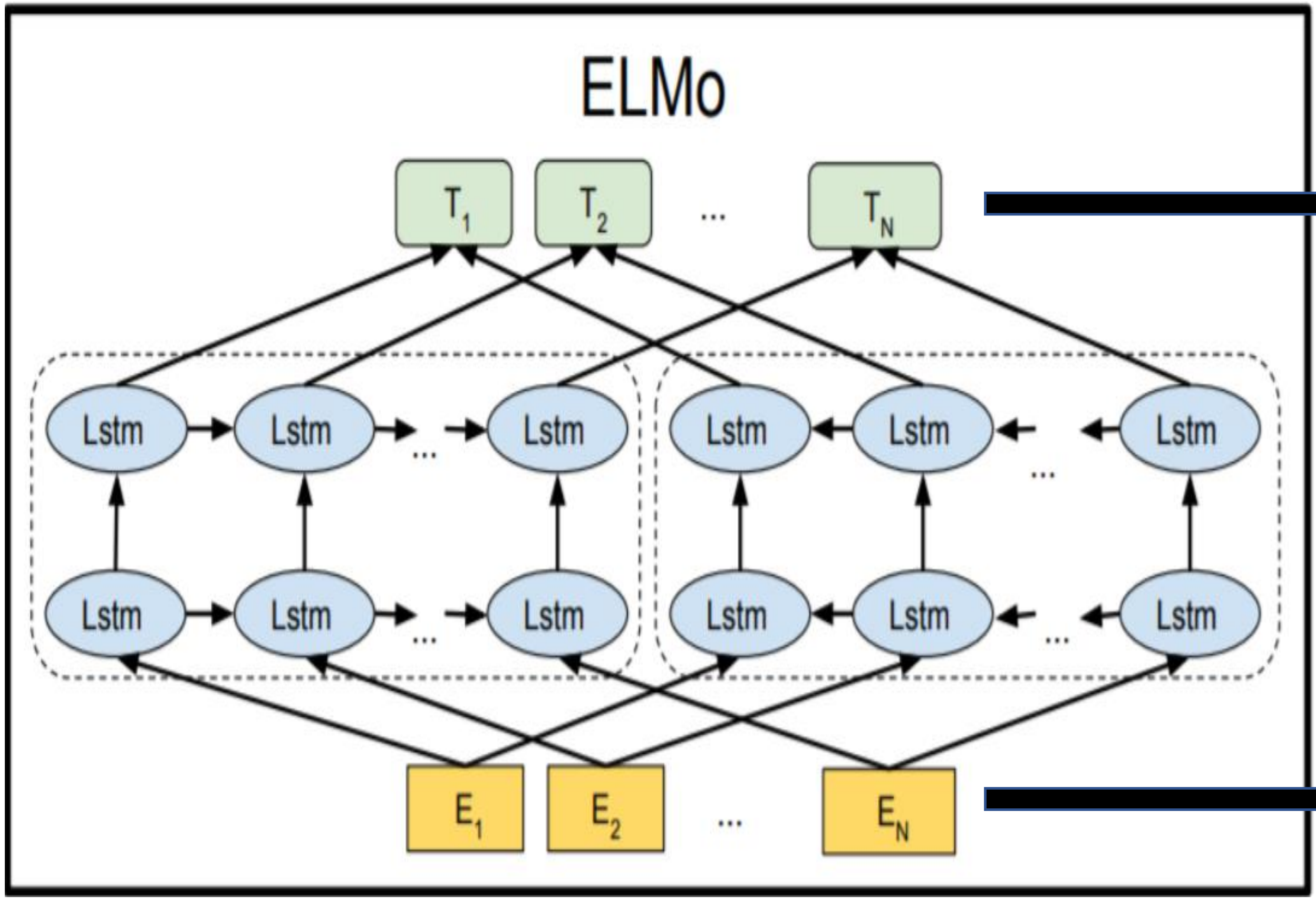
NLP2팀 4주차(2020.05.21.)

한국어 임베딩
5.4 ELMo ~
5.5 트랜스포머 네트워크

황인택

5.4 **E**mbeddings from **L**anguage **Mo**dels

단어 시퀀스가 얼마나 자연스러운지 확률 값을 부여



ELMo

다운스크림 태스크:
가중합

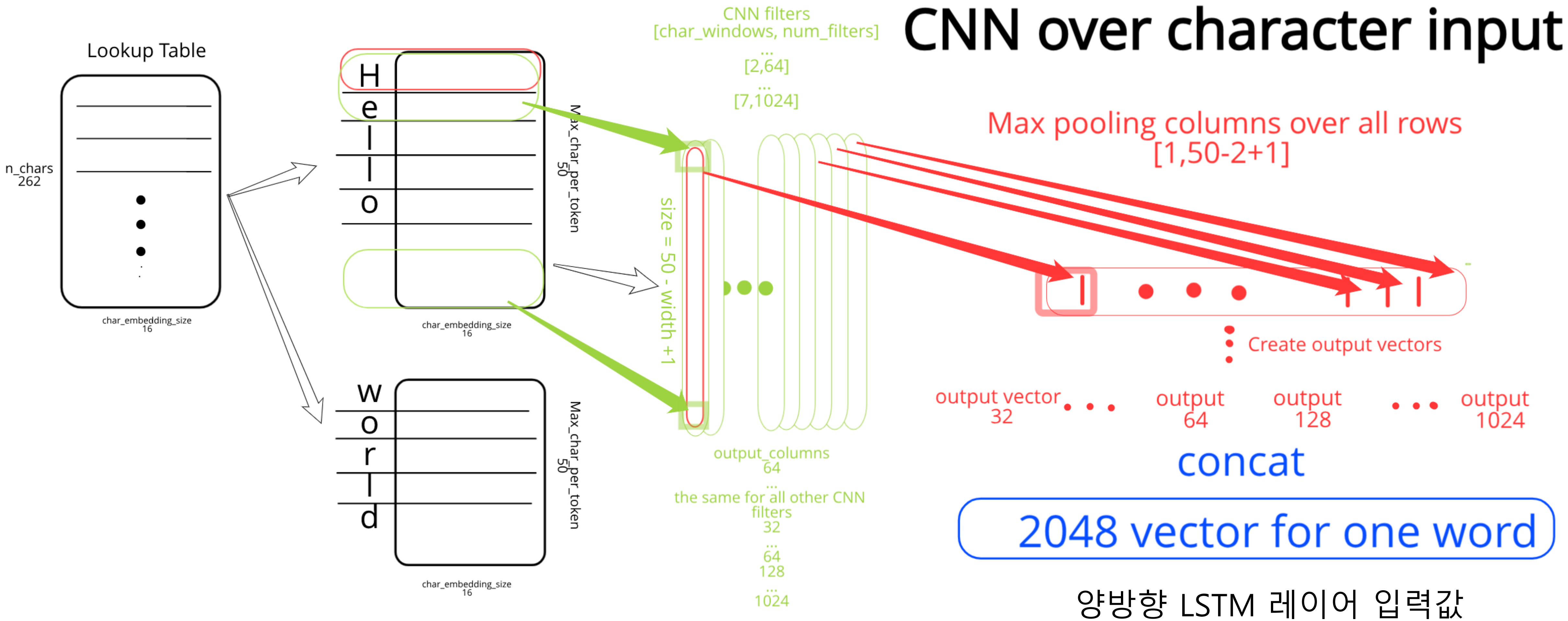
양방향 LSTM

문자 단위
컨볼루션 신경망

Pre-train
(사전학습)

5.4 **E**mbeddings from **L**anguage **Mo**dels

Pre-train(사전학습): 문자 단위 컨볼루션 신경망



양방향 LSTM 레이어 입력값

5.4 **E**mbeddings from **L**anguage **Mo**dels

Bi-LSTM의 input으로 쓰기 전에 하이웨이 네트워크와 차원 조정(projection)을 거침

하이웨이 네트워크

$$y = H(x, W_H) \cdot T(x, W_T) + x \cdot C(x, W_C) = \underbrace{H(x, W_H) \cdot T(x, W_T)}_{\text{얼마나 변형할지}} + \underbrace{x \cdot (1 - T(x, W_T))}_{\text{얼마나 변형하지 않을지}}$$

$H(x, W_H)$: 피드포워드 네트워크, 점곱에 relu

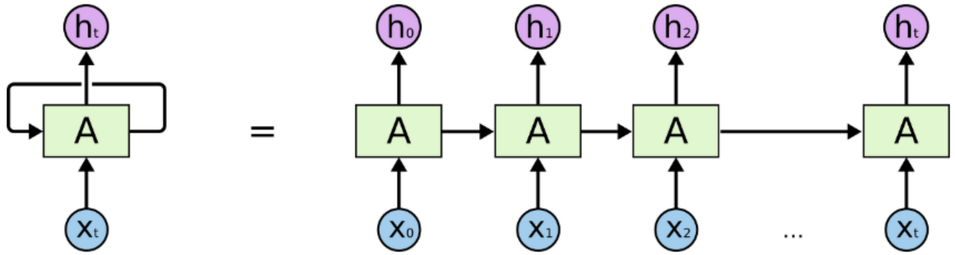
$T(x, W_T)$: 변형 게이트(transform gate), 점곱에 시그모이드

$C(x, W_C)$: 캐리 게이트(carry gate), (1-transform gate)

5.4 **E**mbeddings from **L**anguage **Mo**dels

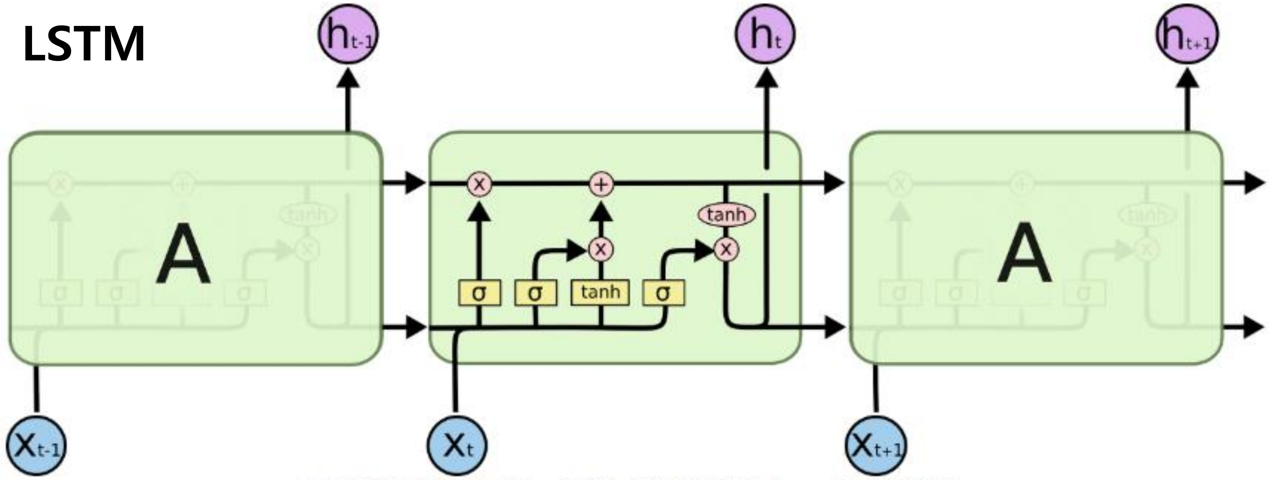
Pre-train(사전학습): bi-LSTM

RNN



풀어 헤쳐진 RNN

LSTM



LSTM의 반복 모듈에는 4개의 상호작용하는 layer가 들어있다.

Gated Recurrent Unit
[Cho et al., EMNLP2014;
Chung, Gulcehre, Cho, Bengio, DLUFL2014]

$$h_t = u_t \odot \tilde{h}_t + (1 - u_t) \odot h_{t-1}$$
$$\tilde{h} = \tanh(W[x_t] + U(r_t \odot h_{t-1}) + b)$$
$$u_t = \sigma(W_u[x_t] + U_u h_{t-1} + b_u)$$
$$r_t = \sigma(W_r[x_t] + U_r h_{t-1} + b_r)$$

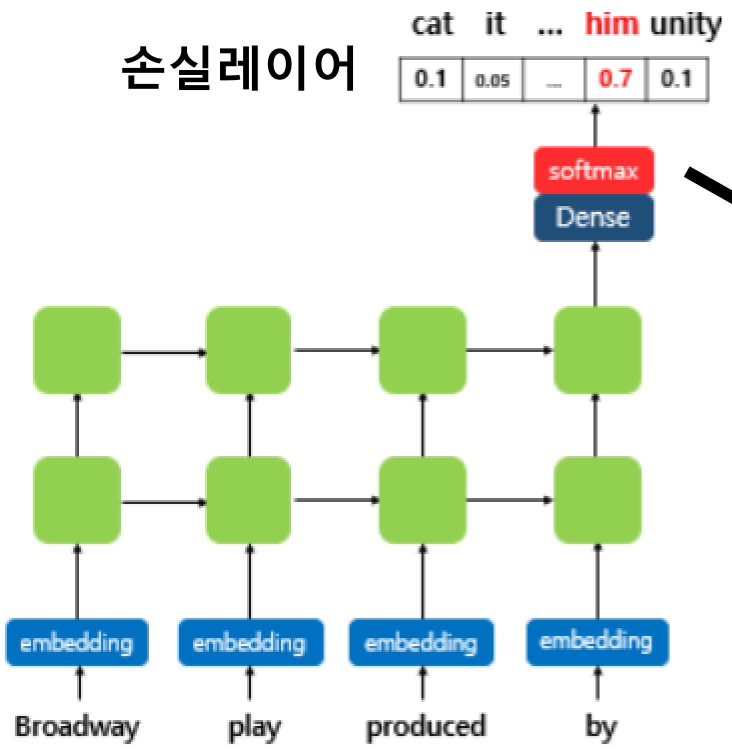
Summing previous & new candidate hidden states gives direct gradient flow & more effective memory

Long Short-Term Memory
[Hochreiter & Schmidhuber, NC1999;
Gers, Thesis2001]

$$h_t = o_t \odot \tanh(c_t)$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$
$$\tilde{c}_t = \tanh(W_c[x_t] + U_c h_{t-1} + b_c)$$
$$o_t = \sigma(W_o[x_t] + U_o h_{t-1} + b_o)$$
$$i_t = \sigma(W_i[x_t] + U_i h_{t-1} + b_i)$$
$$f_t = \sigma(W_f[x_t] + U_f h_{t-1} + b_f)$$

5.4 **E**mbeddings from **L**anguage **Mo**dels

Pre-train(사전학습): bi-LSTM



원핫벡터로 크로스 엔트로피 계산 후
지속적으로 업데이트

메모리 소모가 심해 샘플링 후 softmax 계산

5.4 **E**mbeddings from **L**anguage **Mo**dels

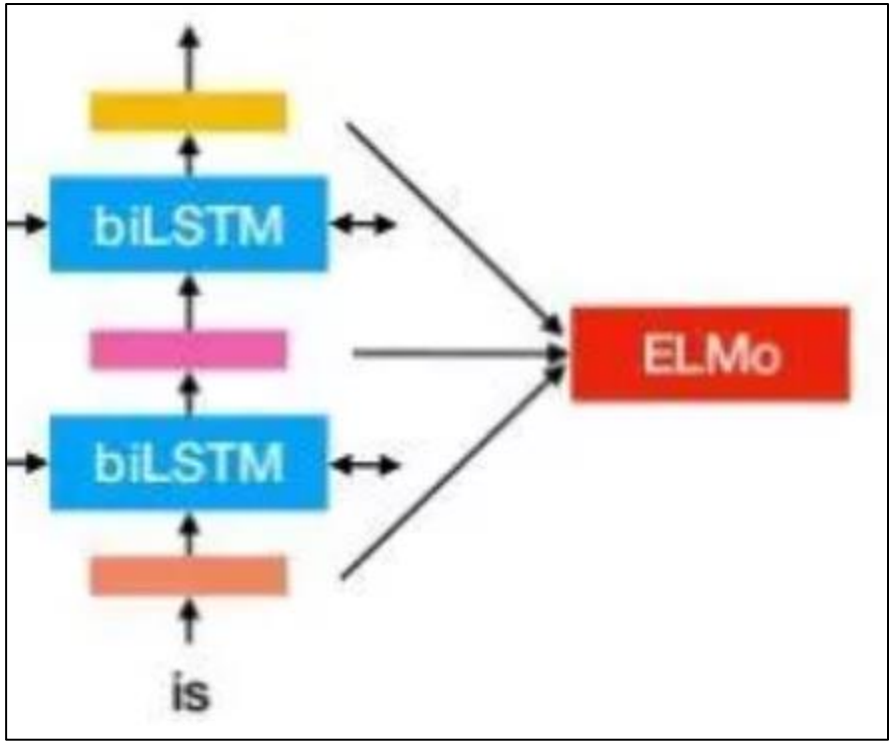
다운스트림: ELMo 임베딩

$$\begin{aligned} R_k &= \{\mathbf{x}_k^{LM}, \overrightarrow{\mathbf{h}}_{k,j}^{LM}, \overleftarrow{\mathbf{h}}_{k,j}^{LM} \mid j = 1, \ldots, L\} \\ &= \{\mathbf{h}_{k,j}^{LM} \mid j = 0, \ldots, L\}, \end{aligned}$$

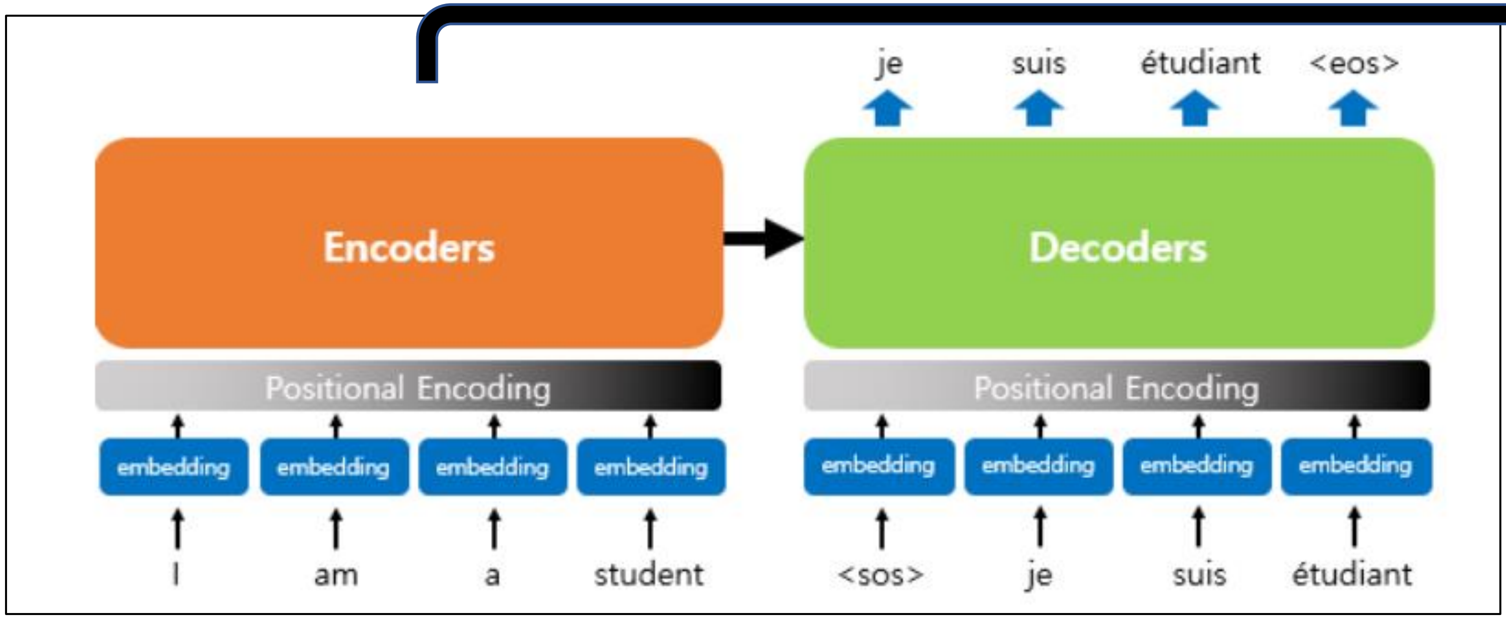
$$\mathbf{ELMo}_k^{task} = E(R_k; \Theta^{task}) = \gamma^{task} \sum_{j=0}^{L} s_j^{task} \mathbf{h}_{k,j}^{LM}.$$

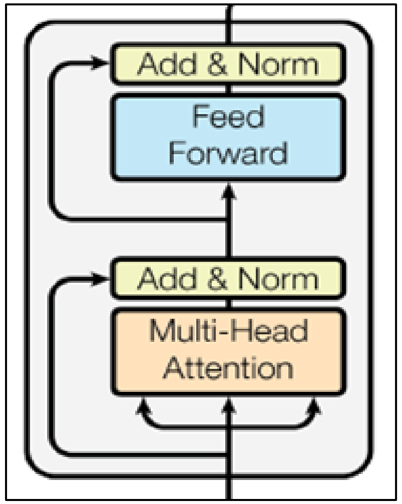
각 task의 특성에 맞게 여러 층, 2가지 방향에서 얻어진 hidden state들을 조합
얼마나 조합할 지도 학습 – r, s

s: softmax로 각 레이어마다 적용할 가중치
r: 벡터의 크기를 결정하는 태스크별 가중치

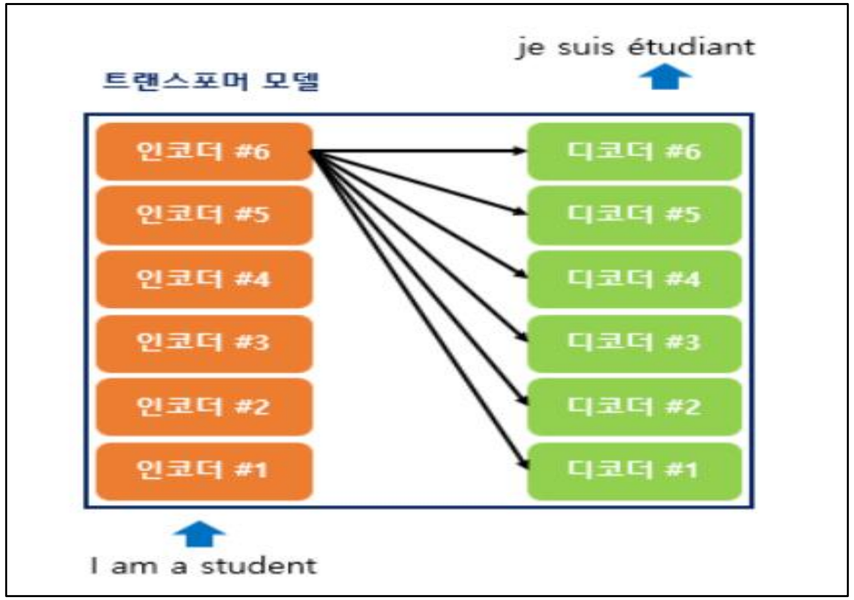


5.5 트랜스포머 네트워크





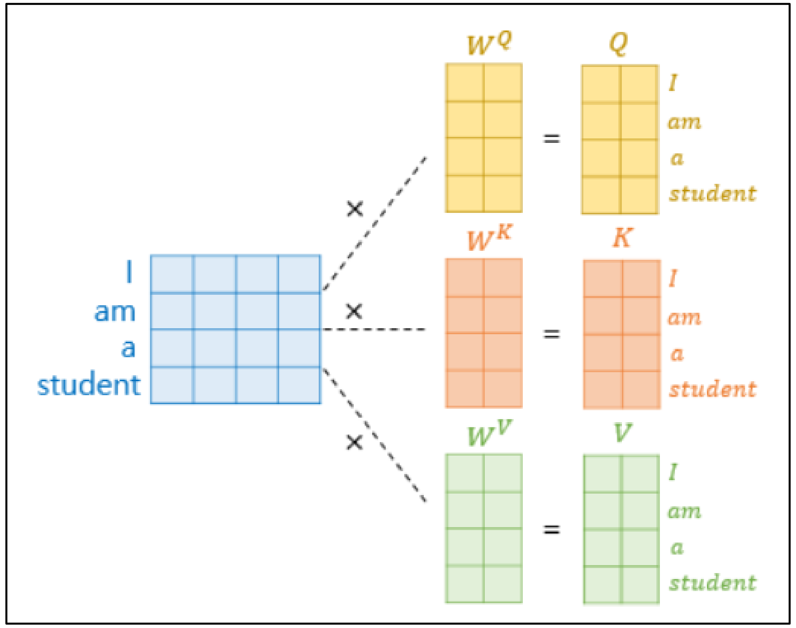
트랜스포머 블록

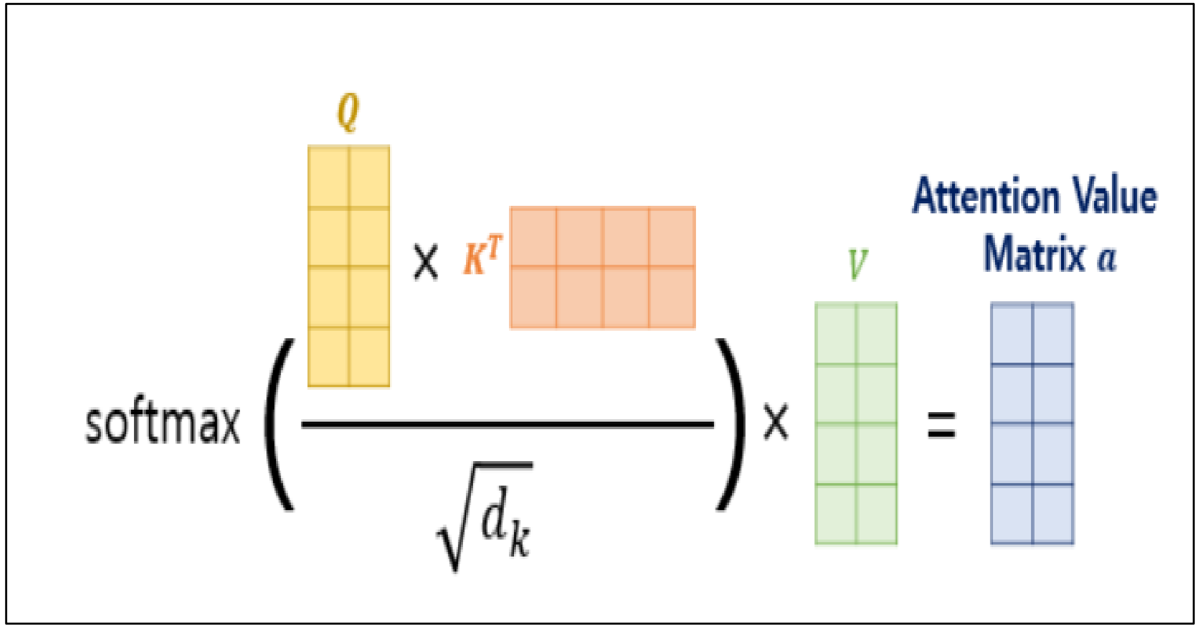


5.5 트랜스포머 네트워크

Scaled Dot-Product Attention

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



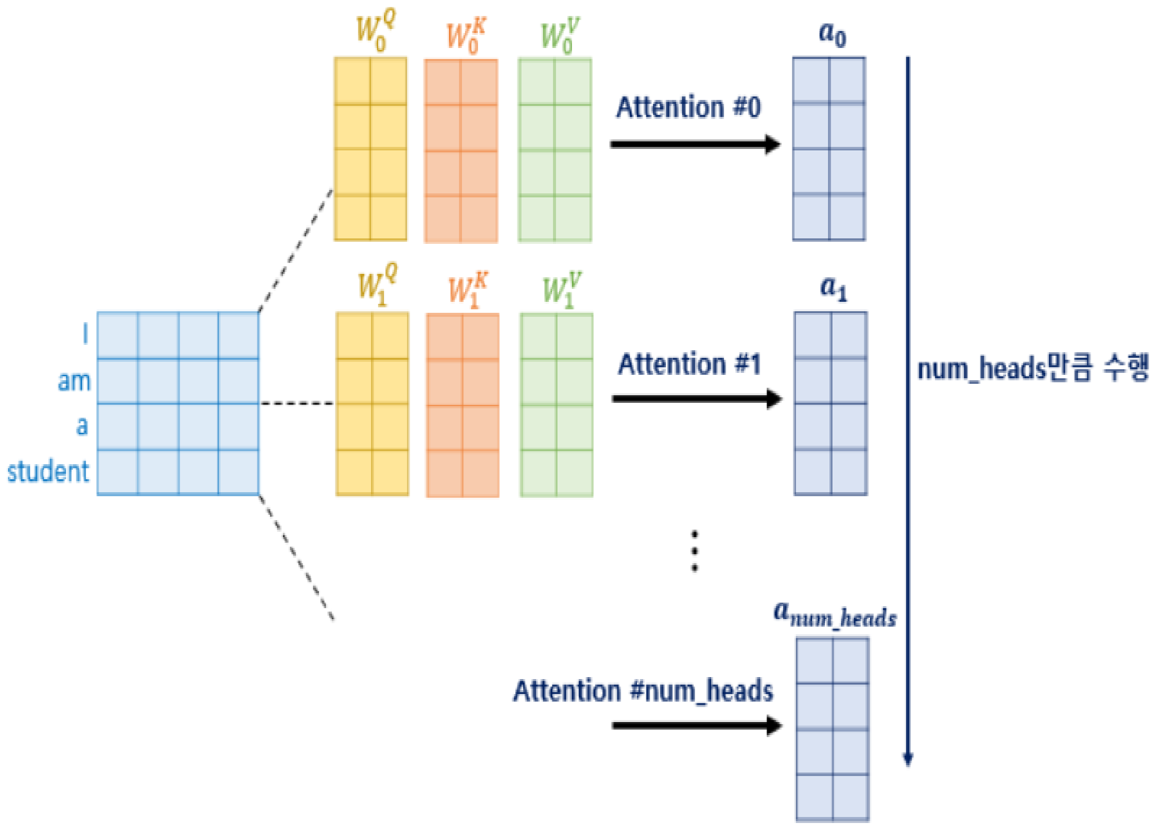


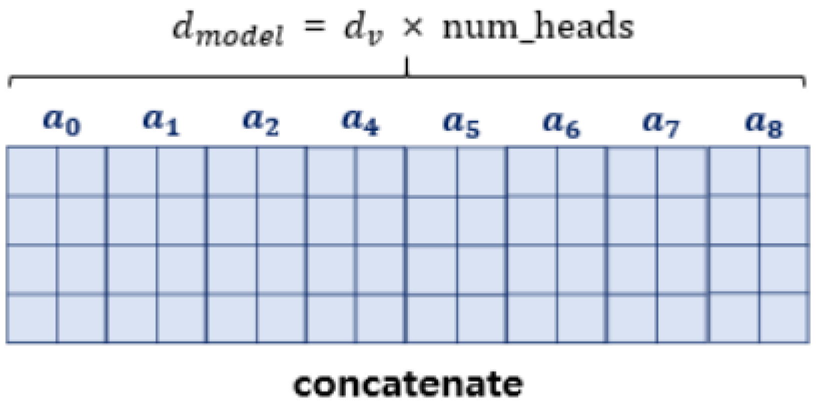
셀프 어텐션

로컬 문맥만 살피는 CNN과 달리, 모든 단어 쌍 관계 파악 가능
RNN과 달리, 긴 시퀀스에서도 이전 입력 단어를 까먹지 않음

5.5 트랜스포머 네트워크

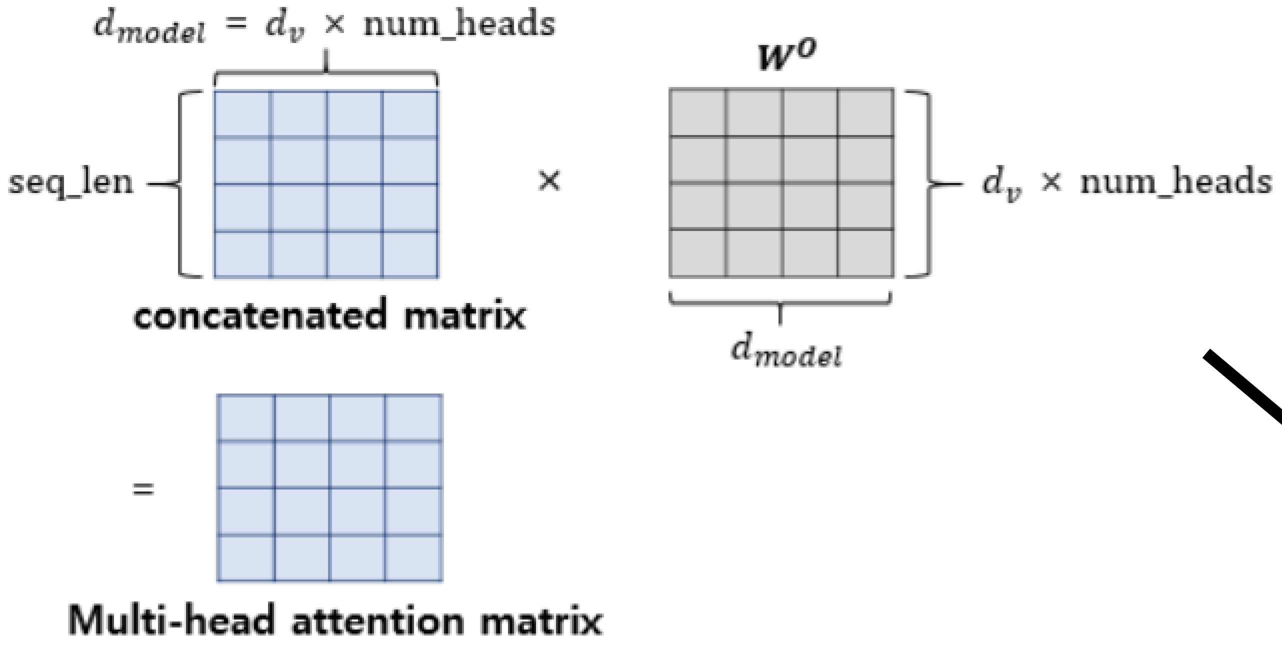
Multi-Head Attention





5.5 트랜스포머 네트워크

Multi-Head Attention

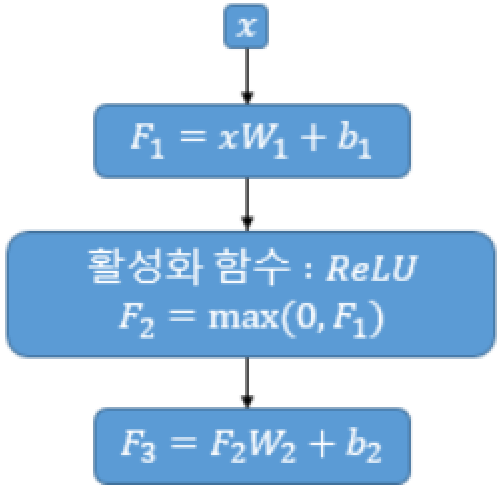


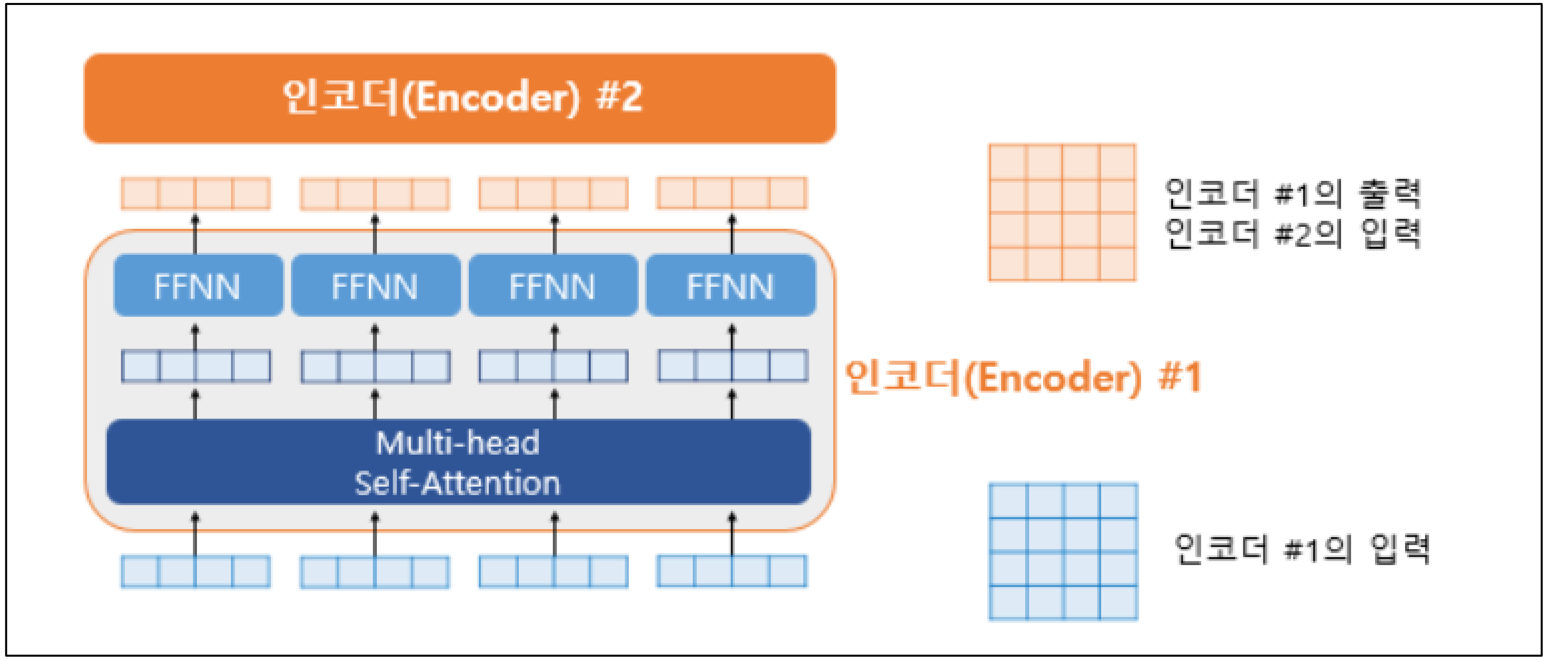
W 내적으로 트랜스포머 블록 입력 행렬의 크기와 맞춰줌

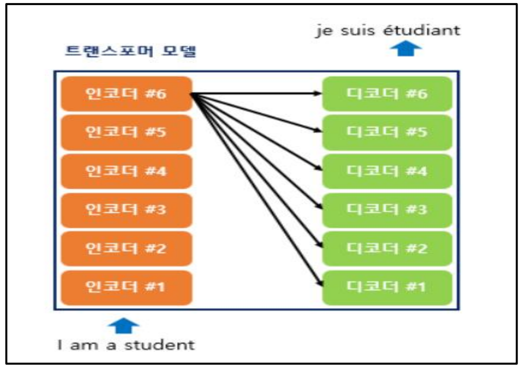
5.5 트랜스포머 네트워크

Position-wise FeedForward Neural Networks(FFNN, FFN)

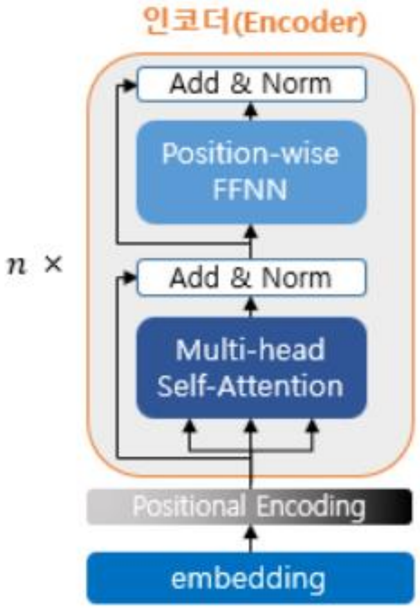
$$FFNN(x) = MAX(0, xW_1 + b_1)W_2 + b_2$$



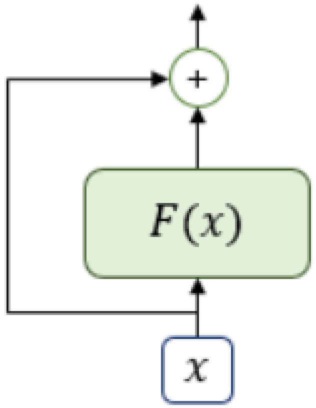




5.5 트랜스포머 네트워크



1) 잔차연결(add, residual connection)



2) 층 정규화(Norm, Layer Normalization)

$$\mu^l = \frac{1}{H}\sum_{i=1}^{H} a_i^l \qquad \sigma^l = \sqrt{\frac{1}{H}\sum_{i=1}^{H}\left(a_i^l - \mu^l\right)^2}$$

히든 유닛의 개수