Week3_예습과제_김도희

Attention Is All You Need

> 💡 We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions
> entirely.

1. Introduction

- 이전

  RNN, LSTM, gated RNN등 → Sequence modeling 에 많이 사용

  Attention → input이나 output seq의 길이에 영향을 받지 않음

- 현재

  Transformer 모델은 이전에 많이 사용되던 RNN이나 LSTM 모델과 달리, **순차적인 처리** 없이 전체 시퀀스를 한 번에 처리할 수 있는 방식으로 설계되었다.

2. Background

- Reducing sequential computation

- Self attention

  입력 시퀀스의 각 요소가 다른 요소들과 상호작용하면서, 해당 시점에서 가장 관련성 높은 정보를 동적으로 선택하는 방식

- End to end memory network

  LSTM이나 RNN 기반 모델들은 순차적인 정보 처리에 강점이 있지만, 장기 의존성(long-term dependencies) 문제를 완벽하게 해결하지는 못했다.

  ⇒ 기억을 사용하여 추론과 Q&A 작업에서 더 나은 성능을 목표로 하는 신경망 모델

3. Model Architecture

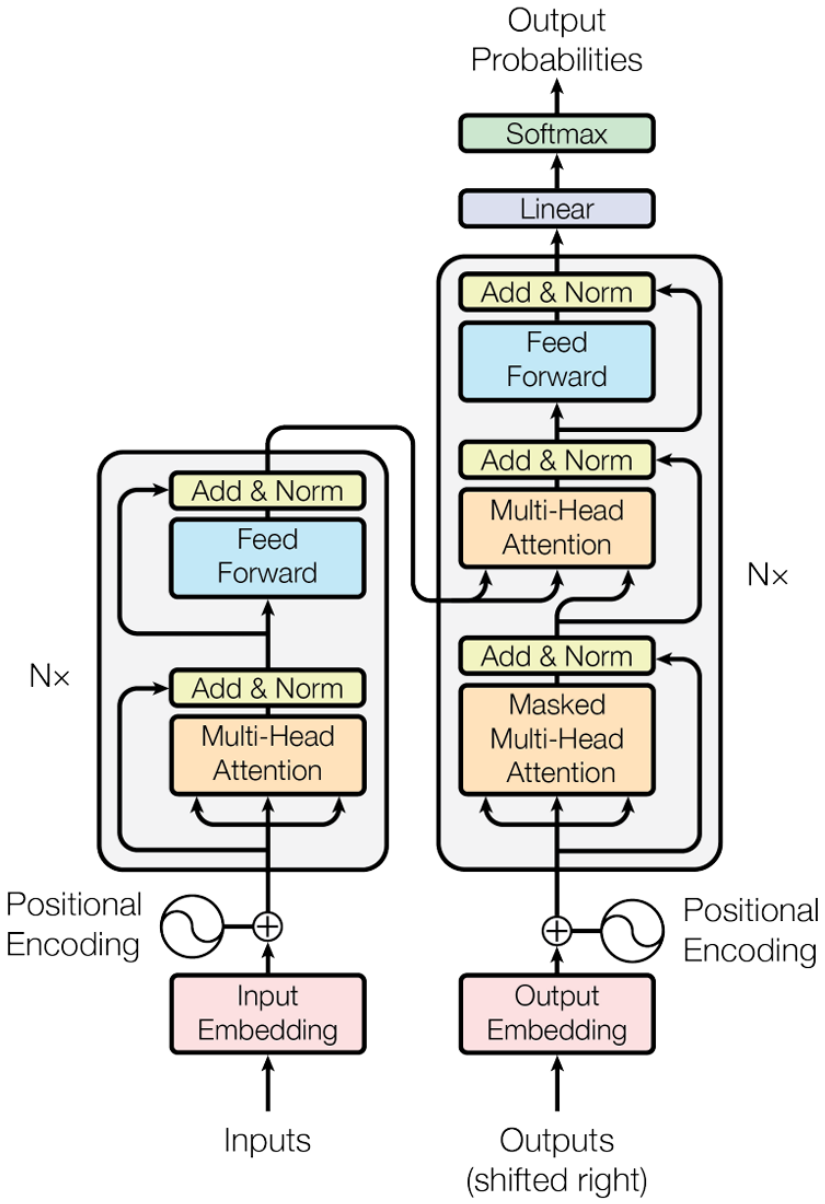

Figure 1: The Transformer - model architecture.

$(x_1, x_2, ..., x_n)$ ⇒ [Encoder] ⇒ $(z_1, z_2, ..., z_n)$ ⇒ [Decoder] ⇒ $(y_1, y_2, ..., y_n)$

3.1 Encoder and Decoder Stacks

- Encoder(x 6)
  - multi-head self attention mechanism
  - simple, positionwise fully connected feed forward network

  ⇒ 두개의 sublayer에 residual connection을 한 후 layer normalization 수행
- Decoder(x 6)
  - multi-head self attention mechanism
  - simple, positionwise fully connected feed forward network
  - Masked multi-head self attention

3.2 Attention

query와 key-value를 output과 weighted sum으로 mapping

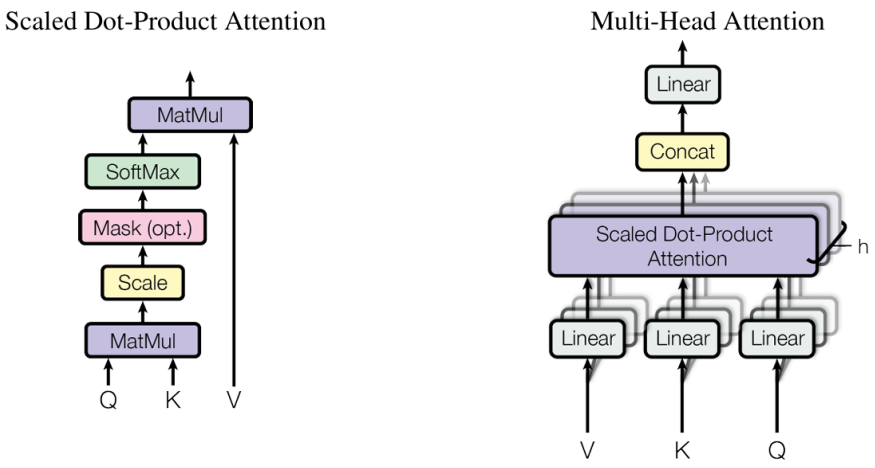

Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

- **Scaled Dot Product Attention**
  - MatMul: query 행렬 Q과 key 행렬 K의 내적
  - Scale: $\sqrt{d_k}$로 나눠줌으로써 스케일링

- Mask(opt.): Q와 K의 내적을 통해 구한 attention score matrix에 masking → 첫 번째 decoder에서만
- Softmax: 마지막 차원인 d_model을 따라 softmax function을 취하여 확률로 만들어준다.
- MatMul: value 행렬 V와 가중평균

- **Multi-Head Attention**

  multi-head attention은 단어 벡터가 dmodel/h 차원으로 나뉘어 단어 벡터의 특성이 평균내어지는 것을 막고 다양한 단어 벡터 표현을 유지할 수 있게 해준다.

3.3 Position-wise Feed-Forward Networks

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

Transformer의 모든 encoder block과 decoder block의 마지막 부분에 FFNN이 있다. 이는 Transformer에 ReLu를 이용하여non-linearity를 더해준다.

3.4 Embedding and Softmax

input token과 output token을 dimension d_{model}로 바꾸기 위해 학습된 embedding을 사용하였다.

다음 token의 확률을 예측하기 위해 학습된 선형변환과 softmax 함수를 사용했다.

3.5 Positional Encoding

Seq에서 token에 대한 상대적이거나 절대적인 정보를 제공해야만 했다. 이를 위해 Positional Encoding을 도입하였다.

4. Why self Attention

encoder-decoder attention에서는 인코더에서 디코더로 넘어오는 정보에 가중치를 부여하는 방식으로 작동하면서 hidden state를 사용한다. 반면, self attention은 입력되는 텍스트

데이터 내의 단어들끼리 서로의 간의 관계를 파악하기 위해서 입력되는 단어의 임베딩 벡터를 사용한다. 계산 복잡도 측면에서 더 빠르다.

Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

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)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

5. Training

5.1 Training Data and Batching

- Dataset
  - standard WMT 2014 English-German dataset(4.5M)
  - WMT 2014 English-French dataset(36M)
- Batching
  - 25000 source tokens and 25000 target tokens

5.2 Hardware and Schedule

8 NVIDIA P100 GPU, 0.4sec/training loop

5.3 Optimizer

Adam optimizer with β1 = 0.9, β2 = 0.98 and ϵ = 10−9

$$lrate = d_{\text{model}}^{-0.5} \cdot \min(step\_num^{-0.5}, step\_num \cdot warmup\_steps^{-1.5}) \quad (3)$$

5.4 Regularization

- Residual Dropout

- Label smoothing

6. Results

6.1 Machine Translation

Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	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**	**41.8**	$2.3 \cdot 10^{19}$	

English-German과 English-French 모두에서 training cost는 기존 모델들에 비해 훨씬 적지만, score는 더 높아진 결과를 확인할 수 있다.

6.2 Model Variaiton

각 요소들의 중요도를 평가하기 위해 수행

Table 3: Variations on the Transformer architecture. Unlisted values are identical to those of the base model. All metrics are on the English-to-German translation development set, newstest2013. Listed perplexities are per-wordpiece, according to our byte-pair encoding, and should not be compared to per-word perplexities.

	N	d_{model}	d_{ff}	h	d_k	d_v	P_{drop}	ϵ_{ls}	train steps	PPL (dev)	BLEU (dev)	params $\times 10^6$
base	6	512	2048	8	64	64	0.1	0.1	100K	4.92	25.8	65
(A)				1	512	512				5.29	24.9	
				4	128	128				5.00	25.5	
				16	32	32				4.91	25.8	
				32	16	16				5.01	25.4	
(B)					16					5.16	25.1	58
					32					5.01	25.4	60
(C)	2									6.11	23.7	36
	4									5.19	25.3	50
	8									4.88	25.5	80
		256			32	32				5.75	24.5	28
		1024			128	128				4.66	26.0	168
			1024							5.12	25.4	53
			4096							4.75	26.2	90
(D)							0.0			5.77	24.6	
							0.2			4.95	25.5	
								0.0		4.67	25.3	
								0.2		5.47	25.7	
(E)	positional embedding instead of sinusoids									4.92	25.7	
big	6	1024	4096	16			0.3		300K	**4.33**	**26.4**	213

6.3 English Constituency Parsing

영어 구조를 이해하고 정보를 추출하는 task에서도 좋은 성능을 보여줌.

Table 4: The Transformer generalizes well to English constituency parsing (Results are on Section 23 of WSJ)

Parser	Training	WSJ 23 F1
Vinyals & Kaiser el al. (2014) [37]	WSJ only, discriminative	88.3
Petrov et al. (2006) [29]	WSJ only, discriminative	90.4
Zhu et al. (2013) [40]	WSJ only, discriminative	90.4
Dyer et al. (2016) [8]	WSJ only, discriminative	91.7
Transformer (4 layers)	WSJ only, discriminative	91.3
Zhu et al. (2013) [40]	semi-supervised	91.3
Huang & Harper (2009) [14]	semi-supervised	91.3
McClosky et al. (2006) [26]	semi-supervised	92.1
Vinyals & Kaiser el al. (2014) [37]	semi-supervised	92.1
Transformer (4 layers)	semi-supervised	92.7
Luong et al. (2015) [23]	multi-task	93.0
Dyer et al. (2016) [8]	generative	93.3

7. Conclusion

전적으로 Attention에 기반한 Transformer를 제안하였다. 이는 기존에 사용하던, encoder-decoder architecture를 대체할 수 있다. 속도와 성능 측면에서 모두 이전의 구조보다 더 뛰어나다.