# Attention & Transformer

Minjong Lee

POSTECH CSE

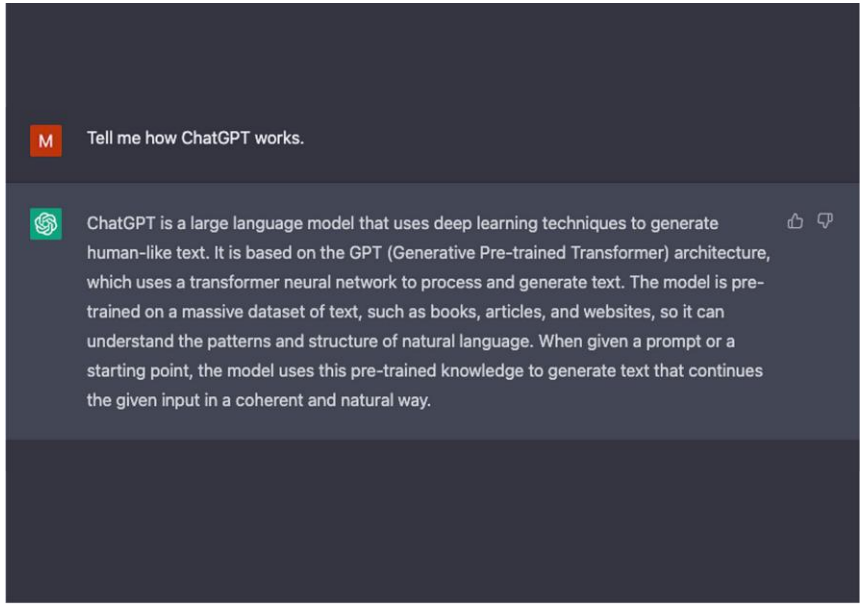
minjong.lee@postech.ac.kr

# 목차

- Attention
- Transformer

# Attention? Transformer?

- 최근 화제인 ChatGPT와 같은 Transformer-based model이 좋은 성능을 보여주고 있다
  - Transformer의 구조는 attention으로 이루어져 있다

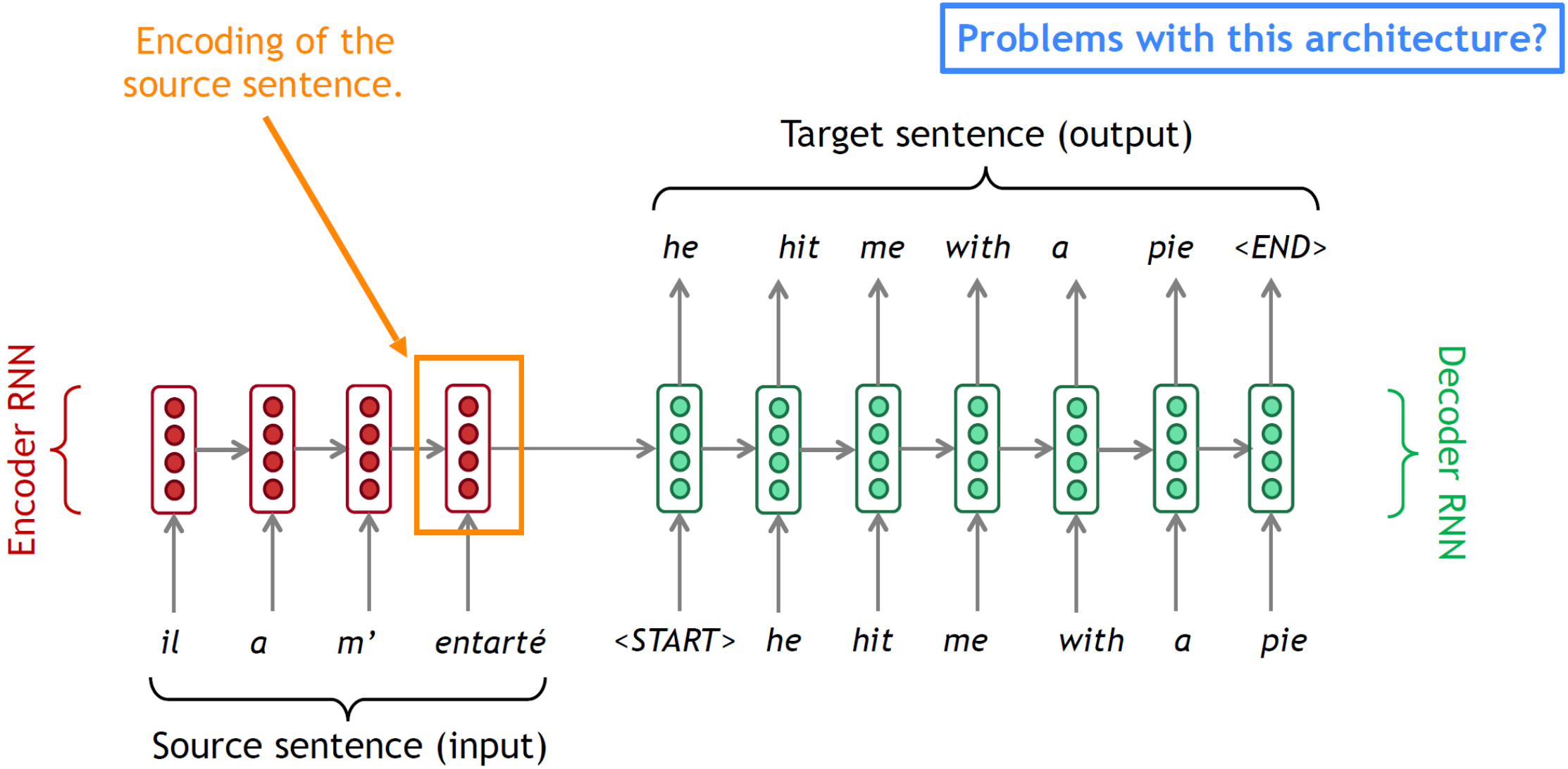


# Attention

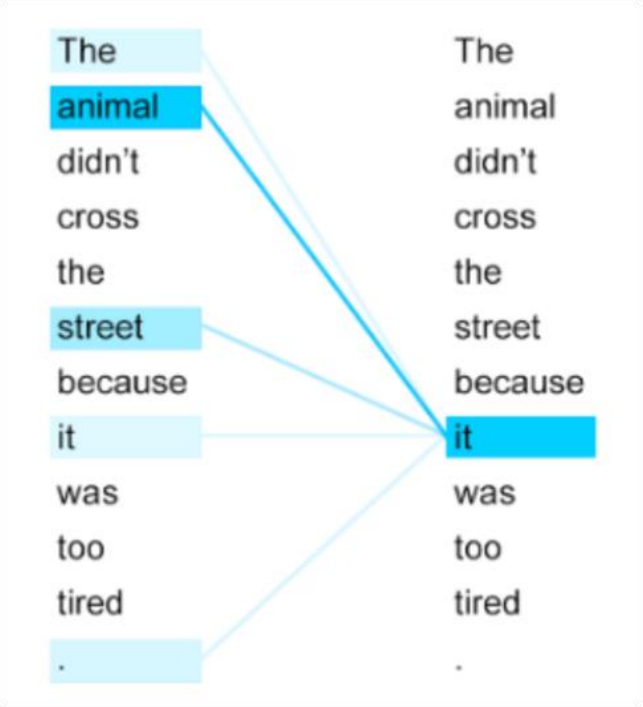
- 데이터 측면에서 봤을 때는 여러 데이터들 중 어떤 데이터에 집중해야 할 지 사용된다!

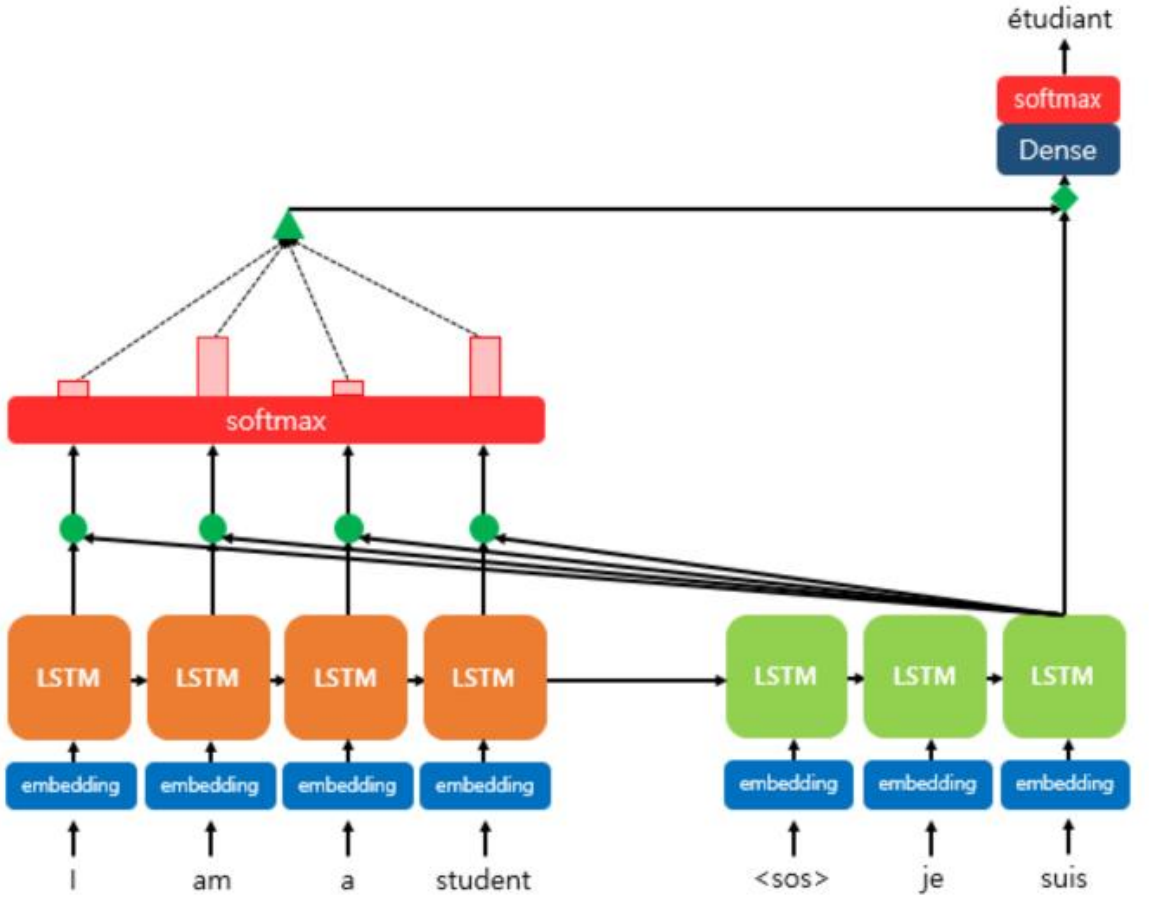
➢ 특히 자연어 데이터에서 많이 사용되며, 요즘은 비전이나 딥러닝이 활용되는 많은 분야에 적용됨. 데이터의 길이 등에 영향을 받지 않고 굉장히 Flexible한 구조를 디자인 할 수 있는 구조!

# Sequence-to-Sequence: the Bottleneck Problem



# Attention

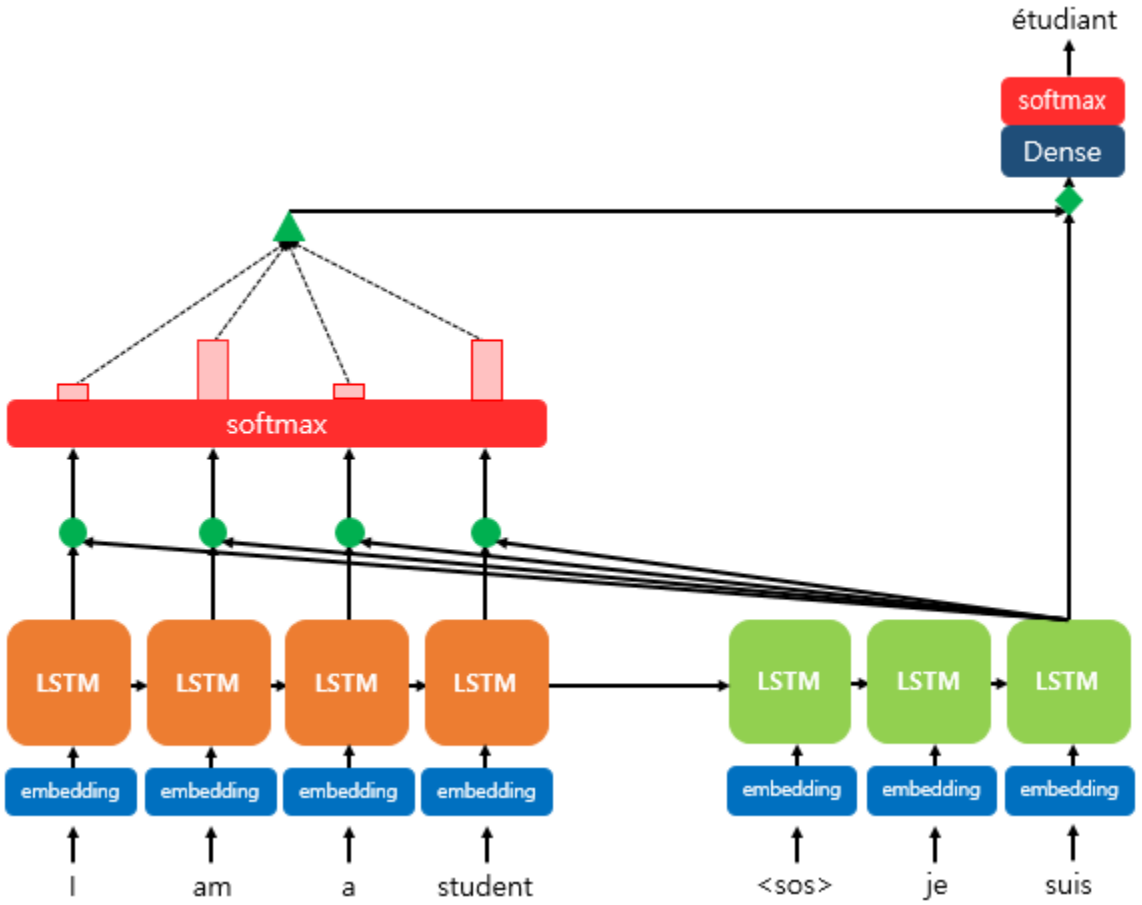

The
animal
didn't
cross
the
street
because
it
was
too
tired
.
The
animal
didn't
cross
the
street
because
it
was
too
tired
.


étudiant
softmax
Dense
softmax
LSTM
LSTM
LSTM
LSTM
LSTM
LSTM
LSTM
embedding
embedding
embedding
embedding
embedding
embedding
embedding
I
am
a
student
<sos>
je
suis

https://wikidocs.net/22893

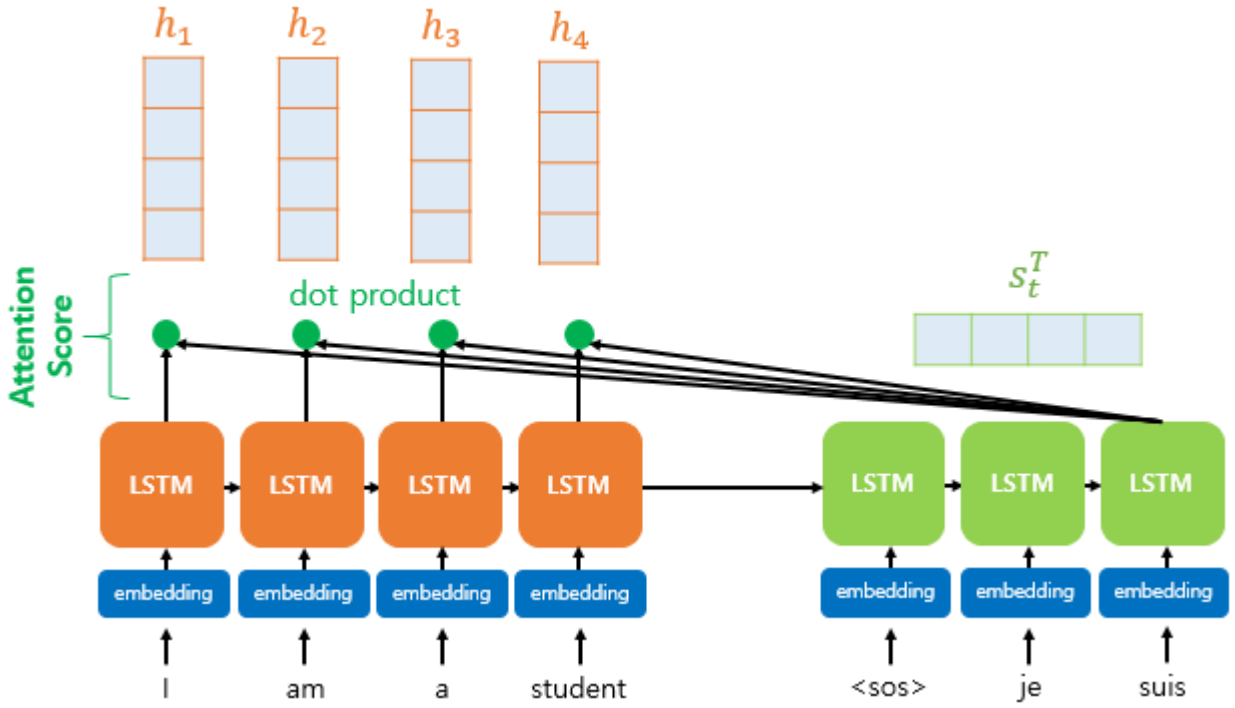
# Dot-Product Attention

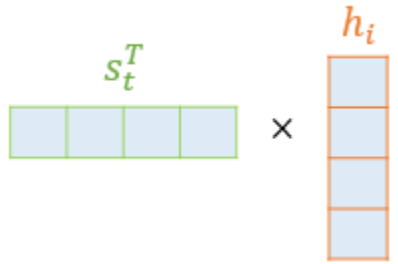
- Seq2Seq에 사용되는 attention 기법 중 Dot-Product Attention에 대해 알아보자



# Dot-Product Attention

- 어텐션 스코어(Attention Score)를 구한다



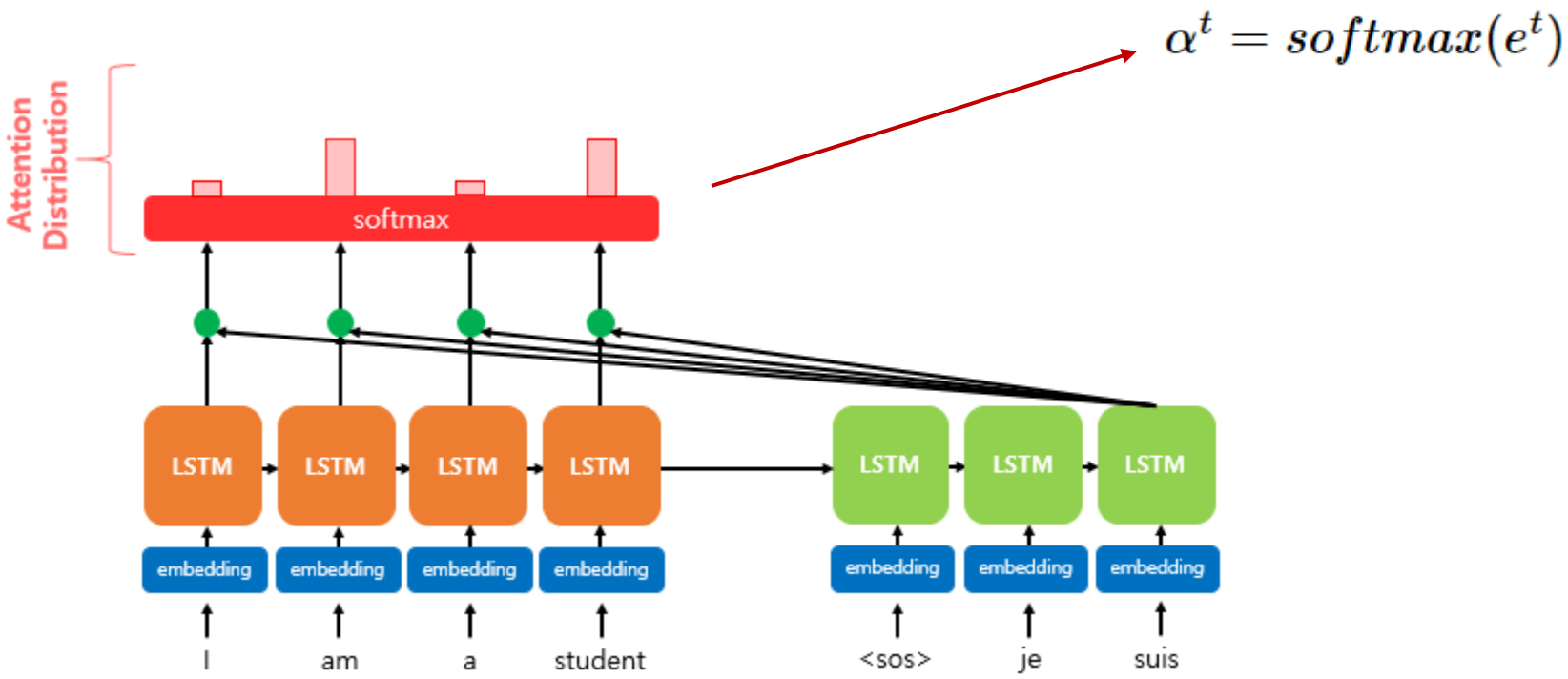




$$score(s_t,\ h_i) = s_t^T h_i$$

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

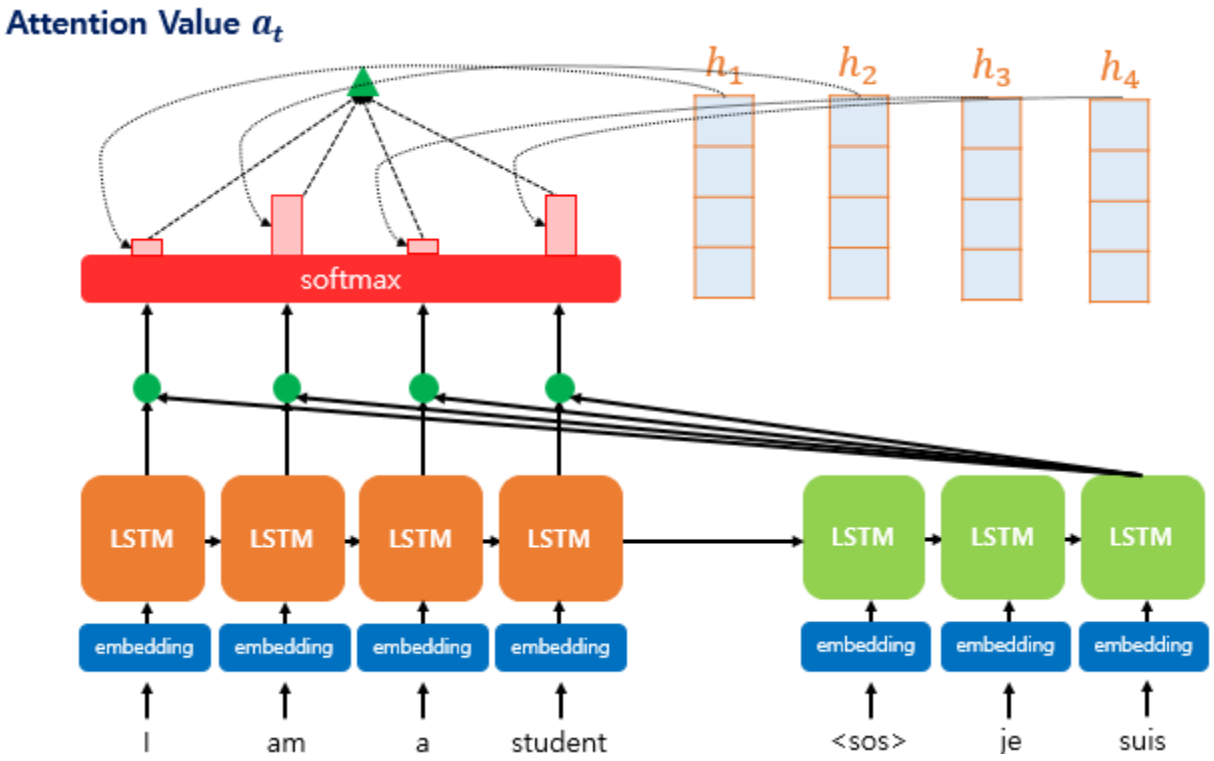
# Dot-Product Attention

- Attention distribution을 계산한다
  - softmax 함수를 사용한다



# Dot-Product Attention

- Attention value $a_t = \sum_{i=1}^{N} \alpha_i^t h_i$ 를 계산한다

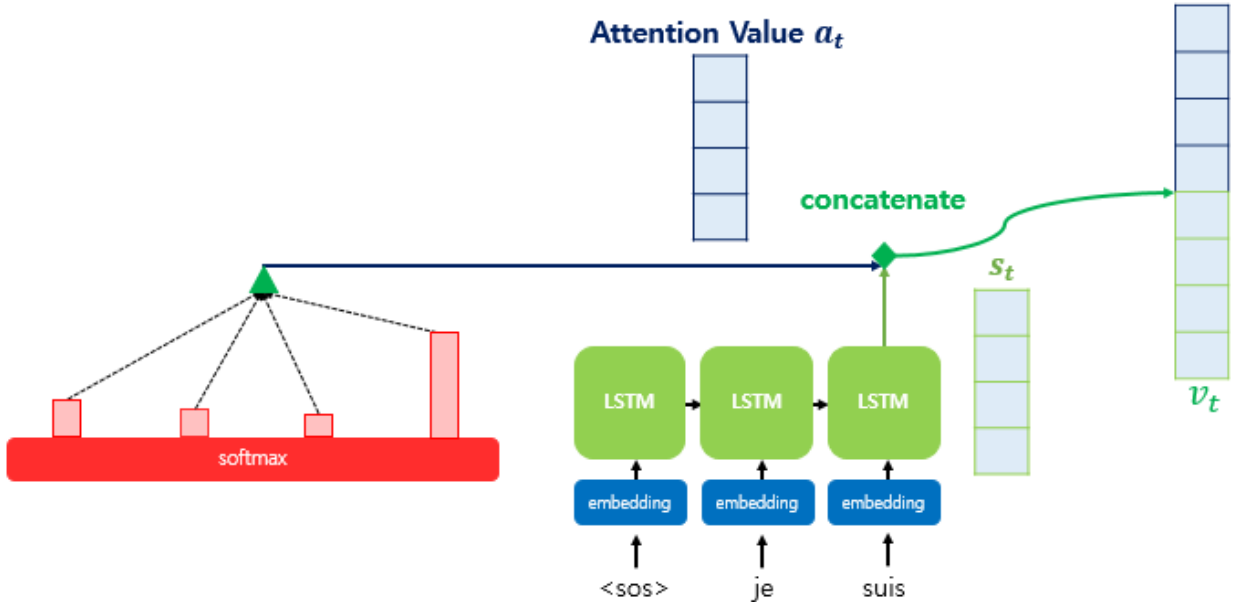



$$\alpha^t = softmax(e^t)$$

$$a_t = \sum_{i=1}^{N} \alpha_i^t h_i$$

# Dot-Product Attention

- 디코더의 hidden state와 Attention value $a_t$ 를 연결하고, FC layer를 통과시킨다 (예측값 출력)
  - $\boldsymbol{v}_t = [\boldsymbol{a}_t, \boldsymbol{s}_t]$
  - $\text{output} = \text{FC}(\boldsymbol{v}_t)$
    - Task에 따라 출력에 softmax 등을 취할 수 있다

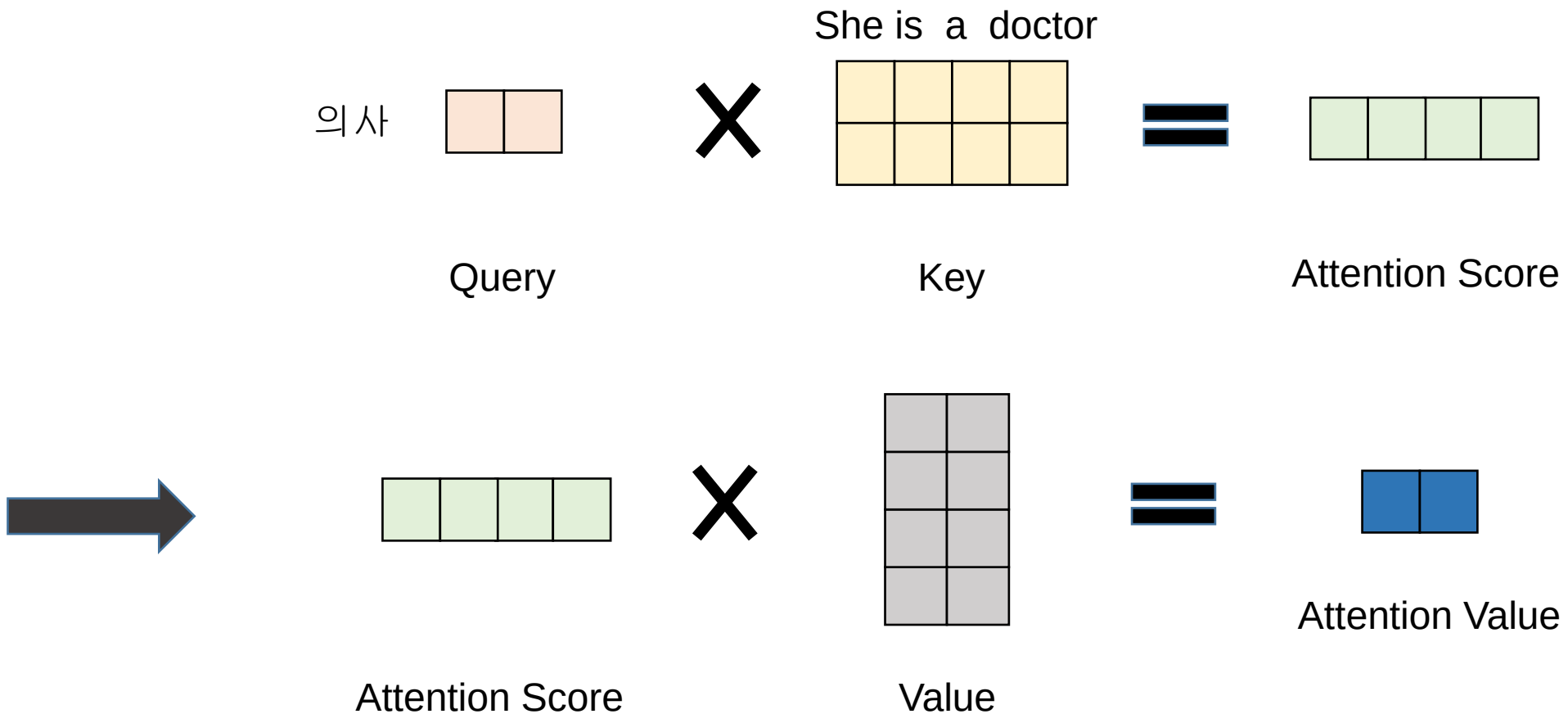


# Score function 종류

| 이름 | 스코어 함수 | **Defined by** |
|---|---|---|
| $dot$ | $score(s_t,\ h_i) = s_t^T h_i$ | Luong et al. (2015) |
| $scaled\ dot$ | $score(s_t,\ h_i) = \frac{s_t^T h_i}{\sqrt{n}}$ | Vaswani et al. (2017) |
| $general$ | $score(s_t,\ h_i) = s_t^T W_a h_i$ // 단, $W_a$는 학습 가능한 가중치 행렬 | Luong et al. (2015) |
| $concat$ | $score(s_t,\ h_i) = W_a^T\ tanh(W_b[s_t; h_i]), score(s_t,\ h_i) = W_a^T\ tanh(W_b s_t + W_c h_i)$ | Bahdanau et al. (2015) |
| $location - base$ | $\alpha_t = softmax(W_a s_t)$ // $\alpha_t$ 산출 시에 $s_t$만 사용하는 방법. | Luong et al. (2015) |

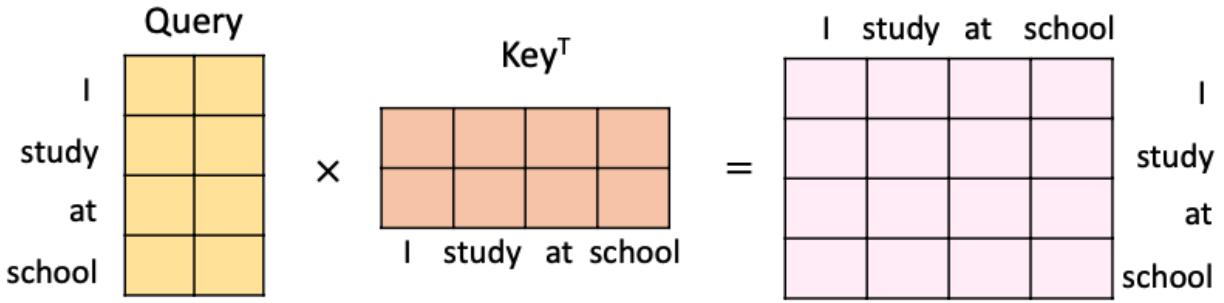
# Attention

- Attention의 구성 요소! (맥락의 정보를 흡수!!)

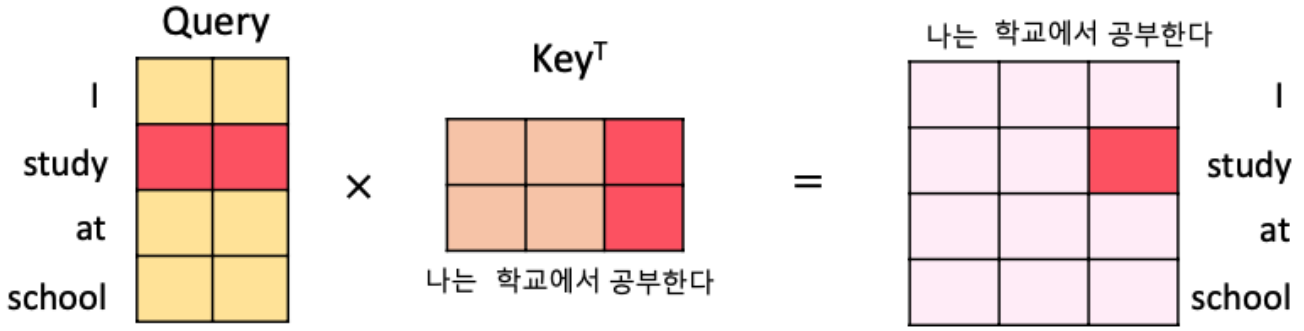


# Attention 종류

- Self Attention
  - 스스로의 관계를 파악




- Cross Attention
  - 다른것과의 관계를 파악 (ex 번역)



# 목차

# Transformer

- 기존의 모델 → Attention + RNN

- Transformer
  - Attention만으로 구성된 모델
  - 현재 가장 널리 사용되는 구조 중 하나 (RNN은 요즘 잘 사용되지 않음)

**Attention Is All You Need**


**Ashish Vaswani*** Google Brain avaswani@google.com

**Noam Shazeer*** Google Brain noam@google.com

**Niki Parmar*** Google Research nikip@google.com

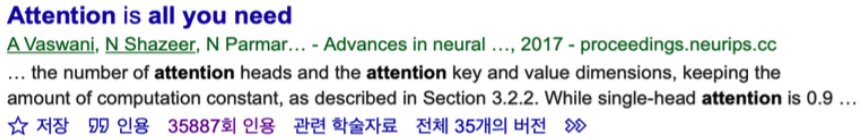
**Jakob Uszkoreit*** Google Research usz@google.com

**Llion Jones*** Google Research llion@google.com

**Aidan N. Gomez*** † University of Toronto aidan@cs.toronto.edu

**Łukasz Kaiser*** Google Brain lukaszkaiser@google.com

**Illia Polosukhin*** ‡ illia.polosukhin@gmail.com

**Attention is all you need**

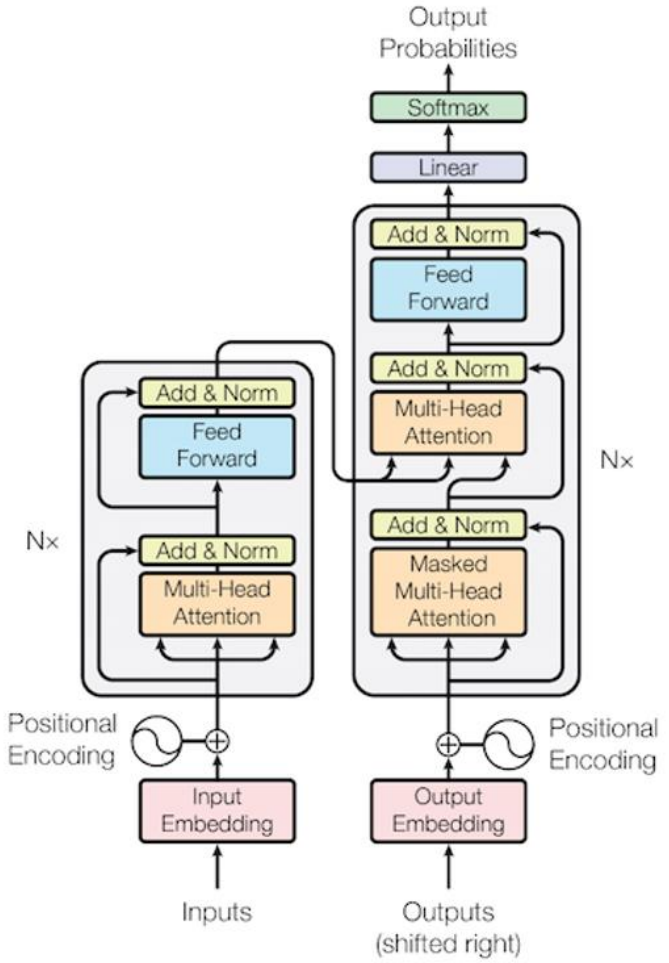
A Vaswani, N Shazeer, N Parmar… - Advances in neural …, 2017 - proceedings.neurips.cc

… the number of **attention** heads and the **attention** key and value dimensions, keeping the amount of computation constant, as described in Section 3.2.2. While single-head **attention** is 0.9 …

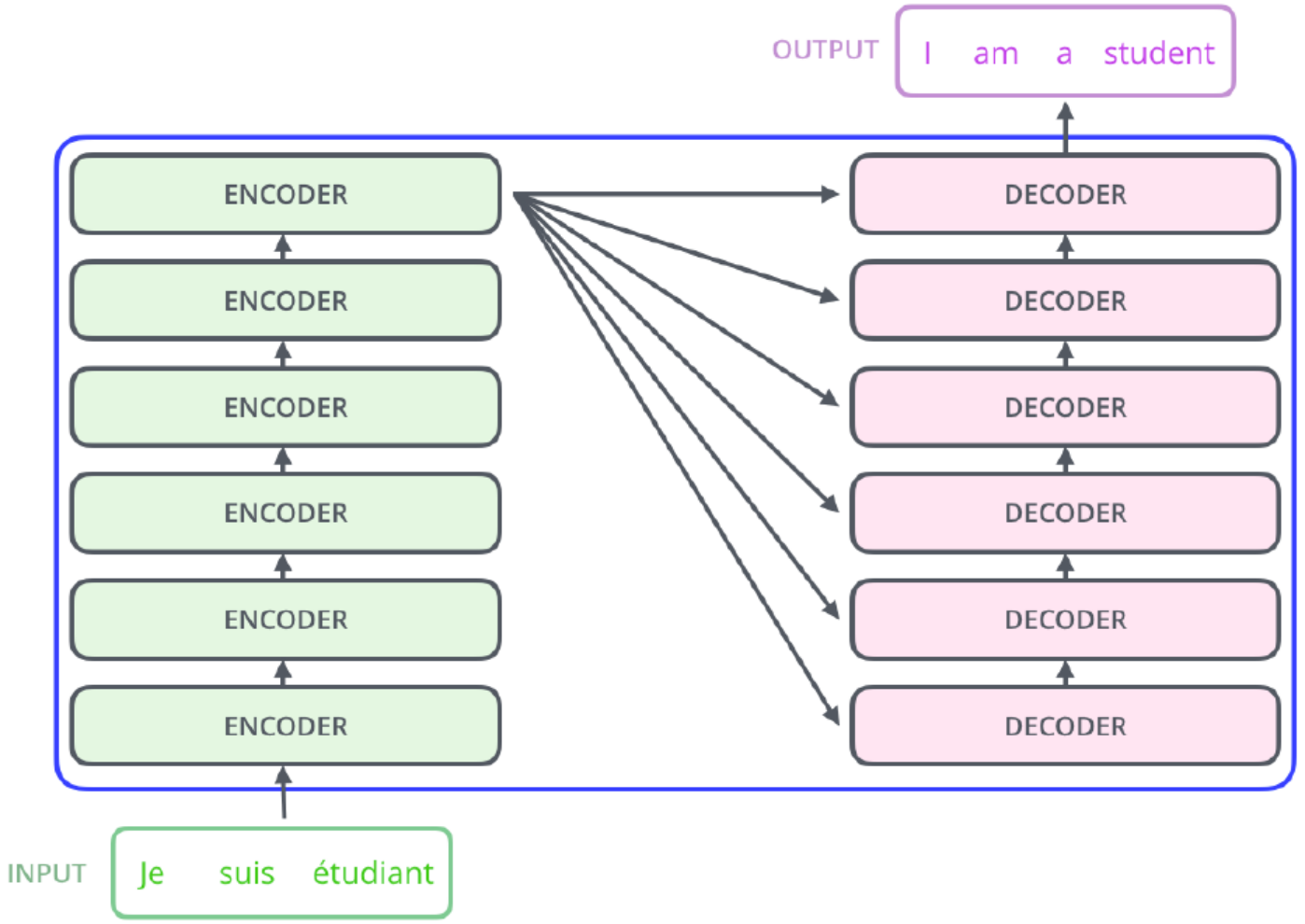
☆ 저장 99 인용 35887회 인용 관련 학술자료 전체 35개의 버전 »

# Transformer

- Encoder + Decoder
  - 각 모델 내에서 self attention을 수행하며
  - Encoder와 Decoder 사이에서 Cross attention을 수행한다



# Transformer



# Transformer: Multi Head Attention