北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

学术分享

Transformer入门

北京理工大学宇航学院

李嘉鑫

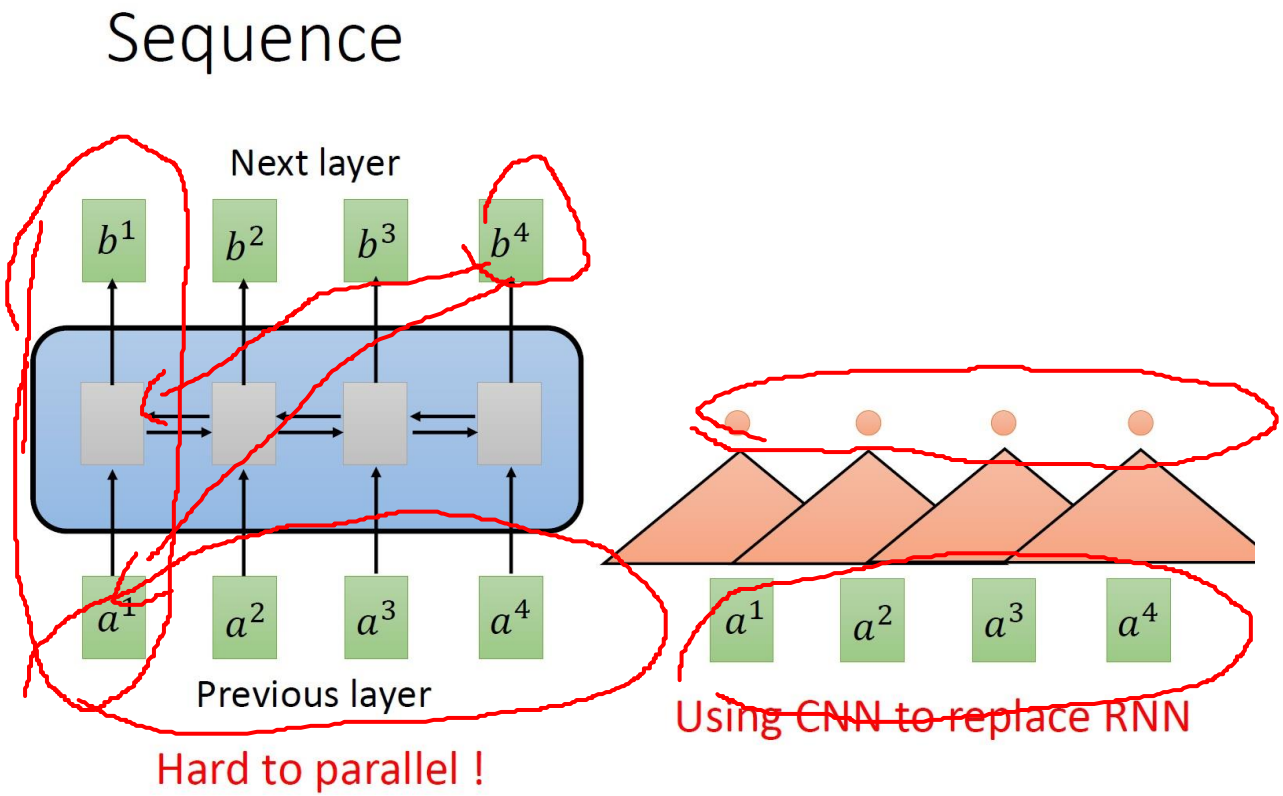
◆Transfomer的前世今生

Transformer是Google团队在2017年提出的一种NLP经典模型。

1. 采用**Self-Attention**机制
2. 不采用RNN的顺序结构，能够并行化训练，且拥有全局信息

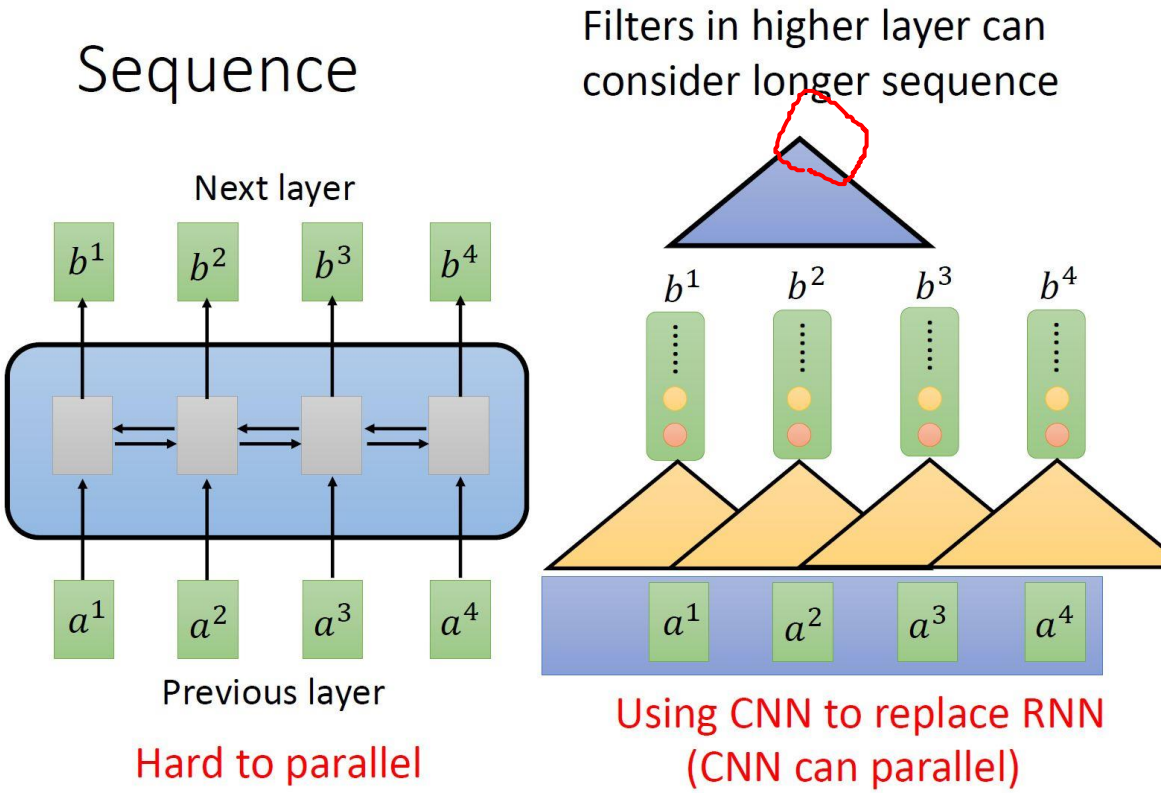
◆RNN与CNN

网络结构



◆RNN与CNN

网络结构



◆Self-attention

网络结构

Self-attention

https://arxiv.org/abs/1706.03762

q: query (to match others)

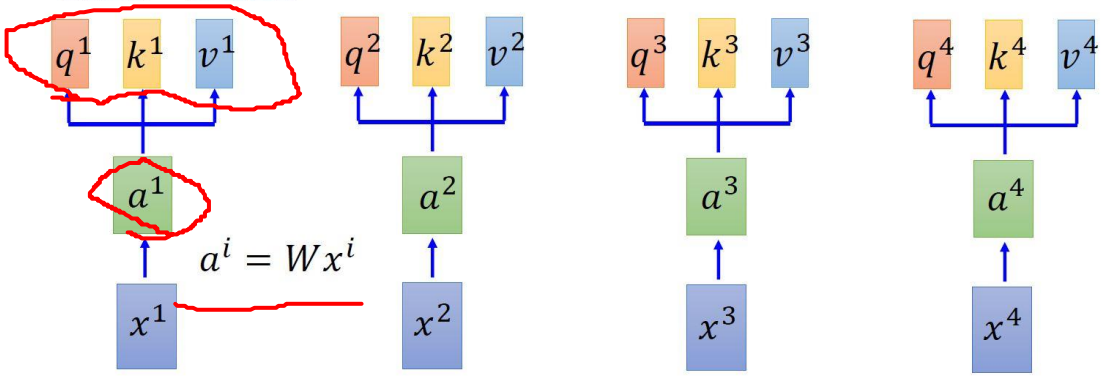
$$q^i = W^q a^i$$

k: key (to be matched)

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

v: information to be extracted

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



◆Self-attention

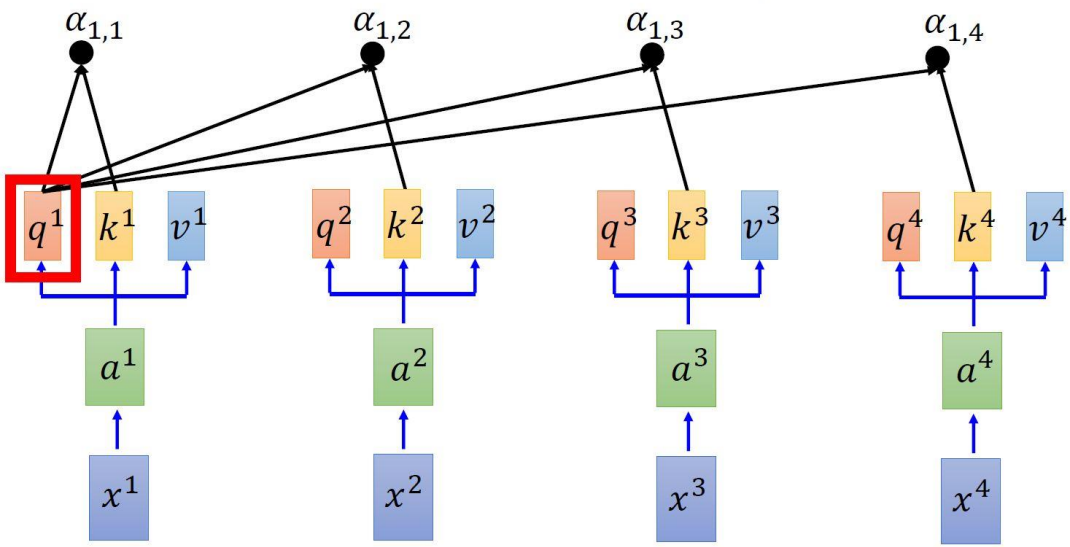
网络结构

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

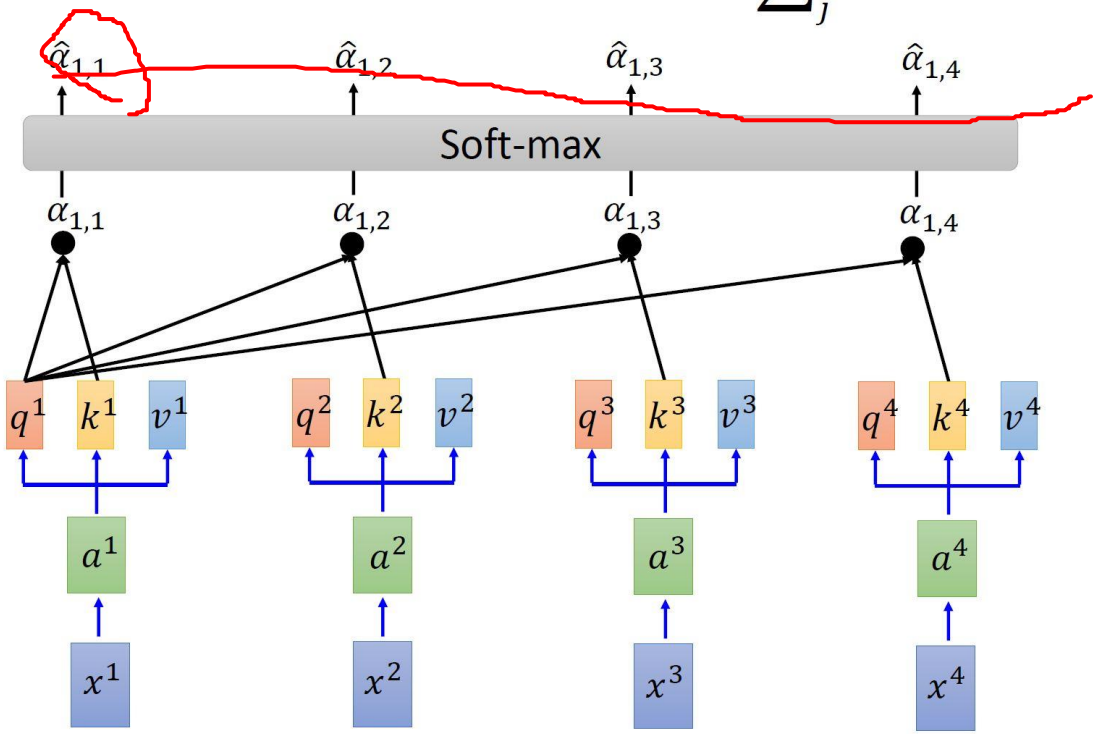


◆Self-attention

网络结构

Self-attention

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



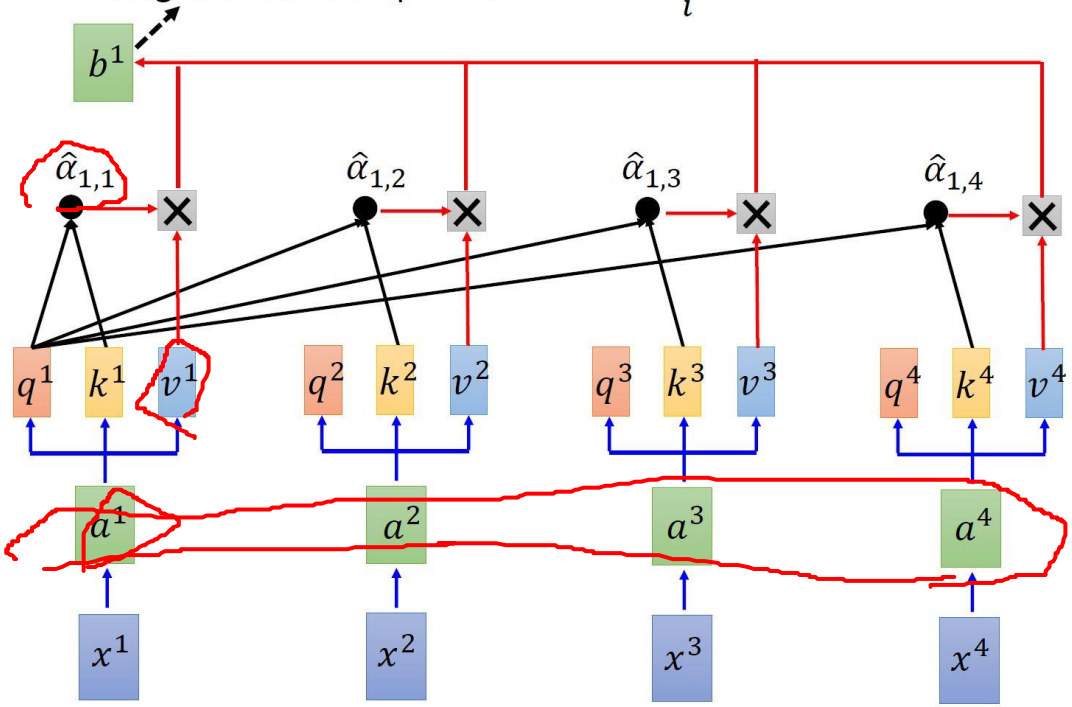
◆Self-attention

网络结构

Self-attention

Considering the whole sequence

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



◆Self-attention

q k v

网络结构

Self-attention

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

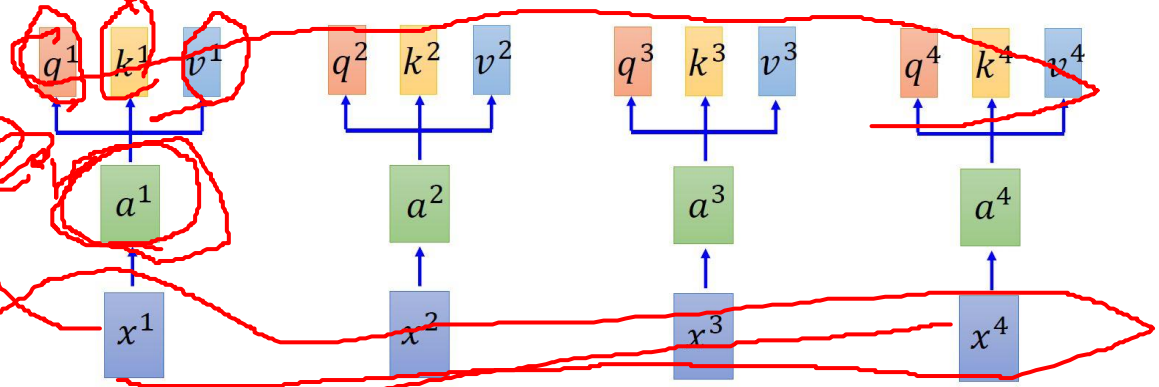
$$k^i = W^k a^i$$

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

$$\underbrace{q^1\, q^2\, q^3\, q^4}_{Q} = W^q \underbrace{a^1\, a^2\, a^3\, a^4}_{I}$$

$$\underbrace{k^1\, k^2\, k^3\, k^4}_{K} = W^k \underbrace{a^1\, a^2\, a^3\, a^4}_{I}$$

$$\underbrace{v^1\, v^2\, v^3\, v^4}_{V} = W^v \underbrace{a^1\, a^2\, a^3\, a^4}_{I}$$



◆Self-attention

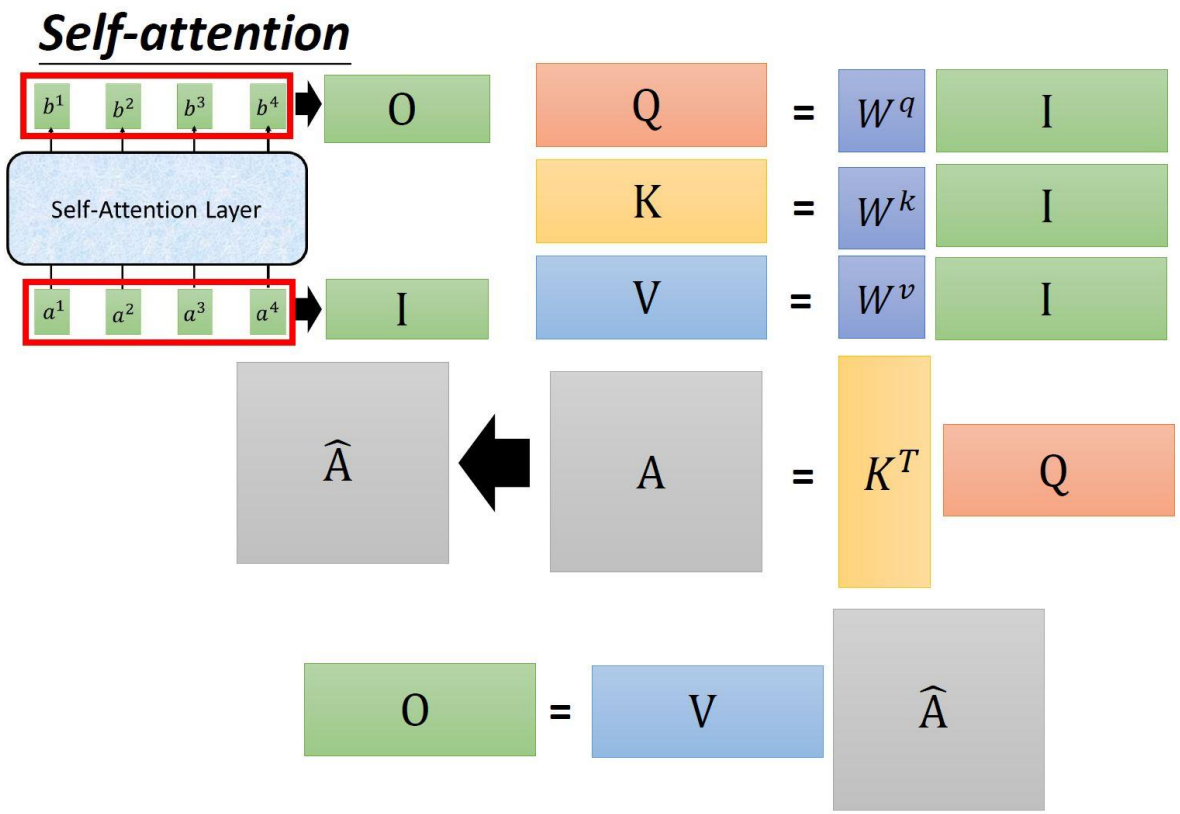
网络结构

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

$$\underbrace{\begin{bmatrix} b^1 & b^2 & b^3 & b^4 \end{bmatrix}}_{O} = \underbrace{\begin{bmatrix} v^1 & v^2 & v^3 & v^4 \end{bmatrix}}_{V} \underbrace{\begin{bmatrix} \hat{\alpha}_{1,1} & \hat{\alpha}_{2,1} & \hat{\alpha}_{3,1} & \hat{\alpha}_{4,1} \\ \hat{\alpha}_{1,2} & \hat{\alpha}_{2,2} & \hat{\alpha}_{3,2} & \hat{\alpha}_{4,2} \\ \hat{\alpha}_{1,3} & \hat{\alpha}_{2,3} & \hat{\alpha}_{3,3} & \hat{\alpha}_{4,3} \\ \hat{\alpha}_{1,4} & \hat{\alpha}_{2,4} & \hat{\alpha}_{3,4} & \hat{\alpha}_{4,4} \end{bmatrix}}_{\hat{A}}$$

◆Self-attention

网络结构



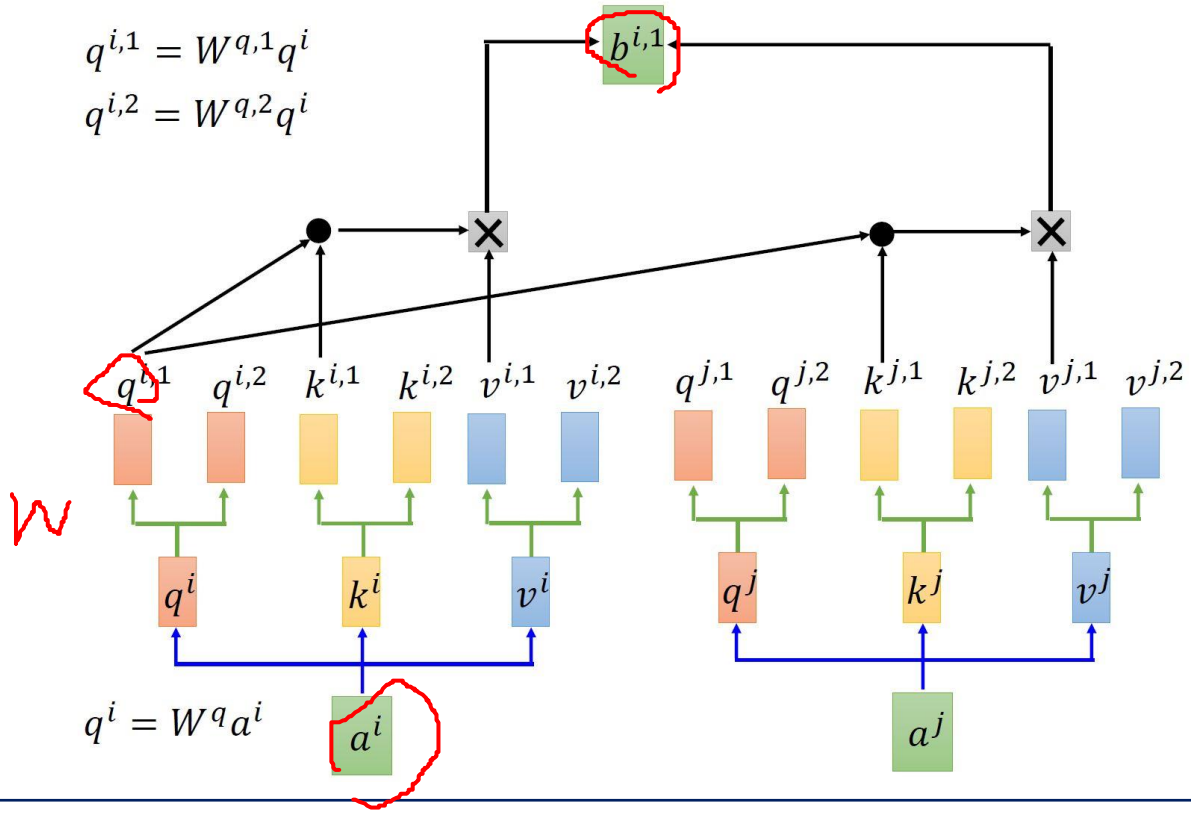
◆Multi-head Self-attention

网络结构

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

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

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



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

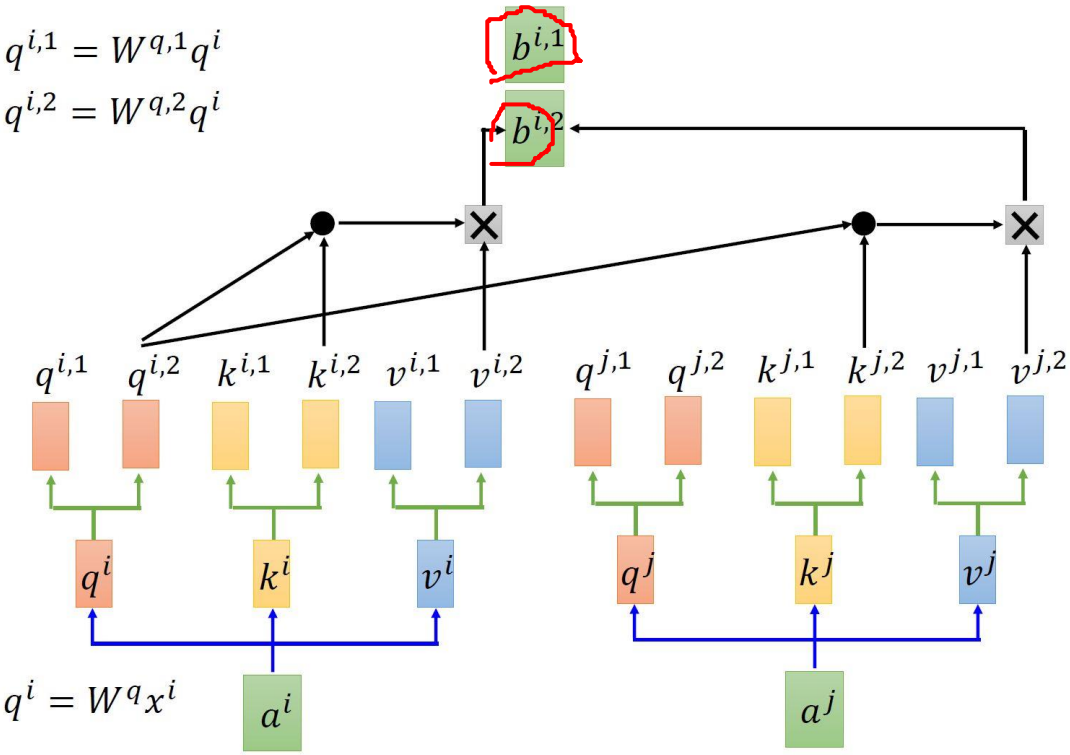
◆Multi-head Self-attention

网络结构

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

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

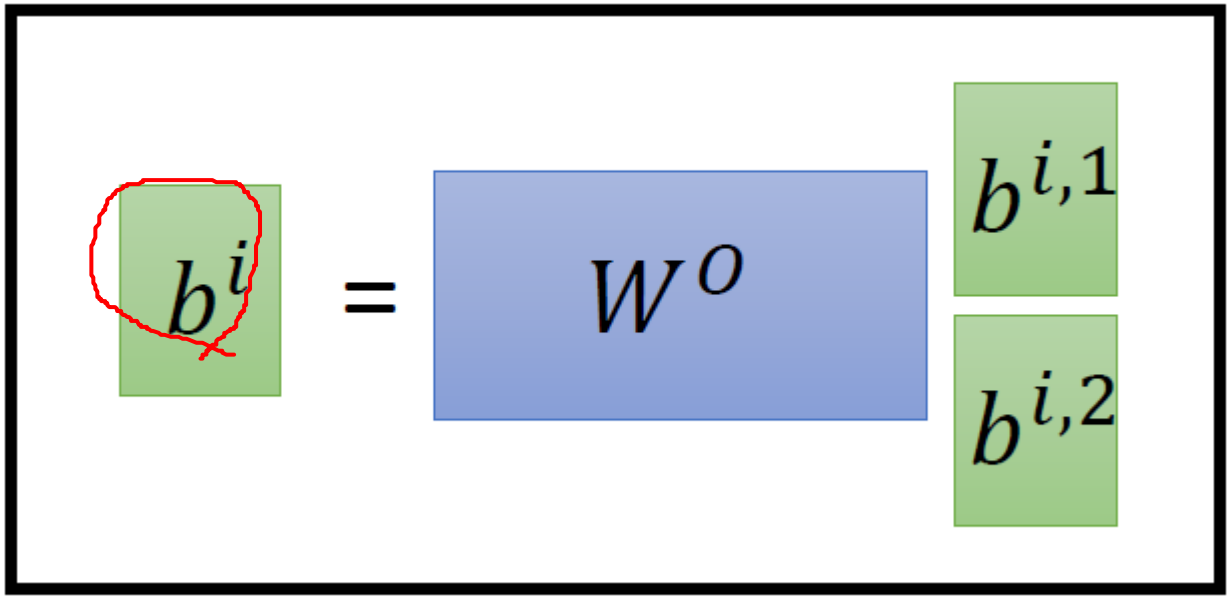
$$q^{i,2} = W^{q,2} q^i$$



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

◆Multi-head Self-attention

网络结构

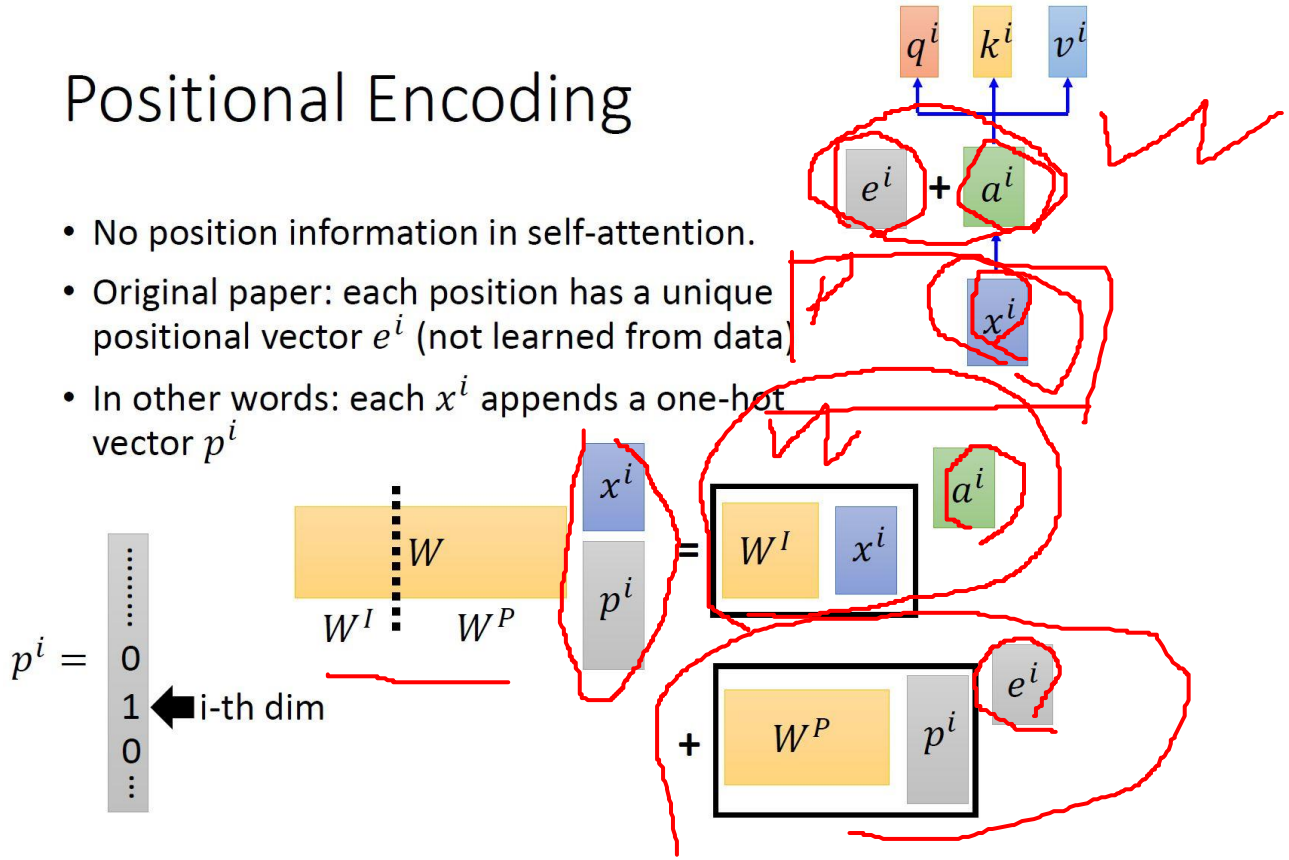


◆Position Encoding

网络结构

Positional Encoding

- No position information in self-attention.
- Original paper: each position has a unique positional vector e^i (not learned from data)
- In other words: each x^i appends a one-hot vector p^i



◆Position Encoding

网络结构

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

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

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

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

◆Position Encoding

网络结构

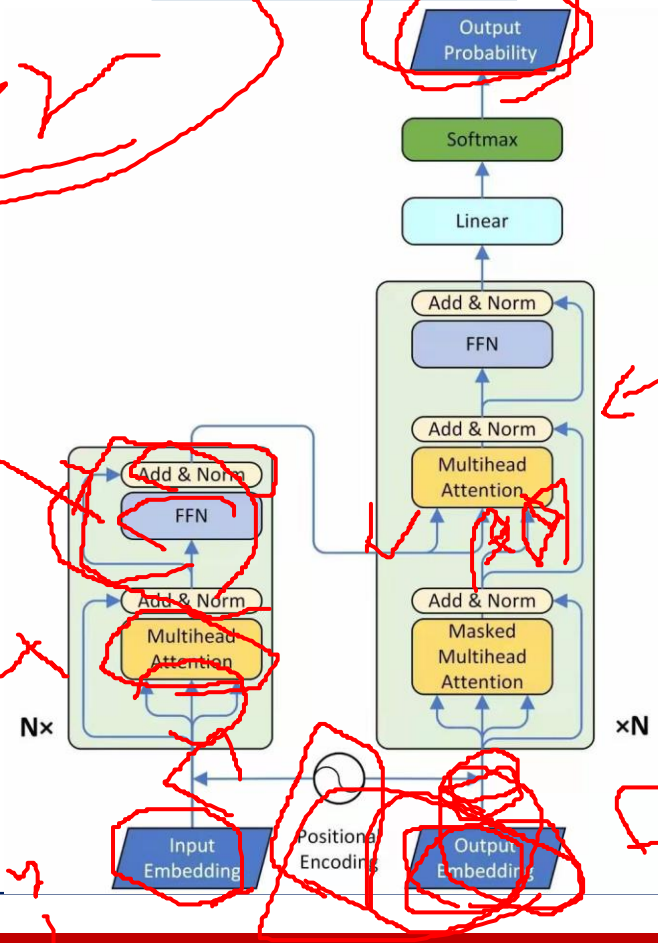
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}})$$

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

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

◆Transformer

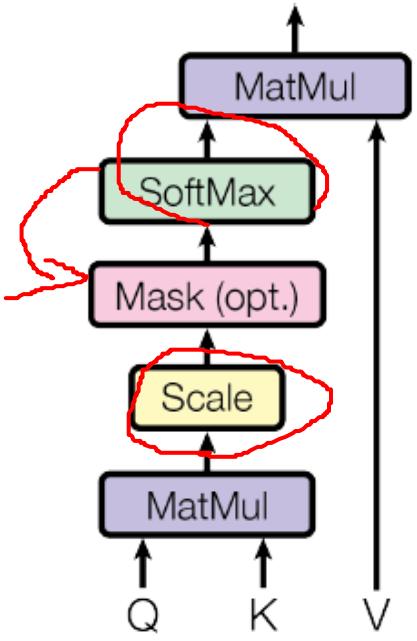
网络结构



◆Transformer

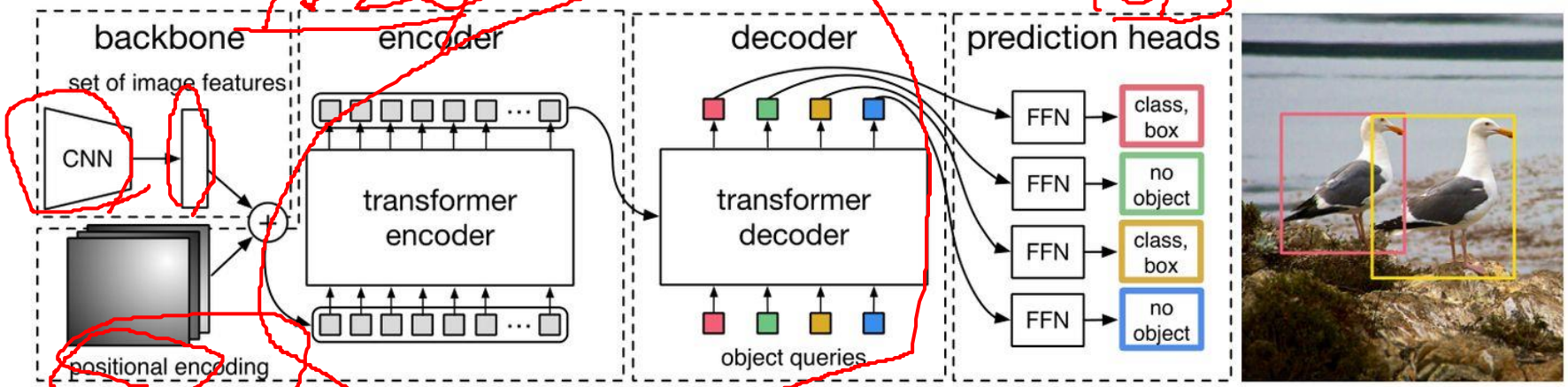
网络结构

Scaled Dot-Product Attention



◆DETR

网络结构


backbone
set of image features
CNN
positional encoding
encoder
transformer encoder
decoder
transformer decoder
object queries
prediction heads
FFN
FFN
FFN
FFN
class, box
no object
class, box
no object

◆DETR

网络结构

$$a)\quad PE_{(pos_x,2i)} = sin(pos_x/10000^{2i/128})$$

$$b)\quad PE_{(pos_x,2i+1)} = cos(pos_x/10000^{2i/128})$$

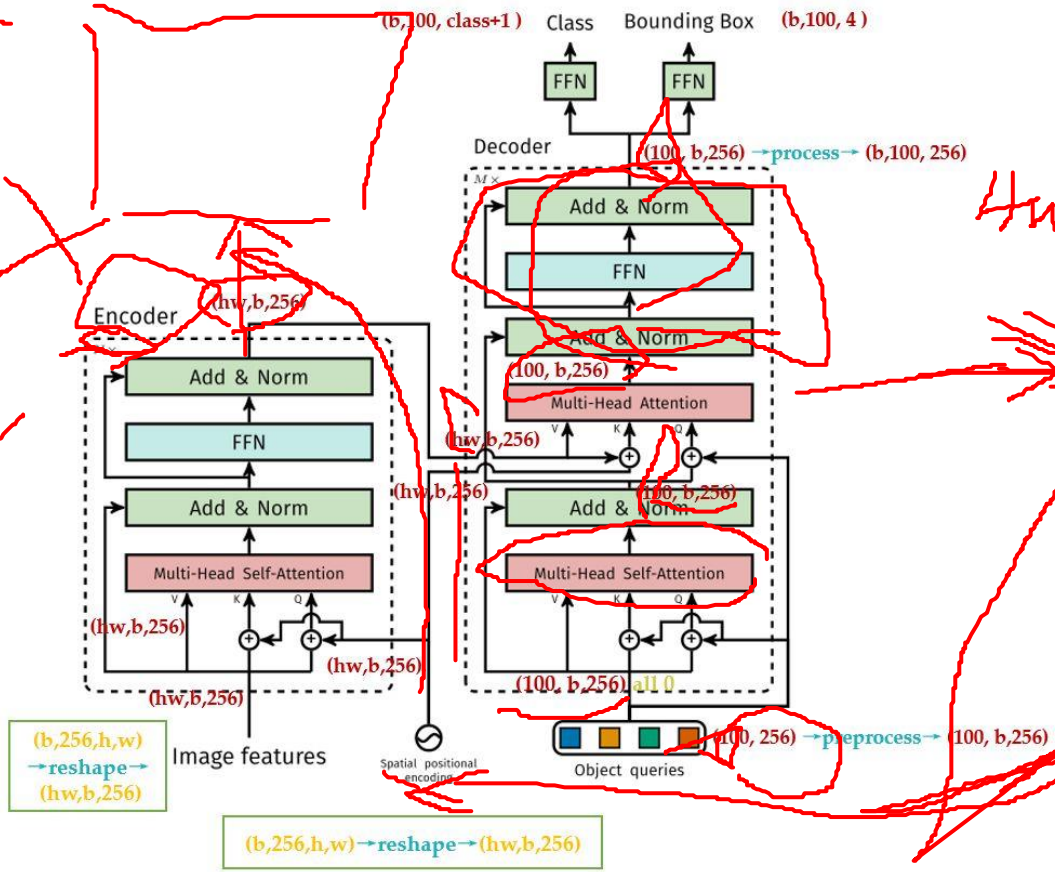
$$c)\quad PE_{(pos_y,2i)} = sin(pos_y/10000^{2i/128})$$

$$d)\quad PE_{(pos_y,2i+1)} = cos(pos_y/10000^{2i/128})$$

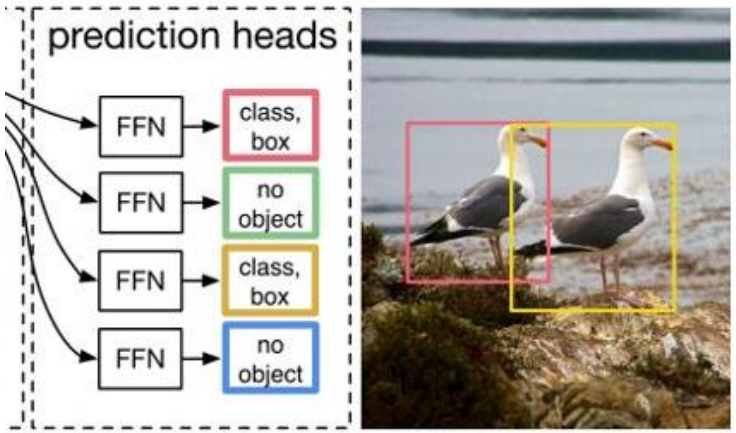
北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

◆DETR

网络结构



◆DETR的损失函数



$$L_{\text{Hungarian}}(y, \hat{y}) = \sum_{i=1}^{N} \left[-\log \hat{p}_{\hat{\sigma}(i)}(c_i) + 1_{\{c_i \neq \varnothing\}} \, L_{box}(b_i, \hat{b}_{\hat{\sigma}(i)}) \right]$$

$$L_{box}(b_i, \hat{b}_{\hat{\sigma}(i)}) = \lambda_{\text{iou}} L_{iou}(b_i, \hat{b}_{\sigma(i)}) + \lambda_{\text{L1}} ||b_i - \hat{b}_{\hat{\sigma}(i)}||_1, \; where \; \lambda_{\text{iou}}, \lambda_{\text{L1}} \in R$$