NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE

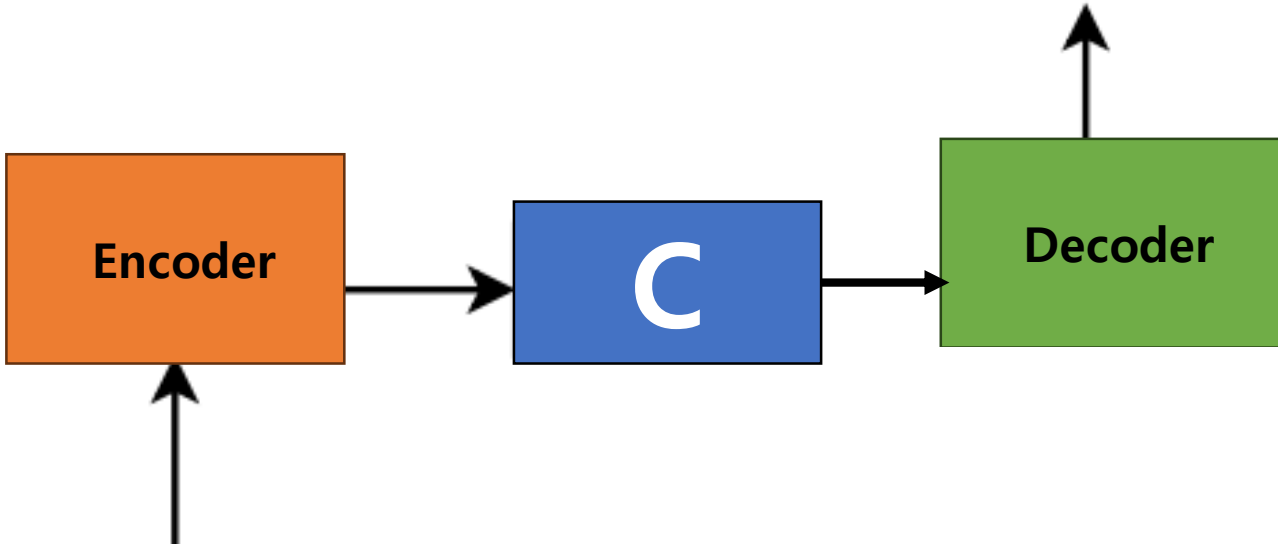
Dzmitry Bahdanau Jacobs University Bremen, Germany
(2015)

2024.03.03
유하영

Sequence to Sequence Learning with Neural Networks

I am happy.

(output sentence)

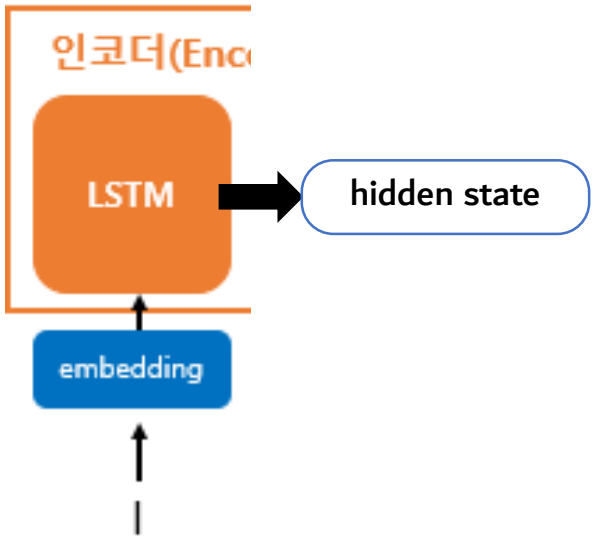


(input sentence)

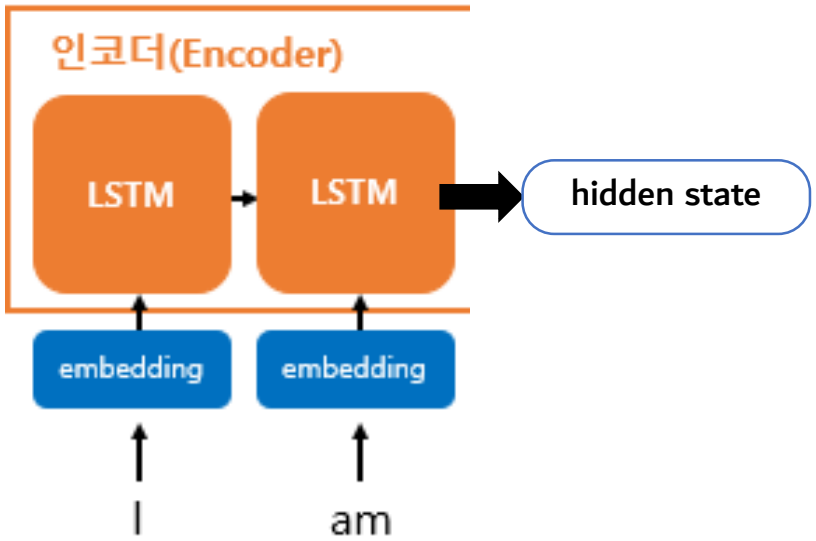
나는 행복하다.

seq2seq

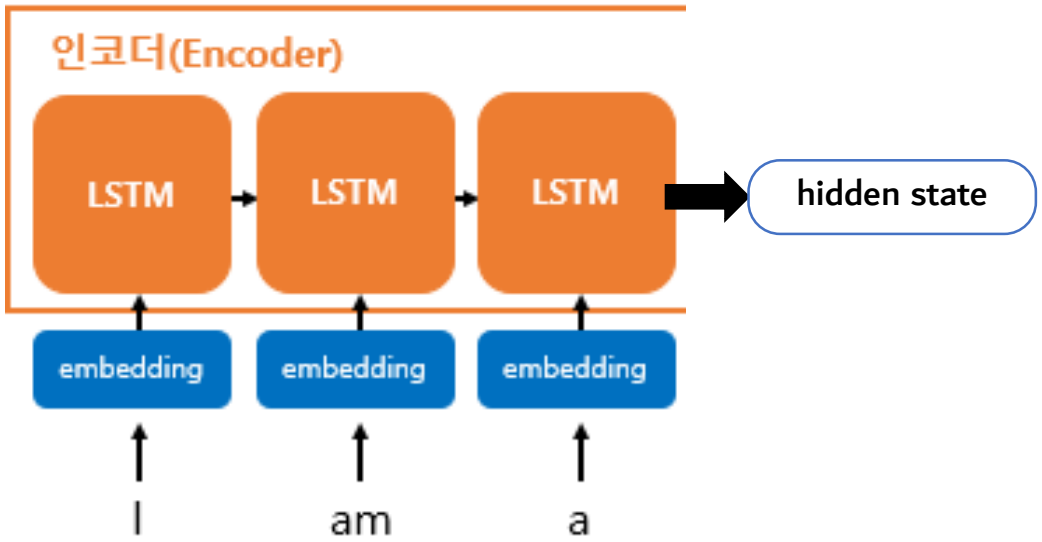
I am a student. → je suis étudiant.



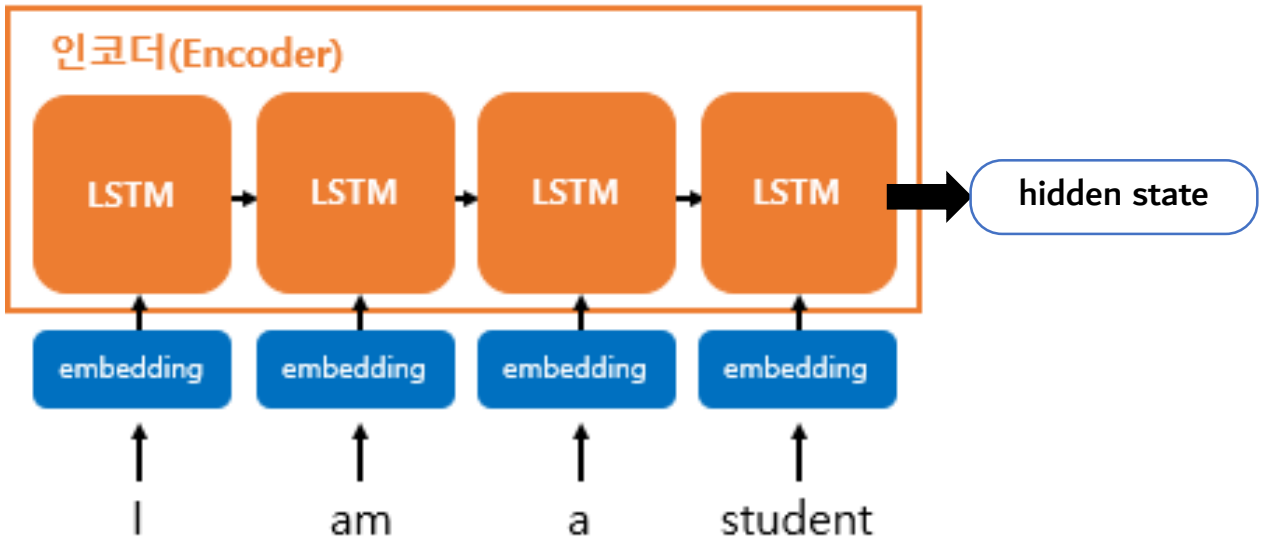
I am a student. → je suis étudiant.



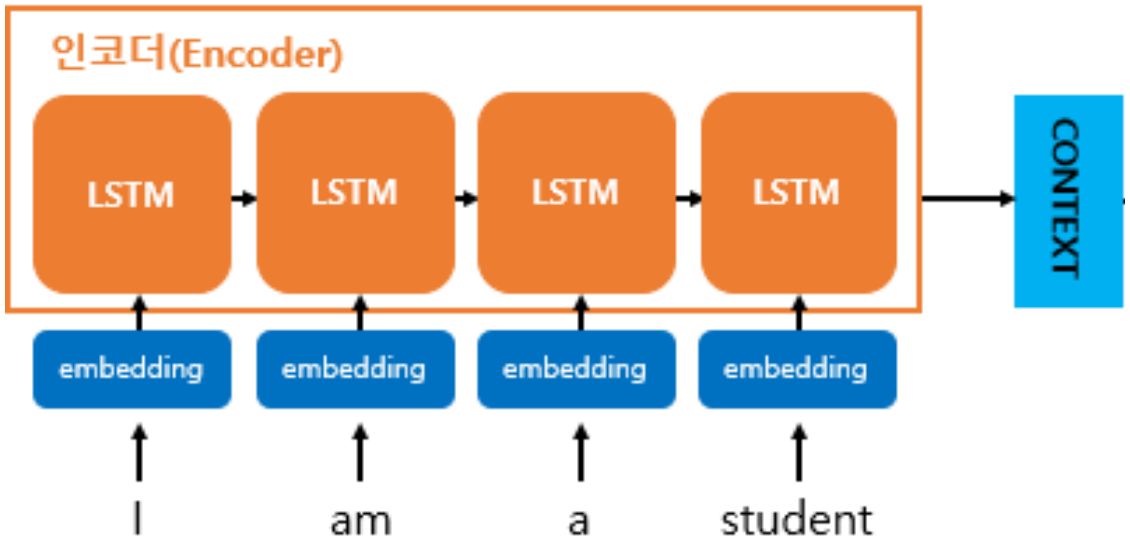
I am a student. → je suis étudiant.



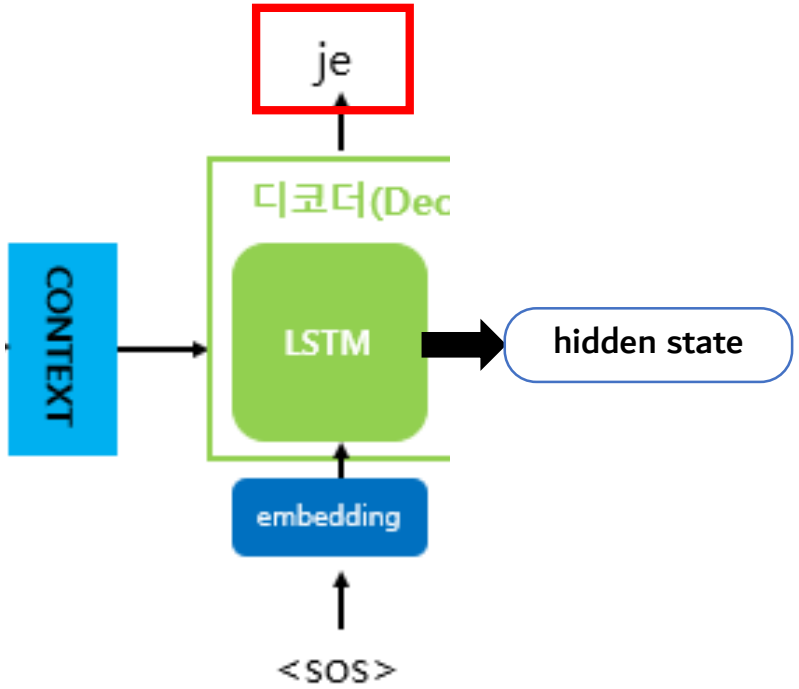
I am a student. → je suis étudiant.



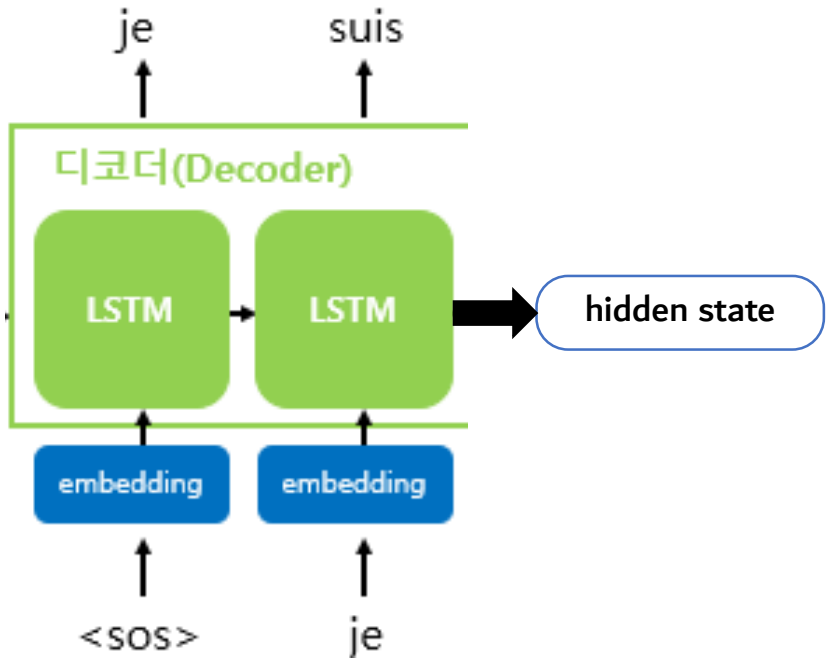
I am a student. → je suis étudiant.



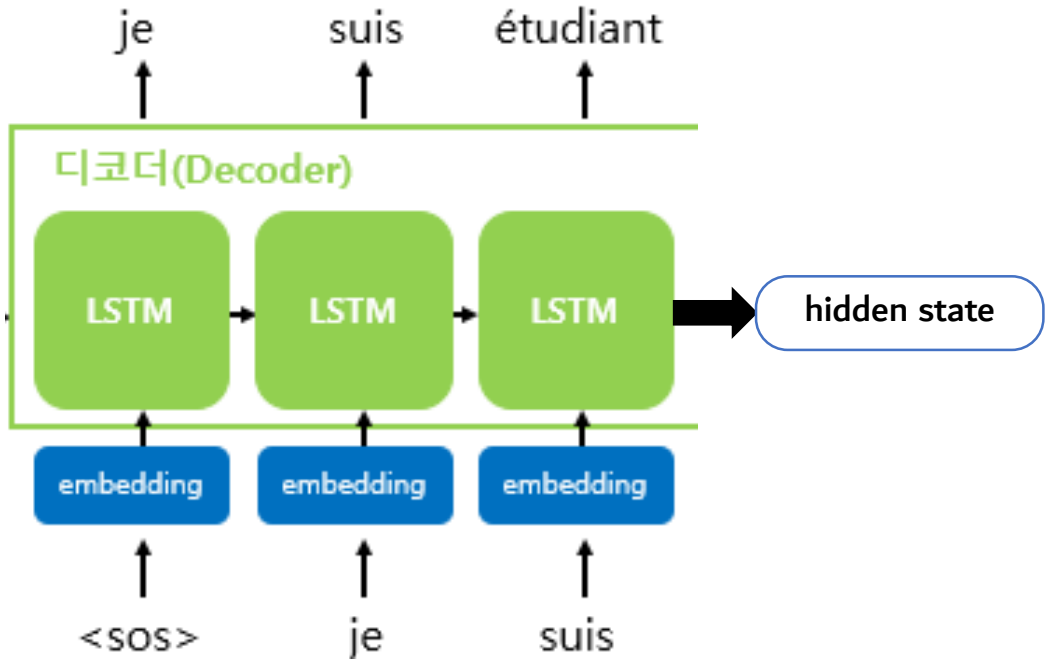
I am a student. → <sos> je suis étudiant.



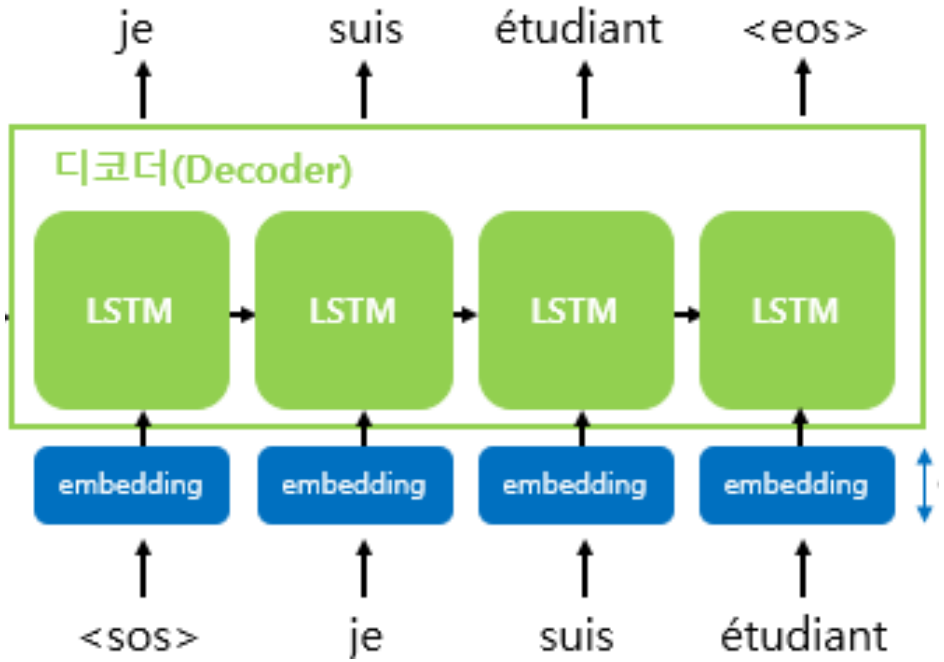
I am a student. → je suis étudiant.



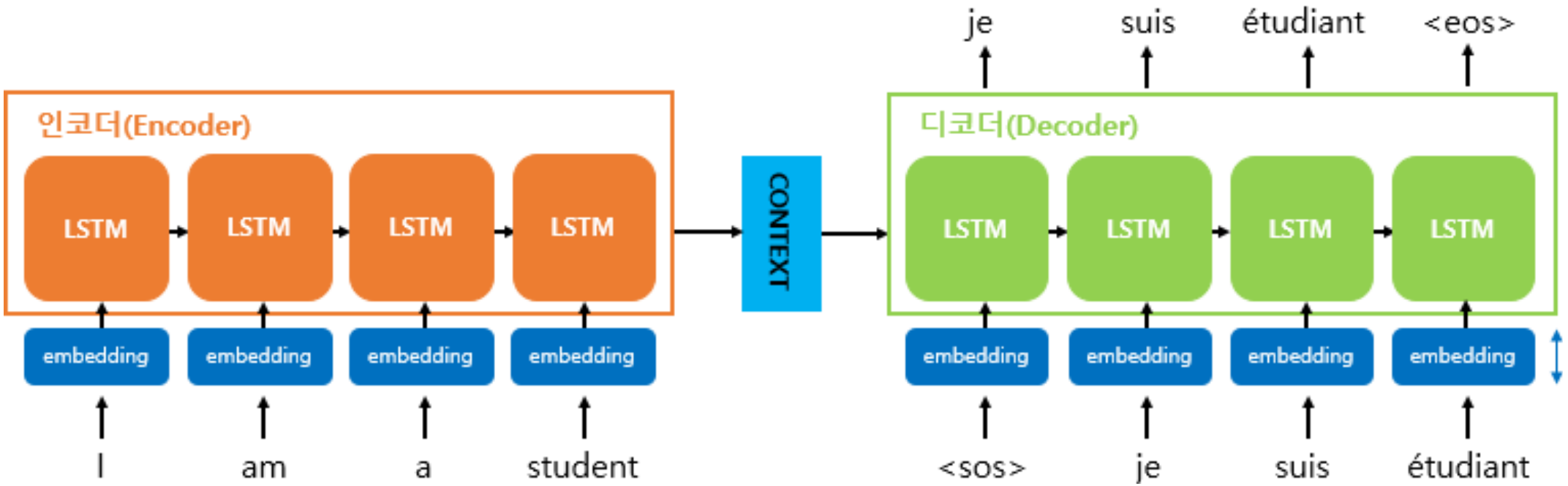
I am a student. → je suis étudiant.



I am a student. → je suis étudiant. <eos>



I am a student. → je suis étudiant.



Sequence to Sequence Learning with Neural Networks

- 본 논문은 기존 **RNN based seq2seq** 모델의 성능을 다음 세 가지 방법으로 개선한다.

- **LSTM based seq2seq**

  1. 입력과 출력에 대한 서로 다른 **2개의 LSTM**을 사용한다.

  2. **Deep LSTM**을 사용하여 **Shallow LSTM** 보다 좋은 성능을 제공한다.

  3. **입력의 순서를 뒤집어서 제공**한다. **(reversed order Input sequence)**

seq2seq 방식의 한계점

- context vector에 source 문장의 모든 정보를 압축하다보니, **bottleneck 문제**가 발생하여 모델 성능 하락의 원인이 됨

seq2seq with Attention

- Decoder에서 하나의 output을 내놓을 때마다, **입력 문장을 순차적으로 탐색해서 현재 생성하려는 부분과 가장 관련있는 영역을 적용**

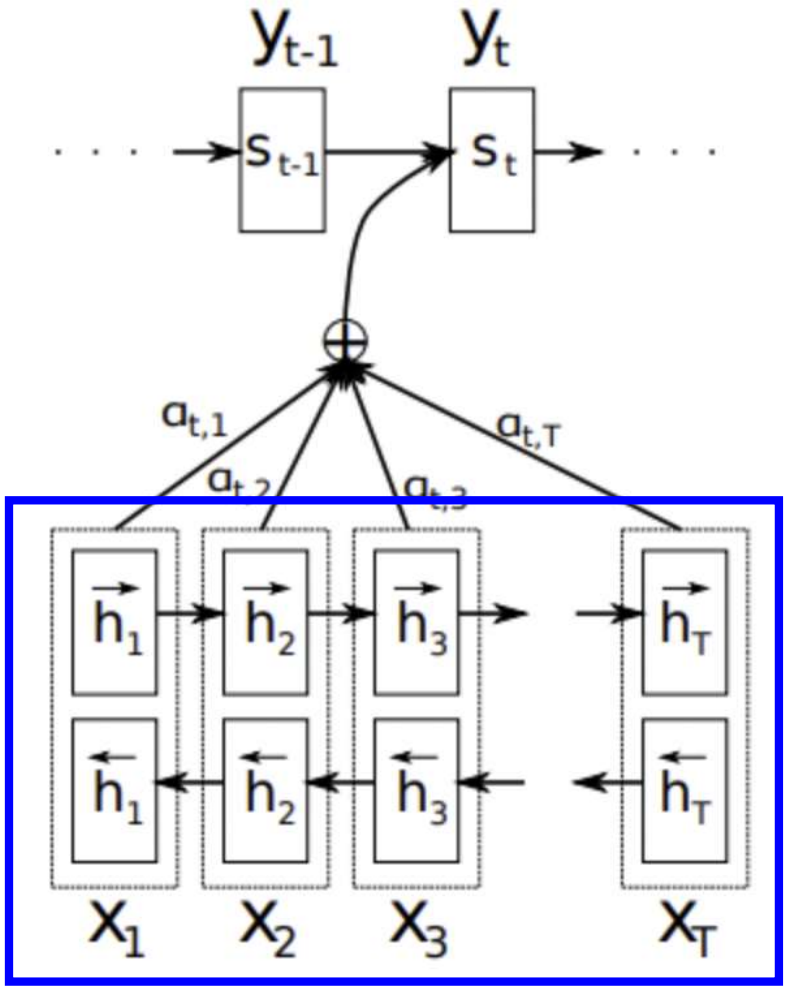
Encoder

- Encoder는 입력으로 제공되는 문장 $x = (x_1, x_2, \dots, x_T)$ 을 고정된 길이의 Vector C로 변환하게 된다.

- Bidirectional RNN 사용
  두 개의 RNN(forward RNN, backward RNN)을 사용하여 입력단어 x_j에 대한 각각의 hidden state를 생성한다.

- 매 time step j마다 생성된 forward hidden state와 backward hidden state를 concatenate하여 j번째 단어에 대한 **하나의 hidden state를 생성**

$$h_j = \left[\overrightarrow{h}_j^\top ; \overleftarrow{h}_j^\top\right]^\top$$



Decoder

- S_i : time step i 일때, Decoder의 hidden state

- **Score 계산**

각 time step j에서의 Encoder의 정보와
decoder의 time step i에서의 정보가
얼마나 연관성이 있는지를 계산 (연관성이 높은 단어를 추출하기 위함)

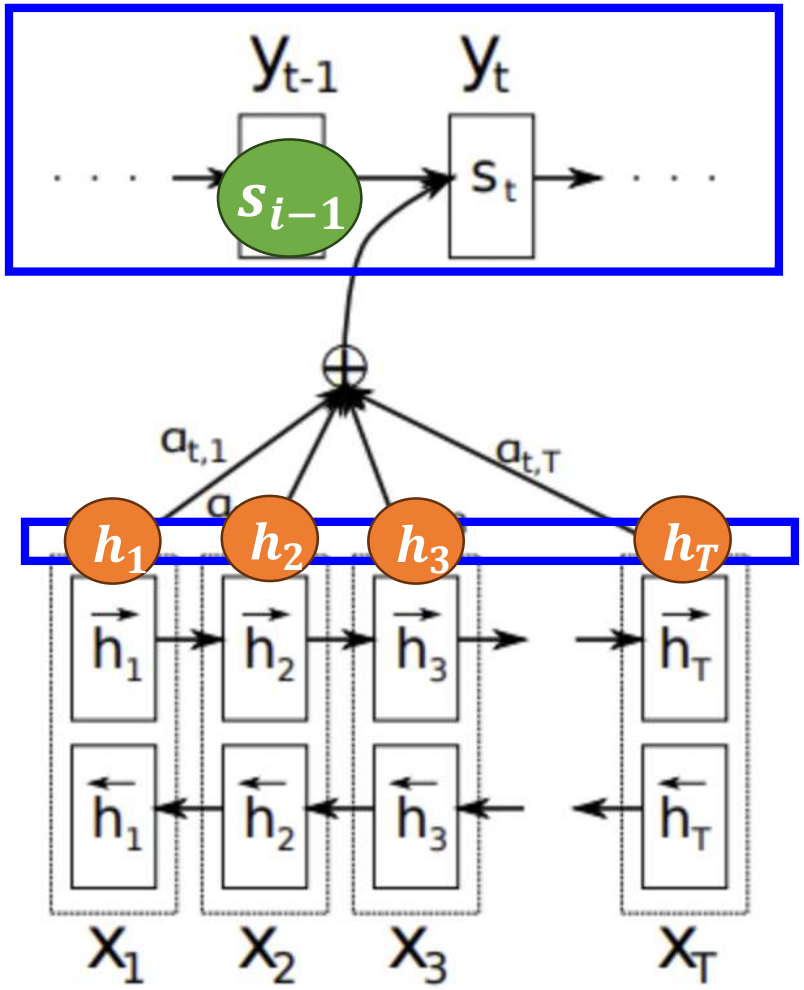
이전시점 Decoder의 hidden state

현시점 j에서 Encoder의 hidden state

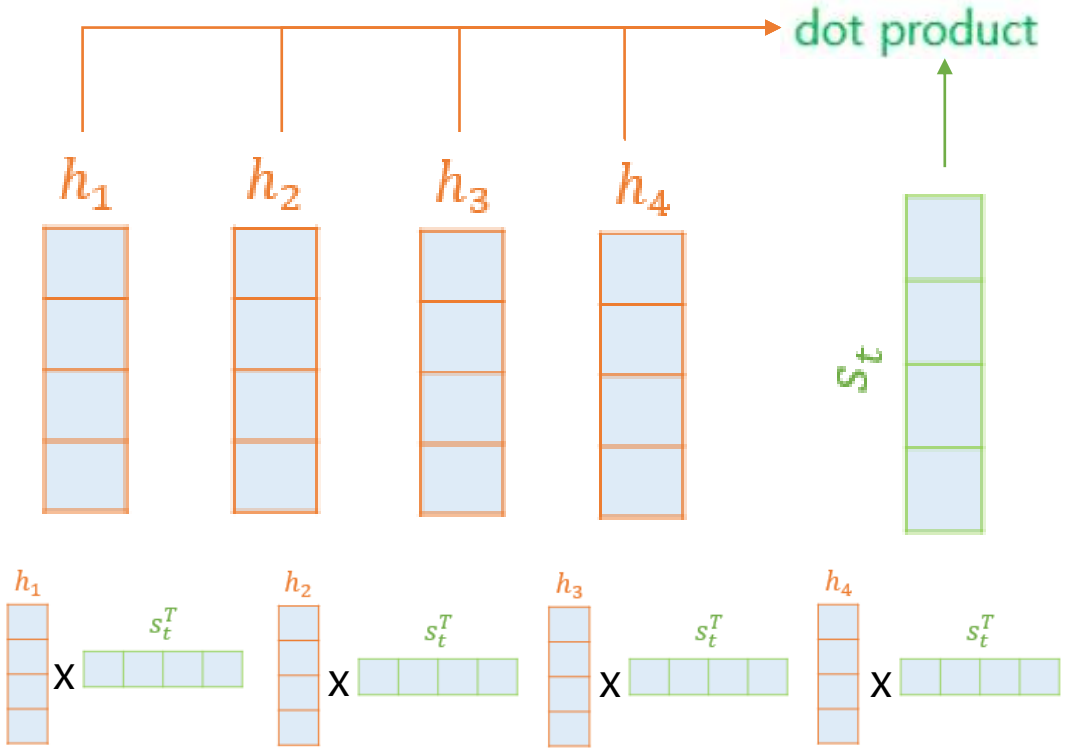
$$e_{ij} = a(s_{i-1}, h_j)$$

ALIGNMENT MODEL

$$a(s_{i-1}, h_j) = v_a^\top \tanh\left(W_a s_{i-1} + U_a h_j\right),$$

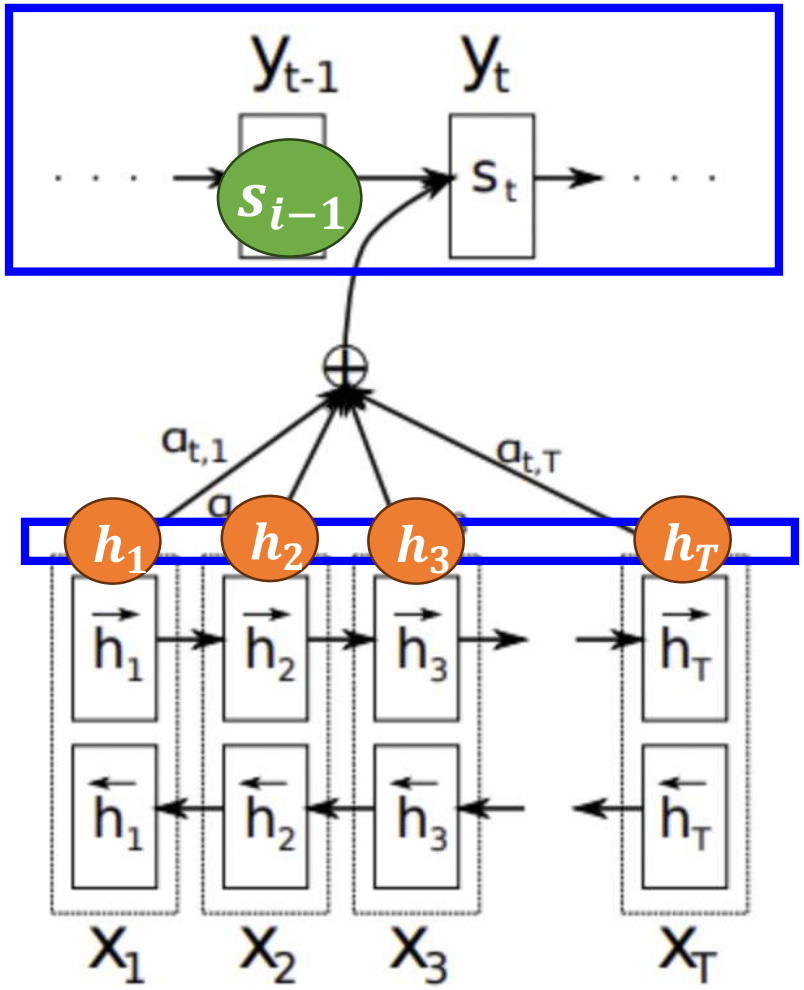


Dot-Product Attention



$$score(s_t,\ h_i) = s_t^T h_i$$

Attention Score $e^t = [s_t^T h_1, \ldots, s_t^T h_N]$



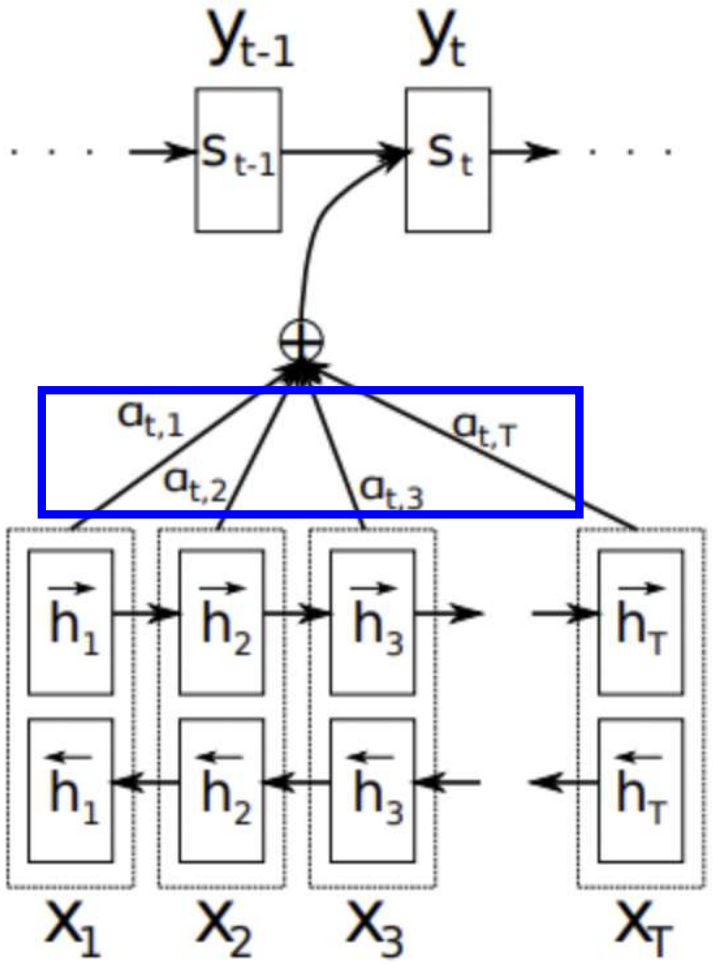
Decoder

- **Attention weight**

- score 값을 softmax 취한 것
- target word y_i가 source word x_j에 어느정도 연관이 있는지를 나타냄

Decoder의 특정 시점 i에 대해, Encoder의 모든 시점 j에 대한 attention weight(α_{ij})를 계산

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})},$$

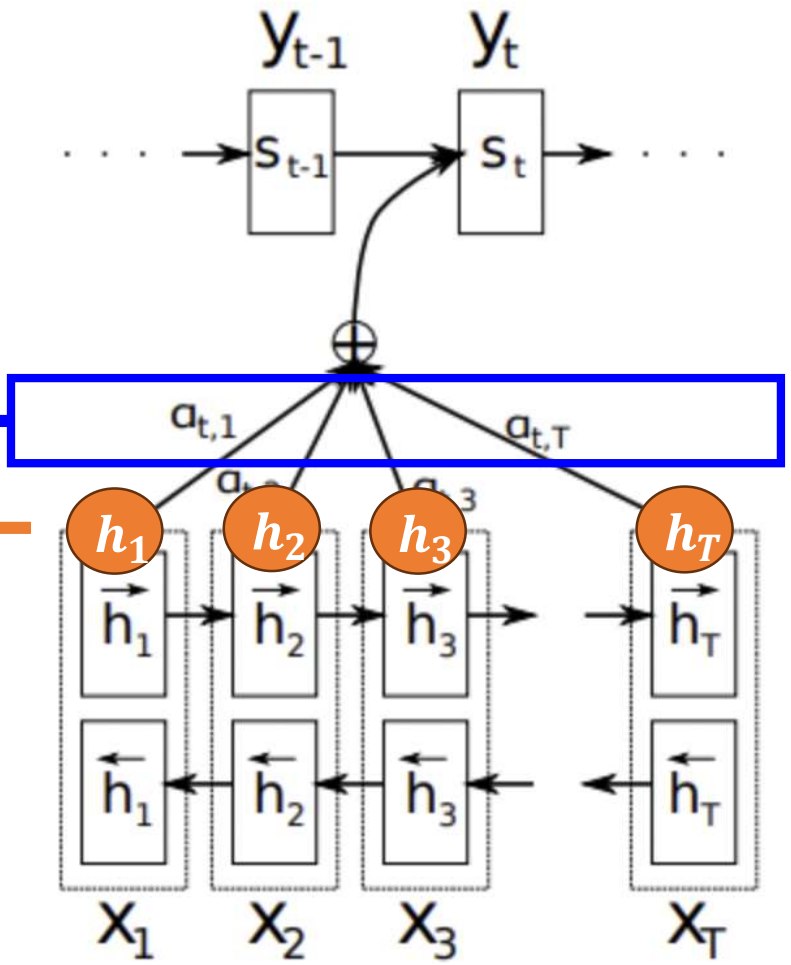


Decoder

- **Context Vector**

어텐션 메커니즘을 사용하는 seq2seq 모델에서는
각 디코딩 시점마다 context vector을 다시 계산한다.

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j .$$

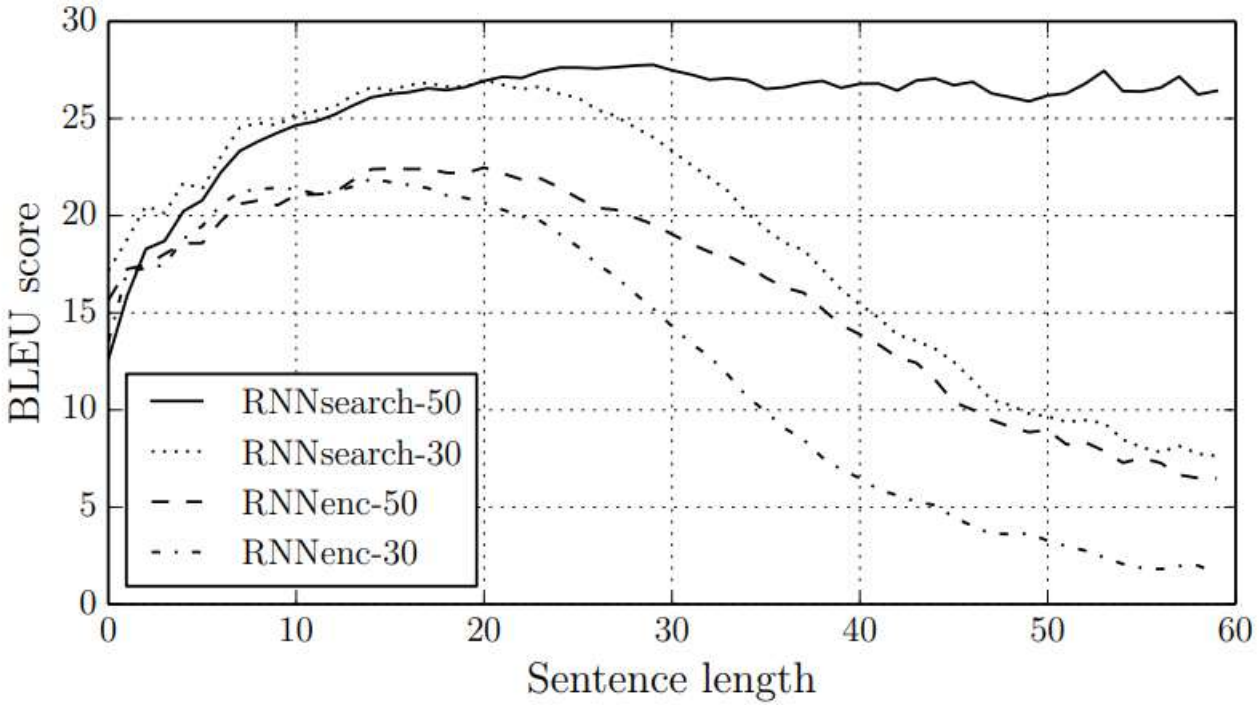


[그림-1]

Results

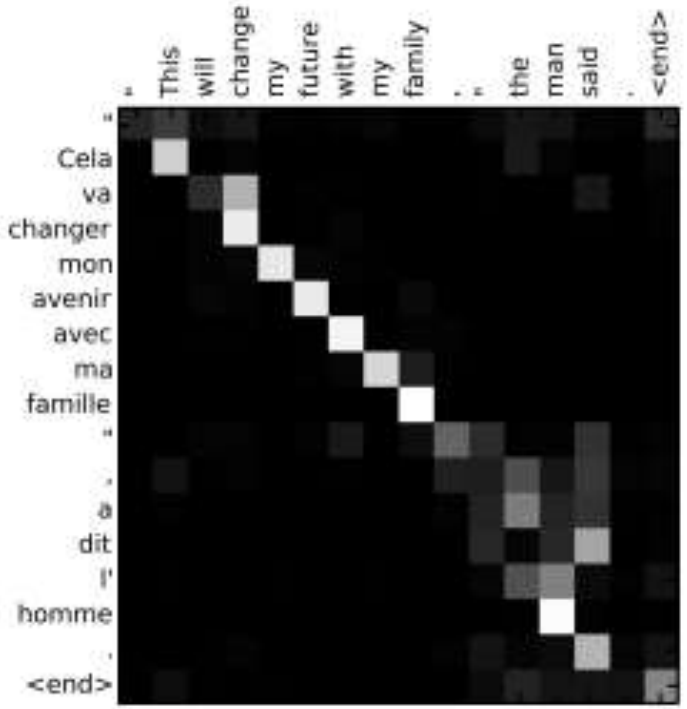
RNNencdec : RNN 인코더-디코더
RNNsearch : 논문에서 제안한 모델

Model	All
RNNencdec-30	13.93
RNNsearch-30	21.50
RNNencdec-50	17.82
RNNsearch-50	26.75



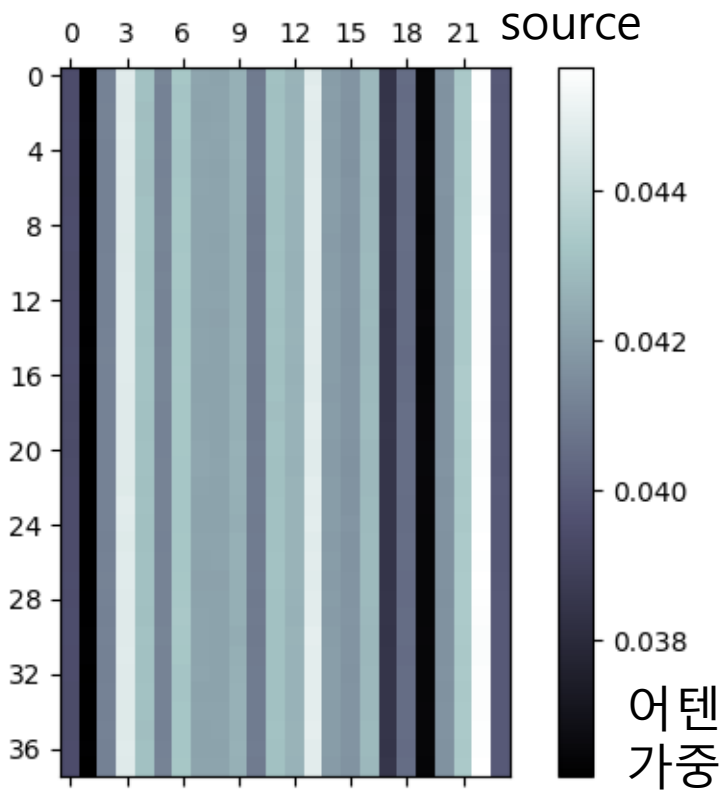
source :
사실 판다가 뭐가 그렇게 특별하다고 난리냐고 생각도 했었는데 큰 착각이었어요. 나무에 올라간 푸바오가 맛있게 대나무를 먹는 걸 보는데 왜 사람들이 열광할 수밖에 없는지 알겠더라고요.

target :
In fact, I wondered what was so special about pandas, but it was a big mistake. When I saw the fubao on the tree eating bamboo deliciously, I understood why people were so enthusiastic.



(d)

어텐션 가중치의 크기



어텐션
가중치의 크기