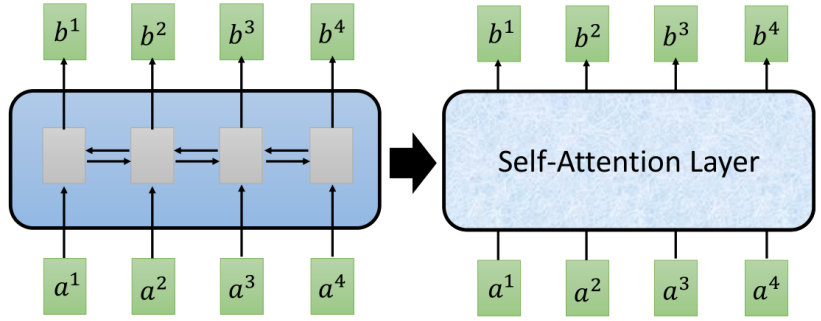
为什么采用Attention机制 ：

RNN机制问题：

- 时间片 t 的计算依赖 t-1 时刻的计算结果，这样限制了模型的并行能力；
- 顺序计算的过程中信息会丢失，尽管LSTM等门机制的结构一定程度上缓解了长期依赖的问题，但是对于特别长期的依赖现象,LSTM依旧无能为力。

Transformer如何解决这两个问题？

- 它使用了Attention机制，将序列中的任意两个位置之间的距离是缩小为一个常量；
- 它不是类似RNN的顺序结构，因此具有更好的并行性，符合现有的GPU框架。

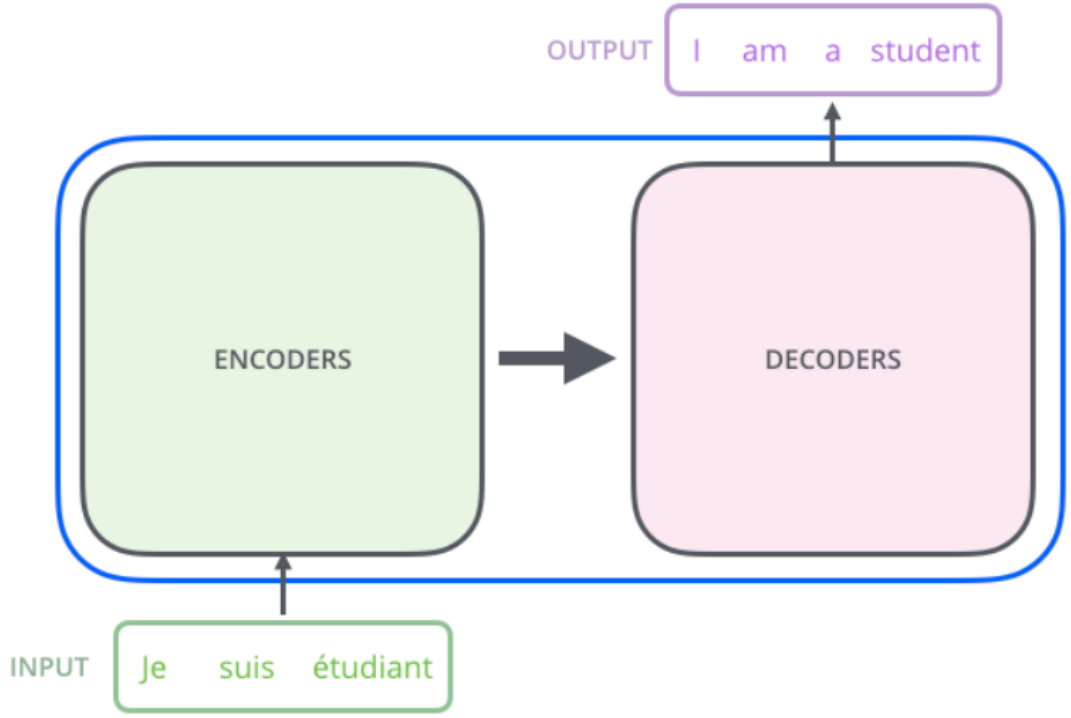


[1] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need [C]//Advances in Neural Information Processing Systems. 2017: 5998-6008.
[2]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

Transformer 详解（机器翻译）

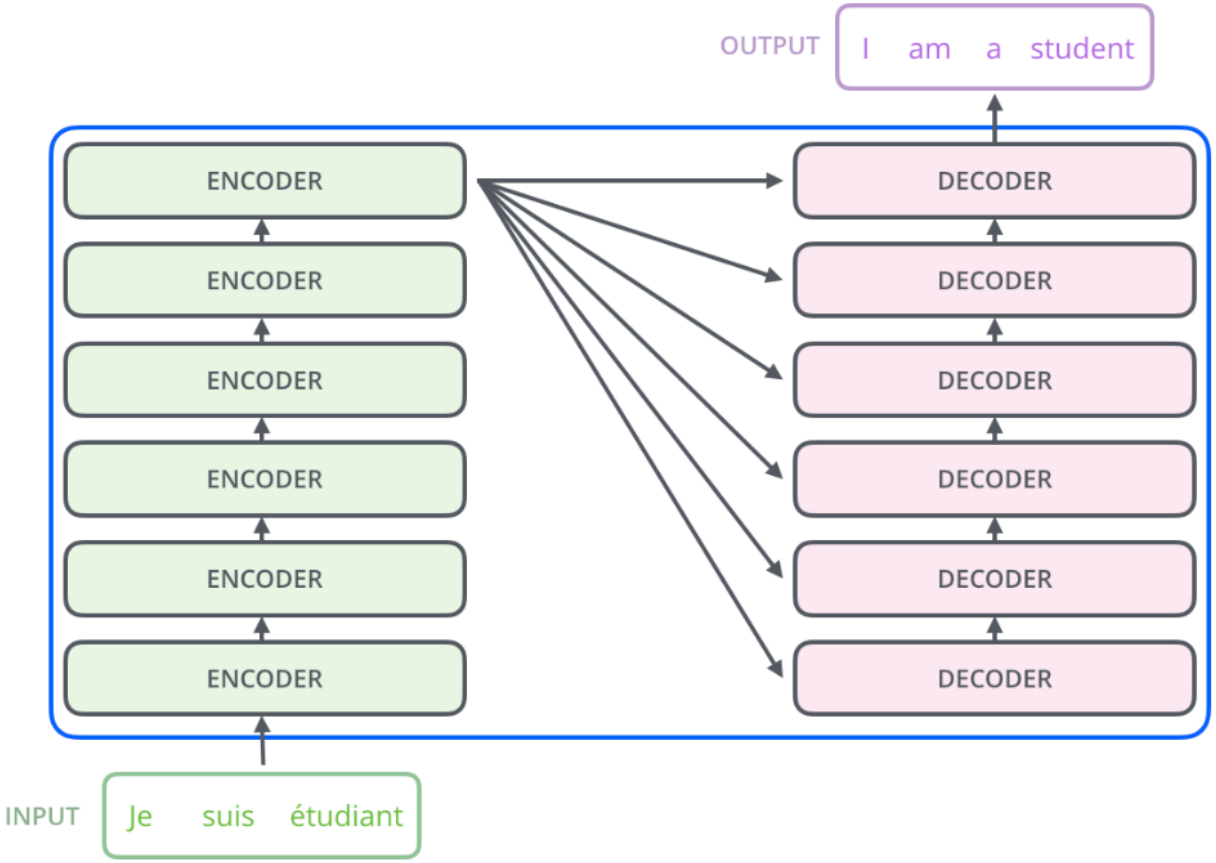
[1]https://jalammar.github.io/illustrated-transformer/

Transformer 详解（机器翻译）



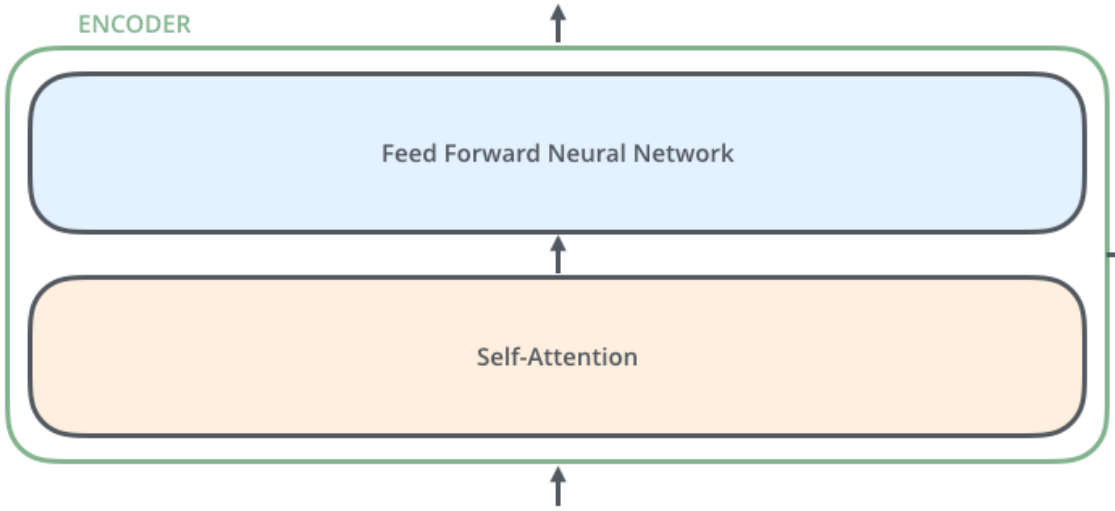
[1]https://jalammar.github.io/illustrated-transformer/

Transformer 详解（机器翻译）



[1]https://jalammar.github.io/illustrated-transformer/

Transformer 详解（机器翻译）



$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{1}$$

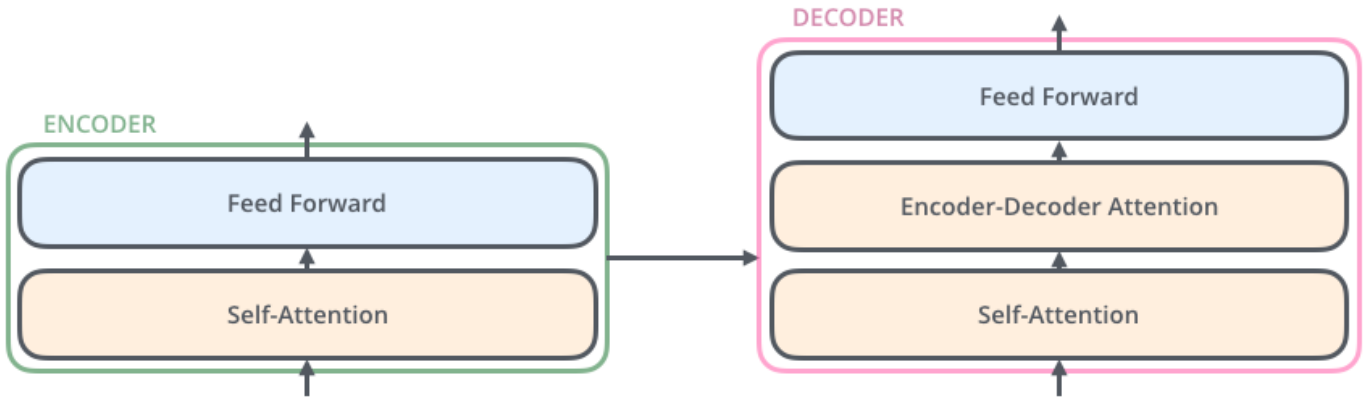
$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{2}$$

[1]https://jalammar.github.io/illustrated-transformer/
[2] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need [C]//Advances in Neural Information Processing Systems. 2017: 5998-6008.

Transformer 详解（机器翻译）

Decoder和Encoder的不同之处在于Decoder多了一个Encoder-Decoder Attention，两个Attention分别用于计算输入和输出的权值：

- Self-Attention：当前翻译和已经翻译的前文之间的关系；
- Encoder-Decnoder Attention：当前翻译和编码的特征向量之间的关系。



[1]https://jalammar.github.io/illustrated-transformer/
[2] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need [C]//Advances in Neural Information Processing Systems. 2017: 5998-6008.

Transformer 输入（机器翻译）

使用词嵌入算法将语料转换为向量进行输入；

[1]https://jalammar.github.io/illustrated-transformer/

Self-attention

https://arxiv.org/abs/1706.03762

q: query (to match others)

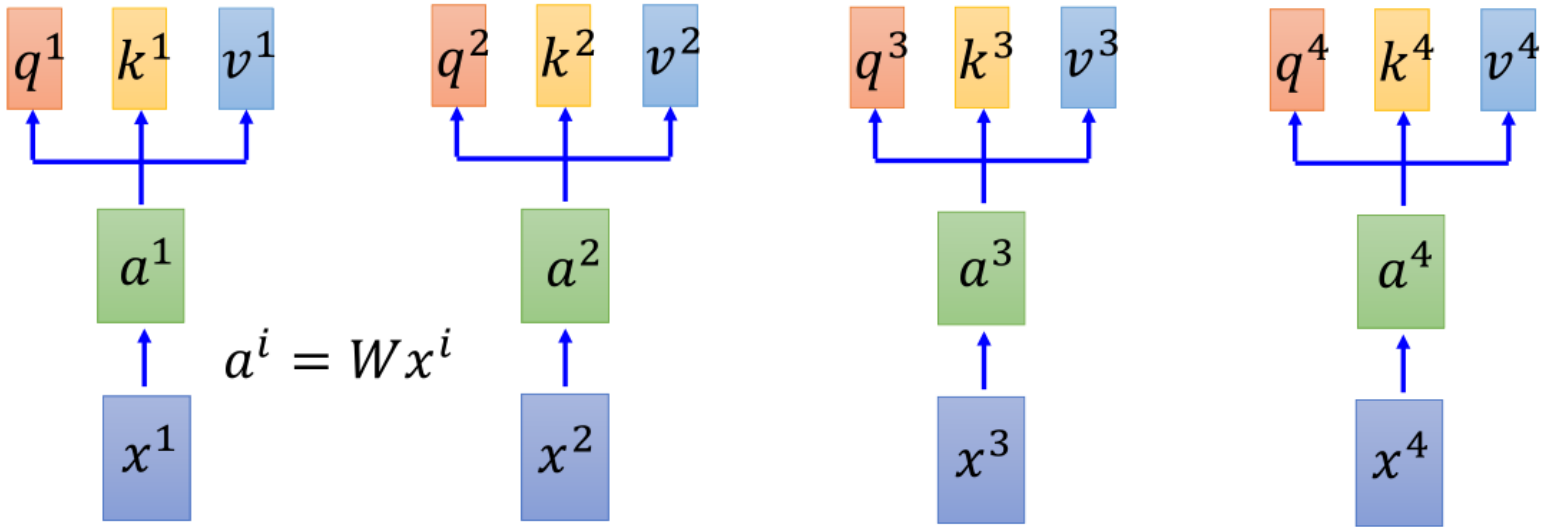
$$q^i = W^q a^i$$

k: key (to be matched)

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

v: information to be extracted

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



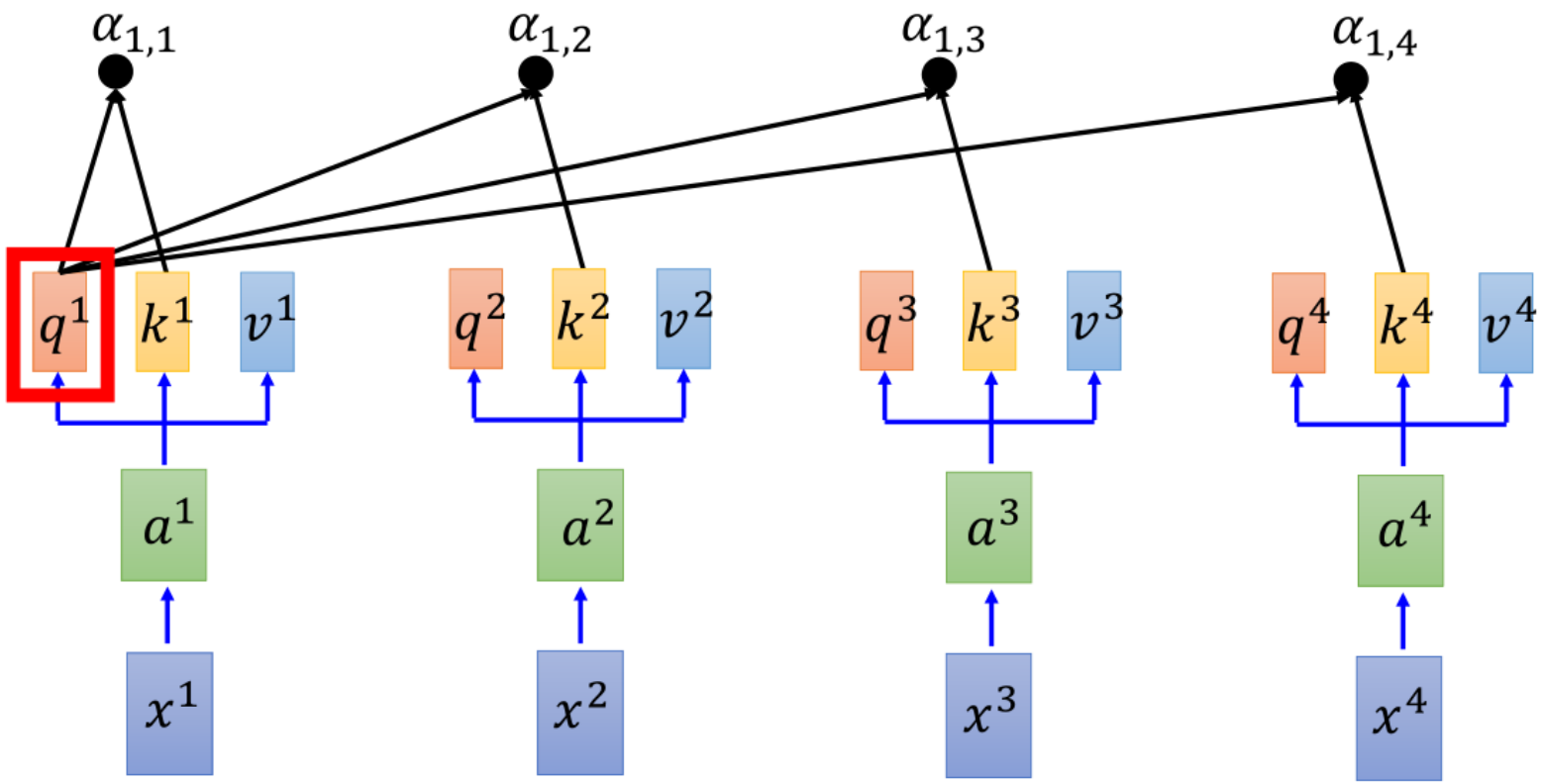
[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

Self-attention

拿每個 query q 去對每個 key k 做 attention

d is the dim of q and k

Scaled Dot-Product Attention: $\alpha_{1,i} = \underbrace{q^1 \cdot k^i}_{\text{dot product}} / \sqrt{d}$

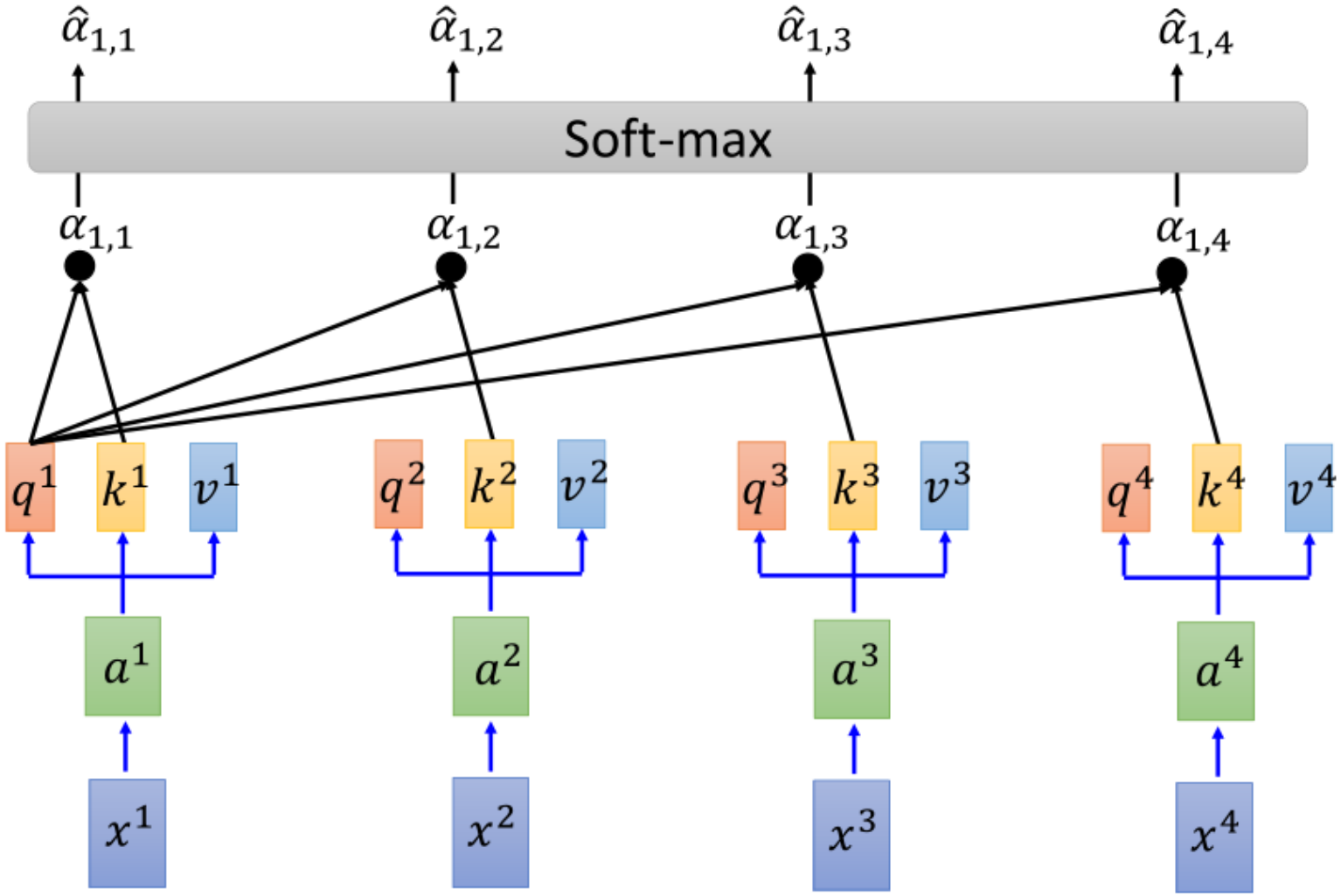


$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

Self-attention

$$\hat{\alpha}_{1,i} = exp(\alpha_{1,i}) / \sum_j exp(\alpha_{1,j})$$



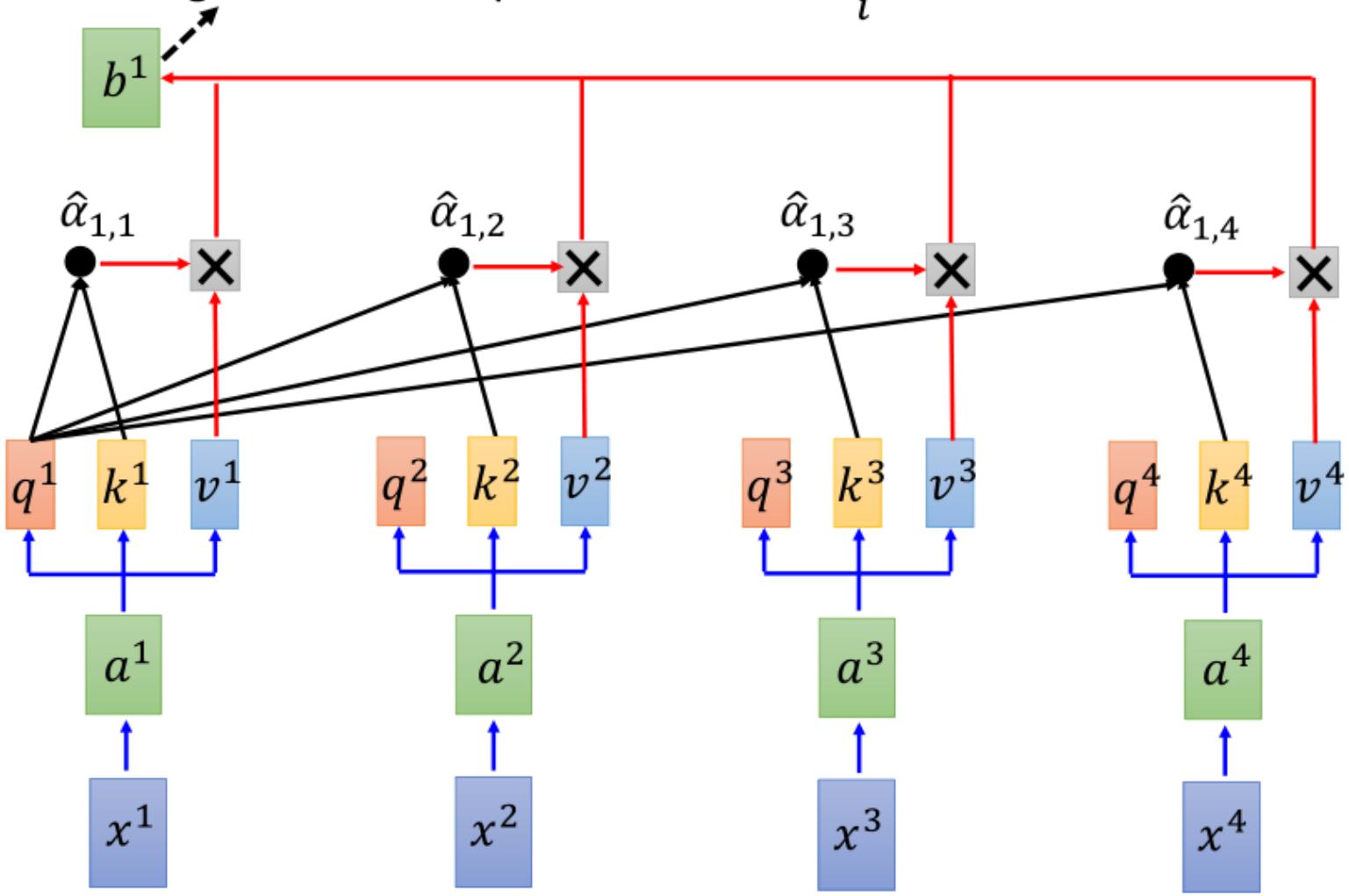
[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

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

Self-attention

Considering the whole sequence

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

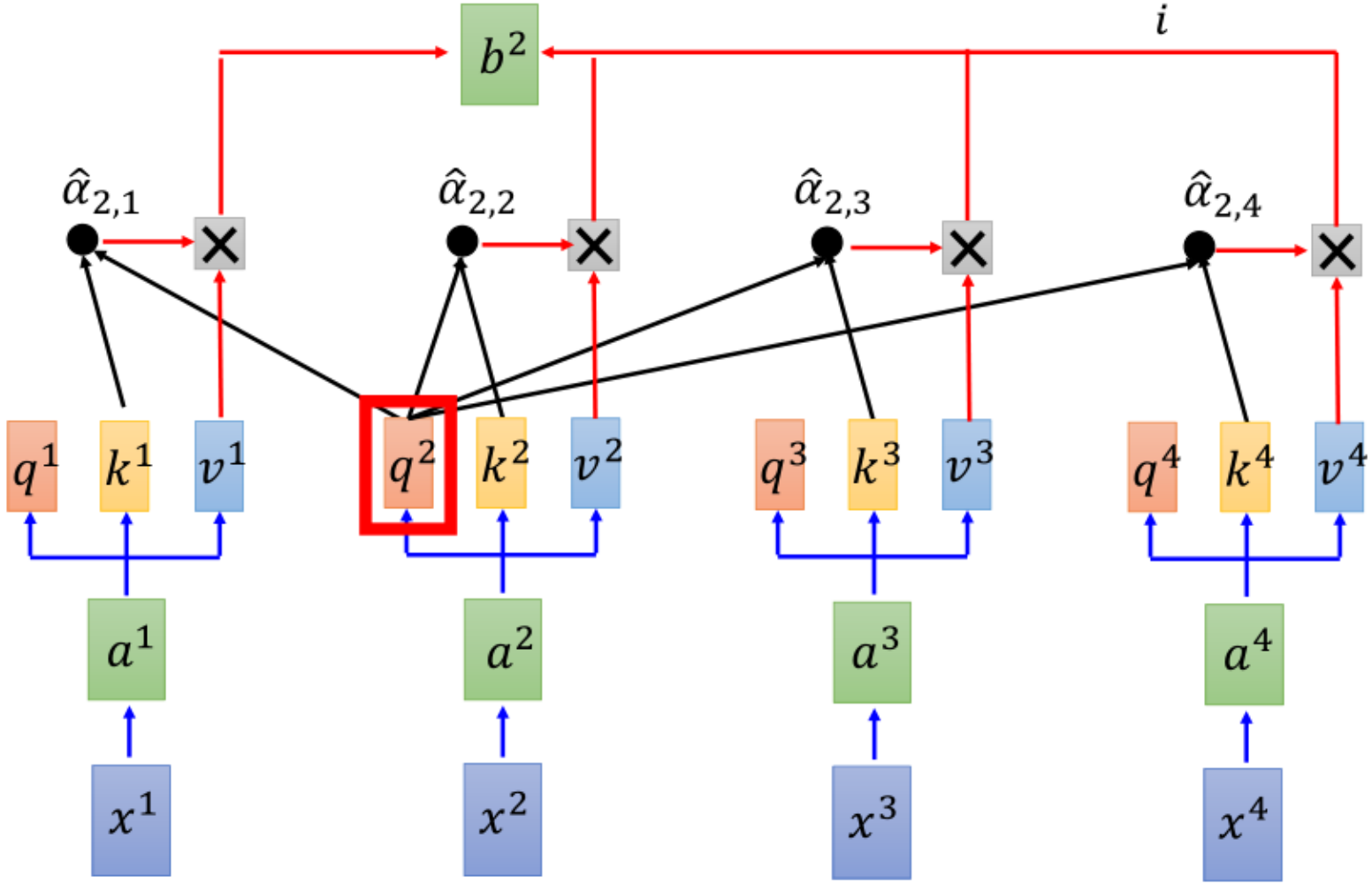


[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

Self-attention

拿每個 query q 去對每個 key k 做 attention

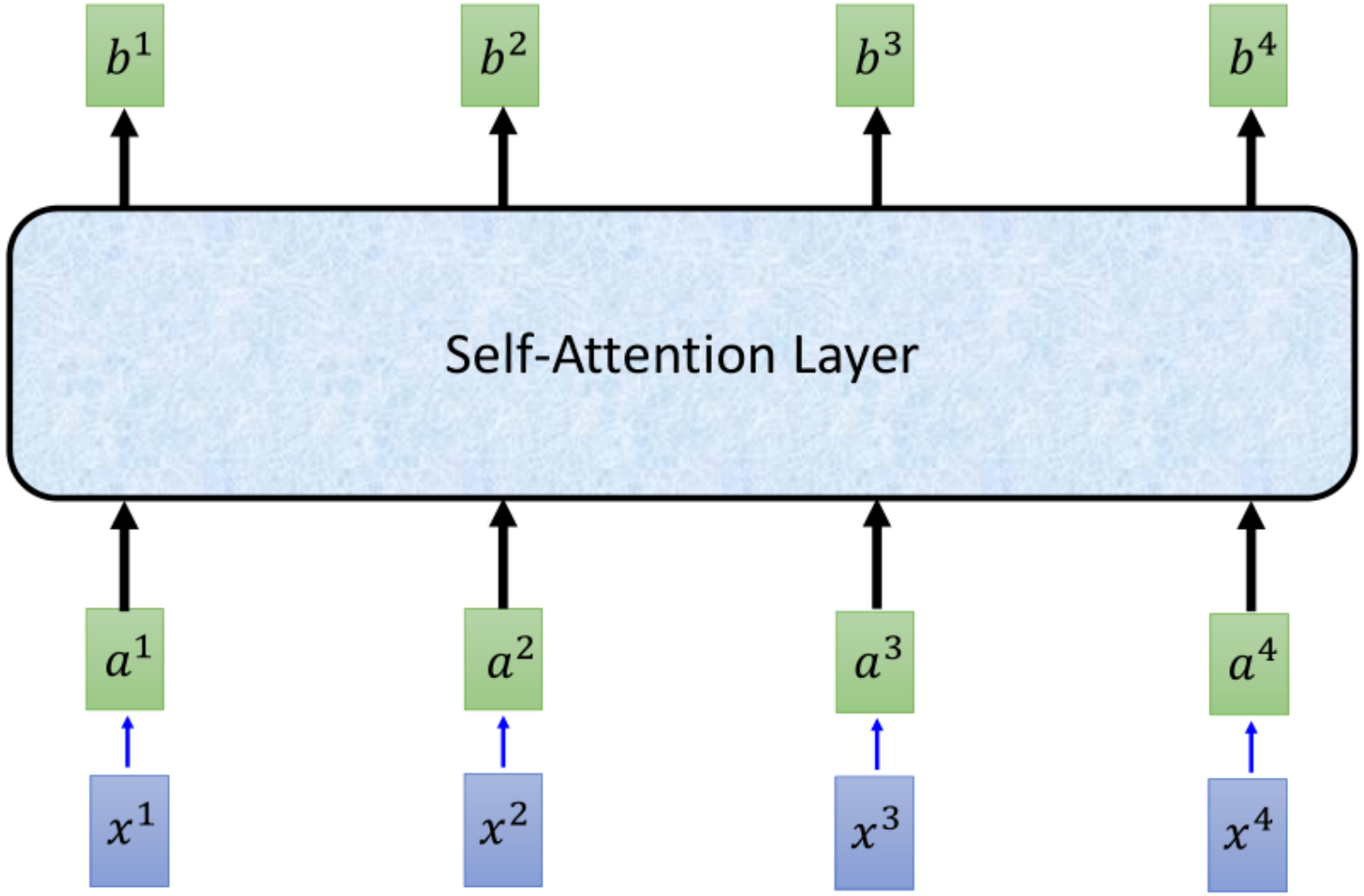
$$b^2 = \sum_i \hat{\alpha}_{2,i} v^i$$

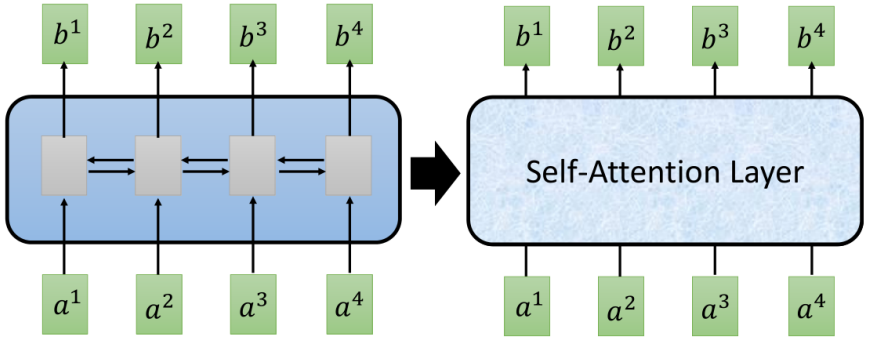


[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

Self-attention

b^1, b^2, b^3, b^4 can be parallelly computed.





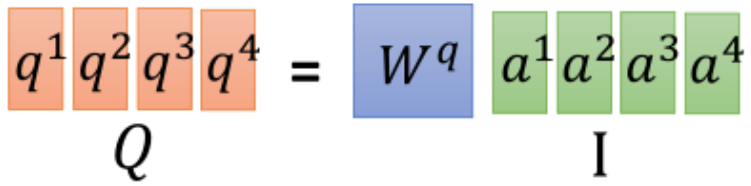
[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

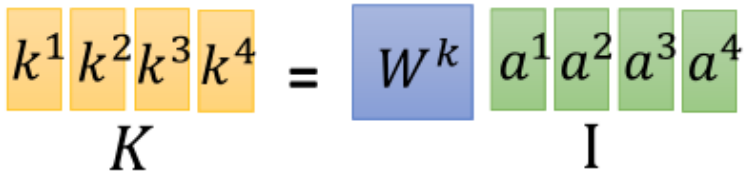
Self-attention

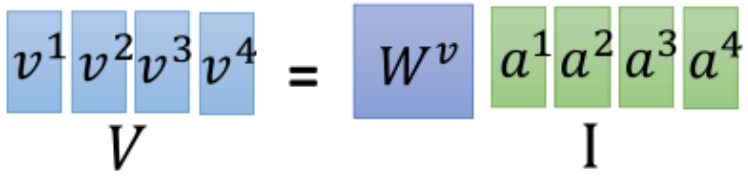
$$q^i = W^q a^i$$

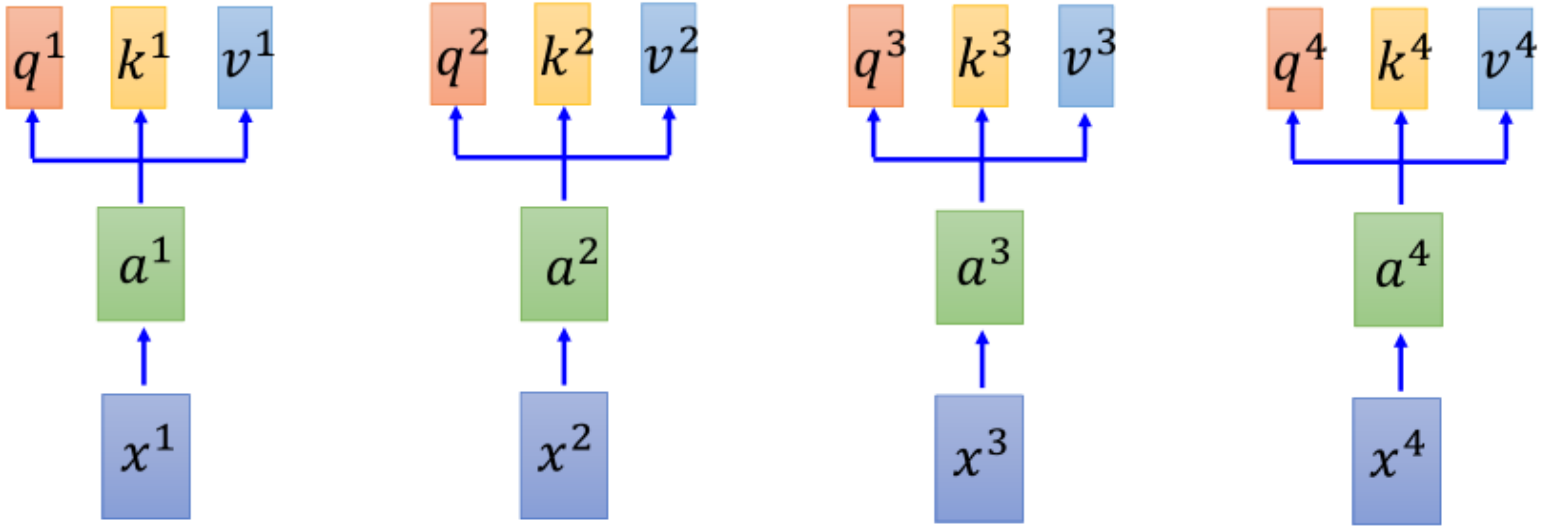
$$k^i = W^k a^i$$

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







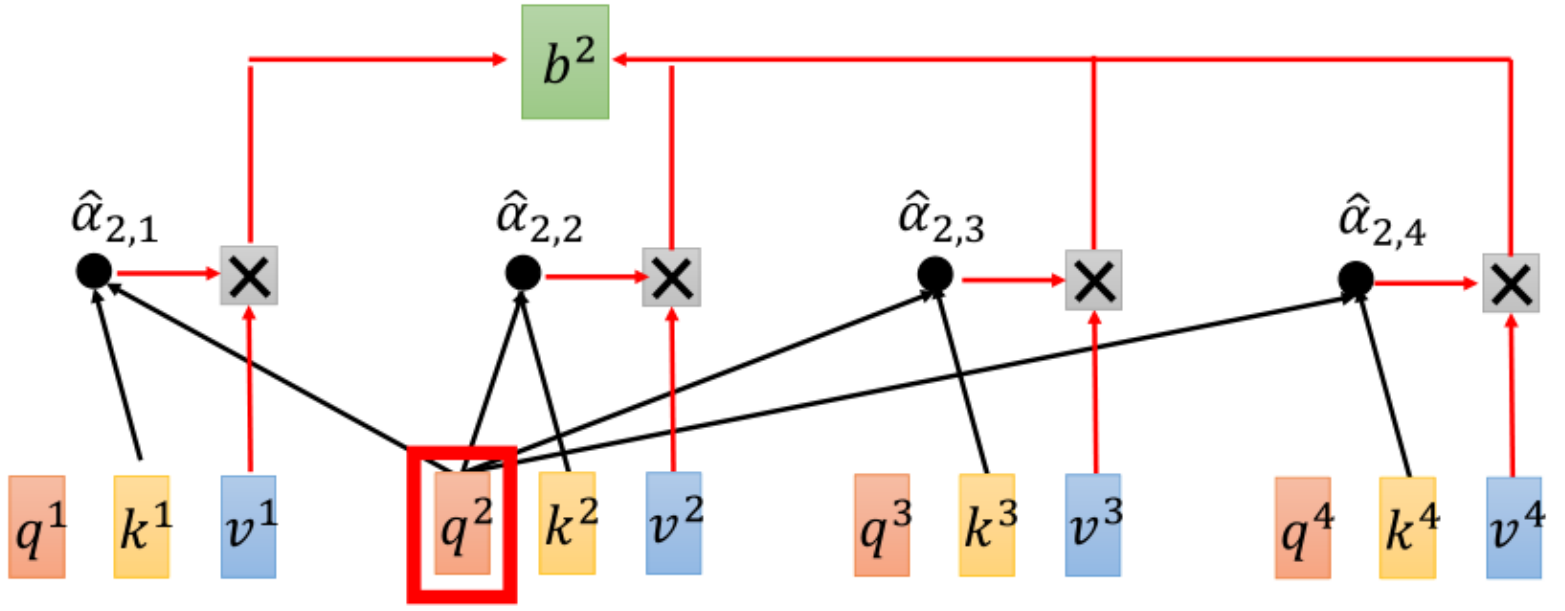


$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

Self-attention

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



$$\begin{bmatrix} \hat{\alpha}_{1,1} & \hat{\alpha}_{2,1} & \hat{\alpha}_{3,1} & \hat{\alpha}_{4,1} \\ \hat{\alpha}_{1,2} & \hat{\alpha}_{2,2} & \hat{\alpha}_{3,2} & \hat{\alpha}_{4,2} \\ \hat{\alpha}_{1,3} & \hat{\alpha}_{2,3} & \hat{\alpha}_{3,3} & \hat{\alpha}_{4,3} \\ \hat{\alpha}_{1,4} & \hat{\alpha}_{2,4} & \hat{\alpha}_{3,4} & \hat{\alpha}_{4,4} \end{bmatrix} \leftarrow \begin{bmatrix} \alpha_{1,1} & \alpha_{2,1} & \alpha_{3,1} & \alpha_{4,1} \\ \alpha_{1,2} & \alpha_{2,2} & \alpha_{3,2} & \alpha_{4,2} \\ \alpha_{1,3} & \alpha_{2,3} & \alpha_{3,3} & \alpha_{4,3} \\ \alpha_{1,4} & \alpha_{2,4} & \alpha_{3,4} & \alpha_{4,4} \end{bmatrix} = \begin{bmatrix} k^1 \\ k^2 \\ k^3 \\ k^4 \end{bmatrix} \begin{bmatrix} q^1 & q^2 & q^3 & q^4 \end{bmatrix}$$

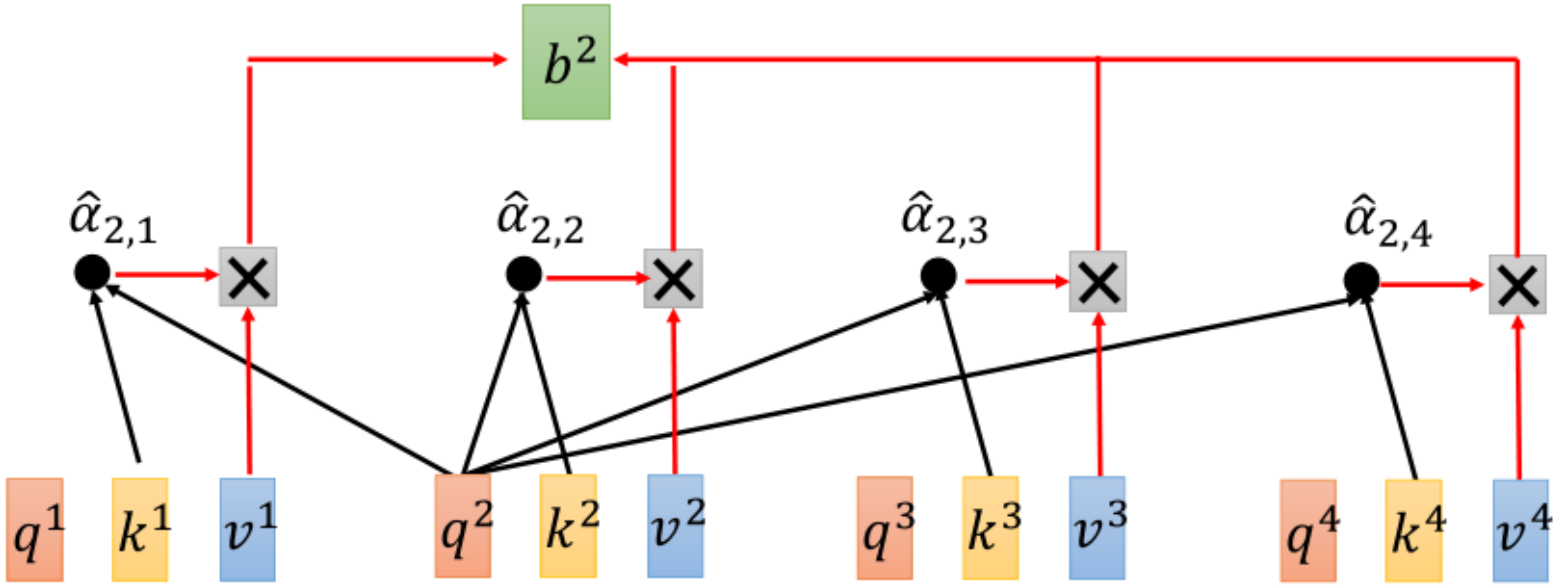
$\hat{A}$ A K^T Q

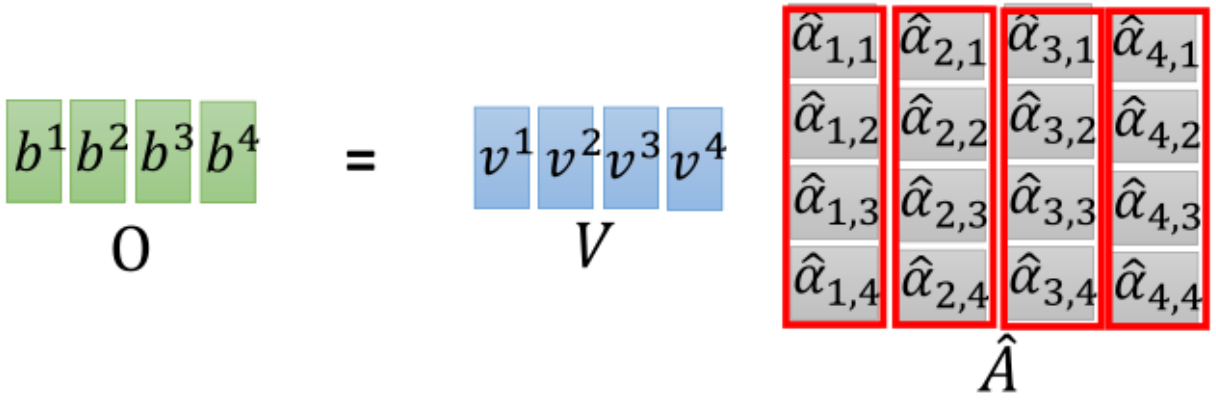
[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

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

Self-attention

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



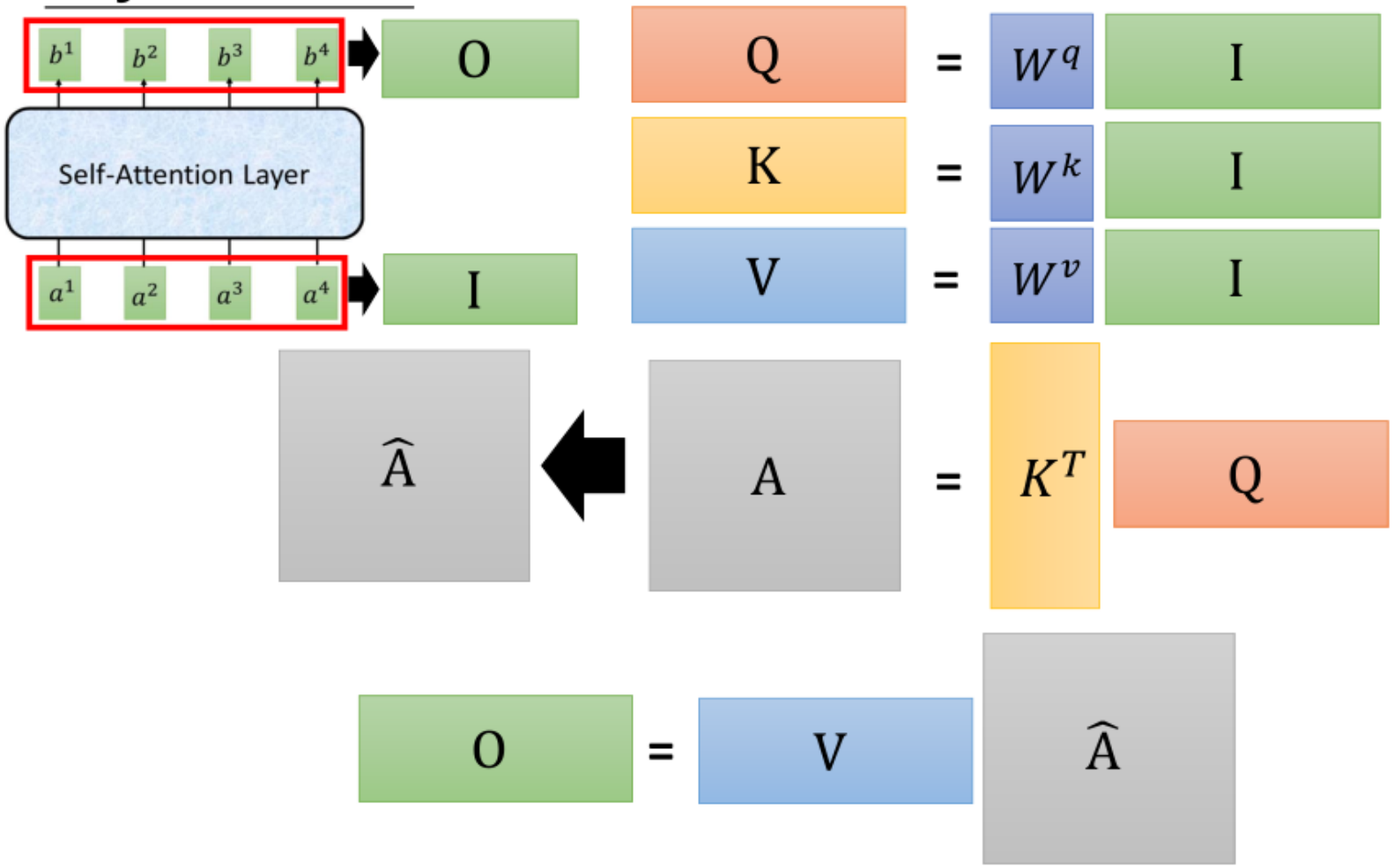


O V $\hat{A}$

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

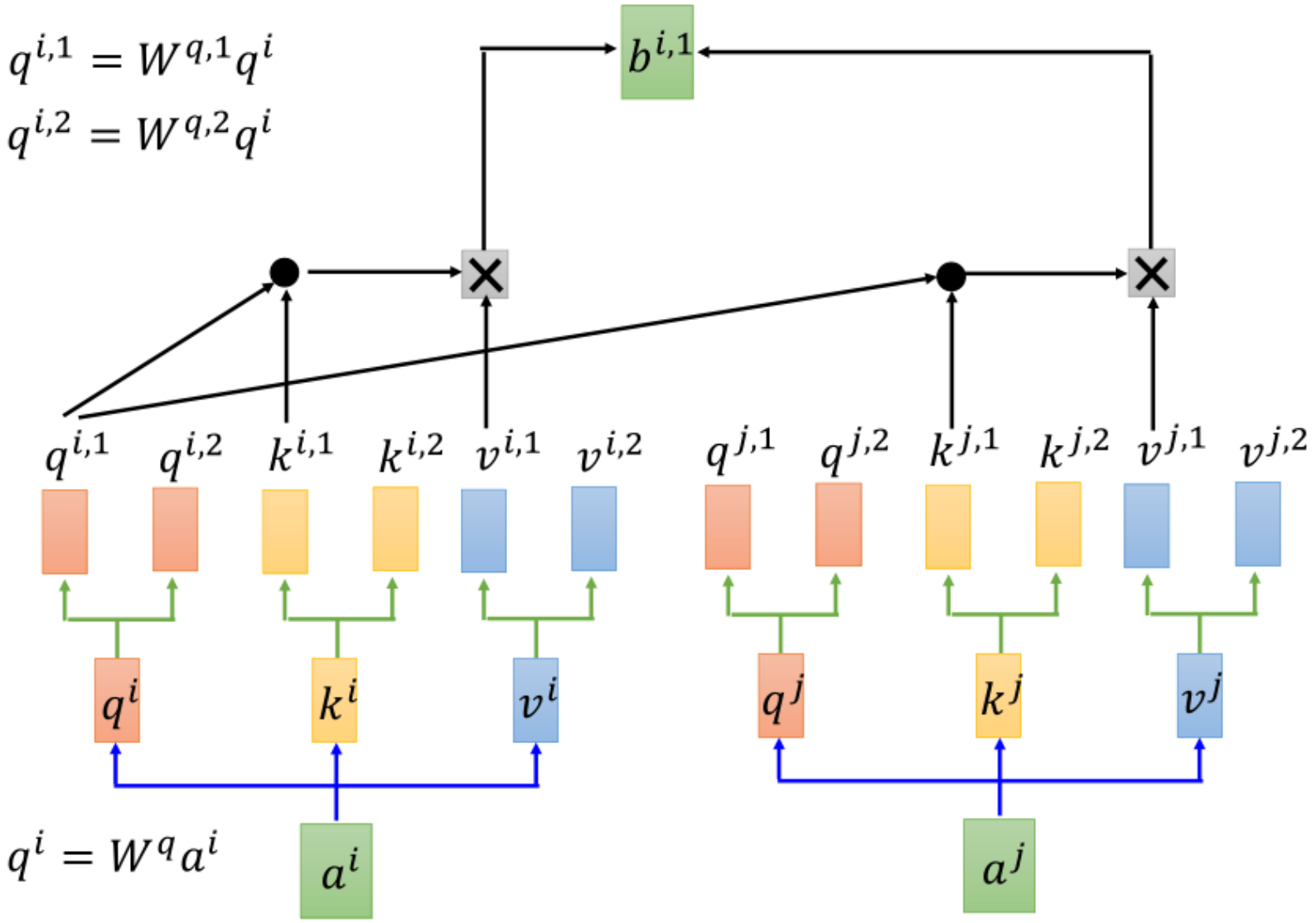
[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

Self-attention



[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

Multi-head Self-attention (2 heads as example)

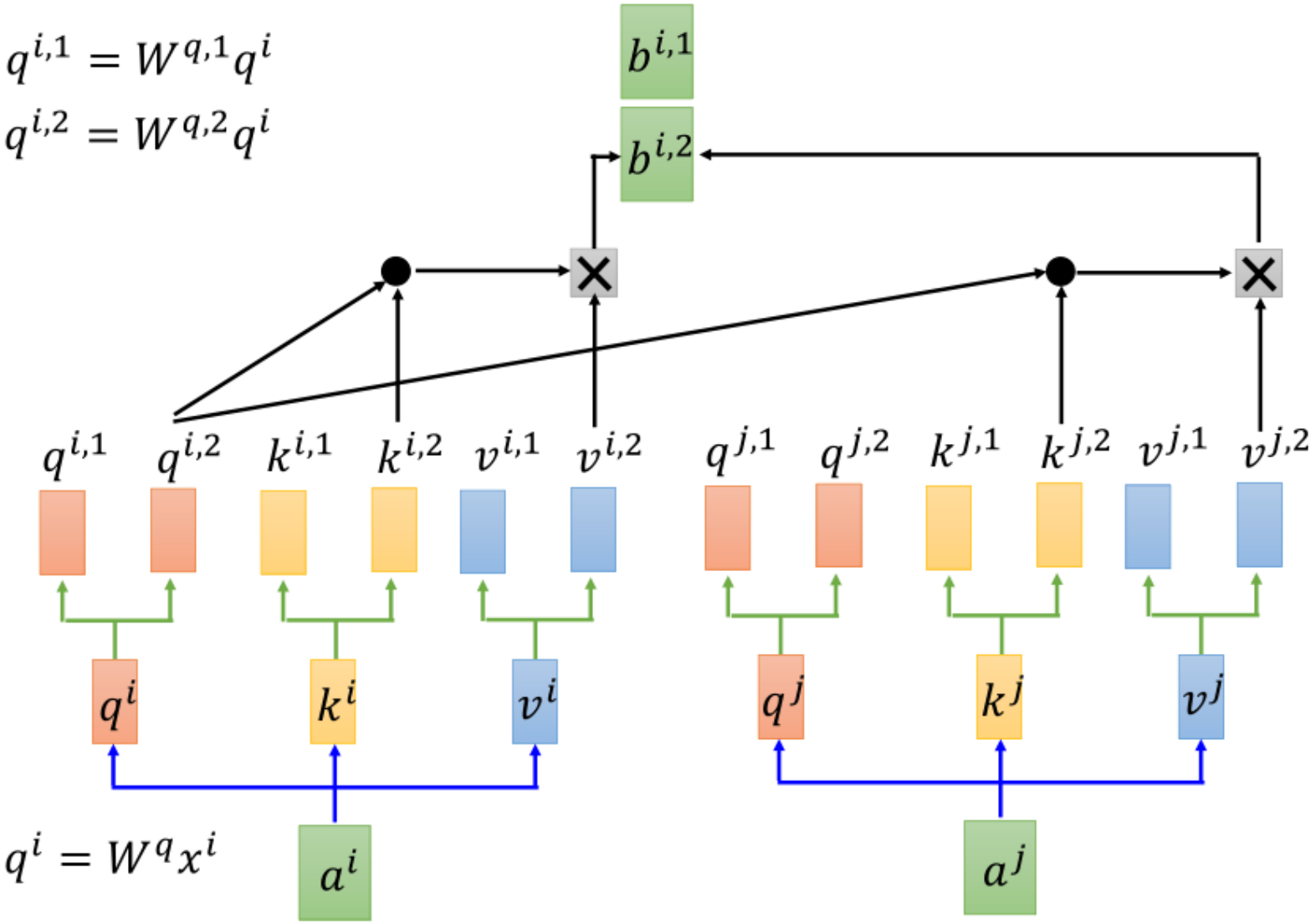


[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

Multi-head Self-attention (2 heads as example)

$$q^{i,1} = W^{q,1} q^i$$

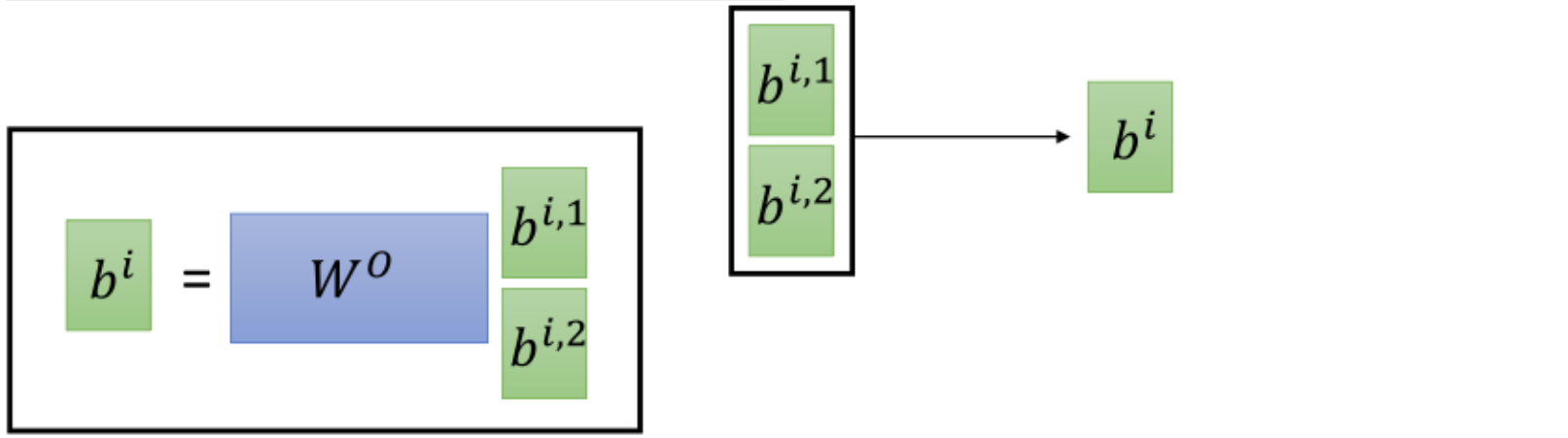
$$q^{i,2} = W^{q,2} q^i$$

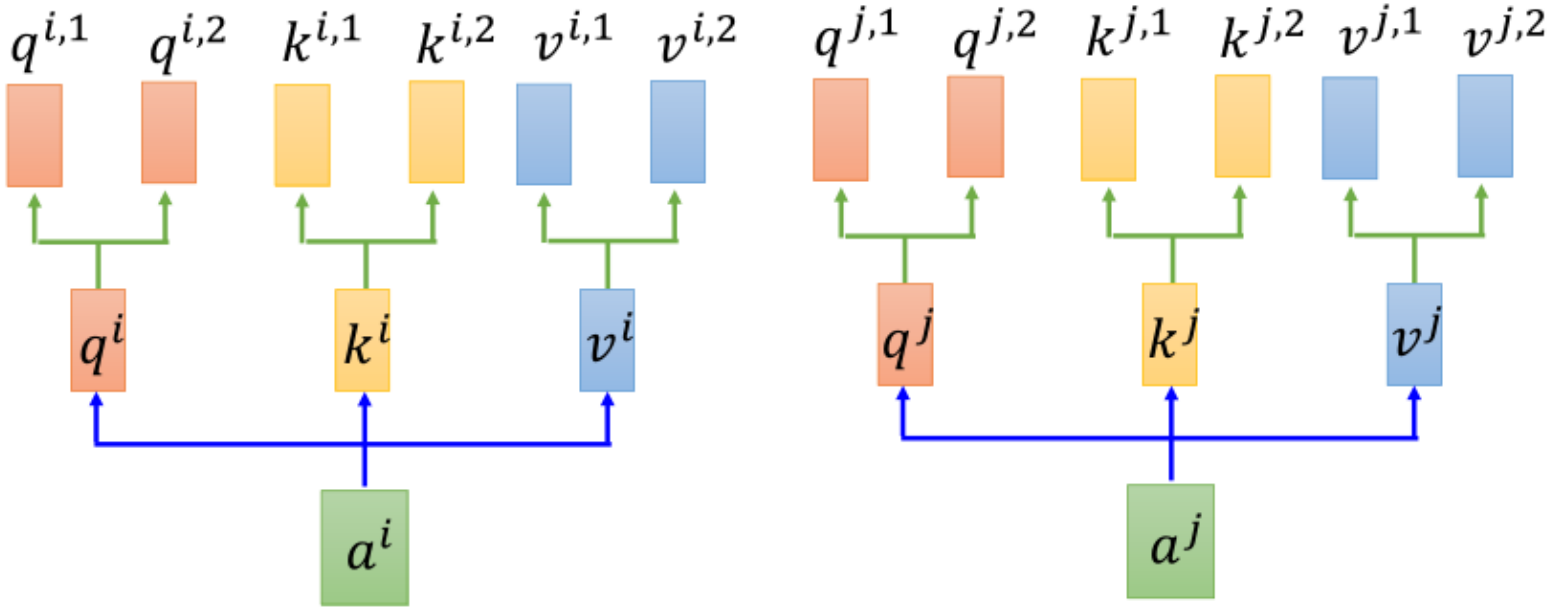


$$q^i = W^q x^i$$

[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

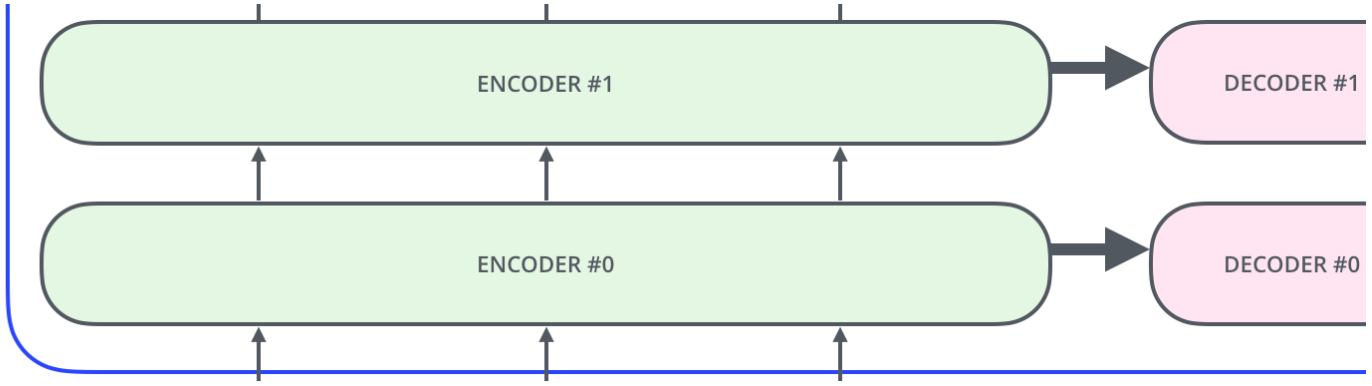
Multi-head Self-attention (2 heads as example)

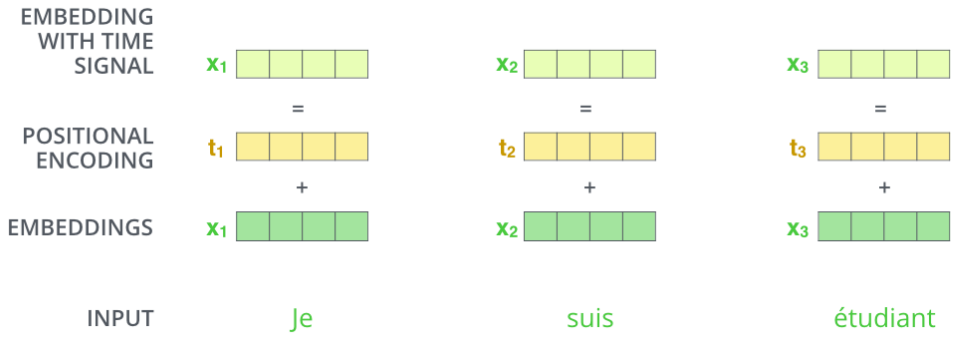




[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

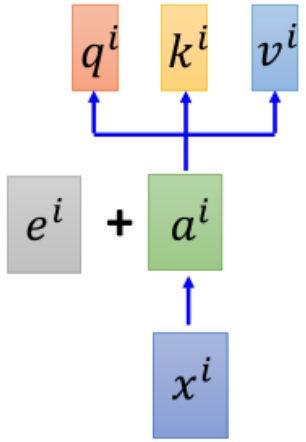
Positional Encoding





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

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



在上式中， pos 表示单词的位置， i 表示单词的维度。关于位置编码的实现可在Google开源的算法中 `get_timing_signal_1d()` 函数找到对应的代码。

作者这么设计的原因是考虑到在NLP任务中，除了单词的绝对位置，单词的相对位置也非常重要。根据公式 $sin(\alpha+\beta) = sin\alpha cos\beta + cos\alpha sin\beta$ 以及 $cos(\alpha+\beta) = cos\alpha cos\beta - sin\alpha sin\beta$ ，这表明位置 $k+p$ 的位置向量可以表示为位置 k 的特征向量的线性变化，这为模型捕捉单词之间的相对位置关系提供了非常大的便利。

完整的Transformer结构

Encoder

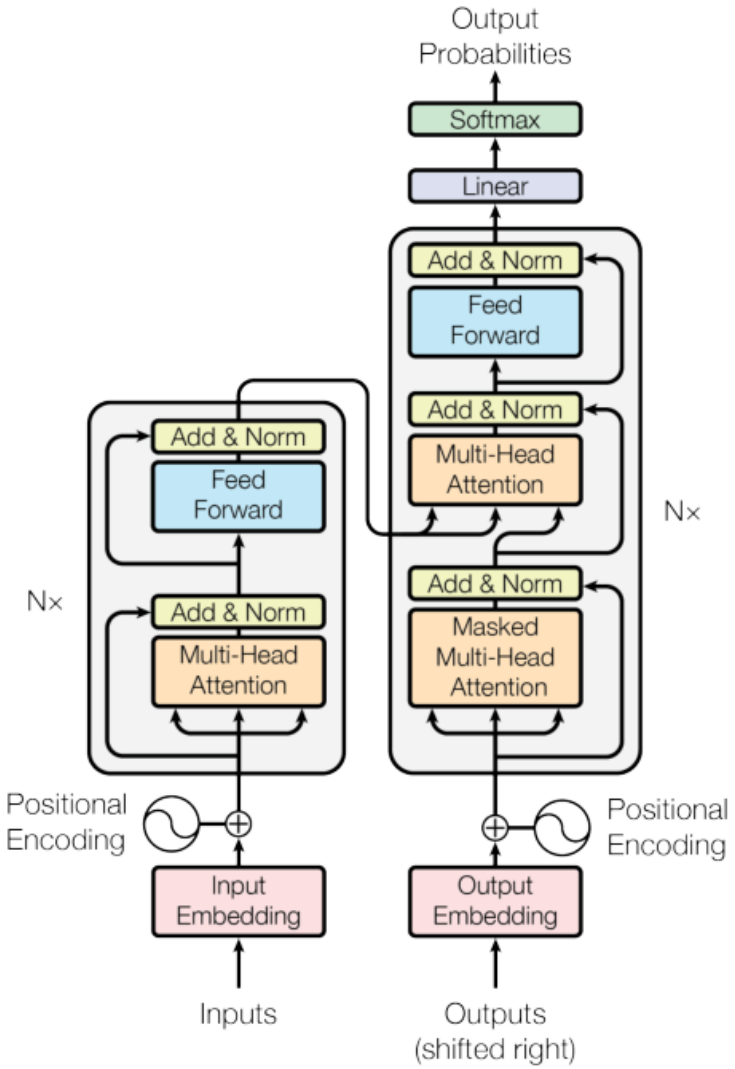
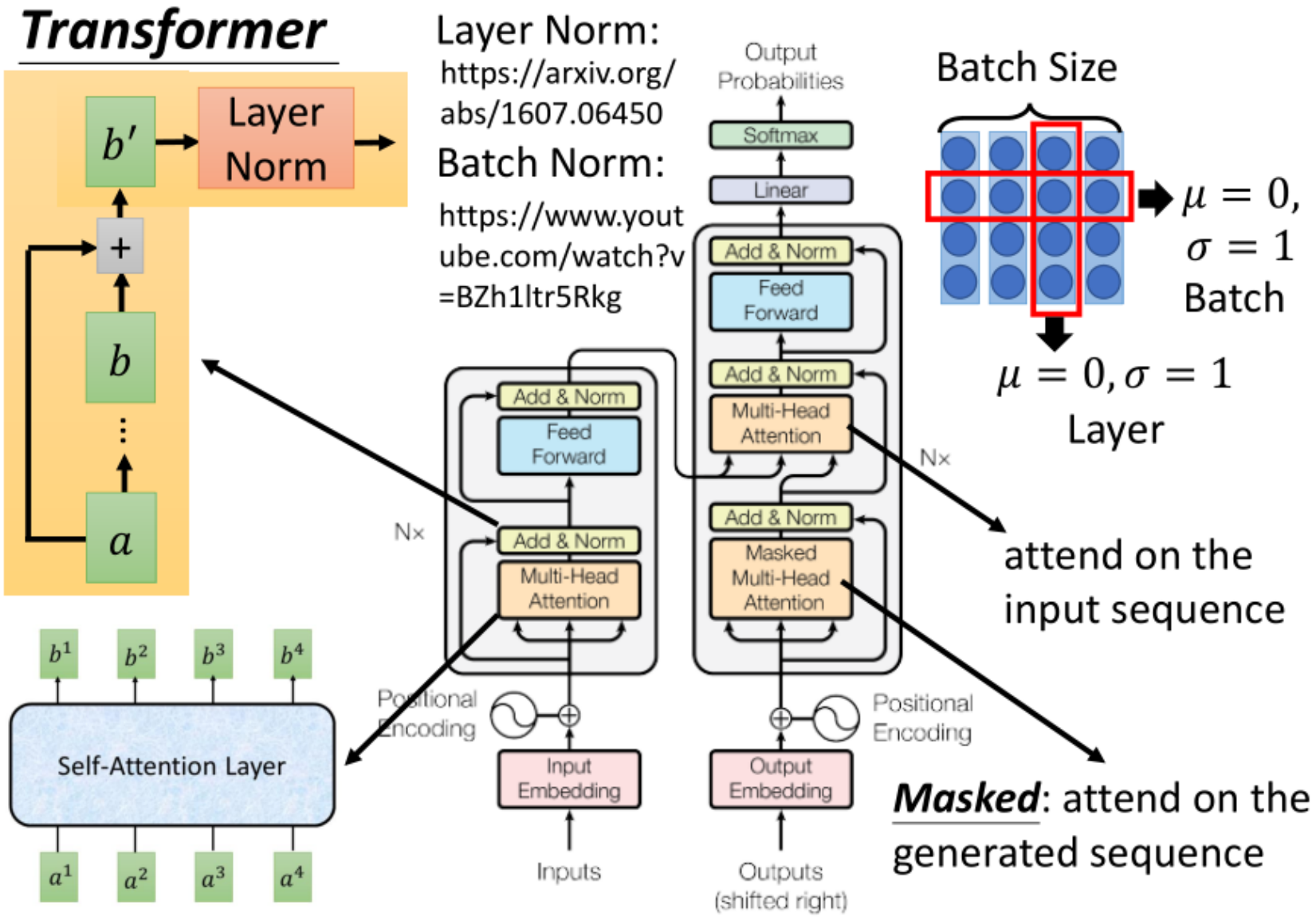


Figure 1: The Transformer - model architecture.

Decoder



[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

解码器解码之后，解码的特征向量经过一层激活函数为softmax的全连接层之后得到反映每个单词概率的输出向量。此时我们便可以通过CTC等损失函数训练模型了

在encoder-decoder attention中,Q 来自于解码器的上一个输出，K和V则来自于与编码器的输出

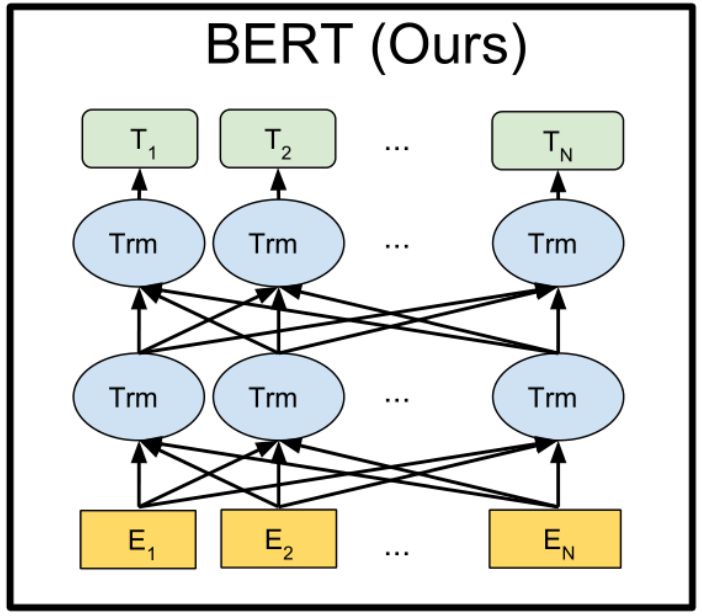
由于在机器翻译中，解码过程是一个顺序操作的过程，也就是当解码第k个特征向量时，我们只能看到第k-1 及其之前的解码结果，论文中把这种情况下的multi-head attention叫做masked multi-head attention。

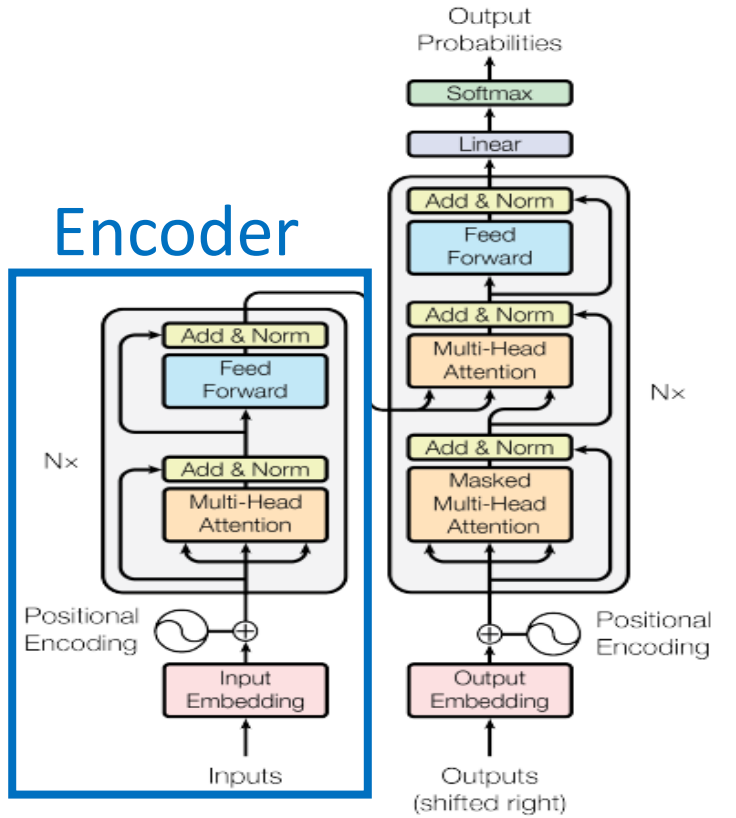
BERT

1.双向Transformer的Encoder；

2.Masked LM和Next Sentence Prediction方法；

3.Learned from a large amount of text without annotation；

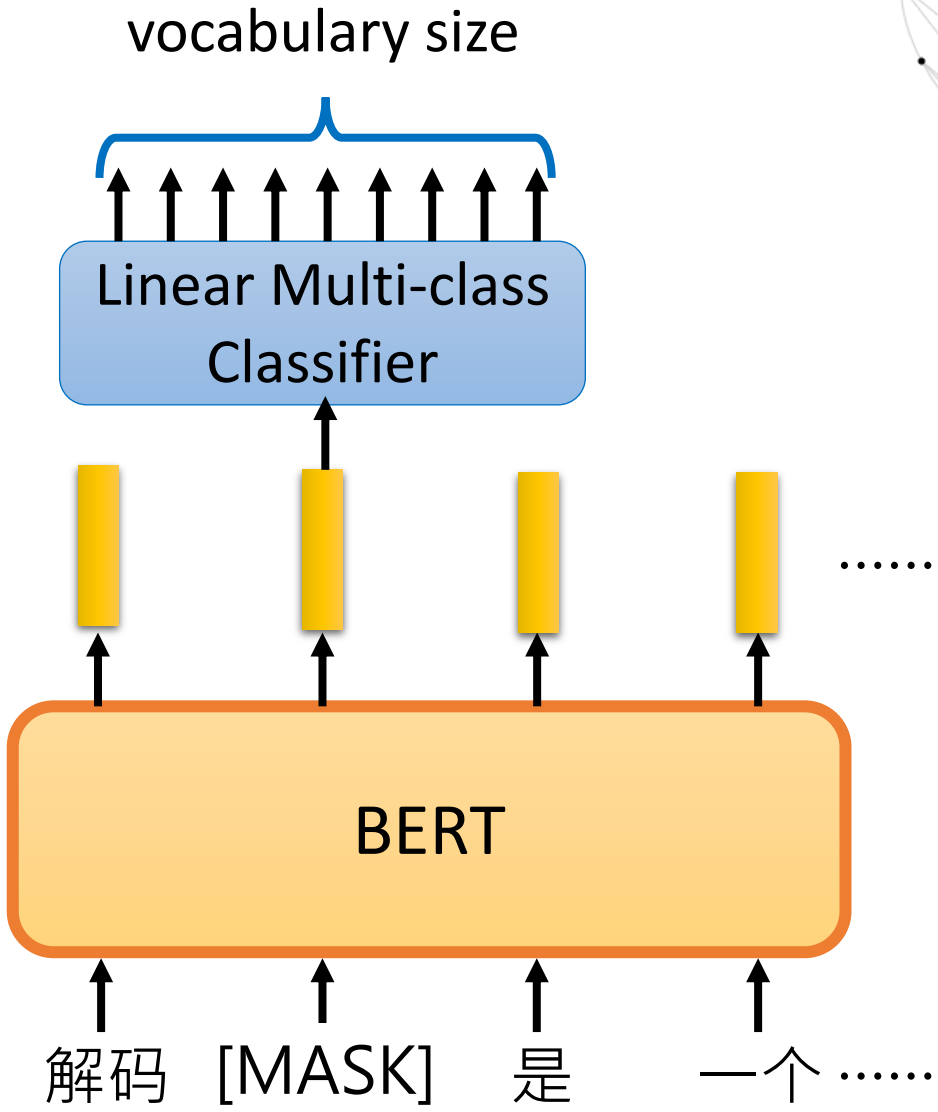




[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/BERT%20(v3).pdf

Approach 1: Masked LM

Predicting the masked word



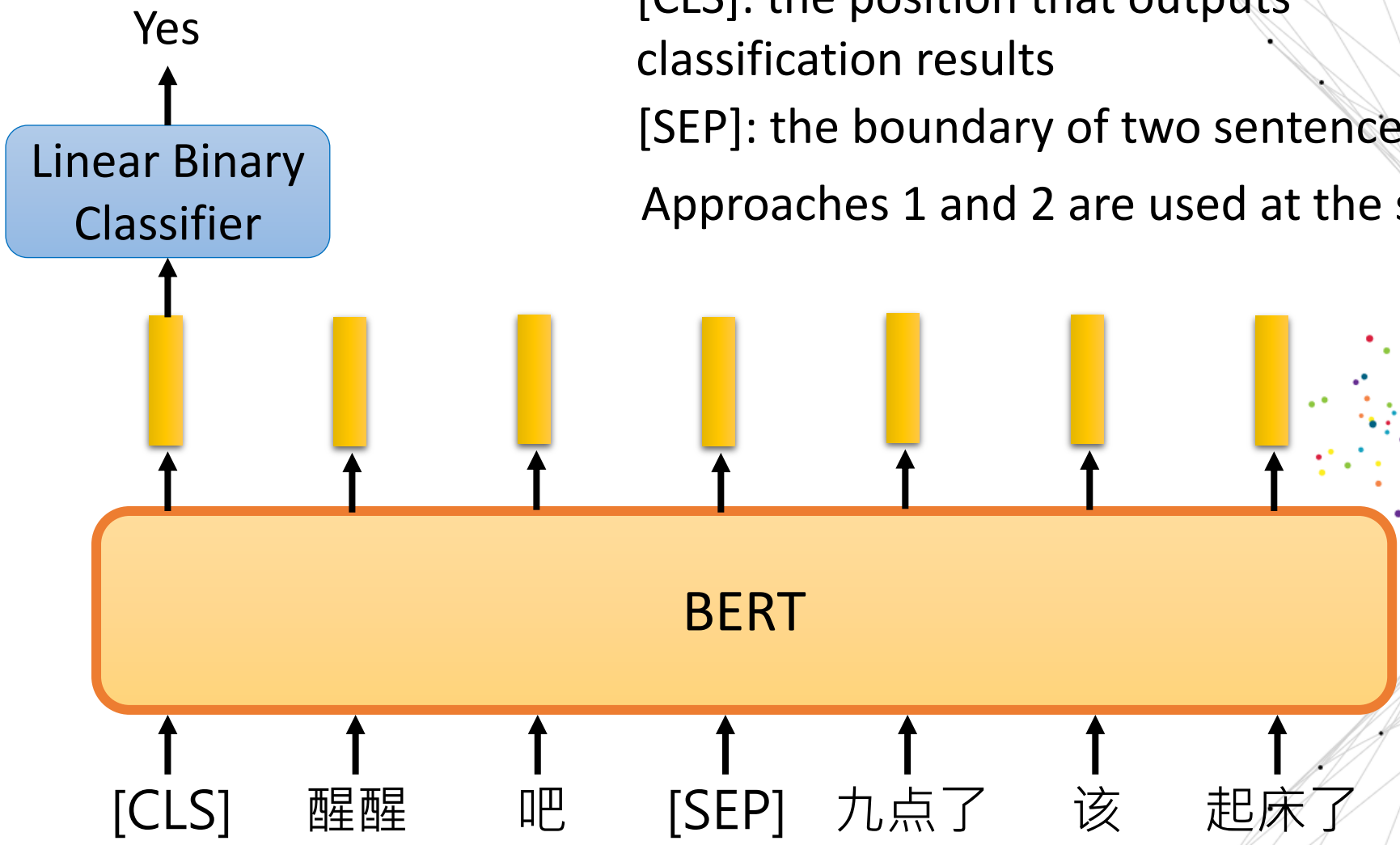
[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/BERT%20(v3).pdf

Approach2: Next Sentence Prediction

[CLS]: the position that outputs classification results

[SEP]: the boundary of two sentences

Approaches 1 and 2 are used at the same time.

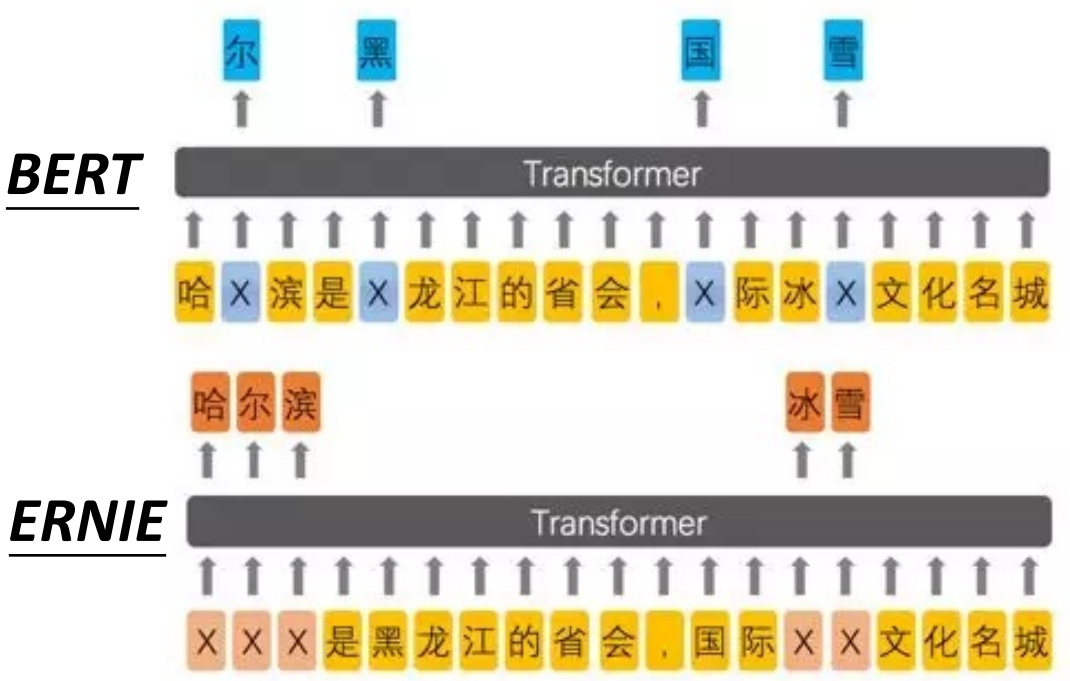


[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/BERT%20(v3).pdf

Rank	Name	Model	URL	Score	CoLA	SST-2	MRPC	STS-B	QQP	MNLI-m	MNLI-mm	QNLI	RTE	WNLI	AX
1	ERNIE Team - Baidu	ERNIE		90.4	74.4	97.5	93.5/91.4	93.0/92.6	75.2/90.9	91.4	91.0	96.6	90.9	94.5	51.7
2	PING-AN Omni-Sinitic	ALBERT + DAAF + NAS		90.4	71.5	97.2	94.0/92.0	93.0/92.4	76.1/91.0	91.6	91.3	97.5	91.7	94.5	51.2
3	Alibaba DAMO NLP	StructBERT		90.3	75.3	97.1	93.9/91.9	93.0/92.5	74.8/91.0	90.9	90.7	96.4	90.2	94.5	49.1
4	T5 Team - Google	T5		90.3	71.6	97.5	92.8/90.4	93.1/92.8	75.1/90.6	92.2	91.9	96.9	92.8	94.5	53.1
5	Microsoft D365 AI & MSR AI & GATECH	MT-DNN-SMART		89.9	69.5	97.5	93.7/91.6	92.9/92.5	73.9/90.2	91.0	90.8	99.2	89.7	94.5	50.2
6	ELECTRA Team	ELECTRA-Large + Standard Tricks		89.4	71.7	97.1	93.1/90.7	92.9/92.5	75.6/90.8	91.3	90.8	95.8	89.8	91.8	50.7
7	Huawei Noah's Ark Lab	NEZHA-Large		88.7	67.4	97.2	93.2/91.0	92.2/91.6	74.1/90.2	90.8	90.2	95.7	88.5	93.2	45.0
8	Microsoft D365 AI & UMD	FreeLB-RoBERTa (ensemble)		88.4	68.0	96.8	93.1/90.8	92.3/92.1	74.8/90.3	91.1	90.7	95.6	88.7	89.0	50.1
9	Junjie Yang	HIRE-RoBERTa		88.3	68.6	97.1	93.0/90.7	92.4/92.0	74.3/90.2	90.7	90.4	95.5	87.9	89.0	49.3
10	Facebook AI	RoBERTa		88.1	67.8	96.7	92.3/89.8	92.2/91.9	74.3/90.2	90.8	90.2	95.4	88.2	89.0	48.7
11	Microsoft D365 AI & MSR AI	MT-DNN-ensemble		87.6	68.4	96.5	92.7/90.3	91.1/90.7	73.7/89.9	87.9	87.4	96.0	86.3	89.0	42.8
12	GLUE Human Baselines	GLUE Human Baselines		87.1	66.4	97.8	86.3/80.8	92.7/92.6	59.5/80.4	92.0	92.8	91.2	93.6	95.9	-
13	Stanford Hazy Research	Snorkel MeTaL		83.2	63.8	96.2	91.5/88.5	90.1/89.7	73.1/89.9	87.6	87.2	93.9	80.9	65.1	39.9
14	XLM Systems	XLM (English only)		83.1	62.9	95.6	90.7/87.1	88.8/88.2	73.2/89.8	89.1	88.5	94.0	76.0	71.9	44.7
15	Zhuosheng Zhang	SemBERT		82.9	62.3	94.6	91.2/88.3	87.8/86.7	72.8/89.8	87.6	86.3	94.6	84.5	65.1	42.4

Enhanced Representation through Knowledge Integration (ERNIE)

- Designed for Chinese



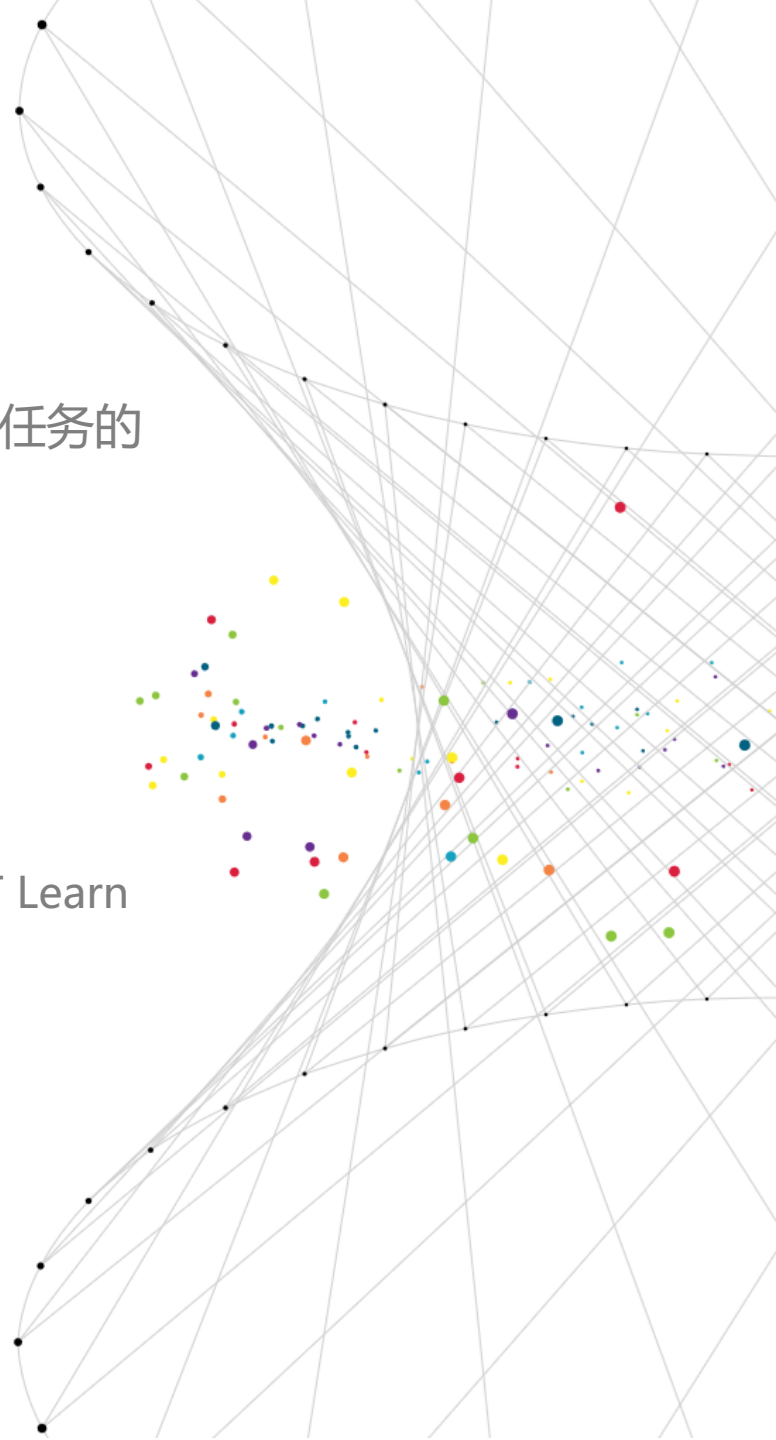
Source of image:
https://zhuanlan.zhihu.com/p/59436589

https://arxiv.org/abs/1904.09223

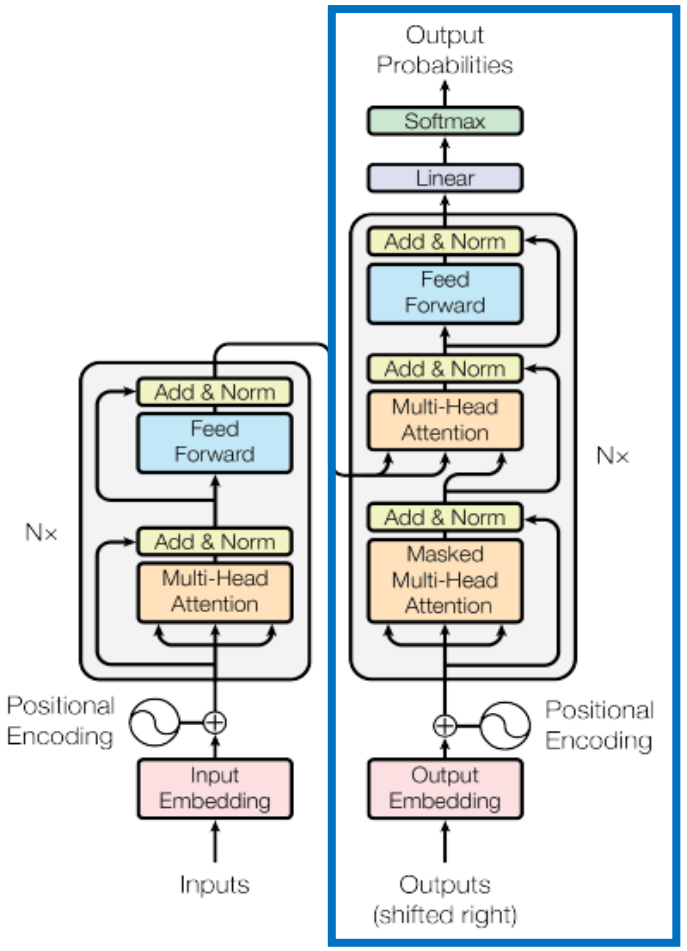
BERT的“里程碑”意义在于：证明了一个非常深的模型可以显著提高NLP任务的准确率，而这个模型可以从无标记数据集中预训练得到。
https://www.zhihu.com/question/298203515

- What does BERT learn about the structure of language?

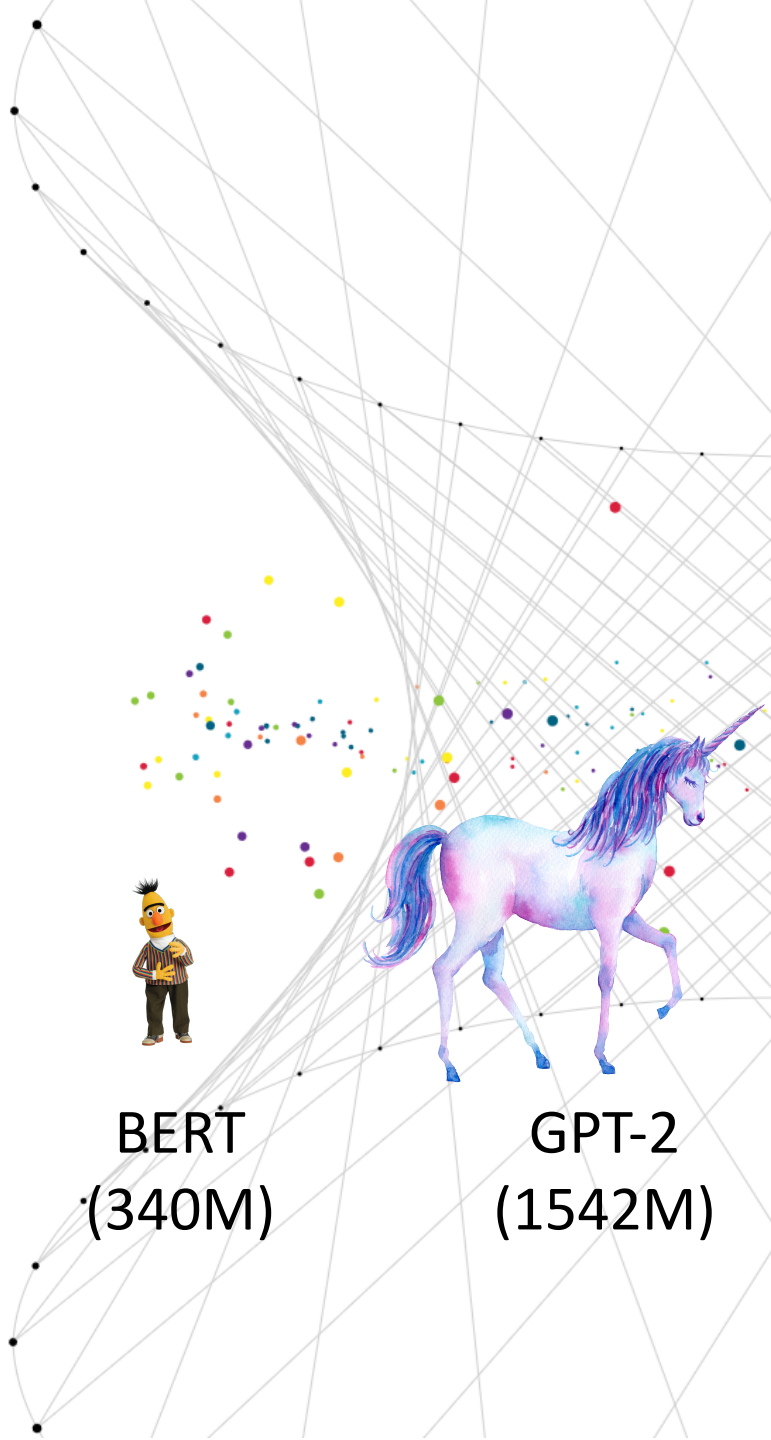
Jawahar, Ganesh & Sagot, Benoît & Seddah, Djamé. (2019). What Does BERT Learn about the Structure of Language?. 3651-3657. 10.18653/v1/P19-1356.



Generative Pre-Training (GPT)



Transformer Decoder



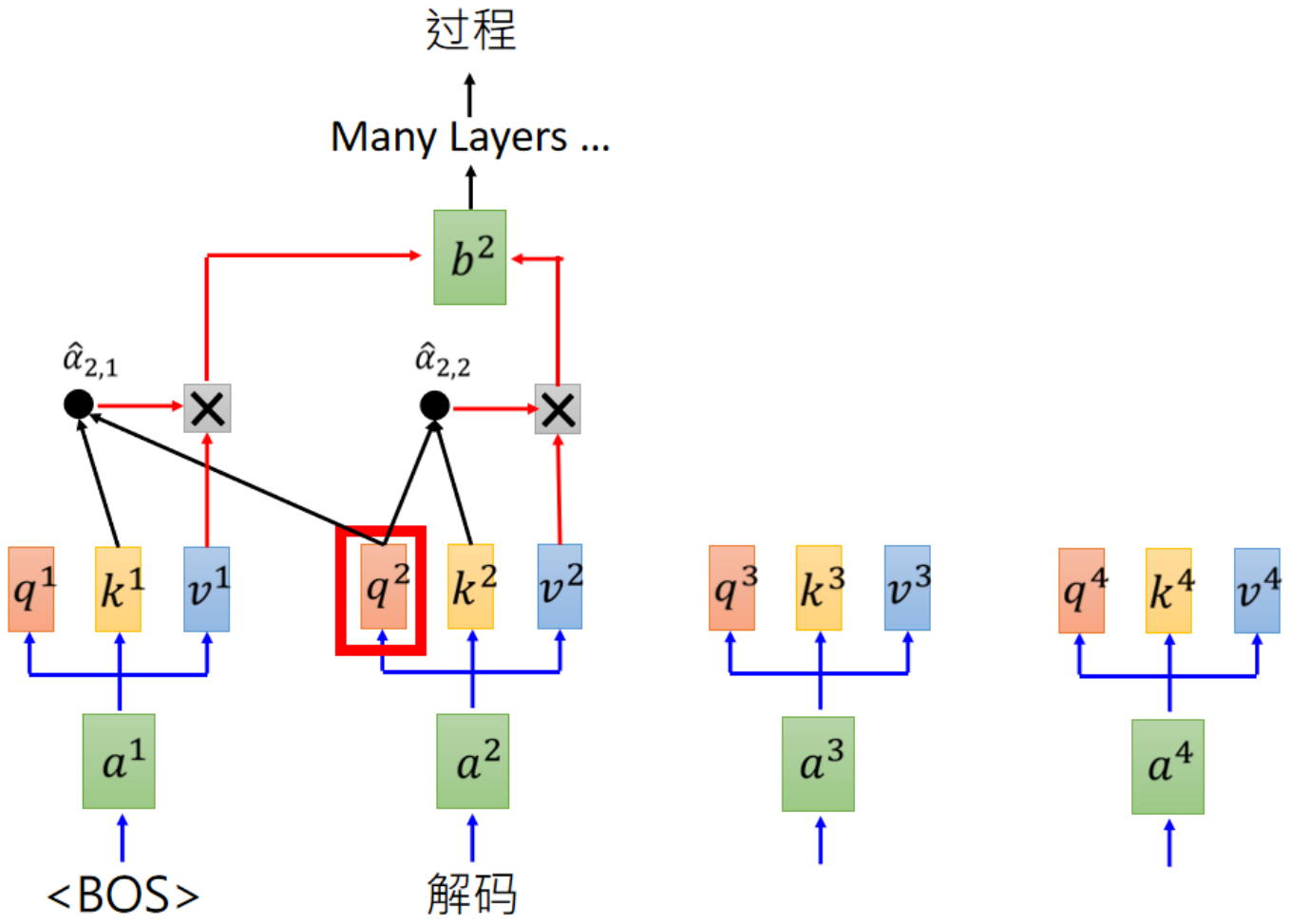
ELMO (94M)

BERT (340M)

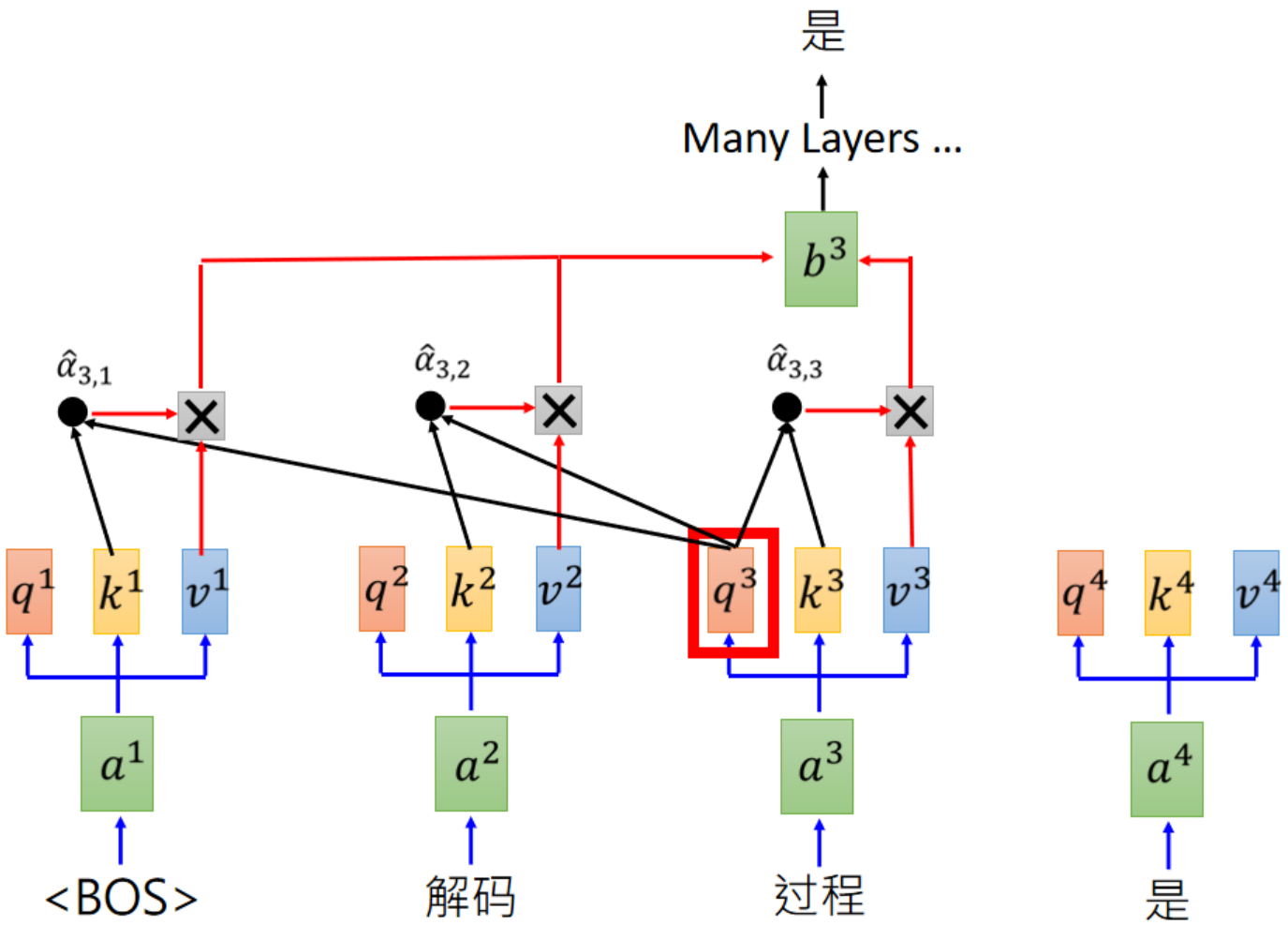
GPT-2 (1542M)

[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/BERT%20(v3).pdf

Generative Pre-Training (GPT)



Generative Pre-Training (GPT)



[1]http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/BERT%20(v3).pdf