Transformer to BERT

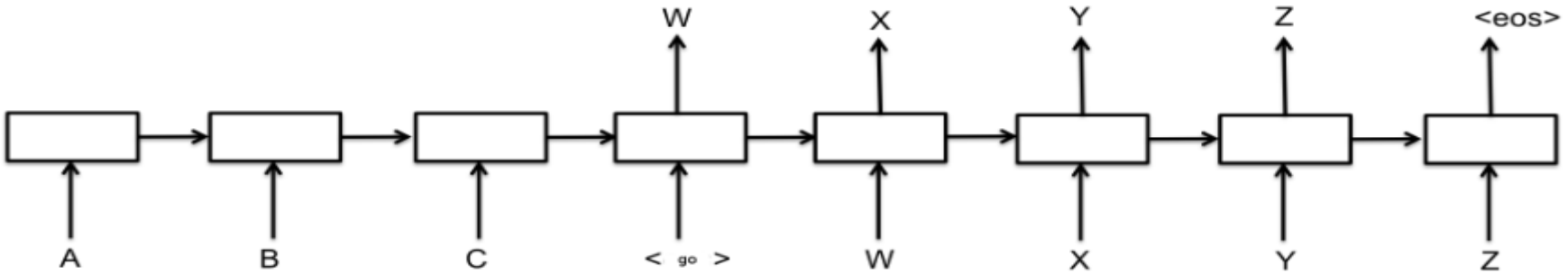
20191106

MEANIMO_NLP

한승우

목차

Sequence to Sequence

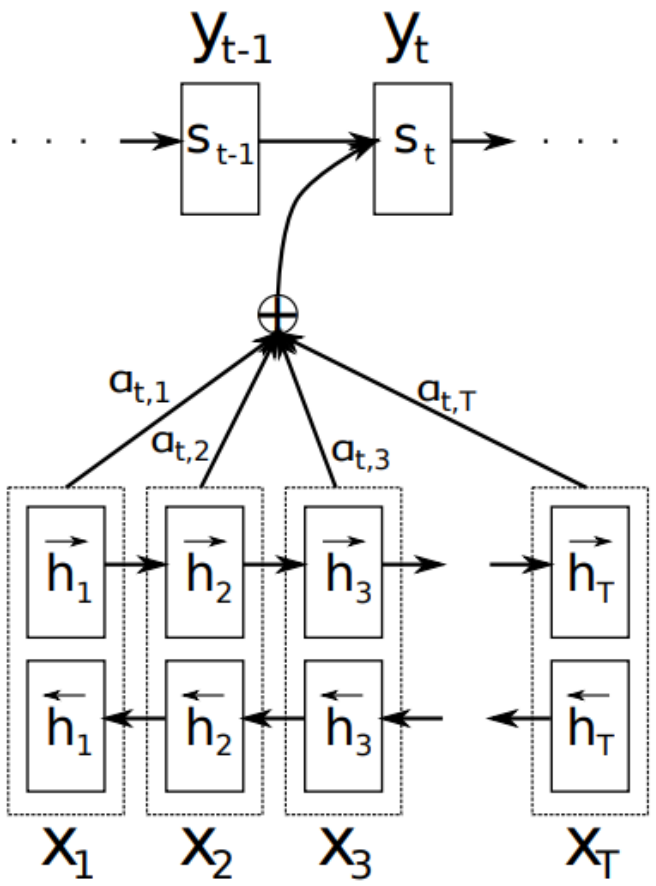


Encoder : Input sentence [A,B,C] 의 언어적 지식을 학습
Decoder : Output(target) sentence [W,X,Y,Z]의 언어적 지식을 학습

Sequence to Sequence 가 학습하는 기준은 $maximize \sum P_\theta \left(y_{1:m} | x_{1:n}\right)$
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



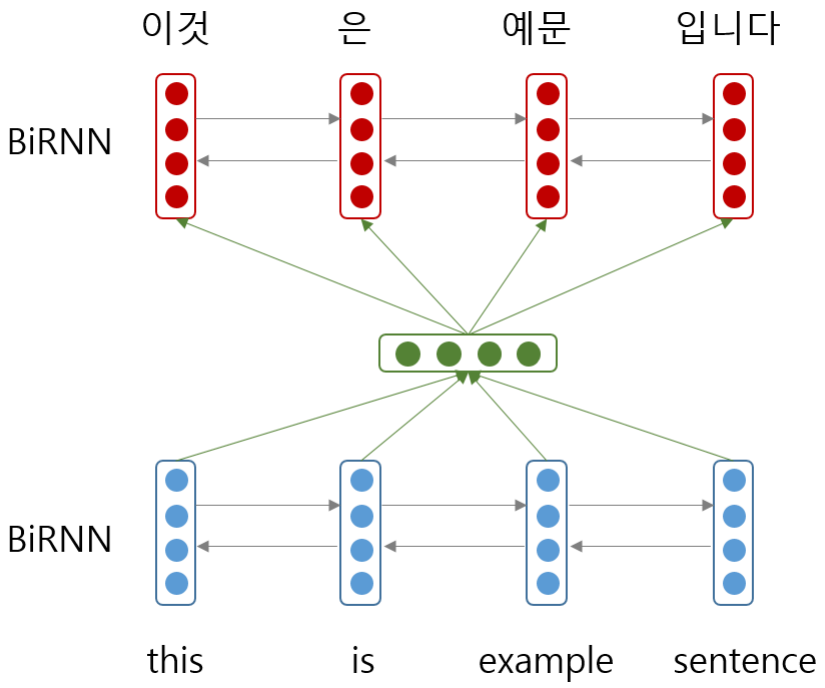
"하나의 Context Vector c 로는 Decoder의 각 단어를 충분히 표현할 수 없다."

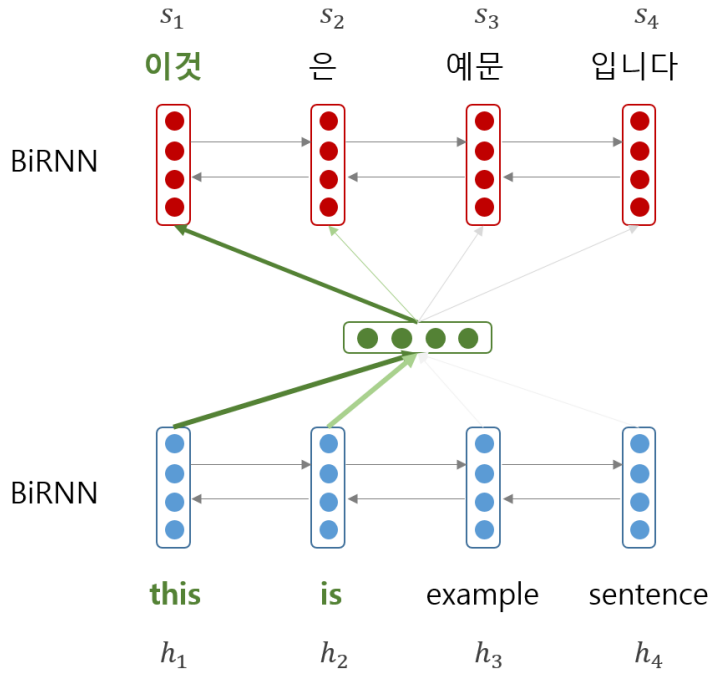
그래서,

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

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

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

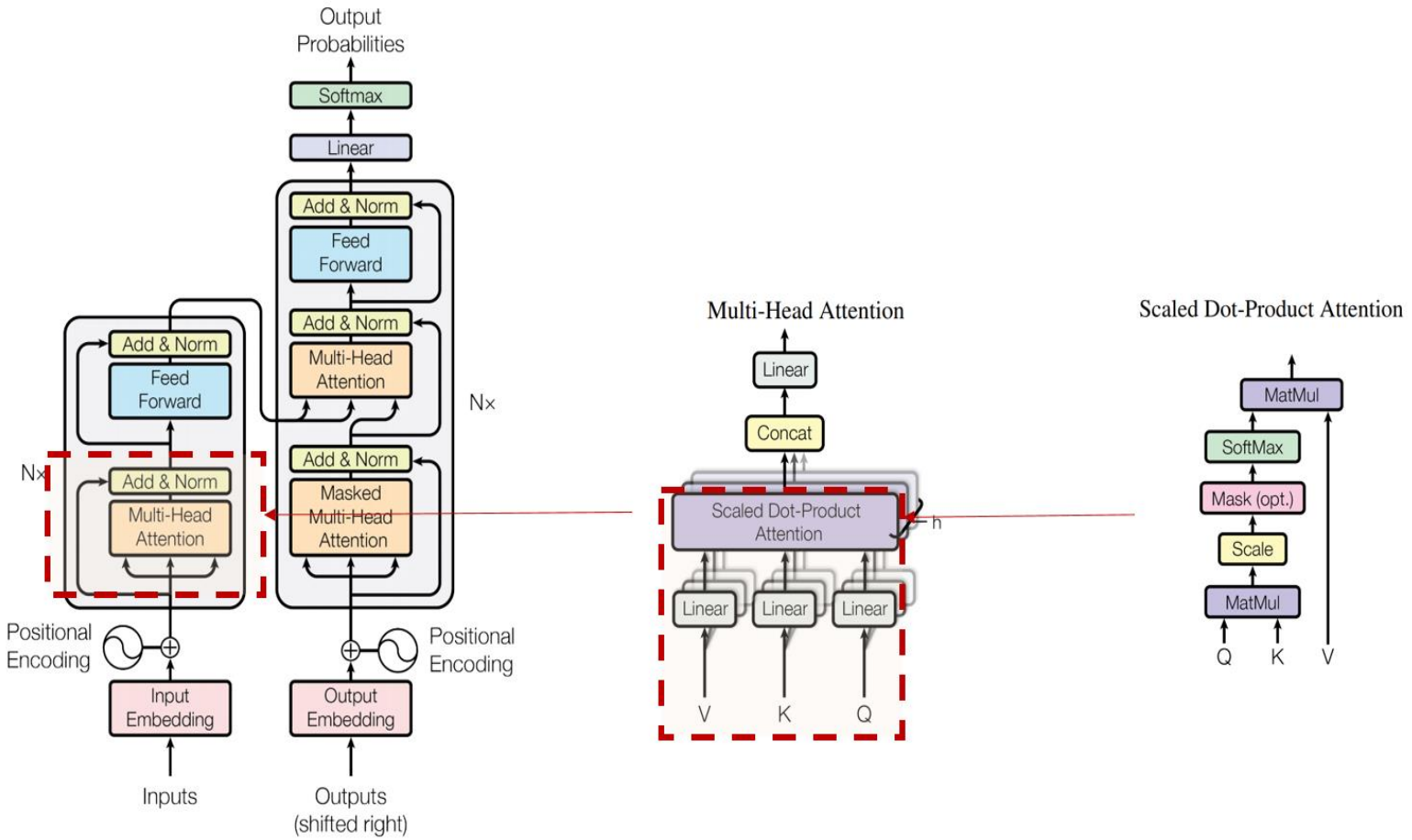
“Encoder 와 Decoder에 모두 Attention을 이용하자!!”

Transformer

Transformer 네트워크의 구성

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

Seq2Seq 과 Transformer



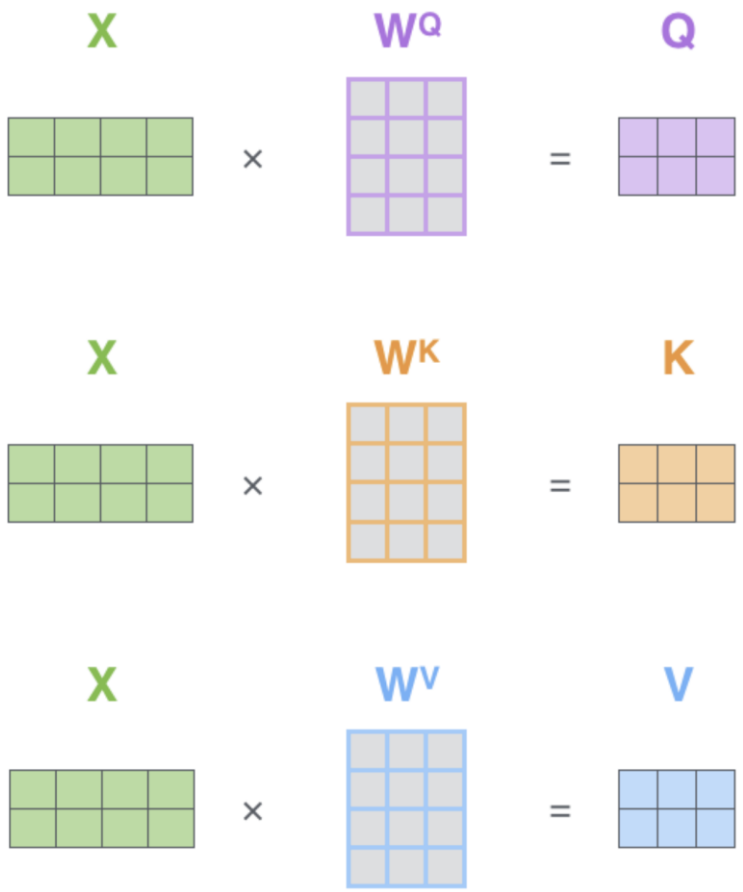
Seq2Seq ~= Transformer

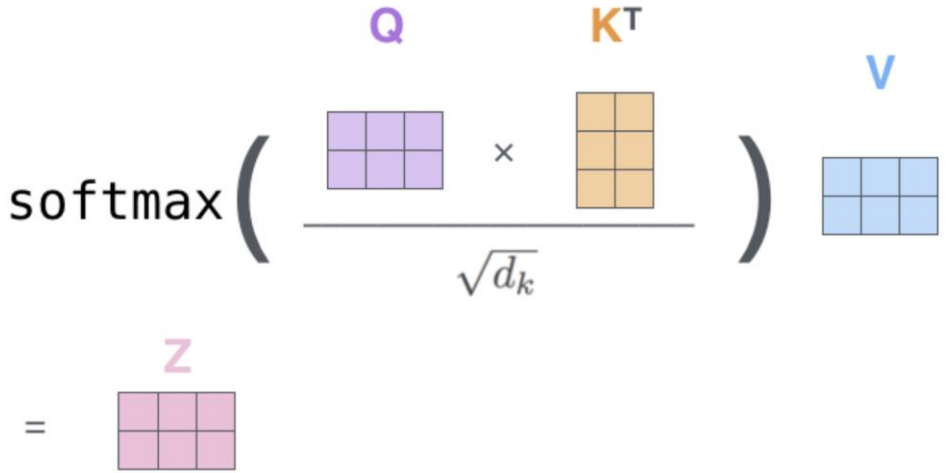
$s_{i-1} = Query$
$\mathrm{h_j} = Key, \mathrm{Value}$

Seq2Seq != Transformer

Additive Attention
!=
Multiplicative Attention

Scaled Dot-Product Attention



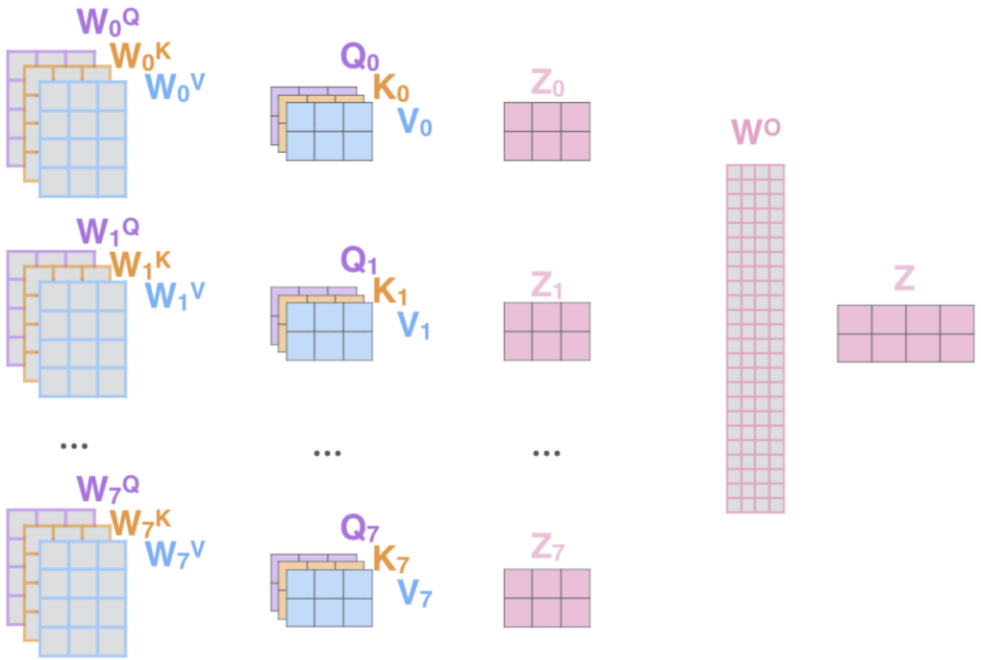


1. Dot-Product
2. Attention
3. Scaled

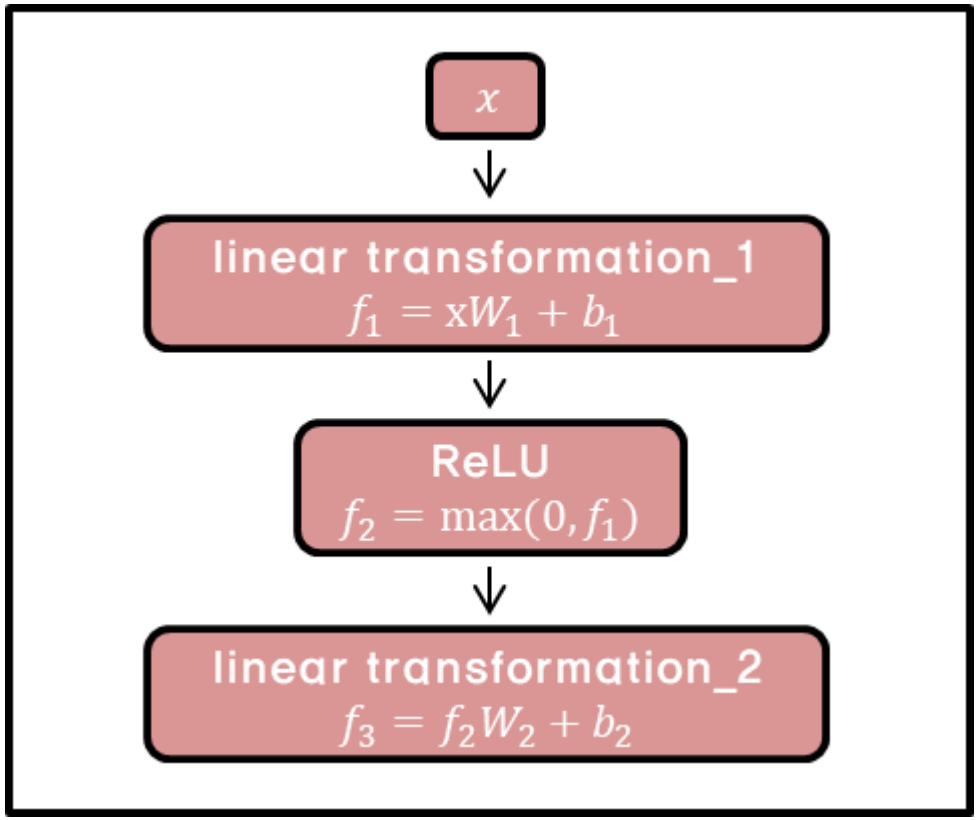
Multi-Head Attention

MultiHead(Q,K,V) = Concat($head_1, head_2 \ldots., head_h)W^O$

* $head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$



Pointwise Feedforward Networks



FFN(x) =max(0,x*W1+b1)W2 +b2

BERT

Transformer

Transformer

Transformer