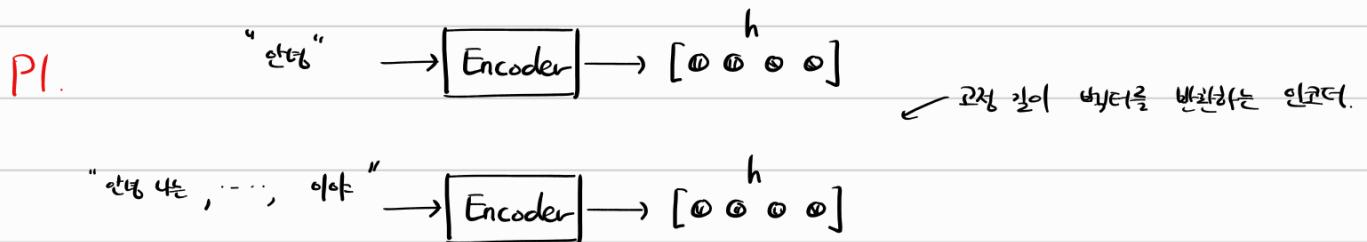
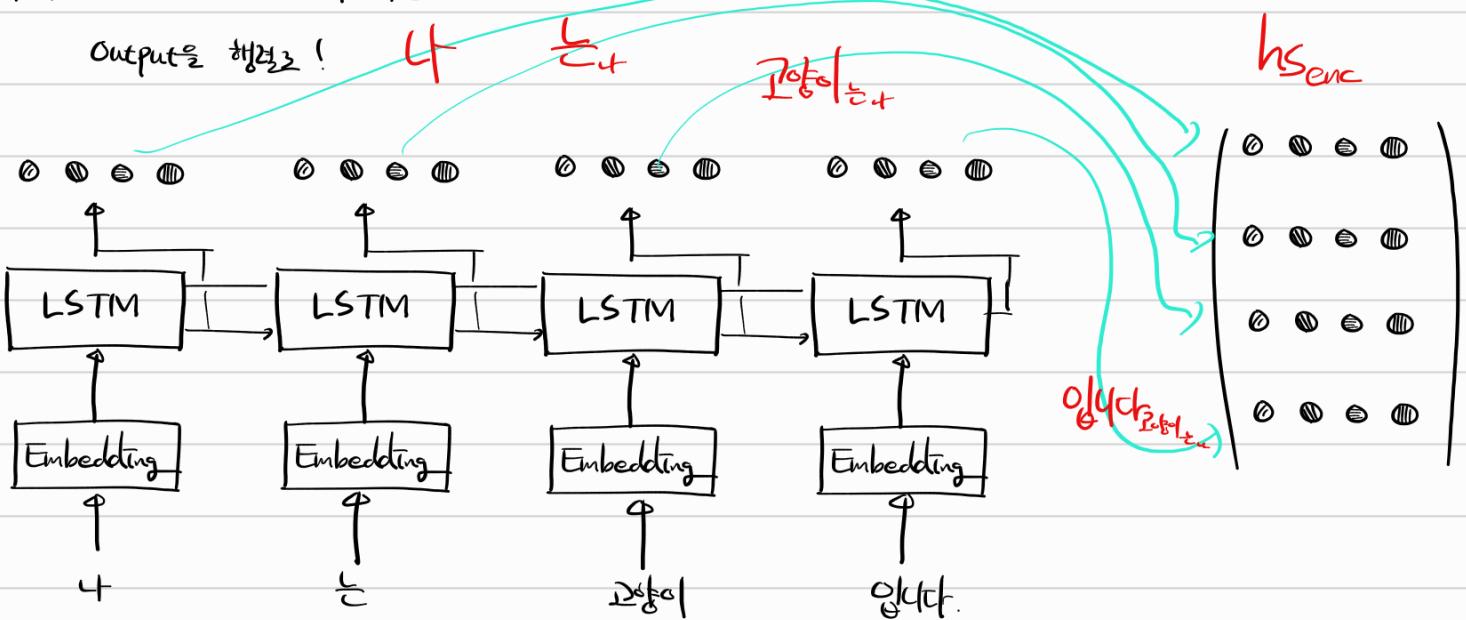


CH 1. Seq2Seq의 커다란 개선

Rmk 1.1 Encoder의 한계



Thm 1.2 Encoder의 개선



Thm 1.3 Decoder의 개선

① Alignment, 나는 고양이 입니다, 우리는 어떤 단어에 주목하여 그 단어의 번역을 수시로 사용!

choice는 맵이 안돼...

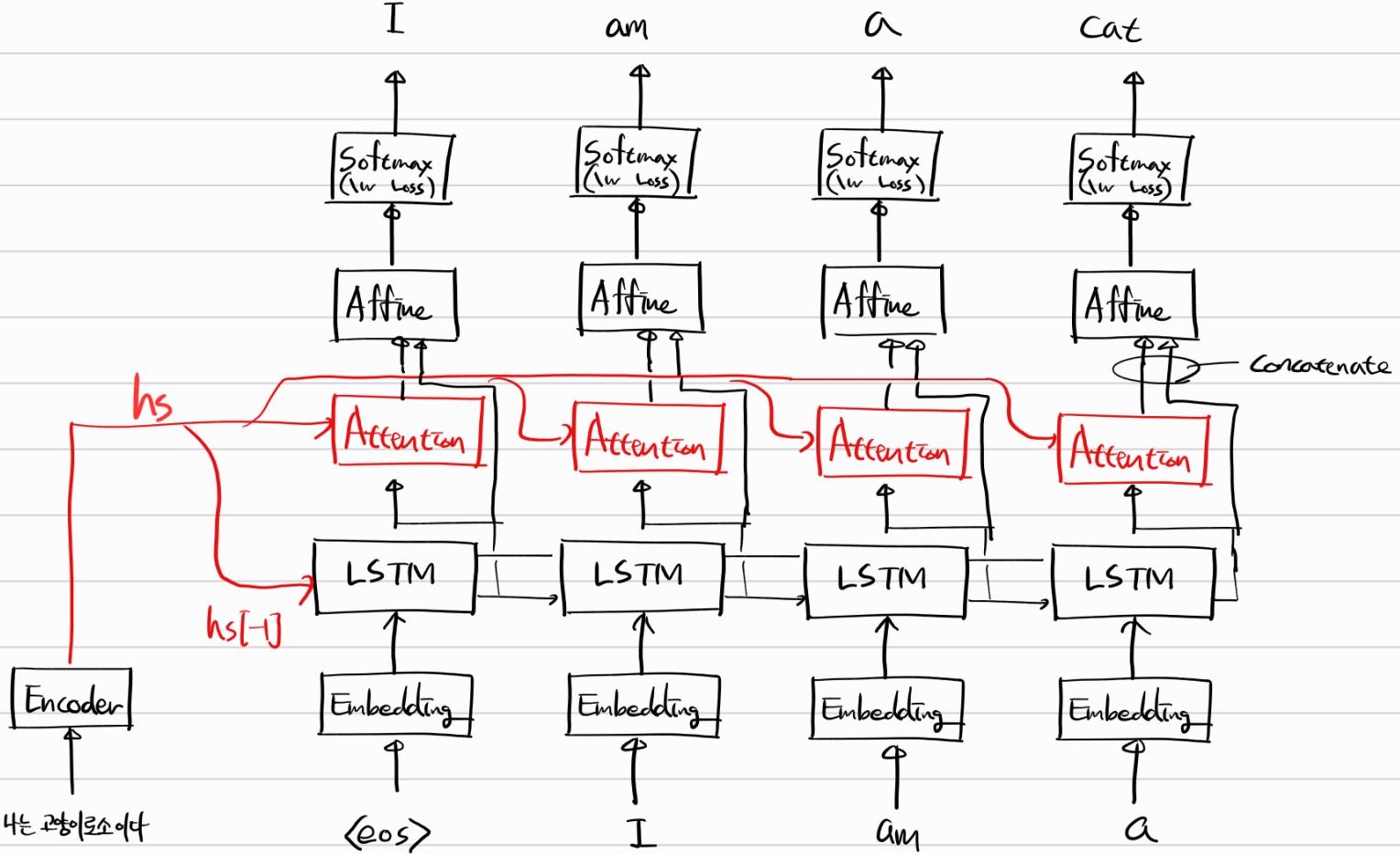
I am a cat

Goal: 출발어 ~ 도착어인 출발어 단어의 정보를 골라내는 것.

즉, 필요한 정보에만 주목하여 그 정보로부터 시계열 번역을 수행하는 것.

⇒ This is Attention.

② Attention



def 1.4 Context vector & weight sum

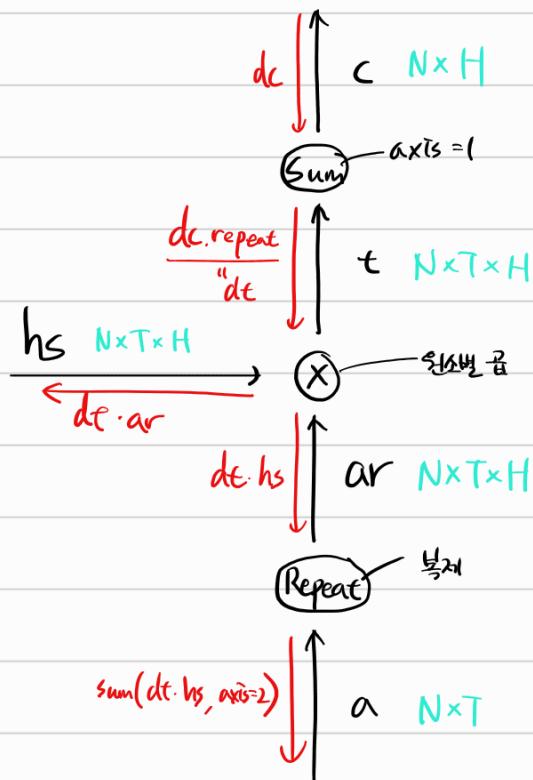
$$\begin{aligned}
 & \sum_{\text{나는 고양이이다}} \left(\begin{array}{c} \text{나} \\ \text{는} \\ \text{고양이} \\ \text{다} \\ \text{이다} \end{array} \right) \times \left(\begin{array}{c} 0.8 \\ 0.01 \\ 0.1 \\ \vdots \\ 0 \end{array} \right) = \left[\begin{array}{c} 0 \\ 0.01 \\ 0.1 \\ \vdots \\ 0 \end{array} \right] \\
 & \quad \text{Context Vector}
 \end{aligned}$$

The diagram shows the calculation of a context vector. It uses a matrix multiplication operation (\times) between a row vector hs (representing the hidden states of the words "나", "는", "고양이", "다", "이다") and a column vector a (representing the weights for each word). The result is a new vector where the second element is scaled by 0.01, while the others remain zero. This is labeled as the "Context Vector".

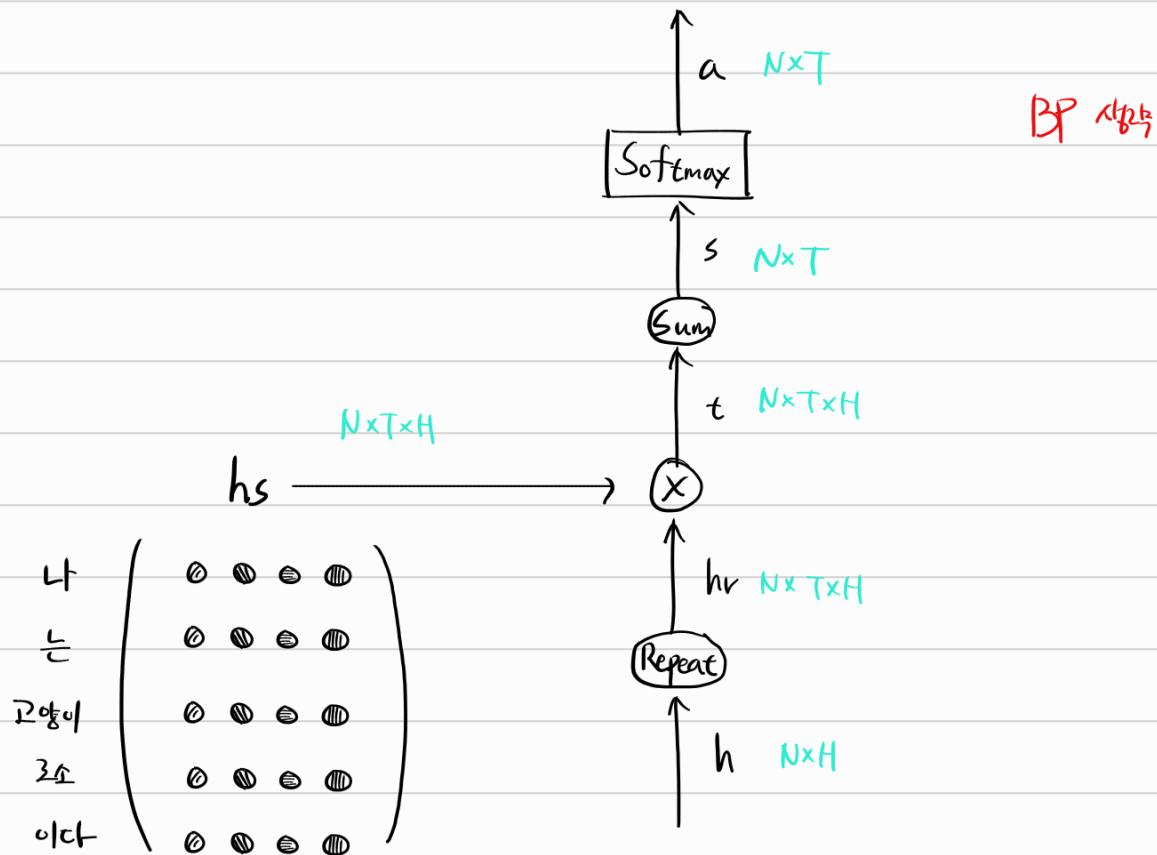
이거 구현을 $\text{matmul}(a, hs)$ 쓰는 것보다 $\text{repeat}()$ 과 $\text{sum}()$ 을 쓰는 편임.

(\because 미니배치 처리를 위해, 물론 np.tensordot() 과 $\text{np.einsum}()$ 도 가능)

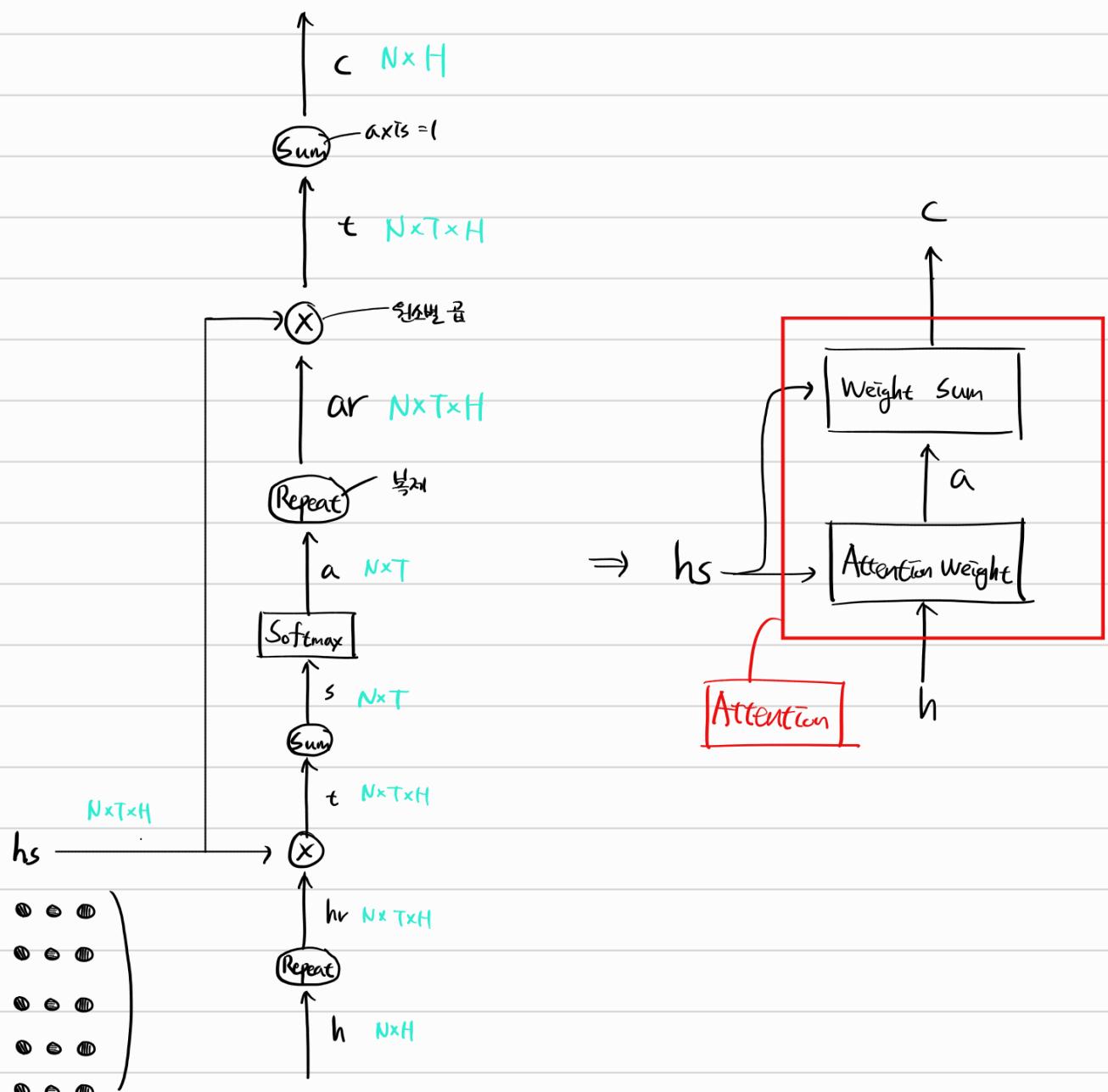
Note 1.5 weight sum 계산 그림



Note 1.6 Attention 계산 그림

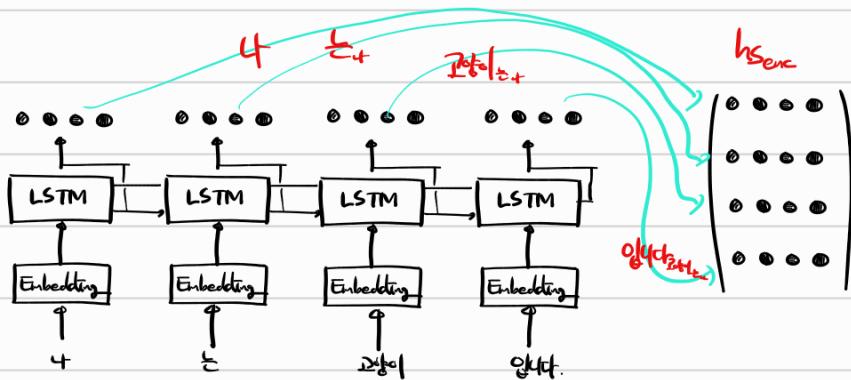


Rmk 1.7 Context Vector 계산 과정

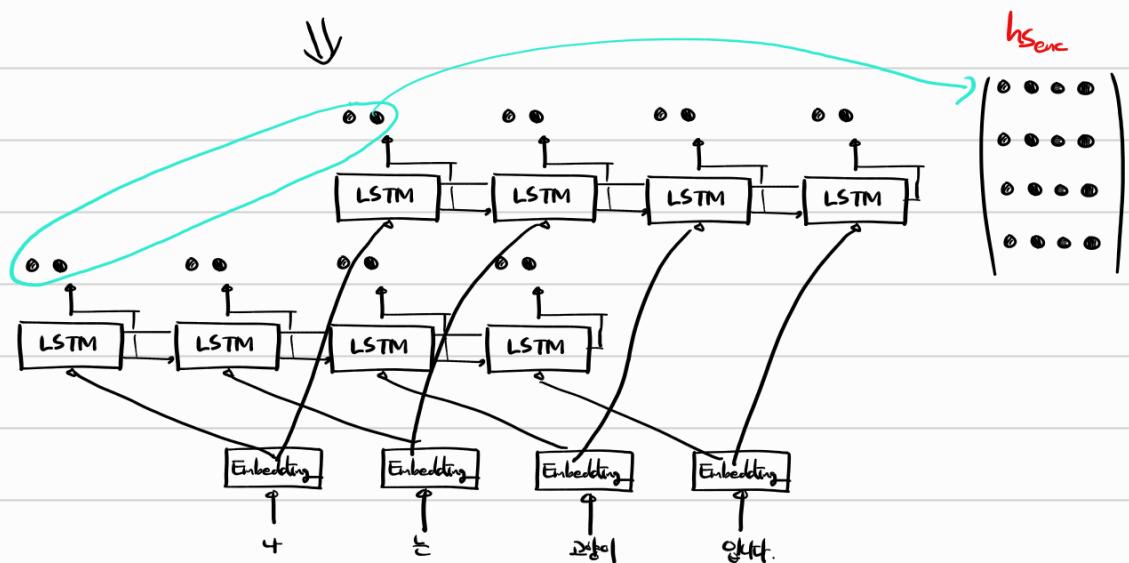


CH.2 어텐션에 대한 추가 이야기.

Def 2.1 양방향 LSTM



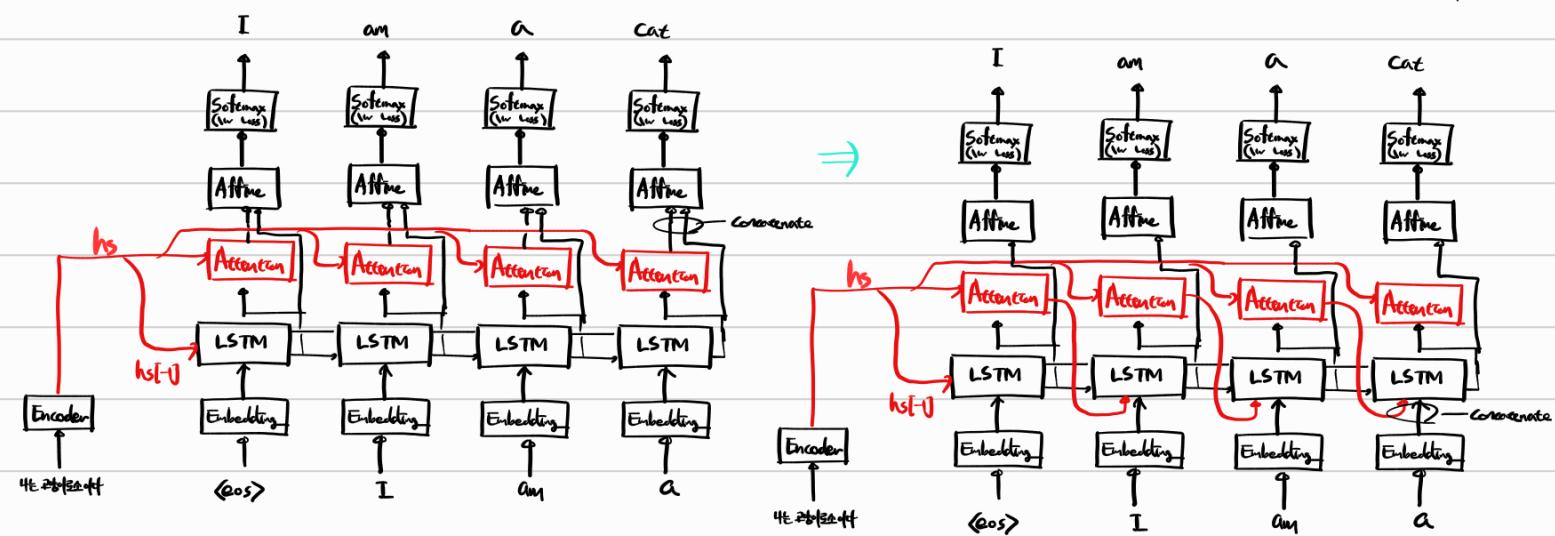
그늘 왼쪽에서 오른쪽으로 읽는 것을 가정



Thm 2.2 Attention의 다양한 활용

opt ①

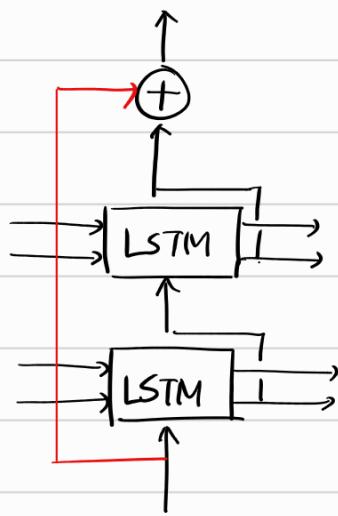
LSTM의 context vector를 사용.



의 유형도 상당히 많은 방법이 있다.

Thm 2.3 skip connection (Residual connection, short-cut)

☞ 보통 Encoder와 Decoder의 LSTM Layer 갈이 통일하는 편.

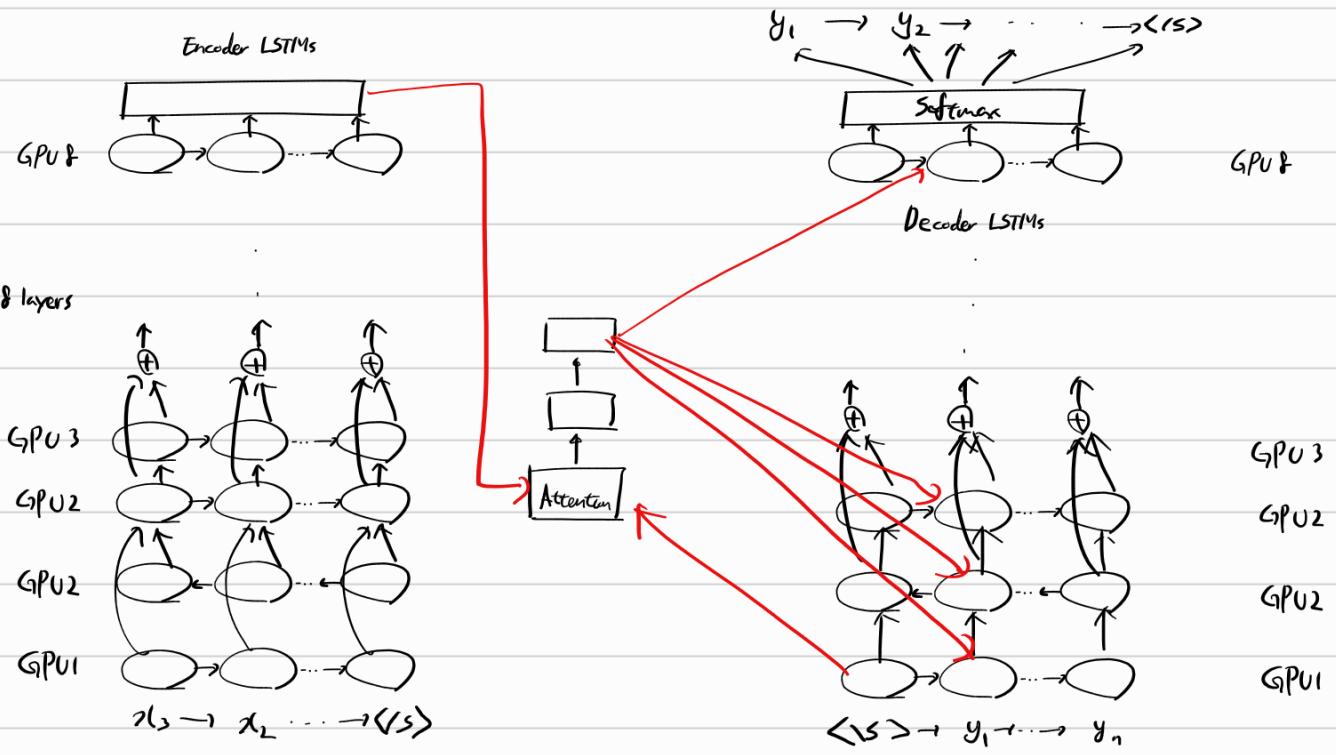


LSTM은 RNN의 시간 방향에서 가로기 소설을 냉기.

각각의 skip connect는 LSTM의 갈이 방향에서 가로기 소설을 냉기

CH 3. Attention's application

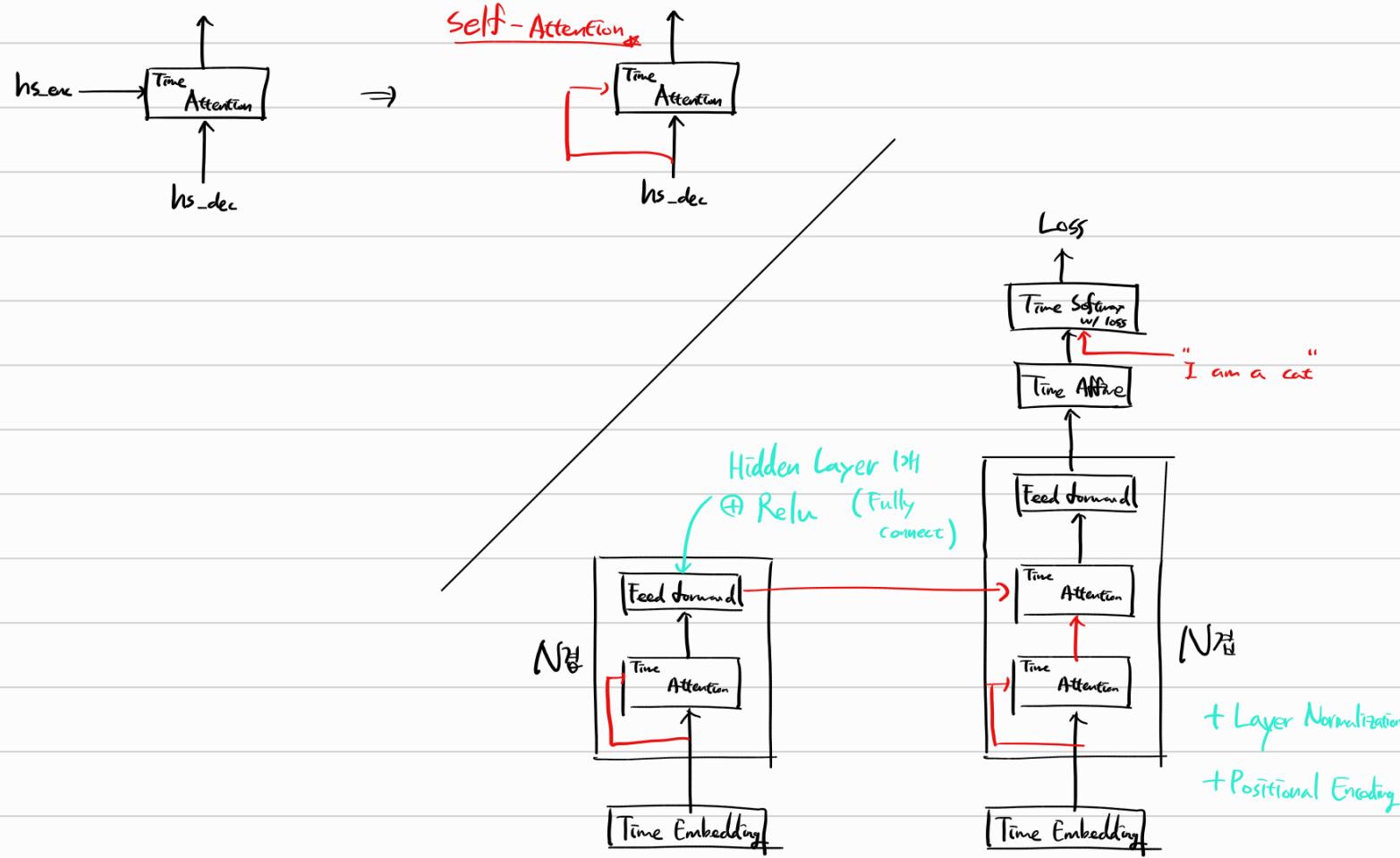
① GNMT



* GNMT < ConvS2S < Transformer

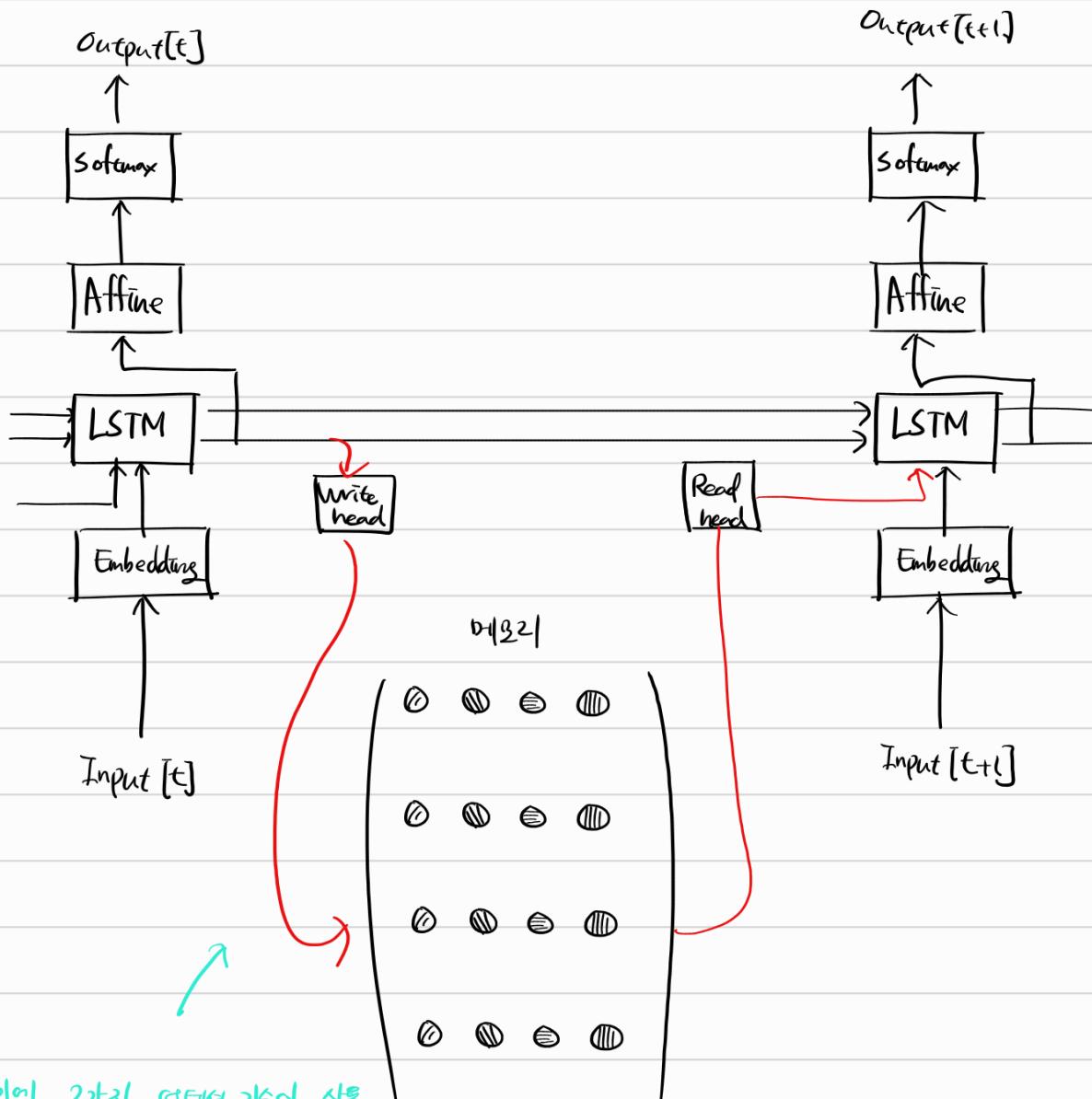
RNN only CNN or 2 encoder.

② Transformer



③ NTM

Memory Cell 이 아닌 전자 회로 메모리를 쓰자!



여기서 2가지 어텐션 기술이 사용

- ① 콘텐츠 기반 (기존 attention)
- ② 위치 기반 : 메모리 위치 흐름