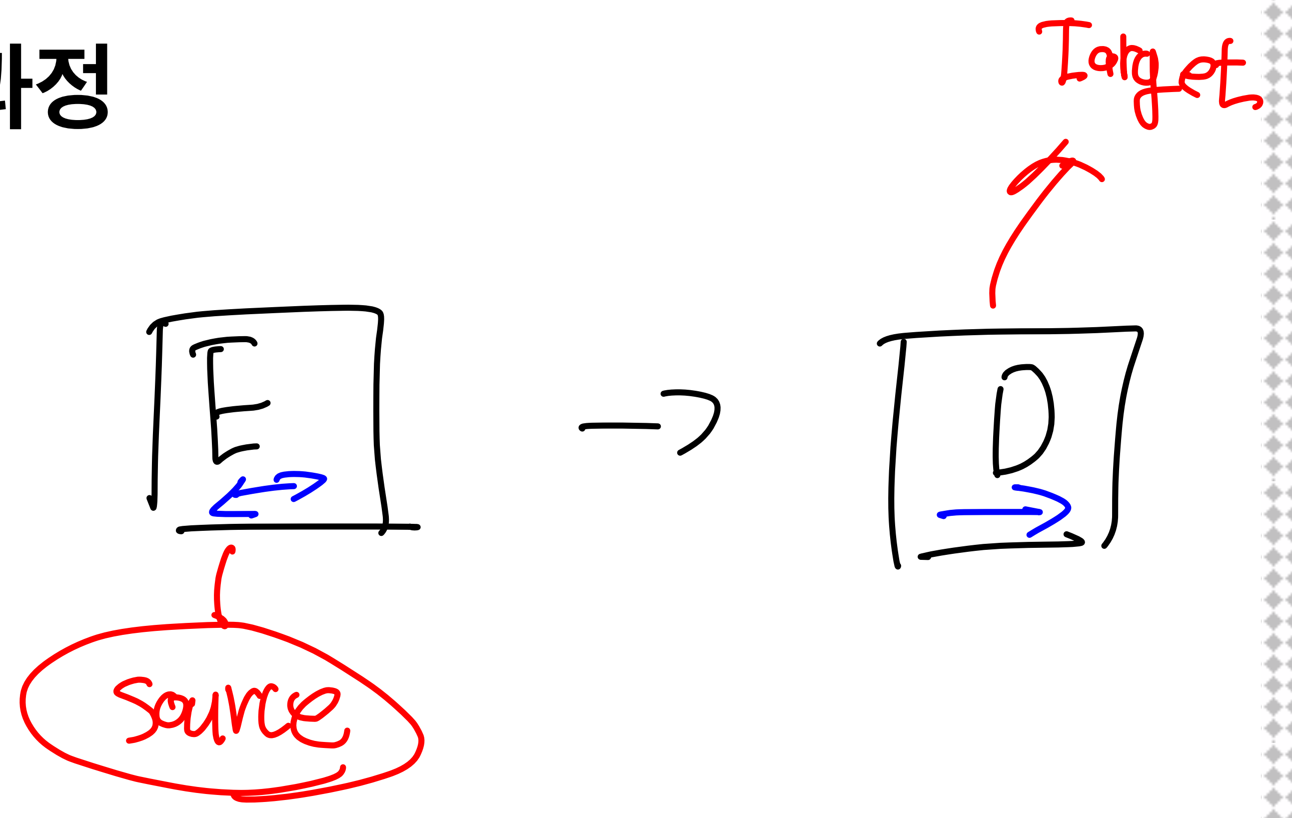
ICT이노베이션스퀘어 AI복합교육 고급 언어과정

자연어처리를 위한 Attention



NMT
+
Attention

현청천

2021.04.19

분류 문제에 좋음 //

Encoder

∘ feature 추출

Soure

∘ Transformer

∘ bert

Target (Next)

Decoder

∘ 생성 모델

Target (Previous)

∘ GPT ex) 소설. 대화. 오타 검색 (교정)

∘ google에서 최근 경향

What is Attention

주어진 목적에 맞는 Source에 집중

ex) 감정분석

할머니 만나는 부분에서 울었습니다. 감동적인 영화입니다.

o 아 - 얘기 중이구나

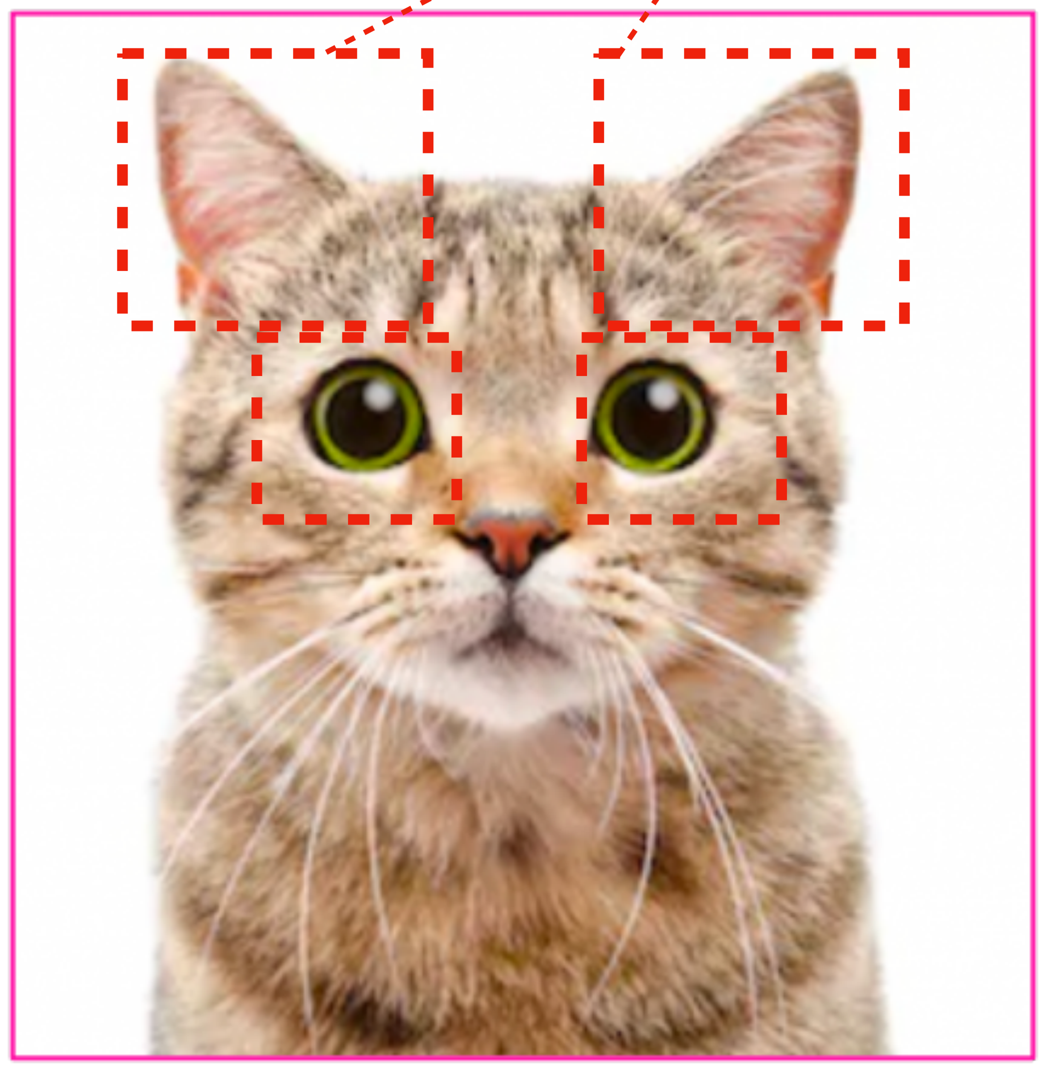
아 - 긍정이구나

긍정

What is Attention

주어진 목적에 맞는 Source에 집중

이미지분류

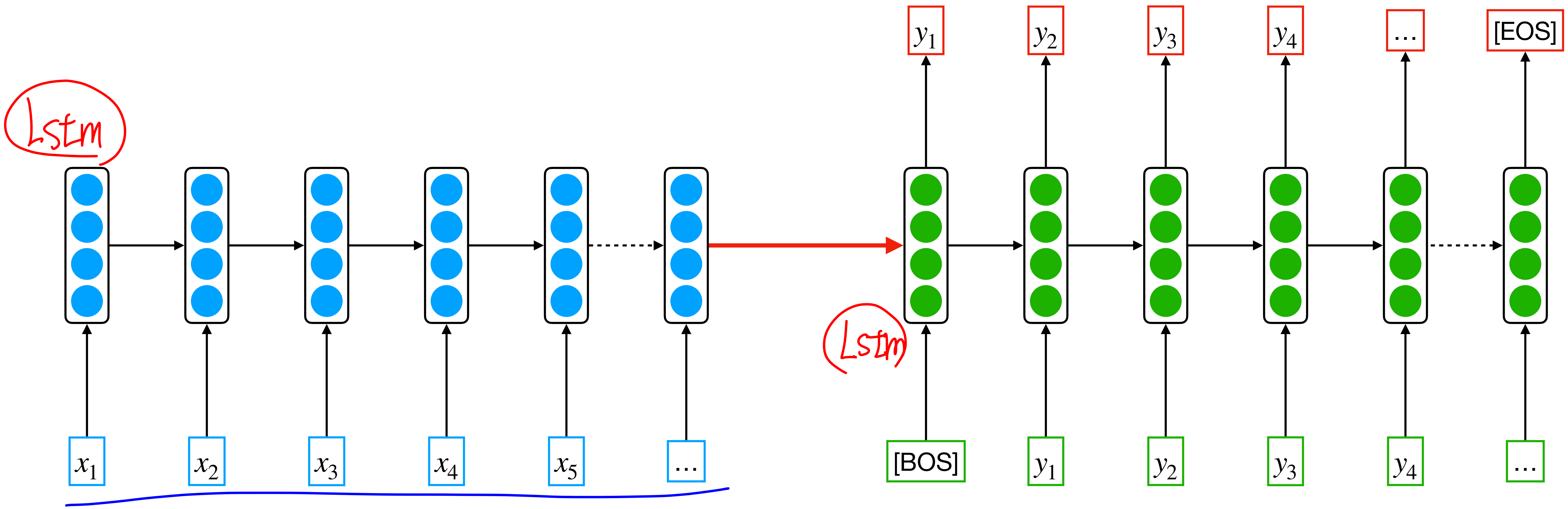


고양이

What is Attention

N-M-T

나는 학생 입니다

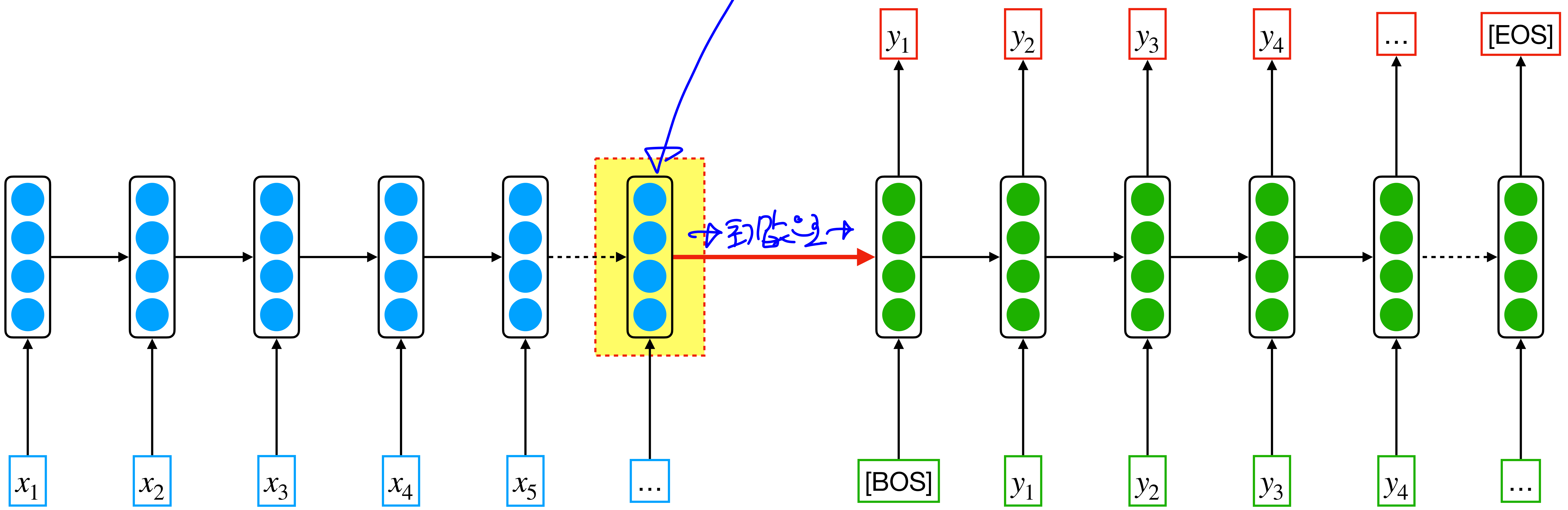


Source

I am student

What is Attention

Encoder Input 정보를 하나의 벡터로 저장



What is Attention

Encoder Input 정보를 하나의 벡터로 저장

한계 ① 긴 문장을 하나의 벡터로 변환하기 어려움 (Information bottleneck)

⇒ 당연히 정보 손실이 생기지.

if) token이 500개일때.

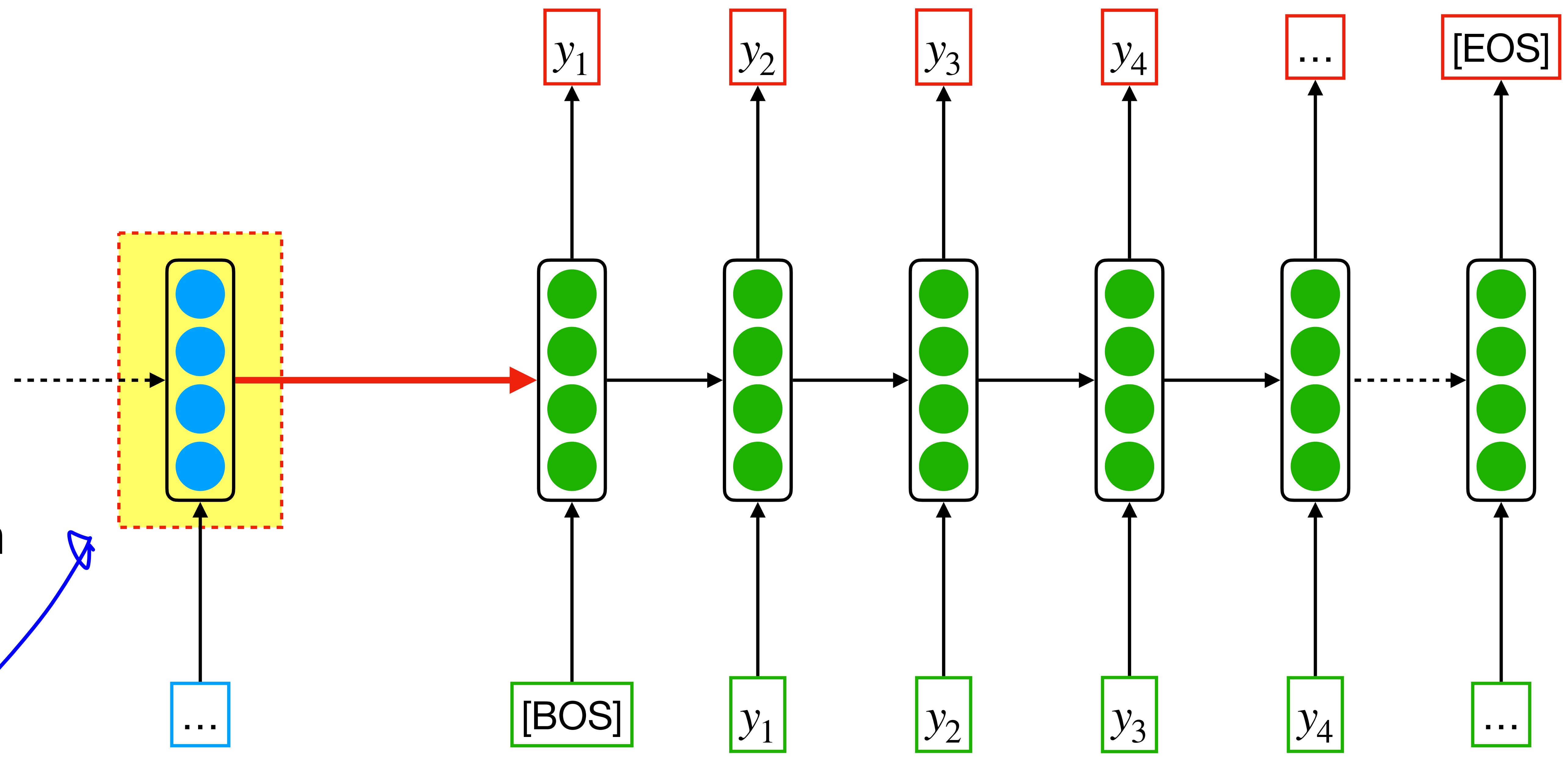
The dominant sequence transduction models are based on complex

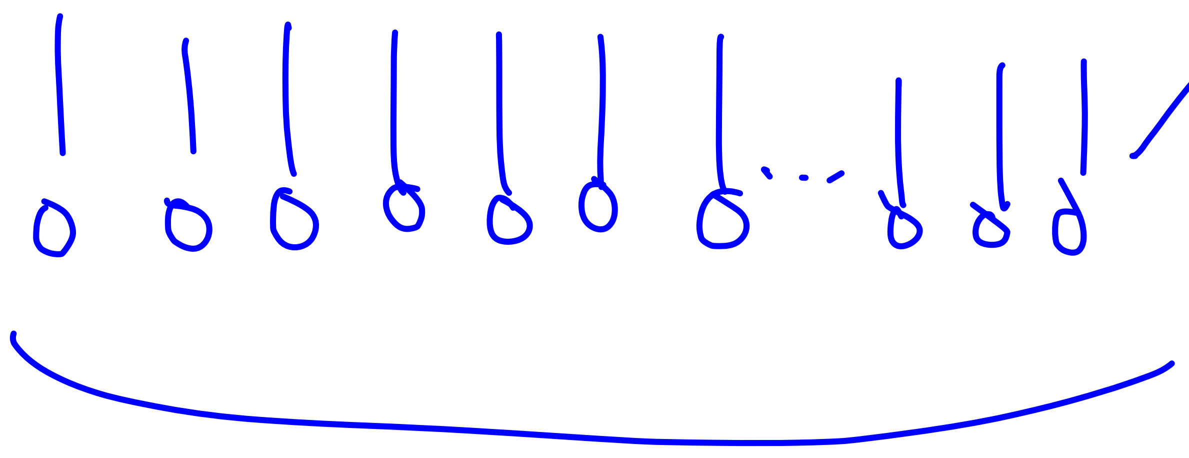
…

…

…

requiring significantly less time to train



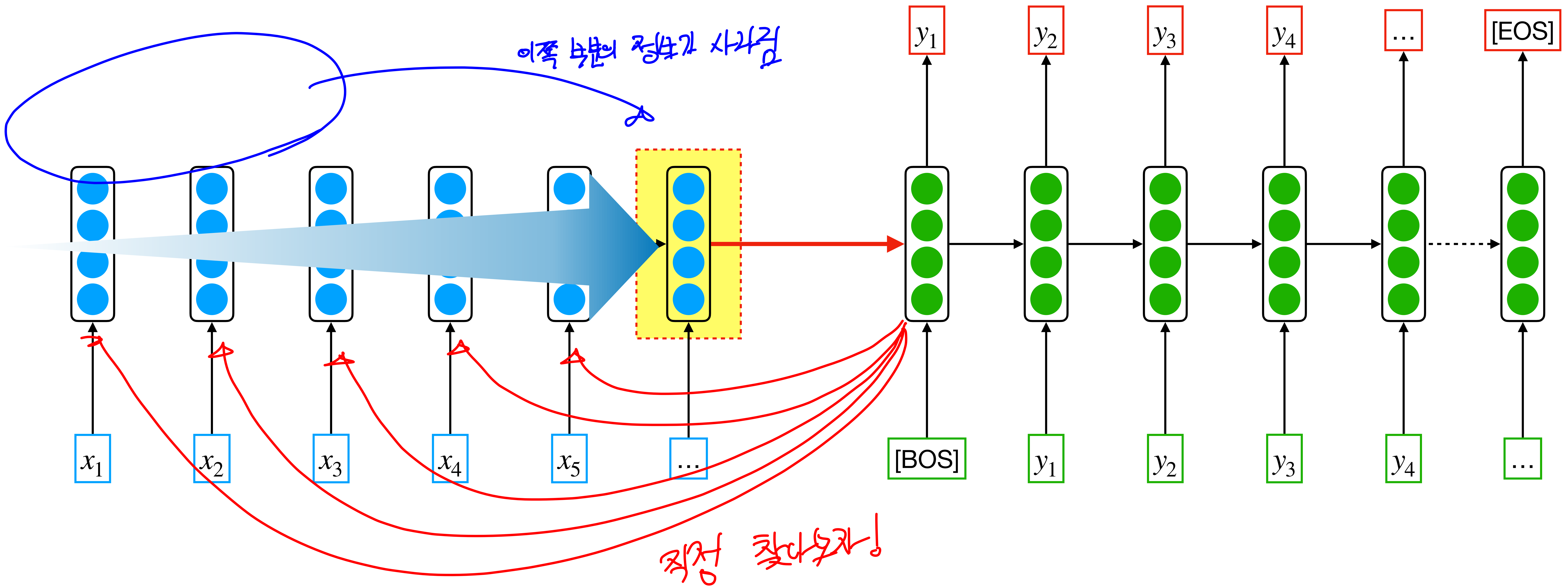


500개.

What is Attention

Encoder Input 정보를 하나의 벡터로 저장

한계② 과거의 정보가 점점 사라짐 (Vanishing gradient)



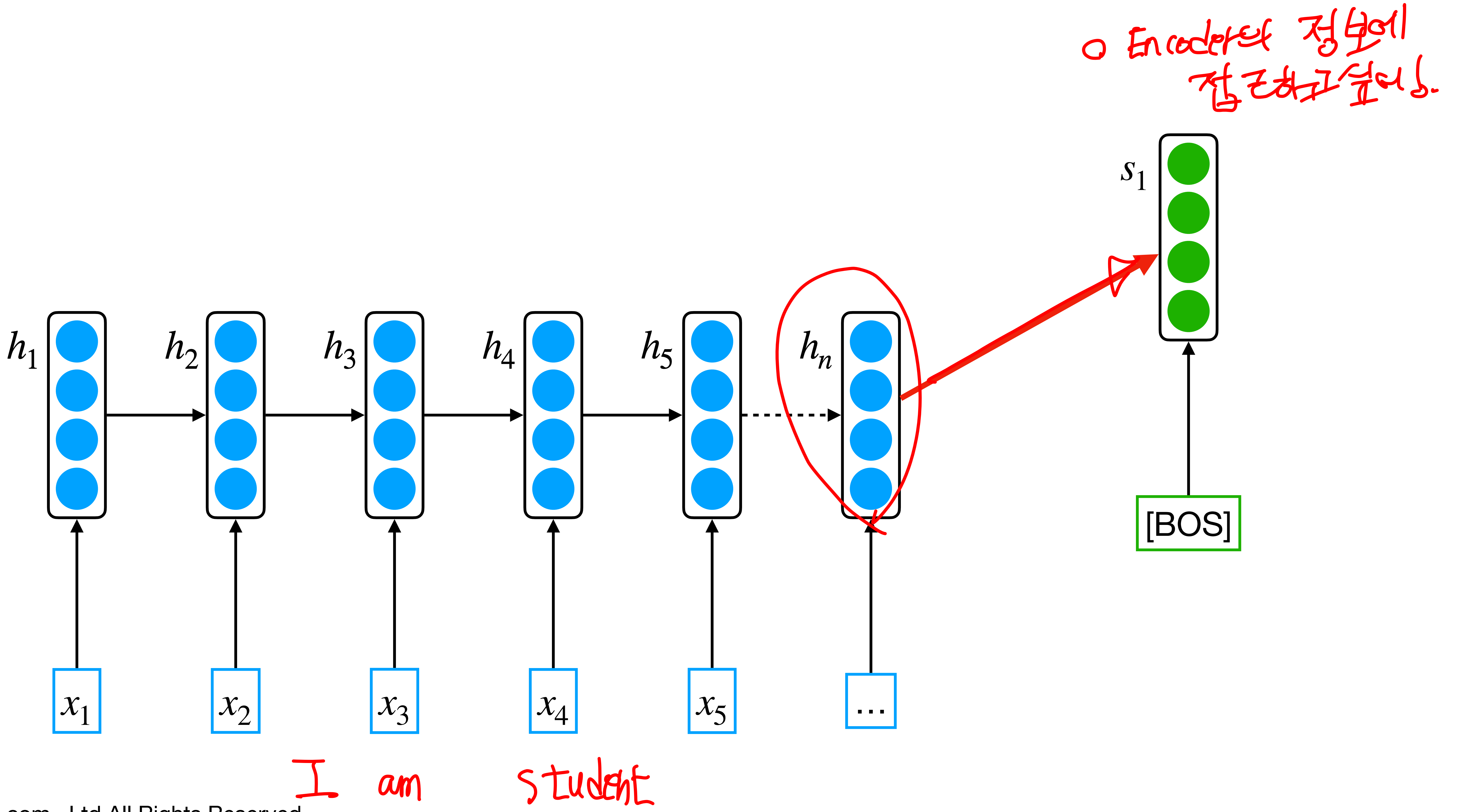
What is Attention

- 두가지 주요 문제 해결
  - 긴 문장을 하나의 벡터로 변환하면서 발생하는 Information bottleneck
  - 과거의 정보가 점점 사라지는 vanishing gradient

Encoder의 특정 부분

Source의 특정 부분을 집중하기 위해 Decoder가 Encoder의 정보를 직접 접근함

Attention Model



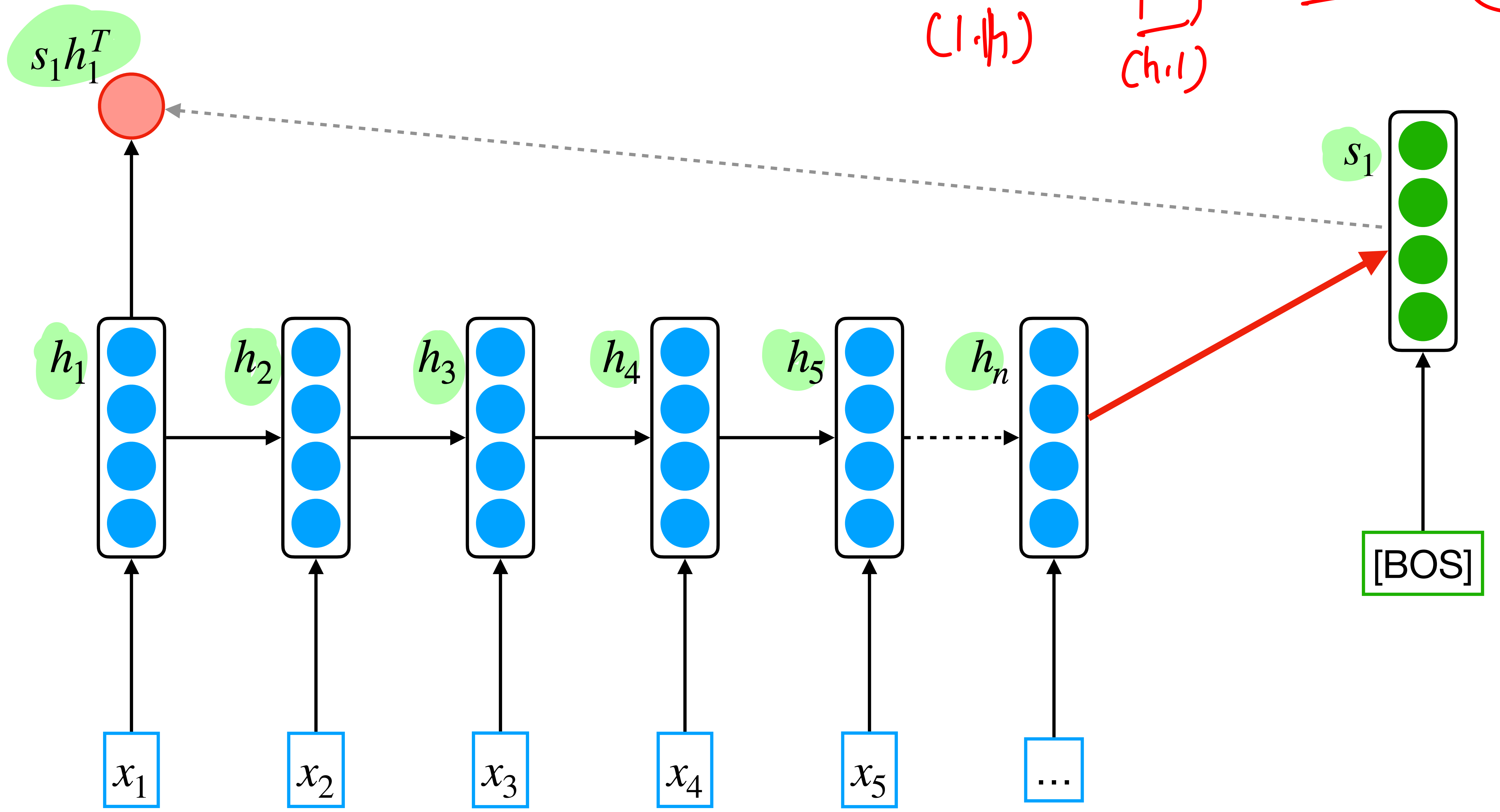
Attention Model

Dot-product

s_1 $(h,1)$ h_1 $(h,1)$

$\boxed{s_1^T} \cdot \boxed{h_1} = 1 =$ 스칼라 값
$(1,h)$ $(h,1)$

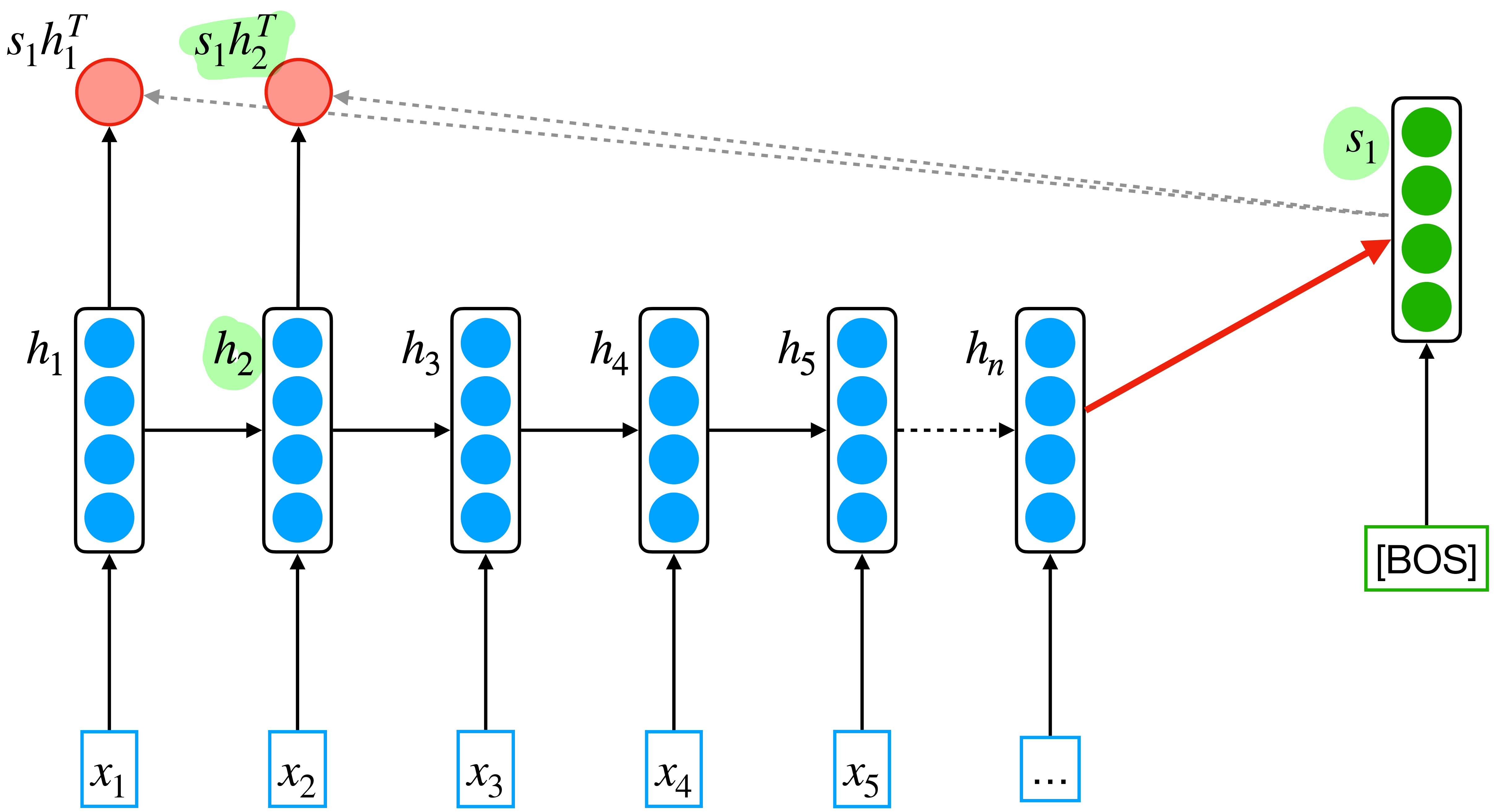
→ Dot-Product
$\vec{a} \cdot \vec{b} = |a||b|\cos\theta$
유사도 지표



Attention Model

shape이 같아야지만 된다.

Dot-product



Attention Model

다 상수니까 //

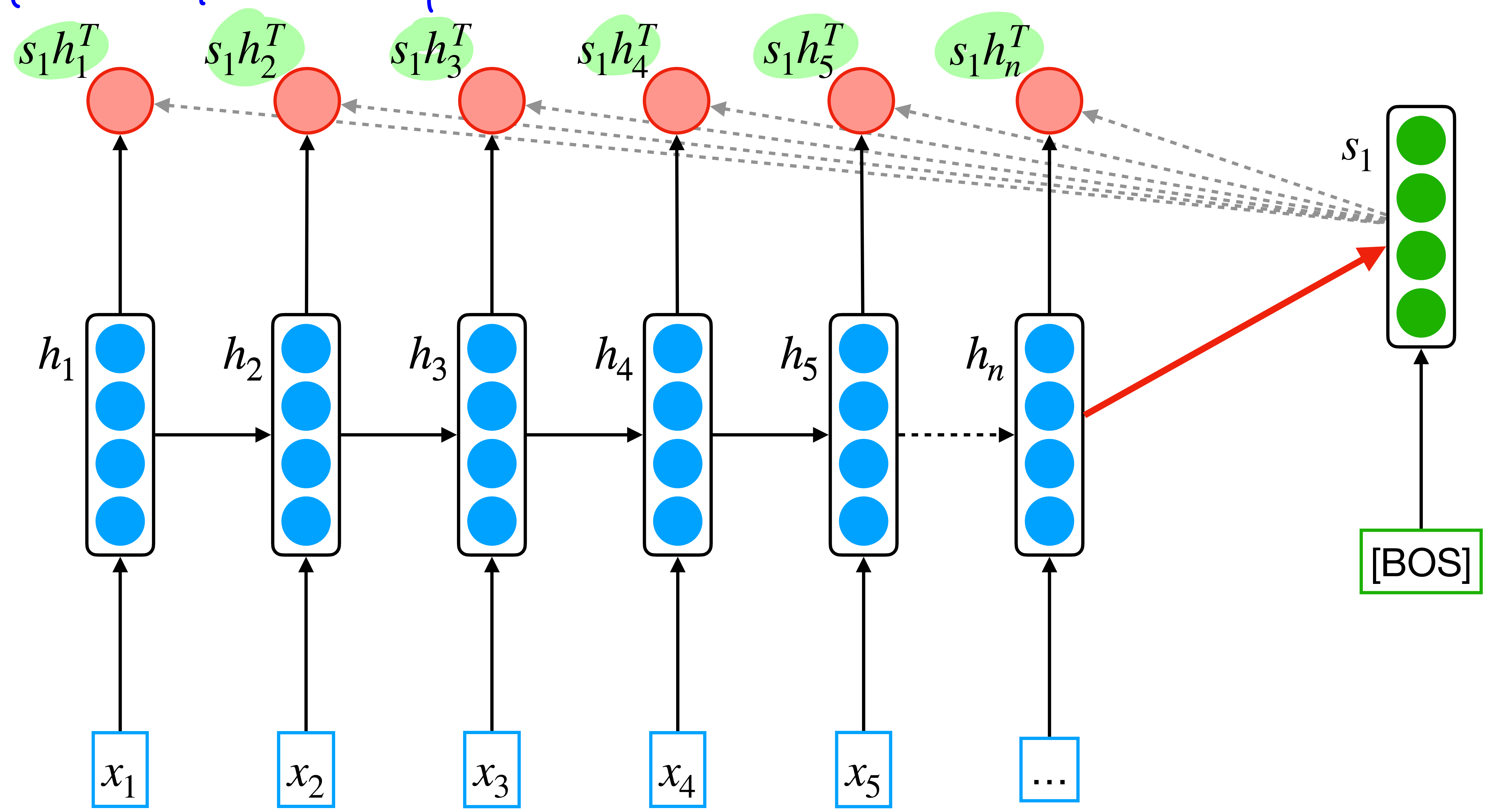
$\cancel{|s_1|} \cdot |h_1| \cdot \cos\theta_1$

Dot-product

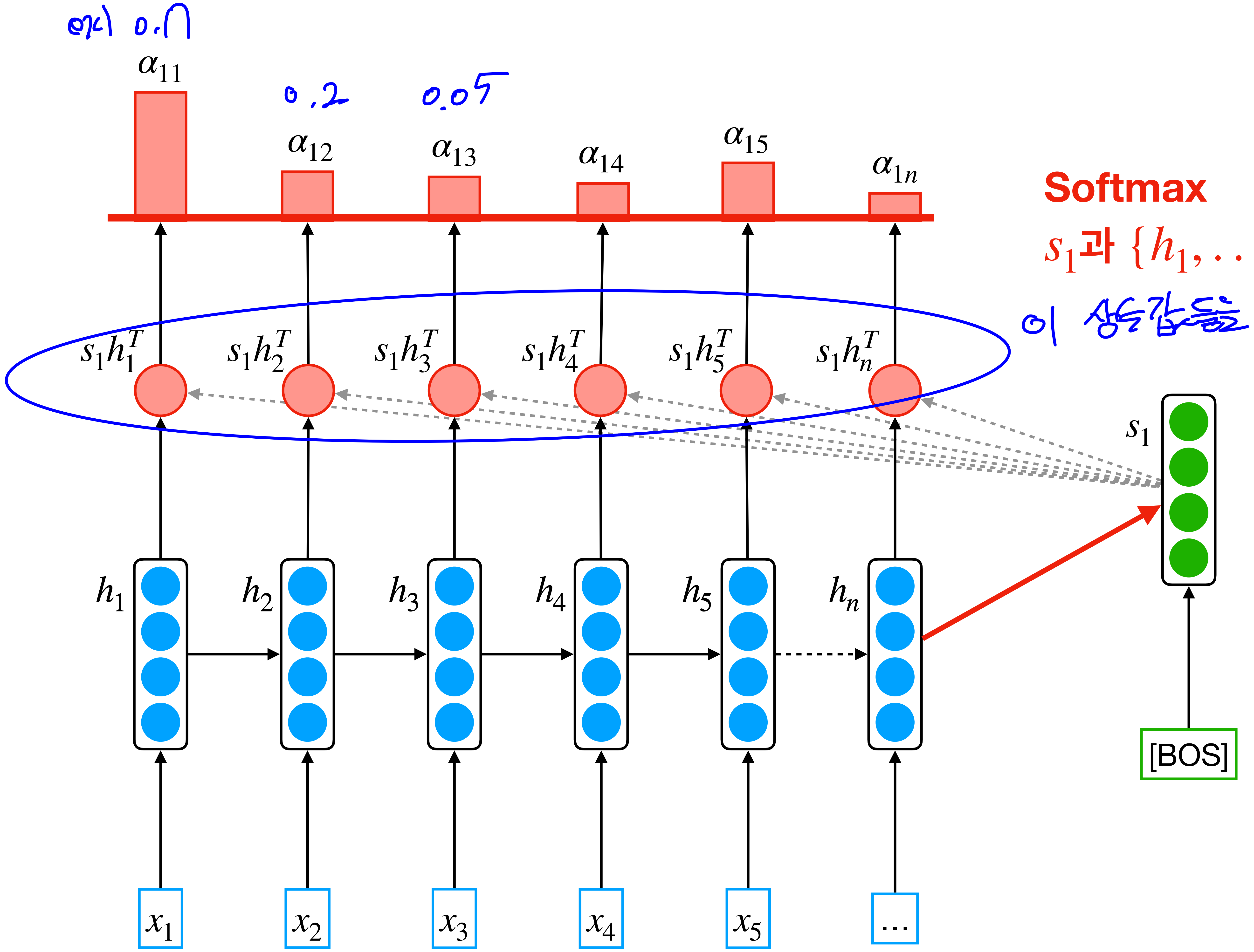
$\cancel{|s_1|} \cdot |h_1| \cos\theta_2$

$\cancel{|s_1|} \cdot |h_1| \cdot \cos\theta_3$

○ s_1 에 대해 각 h_i 들의 dot-Product를 통해 유사도를 측정.



Attention Model

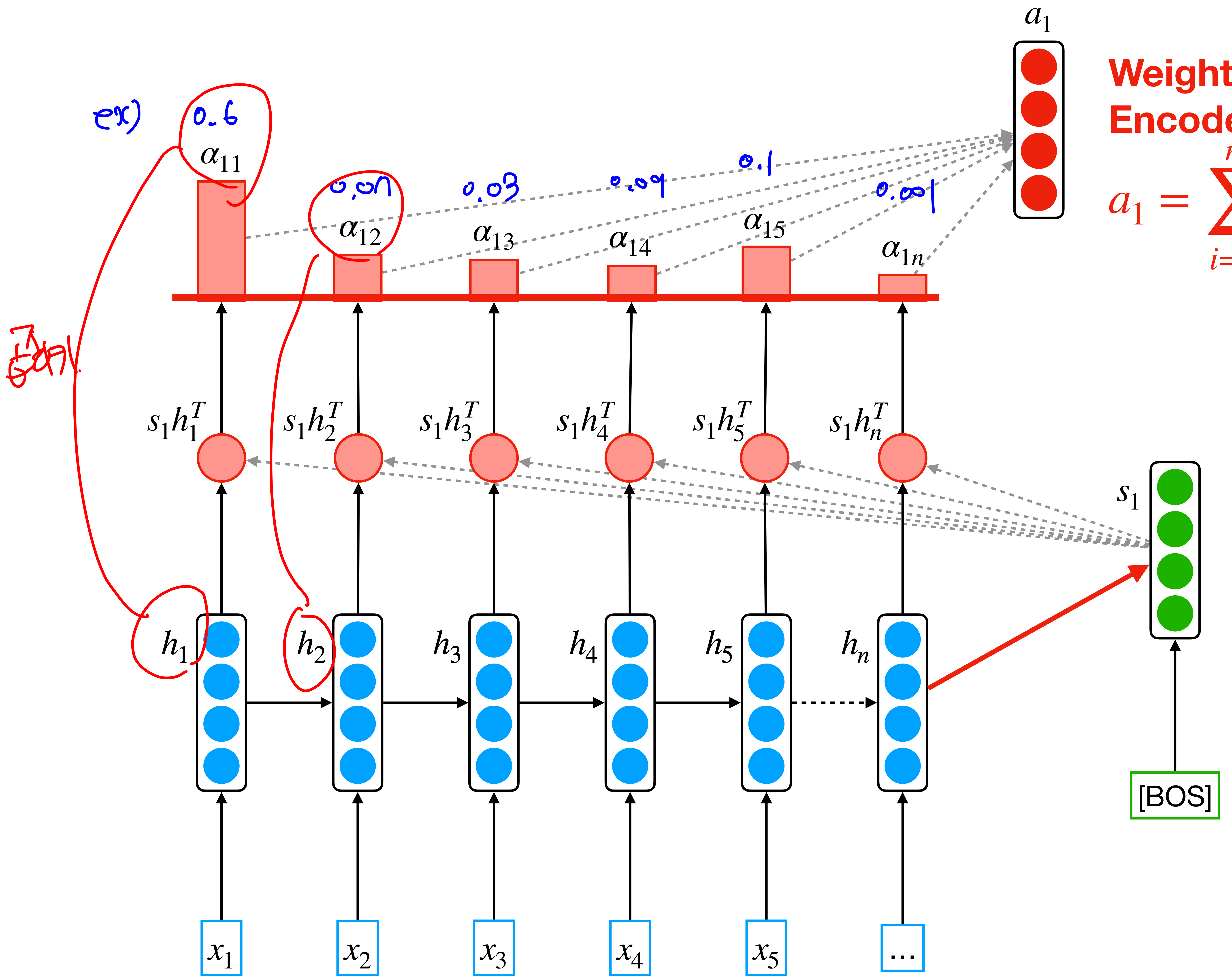


Softmax

s_1과 $\{h_1, .., h_n\}$의 유사도를 확률분포로 변환

이 상승값들을 softmax로 취하는거지!

Attention Model



Weighted Sum

Encoder hidden state의 weighted sum

$$a_1 = \sum_{i=1}^{n} \alpha_{1i} h_i$$

- 유사도를 통해 α를 계산
- 그걸 다시 h와 곱해서 합 ⇒ 다시 a_1

attention output

$0.6h_1 + 0.07h_2 + 0.03h_3 + \cdots 0.001 \cdot h_n$

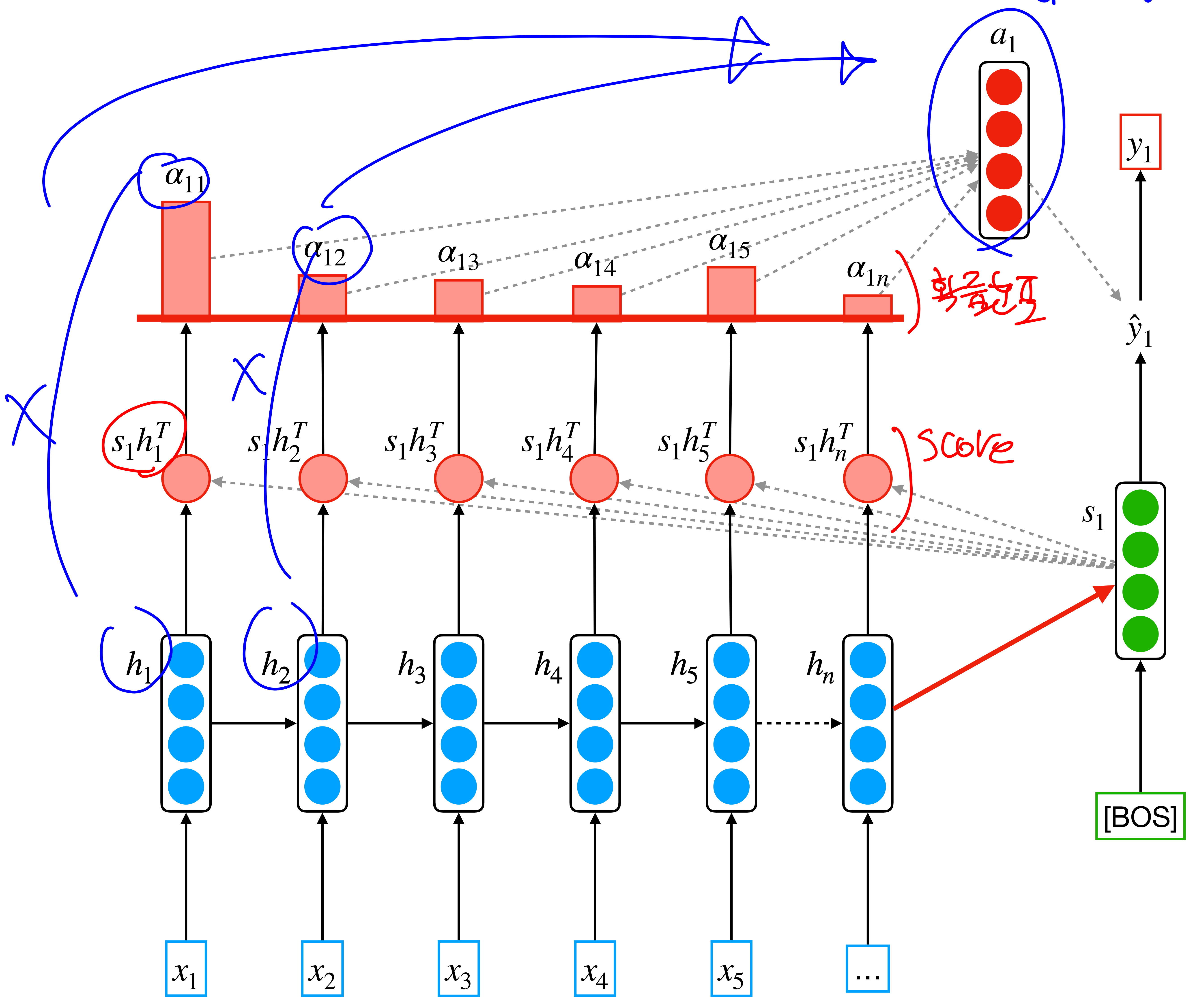
= h_1은 60% 영향.

= h_2는 7% 영향.

⇒ . . .

Attention Model

encoder-weight-sum ⇒ encoder의 정보만 있는거지
S1은 단순히 score를 나타낸거니까
(=유사도)



Concatenate attention output with decoder hidden state $[s_1; a_1]$ and compute output

○ Encoder의 정보만 있잖아!
그러니까 S_1랑 다시
concat해야지!

concat

s_1
a_1

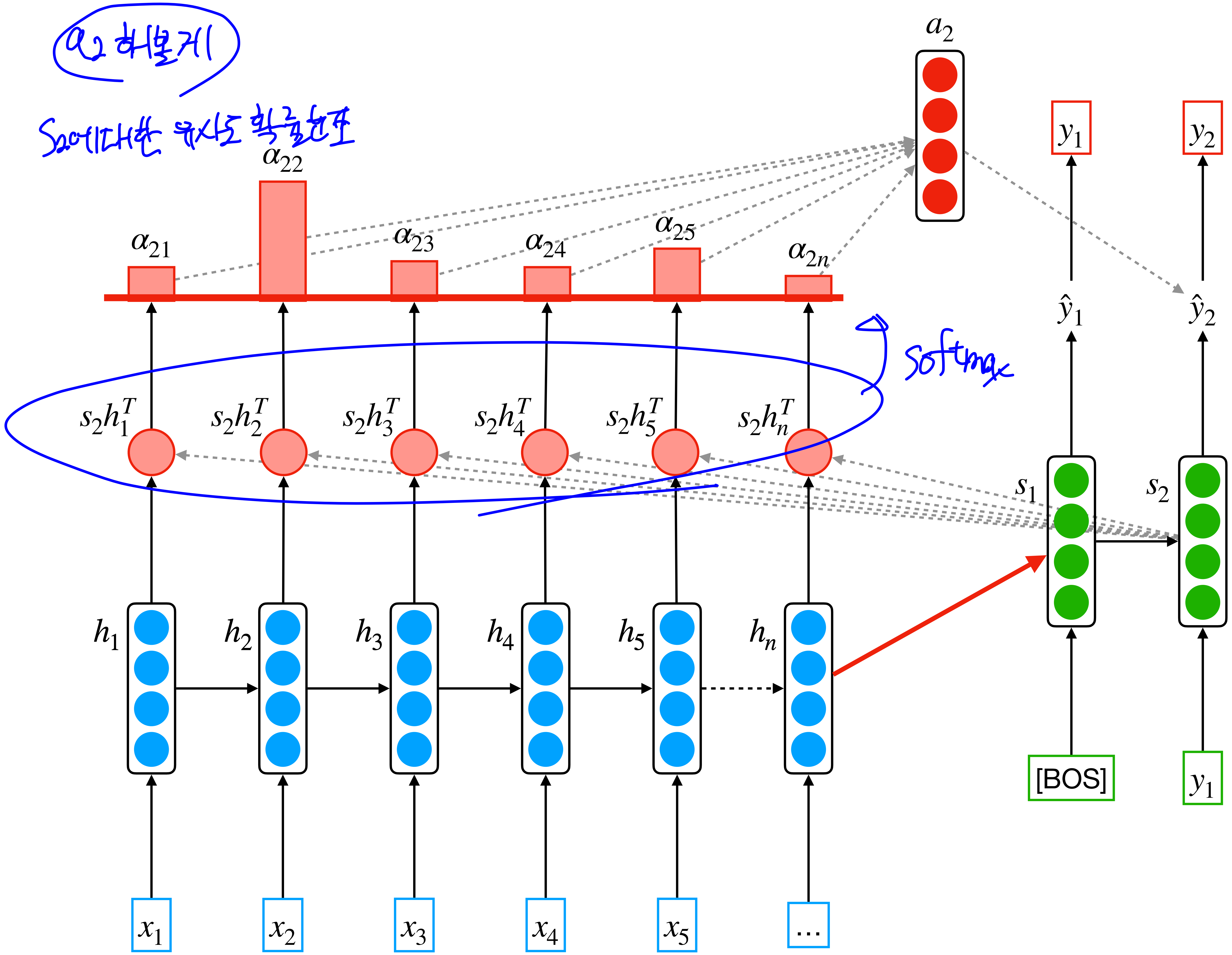
○ Score를 계산한거지
그걸 통해서 확률값하고
그걸 h_1 곱해서 각기 가중치를 준거잖아
그니까 Encoder 정보만있는거지!
그래서 decoder 정보를 합치기 위해서
concat을 해야함

Attention Model

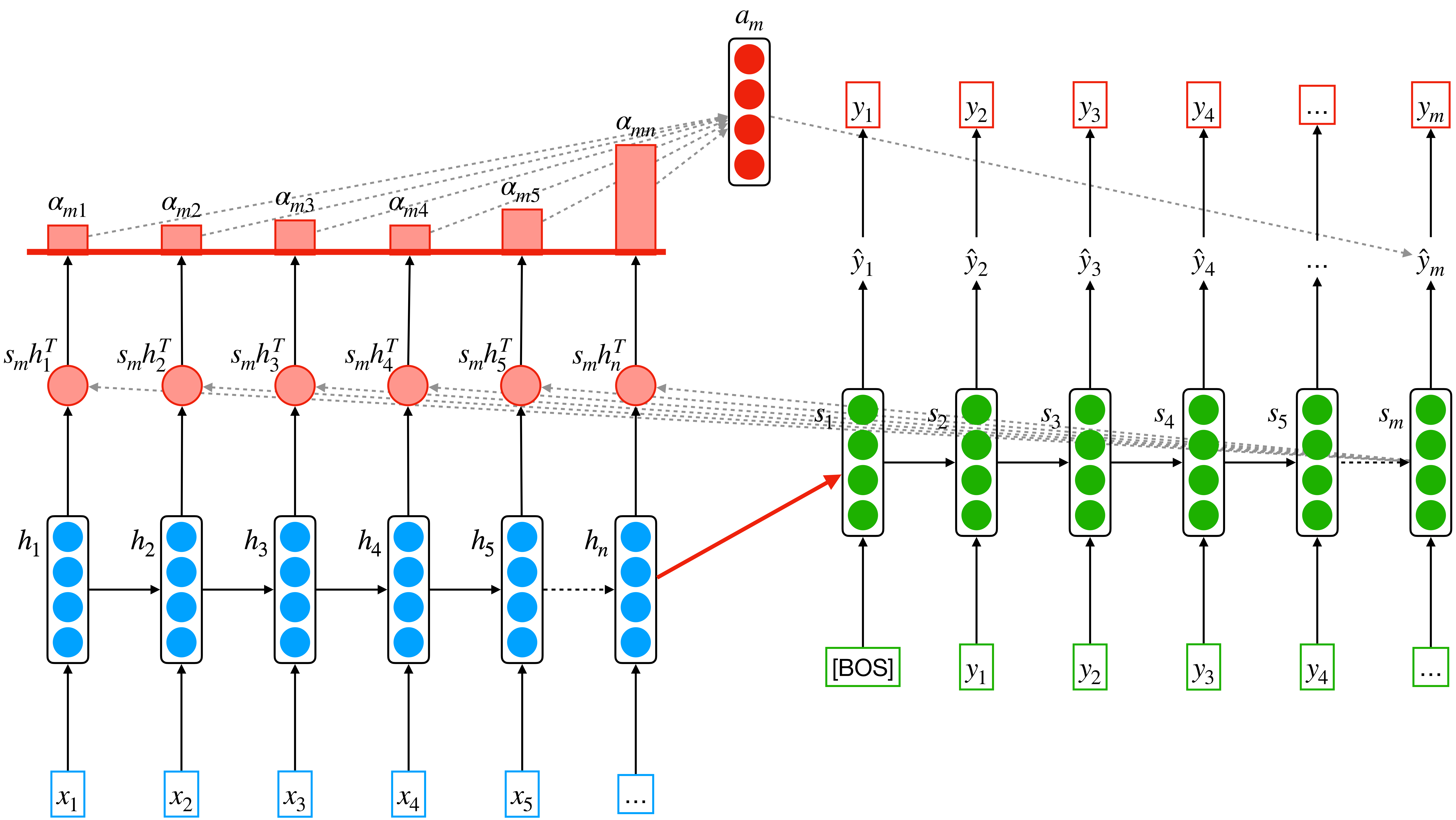
a2 해보기

S2에대한 유사도 확률분포

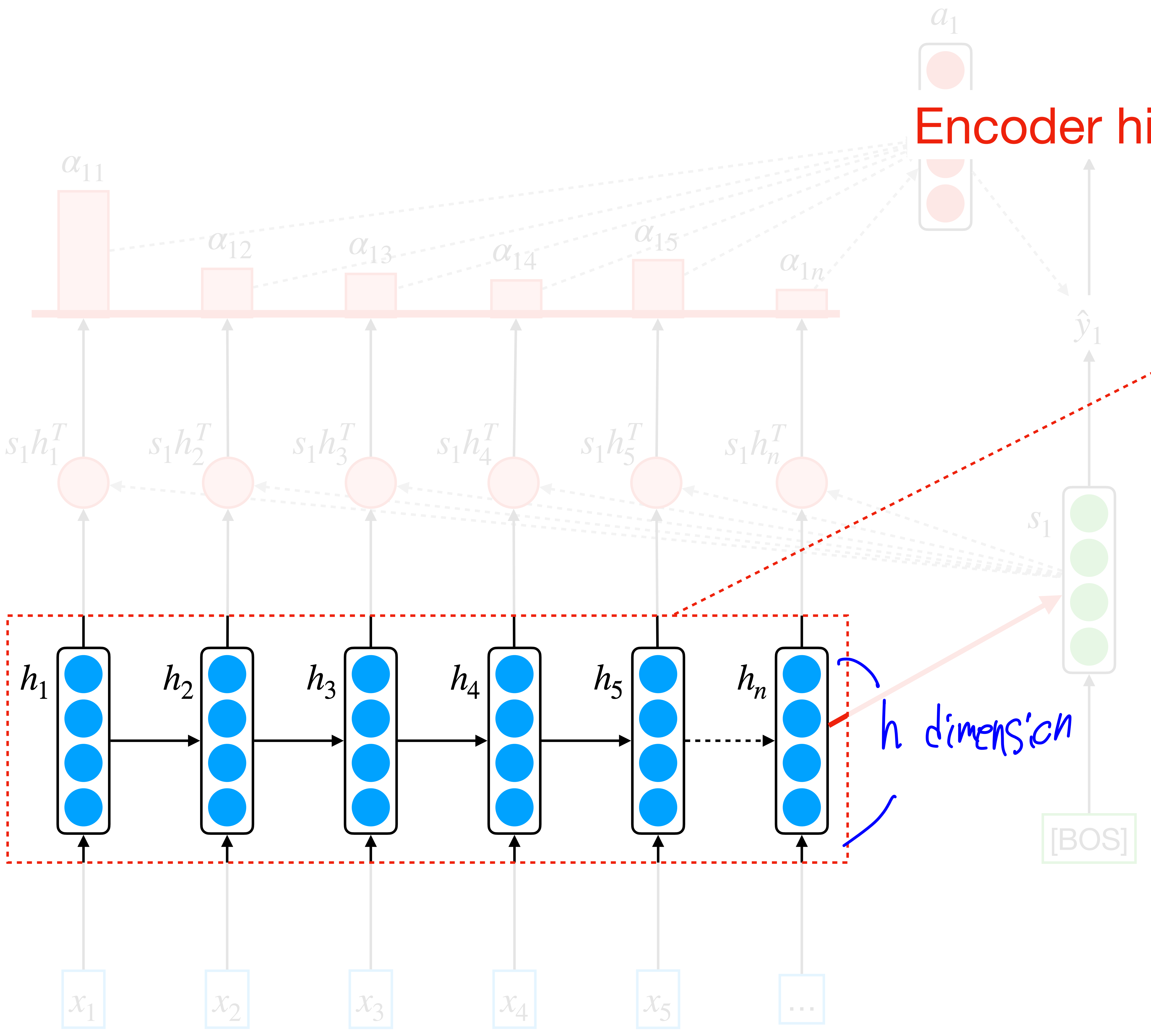
확률분포를
weight 로 sum



Attention Model



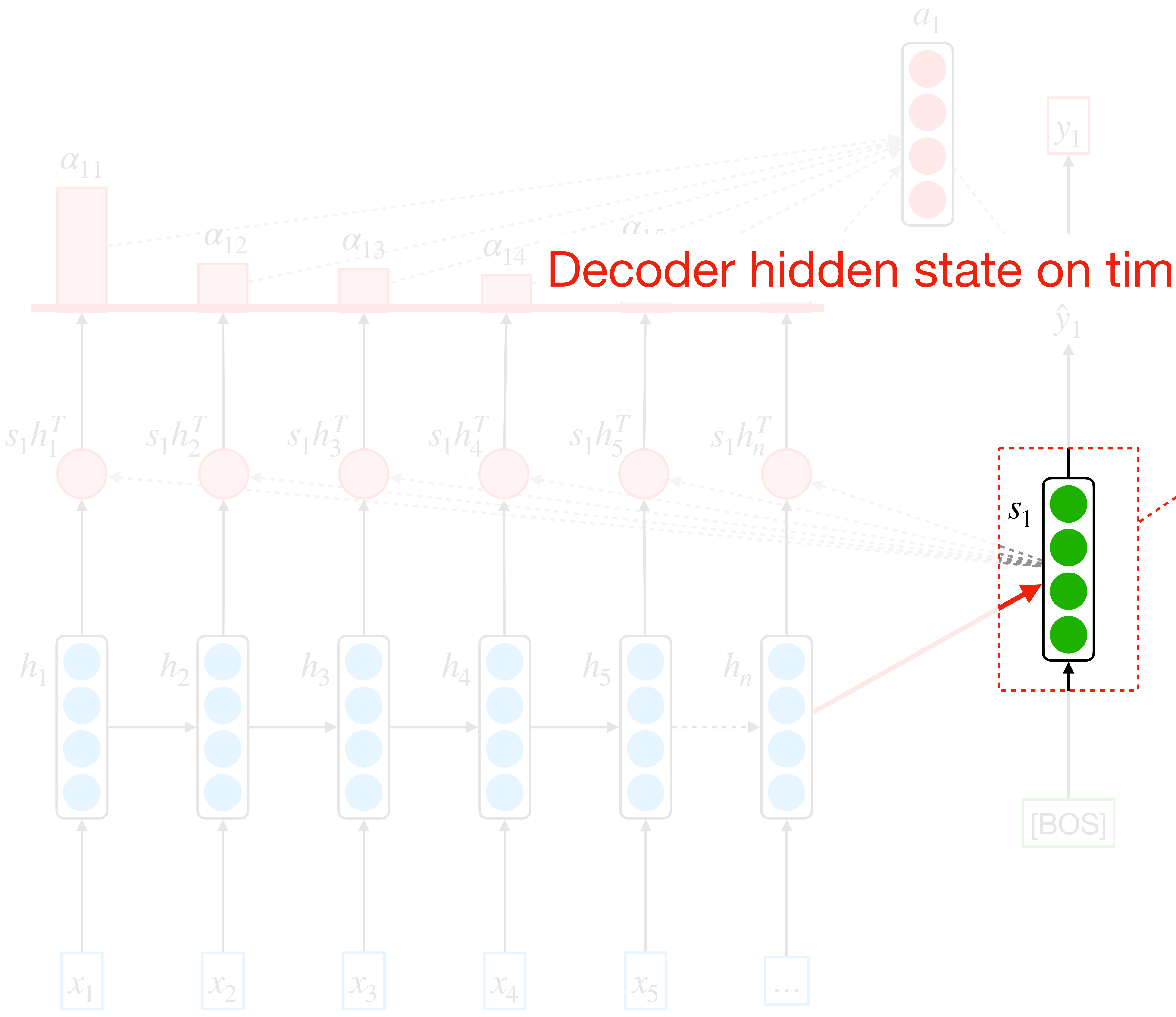
Attention Model (Equation)



Encoder hidden state: $h_1, \ldots, h_n \in \mathbb{R}^h$

h 차원

Attention Model (Equation)



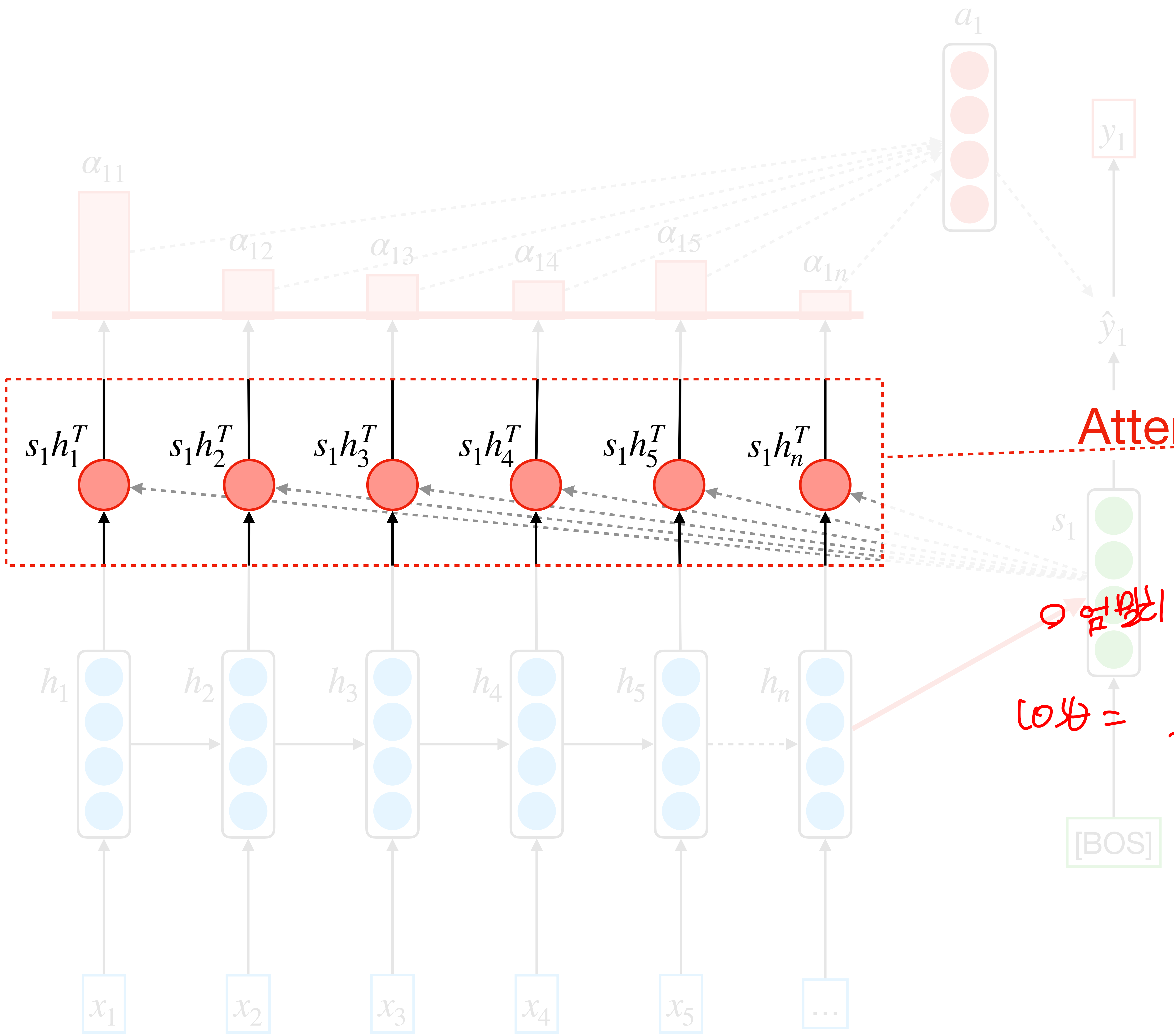
$$h_1, \ldots, h_n \in \mathbb{R}^h$$

Decoder hidden state on timestamp t: $s_t \in \mathbb{R}^h$

이것도 h의 차원인

그래야 dot product 되니까.

Attention Model (Equation)



$$h_1, \ldots, h_n \in \mathbb{R}^h$$

$$s_t \in \mathbb{R}^h$$

Attention score: $e^t = [s_t^T h_1, \ldots, s_t^T h_n] \in \mathbb{R}^n$

$(1, h) \circ (h, 1)$

$= (1, 1)$

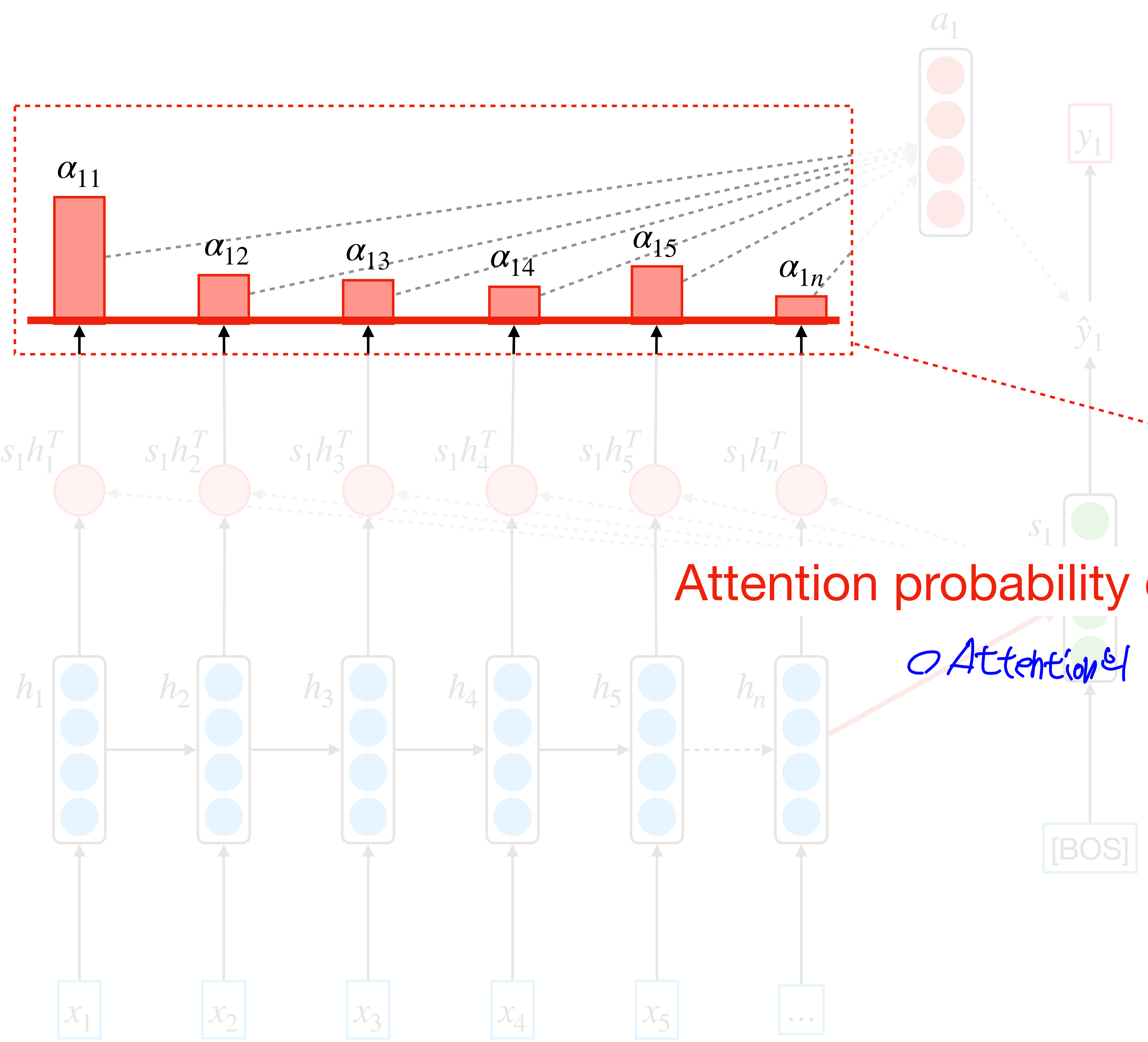
$=$ scalar

scalar가 n개!

○ 얼마나 닮았는지 유사한지

$\cos\theta = \frac{|a||b|}{|a\cdot b|}$ 이게 유사도

Attention Model (Equation)



$$h_1, \ldots, h_n \in \mathbb{R}^h$$

$$s_t \in \mathbb{R}^h$$

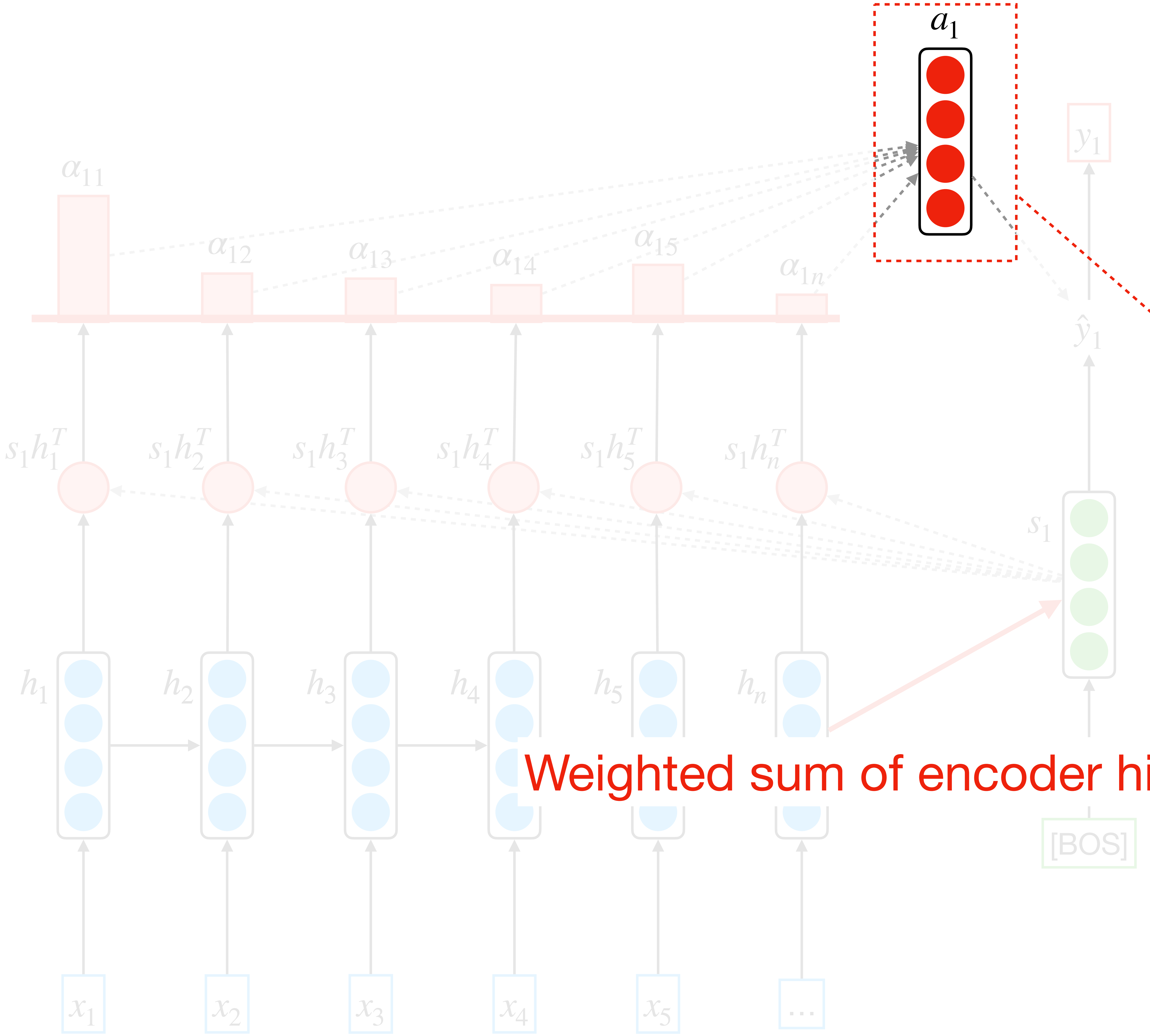
$$e^t = [s_t^T h_1, \ldots, s_t^T h_n] \in \mathbb{R}^n$$

Attention probability distribution: $\alpha^t = \underline{softmax(e^t)} \in \mathbb{R}^n$

○ Attention의 확률분포.

n 차원!

Attention Model (Equation)



$$h_1, \ldots, h_n \in \mathbb{R}^h$$

$$s_t \in \mathbb{R}^h$$

$$e^t = [s_t^T h_1, \ldots, s_t^T h_n] \in \mathbb{R}^n$$

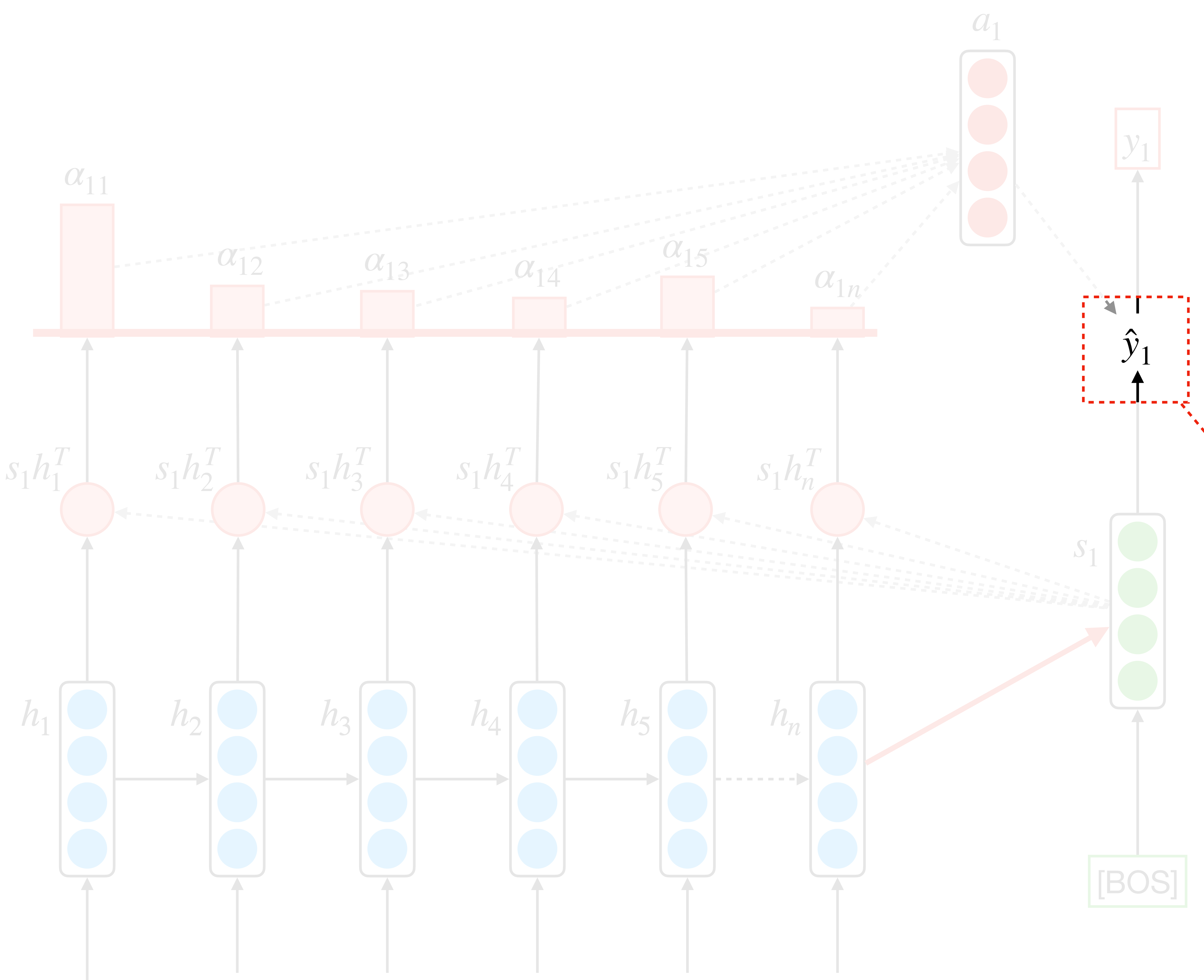
$$\alpha^t = softmax(e^t) \in \mathbb{R}^n$$

Weighted sum of encoder hidden state: $a_t = \sum_{i=1}^{n} \alpha_i^t h_i \in \mathbb{R}^h$

$(1,n) \circ (h,h)$

$= (1,h)$

Attention Model (Equation)



$$h_1, \ldots, h_n \in \mathbb{R}^h$$

$$s_t \in \mathbb{R}^h$$

$$e^t = [s_t^T h_1, \ldots, s_t^T h_n] \in \mathbb{R}^n$$

concat

$$\alpha^t = softmax(e^t) \in \mathbb{R}^n$$

$$a_t = \sum_{i=1}^{n} \alpha_i^t h_i \in \mathbb{R}^h$$

Concatenate attention output and decoder hidden state: $[a_t; s_t] \in \mathbb{R}^{2h}$

x_1 x_2 x_3 x_4 x_5 …

[illegible]

h가 2개니까 2h

Attention Model (Equation)

Encoder hidden state: $h_1, \ldots, h_n \in \mathbb{R}^h$

Decoder hidden state on timestamp t: $s_t \in \mathbb{R}^h$

Attention score: $e^t = [s_t^T h_1, \ldots, s_t^T h_n] \in \mathbb{R}^n$

Attention probability distribution: $\alpha^t = softmax(e^t) \in \mathbb{R}^n$

Weighted sum of encoder hidden state: $a_t = \sum_{i=1}^{n} \alpha_i^t h_i \in \mathbb{R}^h$

concat.

Concatenate attention output and decoder hidden state: $[a_t; s_t] \in \mathbb{R}^2 h$

Attention Model (Advantage)

- Attention을 이용해 NMT의 성능이 많이 좋아짐
  - Decoder가 source의 특정 부분에 집중하도록 한 것이 매우 효과적임 → 가중치를 준거임.
- 한계1 해결 Information bottleneck 문제를 해결 함
  - Decoder가 source에 직접 접근하도록 함
- 한계2 해결 Vanishing gradient 문제를 해결 함
  - 거리가 먼 source의 정보를 접근 할 수 있음
- Attention이 alignment를 학습함

feature map을 보고는 해석이 어려웠지만,
attention map을 보고 해석이 가능해짐

	Education	is	most	powerful	weapon
교육은					
가장					
강력한					
무기					
입니다					

Attention Model (Variants)

Encoder hidden

decoder hidden

Values → $h_1, \ldots, h_n \in \mathbb{R}^h$ $\quad$ $s_t \in \mathbb{R}^h$

o s·h의 dot-product를 했지.

Attention score → $s_t^T \cdot h_{i=n} = e^t \in \mathbb{R}^n$

attention score를 계산하는 다양한 방법이 있음

Attention probability → $\alpha^t = softmax(e^t) \in \mathbb{R}^n$

Attention output → $a_t = \sum_{i=1}^{n} \alpha_i^t h_i \in \mathbb{R}^h$

o h가 2번 쓰임

bi-lstm ⇒ unit = $\frac{d\_model}{2}$ 해서
out을 맞춰주는거고.

Attention Model (Variants)

$$e_t = [e_{t1}, \ldots, e_{tn}] \in \mathbb{R}^n$$

- Dot-product attention
  - $e_i^t = \underbrace{s_t^T h_i}_{\text{내적.}} \in \mathbb{R}$
- Multiplicative attention
  - $e_i^t = \boxed{s_t^T} W \boxed{h_i} \in \mathbb{R}$
  - where $W \in \mathbb{R}^{d_s \times d_h}$

weight shape이 다를때

곱해서 dh로 바꿔줌!

- Additive attention
  - $e_i^t = v^T \tanh(W_h h_i + W_s s_t) \in \mathbb{R}$
  - where $W_h \in \mathbb{R}^{d_v \times d_h}$, $W_s \in \mathbb{R}^{d_v \times d_s}$, $v \in \mathbb{R}^{d_v}$

Attention Tutorial

Attention Tutorial (inputs)

h.m n.h

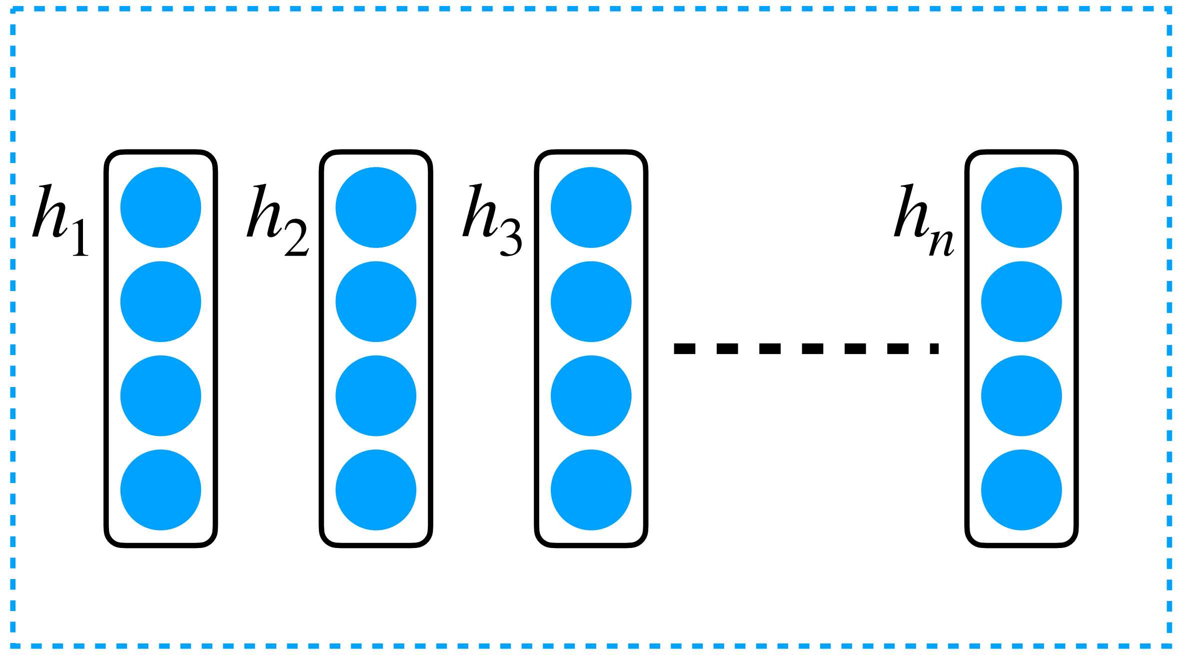
hxn h.m

(nxh) x (h.m)

=(n.m)

n

h

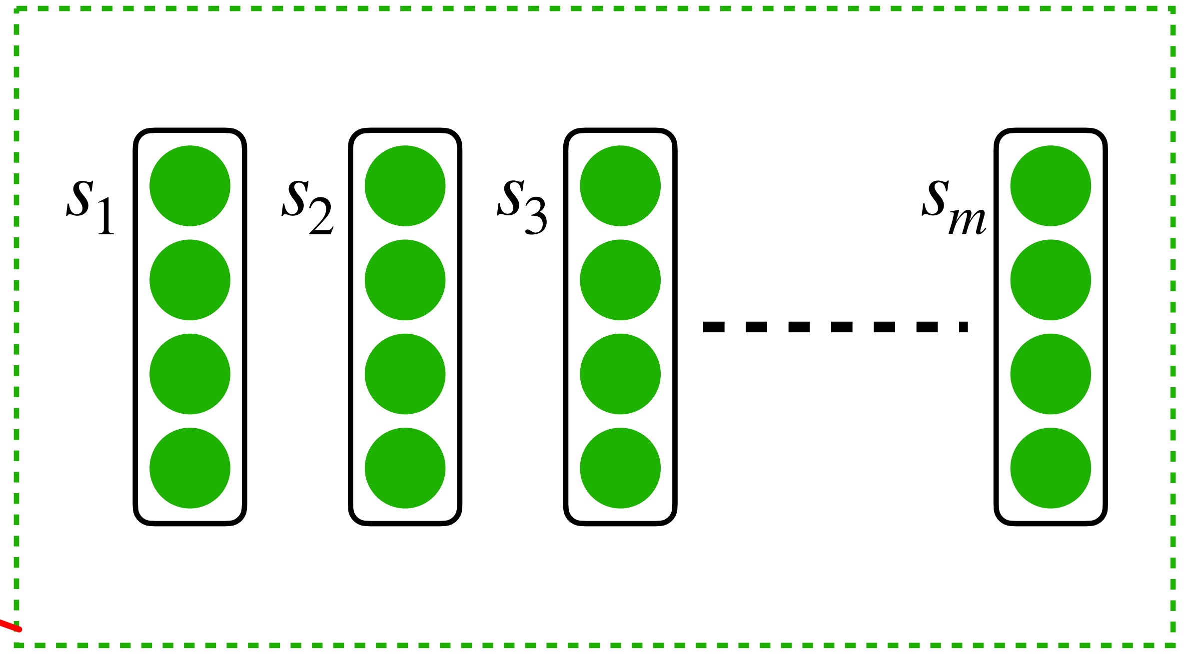


Encoder hidden state: $h \in \mathbb{R}^{h \times n}$

(hxn)

m

h



Decoder hidden state: $s \in \mathbb{R}^{h \times m}$

(hxm)

(hxm)x(nxh)

○ h^T = enc_hidden
○ s^T = dec_hidden

Attention Tutorial (score)

행렬곱하면 score가 한번에 계산되겠지!

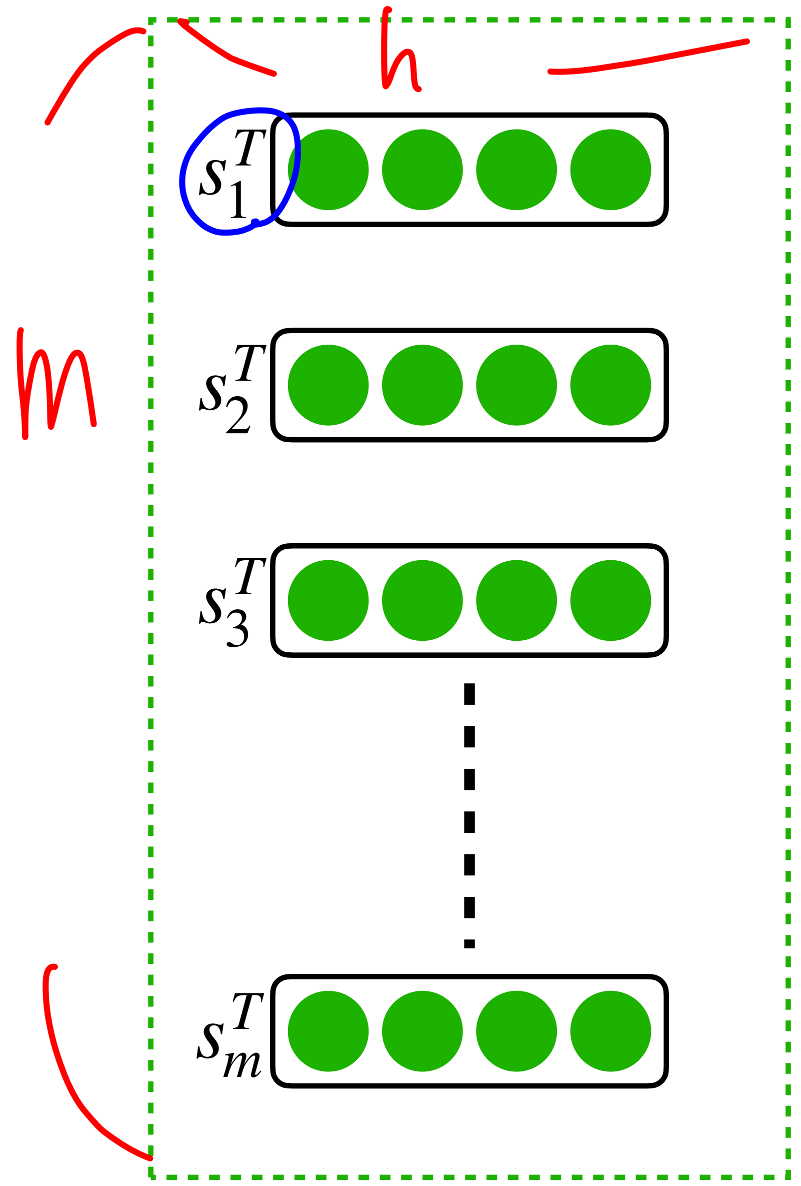
$$e = s^T h \in \mathbb{R}^{m \times n}$$

encoder_hidden T

$(m \times h) \times (h \times n)$
$= (m \times n)$

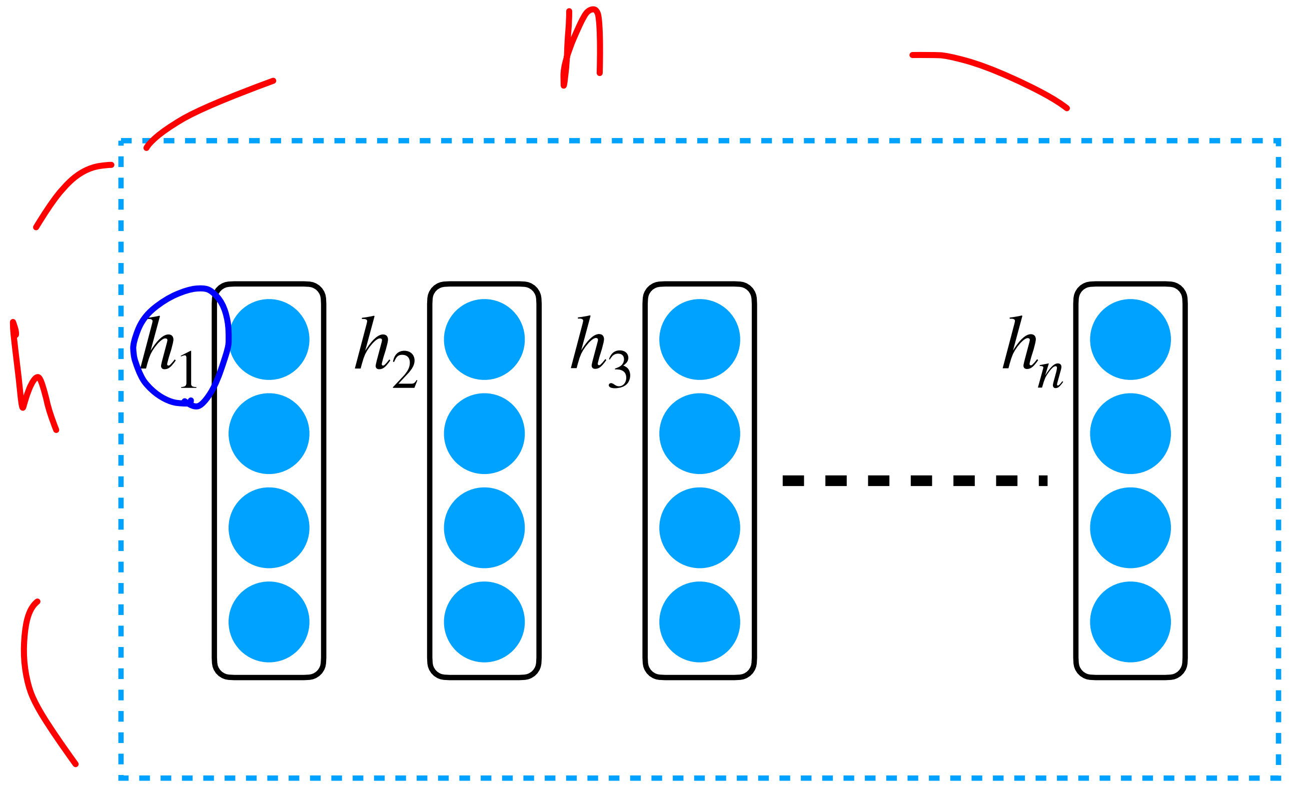
decorder_hidden

s^T



Decoder hidden state: $s^T \in \mathbb{R}^{m \times h}$

×



Encoder hidden state: $h \in \mathbb{R}^{h \times n}$

=

의 score값

$s_1 \cdot h_n$의 score

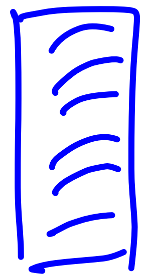
	h_1	h_2	h_3	...	h_n
s_1^T	$s_1^T h_1$	$s_1^T h_2$	$s_1^T h_3$	...	$s_1^T h_n$
s_2^T	$s_2^T h_1$	$s_2^T h_2$	$s_2^T h_3$	...	$s_2^T h_n$
s_3^T	$s_3^T h_1$	$s_3^T h_2$	$s_3^T h_3$	...	$s_3^T h_n$
...	...	...	...	...	...
s_m^T	$s_m^T h_1$	$s_m^T h_2$	$s_m^T h_3$	...	$s_m^T h_n$

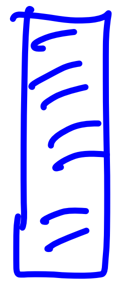
Attention score: $e \in \mathbb{R}^{m \times n}$

$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} e & f \\ g & h \end{pmatrix} = \begin{pmatrix} ae+bg & \cdot \\ \cdot & \cdot \end{pmatrix}$

$e_j^i = s_i^T h_j \in \mathbb{R}$

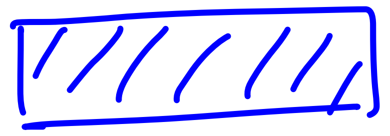
Attention Tutorial (score)





그림에서 (논문에서) (행벡터)

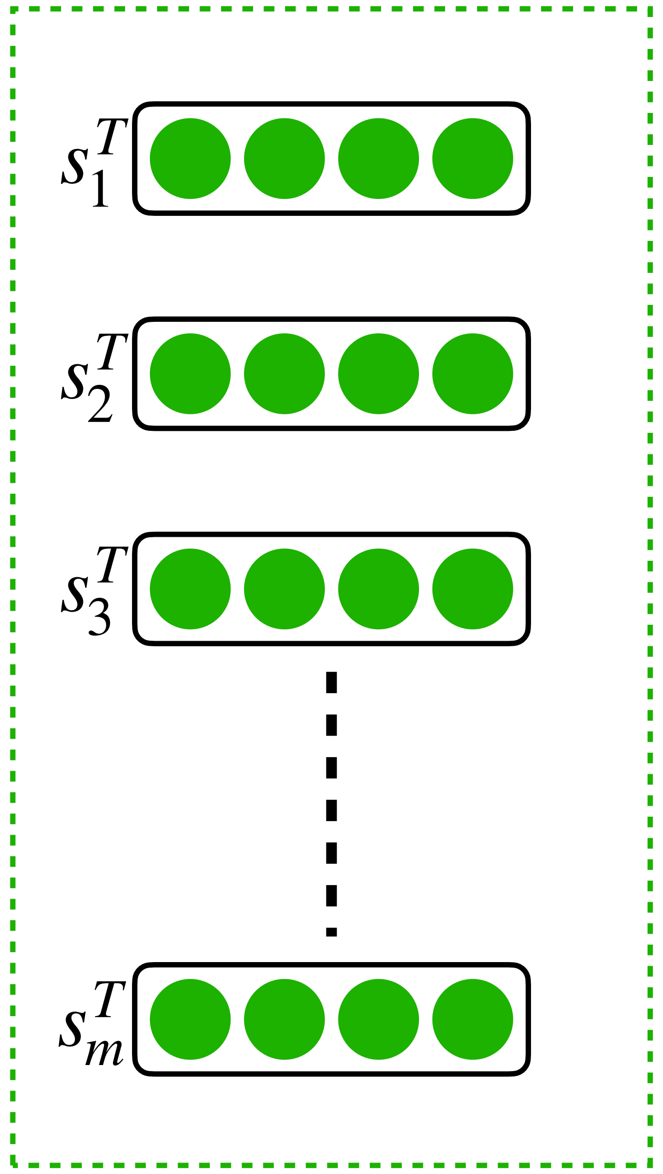
수식.



실제 코드 (열벡터)

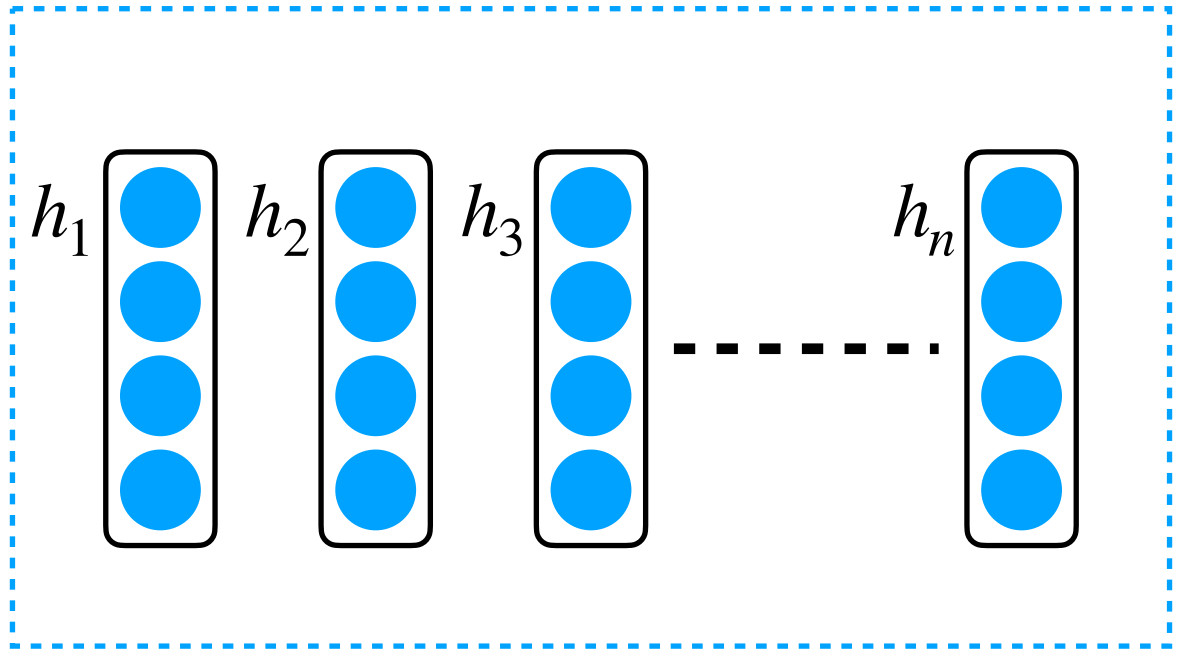
numpy . tf.

$$e = s^T h \in \mathbb{R}^{m \times n}$$



Decoder hidden state: $s^T \in \mathbb{R}^{m \times h}$

$\times$



Encoder hidden state: $h \in \mathbb{R}^{h \times n}$

$=$

	h_1	h_2	h_3	$\cdots$	h_n
s_1^T	$s_1^T h_1$	$s_1^T h_2$	$s_1^T h_3$	...	$s_1^T h_n$
s_2^T	$s_2^T h_1$	$s_2^T h_2$	$s_2^T h_3$	...	$s_2^T h_n$
s_3^T	$s_3^T h_1$	$s_3^T h_2$	$s_3^T h_3$	...	$s_3^T h_n$
$\cdots$	...	...	...	...	...
s_m^T	$s_m^T h_1$	$s_m^T h_2$	$s_m^T h_3$	...	$s_m^T h_n$

Attention score: $e \in \mathbb{R}^{m \times n}$

$$e_j^i = s_i^T h_j \in \mathbb{R}$$

Attention Tutorial (prob)

$$\alpha = softmax(e) \in \mathbb{R}^{m \times n}$$

$$softmax\left(\begin{array}{c|ccccc} & h_1 & h_2 & h_3 & \dots & h_n \\ \hline s_1^T & s_1^T h_1 & s_1^T h_2 & s_1^T h_3 & \dots & s_1^T h_n \\ s_2^T & s_2^T h_1 & s_2^T h_2 & s_2^T h_3 & \dots & s_2^T h_n \\ s_3^T & s_3^T h_1 & s_3^T h_2 & s_3^T h_3 & \dots & s_3^T h_n \\ \dots & \dots & \dots & \dots & \dots & \dots \\ s_m^T & s_m^T h_1 & s_m^T h_2 & s_m^T h_3 & \dots & s_m^T h_n \end{array}\right) = \begin{array}{c|ccccc} & h_1 & h_2 & h_3 & \dots & h_n \\ \hline s_1^T & \alpha_1^1 & \alpha_2^1 & \alpha_3^1 & \dots & \alpha_n^1 \\ s_2^T & \alpha_1^2 & \alpha_2^2 & \alpha_3^2 & \dots & \alpha_n^2 \\ s_3^T & \alpha_1^3 & \alpha_2^3 & \alpha_3^3 & \dots & \alpha_n^3 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ s_m^T & \alpha_1^m & \alpha_2^m & \alpha_3^m & \dots & \alpha_n^m \end{array}$$

s_1에대한 확률 분포 $\longrightarrow$ $\sum = 1$

s_2에 대한 확률분포 $\longrightarrow$ $\sum = 1$

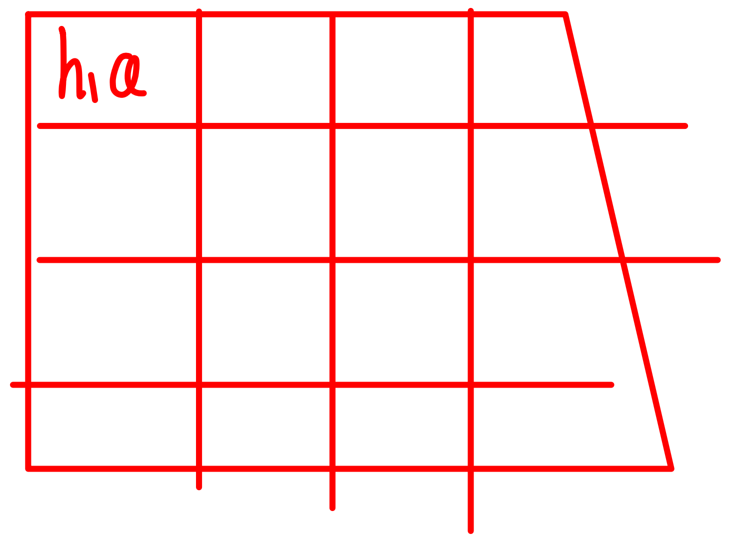
Attention score: $e \in \mathbb{R}^{m \times n}$

행 단위 softmax

Attention prob: $\alpha \in \mathbb{R}^{m \times n}$

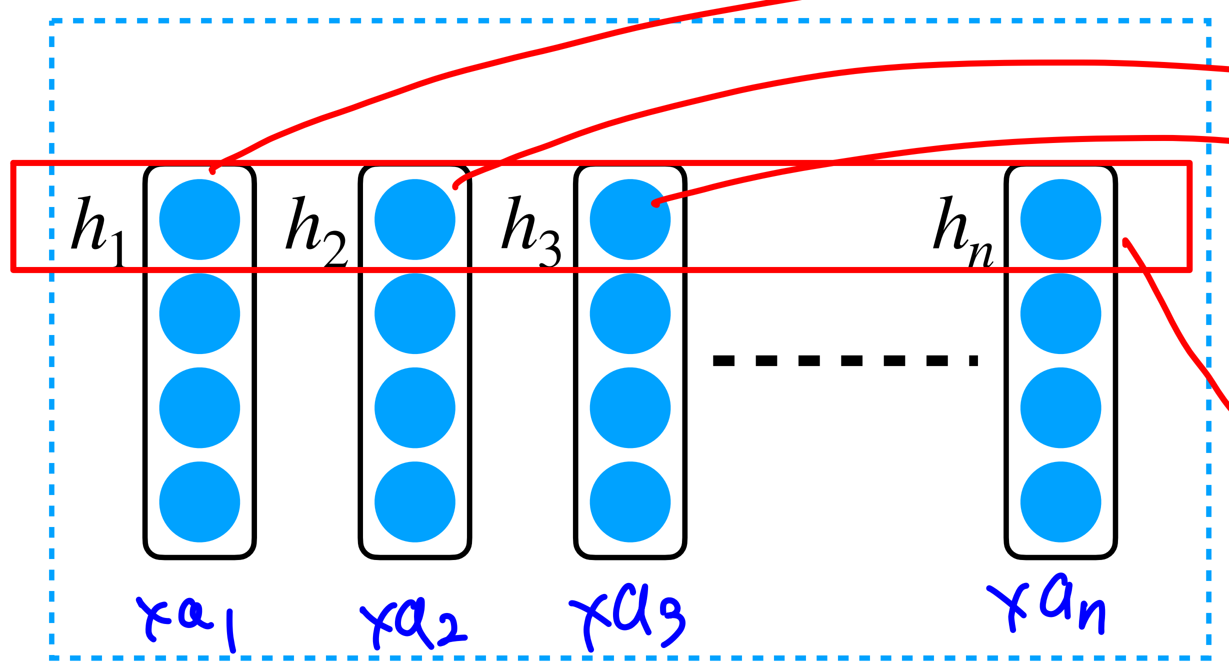
$(A \cdot B)^T = B^T \cdot A^T$

Attention Tutorial (output)



$$a = h\alpha^T \in \mathbb{R}^{h\times m}$$

$a^T = \alpha \cdot h^T$



Encoder hidden state: $h \in \mathbb{R}^{h\times n}$

scalar, a_1^i 0.6 // $[h_1]$ + a_2^i 0.1 // $[h_2]$ + a_3^i 0.01 // $[h_3]$ + … + a_n^i 0.01 $[h_n]$

$(h\times n)$

$= (h\times m)$

$\times$

	s_1	s_2	s_3	…	s_m
h_1^T	α_1^1	α_1^2	α_1^3	…	α_1^m
h_2^T	α_2^1	α_2^2	α_2^3	…	α_2^m
h_3^T	α_3^1	α_3^2	α_3^3	…	α_3^m
…	…	…	…	…	…
h_n^T	α_n^1	α_n^2	α_n^3	…	α_n^m

0.6 0.1 0.01 0.01

Attention prob: $\alpha^T \in \mathbb{R}^{n\times m}$

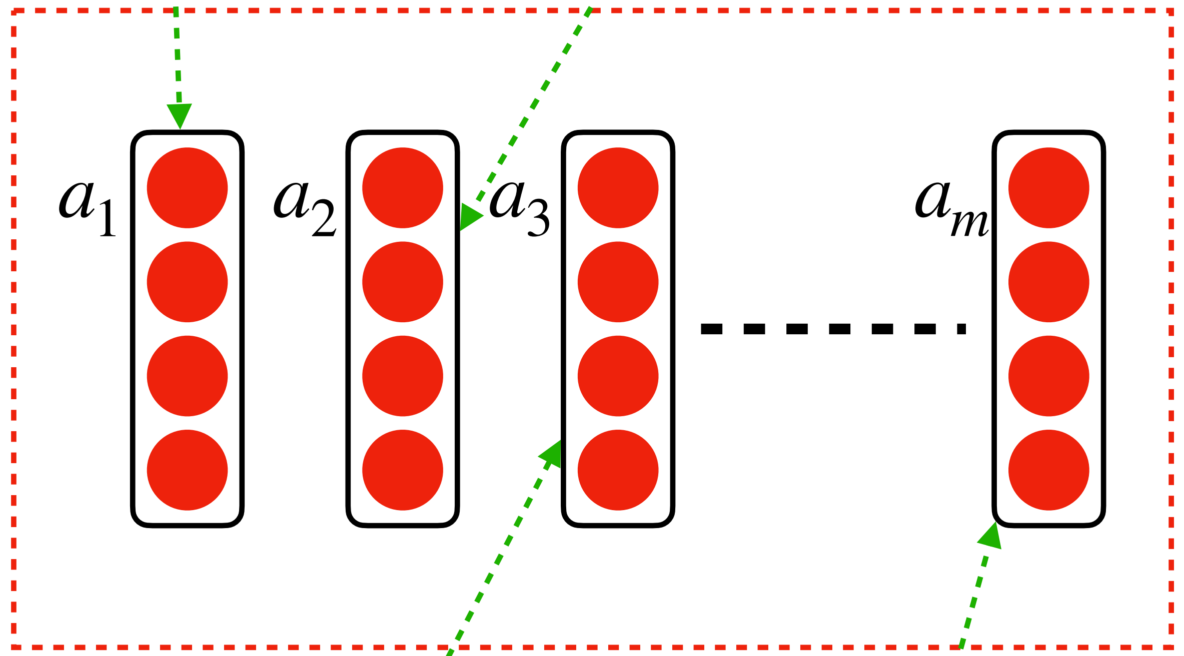
⊜ Transpose 된 상태임 //

$(n\times m)$

$=$

$\alpha_1^1 h_1 + \alpha_2^1 h_2 + \alpha_3^1 h_3 + \ldots + \alpha_n^1 h_n$

$\alpha_1^2 h_1 + \alpha_2^2 h_2 + \alpha_3^2 h_3 + \ldots + \alpha_n^2 h_n$



Attention output: $a \in \mathbb{R}^{h\times m}$

$\alpha_1^m h_1 + \alpha_2^m h_2 + \alpha_3^m h_3 + \ldots + \alpha_n^m h_n$

$\alpha_1^3 h_1 + \alpha_2^3 h_2 + \alpha_3^3 h_3 + \ldots + \alpha_n^3 h_n$

o attention은 weight를 늘린게 아니라. 계산량을 늘린거지 //

o 지표는 보통 bleu를.. count 기반

extraction // + square //

생성모델 // +

감사합니다.