BE530 – Medical Deep Learning

– Transformer –

Byoung-Dai Lee

Division of AI Computer Science and Engineering

Kyonggi University

KYONGGI UNIVERSITY

- Attention Is All You Need
- An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale

Transformer

- Attention Is All You Need

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

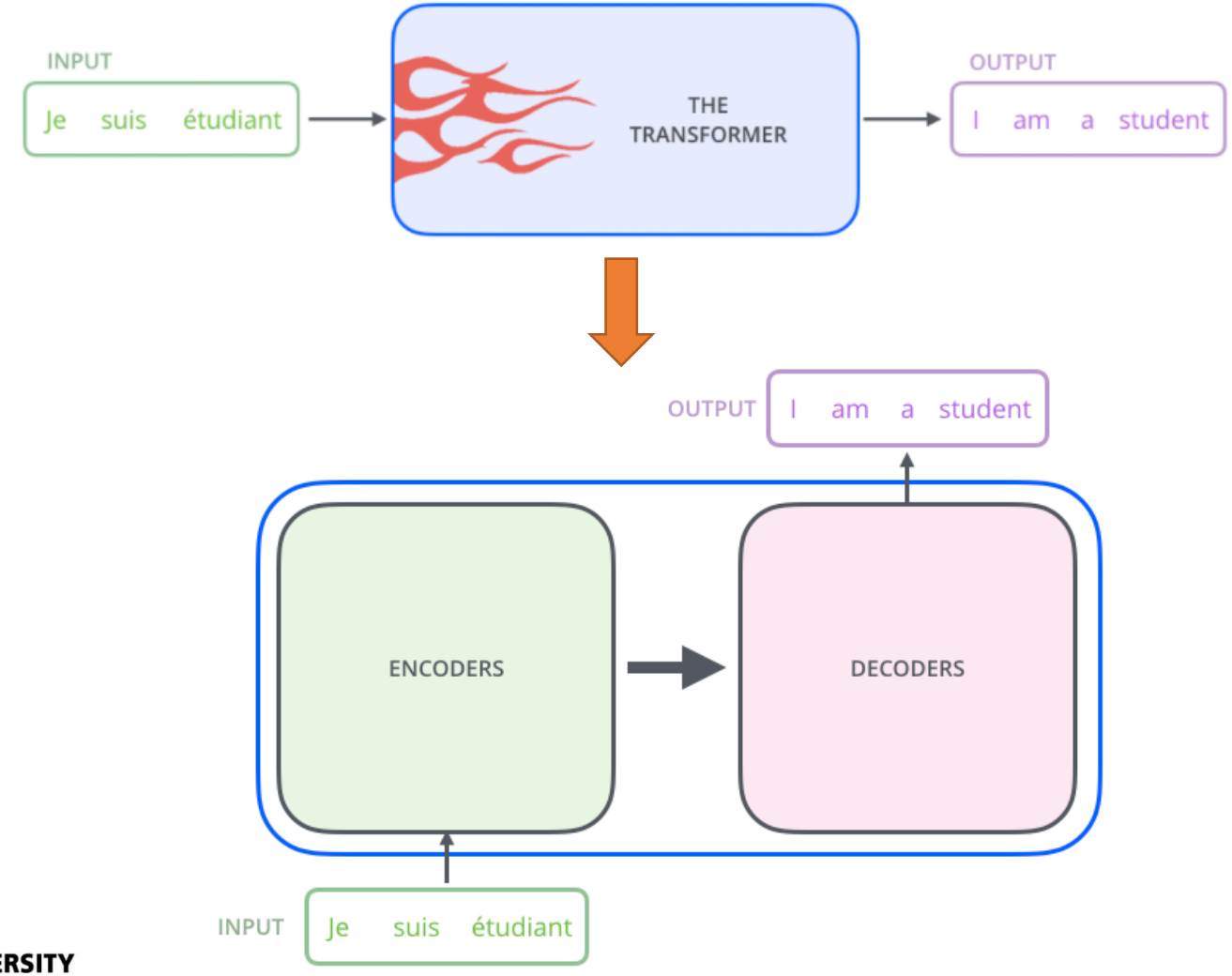
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

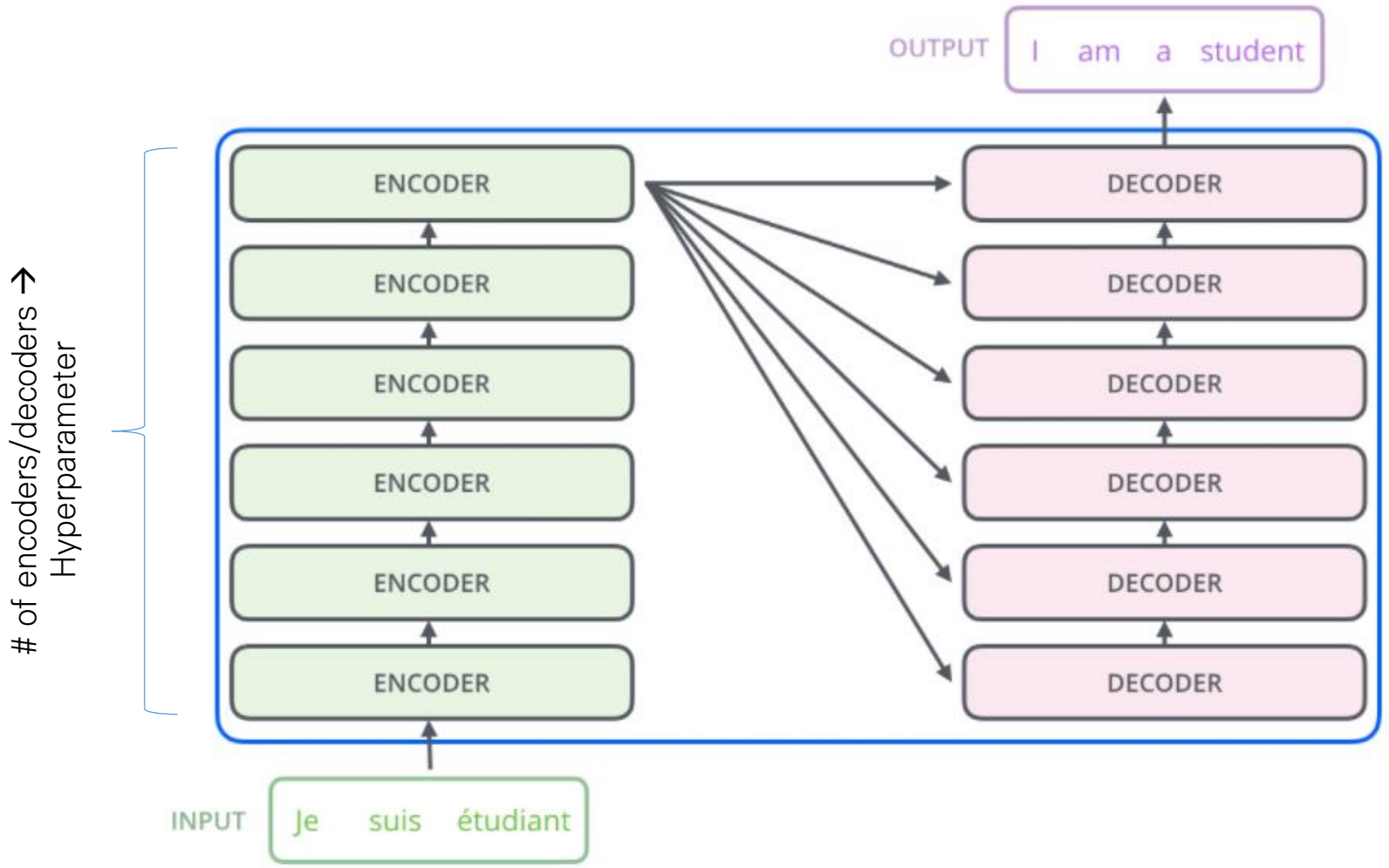
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

High-Level Picture

- An example of machine translation using the Transformer

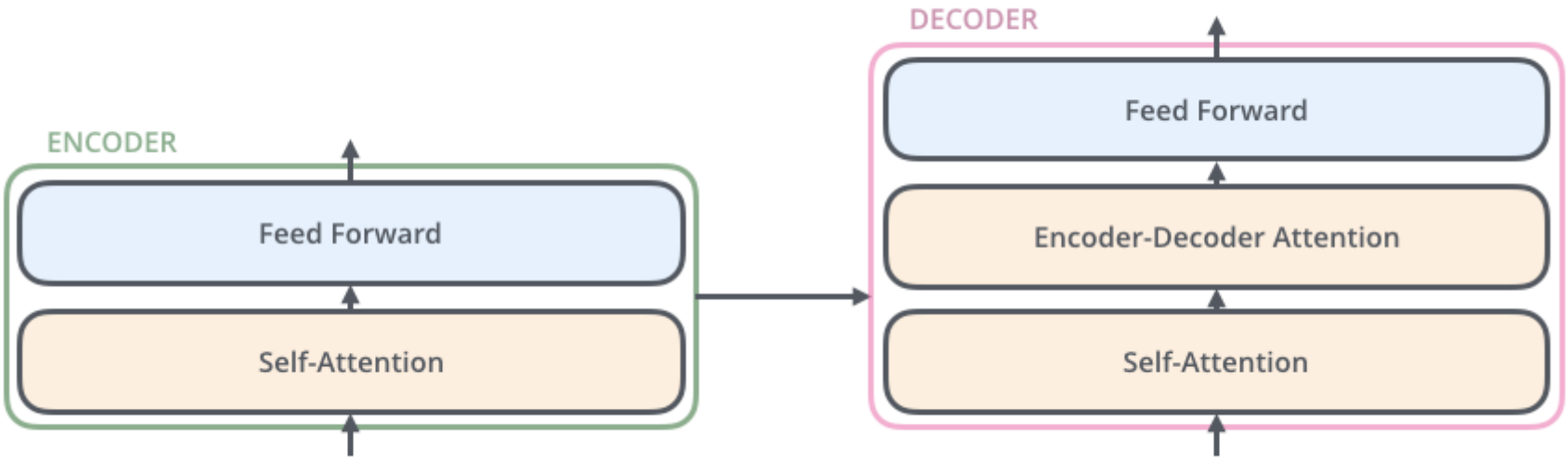


High-Level Picture (cont.)

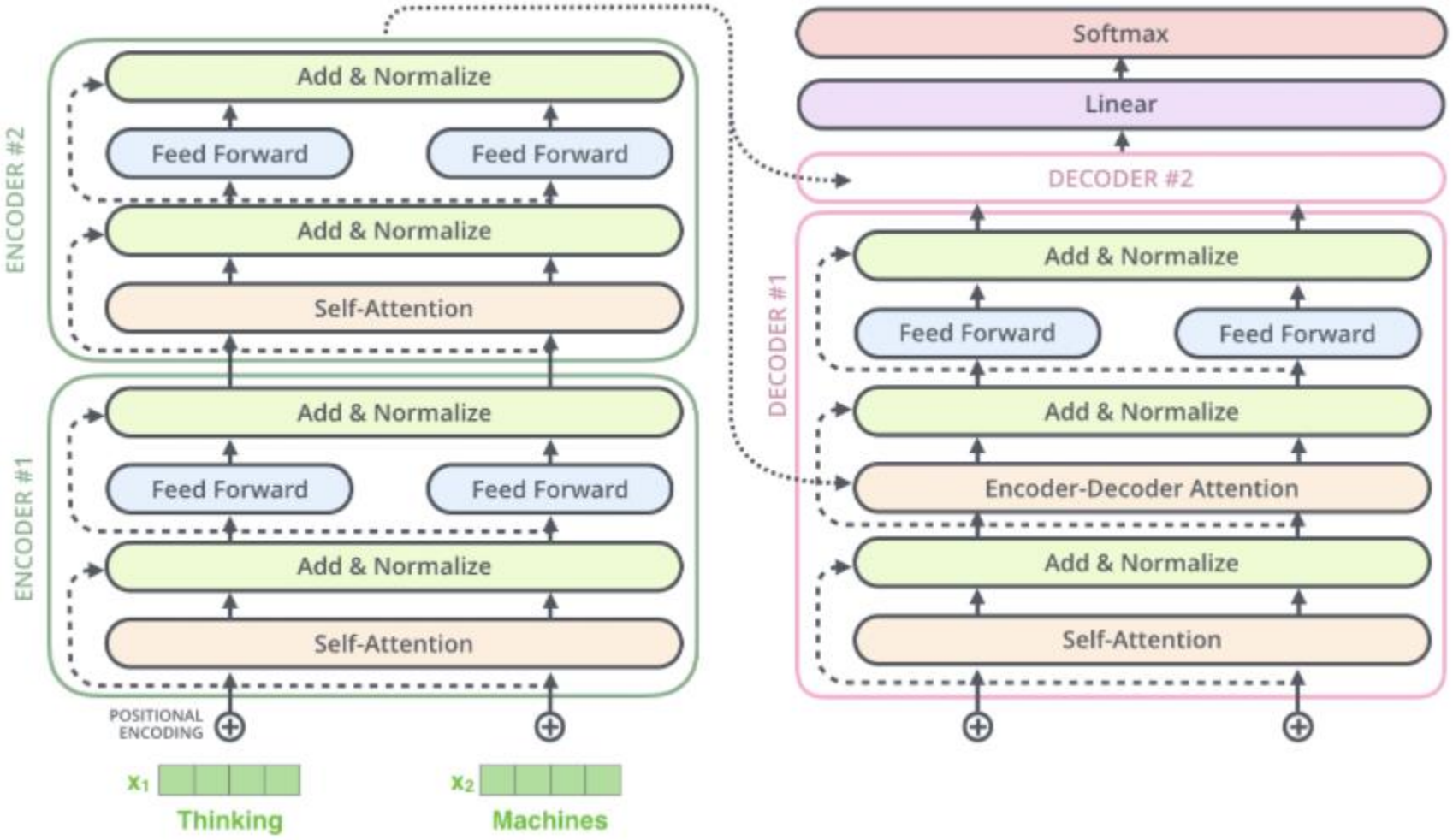


- 모든 Encoder/Decoder들은 동일한 구조를 가지나 weight는 서로 다름

High-Level Picture (cont.)



High-Level Picture (cont.)

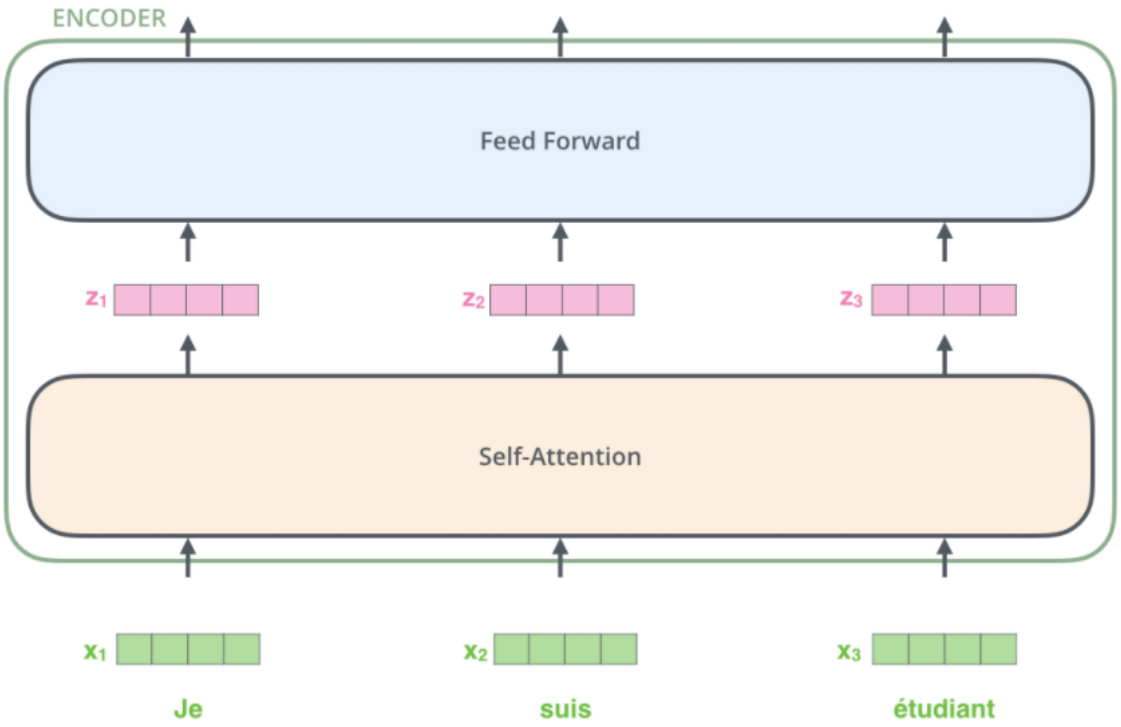


Hyperparameters

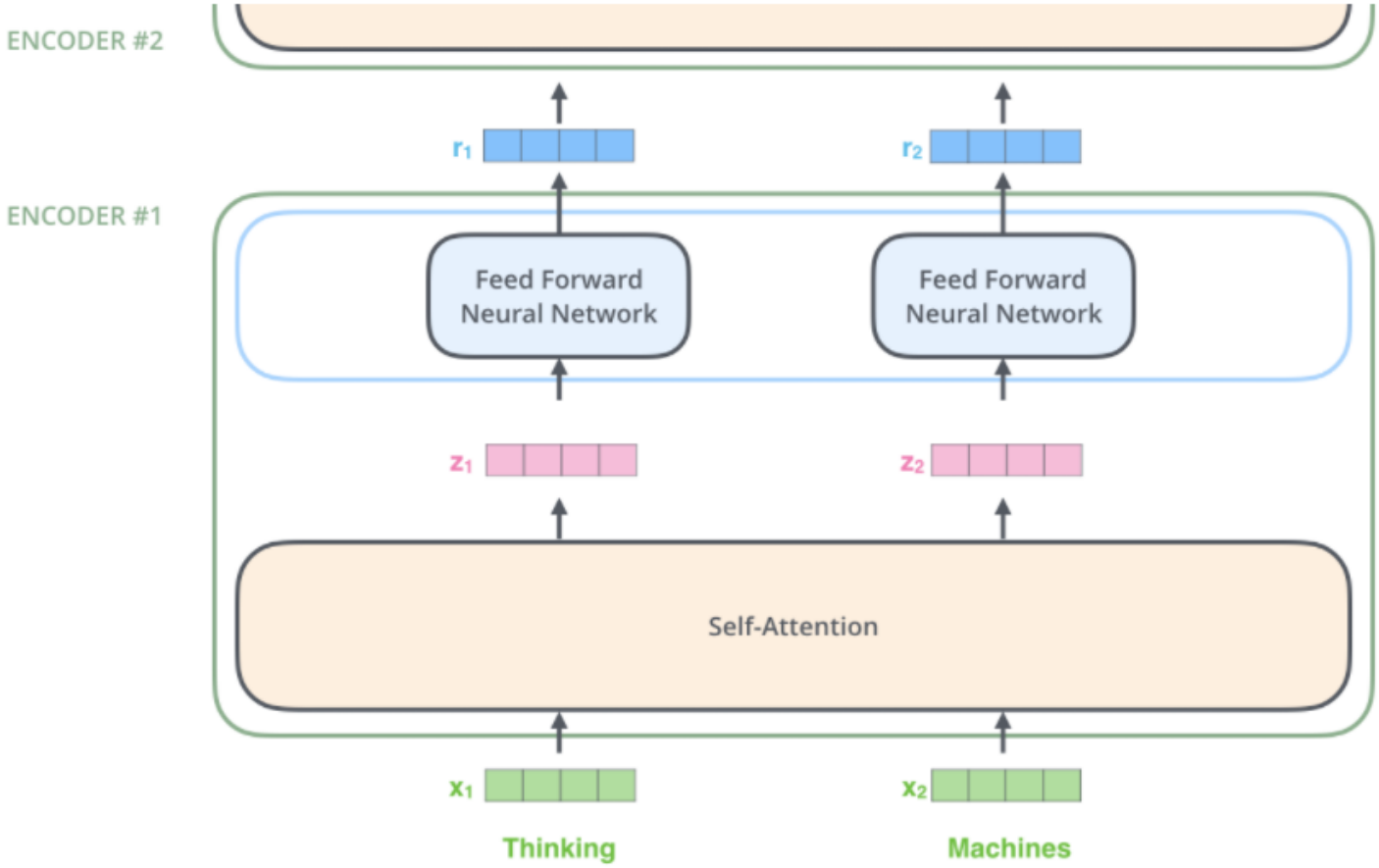
- d_{model} = 512
  - Encoder와 Decoder에서의 정해진 입력과 출력의 크기를 의미함
  - 임베딩 벡터의 차원은 d_{model}이며, Encoder/Decoder가 다음 층의 Encoder/Decoder로 값을 보낼 때도 이 차원을 유지함
- num_layers = 6
  - Encoder와 Decoder가 총 몇 개의 층으로 구성된지를 의미함
- num_heads = 6
  - Transformer는 attention 수행 시 병렬로 수행하고 그 결과값을 다시 하나로 합치는 방식을 채택함. 이 때 병렬 계산 노드의 수를 의미함
- d_{ff} = 2048
  - Transformer 내부에 사용되는 feed forward network에 존재하는 hidden layer의 크기를 의미함
  - Feed forward network의 입력층과 출력층의 크기는 d_{model}로 설정

From the perspective of vectors

- 입력 단어들은 embedding 알고리즘을 통해 벡터로의 변환 필요
- Embedding 변환은 가장 밑단의 encoder 입력 전에만 발생
  - Embedding Vector 리스트의 크기는 hyperparameter로 설정
    - Ex) 학습데이터셋에서 가장 긴 문자의 길이

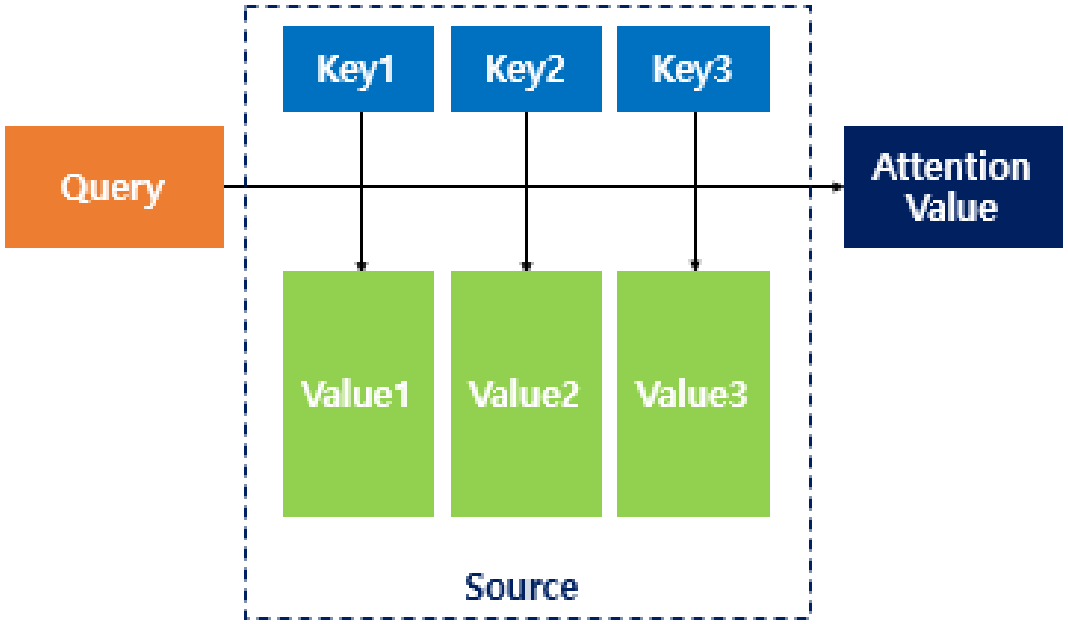


From the perspective of vectors (cont.)



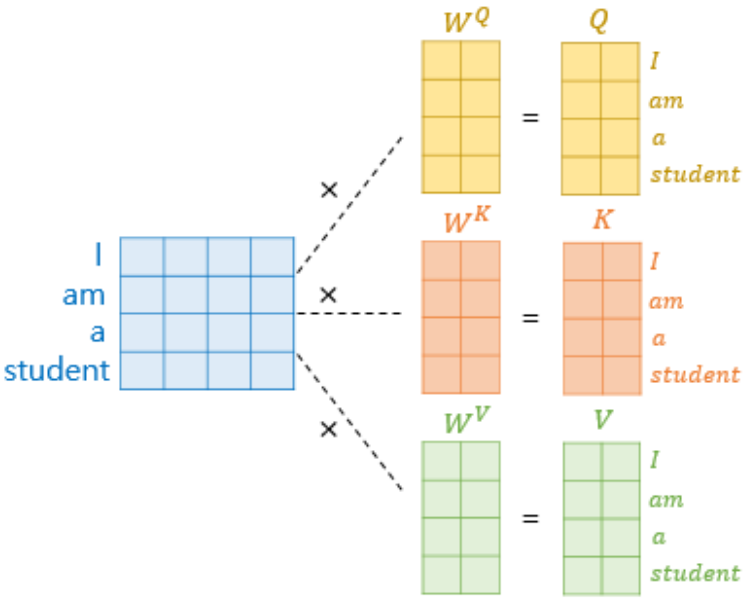
Self-Attention

- Key Concept
  - 현재 위치의 단어 (i.e., Query)를 encoding할 때 다른 단어들 (i.e., Key)과의 연관 관계를 고려하여 이들 단어들의 encoding 값들을 (i.e., Value) 현재 위치의 단어를 encoding할 때 사용



Self-Attention (cont.)

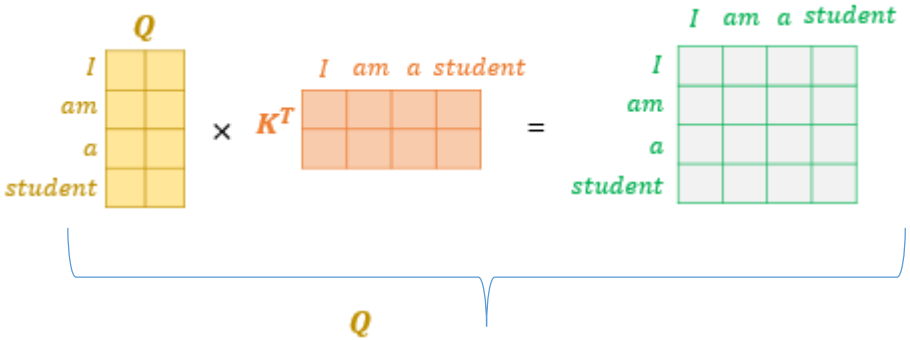
- Step 1) Q, K, V 벡터 생성
  - Encoder에 입력된 embedding 벡터들로부터 새로운 3개의 vector 생성
    - Query (Q), Key (K), Value (V) 벡터
  - 새로운 벡터는 d_{model}의 차원을 가지는 입력 embedding 벡터보다 더 작은 차원을 가짐
    - 논문에서는 64로 설정 (= d_{model}(512)/ num_heads(8))
  - 가중치 행렬 (WQ, ,)은 훈련과정에서 학습
    - 가중치 행렬 크기 – $d_{model} \times (d_{model}/num\_heads)$

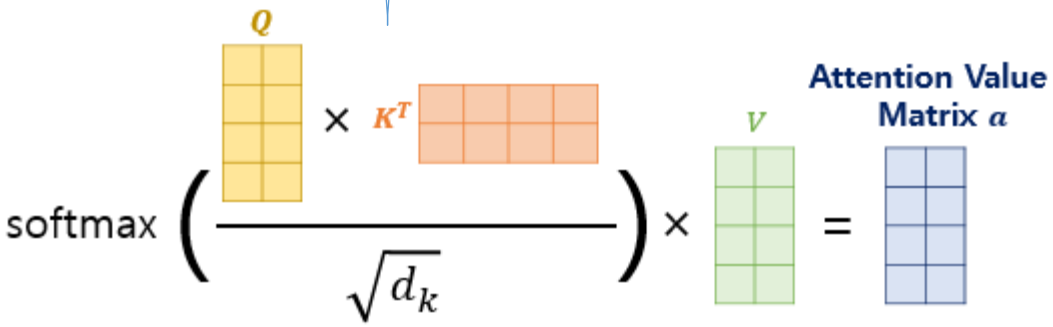


Self-Attention (cont.)

- Step 2) Scaled dot-product Attention

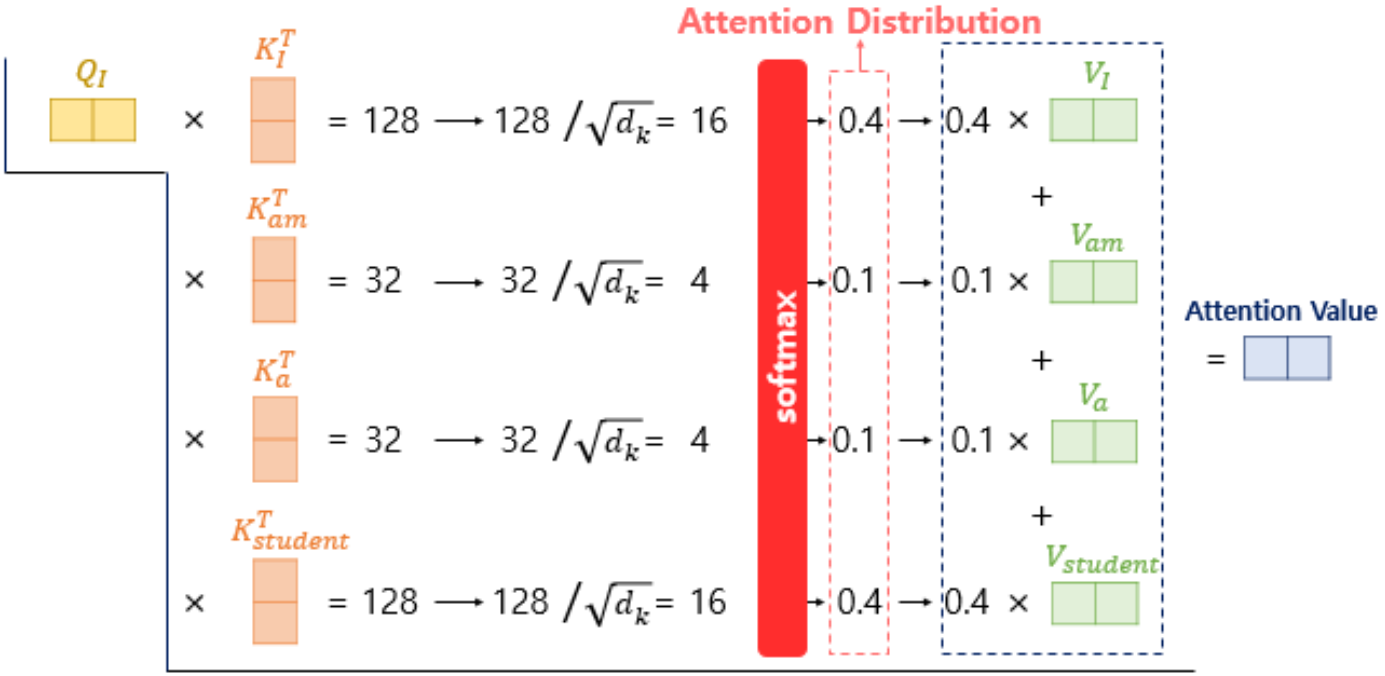
$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$





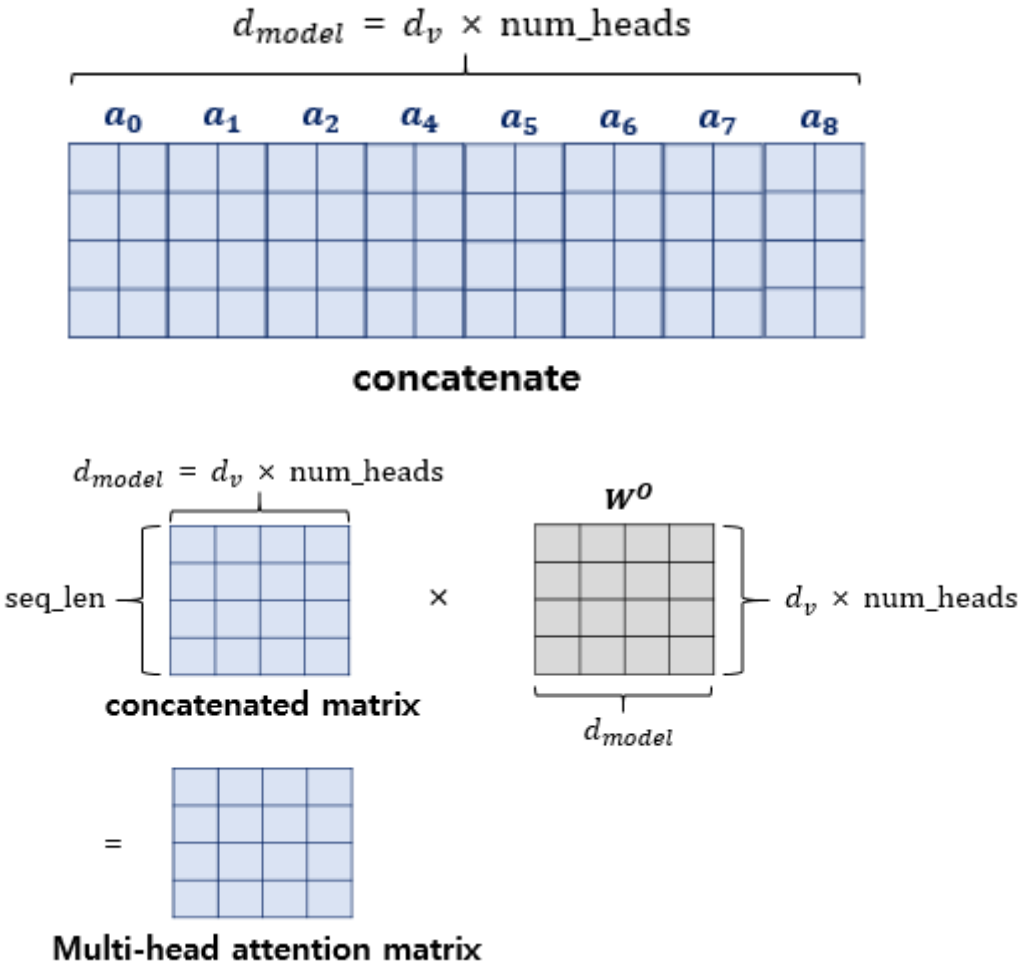
Self-Attention (cont.)

- Step 2) Scaled dot-product Attention (cont.)



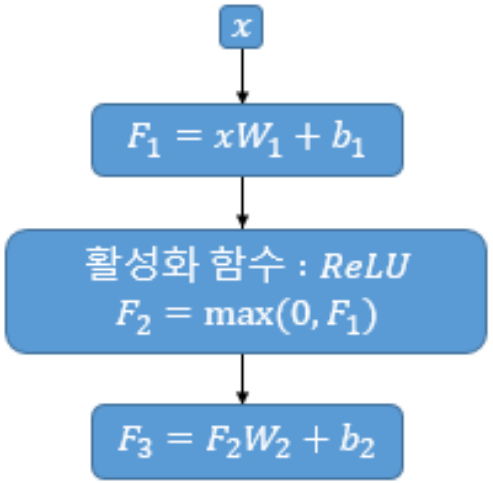
Multi-Headed Attention

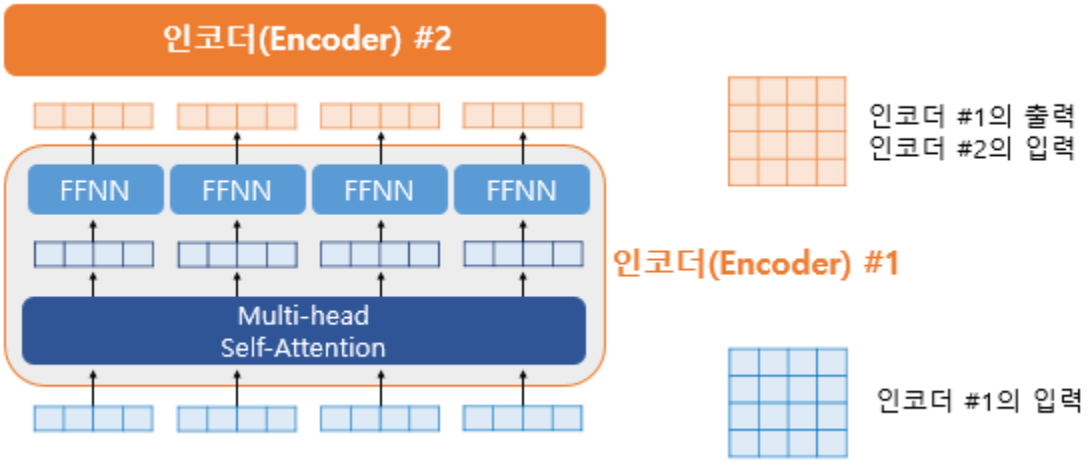
- 복수 개의 Attention을 사용하여 Feature Representation 성능 개선



Position-wise Feed Forward NN

$$FFNN(x) = MAX(0, xW_1 + b_1)W_2 + b_2$$

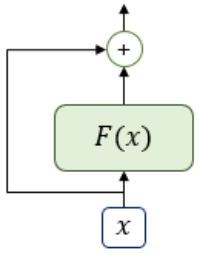


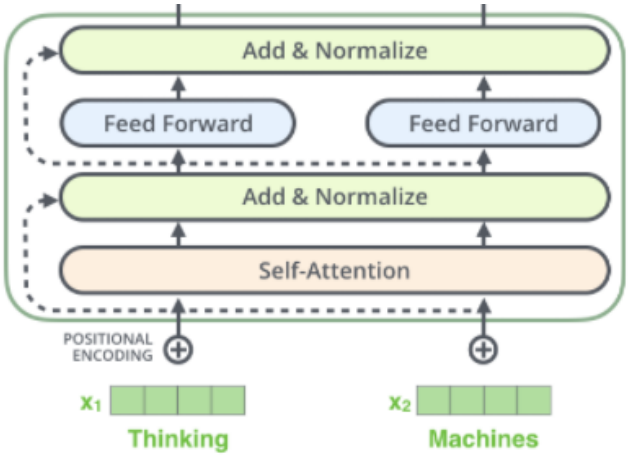


Residual Connection & Layer Normalization

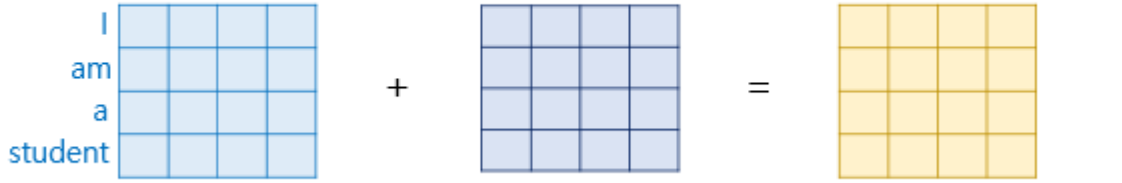
- Residual Connection

$$H(x) = x + F(x)$$



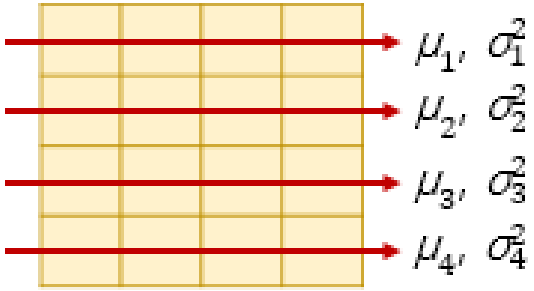


$$H(x) = x + Multi - head\ Attention(x)$$



Residual Connection & Layer Normalization (cont.)

$$LN = LayerNorm(x + Sublayer(x))$$



Residual Connection output

(seq_len, d_model)

$$\hat{x}_{i,k} = \frac{x_{i,k} - \mu_i}{\sqrt{\sigma_i^2 + \epsilon}}$$

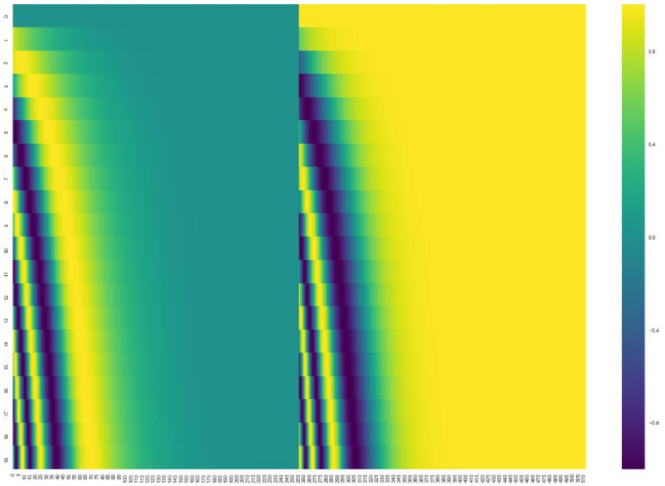
$$ln_i = \gamma \hat{x}_i + \beta = LayerNorm(x_i)$$

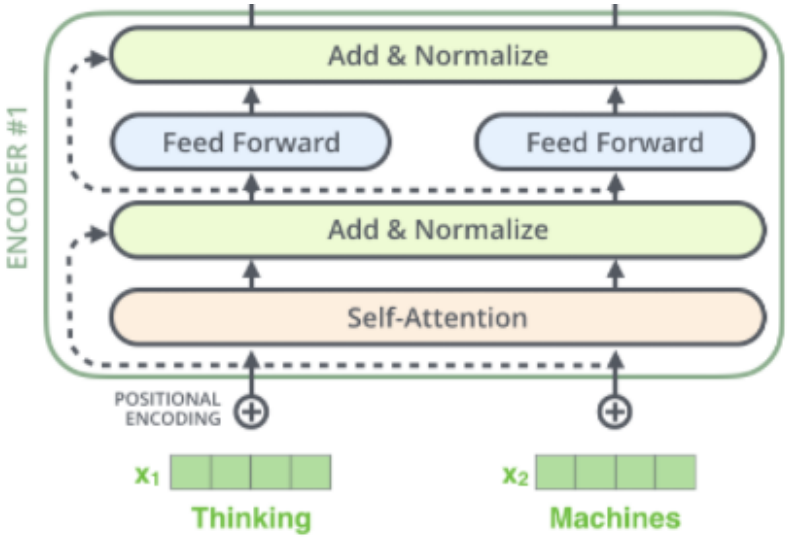
Positional Encoding

- 단어의 위치 정보를 얻기 위해서 각 단어의 embedding 벡터에 위치 정보들을 더하여 모델의 입력으로 사용

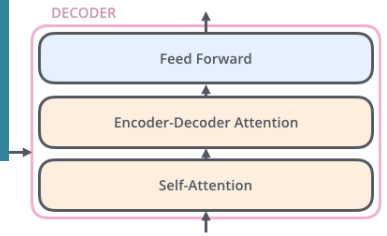
$$PE_{(pos,\,2i)} = sin(pos/10000^{2i/d_{model}})$$

$$PE_{(pos,\,2i+1)} = cos(pos/10000^{2i/d_{model}})$$

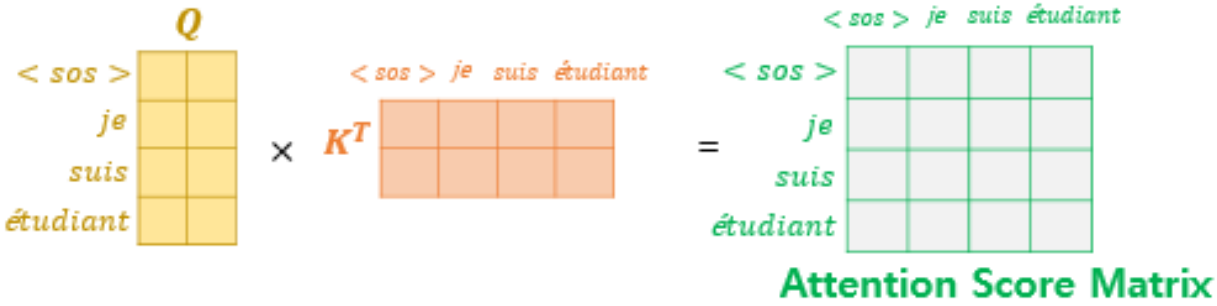


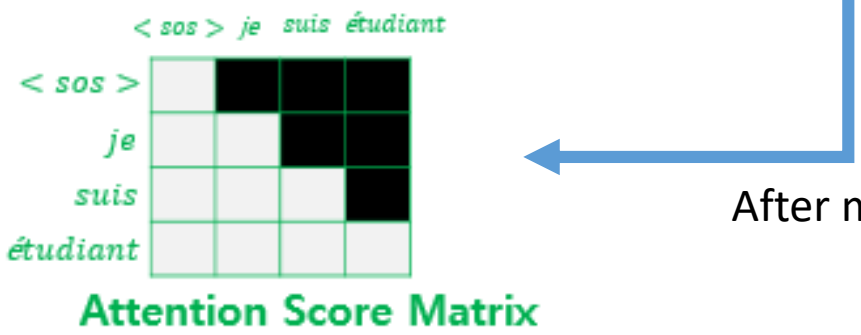


Decoder: Masked MHS-Attention



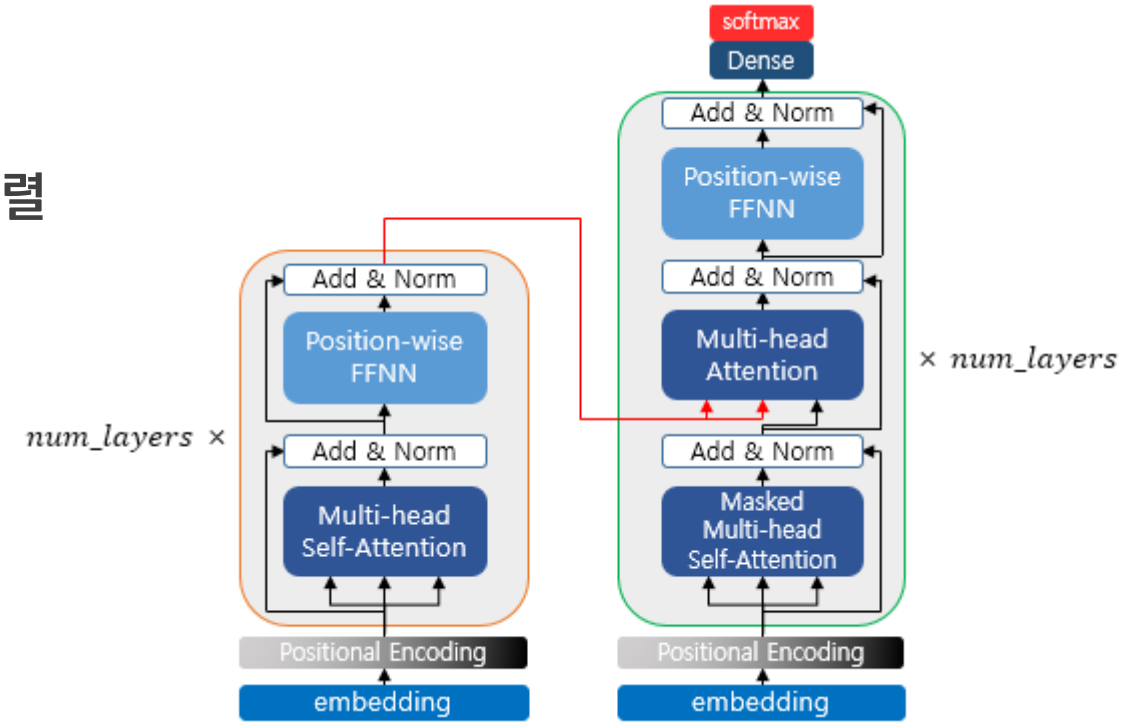
- Decoder에서의 Self-Attention은 현재 위치의 이전 위치에 대해서만 Attend할 수 있음
  - Self-Attention 계산 과정에서 Attention Score 행렬을 대상으로 현재 위치 이후의 위치에 대해 마스킹을 통해 가능





Decoder: Encoder-Decoder Attention

- 인코더의 첫번째 서브층
  - Query = Key = Value
- 디코더의 첫번째 서브층
  - Query = Key = Value
- 디코더의 두번째 서브층
  - Query : 디코더 행렬
  - Key = Value : 인코더 행렬



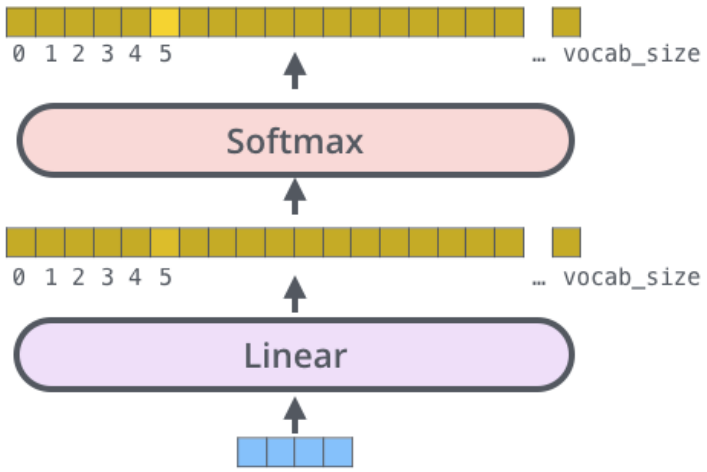
Linear Layer & Softmax Layer

Which word in our vocabulary is associated with this index?

am

Get the index of the cell with the highest value (**argmax**)

5



Vision Transformer (ViT)

- An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale

AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE

Alexey Dosovitskiy[*,†], **Lucas Beyer**[*], **Alexander Kolesnikov**[*], **Dirk Weissenborn**[*], **Xiaohua Zhai**[*], **Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby**[*,†]

[*]equal technical contribution, [†]equal advising

Google Research, Brain Team

{adosovitskiy, neilhoulsby}@google.com

ViT (cont.)

- Key Points
  - Transformer를 최소한의 변경을 가한 후 Image Recognition에 적용
    - 전체 이미지를 패치로 나눈 후 패치를 하나의 word로 간주하여 Transformer에 입력으로 제공
    - Supervised Image Classification
  - CNN과 비교했을 때 Transformer의 일반적인 특징
    - Translation Equivalence, Locality와 같은 CNN 고유의 특징이 부족하기 때문에 insufficient amounts of data를 대상으로 학습할 경우 일반화 성능이 떨어질 수 있음
    - 그러나 충분히 큰 규모의 데이터셋을 대상으로 pretraining 한 후 특정 태스크를 대상으로 소규모 데이터를 이용하여 fine-tuing을 수행할 경우 우수한 성능을 제공함

ViT (cont.)

- Model Overview

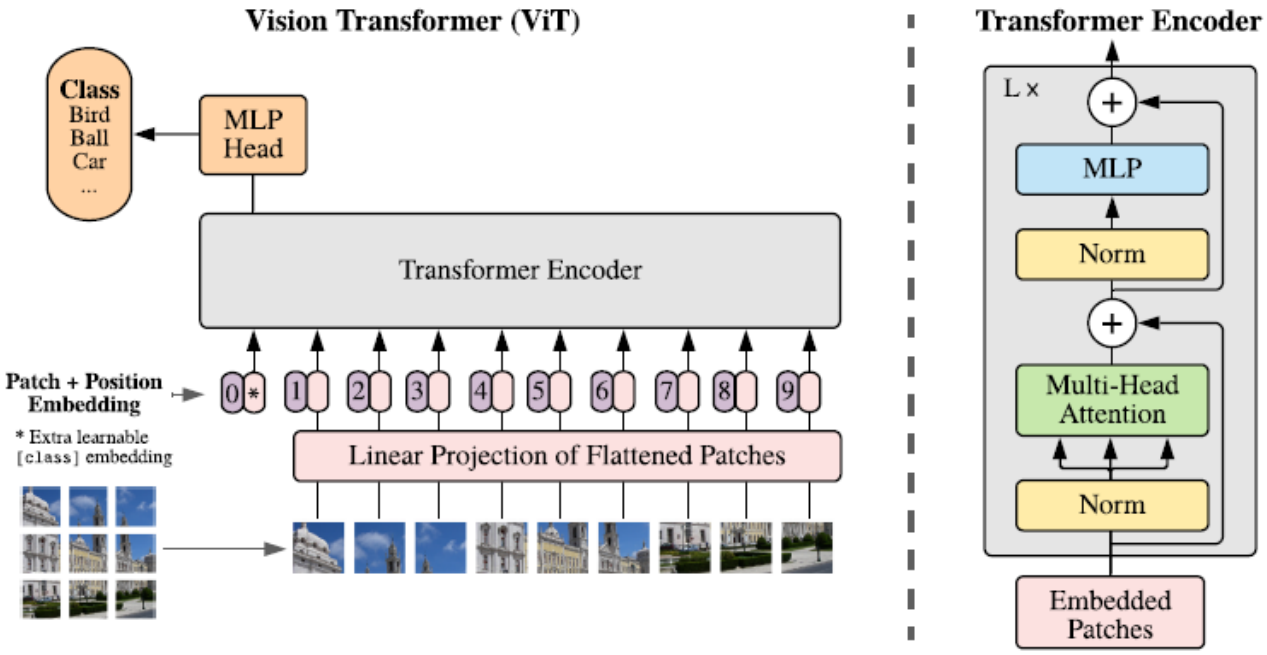


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification token" to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

ViT (cont.)

- Input sequence 생성

$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}}; \mathbf{x}_p^1\mathbf{E}; \mathbf{x}_p^2\mathbf{E}; \cdots; \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos}, \qquad \mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times D},\ \mathbf{E}_{pos} \in \mathbb{R}^{(N+1) \times D} \tag{1}$$

$$\mathbf{z}'_\ell = \text{MSA}(\text{LN}(\mathbf{z}_{\ell-1})) + \mathbf{z}_{\ell-1}, \qquad \ell = 1 \ldots L \tag{2}$$

$$\mathbf{z}_\ell = \text{MLP}(\text{LN}(\mathbf{z}'_\ell)) + \mathbf{z}'_\ell, \qquad \ell = 1 \ldots L \tag{3}$$

$$\mathbf{y} = \text{LN}(\mathbf{z}_L^0) \tag{4}$$

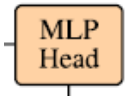
An overview of the model is depicted in Figure 1. The standard Transformer receives as input a 1D sequence of token embeddings. To handle 2D images, we reshape the image $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$ into a sequence of flattened 2D patches $\mathbf{x}_p \in \mathbb{R}^{N \times (P^2 \cdot C)}$, where (H, W) is the resolution of the original image, C is the number of channels, (P, P) is the resolution of each image patch, and $N = HW/P^2$ is the resulting number of patches, which also serves as the effective input sequence length for the Transformer. The Transformer uses constant latent vector size D through all of its layers, so we flatten the patches and map to D dimensions with a trainable linear projection (Eq. 1). We refer to the output of this projection as the patch embeddings.

ViT (cont.)

ViT (cont.)

ViT (cont.)

- Inductive bias



  - self-attention layer – 모든 입력 간의 attention score를 계산하기 때문에 global operation으로 볼 수 있음.
  - MLP layer – 각 패치 영상만을 대상으로 독립적으로 수행하기 때문에 local이라고 할 수 있으며, 해당 패치를 receptive field라고 봤을 때 중요 특징만을 추출하기 때문에 translation equivalent라고 할 수 있음
- Fine-tuning & Higher resolution
  - Fine-tuning 시에는 pre-training 시에 사용된 MLP head를 제거하고 0으로 초기화된 DxK MLP 층으로 교체함
    - D: z_L^0, K: the number of downstream classes
  - 입력 영상이 pre-training에 사용된 영상보다 해상도가 큰 경우
    - 패치 크기 (pxp)는 그대로 유지함. 이 경우 1) 입력 sequence의 크기 (N)가 더 증가하게 되며, pre-training 시 학습된 position encoding이 더 이상 쓸모가 없어짐
    - position encoding의 경우 2D interpolation을 적용함

ViT (cont.)

- Pre-training dataset requirements
  - pre-training 할 때 사용되는 데이터셋이 크면 클수록 ViT의 성능이 개선됨. 그러나 작은 데이터셋을 이용하여 pre-training을 수행할 경우 CNN보다 성능이 떨어짐 (Figure 3)
  - 작은 데이터셋을 이용하면 CNN의 특성 (locality, translation equivalence 등)으로 인해 성능이 우세하지만 대규모 데이터셋을 이용하면 네트워크가 그러한 특성이 없더라도 성능이 좋아짐 (Figure 4)

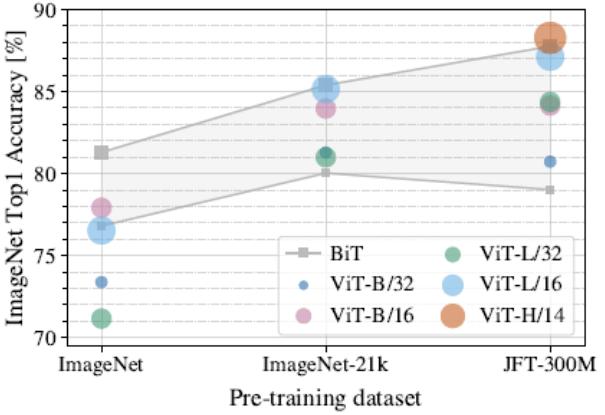


Figure 3: Transfer to ImageNet. While large ViT models perform worse than BiT ResNets (shaded area) when pre-trained on small datasets, they shine when pre-trained on larger datasets. Similarly, larger ViT variants overtake smaller ones as the dataset grows.

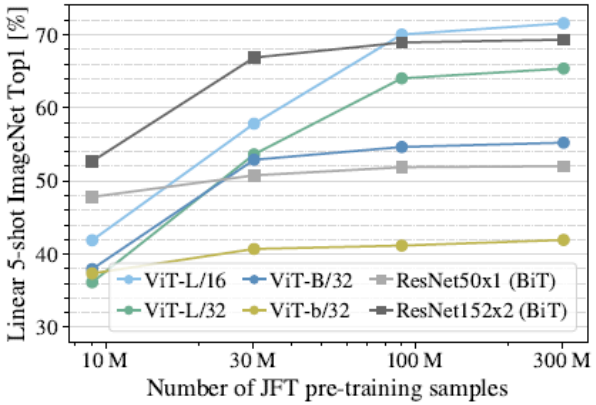
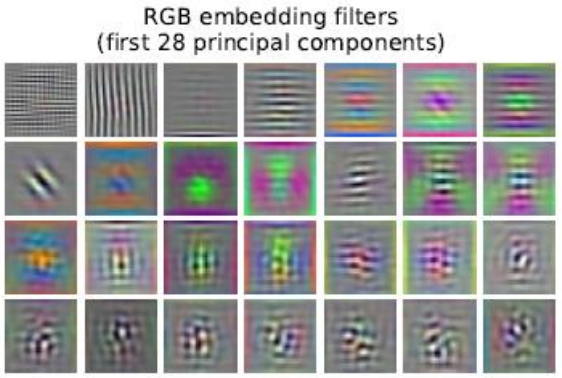
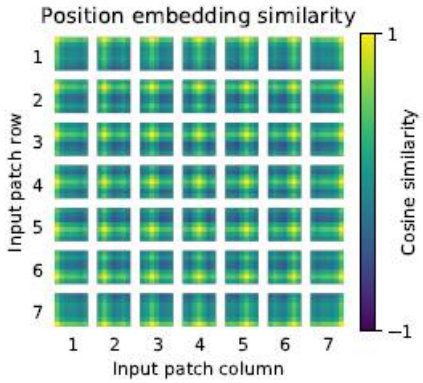


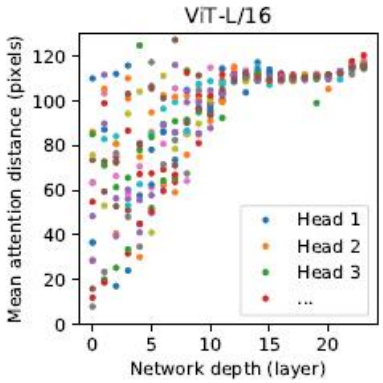
Figure 4: Linear few-shot evaluation on ImageNet versus pre-training size. ResNets perform better with smaller pre-training datasets but plateau sooner than ViT, which performs better with larger pre-training. ViT-b is ViT-B with all hidden dimensions halved.

ViT (cont.)

- Inspecting ViT
  - ViT의 첫번째 layer는 입력 패치를 대상으로 linear projection layer에서 low-dimension vector (1xD)를 생성하는 것임. 이 때 학습된 embedding filter (e.g., E)는 일반적으로 CNN의 저차원에서 볼 수 있는 필터와 유사함
  - Position Embedding
    - 가까운 패치일수록 유사한 position embedding을 적용하게 됨
  - Attention layers를 통해서 입력 이미지의 전체 pixel들을 모두 고려하는 global한 특성을 확인할 수 있음. 또한 일부 attention head의 경우 전체 픽셀보다는 가까운 위치에 있는 픽셀들을 더 중점적으로 고려하는 것도 확인 가능함







ViT (cont.)

- Inspecting ViT (cont.)
  - 최종 학습된 attention은 입력 이미지의 중요한 부분을 강조하고 있음을 확인할 수 있음

D.8 ATTENTION MAPS

To compute maps of the attention from the output token to the input space (Figures 6 and 14), we used Attention Rollout (Abnar & Zuidema, 2020). Briefly, we averaged attention weights of ViT-L/16 across all heads and then recursively multiplied the weight matrices of all layers. This accounts for the mixing of attention across tokens through all layers.

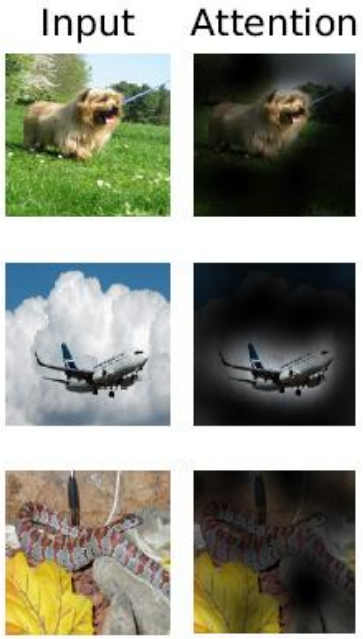


Figure 6: Representative examples of attention from the output token to the input space. See Appendix D.7 for details.

References

- Attention is All You Need, https://https://arxiv.org/abs/1706.03762
- The Illustrated Transformer, https://jalammar.github.io/illustrated-transformer/
  - Korean version: https://nlpinkorean.github.io/illustrated-transformer/
- 딥러닝을 이용한 자연어 입문, https://wikidocs.net/31379
- Youtube Lecture
  - https://www.youtube.com/channel/UCjRHzKZ9VMcWPL7NsvBsKHg
- An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, https://arxiv.org/pdf/2010.11929.pdf


ANY
QUESTIONS?