

# Transformer to BERT

20191106

MEANIMO\_NLP

한승우

# 목차

## **0. Attention**

0-1. Sequence to Sequence

0-2. Attention

0-3. Attention in sequence to sequence

## **1. Transformer**

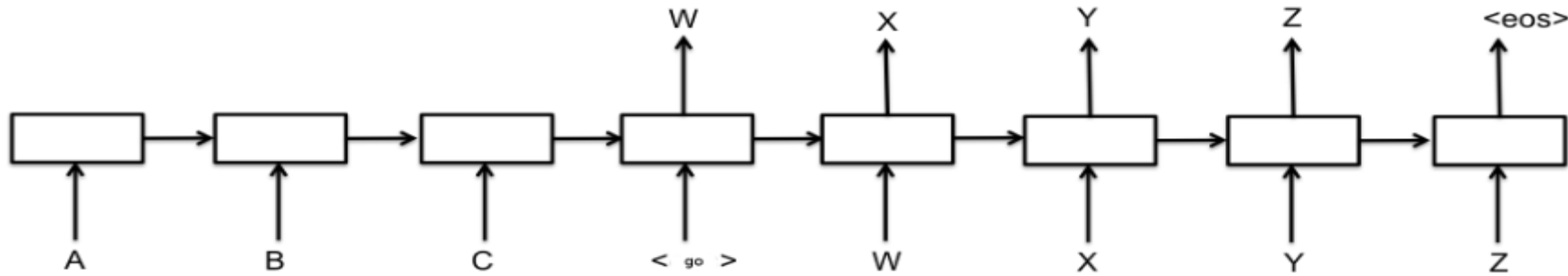
1-1 Scaled Dot-Product Attention

1-2 MultiHead Attention

1-3 Pointwise Feedforward Networks

## **2. BERT**

# Sequence to Sequence



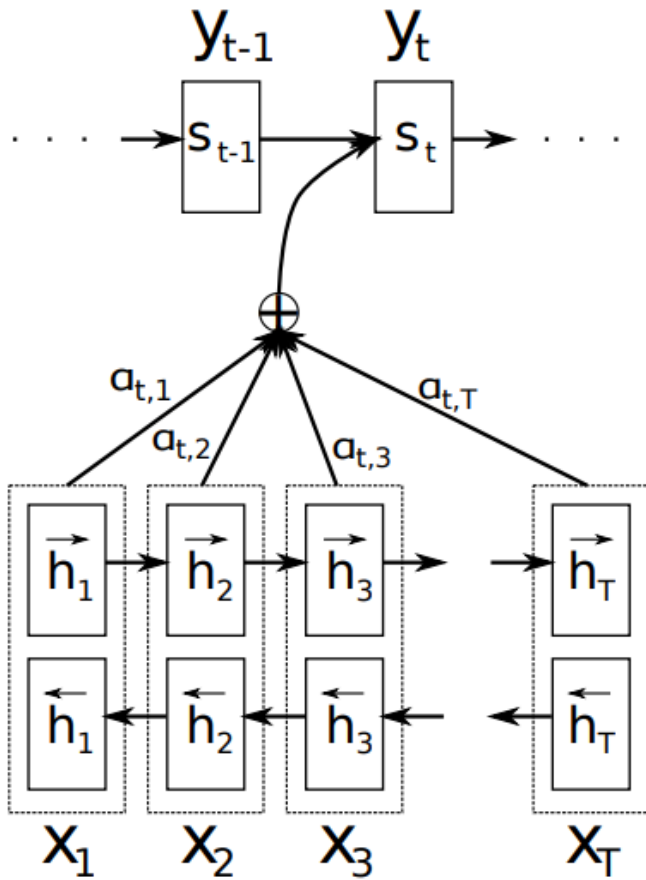
Encoder : Input sentence [A,B,C] 의 언어적 지식을 학습

Decoder : Output(target) sentence [W,X,Y,Z]의 언어적 지식을 학습

Sequence to Sequence 가 학습하는 기준은  $\text{maximize } \sum P_{\theta}(y_{1:m}|x_{1:n})$   
x:1:n과 y1:m의 상관성을 최대화 하는 것

$$P(y_{1:m}|x_{1:n}) = \prod_i P(y_i|y_{1:i-1}, c)$$

# Sequence to Sequence with Attention



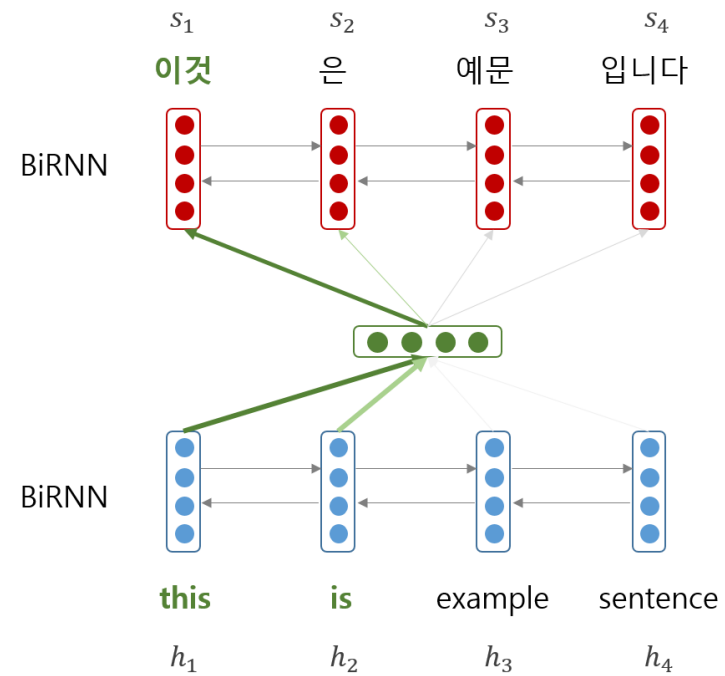
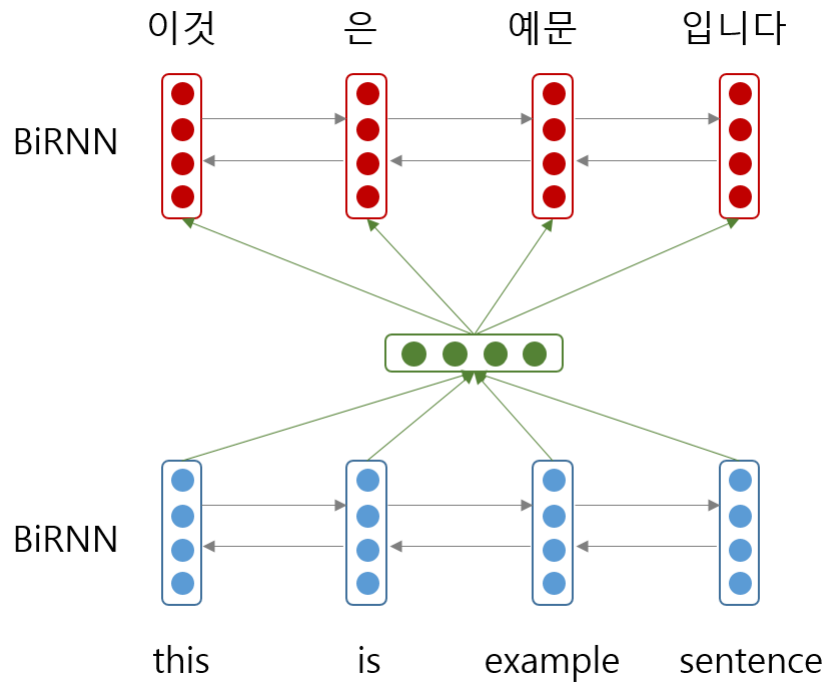
“하나의 Context Vector  $c$  로는 Decoder의 각 단어를 충분히 표현할 수 없다.”

그래서,

$$c_i = \sum_j a_{ij} * h_j$$

- $a_{ij}$  는  $y_i$  를 선택할 때 encoder RNN의  $h_j$  를 얼마나 이용할지에 대한 값
- $h_j$  는 Encoder의 hidden state
- $a_{ij} = \frac{\exp(e_{ij})}{\sum(\exp(e_{ij}))}$
- $e_{ij} = f(s_{i-1}, h_j)$

# Sequence to Sequence with Attention



# Transformer

## Seq2Seq 모델들의 한계점

1. 모델의 크기가 크다.
2. RNN은 반드시 sequence의 마지막 부분까지 계산이 완료되어야 한다.
3. Sequence에 대한 작업을 병렬적으로 진행 불가
4. Long dependency문제

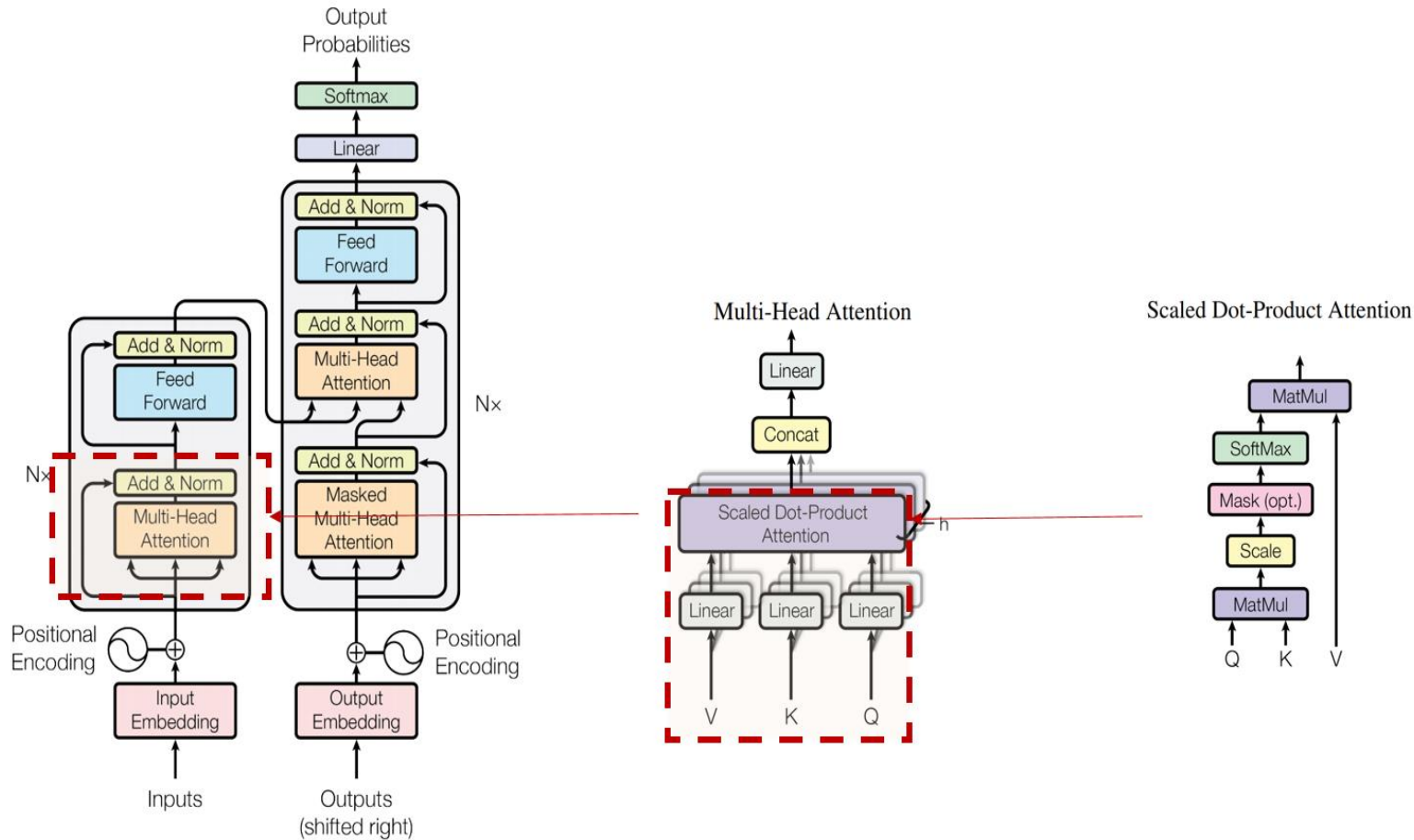
**“Encoder 와 Decoder에 모두 Attention을 이용하자!!”**

# Transformer

## Transformer 네트워크의 구성

1. Scaled Dot-Product Attention
2. MultiHead Attention
3. Pointwise Feedforward Networks

# Seq2Seq 과 Transformer



**Seq2Seq  $\sim$  Transformer**

$s_{i-1} = \text{Query}$   
 $h_j = \text{Key, Value}$

**Seq2Seq  $\neq$  Transformer**

Additive Attention  
 $\neq$   
Multiplicative Attention

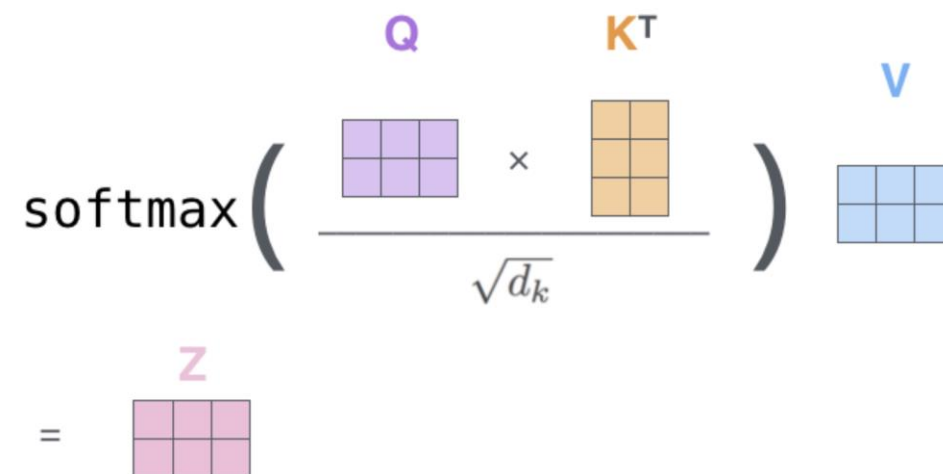


# Scaled Dot-Product Attention

$$\mathbf{X} \times \mathbf{W}^Q = \mathbf{Q}$$


$$\mathbf{X} \times \mathbf{W}^K = \mathbf{K}$$


$$\mathbf{X} \times \mathbf{W}^V = \mathbf{V}$$

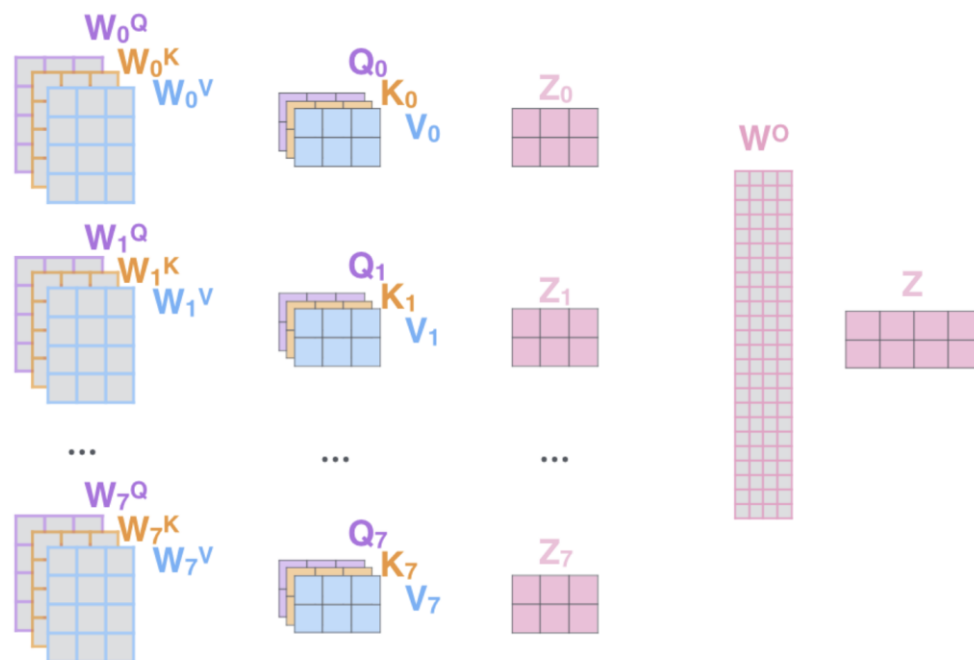

$$\text{softmax}\left(\frac{\mathbf{Q} \times \mathbf{K}^T}{\sqrt{d_k}}\right) \mathbf{V}$$


1. Dot-Product
2. Attention
3. Scaled

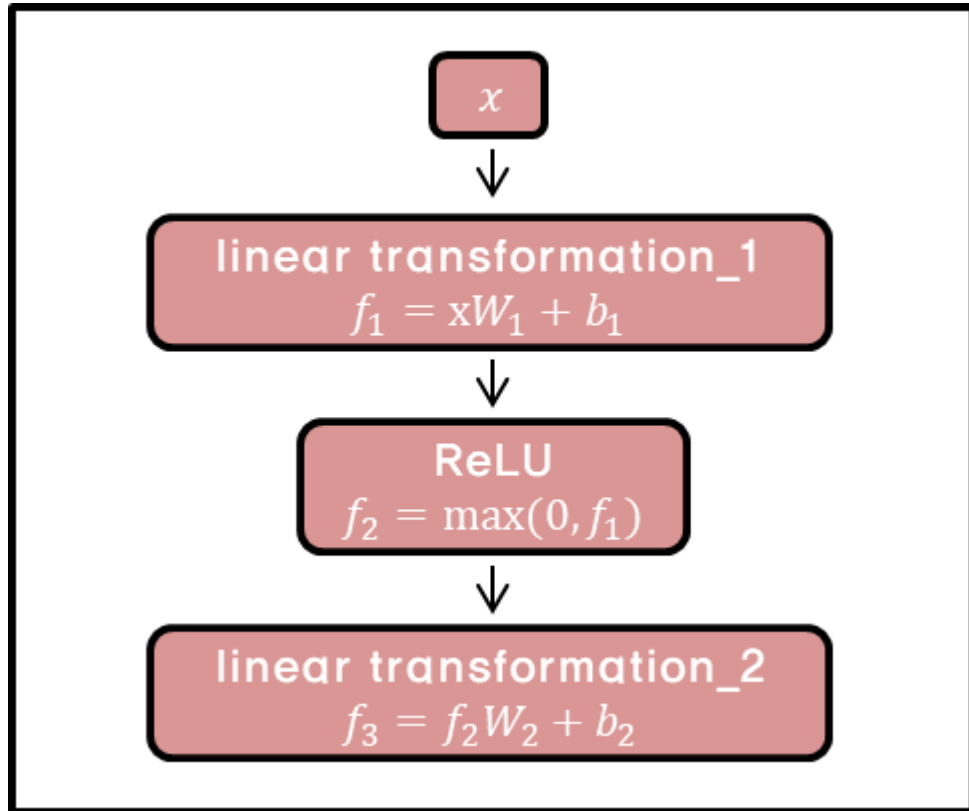
# Multi-Head Attention

$$\text{MultiHead}(Q,K,V) = \text{Concat}(\text{head}_1, \text{head}_2, \dots, \text{head}_h) W^O$$

\*  $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$



# Pointwise Feedforward Networks



$$\text{FFN}(x) = \max(0, x * W_1 + b_1) W_2 + b_2$$

# BERT



# Masked Language Model

1. 단방향 언어 모델 : 나는 어제 \_\_\_\_.
2. 양방향 언어 모델 : 나는 어제 \_\_\_\_ 먹었다.

$$\text{GPT self-attention SDP} = \begin{matrix} & \text{꿈} \\ \text{보다} & \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0.9 & 0 & 0 \end{pmatrix} \\ \text{해몽} & \end{matrix} \begin{pmatrix} V_{\text{꿈}} \\ V_{\text{보다}} \\ V_{\text{해몽}} \end{pmatrix} = \begin{matrix} & \text{꿈} \\ \text{보다} & \begin{pmatrix} 0V_{\text{꿈}} & 0V_{\text{보다}} & 0V_{\text{해몽}} \\ 1V_{\text{꿈}} & 0V_{\text{보다}} & 0V_{\text{해몽}} \\ 0.9V_{\text{꿈}} & 0.1V_{\text{보다}} & 0V_{\text{해몽}} \end{pmatrix} \\ \text{해몽} & \end{matrix}$$

$$\text{BERT self-attention SDP} = \begin{matrix} & \text{꿈} \\ \text{보다} & \begin{pmatrix} 0.3 & 0.1 & 0.6 \\ \dots & \dots & \dots \\ \dots & \dots & \dots \end{pmatrix} \\ \text{해몽} & \end{matrix} \begin{pmatrix} V_{\text{꿈}} \\ V_{\text{보다}} \\ V_{\text{해몽}} \end{pmatrix} = \begin{matrix} & \text{꿈} \\ \text{보다} & \begin{pmatrix} 0.3V_{\text{꿈}} + 0.1V_{\text{보다}} + 0.6V_{\text{해몽}} \\ \dots \\ \dots \end{pmatrix} \\ \text{해몽} & \end{matrix}$$

# BERT Model Structure

