深度學習
Deep Learning

Transformers
自注意力機制模型

Instructor: 林英嘉 (Ying-Jia Lin)
2025/04/07

Course GitHub

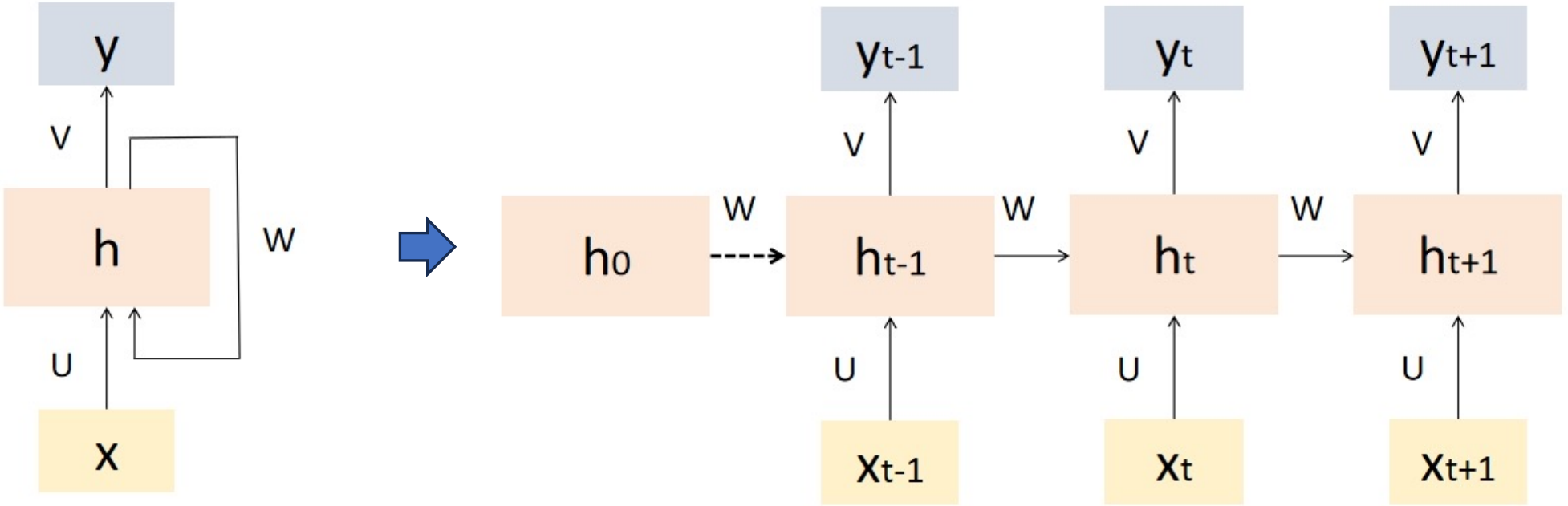
Slido # DL_0407

Outline

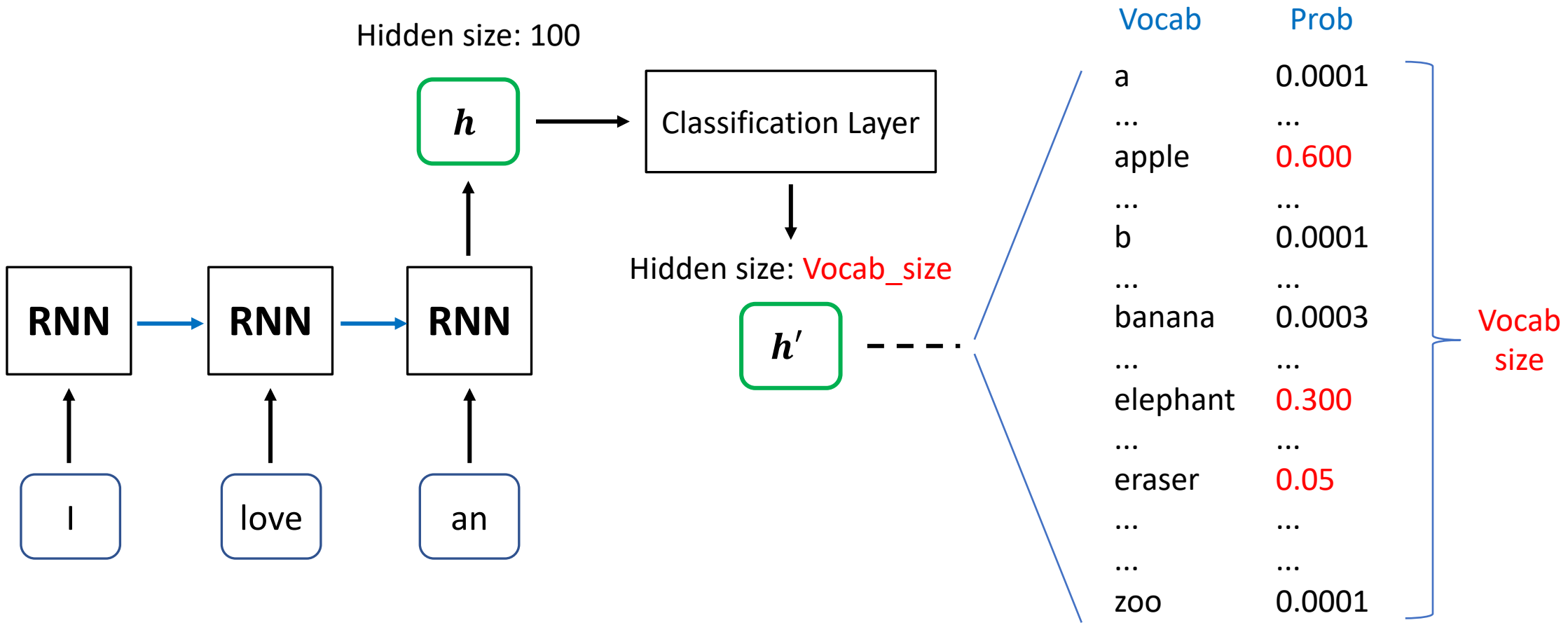
- Weakness of RNNs and CNNs [10 min]
- Transformers [60 min]
- Vision Transformer [20 min]
- Announcement about Midterm Exam [15 min]
- Project introduction [30 min]
- Quiz [15 min]

[Recap] Recurrent Neural Networks (RNN)

- Recurrent Neural Networks (RNNs) are a type of artificial neural network designed to handle sequential data by capturing temporal dependencies.
  - RNN 的“遞迴”是指它在每一個時間步中重複使用相同的參數（同一組權重），並把先前的隱藏狀態傳到下一個時間步



[Recap] Text Generation with RNN



RNN code example

```
class RNNModel(torch.nn.Module):
def __init__(self, vocab_size=30000, embed_dim=300, hidden_dim=300):
    super().__init__()
    self.vocab_size = vocab_size
    self.hidden_dim = hidden_dim
    self.embedding_layer = torch.nn.Embedding(vocab_size, embed_dim)
    self.rnn = torch.nn.RNN(embed_dim, hidden_dim)
    self.fc = torch.nn.Linear(hidden_dim, 2)

def forward(self, X):
    E = self.embedding_layer(X)
    H_out, H_n = self.rnn(E)
    logits = self.fc(H_out[:, -1, :])
    return logits
```

把每個字都轉成長度為 embed_dim 的向量

RNN接受長度為 embed_dim 的向量
每個時間點產生長度為 hidden_dim 的
隱藏狀態 (hidden state)

H_out‘s shape: BS x seq_len x hidden_dim

Tokens vs. words

- token 是 (語言) 模型在每個時間點處理的單位
- word 是語言本身的單位
  - token 可以是 word，也可以是 sub-word
  - 一個 word 可以是一個 token，但單純講 token 不一定指的是 word

Traditional word tokenization

Sub-word Tokenization

Issues with RNNs: Linear Interaction Distance

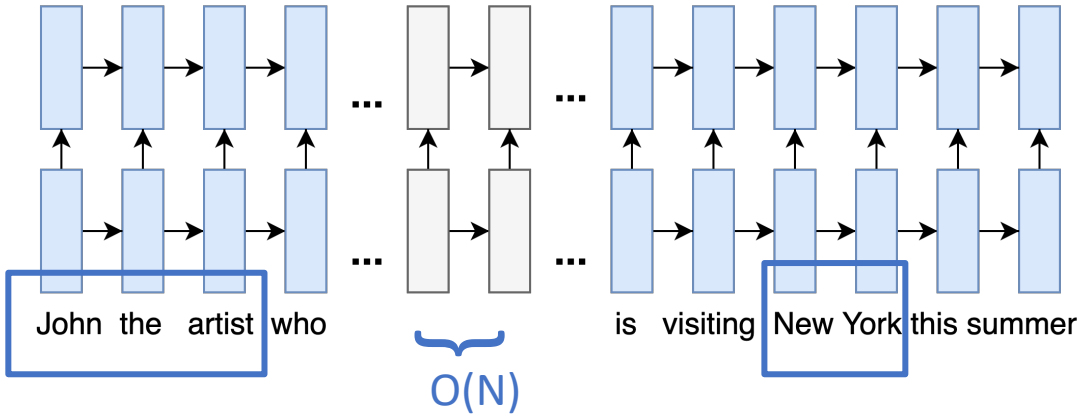
- In RNNs, the degree of words interact with each other is decided by their distance, but we already know that linear importance is not the right way to understand a sentence...

A Long Relative Clause

John, the artist who has painted many Murals in Vienna, is visiting New York this summer.

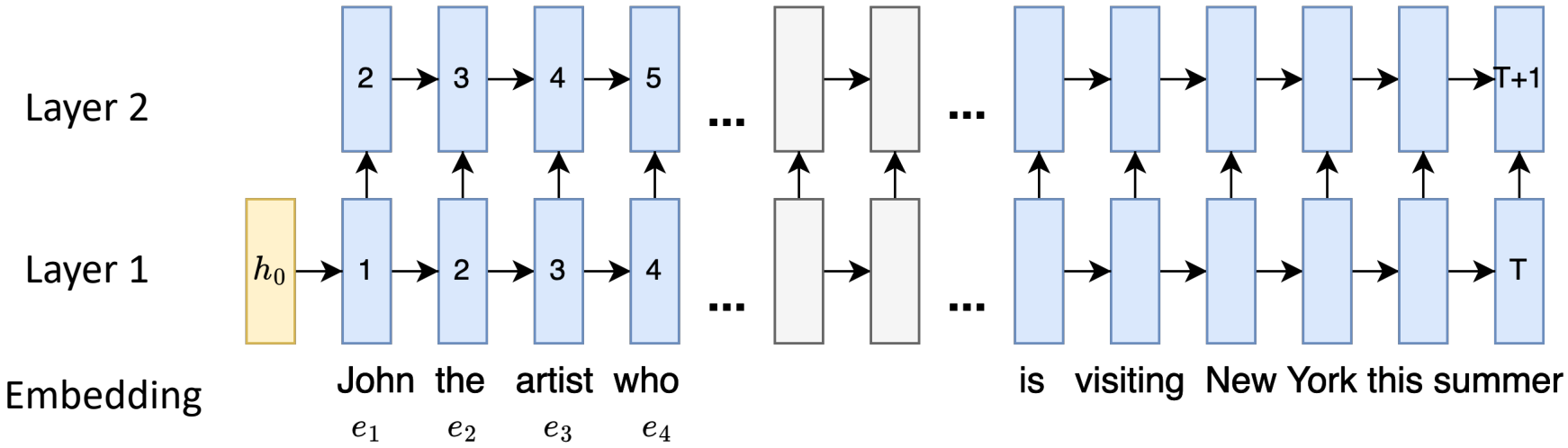
Where is John the artist visiting this summer?
A) Vienna
B) New York

▷ But the RNN may fail to identify New York as the correct answer since the info of "John" has been propagated for almost O(N) (N: sequence length).



Issues with RNNs: No Parallelizability

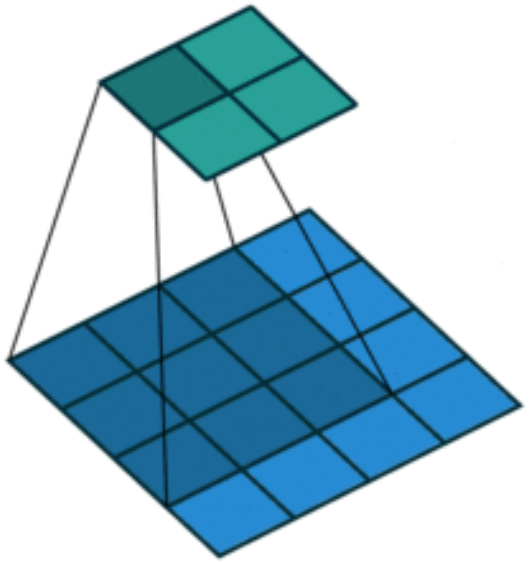
- Past hidden states need to be fully computed before the next hidden state to be ready for computation, which means it is unparallelizable between each timestep. This prevents RNN's use on large corpus or long text.



△ This is a 2-layer, uni-directional RNN. Each blue rectangle is a hidden state at the time-step of the layer. The number indicates the min # of steps required for the hidden state to be computed.

Issues with CNNs: long-term dependencies

Padding = 0, stride = 1



Padding = 0, stride = 2

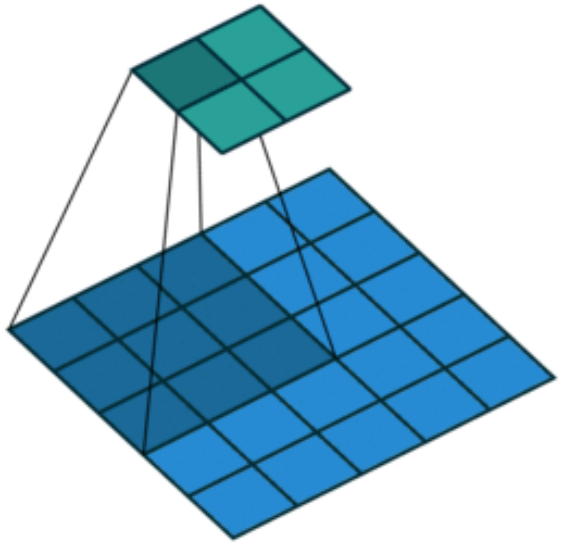
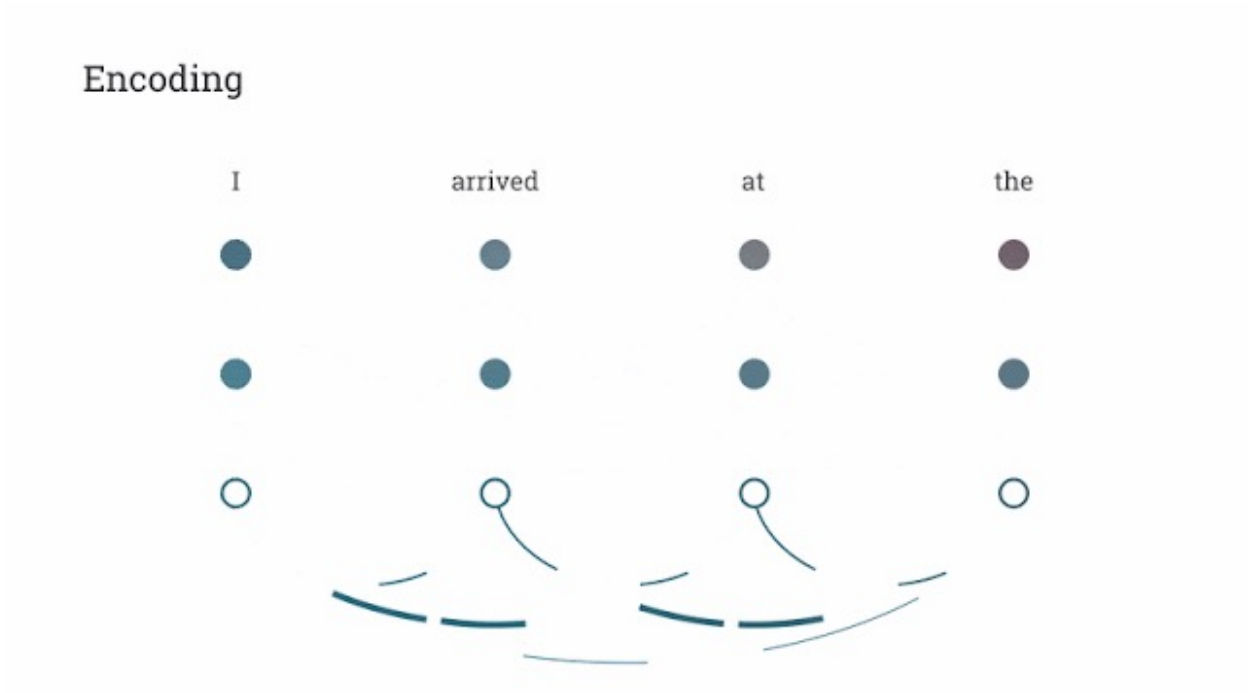


Figure source: https://hannibunny.github.io/mlbook/neuralnetworks/convolutionDemos.html

Transformers

- Attention Is All You Need (Vaswani et al., NeurIPS 2017)

Self-attention : 句子內的每個 token 自己跟自己做attention



Source: An example of the self-attention mechanism following long-distance... | Download Scientific Diagram (researchgate.net)

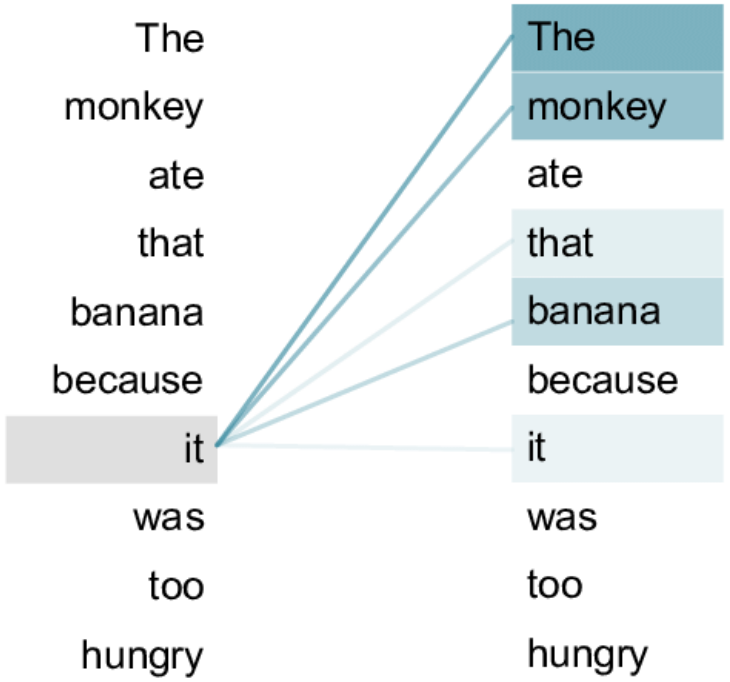


Figure source: https://research.google/blog/transformer-a-novel-neural-network-architecture-for-language-understanding/

Attention Is All You Need

Essential AI

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Anthropic

Niki Parmar* Google Research nikip@google.com

Inceptive

Jakob Uszkoreit* Google Research usz@google.com

Cohere

Sakana AI

Llion Jones* Google Research llion@google.com

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

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

OpenAI

NEAR

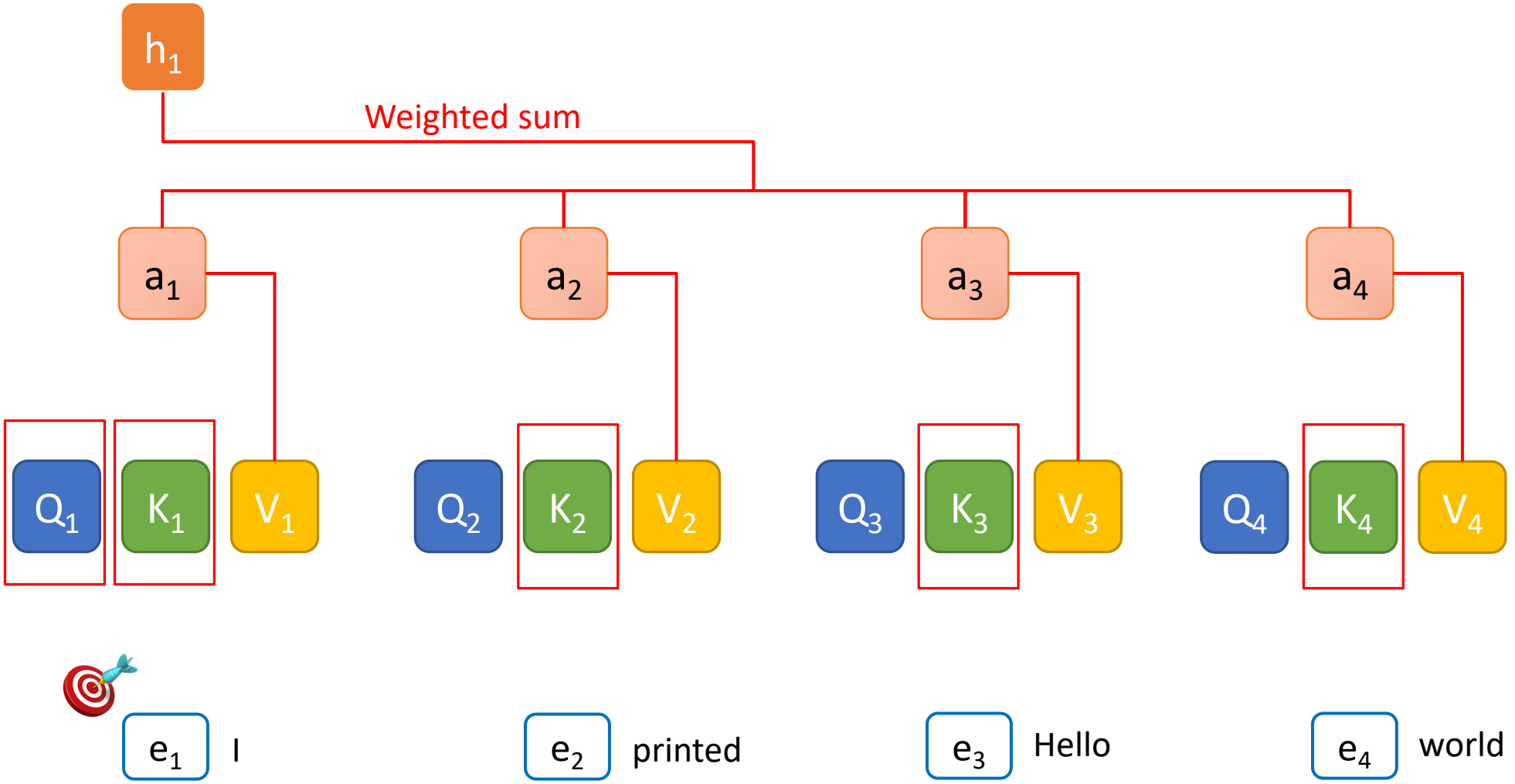
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

https://arxiv.org/abs/1706.03762v6

Vanilla Transformers

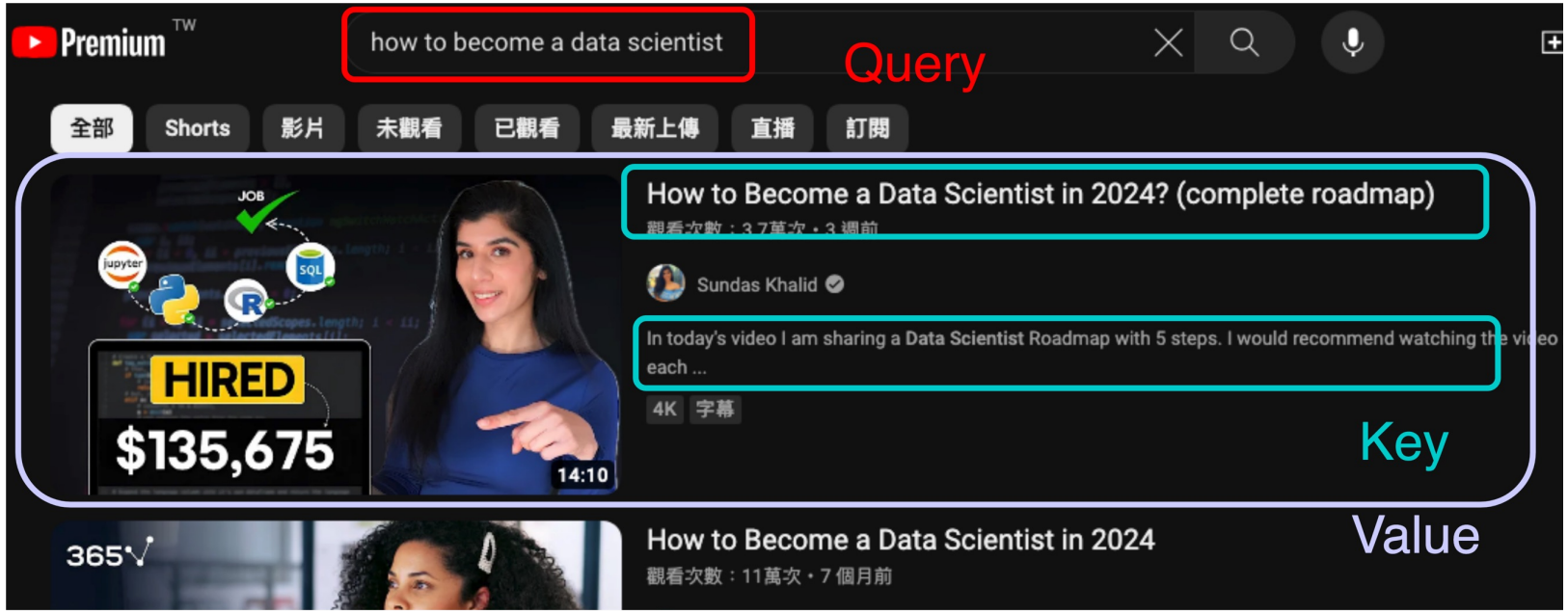
- The vanilla transformers (Vaswani et al., 2017) are also an encoder-decoder model!
- Self-attention exists in both the encoder and decoder parts.

Query (Q), Key (K), and Value (V)



The Concept of Query (Q), Key (K), and Value (V)

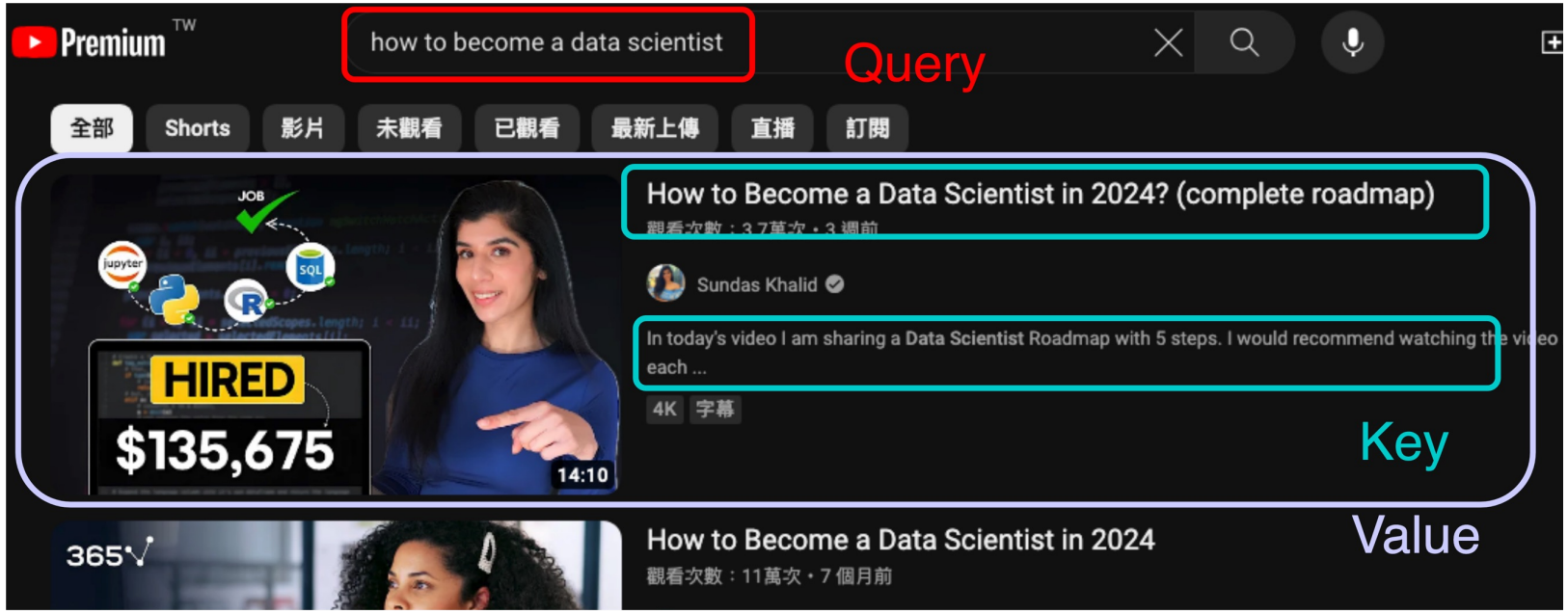
Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).



- The self-attention proposed in Vaswani et al. 2017 is query-key-value attention (QKV attention).
- What is query-key-value attention?

The Concept of Query (Q), Key (K), and Value (V)

Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).



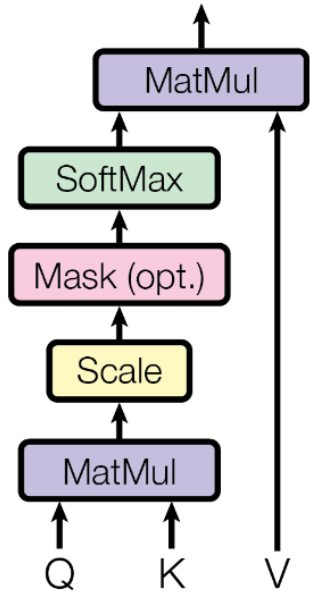
- The self-attention proposed in Vaswani et al. 2017 is query-key-value attention (QKV attention).
- What is query-key-value attention?
  - The whole process is analogous to a retrieval system.
  - We query the search engine (e.g., YouTube), which tries to map our query to the keys such as video title and video descriptions in its database, and then return some best matched videos (values).

為什麼 Transformers 可以平行化？

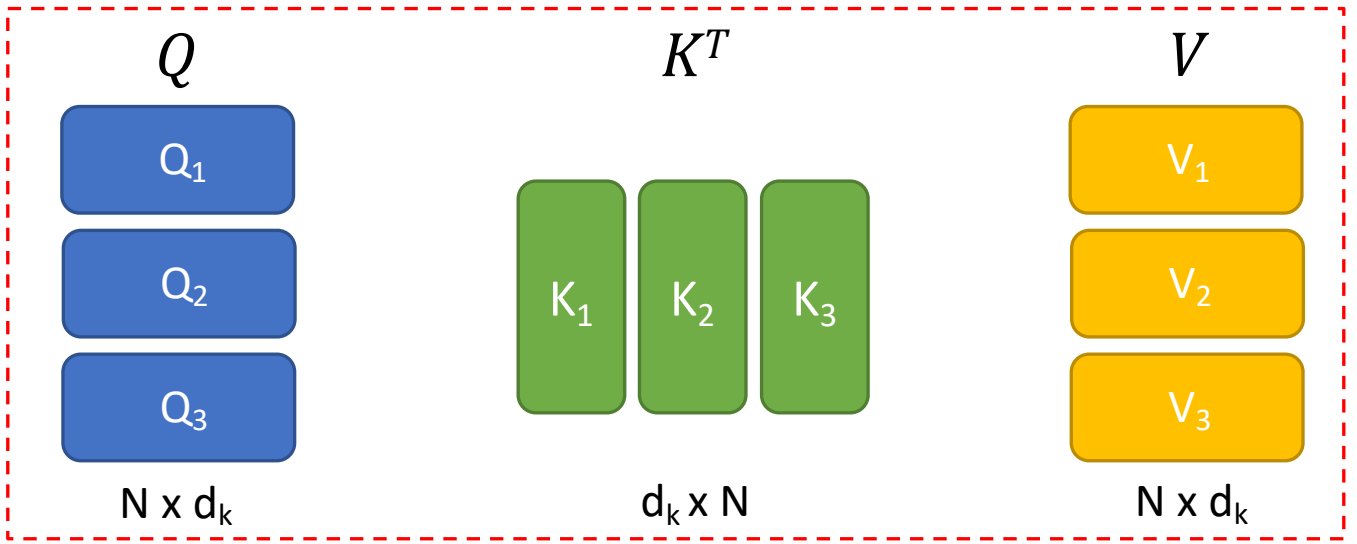
$d_q = d_k = d_v$

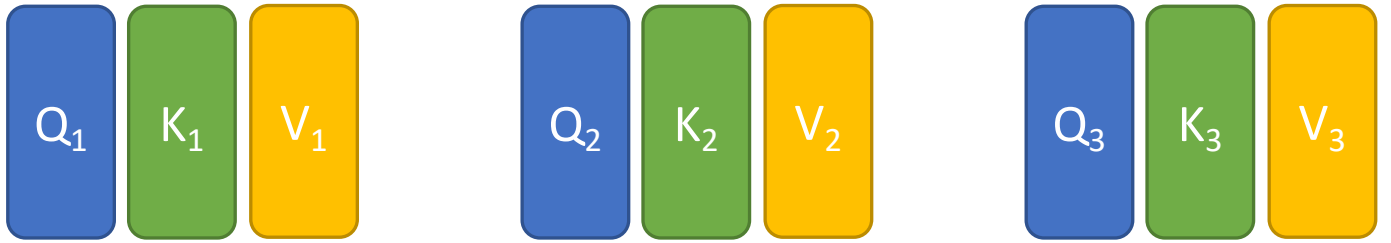
(自注意力機制可以利用矩陣乘積來進行平行化計算)

Scaled Dot-Product Attention



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





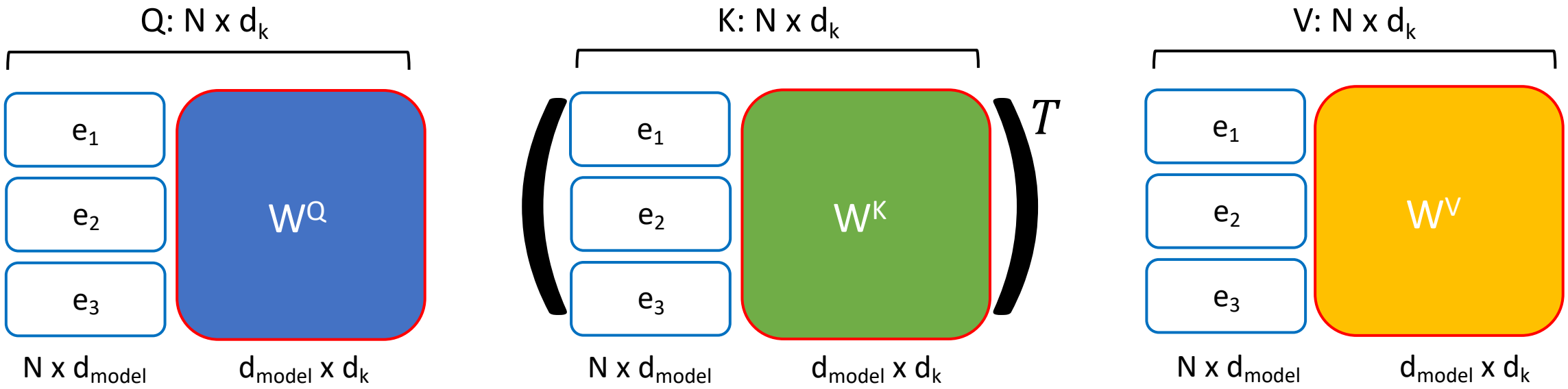
e_1 I

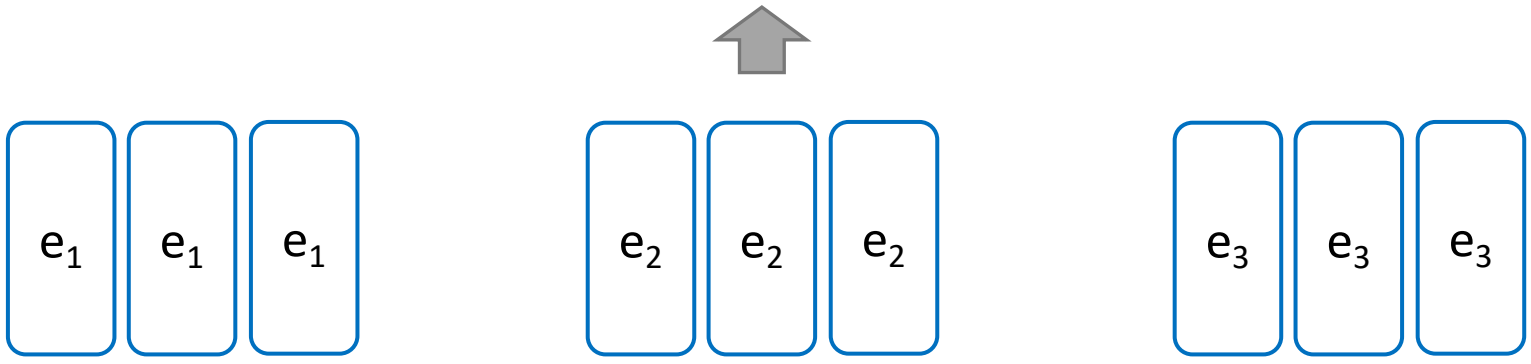
e_2 printed

e_3 Hello

Transformers 的權重值 (1/2)

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

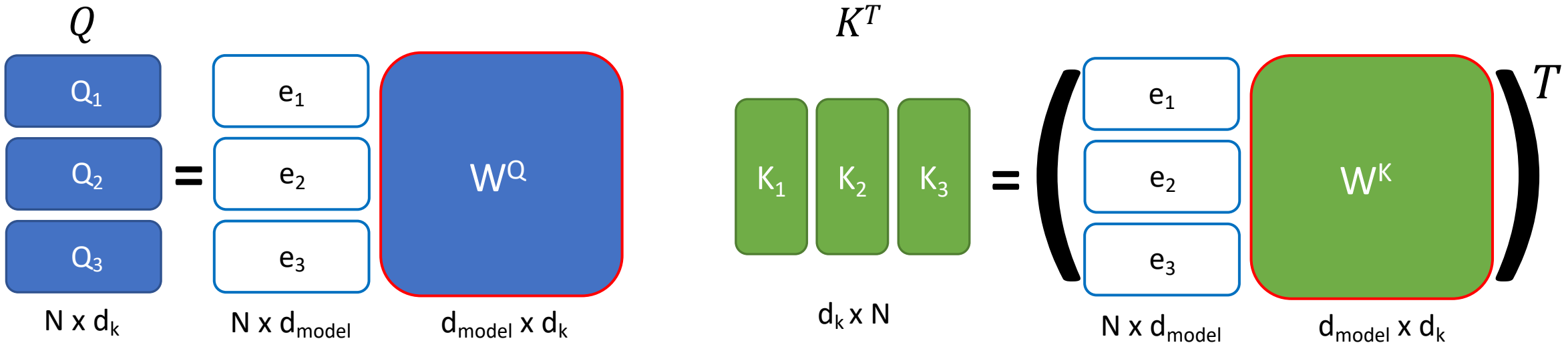


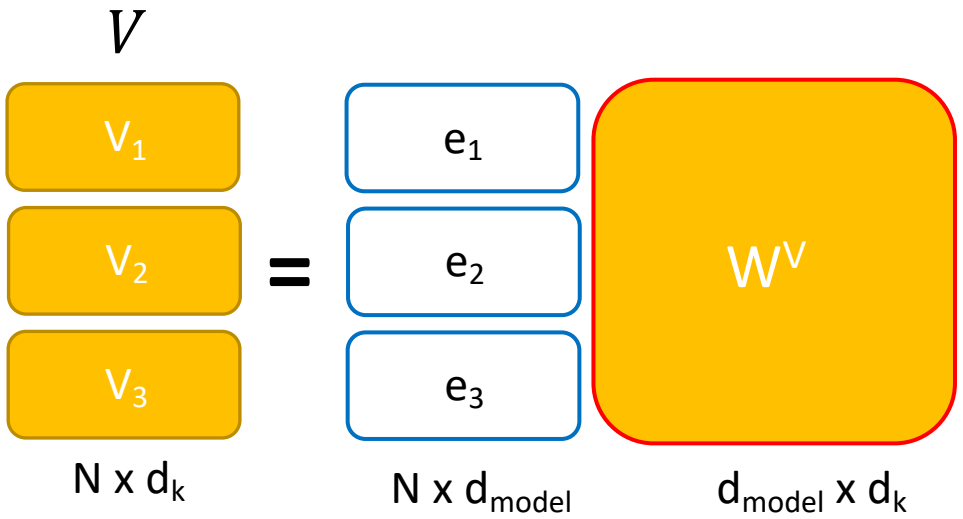


d_{model} : Embedding 的維度大小

Transformers 的權重值 (2/2)

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

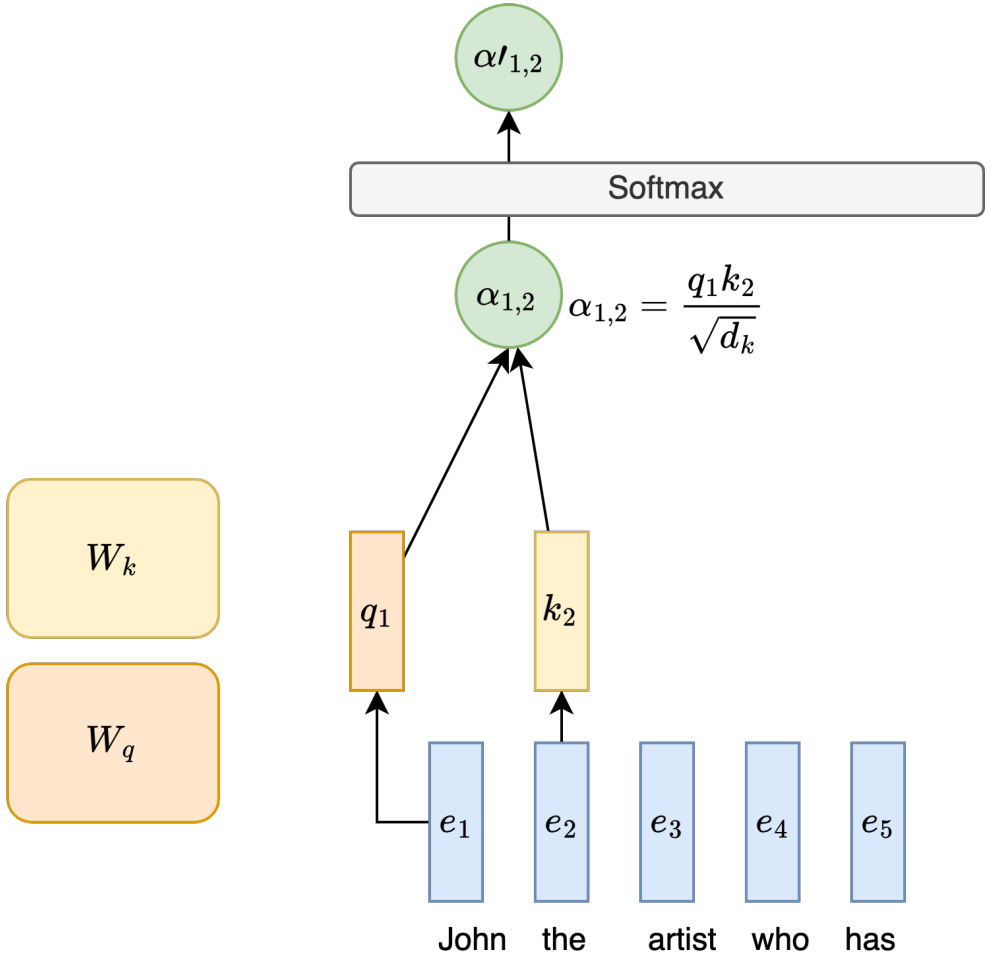




Self-Attention

Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

- Let $e_1, .., e_N; e_i \in \mathbb{R}^d$ be input embeddings of the text sequence.
- Initialize 3 matrices W_q, W_k, W_v.
  - The matrices are used to project the input from $e_1, .., e_N$ to $q_1, .., q_N$, $k_1, .., k_N$ and $v_1, .., v_N$ respectively.
- Do a scaled dot product to get a scalar $\alpha_{1,2}$, meaning "attention score from word 1 to 2."

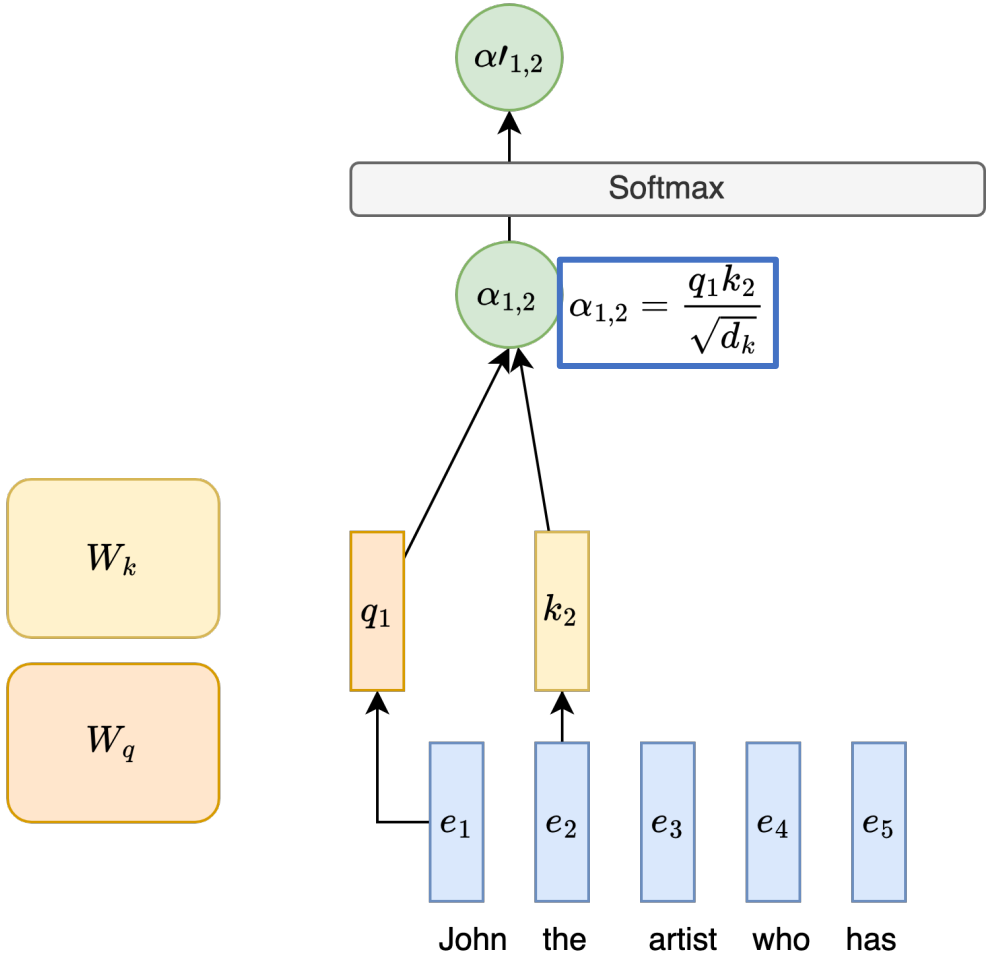


Attention Score

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

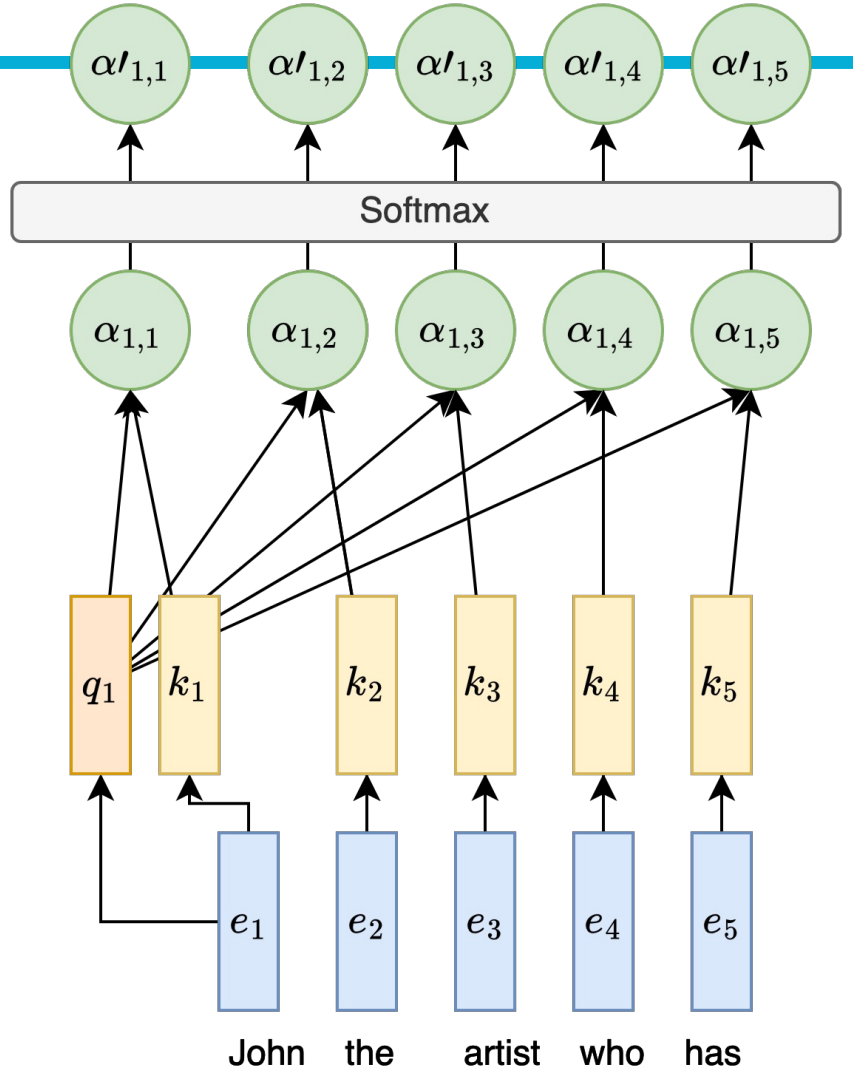
Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

- The original Transformer uses scaled dot product (there are many others).
  - $score(e_i, e_j) = \frac{q_i k_j}{\sqrt{d_k}}$
  - d_k ($= \text{dim of keys} = \text{dim of queries}$)
  - Why Scaling?
    - Because greater d_k with Softmax() results in vanishing gradients.
    - We can ensure $score(e_i, e_j)$ to be numerically stabler, as long as we scaled by $\sqrt{d_k}$.



Attention Score

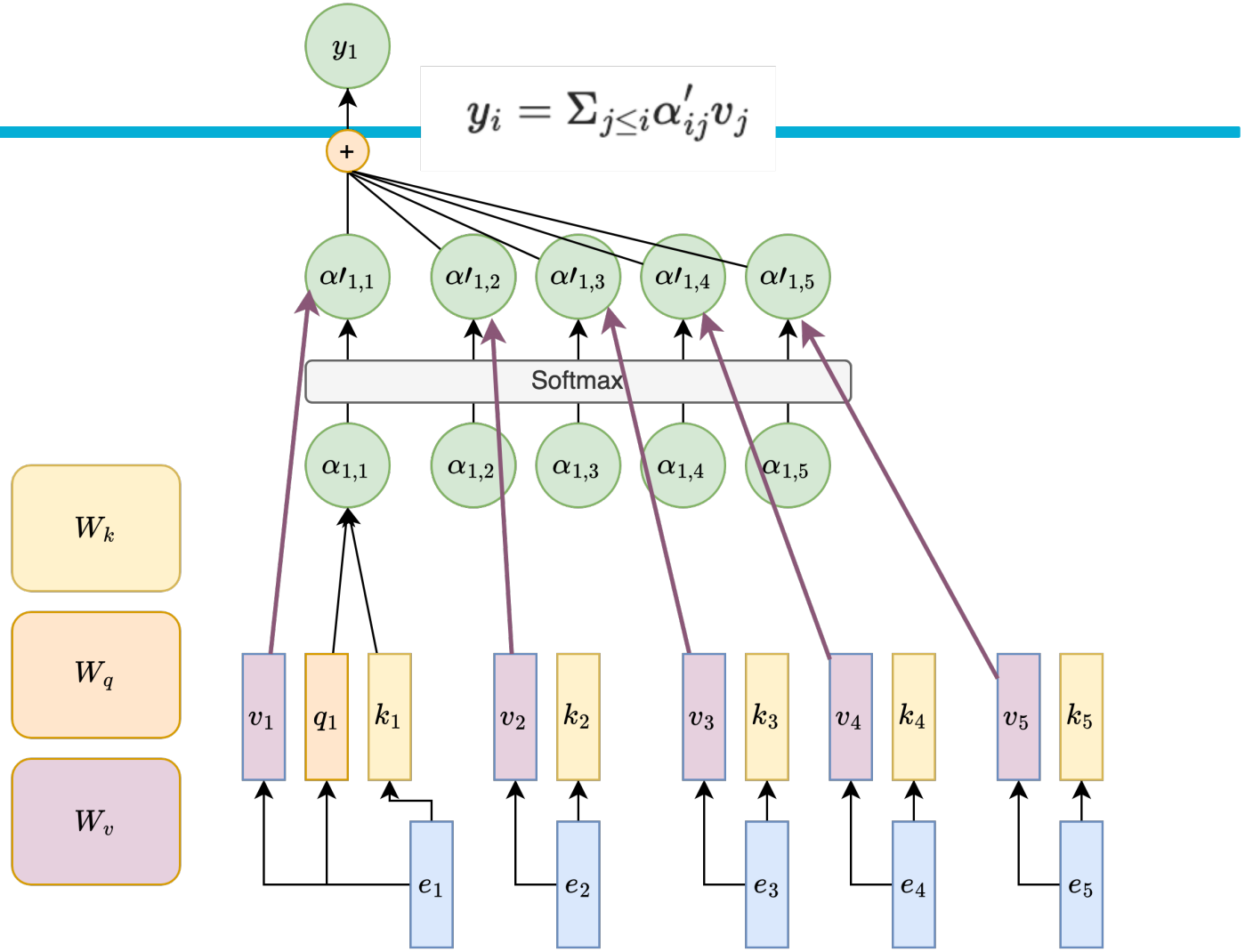
- For every pair of i, j, We calculate the scores. The figure illustrates q = 1.



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

Self-Attention

- The figure illustrates when q = 1.
- The → indicates scalar vector multiplication.



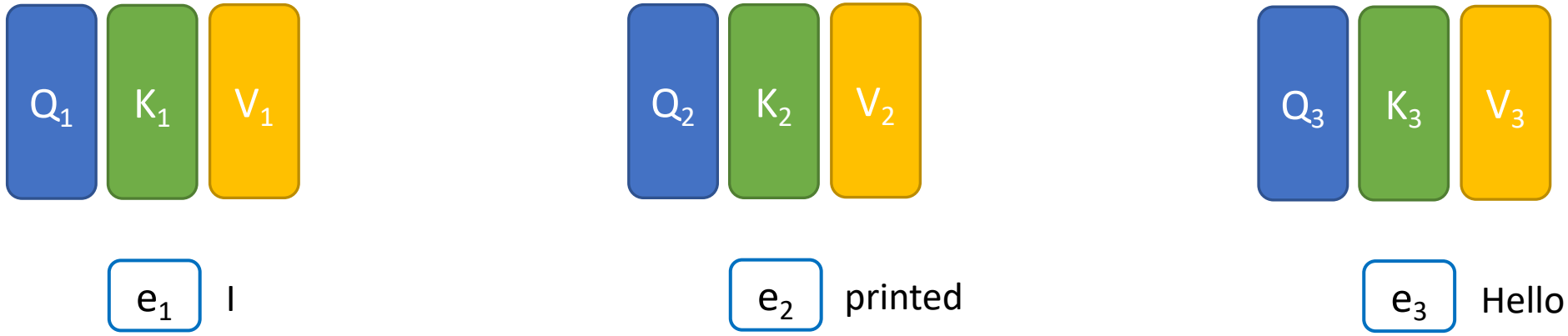
Multi-Head Attention (MHA)

- In p. 12, we describe self-attention as QKV attention, where
  - we query the search engine
  - search engine tries to map our query to the keys
  - output some best matched videos (values)
- What if we want to have queries focusing on different aspects?
  - For example, one set of queries focusing on semantic similarity, another set focusing on passive/active voice.
  - **We need multiple attention mechanisms, i.e. multiple (attention) heads!**

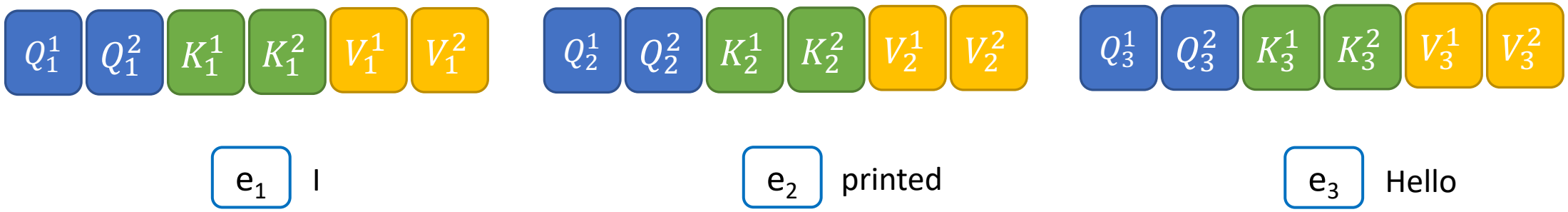
Multi-Head Attention (MHA)

上標: head
下標: token index

Head = 1 (without MHA)

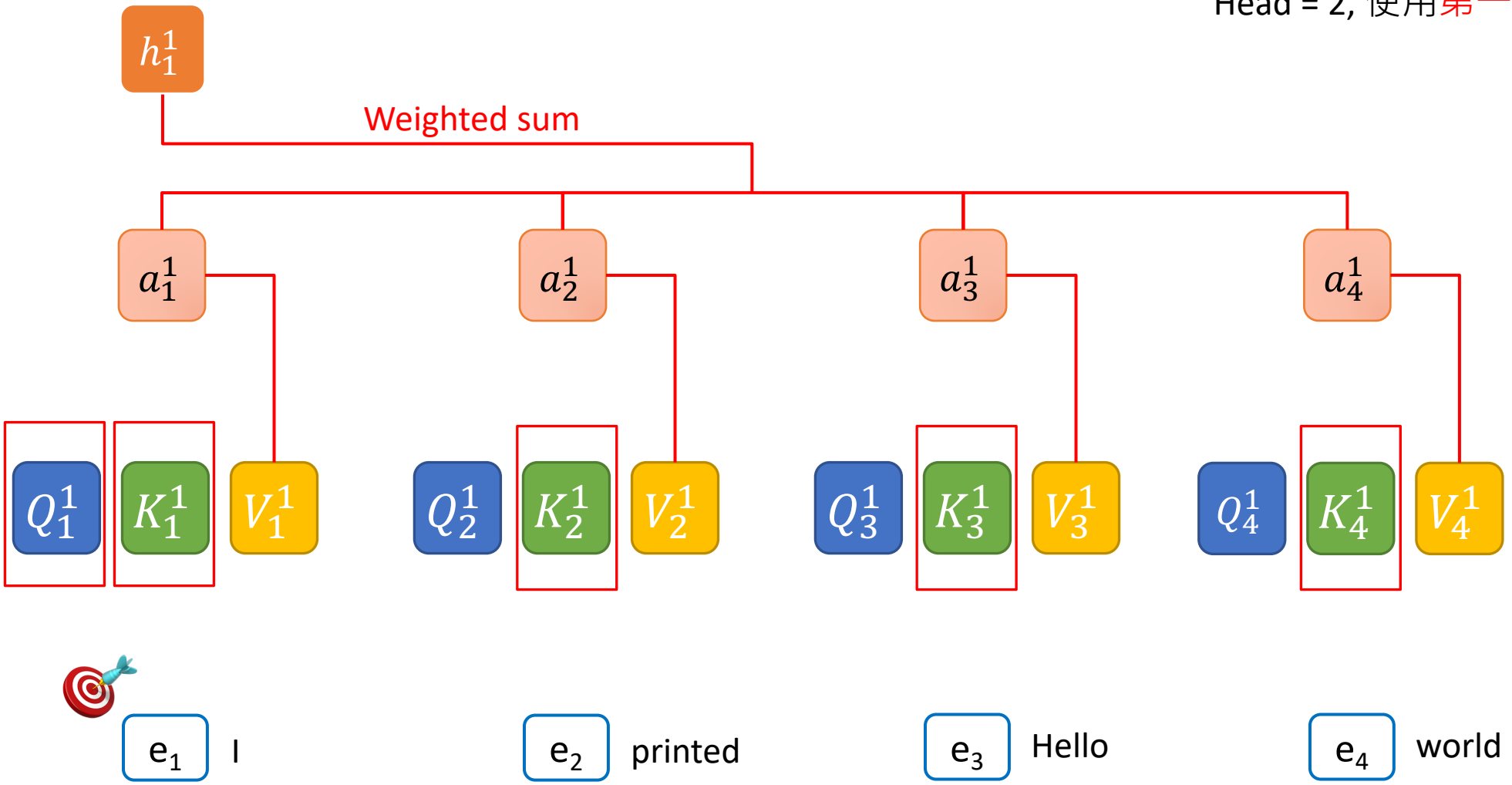


Head = 2 (with MHA)



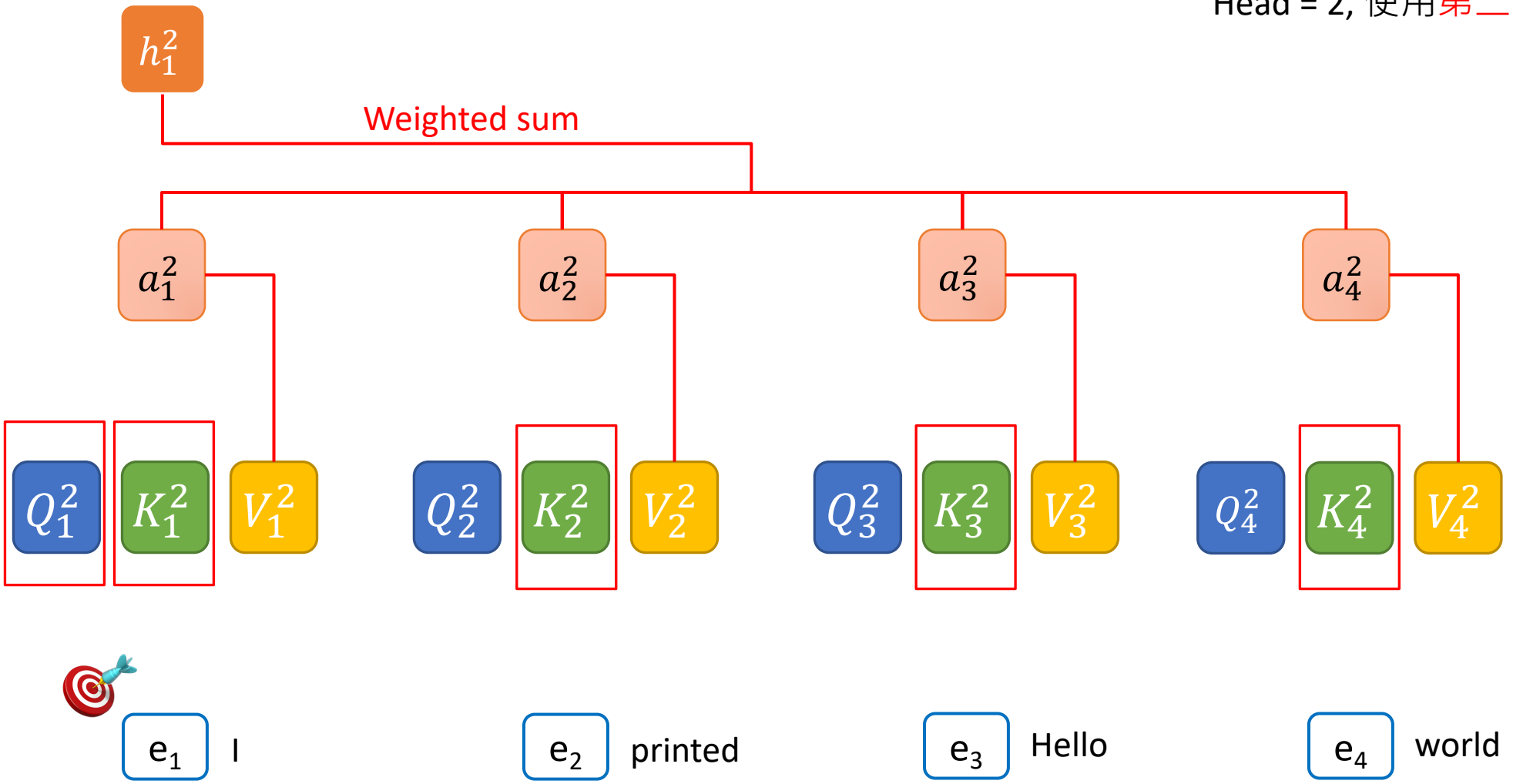
Query (Q), Key (K), and Value (V) with MHA

Head = 2, 使用第一個 head



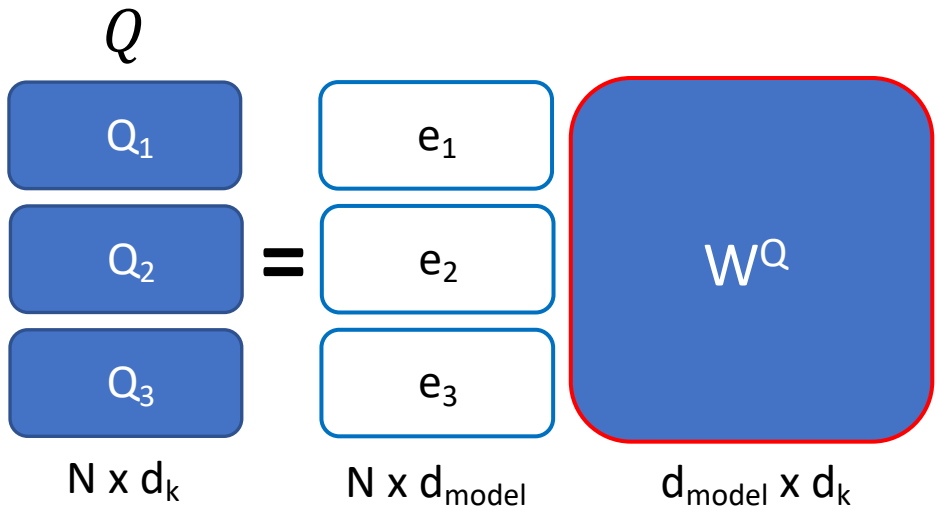
Query (Q), Key (K), and Value (V) with MHA

Head = 2, 使用第二個 head

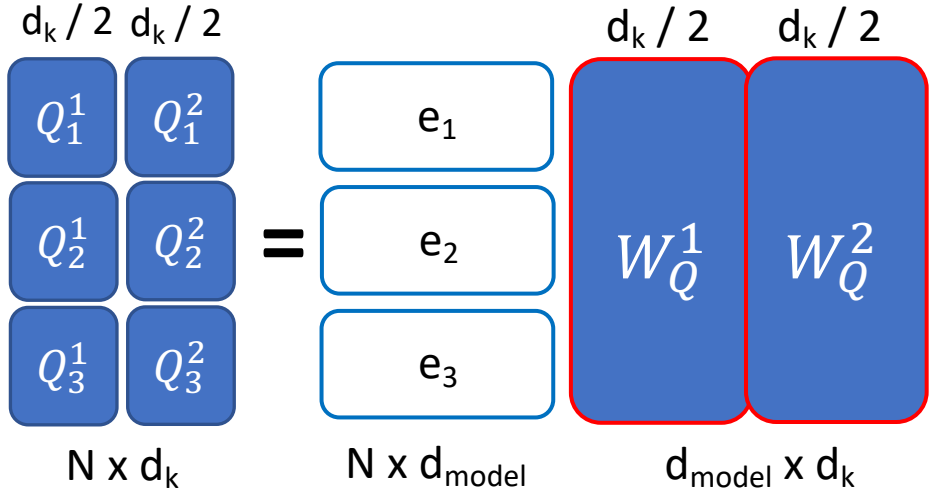


Input Dimensions of MHA

Head = 1 (without MHA)



Head = 2 (with MHA)



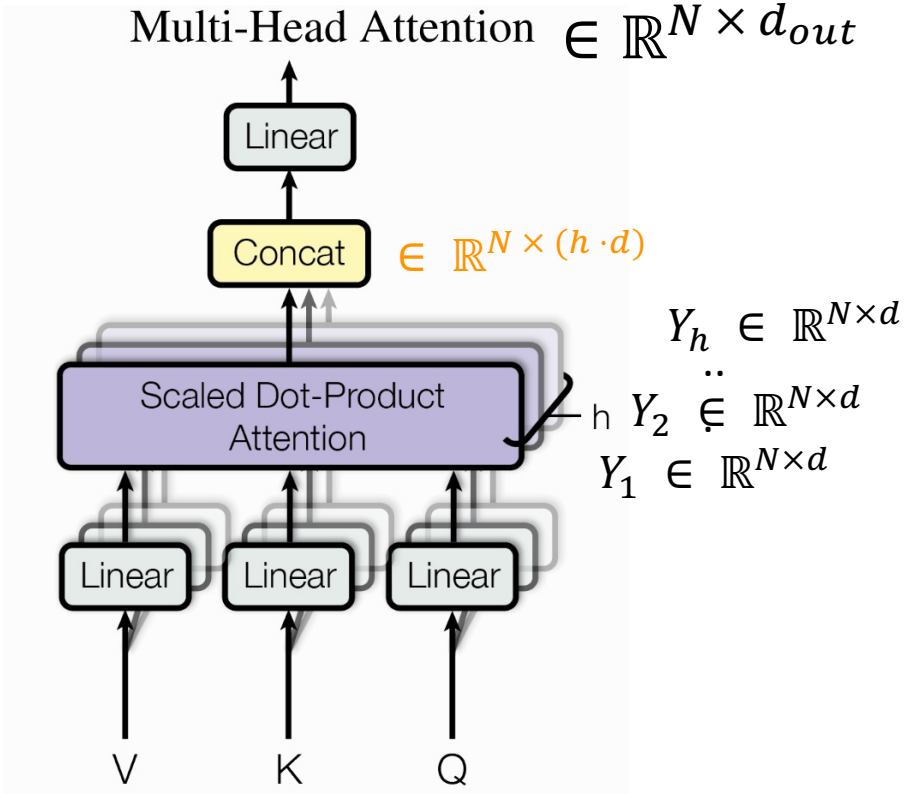
Output Dimensions

$$\text{Multihead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h) W^O$$

$\in \mathbb{R}^{N \times (h \cdot d)}$ $\in \mathbb{R}^{(h \cdot d) \times d_{out}}$

Head = 1 (without MHA)

Head = 2 (with MHA)



MHA 比較好嗎？

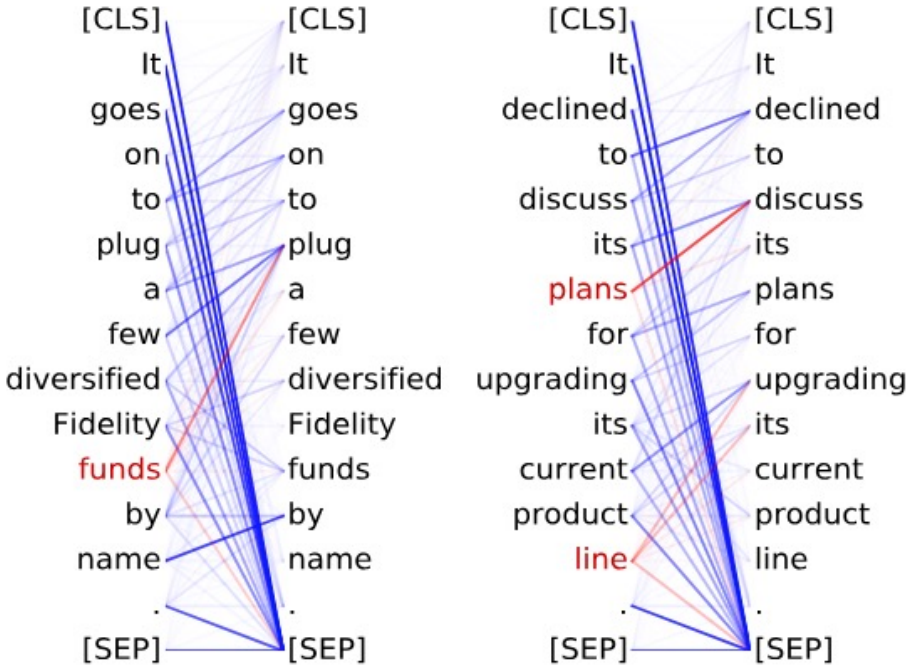
Clark, Kevin, et al. "What Does BERT Look at? An Analysis of BERT's Attention." Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP. 2019.

實驗證明：不同 head 關注的地方不太一樣

<layer>-<head number>

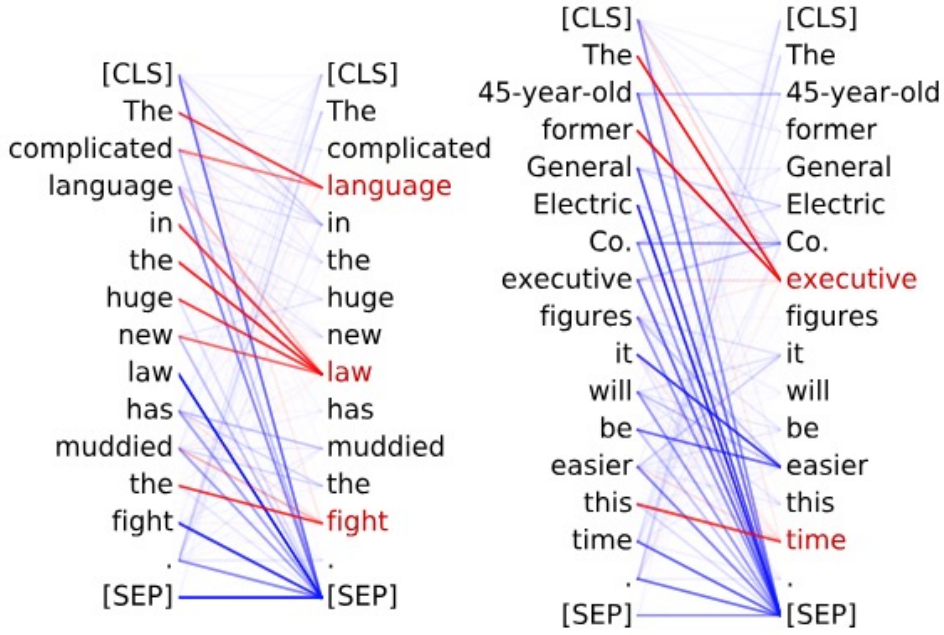
Head 8-10 關注受詞

- **Direct objects** attend to their verbs
- 86.8% accuracy at the `dobj` relation



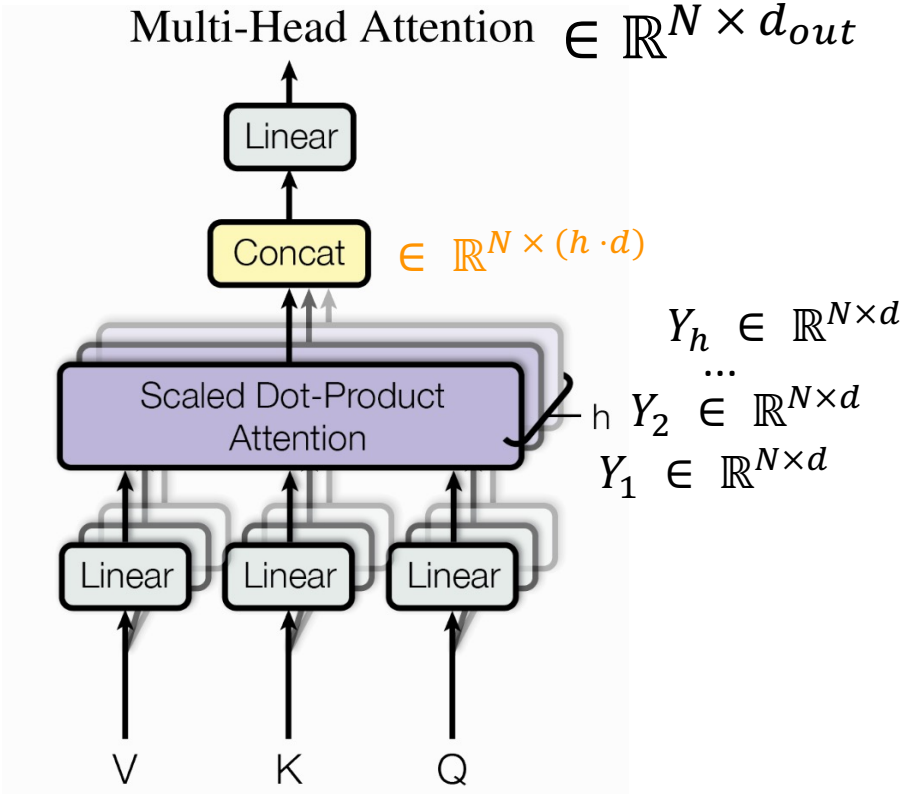
Head 8-11 關注限定詞

- **Noun modifiers** (e.g., determiners) attend to their noun
- 94.3% accuracy at the `det` relation

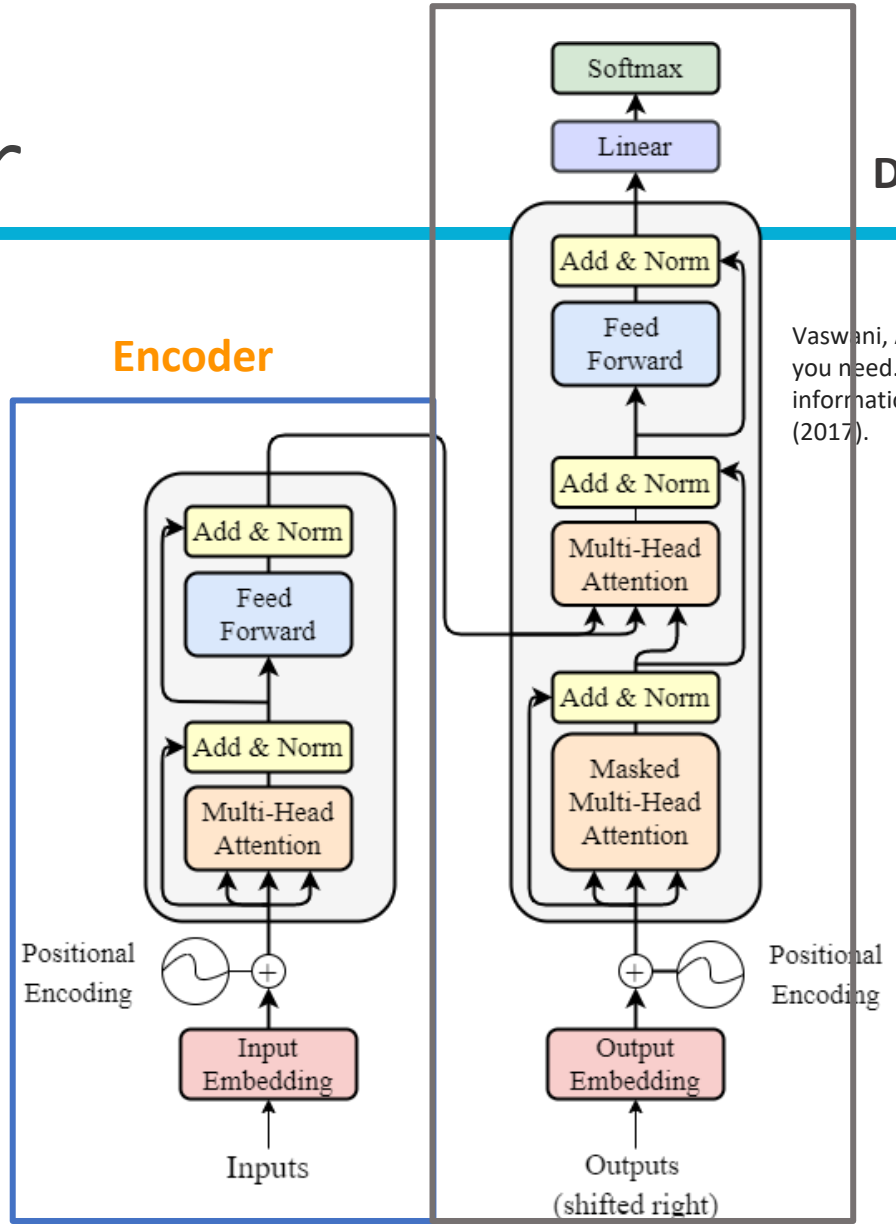


Summary of MHA

- With W^O, the concatenated head outputs can be projected to a preferred dimension (hyperparameter: d_{out}).
- No worries on how #head changes dimension of outputs!



Transformer Encoder-Decoder



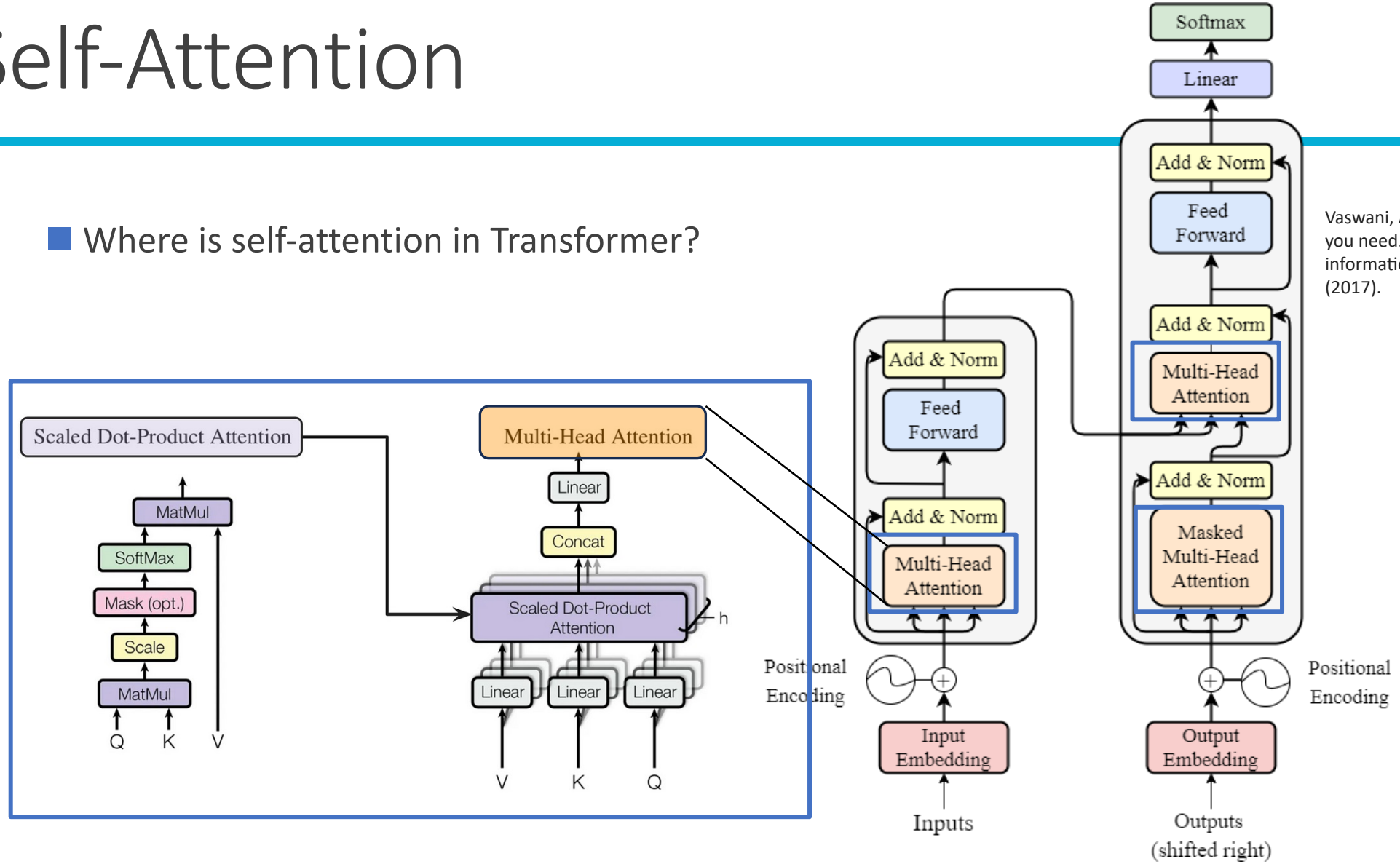
Encoder

Decoder

Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

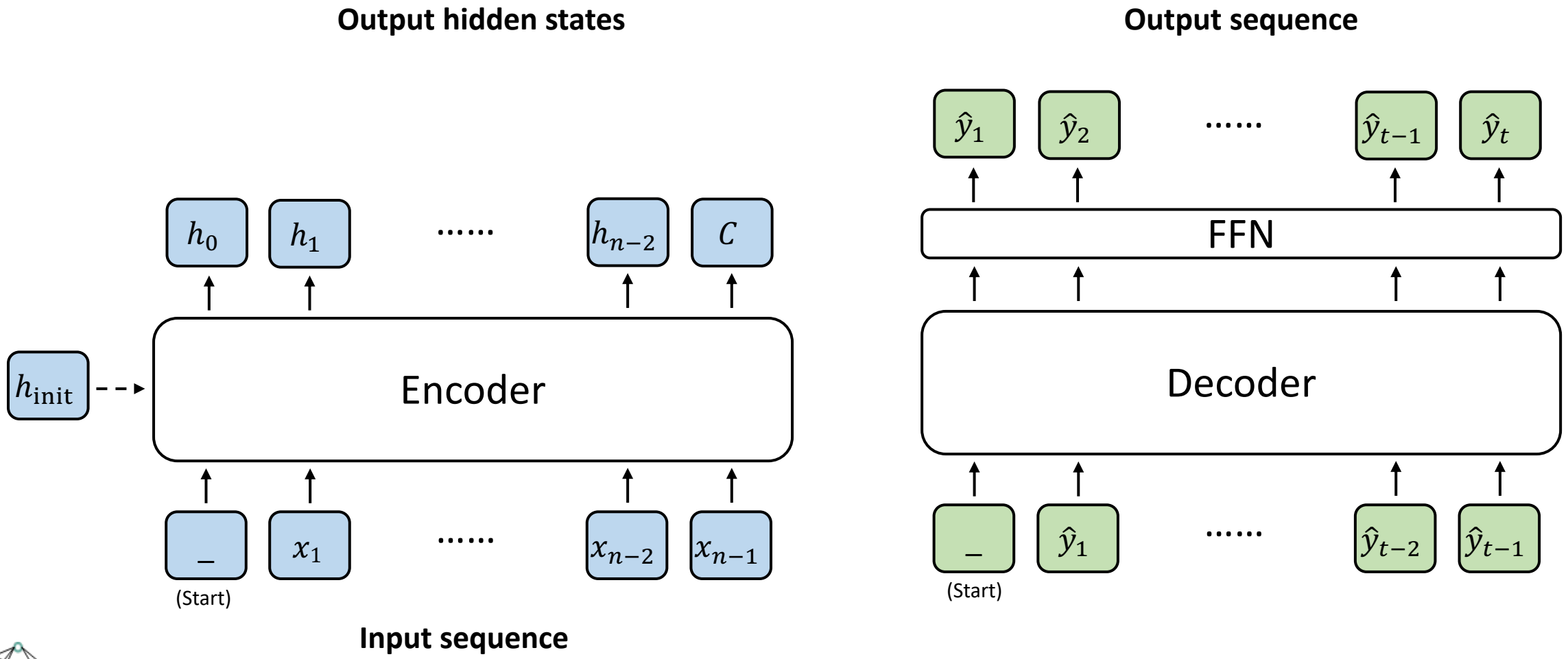
Self-Attention

- Where is self-attention in Transformer?



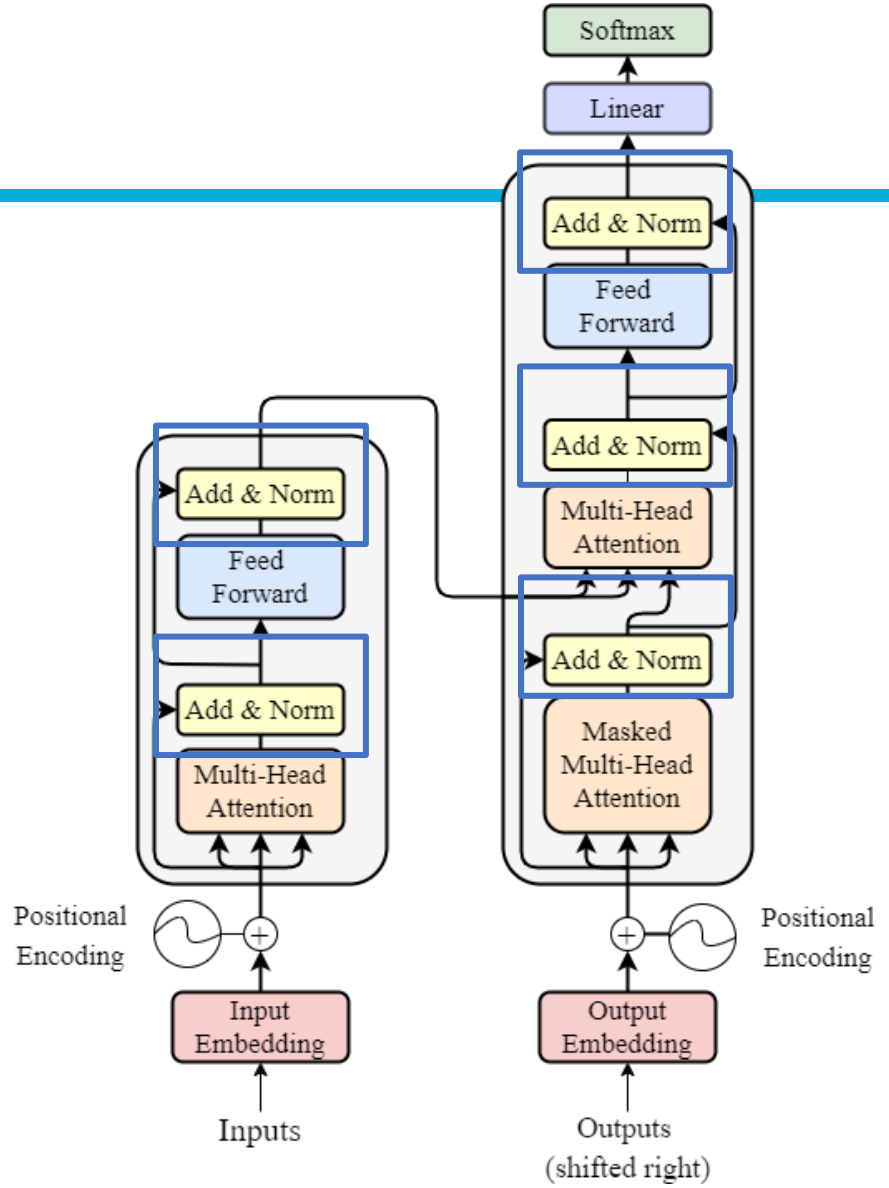
Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Encoder and Decoder



Add & Norm

- Add: Residual Connection
- Norm: Layer Normalization



Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Add & Norm

- Research shows that Residual Connection (He, Kaiming, et al., 2016) stabilizes training.
- Let the layers in between learn the residual relationship (i.e. $Y - X$).

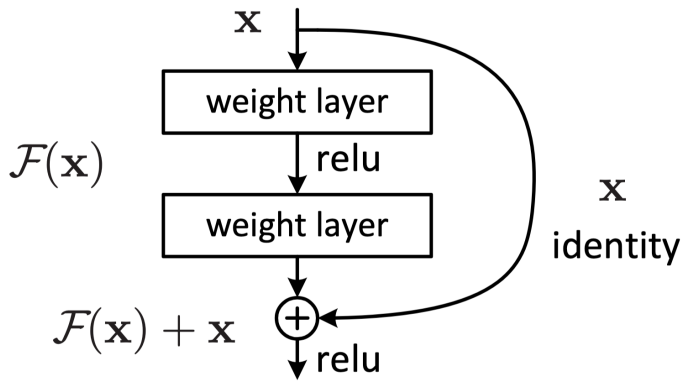
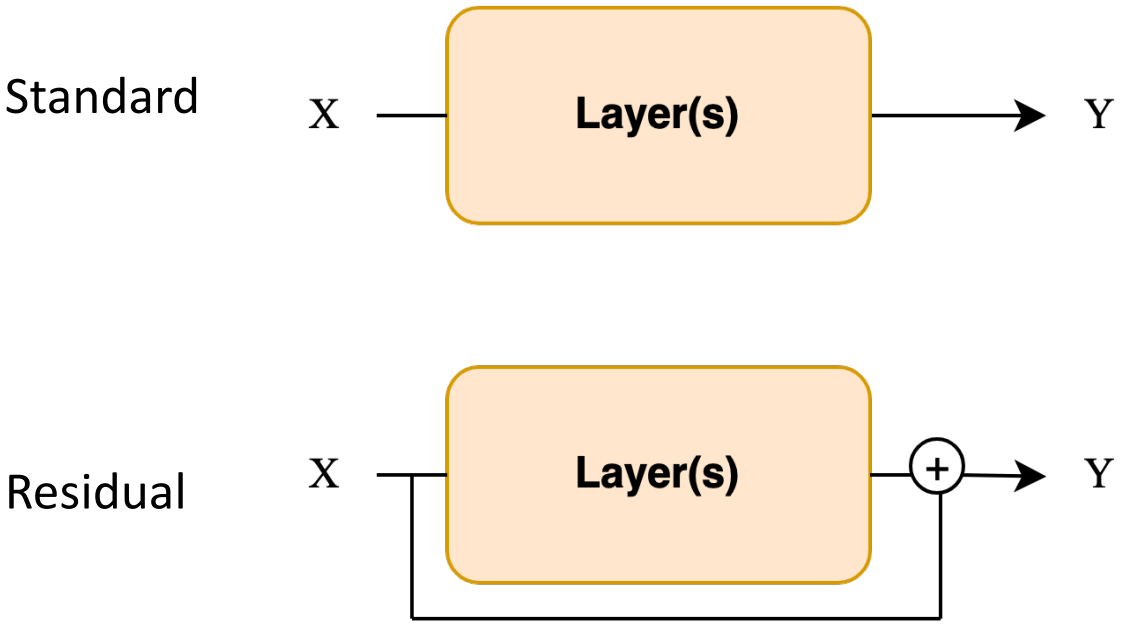


Figure 2. Residual learning: a building block.

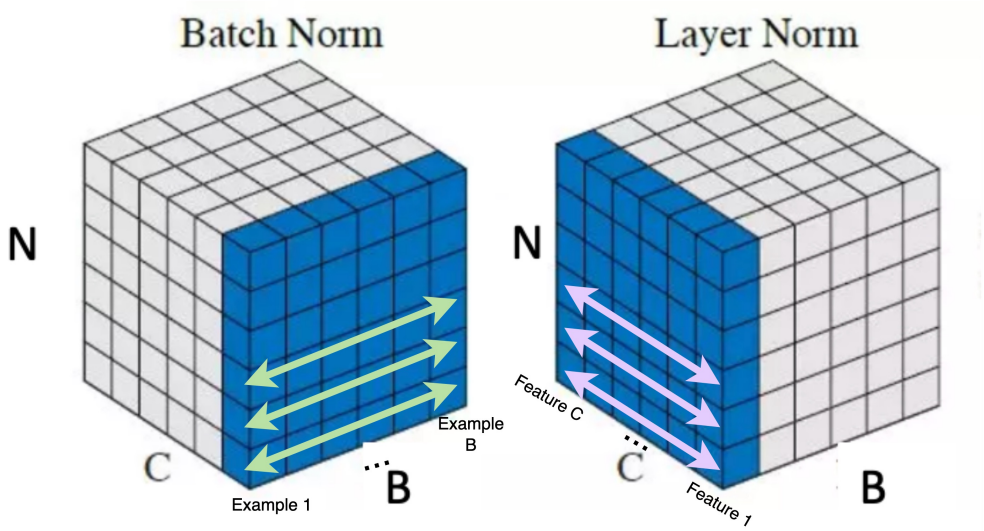


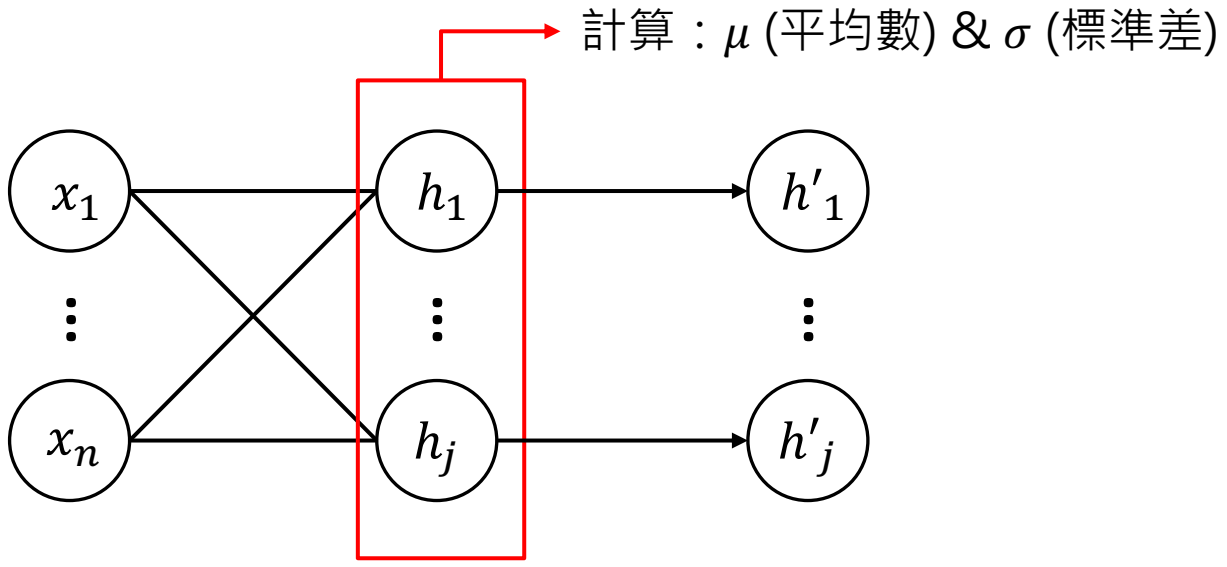
He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition (2016).

Add & Norm

- Layer Norm: For an input vector, calculate its mean and std across embedding dimension (within an example).
  - $\mathbf{norm}(X) = \frac{X-\mu}{\sigma}$

B: Batch Size
C: Channels / Embedding Feature Dimension
N: Sequence length

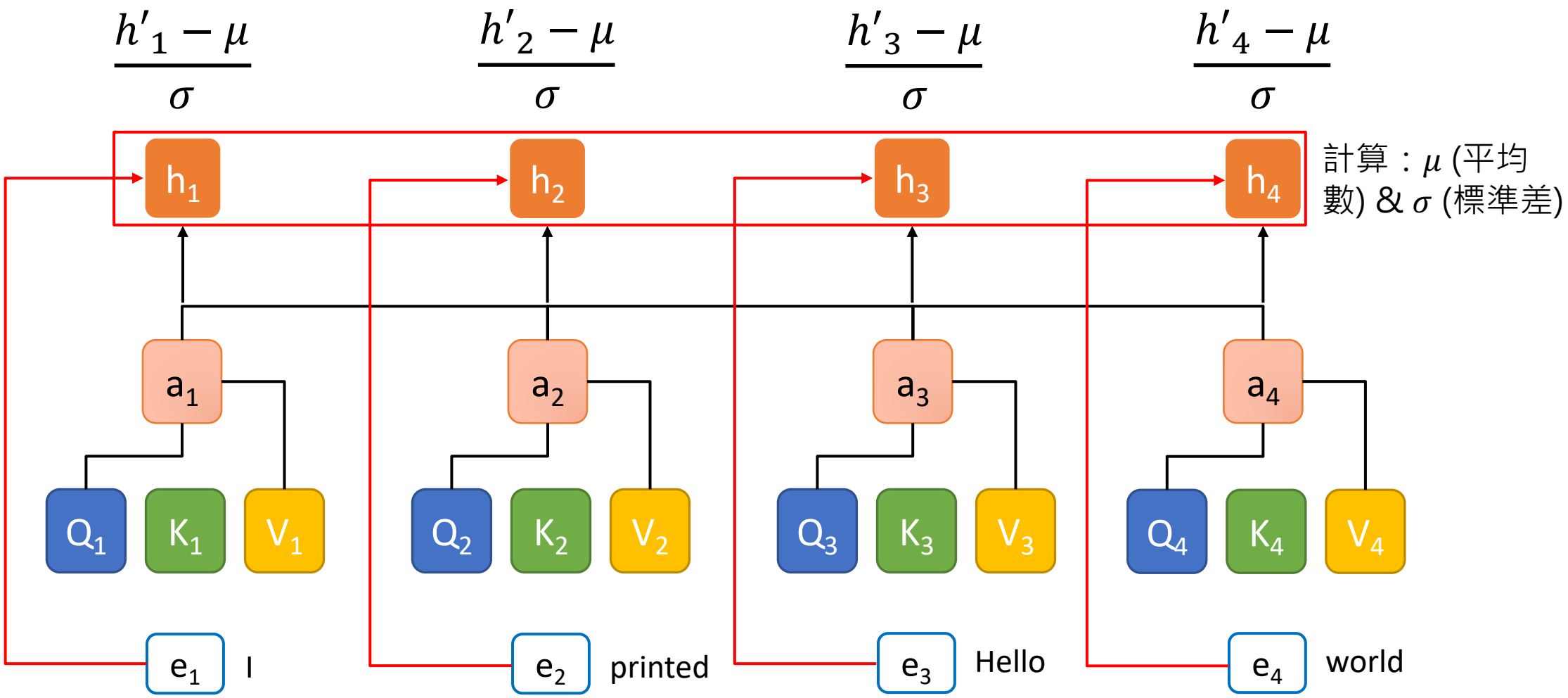




(Batch Normalization)

Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E. Hinton. "Layer normalization." arXiv preprint arXiv:1607.06450 (2016).

Add & Norm



Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E. Hinton. "Layer normalization." arXiv preprint arXiv:1607.06450 (2016).

Why LN not BN?

- 資料長度差距
- 模型實作表現差異

[Intro] Static padding

batch 1

I	love	CGU	[PAD]	[PAD]	[PAD]	
Yesterday	I	watched	a	good	movie	longest
Your	new	shoes	look	nice	[PAD]	

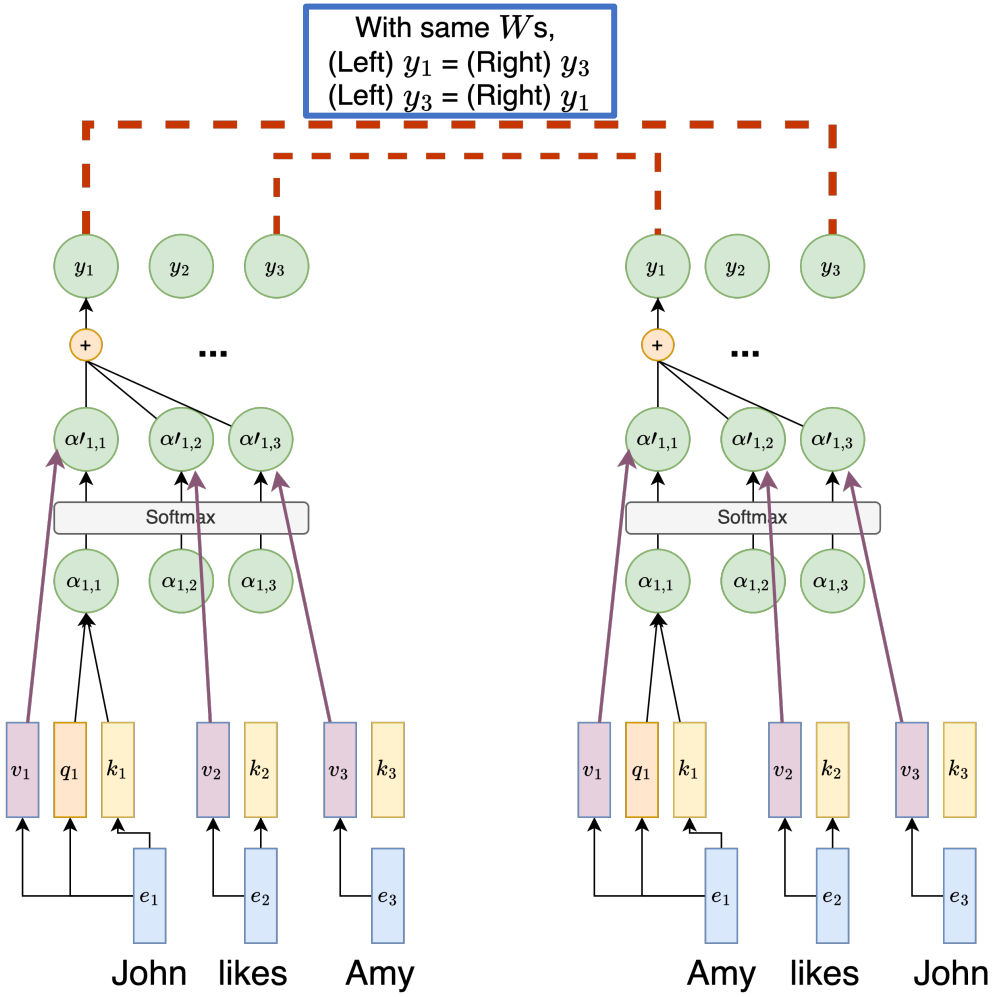
batch 2

Merry	Christmas	[PAD]	[PAD]	[PAD]	[PAD]
I	love	coding	[PAD]	[PAD]	[PAD]
Me	[PAD]	[PAD]	[PAD]	[PAD]	[PAD]

Lack of Position Modeling

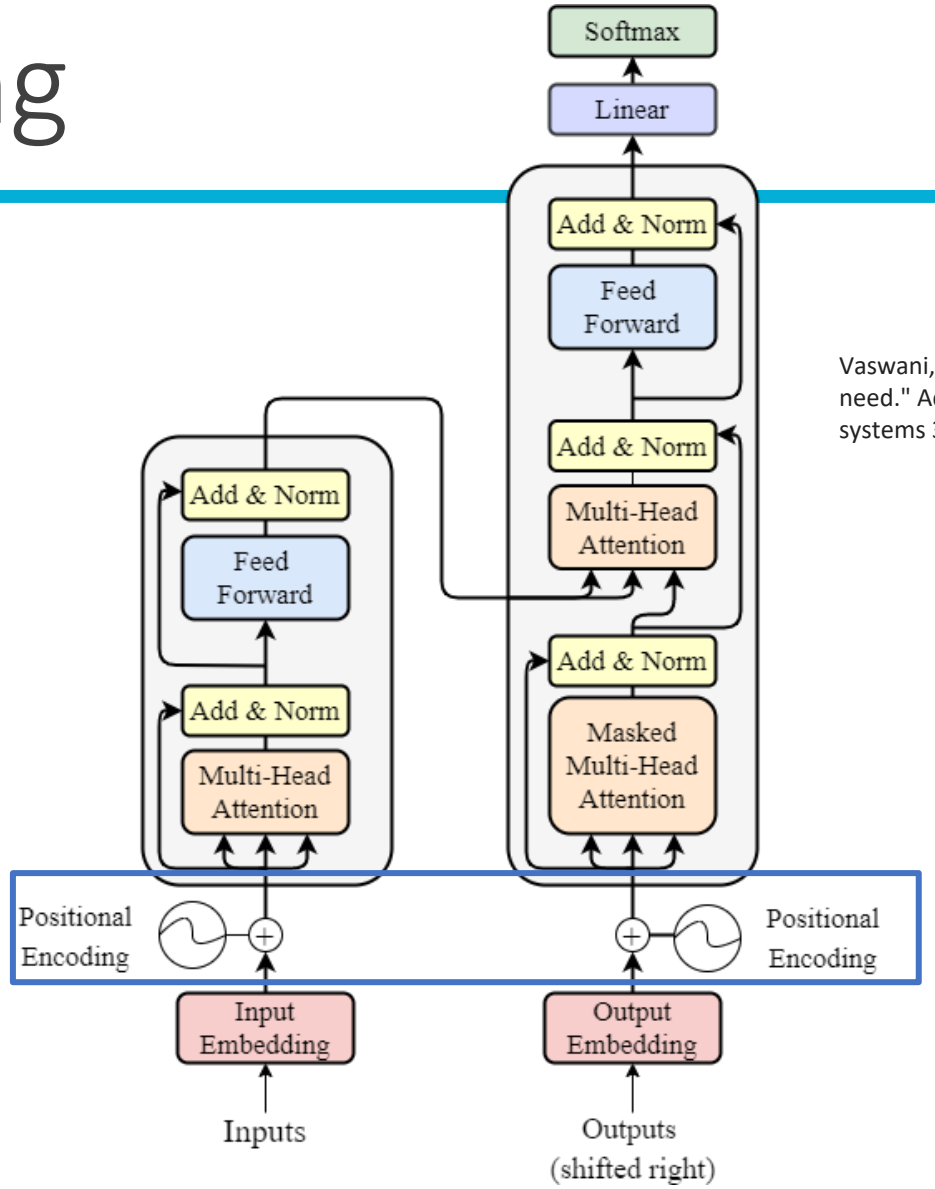
- RNN processes sequences step-by-step.
- However, in Transformers, we have no way to know the relative distance of one word to another!
  - The same word, eg. “Amy”, in the 2 sentences are encoded to the same y (with the same weight matrices).

Problematic because these 2 sentences convey different meanings, but same words are encoded identically!



Positional Encoding

- Where is the Positional Encoding inside the transformer?
- In brief, it is a way to tell the model the position of each token in a sequence.



Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Positional Encoding Equations

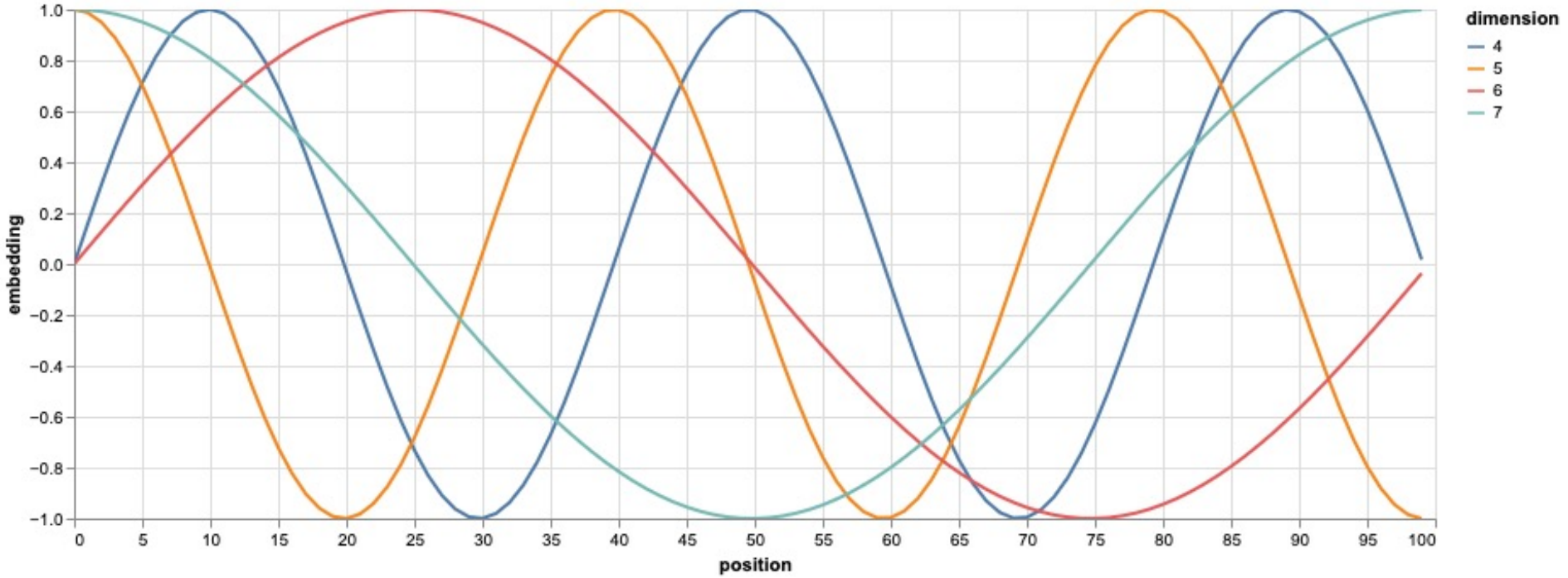
- 創造出相對位置的 embeddings
- 核心精神：利用三角函數的週期性來建構序列資訊
  - 偶數的 embedding 單元 (2i , i = 0, 1, …, d_{model})：Sin 函數
  - 奇數的 embedding 單元 (2i+1, i = 0, 1, …, d_{model})： Cosine 函數

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

輸入序列中任一token的位置

Figure source: https://nlp.seas.harvard.edu/annotated-transformer/#positional-encoding

Positional Encoding Visualization

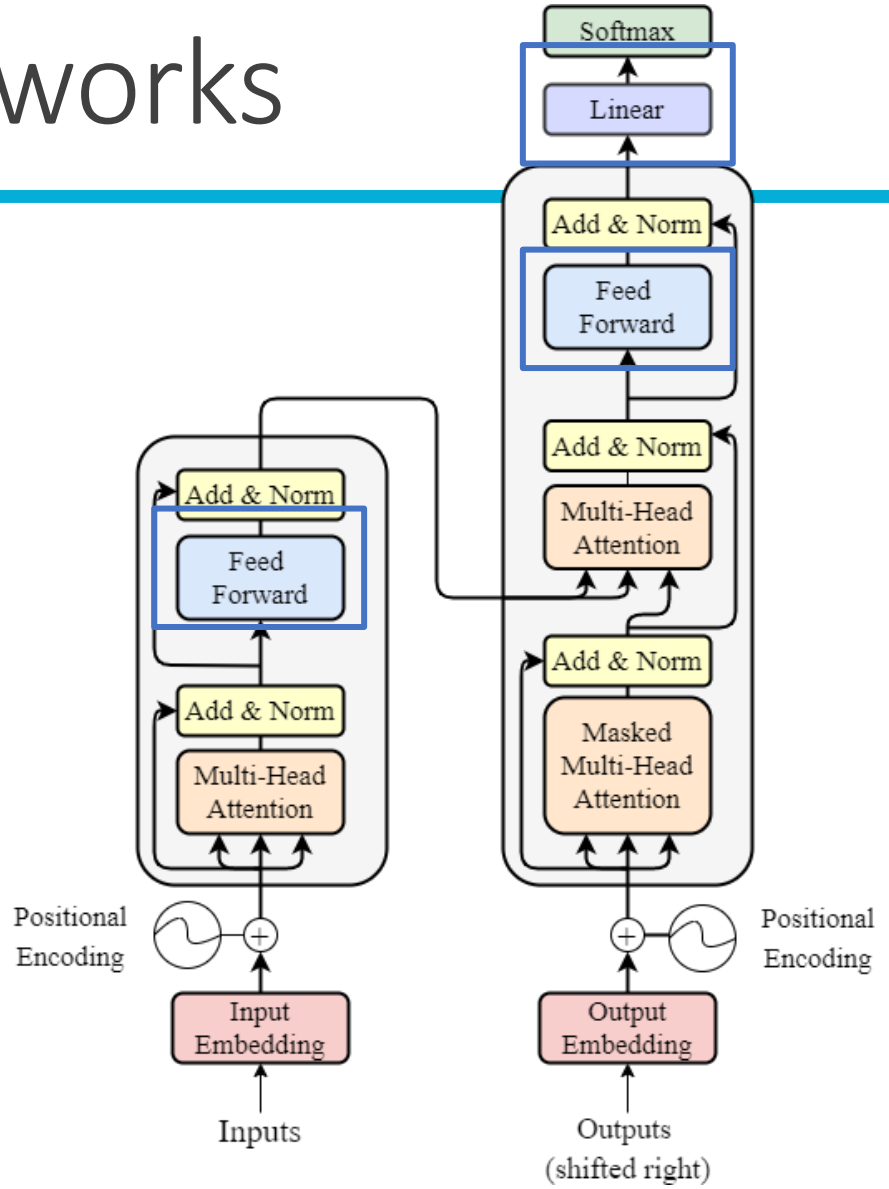


- Question: 為什麼需要同時有 Sin 和 Cosine 的函數？
- Ans: 藉由不同週期性來捕捉序列中各個位置的關係

Figure source: https://nlp.seas.harvard.edu/annotated-transformer/#positional-encoding

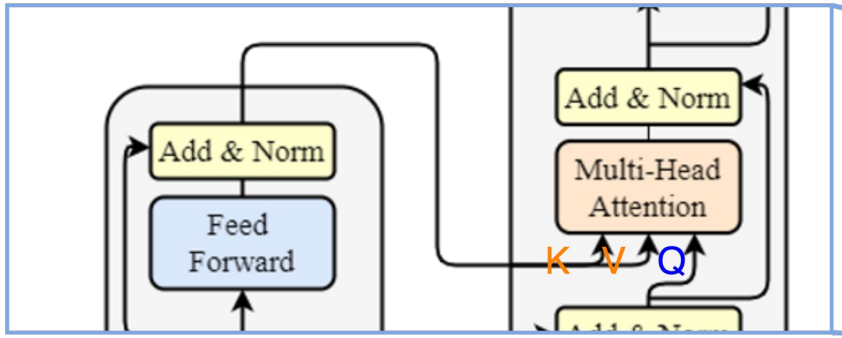
Feed Forward Networks

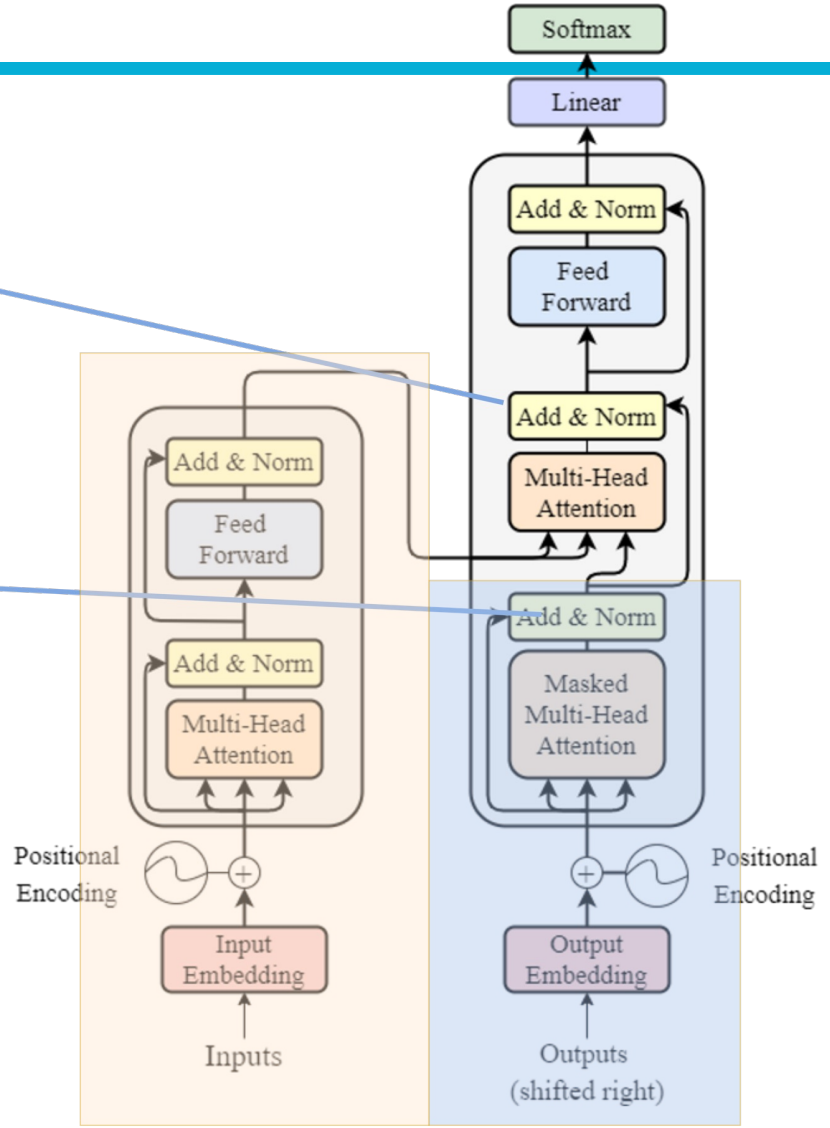
- Simply multiply by a weight matrix.
- Used to project to a specific dimension.



Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Decoder: Cross-Attention





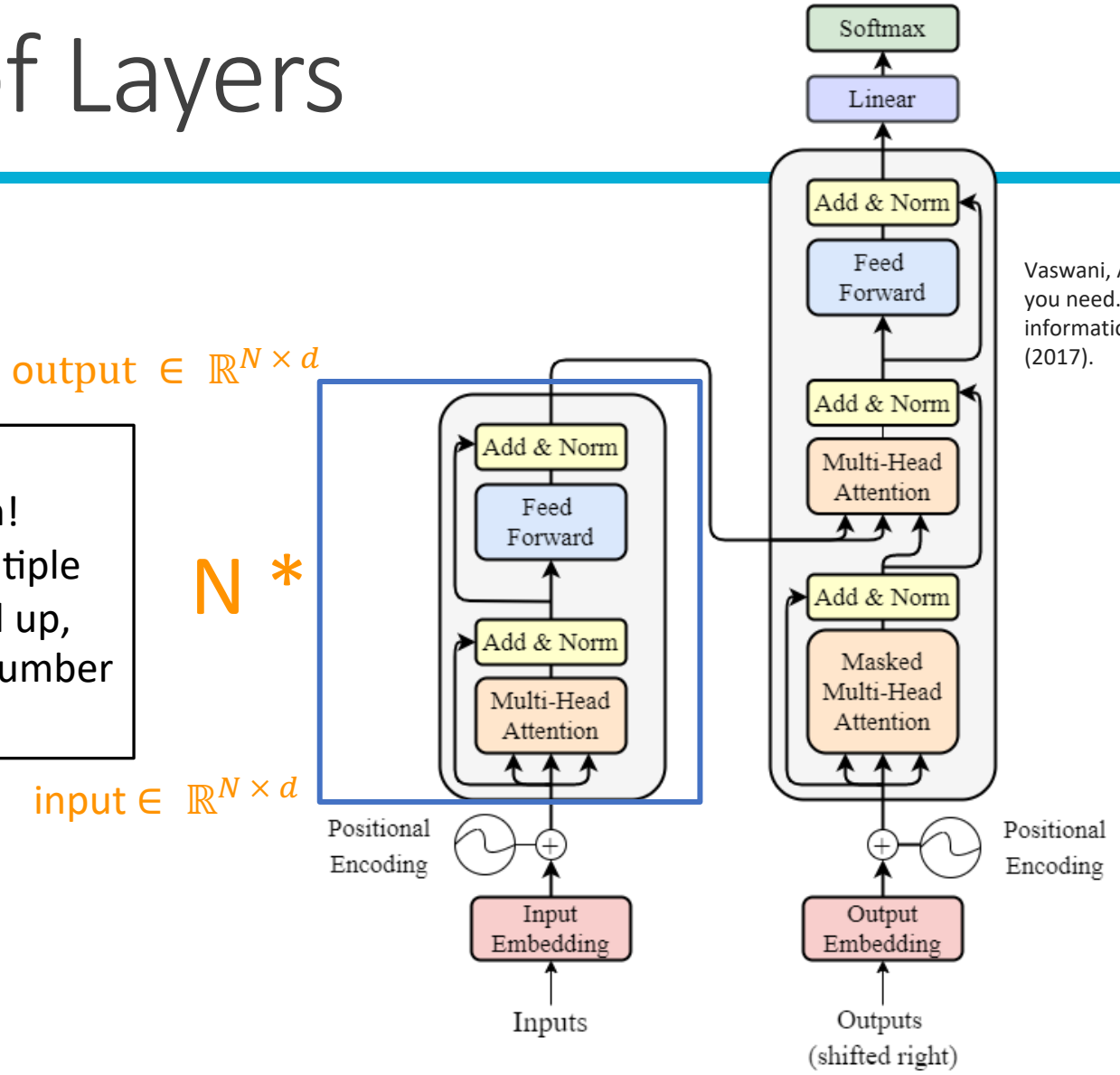
K, V uses the output from encoder as their X;

Q uses the decoder's information as X.

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

Number of Layers

Input, output dimension match!
We can have multiple encoders stacked up, to increase the number of trainable parameters.



Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Transformers' Achievements

1. In original paper, State-of-The-Art (SoTA) of machine translation.

Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	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**	**41.8**	$2.3 \cdot 10^{19}$	

6 encoders,
6 decoders,
h = 512, 100K train steps.

6 encoders,
6 decoders,
h = 1024, 300K train steps.

Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).
https://arxiv.org/abs/1706.03762

Transformer Variants (there are many more)

- Universal Transformer
  - Adds Adaptive Computation Time
  - Dehghani, Mostafa, et al. "Universal transformers." arXiv preprint arXiv:1807.03819 (2018).
- Longformer
  - Tackles long documents
  - Iz Beltagy, et al. (2020). Longformer: The Long-Document Transformer. arXiv:2004.05150.
- Roformer
  - Changes the design of positional encoding
  - Su, Jianlin, et al. "RoFormer: Enhanced Transformer with Rotary Position Embedding. CoRR abs/2104.09864 (2021)." arXiv preprint arXiv:2104.09864 (2021).
- Vision Transformer
  - Tackles vision tasks
  - Alexey Dosovitskiy, et al. “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale.” International Conference on Learning Representations. 2021.

Transformer能不能用在影像上呢？

Vision Transformer (ViT)

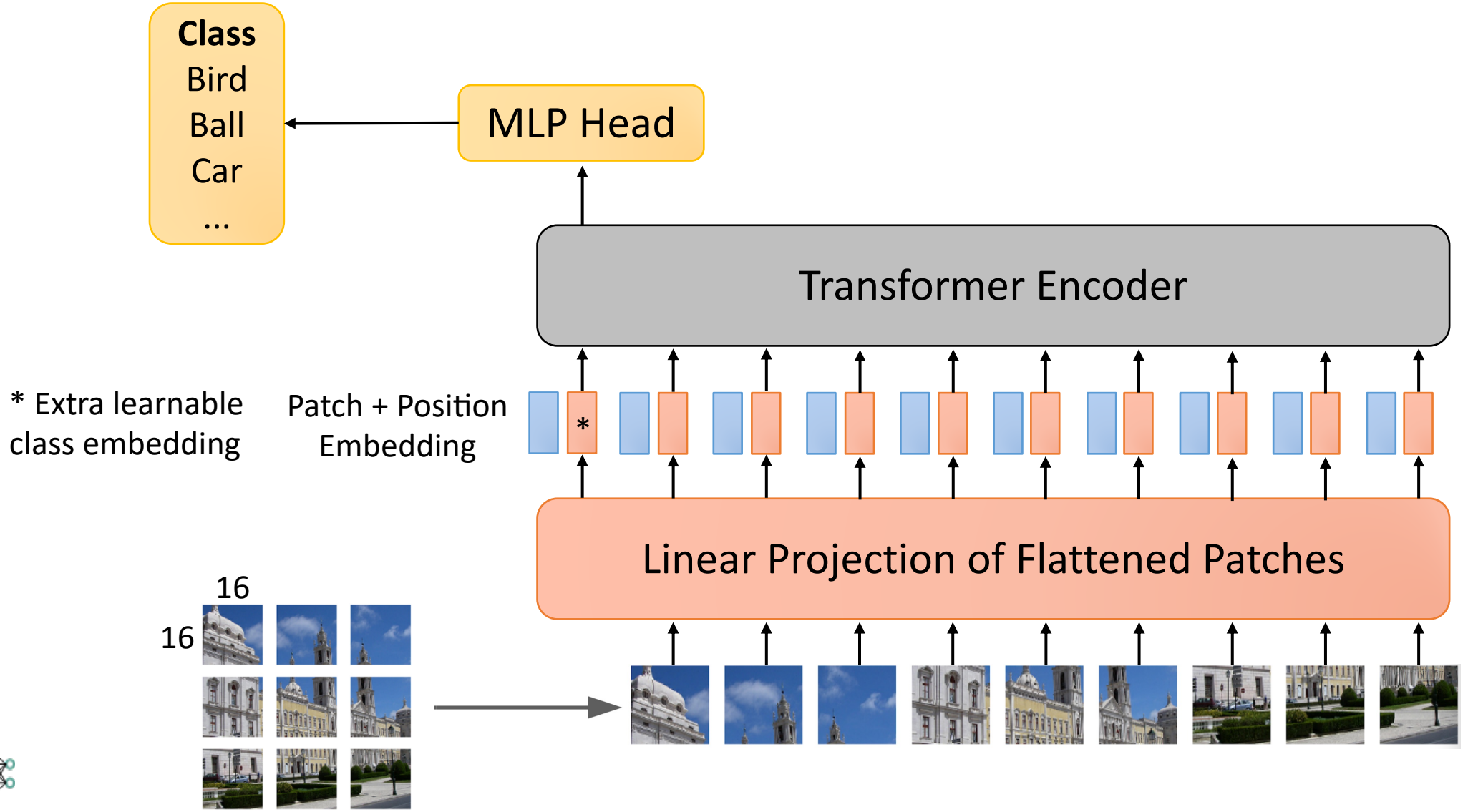
Concept: 把影像區域當成 token 輸入

Published as a conference paper at ICLR 2021

AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE

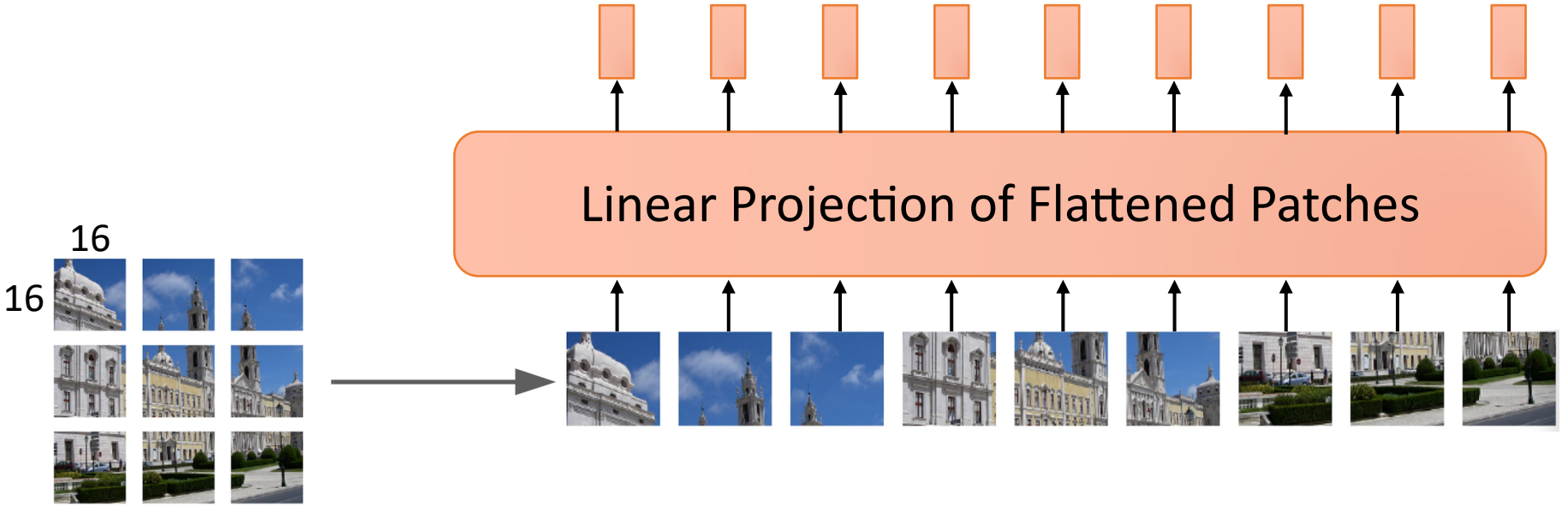
Alexey Dosovitskiy[*,†], **Lucas Beyer**[*], **Alexander Kolesnikov**[*], **Dirk Weissenborn**[*],
Xiaohua Zhai[*], **Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer,**
Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby[*,†]
[*]equal technical contribution, [†]equal advising
Google Research, Brain Team
{adosovitskiy, neilhoulsby}@google.com

Vision Transformer (ViT)



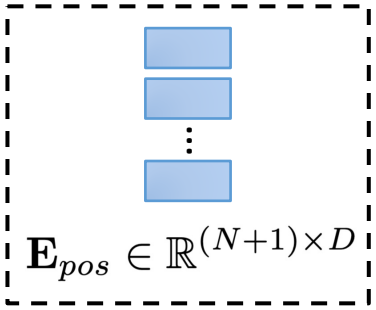
Linear Projection of Flattened Patches

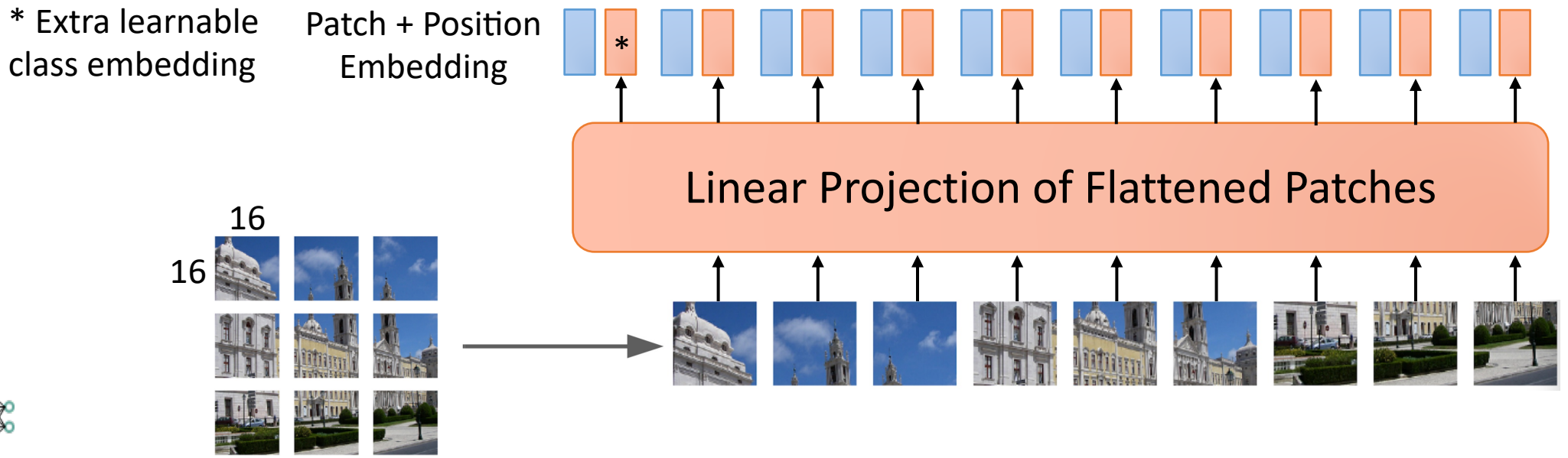
- 假設原始輸入是 224*224 的影像，切成 16*16 的小張影像 (patch)
- 每張 patch 進行 flattening 後，數值總數：16 * 16 * num_channels
- 使用線性轉換層將影像原始資料處理為 image embeddings
  - 輸出的維度為 D，Linear Projection 的矩陣大小為 16 * 16 * num_channels * D



Positional Embedding in ViT

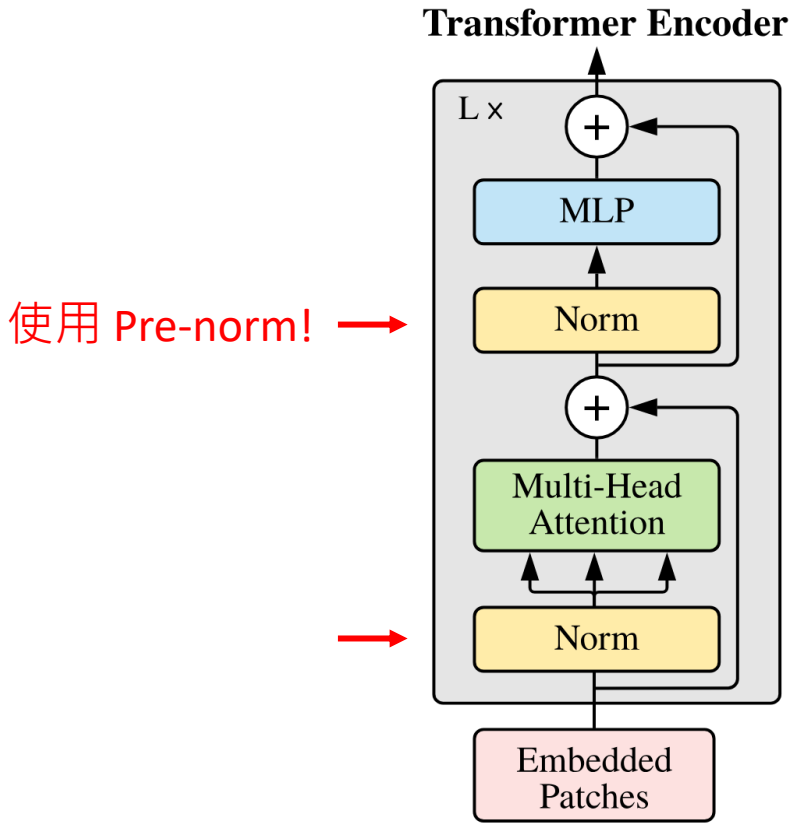
- 假設原始輸入是 224*224 的影像，切成 16*16 的 patches
  - num_patches = 14 * 14 = 196，等同於有 196+1 (CLS) 個 embeddings
  - 每個 patch 各自的 embedding (維度為D) 會在訓練過程中自動更新

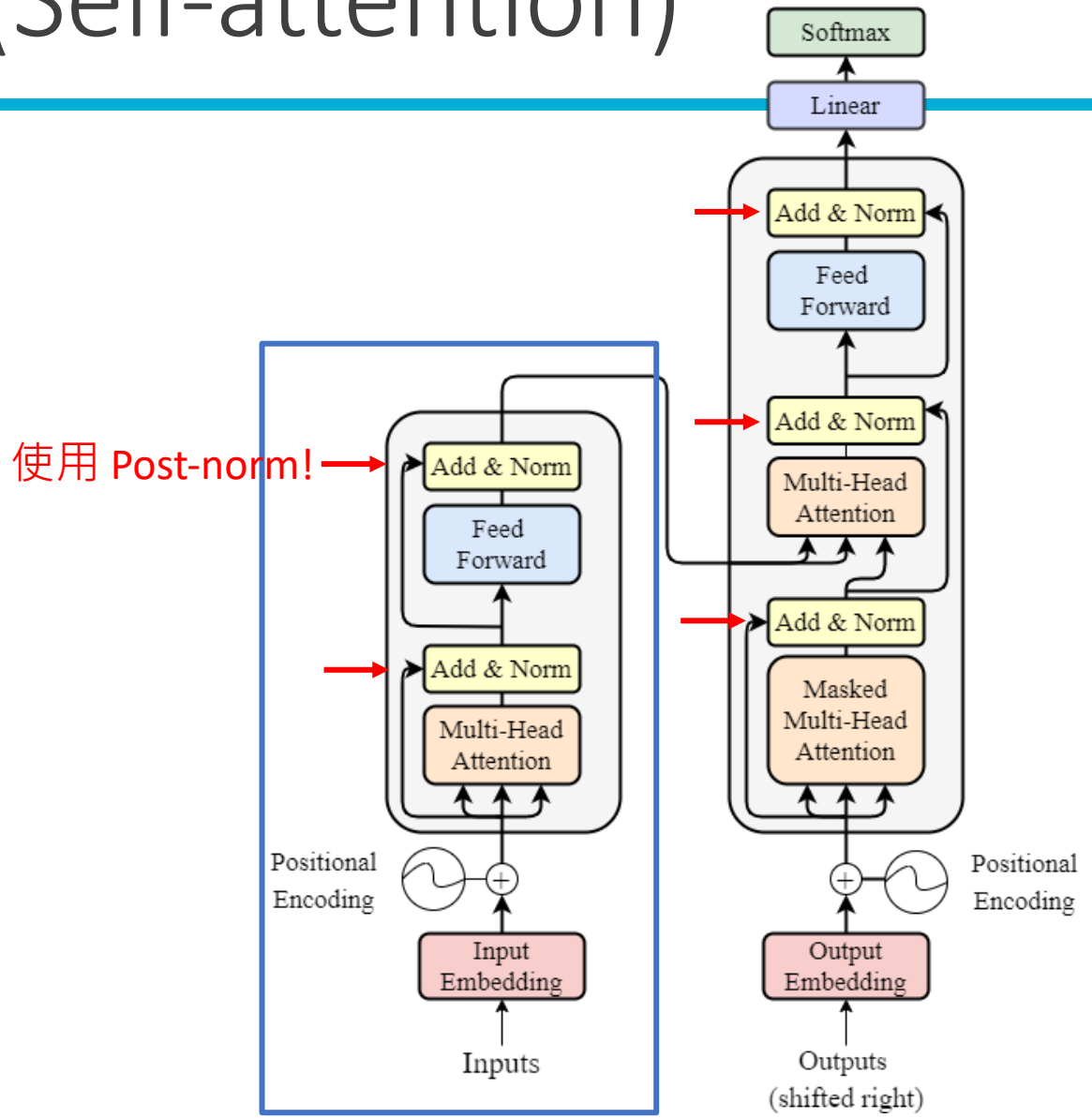




Transformer Layer (Self-attention)

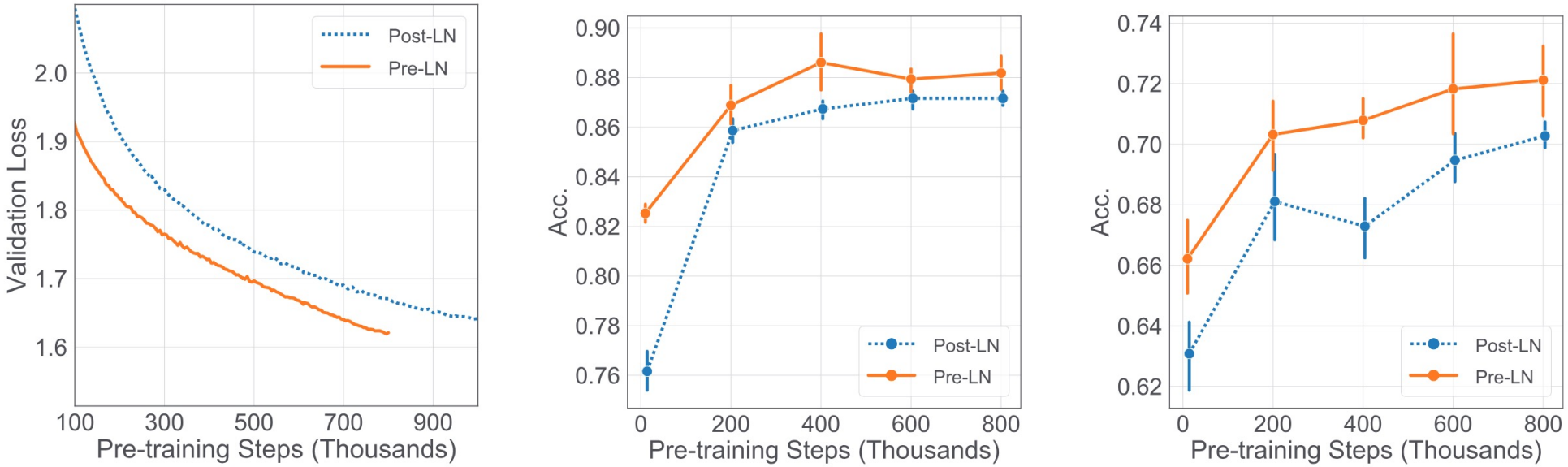
Left: ViT Transformer layer
Right: 原始 Transformers





Why using Pre-Norm (Pre-Normalization)?

Xiong, Ruibin, et al. "On layer normalization in the transformer architecture." International conference on machine learning. 2020.



(a) Validation Loss on BERT (b) Accuracy on MRPC (c) Accuracy on RTE

Figure 5. Performances of the models on unsupervised pre-training (BERT) and downstream tasks

Additional Links

- https://nlp.seas.harvard.edu/annotated-transformer/
- https://datascience.stackexchange.com/questions/82451/why-is-10000-used-as-the-denominator-in-positional-encodings-in-the-transformer

Announcements

Week 9: Mid-term Exam

- 08: 50 – 10:20; 10:30 – 12:00 Mid-term Exam
  - 分成兩梯次進行，使用電腦作答，本週三前公布梯次與座位表
- 題目 (範圍：W1 - W8 PPT 與 code)：
  - 簡答題
  - 程式題 (PyTorch 語法、訓練/測試框架)
    - 可以搜尋 PyTorch 文件 (https://pytorch.org/docs/stable/index.html)
    - 不能 Google、不能使用 LLMs

Thank you!

Instructor: 林英嘉

✉ yjlin@cgu.edu.tw

TA: 林君襄

✉ becky890926@gmail.com