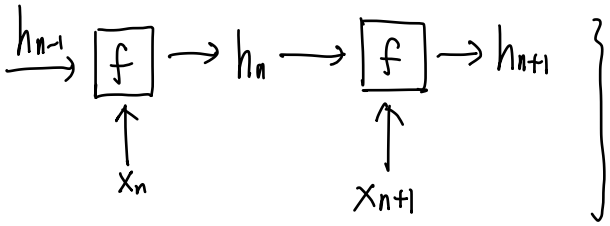
RNN

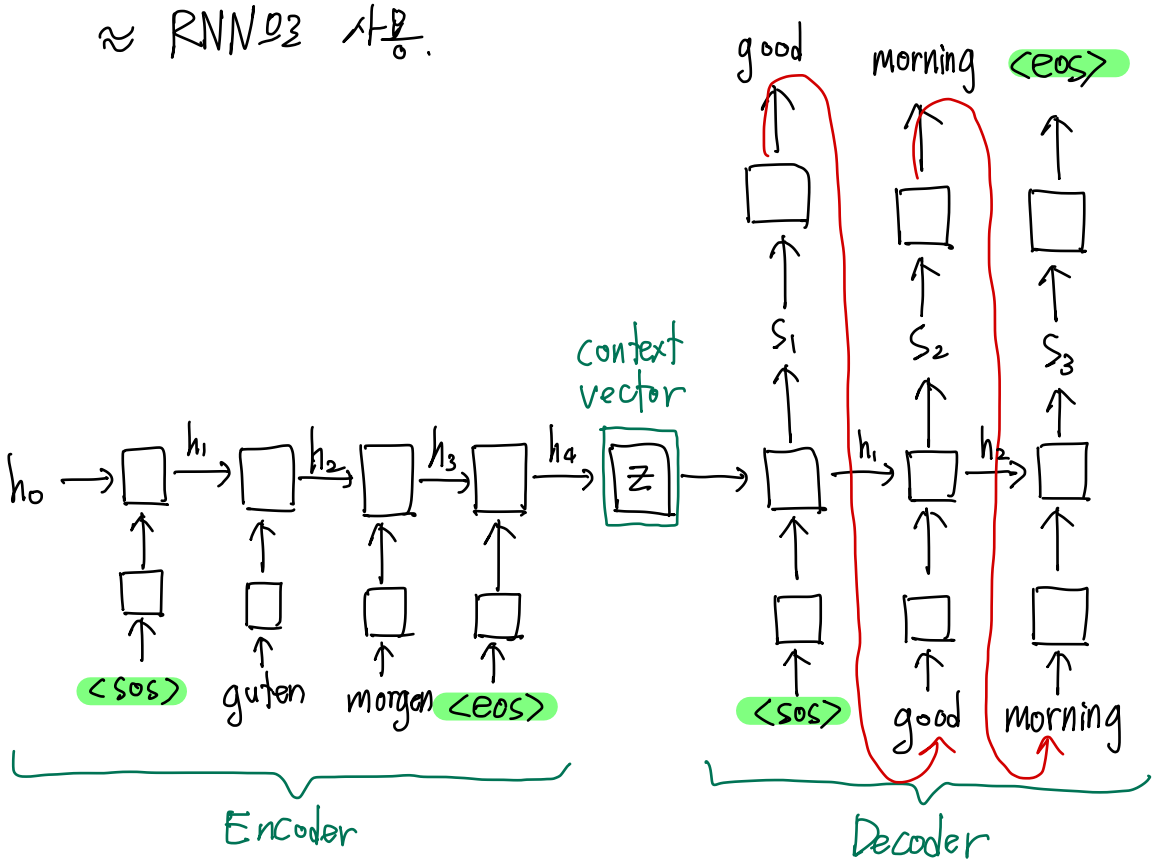


$x_1 \sim x_n$이 요약된 h_n과 x_{n+1}을 받아서 h_{n+1}을 만드는 함수 f를 구현한 것.

$f(x_n, h_{n-1})$

Seq2Seq

- 번역 machine.
- 입력/출력 문장의 단어 수 다름.
- RNN encoder , RNN decoder 사용.
- ▭ : Special token
- 단점
  - ① Hidden state가 멀리있는 내용을 기억해내기 못함. (Context vector)
  - ② 출력한 것을 다시 입력함.
- LSTM : 장기/단기 기억을 구분해서 관리하는 RNN.
  ≈ RNN의 사용.



Embedding Layer

- $\mathbb{R}^{\text{사전차원}} \longrightarrow \mathbb{R}^{500}$
- 사전 차원으로 one-hot encoding 돼 있는 벡터를 W곱해서 차원 축소.

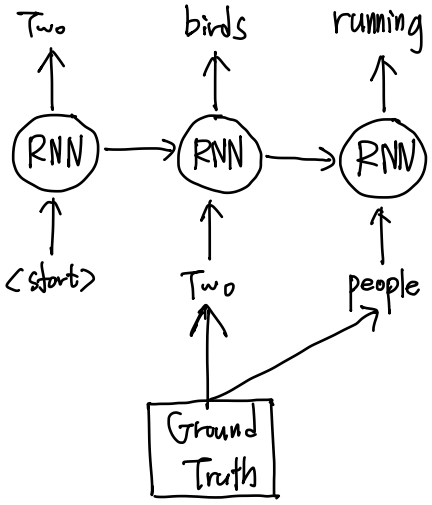
ex) $\begin{bmatrix}0\\0\\\vdots\\0\end{bmatrix} \in \mathbb{R}^{\text{사전}} \longrightarrow W\begin{bmatrix}0\\0\\\vdots\\0\end{bmatrix} \in \mathbb{R}^{500}$

Word2vec

Teacher Forcing

- Decoder에서 사용하는 기법.
- 예측값 말고 정답 입력.
- 다음 단어 맞추기

  ⇒ Language Model



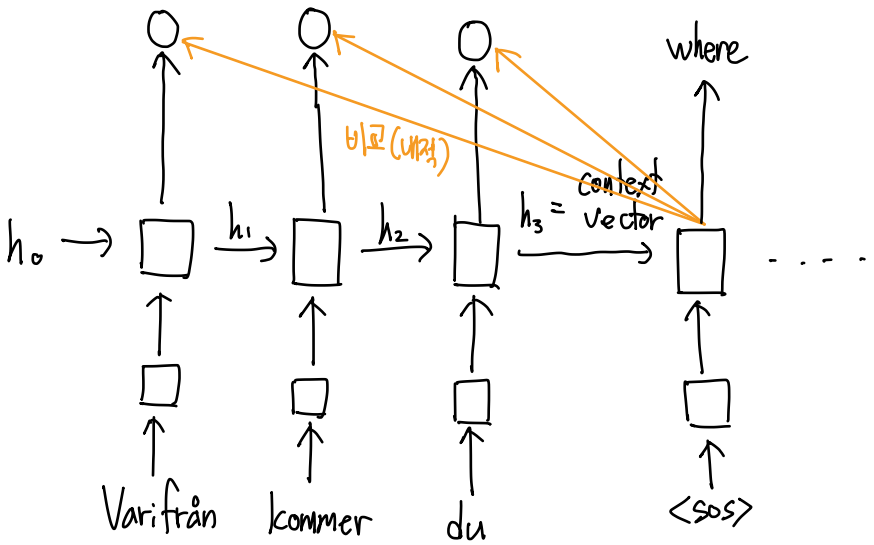
Attention

- Decoder에 처음 입력되는건 context vector라는 그냥 1개의 벡터.

  ⇒ 빈약해서 앞의 내용 잘 기억 못함.

- Decoder가 출력하기 전에 Encoder의 출력들인 hidden state들과 내적을 통해 비교

  ⇒ 뽑아내기전에 상관관계를 비교해서 뽑아낸다!



· $\{v_1, v_2, \cdots, v_n\}$이 $\mathbb{R}^n$의 orthonormal basis라면
↳ 수직, 크기1

↳ Query
$X = c_1v_1 + c_2v_2 + \cdots + c_nv_n$

$= (X \cdot v_1)v_1 + (X \cdot v_2)v_2 + \cdots + (X \cdot v_n)v_n$
(key, value)

변형 ①
$\underbrace{((W_Q X) \cdot (W_K v_1))}_{\alpha_1} \underbrace{W_V v_1}_{\tilde{v}_1} + \underbrace{((W_Q X) \cdot (W_K v_2))}_{\alpha_2} \underbrace{W_V v_2}_{\tilde{v}_2} + \cdots \underbrace{((W_Q X)(W_K v_n))}_{\alpha_n} \underbrace{W_V v_n}_{\tilde{v}_n}$

$\tilde{X} = \alpha_1\tilde{v}_1 + \alpha_2\tilde{v}_2 + \cdots + \alpha_n\tilde{v}_n$

변형 ②
$\frac{e^{\alpha_1}}{\sum_{i=1}^{n} e^{\alpha_i}}\tilde{v}_1 + \frac{e^{\alpha_2}}{\sum_{i=1}^{n} e^{\alpha_i}}\tilde{v}_2 + \cdots + \frac{e^{\alpha_n}}{\sum_{i=1}^{n} e^{\alpha_i}}\tilde{v}_n$

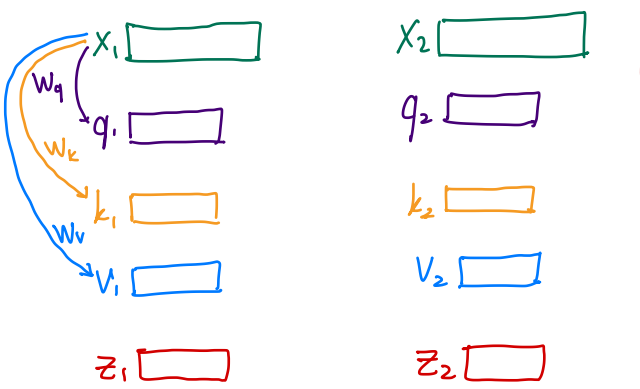
softmax ; skip-gram 처럼.

Self Attention

· 위의 attention 식에서 각각 v_1 or v_2 or $\cdots$ or v_n.

· $v_1, v_2, \cdots, v_n$이 { Query, key, value } 전부 다됨.

· Thinking Machine



$\Rightarrow z_1 = \text{softmax}(q_1 \cdot k_1)v_1 + \text{softmax}(q_1 \cdot k_2)$
(temperature 사용한 softmax)

Multi-headed Attention

- Q, K, W 조합 ⟶ 조금씩 다르게 만들어서 다양한 관점 반영하는것.
- Original transformer model은 head 8개.
- W^O으로 각 head의 가중치 반영.
- parameter : W^Q, W^K, W^V, W^O

Positional Encoding & Embedding

Subword) word-peice , BPE

unaffordable = un + afford + able

= [19, 28, 42] ⇐ 정수 index로 tokenizing.

⇒ 30000 차원으로 차원축소 : pretrained

↓ W (Embedding Matrix) : parameter.

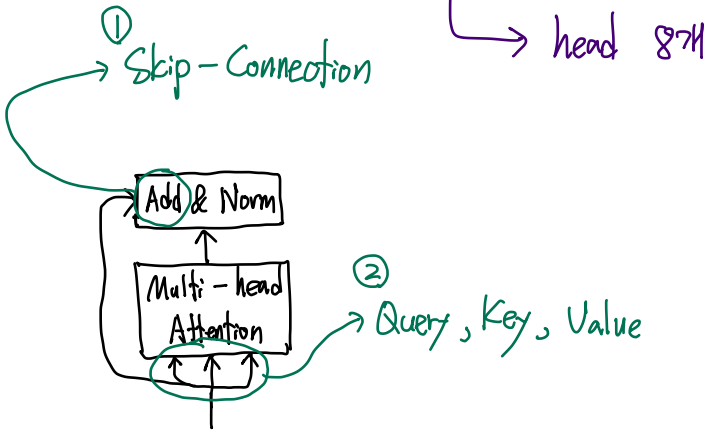
$W[\ ] \in \mathbb{R}^{512}$ 차원축소

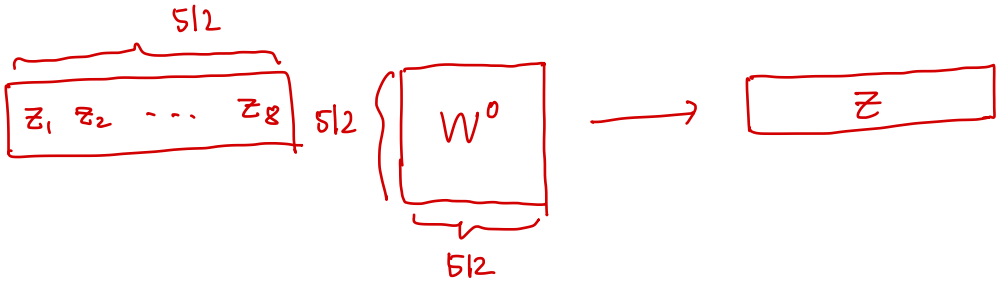
Position Encoding) sin / cos 함수 사용. (짝/홀)

같은 위치 ⇒ 주기 같게

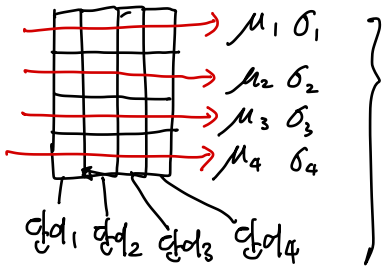
Multi-head Self Attention layer

→ head 8개





[Residual Connection]



Layer Normalization

↳ 어떤 뜻이 더/덜 돌보여야되는지 보여줌.

Pointwise FFN

- z_1, z_2 마다 목적에 맞게 FFN에 넣고 돌림.

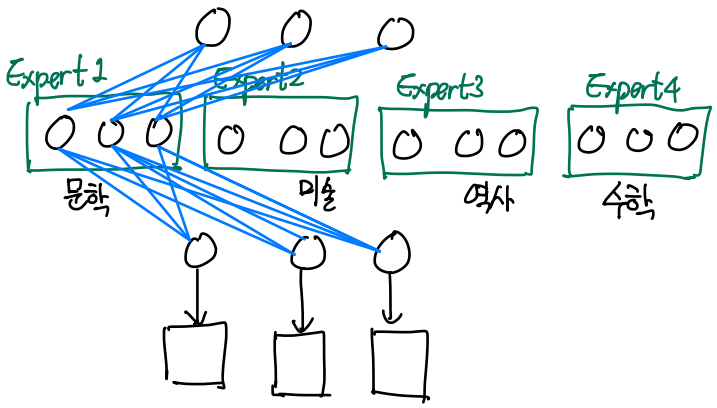
z_1 —x4→ □ → output

512 dim 2048 dim 512 dim

- 대부분의 parameter 잡아먹음.

Mixture of Expert (MOE)

- 각각의 일에 맞게 Expert 중 선택함.



Masked Self Attention

- Self-Attention (Encoder)에서는 전체 문장 알고있지만
  Decoder (번역)에서는 뒤에 어떤 단어 올지 미리 알수 없다.
  → Masked Self-Attention.
  자기자신까지 끊어서 계산한다.
  → n+1개 까지 알으면 그걸 다시 넣어서 반복.

* Key, Value는 Query에 영향 준다.