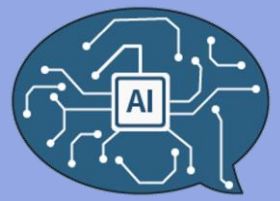
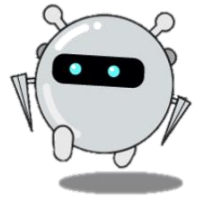


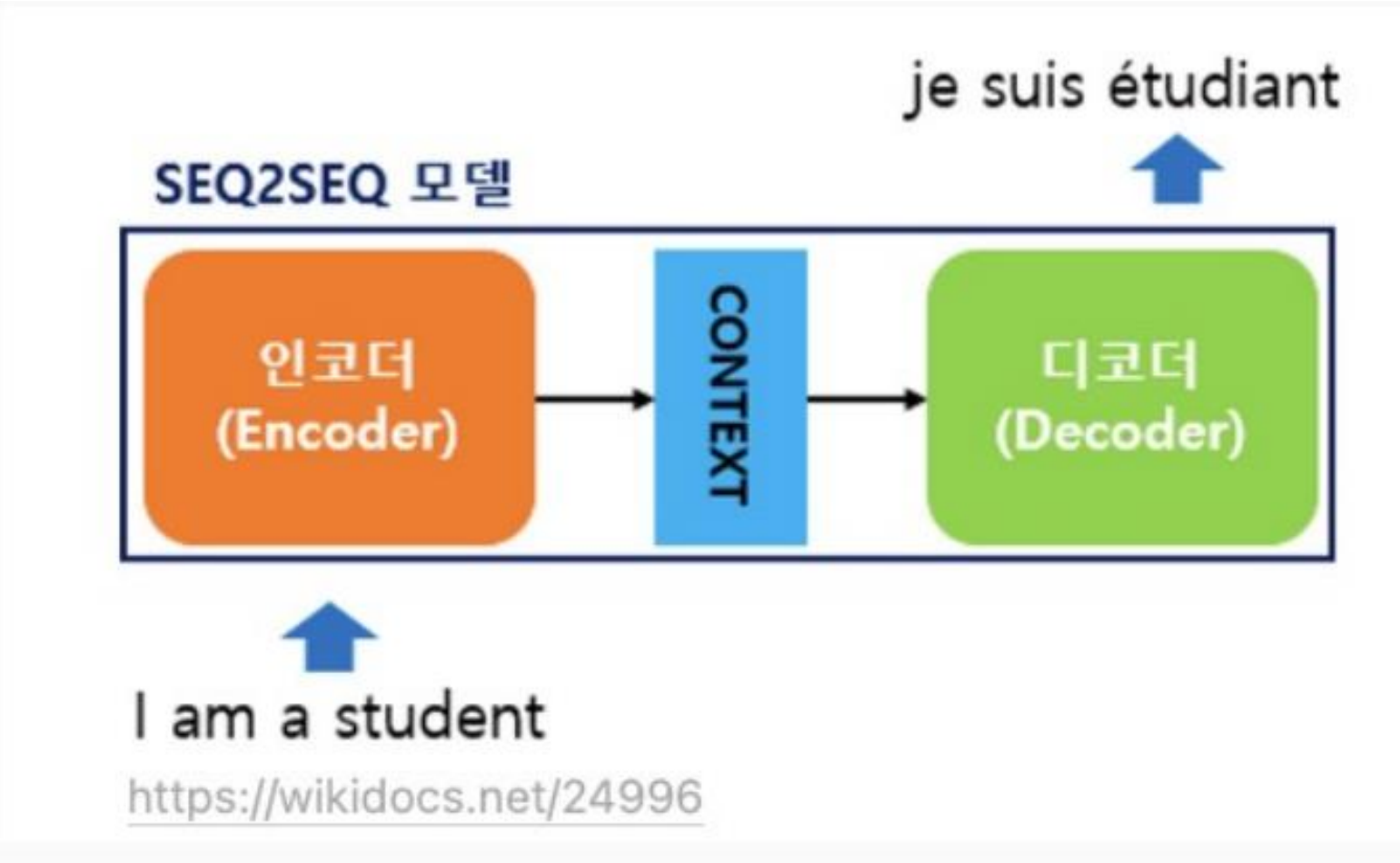
Attention



진행자 : 멘토 현시은

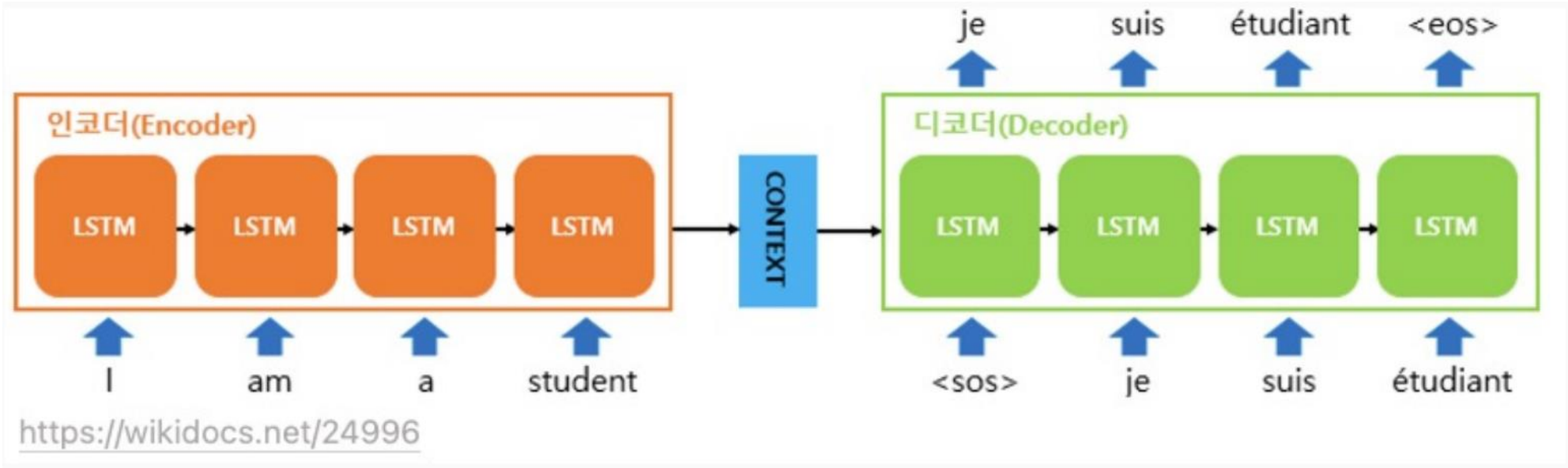
seq2seq 복습

seq2seq란?



- seq2seq는 크게 인코더와 디코더라는 두 개의 모듈로 구성

seq2seq란?

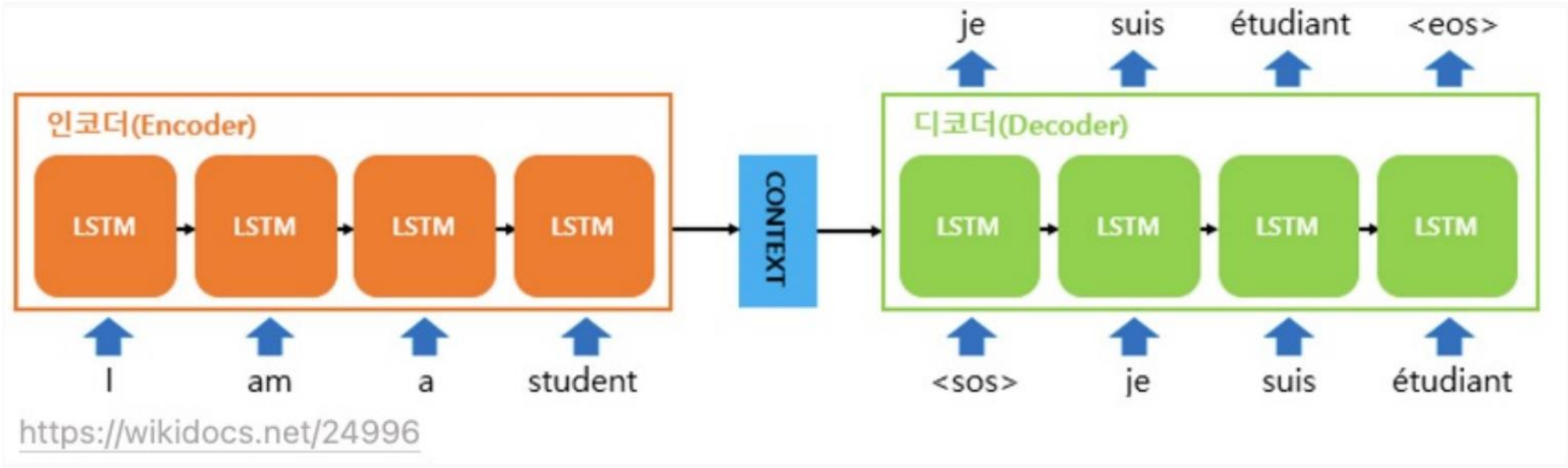


- 인코더는 입력 문장의 모든 단어들을 순차적으로 입력받는다.

- 그 후 마지막 단계에 이 모든 단어 정보들을 압축해서 하나의 벡터로 만든다.

- 이 마지막 벡터를 컨텍스트 벡터(context vector)라고 한다.

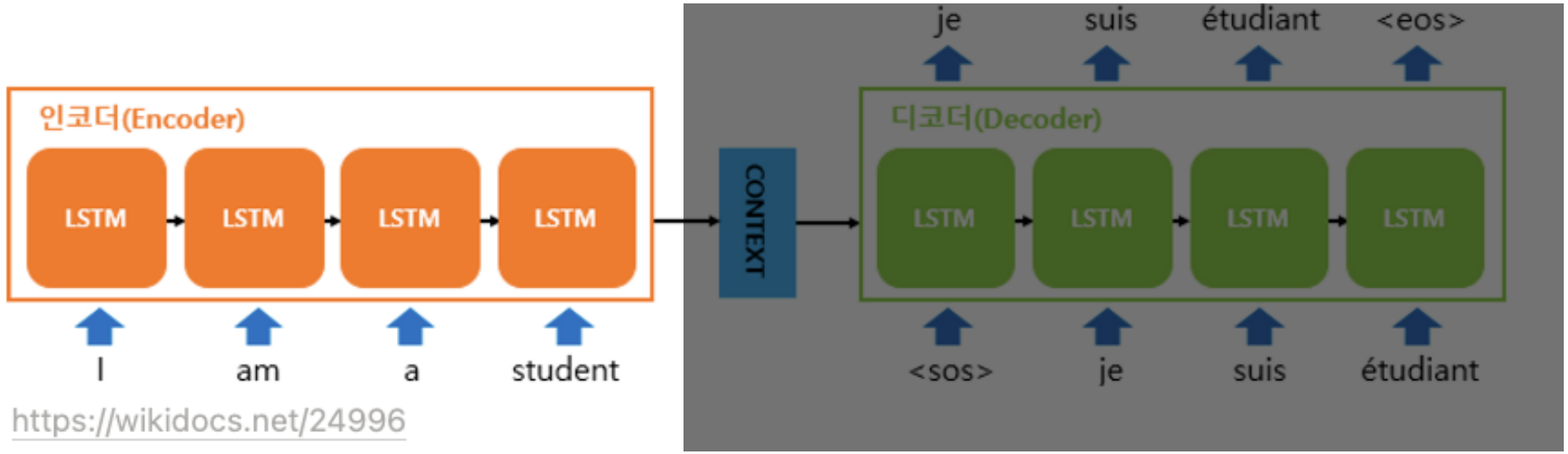
seq2seq란?



- 입력 문장의 정보가 컨텍스트 벡터로 모두 압축되면, 이를 디코더로 전송

- 디코더는 컨텍스트 벡터를 받아서 번역된 단어를 한 개씩 순차적으로 출력.

seq2seq : encoder

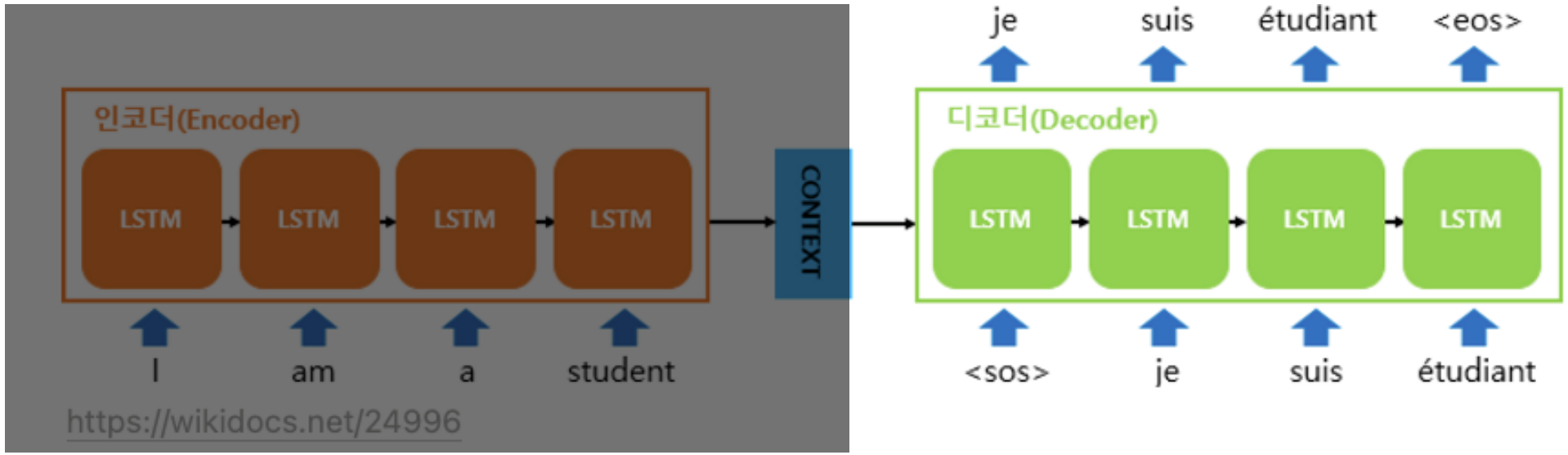


- 입력 문장은 단어 토큰화를 통해 단어 단위로 쪼개진다.

- 단어 토큰 각각은 LSTM 셀의 각 시점의 입력이 된다.

- 모든 단어를 입력받고, 인코더 셀의 마지막 시점의 은닉 상태가 컨텍스트 벡터이다.

seq2seq : decoder

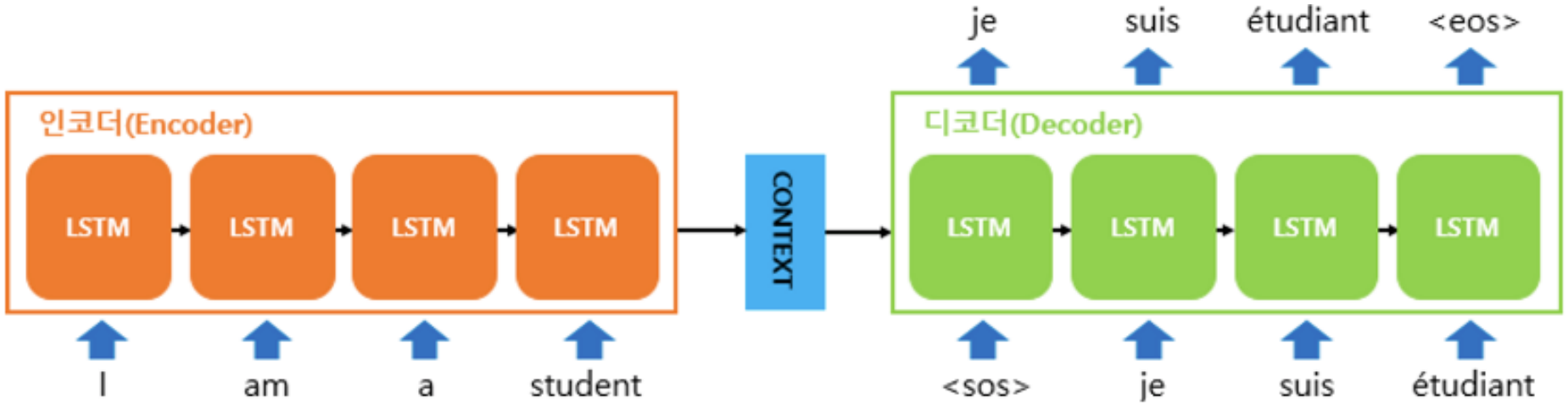


- 컨텍스트 벡터는 디코더 LSTM 셀의 첫번째 은닉 상태에 사용

- 디코더는 <sos>가 입력되면, 다음에 등장할 확률이 높은 단어를 예측

- 디코더는 다음에 올 단어를 예측하고 그 단어를 다음 시점의 셀의 입력으로 넣는 행위를 반복(<eos>가 다음 단어로 예측될 때까지)

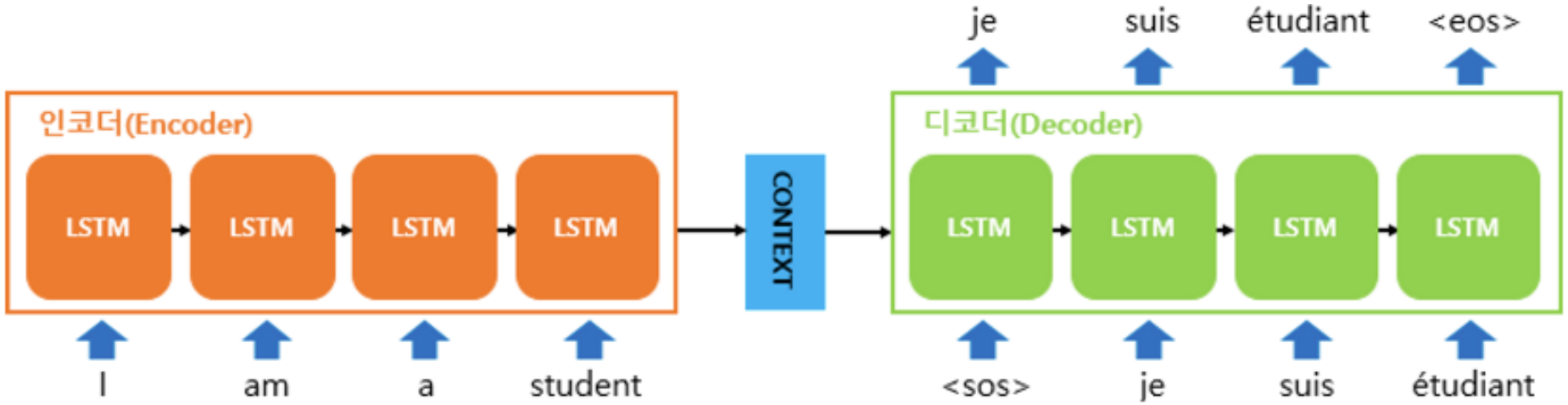
Teacher Force



https://wikidocs.net/24996

- 훈련 과정에서 디코더에게 인코더가 보낸 컨텍스트 벡터와 함께 실제 정답까지 입력시킨다.
- 즉, 디코더에게 정답을 알려주면서 훈련시킨다.

seq2seq의 한계



https://wikidocs.net/24996

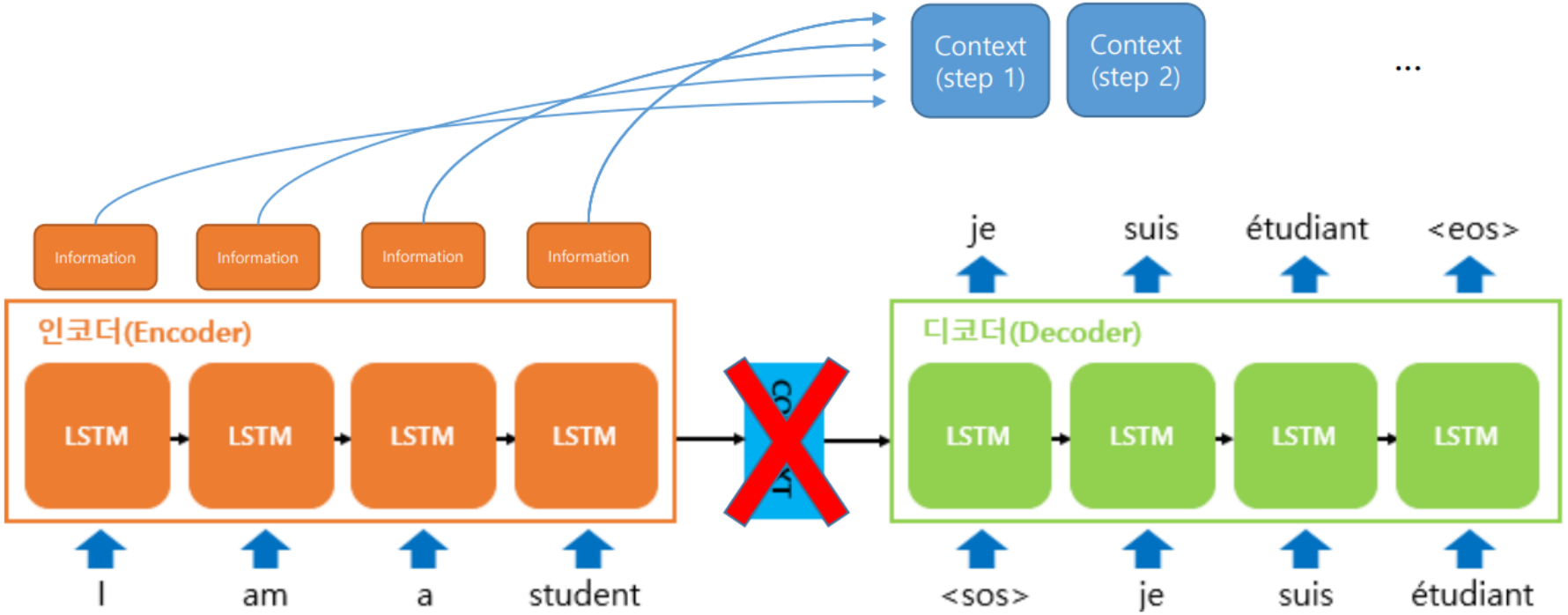
- seq2seq의 특성 상, hidden state에서 앞쪽 input token의 정보는 희미해진다.

- 즉, input sequence의 길이가 길 경우, 컨텍스트 벡터에서 input의 정보를 정확하게 압축하기 어렵다.

- 이를 해결한 것이 Attention

Attention

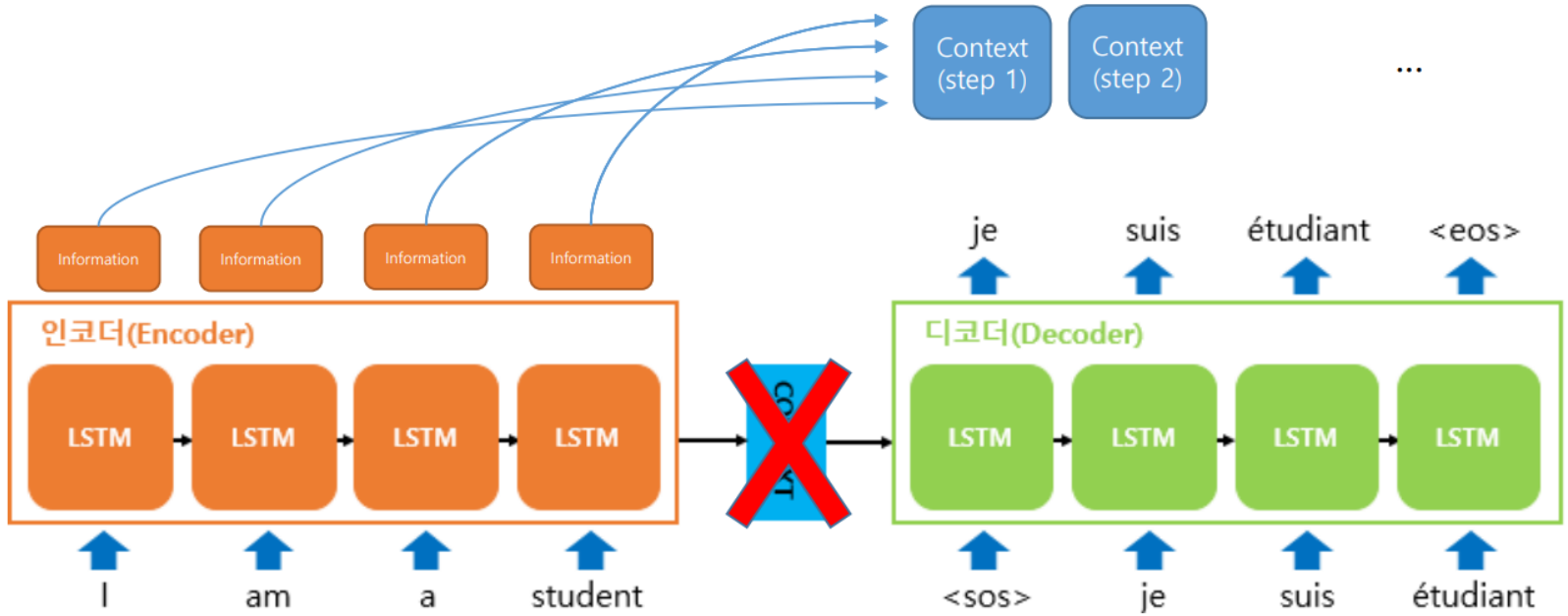
Attention



- 올바른 출력을 위해서는 어떤 input 토큰을 더 많이 고려해야 할까?

- Decoding step에 따라 “attention weight”가 달라진다.

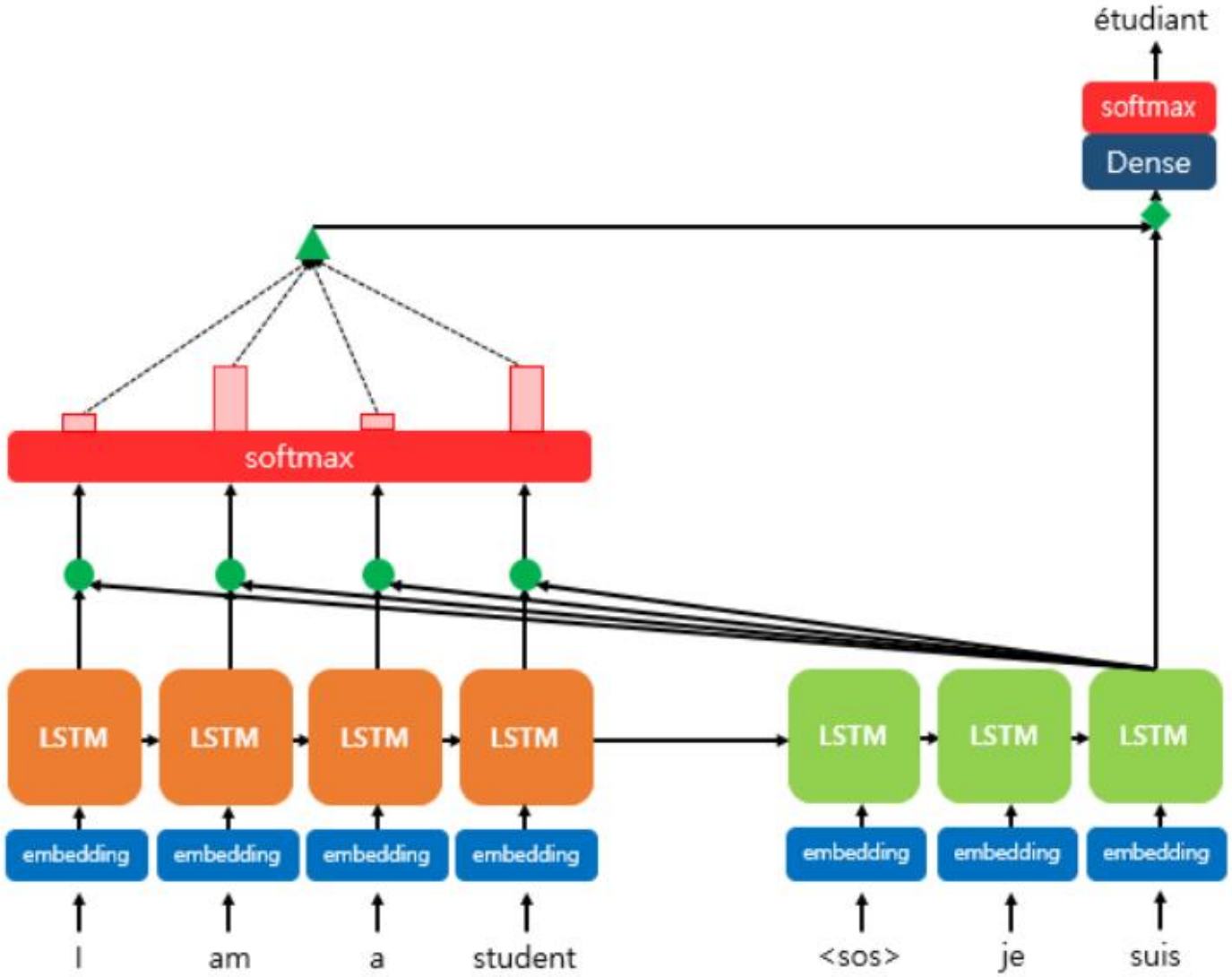
Attention



- 각 Decoding step에서 각기 다른 context vector를 활용한다.

-이 때의 context vector는 encoder가 각 input 토큰을 압축한 정보의 가중합이다

Attention의 전체적인 과정



Attention의 전체적인 과정

1) Attention Score를 구한다.

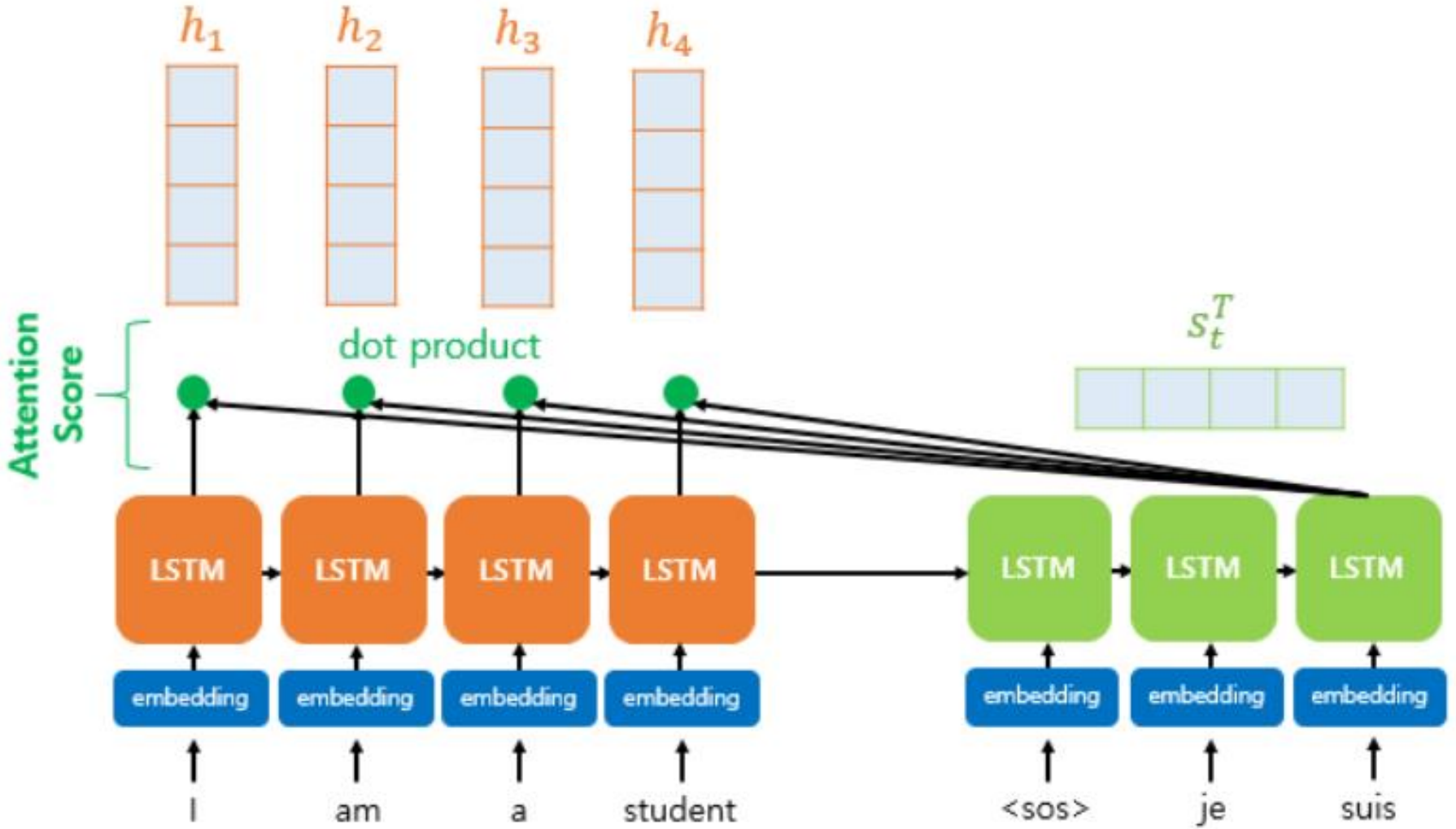
2) Attention Score 를 Softmax 함수에 통과시켜서 Attention Weight를 구한다

3) 각 인코더의 Attention Weight와 hidden state를 가중합하여 Context Vector를 구한다.

4) Context Vector와 디코더의 t 시점의 Output 값(y_t)을 연결한다.(Concatenate)

Attention의 전체적인 과정

1) Attention Score를 구한다.



Attention의 전체적인 과정

1) Attention Score를 구한다.

attention score = $\vec{e}_t = H^e W_\alpha \vec{h}_t^d$

(1) H^e: 인코더의 모든 hidden descriptor 모음 D-차원의 모든 T개의 인코더 hidden descriptors 행렬 (T,D)

$$H^e = \begin{bmatrix} \leftarrow & \vec{h}_1^e & \rightarrow \\ \leftarrow & \vec{h}_2^e & \rightarrow \\ \vdots & \vdots & \vdots \\ \leftarrow & \vec{h}_T^e & \rightarrow \end{bmatrix} \begin{matrix} \uparrow \\ \\ \\ \downarrow \end{matrix} \; T \quad (\leftarrow D \rightarrow)$$

(2) W_a: 학습 가능한 매개변수인 가중치에 대한 행렬 (D,D)

$$W_\alpha = \begin{bmatrix} \uparrow & \uparrow & \cdots & \uparrow \\ \vec{W}_1 & \vec{W}_2 & \cdots & \vec{W}_D \\ \downarrow & \downarrow & \cdots & \downarrow \end{bmatrix} \begin{matrix} \uparrow \\ D \\ \downarrow \end{matrix} \quad (\leftarrow\!\!\leftarrow D \rightarrow\!\!\rightarrow)$$

(3) h_t^d: t step에서의 디코더 hidden descriptor (D-차원)

Attention의 전체적인 과정

1) Attention Score를 구한다.

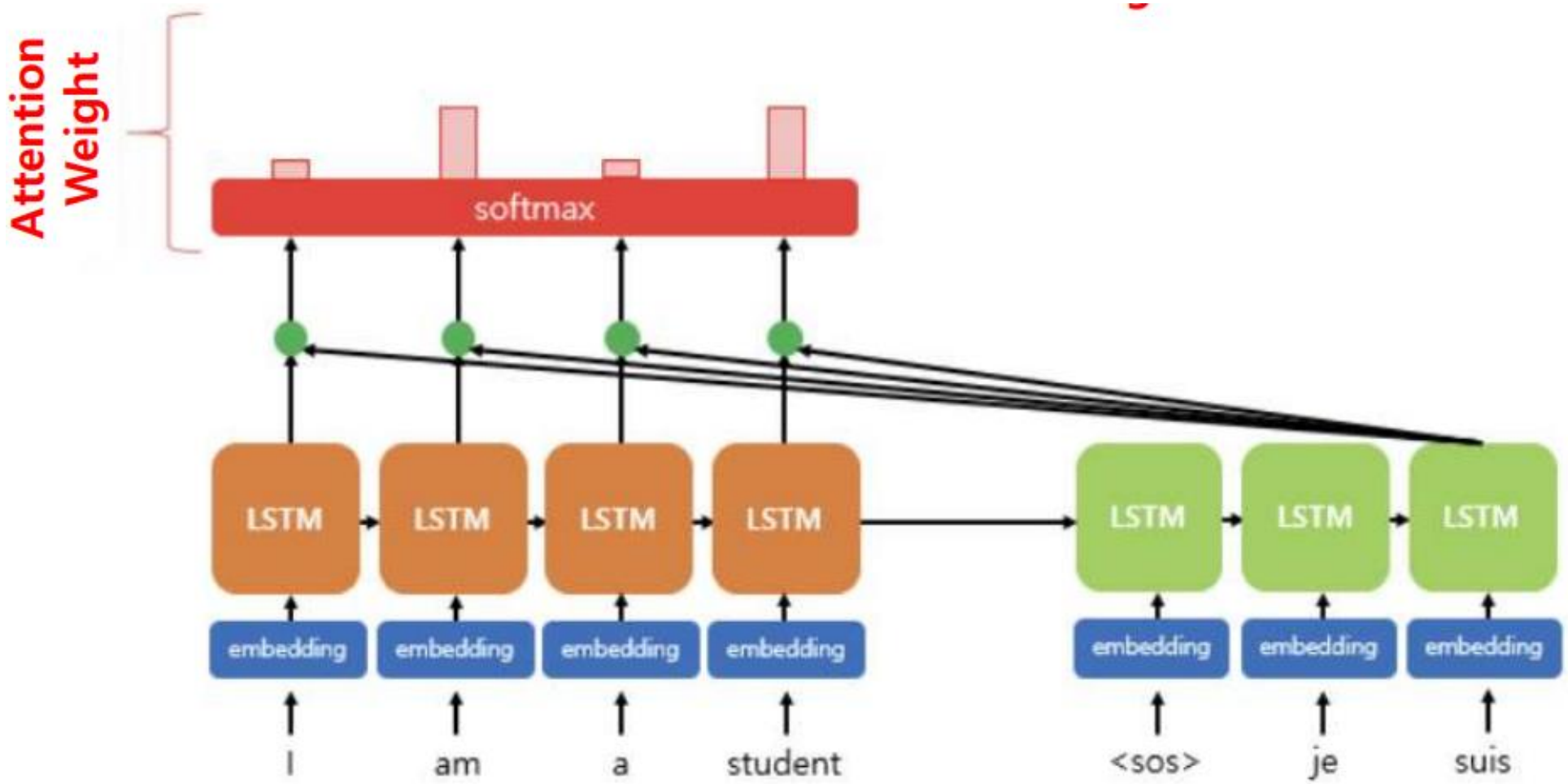
$$H^e W_\alpha = \begin{bmatrix} \vec{h}_1^e \cdot \vec{W}_1 & \vec{h}_1^e \cdot \vec{W}_2 & \cdots & \vec{h}_1^e \cdot \vec{W}_D \\ \vec{h}_2^e \cdot \vec{W}_1 & \vec{h}_2^e \cdot \vec{W}_2 & \cdots & \vec{h}_2^e \cdot \vec{W}_D \\ \vdots & \vdots & \ddots & \vdots \\ \vec{h}_T^e \cdot \vec{W}_1 & \vec{h}_T^e \cdot \vec{W}_2 & \cdots & \vec{h}_T^e \cdot \vec{W}_D \end{bmatrix} \begin{matrix} \uparrow \\ T \\ \downarrow \end{matrix} \quad (D \rightarrow)$$

$$H^e W_\alpha \mathbf{h}_t^d = \begin{bmatrix} h_1^e \cdot W_1 & h_1^e \cdot W_2 & \cdots & h_1^e \cdot D \\ h_2^e \cdot W_1 & h_2^e \cdot W_2 & \cdots & h_2^e \cdot W_D \\ \vdots & \vdots & \ddots & \vdots \\ h_T^e \cdot W_1 & h_T^e \cdot W_2 & \cdots & h_T^e \cdot W_D \end{bmatrix} \begin{bmatrix} (h_t^d)_1 \\ (h_t^d)_2 \\ \vdots \\ (h_t^d)_D \end{bmatrix}$$

$$= \begin{bmatrix} e_{t,1} \\ e_{t,2} \\ \vdots \\ e_{t,T} \end{bmatrix}$$

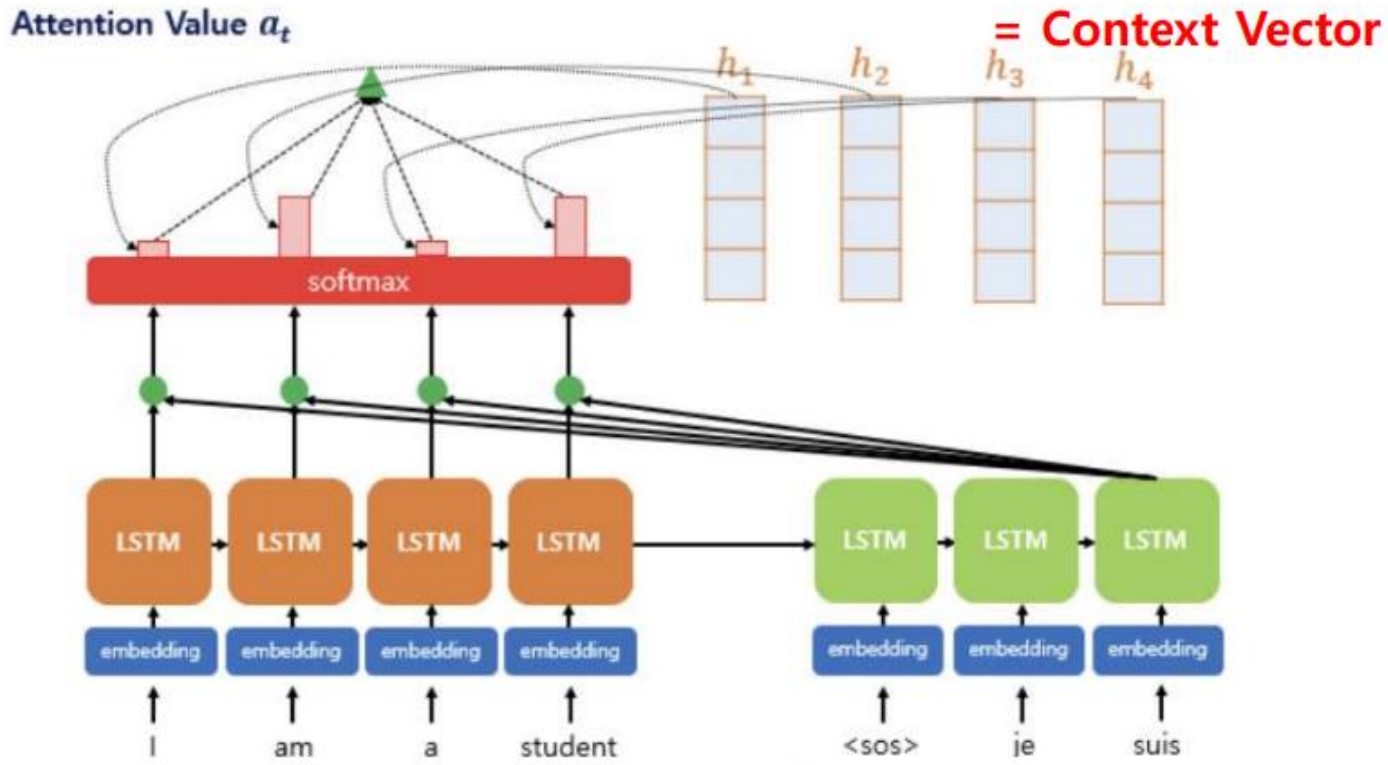
Attention의 전체적인 과정

2) Attention Score 를 Softmax 함수에 통과시켜서 Attention Weight를 구한다



Attention의 전체적인 과정

3) 각 인코더의 Attention Weight와 hidden state를 가중합하여 Context Vector를 구한다.

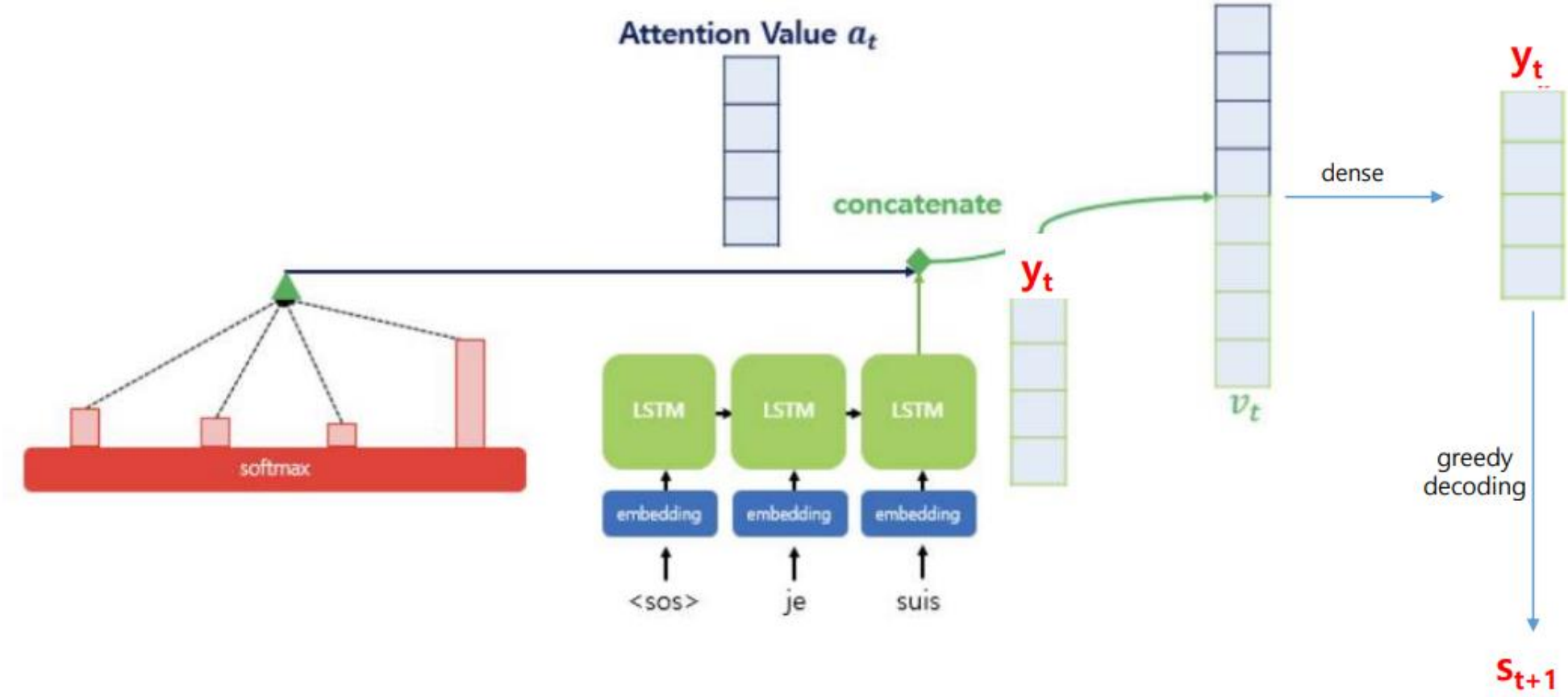


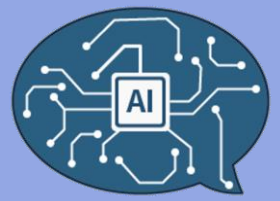
$$c_t = \sum_{j=1}^{T} \alpha_j h_j^e = \alpha_1(h_1^e) + \alpha_2(h_2^e) + \cdots + \alpha_T(h_T^e)$$

Context Vector는 Attention Value(c_t)이라고도 불린다.

Attention의 전체적인 과정

4) Context Vector와 디코더의 t 시점의 Output 값(y_t)을 연결한다.(Concatenate)





감사합니다

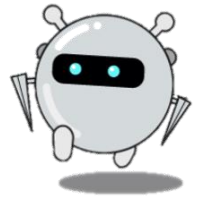


그림 출처 : https://wikidocs.net/book/2155