KOREA UNIVERSITY

**DSBA** Transformer survey paper study

# A Survey of Transformers
## Appendix: Complexity, Parameters, and Scaling
arXiv preprint


DSBA
Data Science & Business Analytics

**고려대학교 산업경영공학과**

**Data Science & Business Analytics Lab**
**이유경, 김명섭, 윤훈상, 김지나, 허재혁, 김수빈**

발표자 : 김명섭

# 00 Complexity and Parameters

## Matrix, Complexity, and Parameters

- 행렬 연산과 Complexity, Parameter의 관계를 알아보면...

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \times \begin{bmatrix} \mathbf{1} \\ \mathbf{2} \\ \mathbf{3} \end{bmatrix} = \begin{bmatrix} 1 \times 1 + 2 \times 2 + 3 \times 3 \\ 4 \times 1 + 5 \times 2 + 6 \times 3 \end{bmatrix} = \begin{bmatrix} 14 \\ 32 \end{bmatrix}$$

$(2 \times 3)$ $(3 \times 1)$ $(2 \times 1)$

$$\boldsymbol{Computational\ Complexity\ (Mult) = 2 \times 3 \times 1 = 6}$$

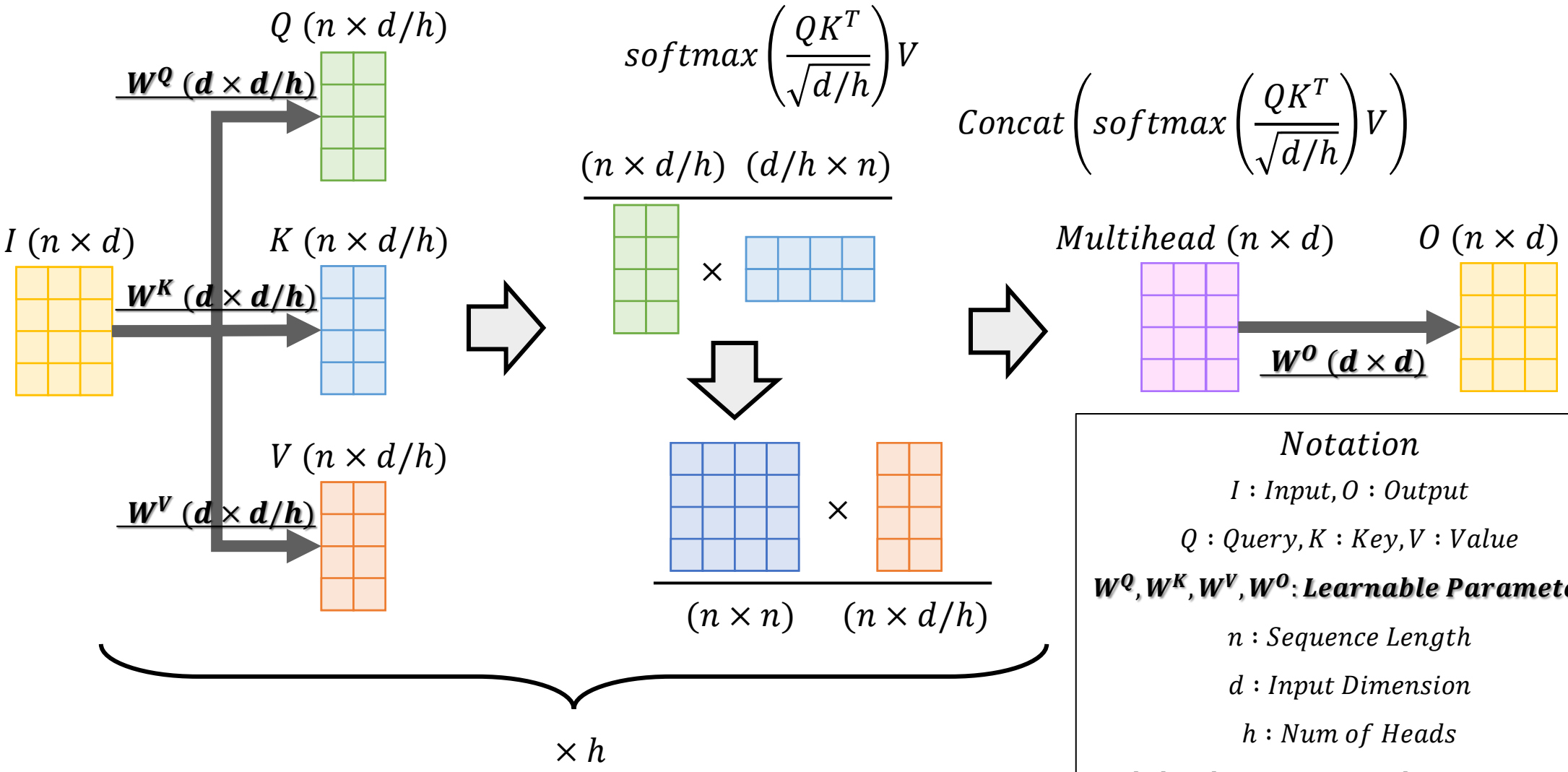
$$\boldsymbol{Parameters = 3 \times 1 = 3}$$

$(3 \times 1)$ *size matirx is* ***learnable***

# 00 Complexity and Parameters

## Computational Complexity of Self-Attention

| Module | Complexity | #Parameters |
|---|---|---|
| Self-attention | $O(n^2 \cdot d)$ | $4d^2$ |
| Position-wise FFN | $O(n \cdot d^2)$ | $8d^2$ |




### Notation

$I : Input, O : Output$

$Q : Query, K : Key, V : Value$

$\boldsymbol{W^Q, W^K, W^V, W^O}$ : ***Learnable Parameters***

$n : Sequence\ Length$

$d : Input\ Dimension$

$h : Num\ of\ Heads$

$Multihead : Concatenated\ Attention\ Value$

# 00 Complexity and Parameters

## Computational Complexity of Self-Attention

| Module | Complexity | #Parameters |
|---|---|---|
| Self-attention | $O(n^2 \cdot d)$ | $4d^2$ |
| Position-wise FFN | $O(n \cdot d^2)$ | $8d^2$ |

Query Projection : $I\boldsymbol{W^Q} \rightarrow (n \times d) \times (\boldsymbol{d \times d/h}) \rightarrow \frac{nd^2}{h}$

Key Projection : $I\boldsymbol{W^K} \rightarrow (n \times d) \times (\boldsymbol{d \times d/h}) \rightarrow \frac{nd^2}{h}$

Value Projection : $I\boldsymbol{W^V} \rightarrow (n \times d) \times (\boldsymbol{d \times d/h}) \rightarrow \frac{nd^2}{h}$

Self-Attention : $softmax\left(\frac{QK^T}{\sqrt{d/h}}\right)V \rightarrow \frac{n^2d}{h} + \frac{n^2d}{h} = \frac{2n^2d}{h}$

Attention Score : $\frac{QK^T}{\sqrt{d/h}} = A \rightarrow (n \times d/h) \times (d/h \times n) \rightarrow \frac{n^2d}{h}$

Attention Value : $softmax(A)V \rightarrow (n \times n) \times (n \times d/h) \rightarrow \frac{n^2d}{h}$

$\times h$

Output Projection : $Multihead\boldsymbol{W^O} \rightarrow (n \times d) \times (\boldsymbol{d \times d}) \rightarrow nd^2$

*Notation*

$I : Input, O : Output$

$Q : Query, K : Key, V : Value$

$\boldsymbol{W^Q, W^K, W^V, W^O}$ : ***Learnable Parameters***

$n : Sequence\ Length$

$d : Input\ Dimension$

$h : Num\ of\ Heads$

$Multihead : Concatenated\ Attention\ Value$

**Multi-Head Self-Attention Computational Complexity = $\boldsymbol{4nd^2 + 2n^2d \rightarrow O(nd^2 + n^2d)}$**

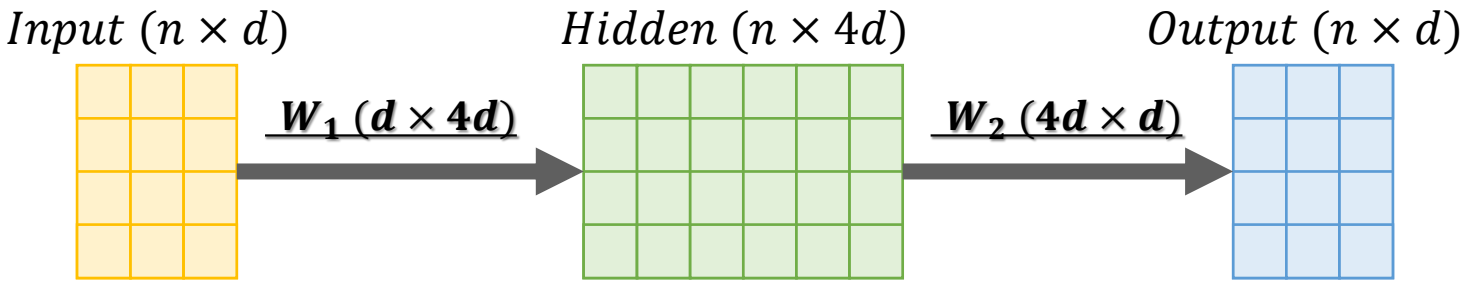
**Self-Attention Computational Complexity = $\boldsymbol{2n^2d \rightarrow O(n^2d)}$**

**Num of Parameters = $\underline{\boldsymbol{4d^2}}$**

# 00 Complexity and Parameters

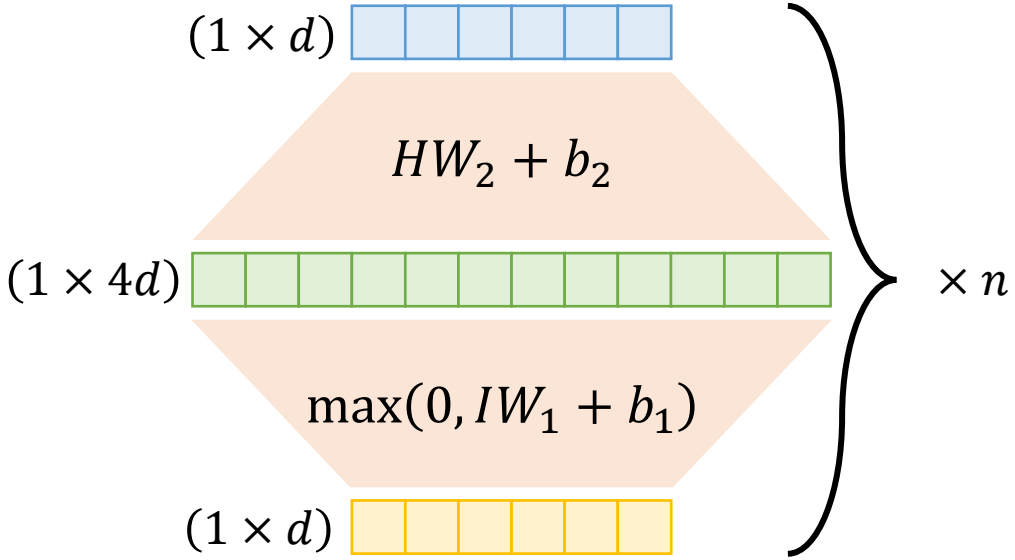
## Computational Complexity of Position-wise FFN

| Module | Complexity | #Parameters |
|---|---|---|
| Self-attention | $O(n^2 \cdot d)$ | $4d^2$ |
| Position-wise FFN | $O(n \cdot d^2)$ | $8d^2$ |




Input to Hidden : $I\boldsymbol{W_1} = H \rightarrow (n \times d) \times \boldsymbol{(d \times 4d)} \rightarrow 4nd^2$

Hidden to Output : $H\boldsymbol{W_2} \rightarrow (n \times 4d) \times \boldsymbol{(4d \times d)} \rightarrow 4nd^2$




<Position-wise FFNs>

*Notation*

$I$ : *Input*, $H$ : *Hidden*, $O$ : *Output*

$\boldsymbol{W_1}, \boldsymbol{W_2}$: ***Learnable Parameters***

$n$ : *Sequence Length*

$d$ : *Input Dimension*

**Position-wise FFN Computational Complexity = $\boldsymbol{8nd^2} \rightarrow \boldsymbol{O(nd^2)}$**

**Num of Parameters = $\underline{\boldsymbol{8d^2}}$**

# 00 Complexity and Parameters

## Computational Complexity of Self-Attention

https://stackoverflow.com/questions/65703260

**Question:**

> Therefore, the total complexity of the layer is `O(n^2 d + n d^2)`, which is worse than that of a traditional RNN layer. I obtained the same result for multi-headed attention too, on considering the appropriate intermediate representation dimensionalities ( `dk` , `dv` ) and finally multiplying by the number of heads `h` .

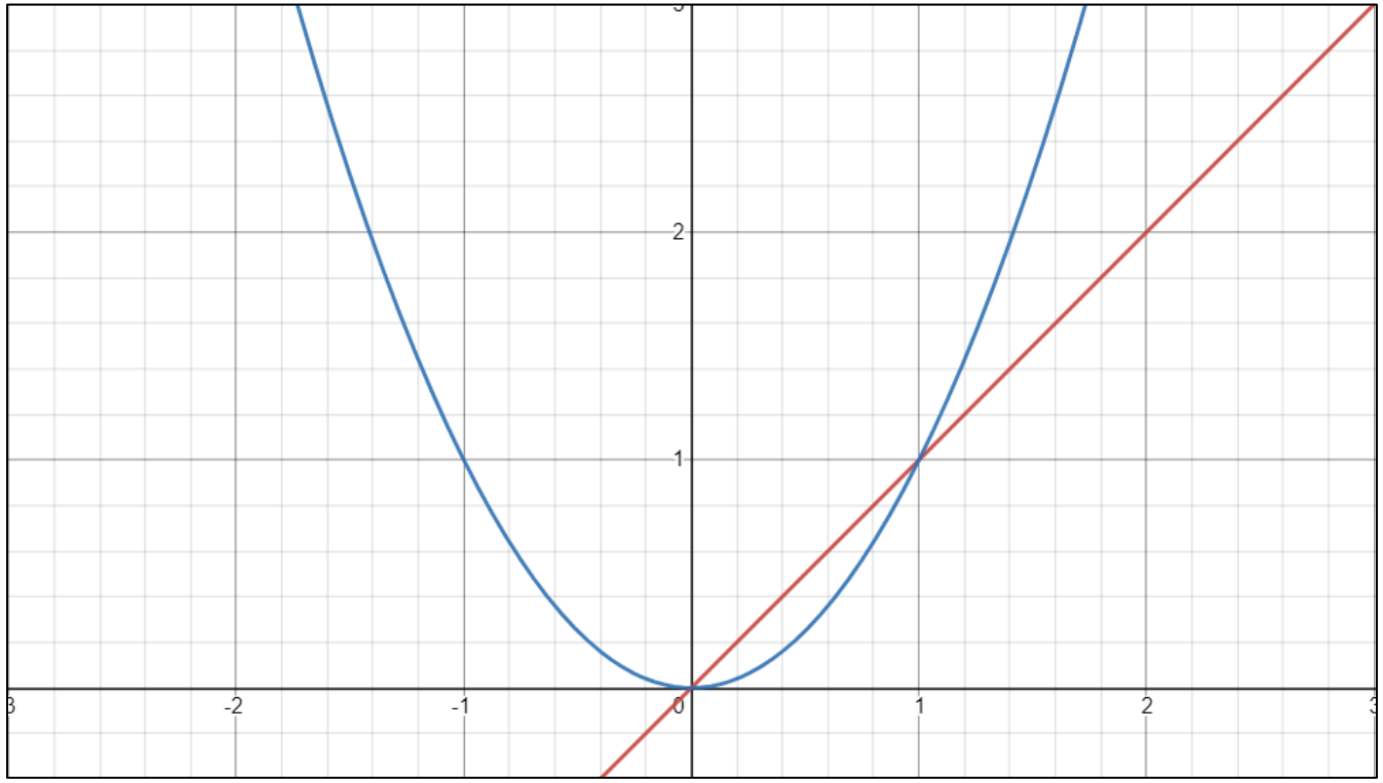
**Answer:**

> When the original Attention paper was first introduced, it didn't require to calculate `Q` , `V` and `K` matrices, as the values were taken directly from the hidden states of the RNNs, and thus the complexity of Attention layer **is** `O(n^2·d)` .
>
> Now, to understand what `Table 1` contains please keep in mind how most people scan papers: they read title, abstract, then look at figures and tables. Only then if the results were interesting, they read the paper more thoroughly. So, the main idea of the `Attention is all you need` paper was to replace the RNN layers completely with attention mechanism in seq2seq setting because RNNs were really slow to train. If you look at the `Table 1` in this context, you see that it compares RNN, CNN and Attention and highlights the motivation for the paper: using Attention should have been beneficial over RNNs and CNNs. It should have been advantageous in 3 aspects: constant amount of calculation steps, constant amount of operations **and** lower computational complexity for usual Google setting, where `n ~= 100` and

# 00 Complexity and Parameters

## Quadratic Computation Complexity

https://www.desmos.com/calculator?lang=ko

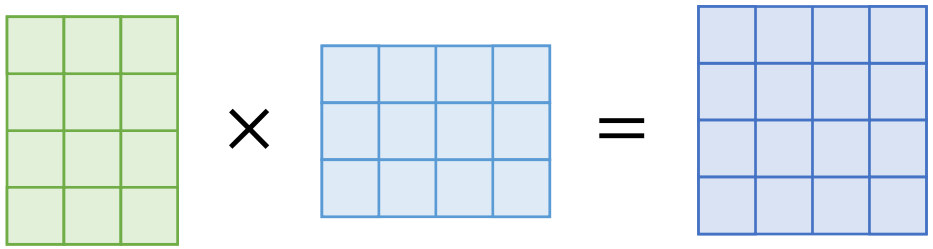


# 00 Scaling

## Scaled Dot-Product Attention

Algebra of random variables

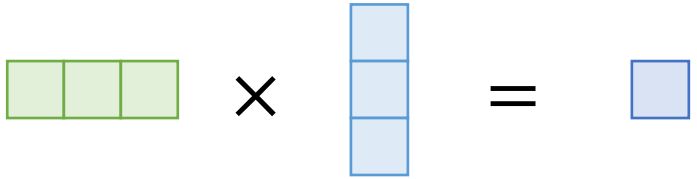
$Q\ (n \times d_k)$ $K\ (d_k \times n)$



$Attention\ Score\ Matrix$
$(n \times n)$

$$softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

$q\ (1 \times d_k)$ $k\ (d_k \times 1)$



$Attention\ Score$
$(scalar)$

$$Attention\ Score = q \cdot k = \sum_{i=1}^{d_k} q_i k_i$$

- ❖ *if all random variables are independent,*
- ✓ $E[\sum_{i=1}^{n} X_i] = \sum_{i=1}^{n} E[X_i]$
- ✓ $E[\prod_{i=1}^{n} X_i] = \prod_{i=1}^{n} E[X_i]$
- ✓ $Var[\sum_{i=1}^{n} X_i] = \sum_{i=1}^{n} Var[X_i]$
- ✓ $Var[\prod_{i=1}^{n} X_i] = \prod_{i=1}^{n}(Var[X_i] + (E[X_i])^2) - \prod_{i=1}^{n}(E[X_i])^2$

$$if\ \ q_i \sim E[q_i] = 0, Var[q_i] = 1,$$
$$k_i \sim E[k_i] = 0, Var[k_i] = 1, q_i \perp k_i,$$
$$\rightarrow E[q \cdot k] = 0, Var[q \cdot k] = d_k$$

# 00 Scaling

## Scaled Dot-Product Attention

He et al., DeBERTa: Decoding-Enhanced BERT with Disentangled Attention, ICLR, 2021

**Algorithm 1** Disentangled Attention

**Input:** Hidden state $\boldsymbol{H}$, relative distance embedding $\boldsymbol{P}$, relative distance matrix $\boldsymbol{\delta}$. Content projection matrix $\boldsymbol{W_{k,c}}$, $\boldsymbol{W_{q,c}}$, $\boldsymbol{W_{v,c}}$, position projection matrix $\boldsymbol{W_{k,r}}$, $\boldsymbol{W_{q,r}}$.

$\boldsymbol{K_c} = \boldsymbol{H}\boldsymbol{W_{k,c}}$, $\boldsymbol{Q_c} = \boldsymbol{H}\boldsymbol{W_{q,c}}$, $\boldsymbol{V_c} = \boldsymbol{H}\boldsymbol{W_{v,c}}$, $\boldsymbol{K_r} = \boldsymbol{P}\boldsymbol{W_{k,r}}$, $\boldsymbol{Q_r} = \boldsymbol{P}\boldsymbol{W_{q,r}}$
$\boldsymbol{A_{c\to c}} = \boldsymbol{Q_c}\boldsymbol{K_c}^\intercal$
**for** $i = 0, ..., N-1$ **do**
    $\tilde{\boldsymbol{A}}_{\boldsymbol{c\to p}}[i,:] = \boldsymbol{Q_c}[i,:]\boldsymbol{K_r}^\intercal$
**end for**
**for** $i = 0, ..., N-1$ **do**
    **for** $j = 0, ..., N-1$ **do**
        $\boldsymbol{A_{c\to p}}[i,j] = \tilde{\boldsymbol{A}}_{\boldsymbol{c\to p}}[i, \boldsymbol{\delta}[i,j]]$
    **end for**
**end for**
**for** $j = 0, ..., N-1$ **do**
    $\tilde{\boldsymbol{A}}_{\boldsymbol{p\to c}}[:,j] = \boldsymbol{K_c}[j,:]\boldsymbol{Q_r}^\intercal$
**end for**
**for** $j = 0, ..., N-1$ **do**
    **for** $i = 0, ..., N-1$ **do**
        $\boldsymbol{A_{p\to c}}[i,j] = \tilde{\boldsymbol{A}}_{\boldsymbol{p\to c}}[\boldsymbol{\delta}[j,i], j]$
    **end for**
**end for**
$\tilde{\boldsymbol{A}} = \boldsymbol{A_{c\to c}} + \boldsymbol{A_{c\to p}} + \boldsymbol{A_{p\to c}}$
$\boldsymbol{H_o} = \text{softmax}(\frac{\tilde{\boldsymbol{A}}}{\sqrt{3d}})\boldsymbol{V_c}$

**Output:** $\boldsymbol{H_o}$