Attention Mechanism

Cho Sung Man

첨가물

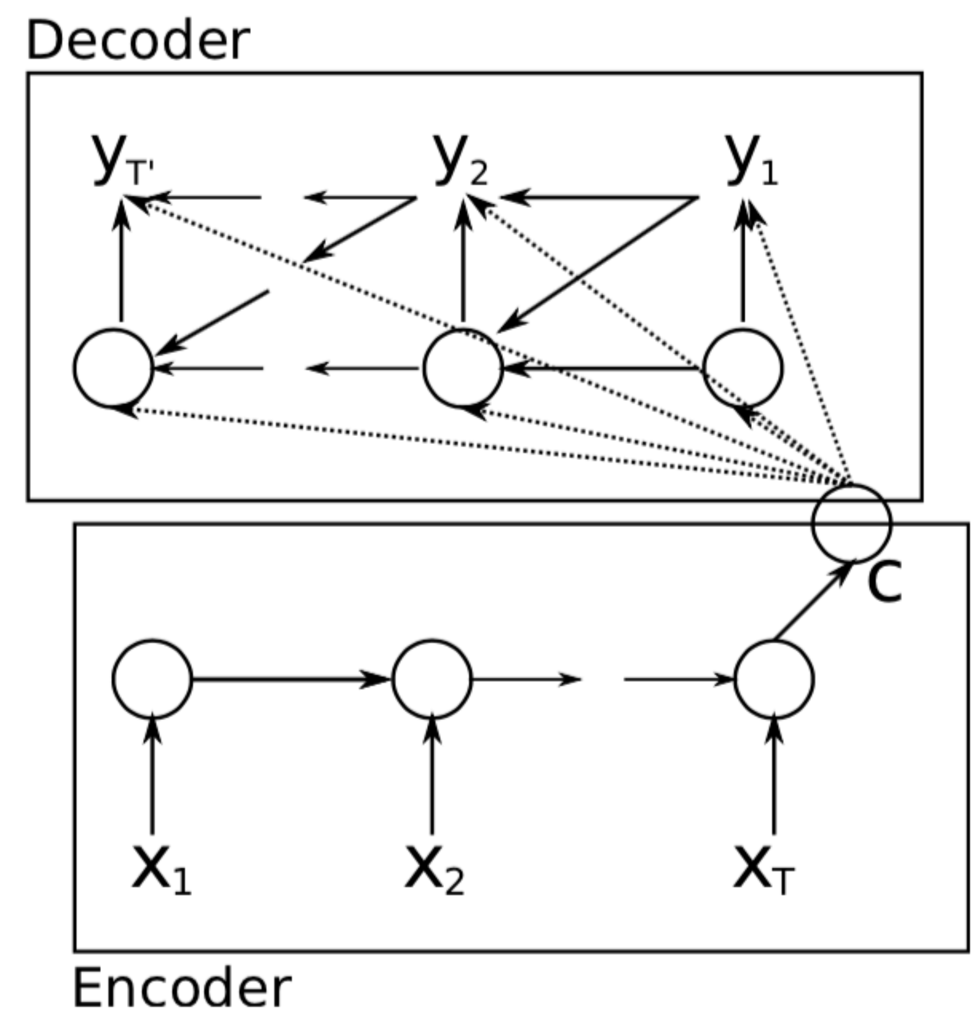
- 왜 Attention Mechanism 을 사용했는지 ?
- Attention Mechanism 의 동작 과정
- 약간의 수학..?

왜 ??

- 일반적으로, LSTM이 Long-term dependency 문제를 잘 처리 한다고 알고 있었지만, 여전히 문제가 있다 !

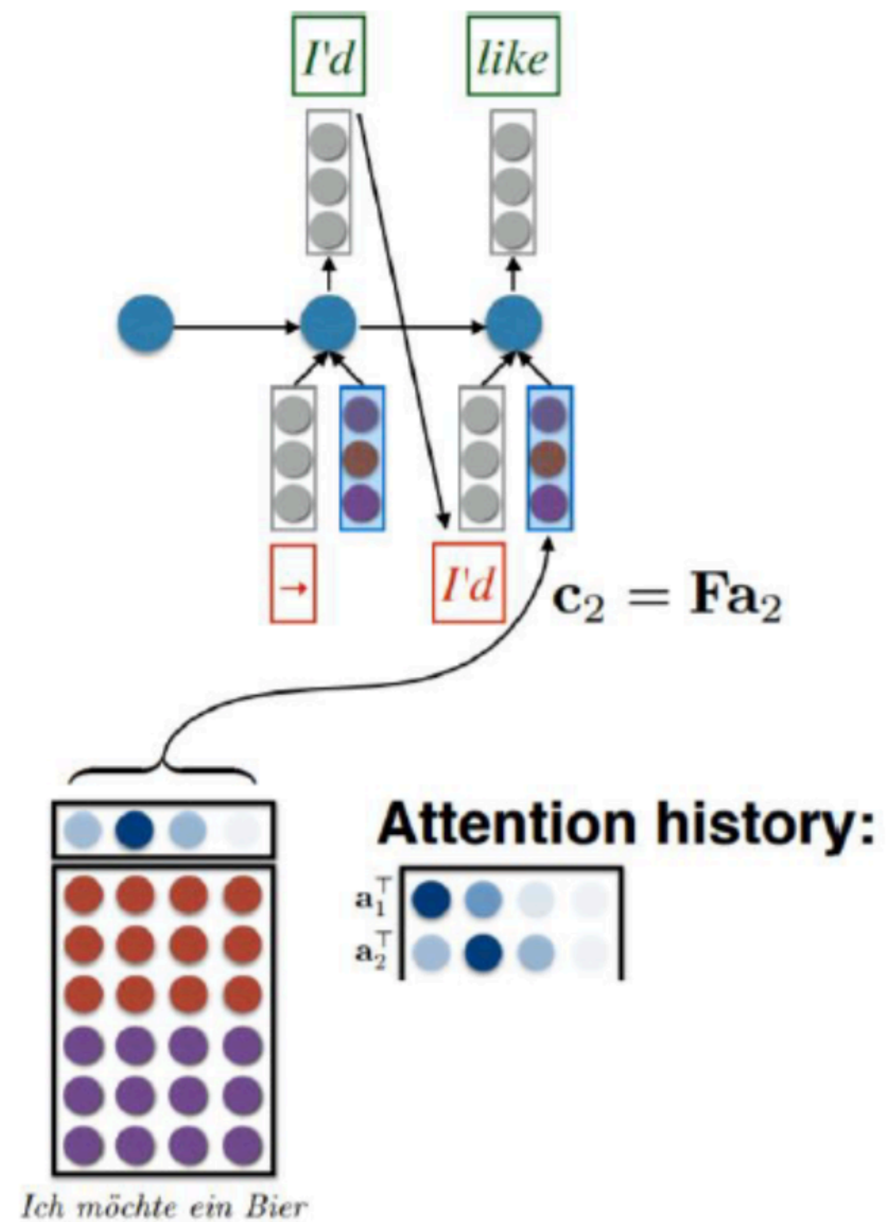
- Fixed-length vector는 basic encoder-decoder 구조의 성능향상에 병목을 일으킨다.

[RNN Encoder-Decoder Architecture]



그래서 ?

- 일반적으로, LSTM이 Long-term dependency 문제를 잘 처리 한다고 알고 있었지만, 여전히 문제가 있다 !
  - Target Word에 해당하는 부분을 Input Sequence 에서 찾아내자 !
- Fixed-length vector는 basic encoder-decoder 구조의 성능향상에 병목을 일으킨다.
  - Input Sequence를 fixed-length vector 하나에 넣지 말고, vector sequence로 구성하자 !
- Long Sentence에 대하여 훨씬 더 좋은 결과를 얻음 :)




존나좋군?

우선 RNN을 다시보면

RNN 리뷰

$$\mathbf{x} = (x_1, \cdots, x_{T_x}) \qquad \mathbf{y} = (y_1, \cdots, y_{T_y}).$$

NMT의 목표 : $\arg\max_{\mathbf{y}} p(\mathbf{y} \mid \mathbf{x}).$

Hidden State Vecotr : $h_t = f(x_t, h_{t-1})$

$$c = q(\{h_1, \cdots, h_{T_x}\}),$$

Non-linear Function

좀 더 자세히 알아볼까..?

Decoder

$$p(\mathbf{y}) = \prod_{t=1}^{T} p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c),$$

$$p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c) = g(y_{t-1}, s_t, c),$$

Decoder

$$p(\mathbf{y}) = \prod_{t=1}^{T} p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c),$$

Previous Predicted Vector

$$p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c) = g(y_{t-1}, s_t, c),$$

Decoder

$$p(\mathbf{y}) = \prod_{t=1}^{T} p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c),$$

Context Vector

$$p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c) = g(y_{t-1}, s_t, c),$$

Decoder

$$p(\mathbf{y}) = \prod_{t=1}^{T} p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c),$$

$$p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c) = g(y_{t-1}, s_t, c),$$

Non-linear Function

Decoder

$$p(\mathbf{y}) = \prod_{t=1}^{T} p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c),$$

$$p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c) = g(y_{t-1}, s_t, c),$$

Hidden State

Decoder

$$p(\mathbf{y}) = \prod_{t=1}^{T} p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c),$$

$$p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c) = g(y_{t-1}, s_t, c),$$

Context Vector

그럼 Context Vector는 ..?

유사의 척도를 구하자

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

유사의 척도를 구하자

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

직전 스텝의 Hidden State Vector

유사의 척도를 구하자

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

인코더의 j 번째 Column Vector

유사의 척도를 구하자

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

Alignment Model

Alignment Model은 s_{i-1} 과 h_j 간 유사도를 잘 뽑아낼 수 있다면 다양한 변형 가능.

$$\boxed{\gamma^T tanh(WF + V_{S_i-1}) \qquad FVS_{i-1}}$$

자매품

Attention & Context

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

'합이 1이 되는 확률값' 으로 변형

$$\overrightarrow{\alpha_i} = \left[\alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{iT_x}\right]$$

Decoder가 i 번째 단어를 예측할 때 쓰이는 Attention Vector α_i

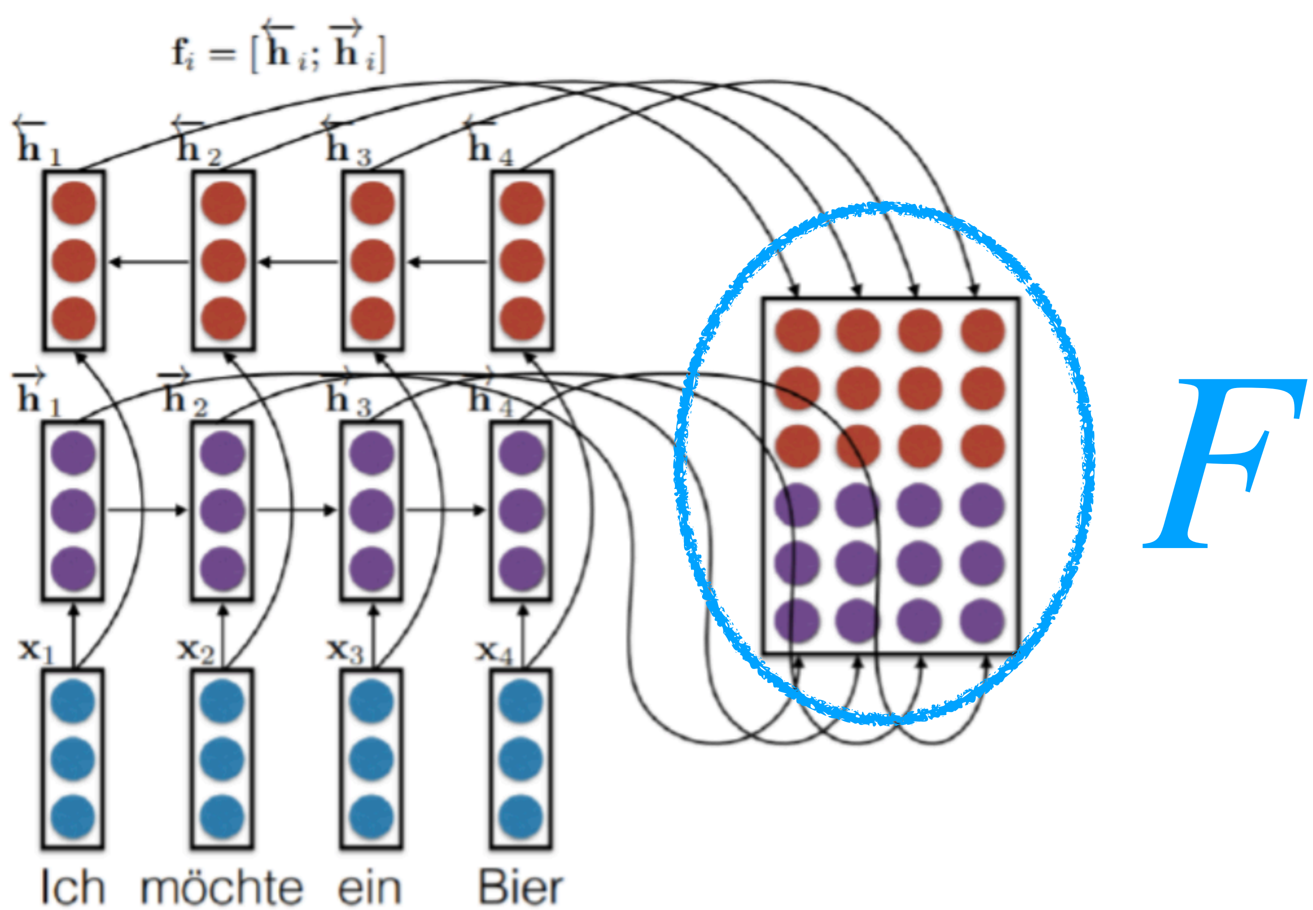
$$\overrightarrow{c_i} = \sum_{j=1}^{T_x} \alpha_{ij} h_j = F\overrightarrow{\alpha_i}$$

i 번째 단어를 예측할 때 쓰이는 Context Vector C_i

Attention & Context

$$\overrightarrow{c_i} = \sum_{j=1}^{T_x} \alpha_{ij} h_j = F \overrightarrow{\alpha_i}$$

i 번째 단어를 예측할 때 쓰이는 Context Vector C_i



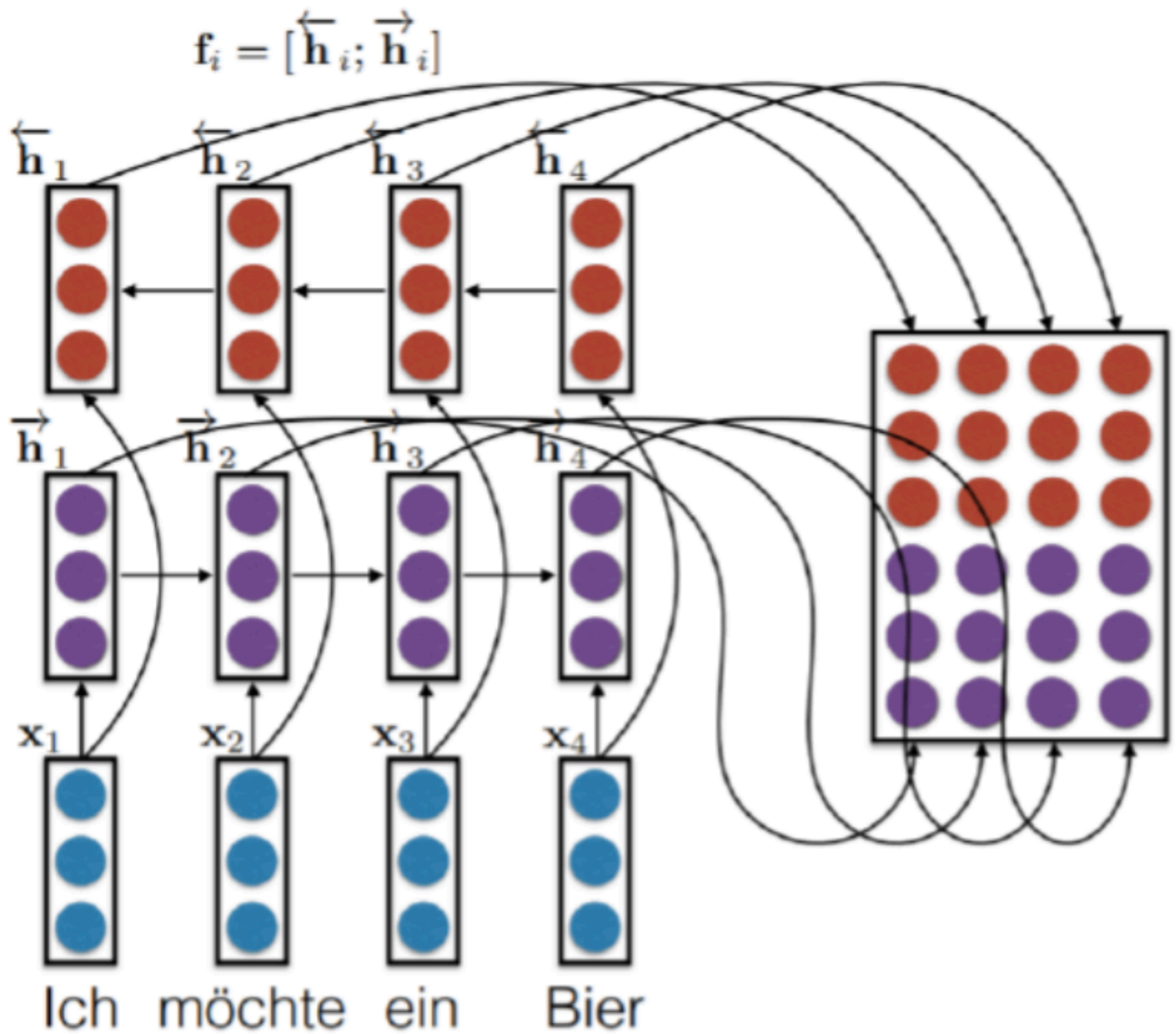
인코더는 아무것도 안바꿔?

Bi-RNN을 사용

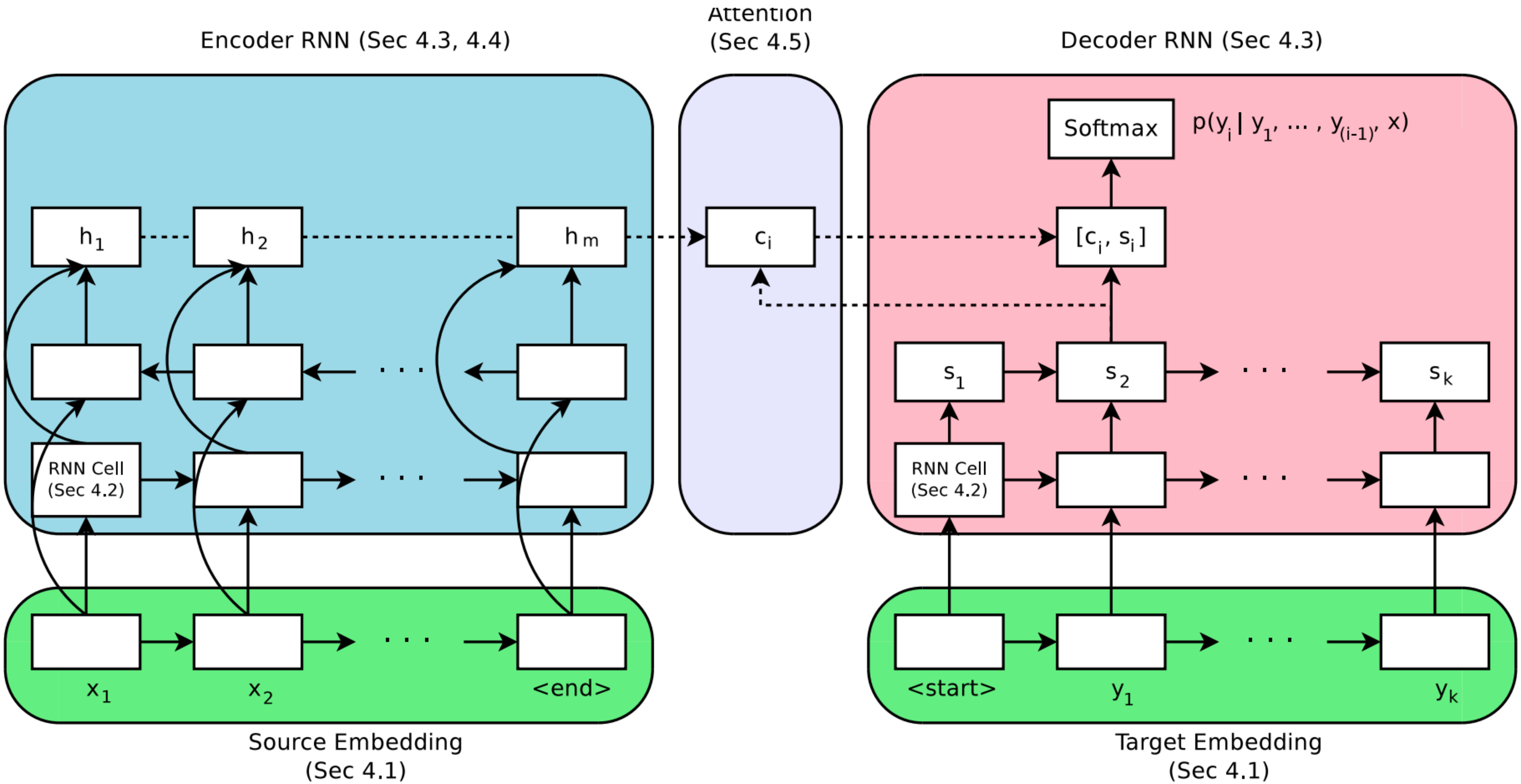
$$(\overrightarrow{h}_1, \cdots, \overrightarrow{h}_{T_x}).$$

$$(\overleftarrow{h}_1, \cdots, \overleftarrow{h}_{T_x}).$$

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



Total Diagram



Thank You.