
NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE

Dzmitry Bahdanau Jacobs University Bremen, Germany
(2015)

2024.03.03
유하영

seq2seq 방식의 한계점

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

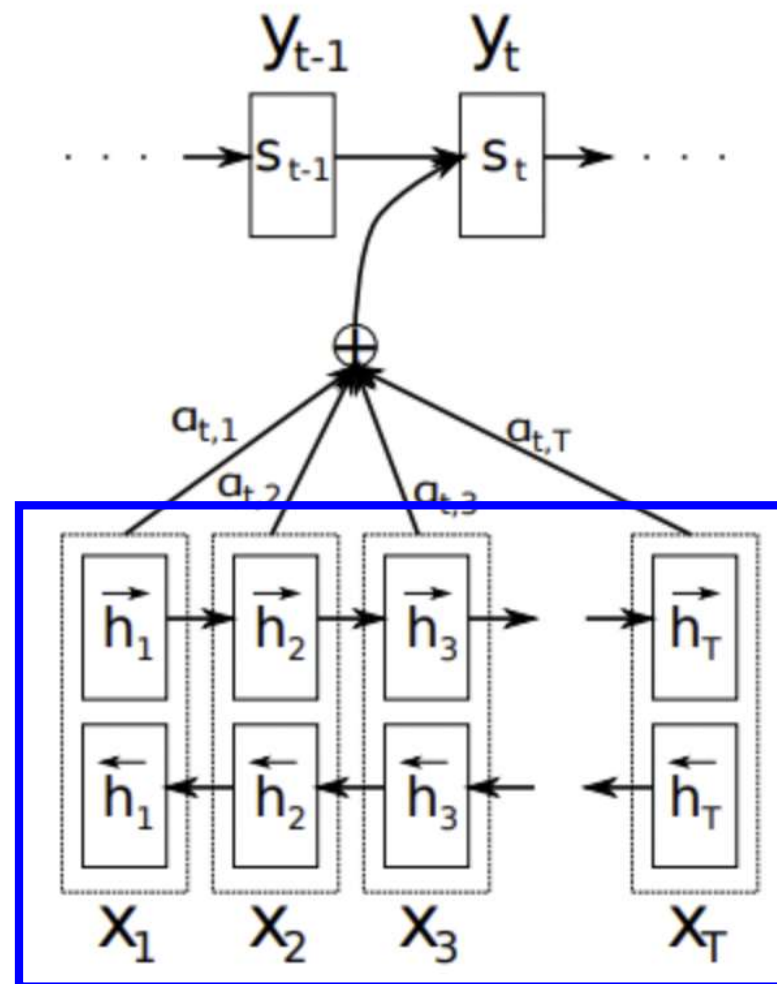
seq2seq with Attention

- 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 C 로 변환하게 된다.
- Bidirectional RNN 사용
두 개의 RNN(forward RNN, backward RNN)을 사용하여 입력단어 x_j 에 대한 각각의 hidden state를 생성한다.
- 매 time step j 마다 생성된 forward hidden state와 backward hidden state를 concatenate하여 j 번째 단어에 대한 하나의 hidden state를 생성

$$h_j = [\vec{h}_j^T; \overleftarrow{h}_j^T]^T$$



Decoder

- S_i : time step i 일때, Decoder의 hidden state

- **Score 계산**

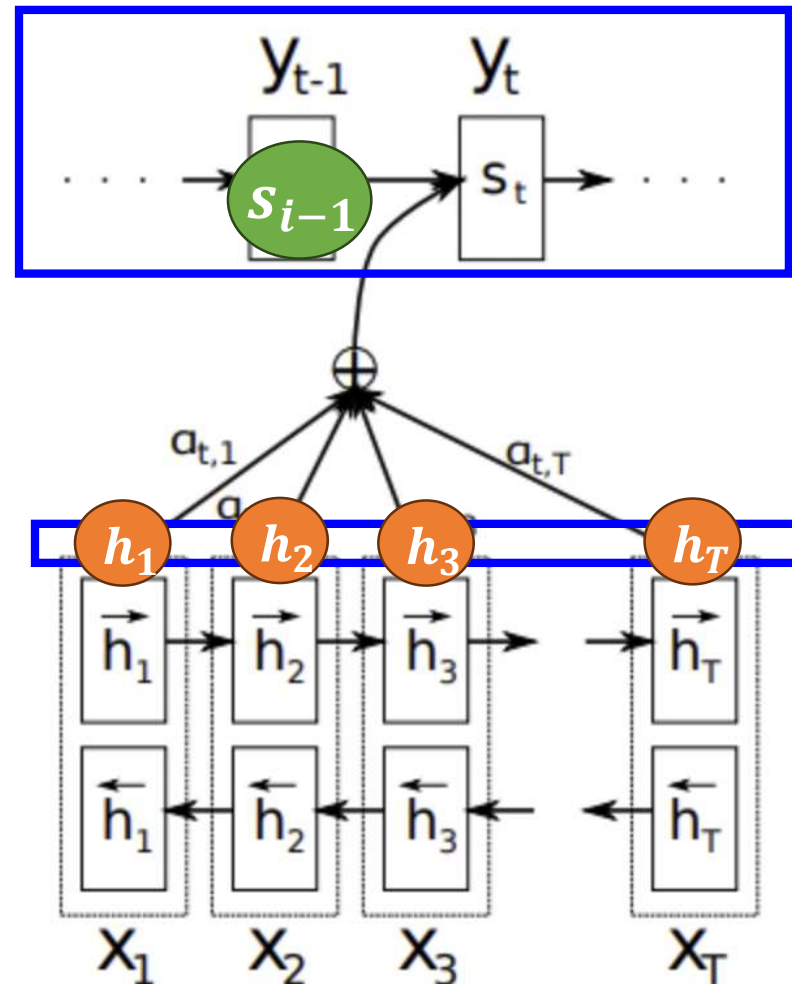
각 time step j 에서의 **Encoder의 정보**와
decoder의 time step i 에서의 정보가
 얼마나 연관성이 있는지를 계산 (연관성이 높은 단어를 추출하기
 위함)

이전시점 현시점 j 에서
 Decoder의 hidden state Encoder의 hidden state

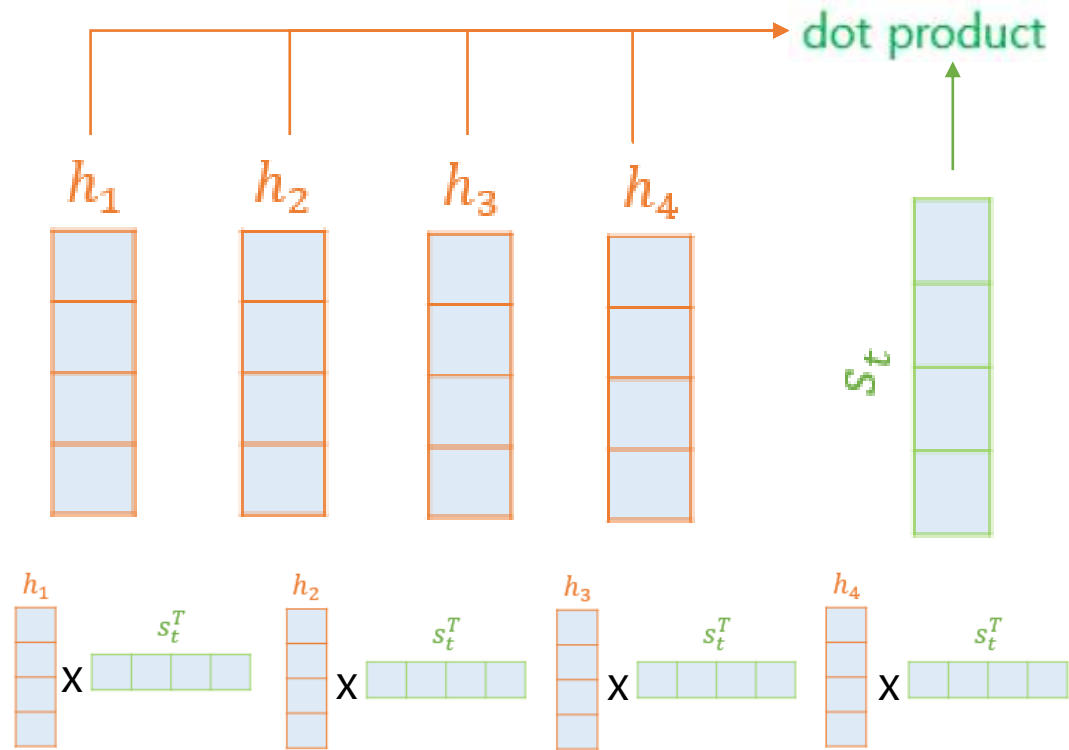
$$e_{ij} = a(s_{i-1}, h_j)$$

ALIGNMENT MODEL

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

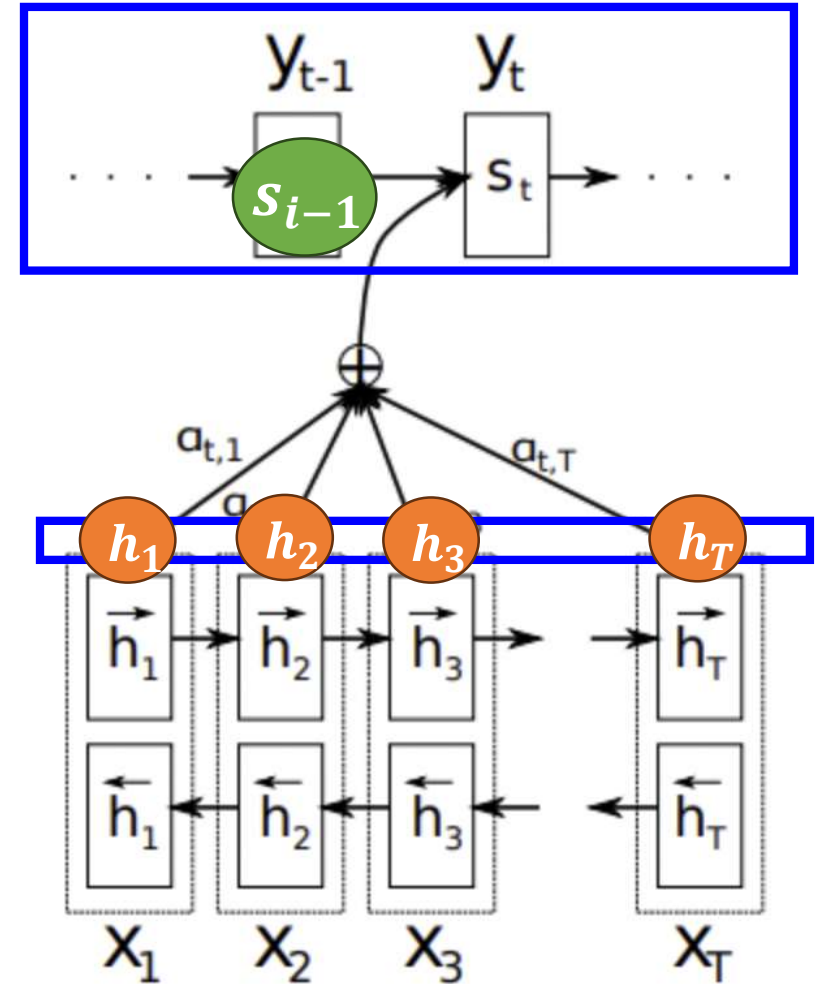


Dot-Product Attention



$$\text{score}(s_t, h_i) = s_t^T h_i$$

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



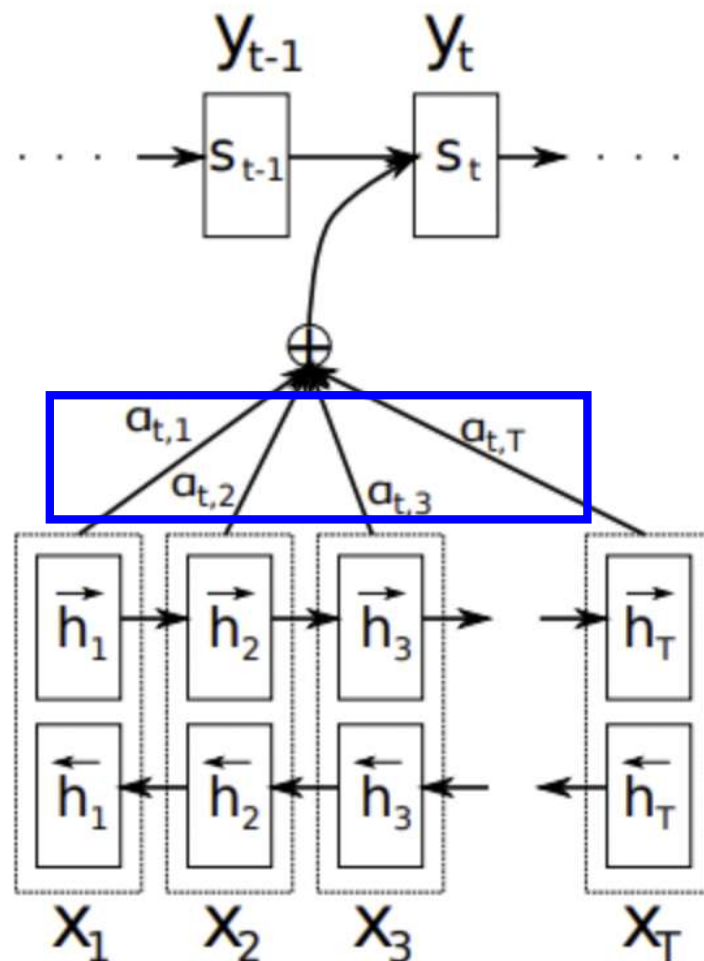
Decoder

- **Attention weight**

- score 값을 softmax 취한 것
- target word y_i 가 source word x_j 에 어느정도 연관이 있는지를 나타냄

Decoder의 특정 시점 i 에 대해, Encoder의 모든 시점 j 에 대한 attention weight(α_{ij})를 계산

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

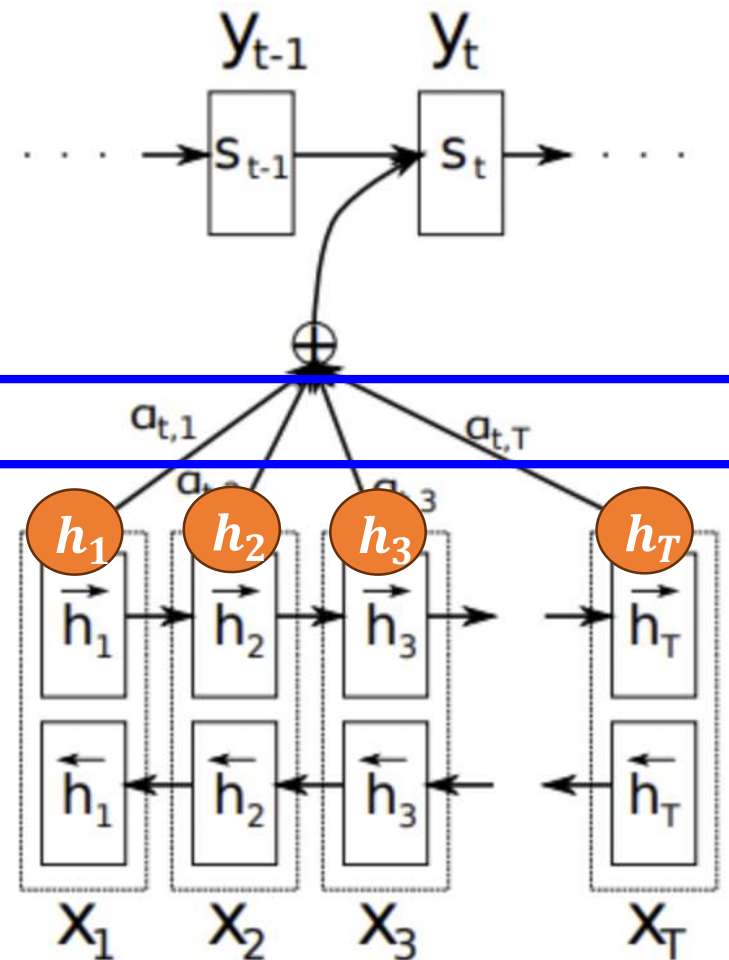


Decoder

- Context Vector**

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

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



[그림-1]

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

