Attention is all you need

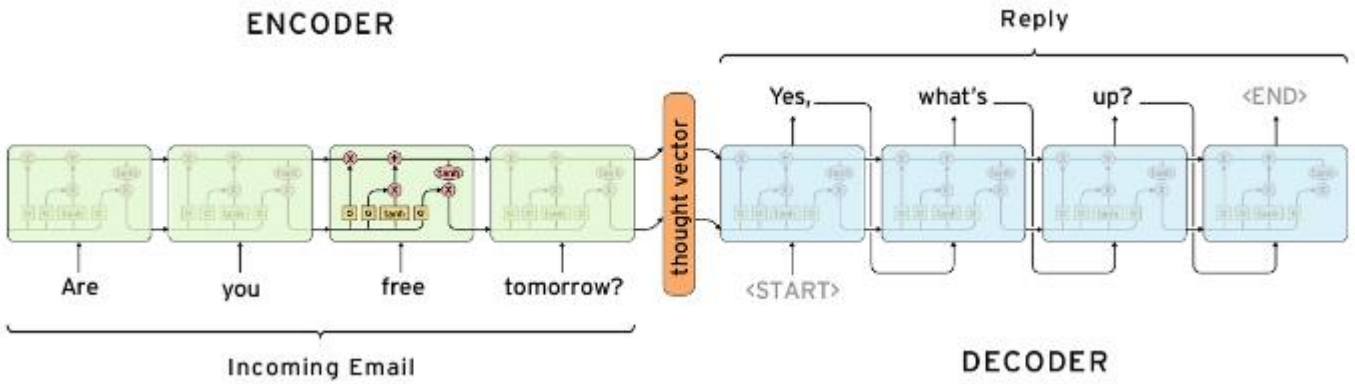
Ashish Vaswani et al.

Google brain, Google research

Advances in Neural Information Processing Systems 30 (NIPS 2017)

abstract

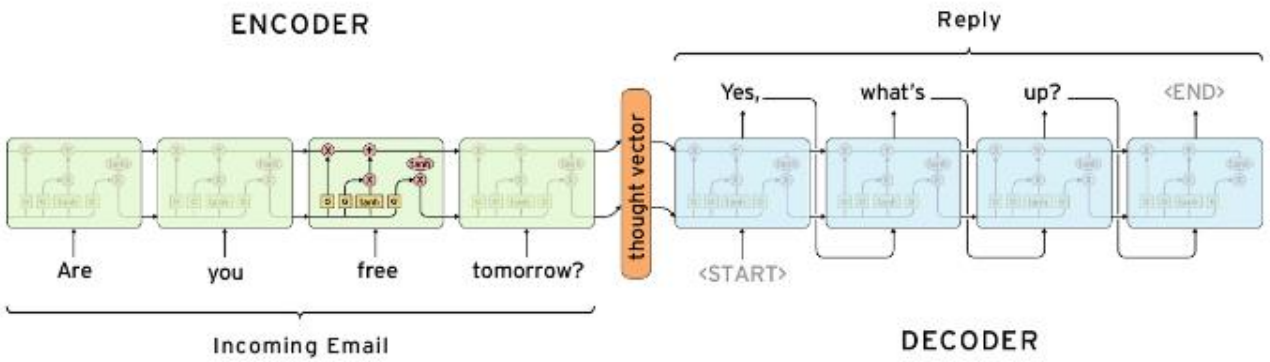
- The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder.
- The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely.

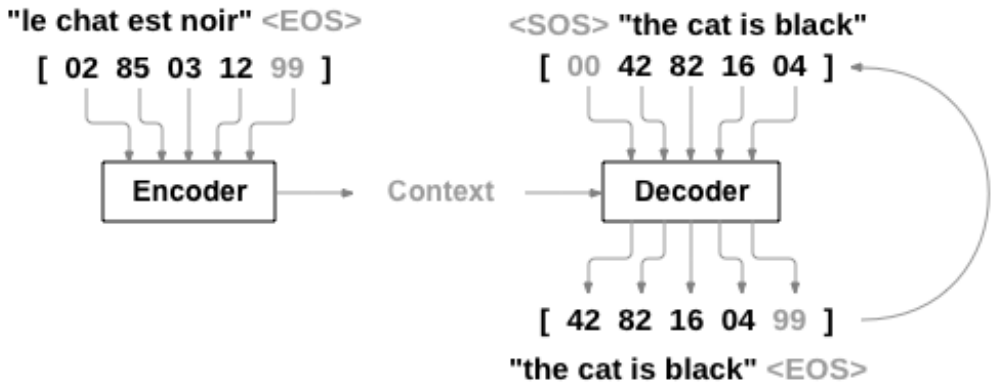


이전의 sequence transduction model에서 RNN, CNN 을 기반으로 하는 encoder – decoder 모델을 사용해 왔음

Attention

seq2seq 모델은 **인코더**에서 입력 시퀀스를 컨텍스트 벡터라는 하나의 고정된 크기의 벡터 표현으로 압축하고, **디코더**는 이 컨텍스트 벡터를 통해서 출력 시퀀스를 만들어 냈다.





Attention

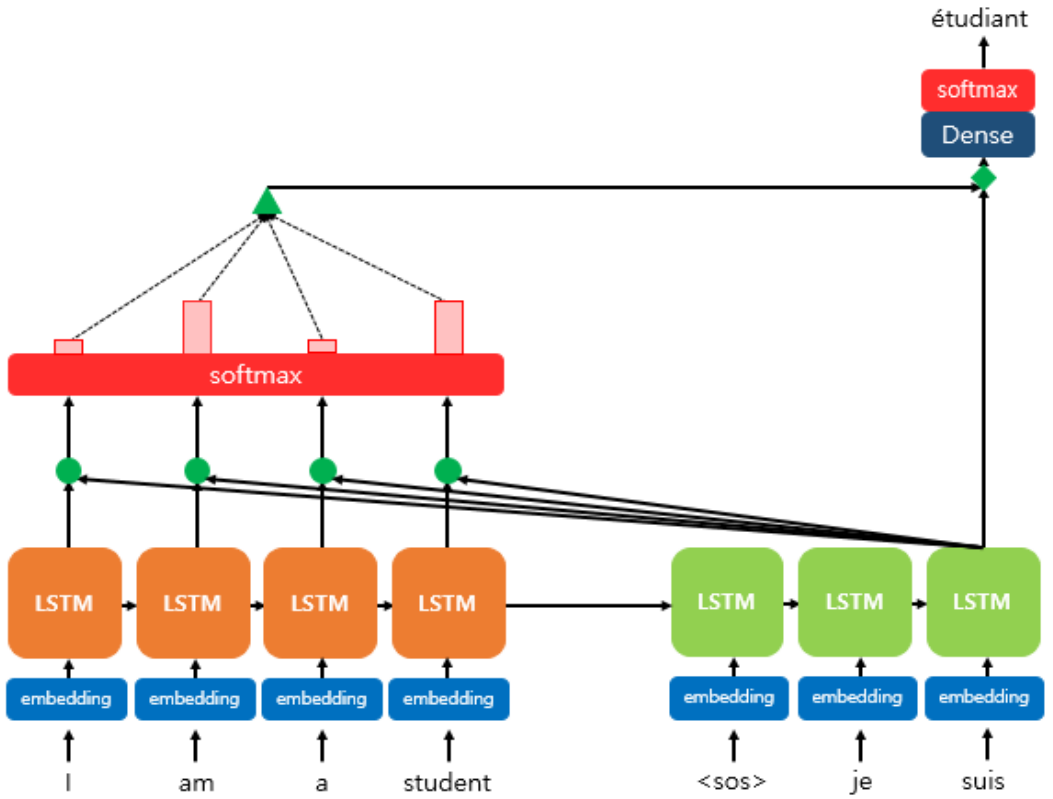
어텐션(Attention)의 아이디어

어텐션의 기본 아이디어는 디코더에서 출력 단어를 예측하는 매 시점(time step)마다, 인코더에서의 전체 입력 문장을 다시 한 번 참고 단, 전체 입력 문장을 전부 다 동일한 비율로 참고하는 것이 아니라, 해당 시점에서 예측해야할 단어와 연관이 있는 입력 단어 부분을 좀 더 집중(attention)해서 보게 된다.

예를 들어 영어 문장 "I am a student."을 프랑스어 문장 "Je suis étudiant. " 번역 상황에서. 출력 시퀀스의 단어 "étudiant"(프랑스어로 학생을 뜻하는 단어)는 입력 시퀀스의 단어 "i", "am", "a", "student" 중 "student"(영어로 학생을 뜻하는 단어)와 연관이 깊다. 이때 어텐션 메커니즘은 디코더가 "étudiant"를 출력하기 직전의 은닉 상태는 인코더가 "student"를 입력받은 직후의 은닉 상태와 유사할 것이라 가정한다.

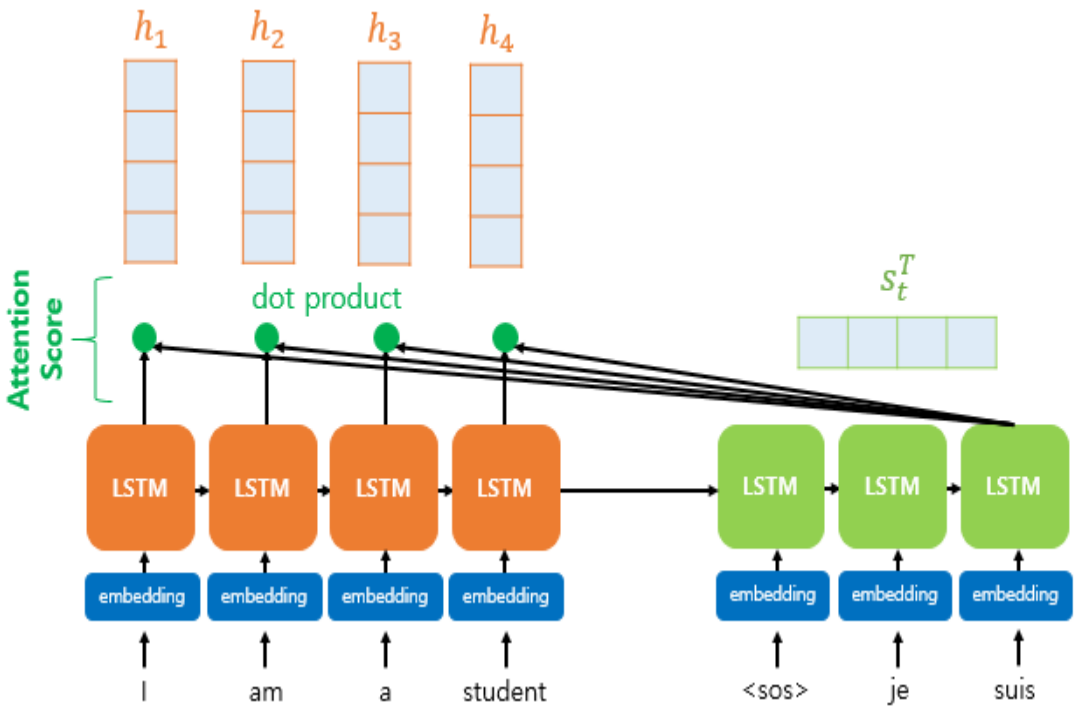
Attention

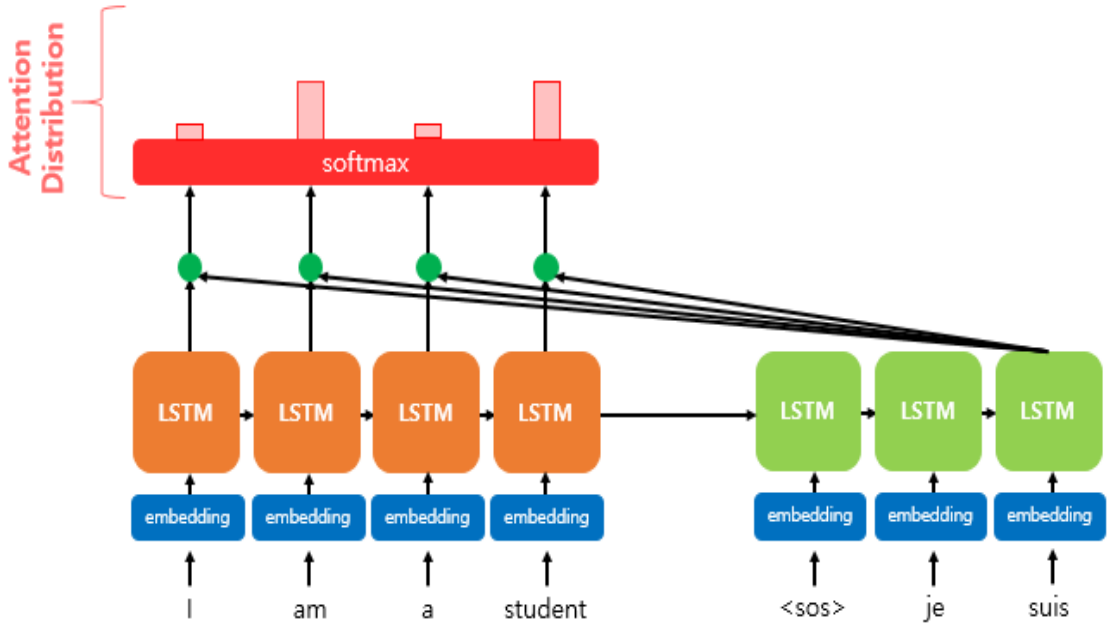
- 닷-프로덕트 어텐션(Dot-Product Attention)



Attention

- 어텐션 스코어(Attention Score)를 구하고 어텐션 분포(Attention Distribution)를 구한다



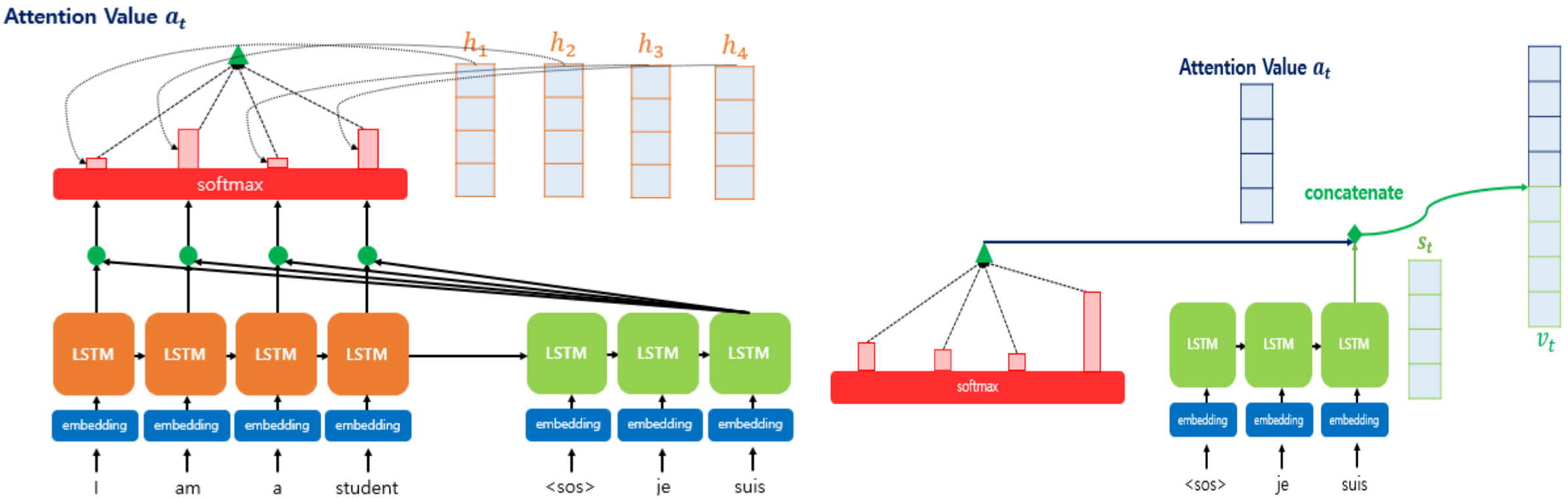


Attention

- 디코더의 셀은 두 개의 입력값을 필요로 하는데, 바로 이전 시점인 t-1의 은닉 상태와 이전 시점 t-1에 나온 출력 단어를 사용
- 어텐션 메커니즘에서는 출력 단어 예측에 또 다른 값인어텐션 값(Attention Value)이라는 새로운 값을 사용 . t번째 단어를 예측하기 위한 어텐션 값을 a_t라고 정의
- 어텐션 스코어(Attention Score)를 구하기 위해서 . S^T_t 와 h_i 를 내적한 것들의 모음을 e^t라고 정의
- 어텐션 스코어 란 현재 디코더의 시점 a_t 에서 단어를 예측하기 위해, 인코더의 모든 은닉 상태 각각이 디코더의 현 시점의 은닉 상태 s_t와 얼마나 유사한지를 판단하는 스코어값 디코더의 시점 t에서의 어텐션 가중치의 모음값인 어텐션 분포를 알파α^t이라고 할 때 $\alpha^t = \text{softmax}(e^t)$

Attention

- 어텐션 값(Attention Value)을 구하고 어텐션 값과 디코더의 t 시점의 은닉 상태를 연결한다.(Concatenate)



Attention

각 인코더의 은닉 상태와 어텐션 가중치 값들을 곱하고, 최종적으로 모두 더한다.

요약하면 가중합(Weighted Sum)을 진행.

어텐션 함수의 출력값인 어텐션 값(Attention Value) α_t 에 대한 식

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

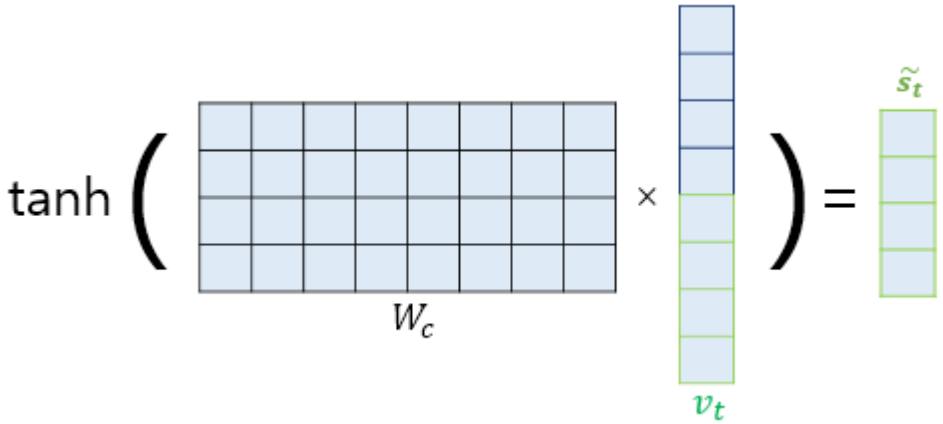
이러한 어텐션 값 α^t 는 종종 인코더의 문맥을 포함하고 있다고하여, 컨텍스트 벡터(context vector)라고도 불린다

Attention

결과적으로 a_t 와 s_t 를 결합 (concatenate)해서 하나의 벡터로 만든다 이를 v_t라고 정의

추가적으로 는 v_t 를 바로 출력층으로 보내기 전에 신경망 연산을 한 번 더 추가.

가중치 행렬과 곱한 후에 하이퍼볼릭탄젠트 함수를 지나도록 하여 출력층 연산을 위한 새로운 벡터인 s_t~~를 얻는다. 어텐션 메커니즘을 사용하지 않는 seq2seq에서는 출력층의 입력이 t시점의 은닉 상태인 s_t였던 반면, 어텐션 메커니즘에서는 출력층의 입력이 s_t~~가 된다.



$$\tilde{s}_t = \tanh(\mathbf{W}_c[a_t; s_t] + b_c)$$

introduction

This inherently sequential nature precludes parallelization within training examples, which becomes critical at longer sequence lengths, as memory constraints limit batching across examples

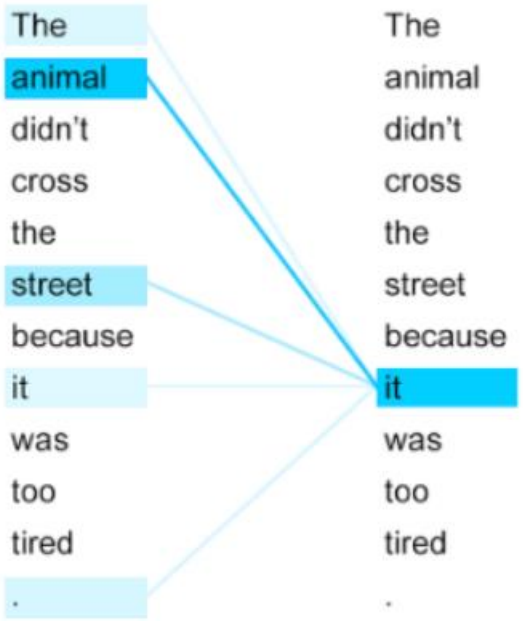
In this work we propose the Transformer, a model architecture eschewing recurrence and instead relying entirely on an attention mechanism to draw global dependencies between input and output.

- 기존의 attention mechanis은 Rnn과 함께 이용되어 왔지만 , 이 연구에서는 순환을 베제하고 입력과 출력에서 attention mechanis만 사용하는 Transfomer를 제안함

Background

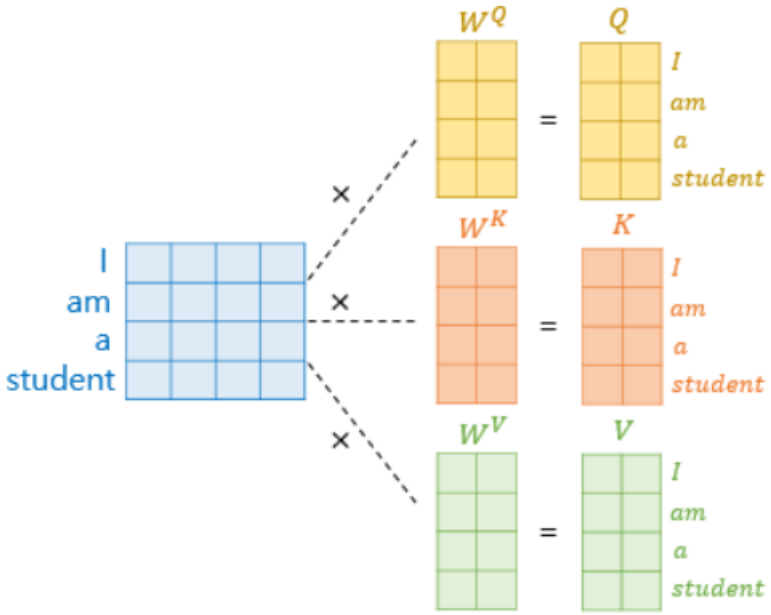
- Self attention

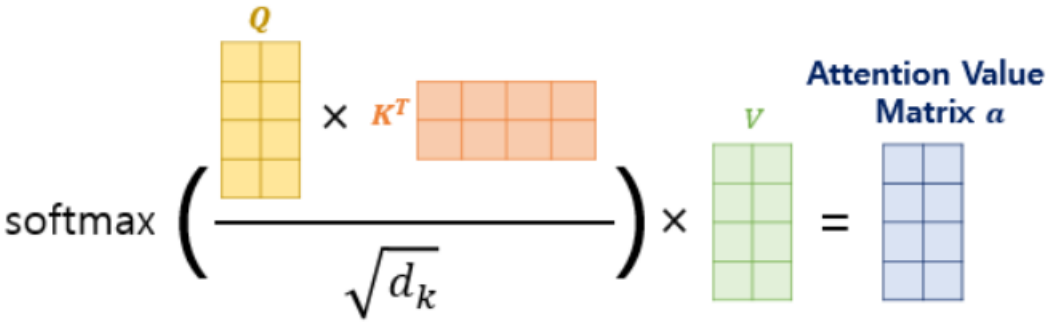
- Self attetntion을 통해 입력 문장내의 단어들끼리의 유사도를 찾을 수 있음



Background

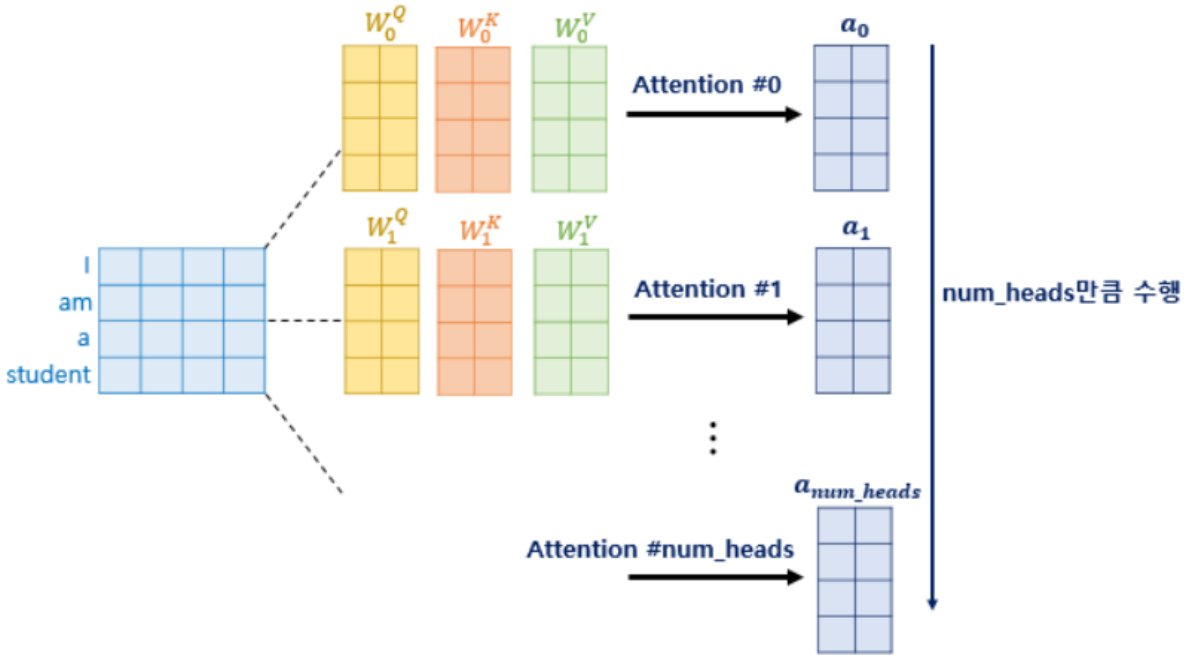
- Self attention

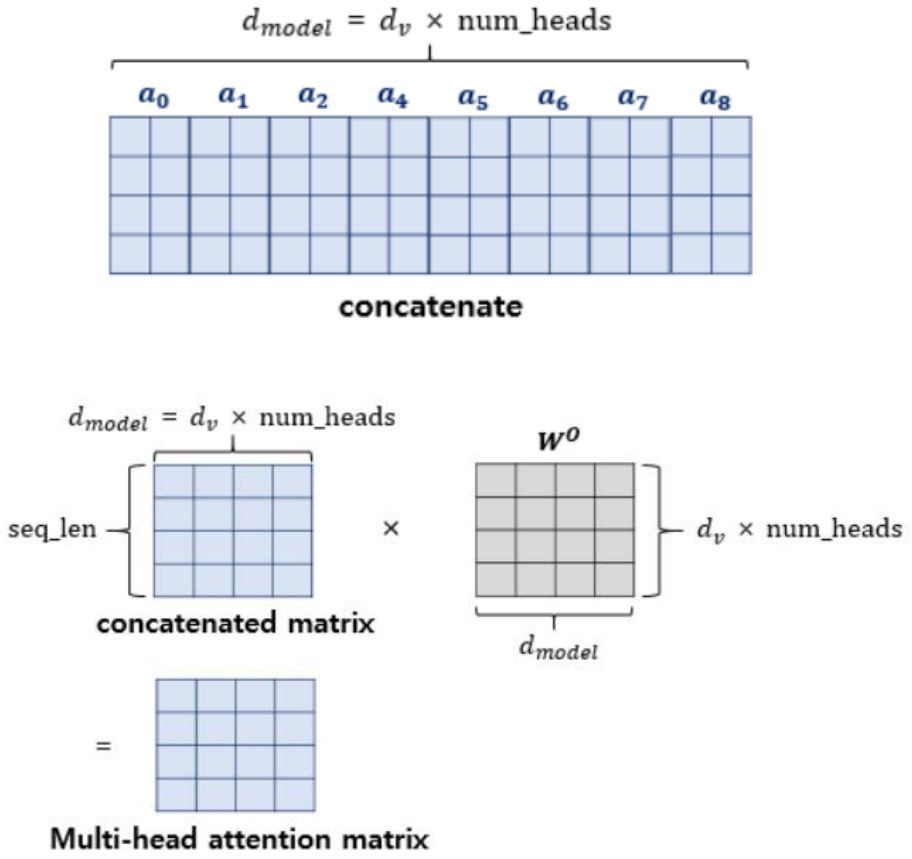




Background

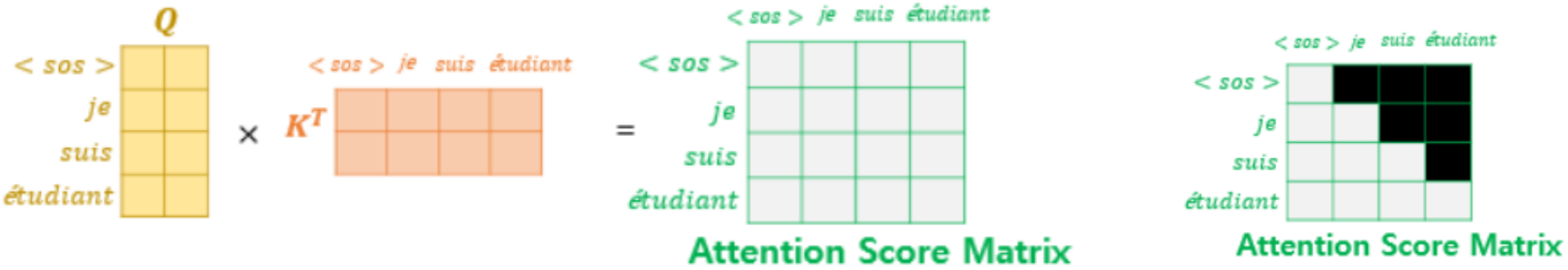
- Multi head attetntion
- 한번의 attetntion을 하는것보다 여러 번의 어탠션을 병렬로 사용하는 것을 효율적이라고 판단





Background

- Look ahead mask
- 마스크 값을 음수 무한의 값으로 넣어 softmax 함수의 출력값이 0%에 가까워지도록 함

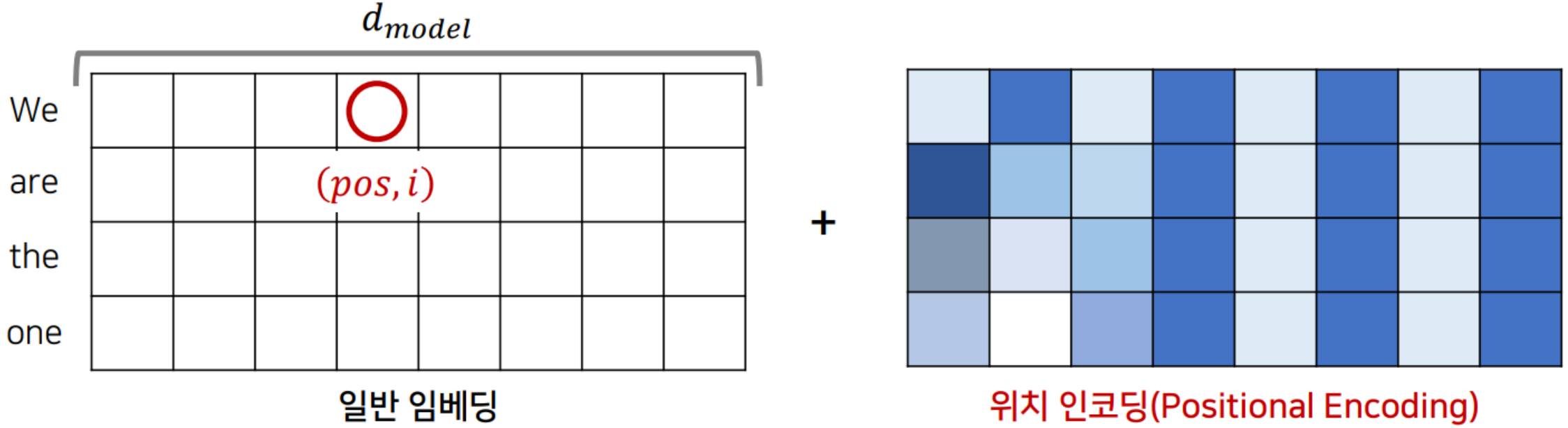


Background

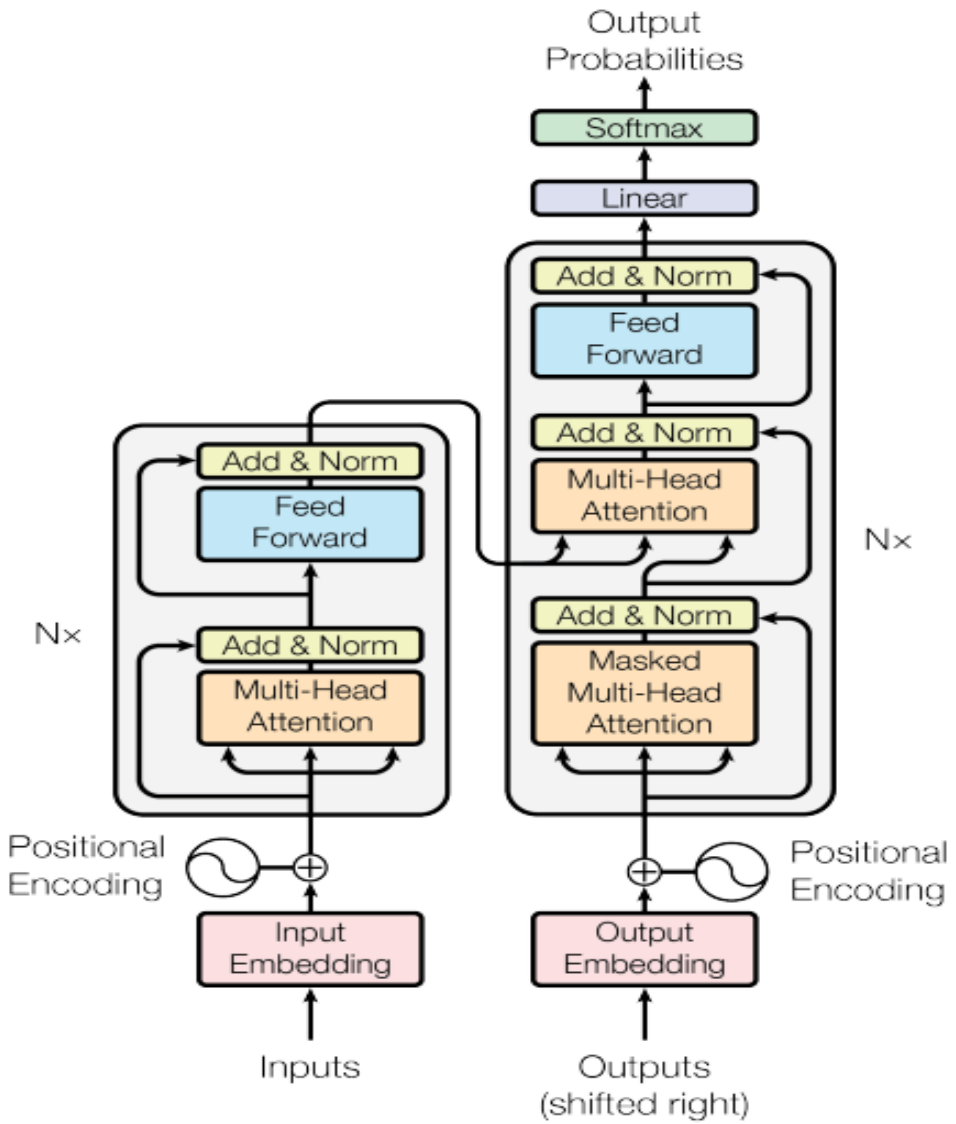
- Postitional Encoding

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

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

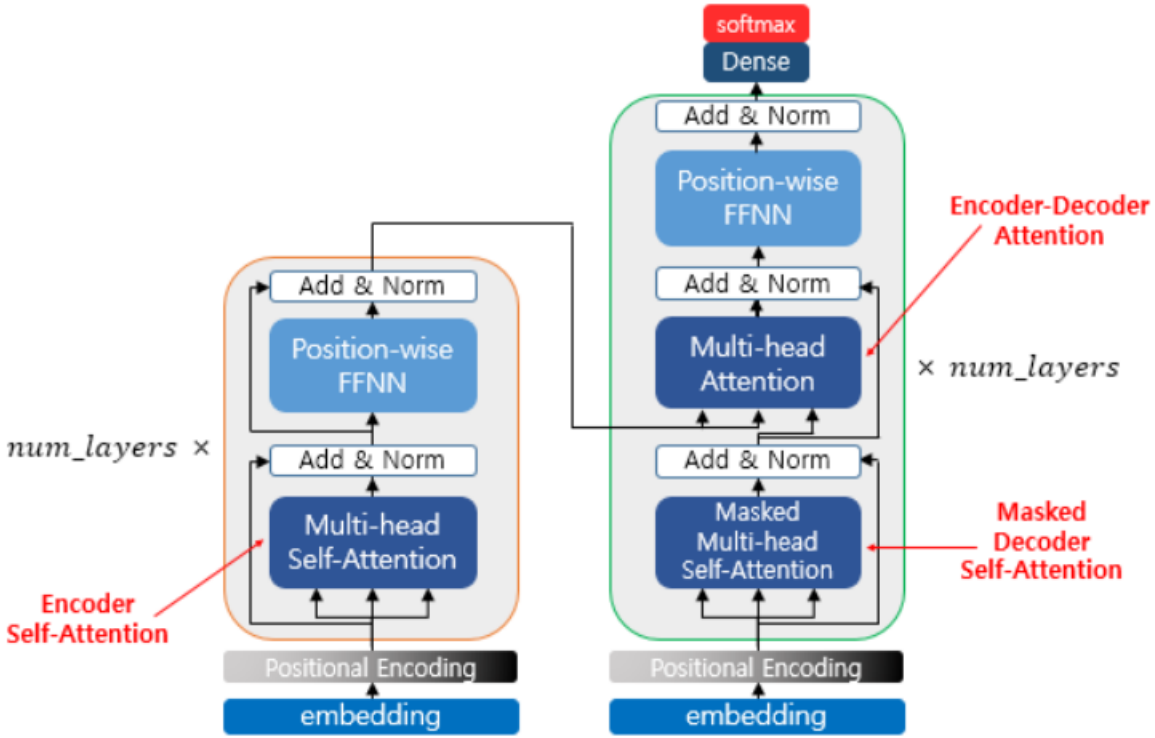


Model Architecture



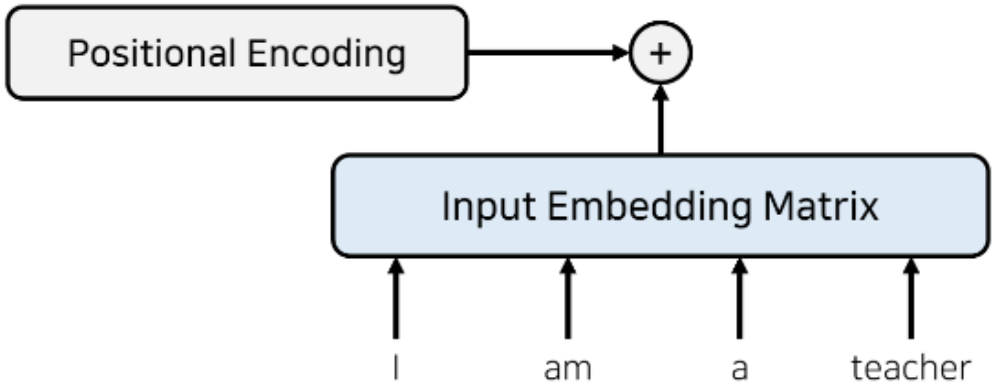
Model Architecture

- Transfomer에서는 총 3가지의 attention이 사용됨
- 인코더의 self attention : query = key = value
- 디코더의 masked self attention : query = key = value
- 디코더의 encoder – decoder attention: query :디코더 벡터 /key ,value = 인코더 벡터



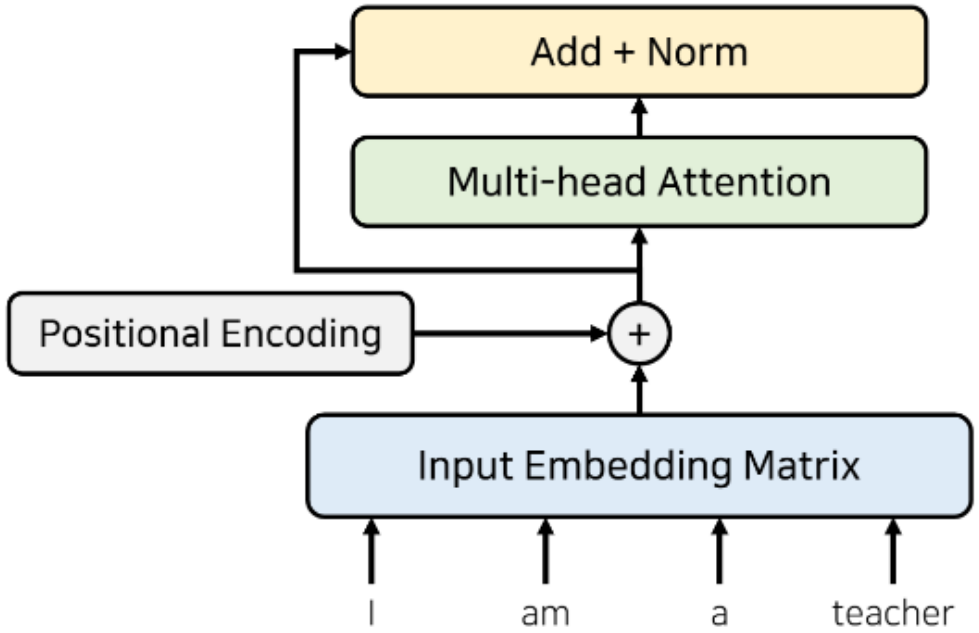
Model Architecture

- Encoder
- Rnn을 사용하지 않기에 임베딩과정에서 위치정보가 포함되게 하기 위해 postitional Encoding을 사용함



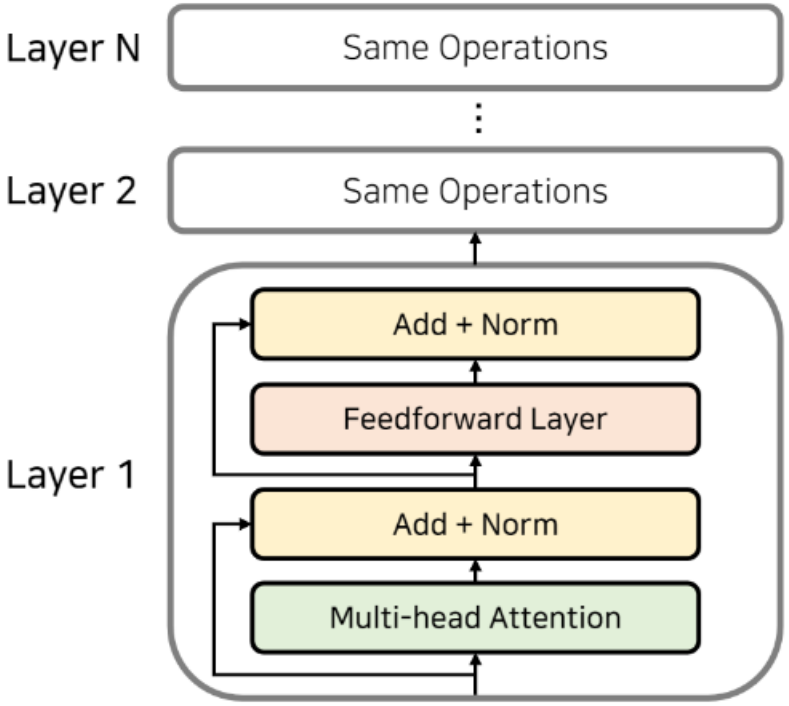
Model Architecture

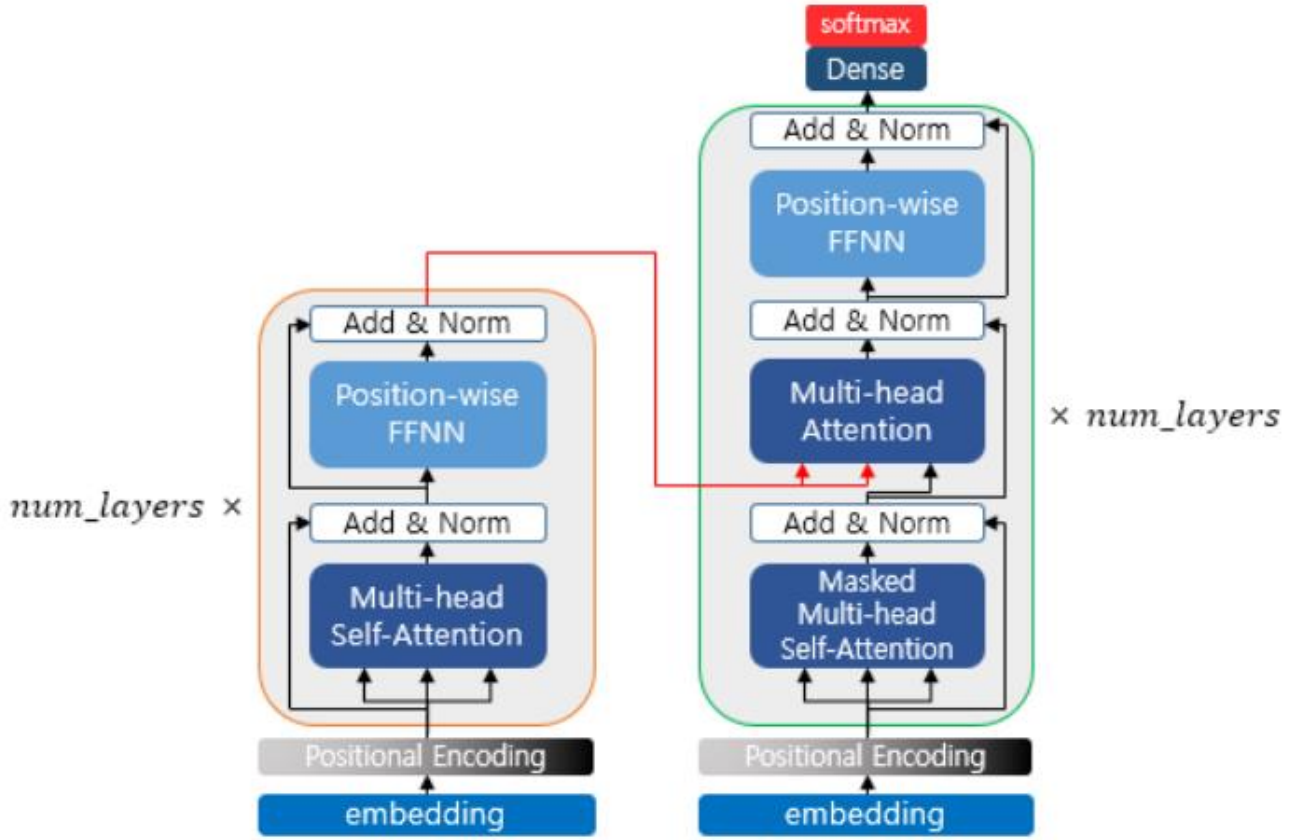
- 임베딩을 진행한 후 attention을 진행함
- 그 후 잔여학습 (Residual learning)을 진행함
- 잔여학습 : 이전의 특정 layer를 뛰어넘어서 복사된 값을 그대로 넣어줌 으로 전체 네트워크는 기존네트워크의 정보를 받으면서 추가적으로 잔여된 부분만 학습하게 만든다.



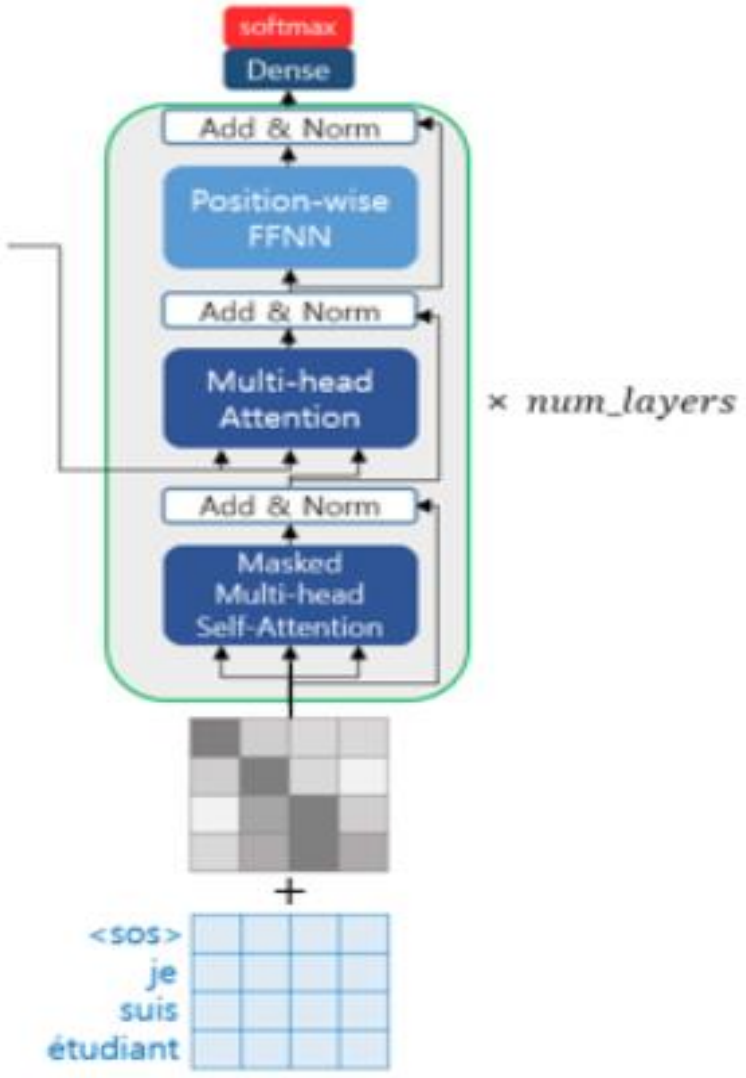
Model Architecture

- 이후에 어텐션과 정규화 과정을 반복





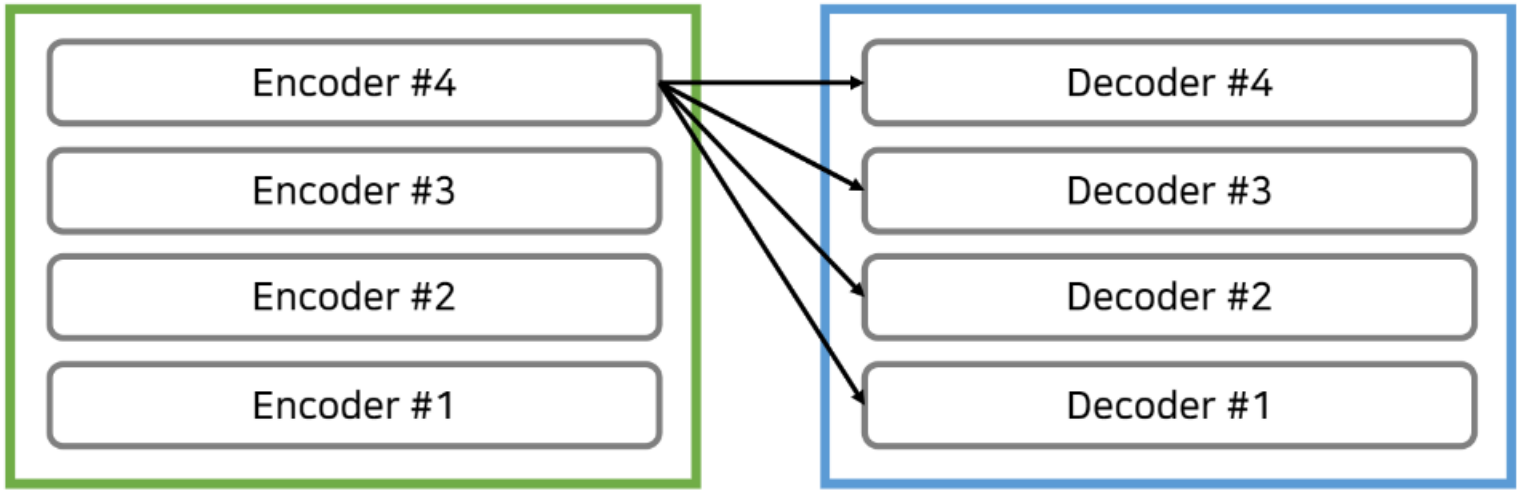
Model Architecture



- seq2seq의 디코더에 사용되는 RNN 계열의 신경망은 입력 단어를 매 시점마다 순차적으로 입력 받으므로 다음 단어 예측에 현재 시점을 포함한 이전 시점에 입력된 단어들만 참고할 수 있다.
- 반면, 트랜스포머는 문장 행렬로 입력을 한 번에 받으므로 현재 시점의 단어를 예측하고자 할 때, 입력 문장 행렬로부터 미래 시점의 단어까지도 참고할 수 있는 현상이 발생한다

Model Architecture

Multihead Attetntion에서는 마지막 encoder의 layer 의 출력이 모든 디코더 레이어에 입력된다



Why self attention

- 각 layer마다 계산 복잡도 줄어듬 , recurrent 없애서 병렬처리 가능
- Long range dependency를 줄일 수 있음
- N 은 sequecnce 길이
- D는 represent 차원 n이 d보다 작을때가 더 많음

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

Training

- Hyper parameters

- 임베딩 차원 , 인코더와 디코더에서 정해진 입력과 출력의 크기 의미 $d_{model} = 512$
- 인코더와 디코더의 층 $\text{num\_layers} = 6$
- 어텐션을 사용할 때, 한 번 하는 것 보다 여러 개로 분할해서 병렬로 어텐션을 수행하고 결과값을 다시 하나로 합침 이때 병렬의 개수 $\text{num\_heads} = 8$

Training

- Training Data and Batching

WMT 2014 English-German dataset

English-French, we used the significantly larger WMT 2014 English-French dataset

- Hardware and Schedule

one machine with 8 NVIDIA P100 GPUs

each training step took about 0.4 seconds

trained the base models for a total of 100,000 steps or 12 hours

big models, step time was 1.0 seconds ,trained for 300,000 steps (3.5 days)

Training

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [15]	23.75			
Deep-Att + PosUnk [32]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [31]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [8]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [26]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [32]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [31]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [8]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	**$3.3 \cdot 10^{18}$**	
Transformer (big)	**28.4**	**41.0**	$2.3 \cdot 10^{19}$	

Conclusion

- Transformer를 제안,
- 이는 attention에 완전히 기반한 첫 번째 시퀀스 변환 모델로, 기존의 Encoder-Decoder 아키텍처에서 주로 사용되던 recurrent 레이어를 다중 헤드 self-attention으로 대체함
- Transformer가 recurrent 또는 convolutional 레이어를 기반으로 한 아키텍처보다 훨씬 빠르게 훈련되고 WMT 2014 영어-독일어 및 WMT 2014 영어-프랑스어 번역 작업에서 state of the art를 달성