Attention is all you need

1. Introduction

기존 시퀀스 모델들은 주로 RNN, LSTM 같은 순환 구조에 의존하고 있으며, 이로 인해 계산의 병렬화가 어렵고 긴 거리의 의존성을 학습하는 데 한계가 있다. 최근에는 attention 기법을 보조적으로 사용해 이런 문제를 일부 해결해 왔지만, 이 논문은 한 걸음 더 나아가 recurrence 없이 self-attention만으로 시퀀스를 처리하는 완전한 구조인 Transformer를 제안한다. 이 모델은 병렬화가 용이하고 학습 속도가 빠르며, WMT 2014 영어-독일어 번역에서 SOTA 성능을 달성했다는 점에서 강력한 효율성과 확장성을 동시에 입증한다.

2. Background

초기 seq2seq 모델은 RNN 기반으로 입력 시퀀스를 하나의 고정된 벡터로 요약한 후, decoder가 이를 기반으로 출력을 생성했는데, 이 방식은 정보 손실의 위험이 크고 긴 문장일수록 성능이 급감했다. 이에 대한 해결책으로 제시된 것이 attention mechanism이다.

대표적으로 Bahdanau et al. (2015)의 방식은 decoder가 인코더의 모든 hidden state에 동적으로 접근하게 하여 context 정보를 효과적으로 활용할 수 있도록 했다. 이러한 attention 개념을 한 단계 더 확장한 것이 self-attention, 즉 입력 시퀀스 내 단어들 간의 상호작용을 직접 모델링하는 방식이다. 논문은 self-attention의 기본 연산 수식을 소개하며, query, key, value를 활용해 입력 간 유사도를 계산하고, 이 가중치 기반으로 정보의 흐름을 조절하는 방식을 설명한다. 이 구조는 RNN 없이도 입력 간 관계를 포착할 수 있어 Transformer의 핵심 설계로 채택된다.

3. Model Architecture

Transformer는 기존의 RNN이나 CNN을 사용하는 시퀀스 모델들과 달리, 오직 attention 메커니즘만을 사용해 전체 구조를 설계한 모델이다. 일반적인 sequence-to-sequence 구조처럼 encoder와 decoder로 구성되며, 각각은 self-attention과 feed-forward layer를 반복적으로 쌓은 형태다. 이 구조는 학습 속도를 크게 향상시키고 병렬 처리에 유리하다는 장점을 가진다.

3.1 Encoder and Decoder Stacks

Encoder는 총 6개의 동일한 레이어로 구성되며, 각 레이어는 multi-head self-attention과 position-wise feed-forward 네트워크로 이루어진다. 각 서브레이어에는 residual connection과 layer normalization이 적용되어 학습 안정성을 높인다. Decoder 역시 6개의 동일한 레이어로 구성되지만, encoder 출력을 참고하는 encoder-decoder attention이 추가로 포함되며, future token을 보지 않도록 마스크된 self-attention이 사용된다. 이로 인해 decoder는 오토레그레시브 방식으로 동작할 수 있다.

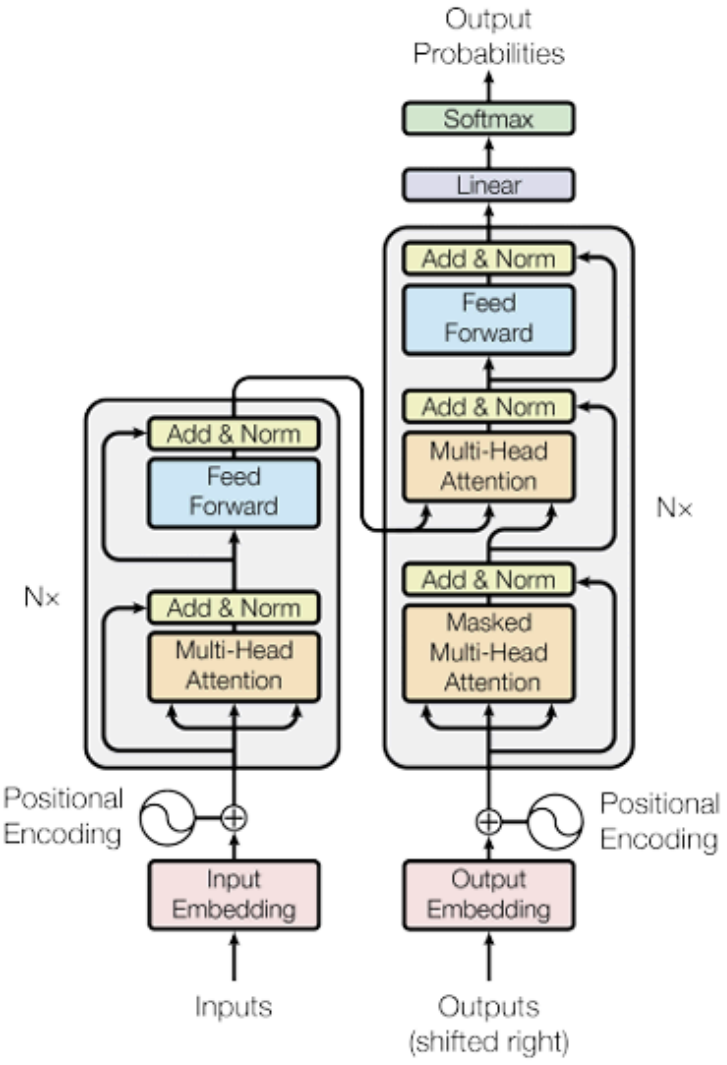


Figure 1: The Transformer - model architecture.

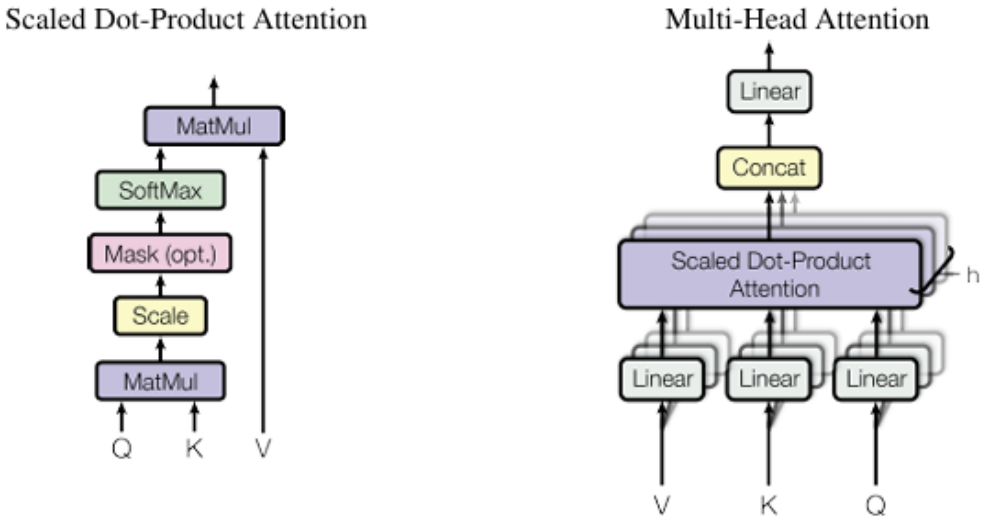
3.2 Attention

Attention은 query, key, value 세 요소를 기반으로 특정 단어가 다른 단어를 얼마나 주목해야 하는지를 계산하는 메커니즘이다. 이 모델에서는 attention을 다양한 형태로 반복적으

로 사용하여 문장 내 의미 관계를 정교하게 학습한다.

3.2.1 Scaled Dot-Product Attention

Scaled Dot-Product Attention은 query와 key의 내적값을 √dk로 나눈 뒤 softmax를 적용하여 value에 대한 가중합을 계산하는 방식이다. dk가 커질수록 내적값이 커지고 softmax의 gradient가 작아지는 현상을 방지하기 위해 스케일링이 도입되었다. 이 방식은 행렬 곱 연산을 통해 매우 빠르게 구현할 수 있으며, 기존의 additive attention보다 계산적으로 효율적이다.



3.2.2 Multi-Head Attention

Multi-head attention은 attention 연산을 한 번만 하는 대신, 여러 개의 head로 나누어 병렬로 수행한 뒤 결과를 합치는 방식이다. 각 head는 서로 다른 하위 표현 공간에서 정보를 추출하기 때문에, 다양한 관계를 동시에 고려할 수 있다. 논문에서는 총 8개의 head를 사용하였고, 각 head의 차원은 64로 설정되어 전체 차원인 512와 일치한다.

3.2.3 Applications of Attention in Our Model

Transformer는 attention을 총 세 가지 방식으로 사용한다. 첫째, encoder 내부에서는 self-attention을 통해 입력 문장 내 단어들 간의 관계를 학습한다. 둘째, decoder 내부에서도 self-attention이 사용되는데, 이때는 미래 단어를 참조하지 않도록 마스크가 적용된다. 셋째, encoder-decoder attention을 통해 decoder가 입력 문장의 정보를 직접 참고할 수 있게 하여, 번역 등의 시퀀스 생성에 필요한 문맥 정보를 전달받을 수 있도록 한다.

3.3 Position-wise Feed-Forward Networks

각 encoder와 decoder 레이어는 attention 이후에 feed-forward 네트워크를 포함한다. 이 네트워크는 모든 위치에 동일하게 적용되며, 두 개의 선형 변환과 그 사이의 ReLU 활성화 함수로 구성된다. 입력 차원과 출력 차원은 512, 내부 hidden 차원은 2048로 설정되어 있다. 이 구조는 사실상 1×1 convolution과 유사한 연산이며, attention 연산 이후 개별 위치별로 비선형 변환을 수행하는 역할을 한다.

3.4 Embeddings and Softmax

입력과 출력 토큰은 모두 512차원의 임베딩 벡터로 변환되며, decoder의 출력은 softmax 함수를 통해 다음 토큰의 확률로 바뀐다. 이때 입력 임베딩과 출력 임베딩, 그리고 pre-softmax linear projection에 동일한 가중치 행렬을 공유하도록 설계되어 파라미터 수를 줄이고 일관된 표현을 유지한다. 또한 임베딩 값은 √dmodel로 스케일링되어 모델 학습의 안정성을 높인다.

3.5 Positional Encoding

Transformer는 RNN이나 CNN처럼 순서 정보가 내장된 구조가 아니기 때문에, 입력 순서를 모델에 알려주기 위해 positional encoding을 추가로 도입한다. 이 논문에서는 sin과 cos 함수를 이용해 각 위치에 고유한 주기적 값을 부여하고, 이를 임베딩 벡터에 더하는 방식으로 위치 정보를 전달한다. 이러한 방식은 학습된 positional embedding과 성능이 비슷했으며, 고정된 수식 기반이므로 길이가 더 긴 문장에도 일반화될 수 있다는 장점이 있다.

4. Why Self-Attention

Transformer에 적용된 Self-Attention은 기존 recurrent, convolutional 레이어가 가진 한계를 극복하고 모델의 성능 향상시키기에 사용되었다. Self-attention의 장점에 3가지 관점이 있다.

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

1. Computational Complexity

시퀀스 길이(n)이 표현 차원(d)보다 작은 경우(대부분의 최신 기계 번역모델)에 self-attention 레이어는 rcurrent 레이어보다 계산적으로 더 빠름.

Self-Attention의 계층당 복잡도는 $O(n^2 \cdot d)$입니다. 이는 시퀀스 길이가 길어질수록 2차적으로 증가하지만, '제한된(restricted) self-attention'을 통해 $O(r \cdot n \cdot d)$로 줄여 매우 긴 시퀀스에서도 효율성을 유지할 수 있다. 여기서 r은 고려하는 이웃의 크기

2. Parallelization

   Self-attention은 모든 위치의 계산을 병렬로 수행할 수 있어 minimum sequential operations 가 $O(1)$으로 매우 효율적이다.

   Convolution 모델도 $O(1)$의 순차작업으로 병렬화가 가능하다.

   반면 rucurrent 모델은 이전 시점 hidden state에 의존하는 특성에 의해 $O(n)$의 순차 작업이 필요하다.

3. Learning Long-range Dependencies

   입력-출력 시퀀스의 어떤 두 위치 사이의 **최대 경로 길이**가 짧을 수록 Long range Dependency를 더 잘 학습이 가능

   Self-attention은 최대 경로 길이가 $O(1)$을 유지한다. 이는 먼 거리의 복잡한 의존성을 학습하는데 유리하다.

   recurrent 모델은 순차적으로 정보를 전달하기에 최대 경로 길이가 $O(n)$이 되어 긴 시퀀스에서 먼 의존성을 학습하기 어렵다.

   convolution 모델은 커널의 크기에 따라 (연속 커널 : $O(n/k)$ or dilated convolution : $O(log_k(n))$)의 최대 경로 길이를 가진다. 추가로 convolution 레이어는 하나의 레이어로 모든 Input, output의 위치를 연결하지 않기에 여러 레이어를 쌓아야 하므로 먼 의존성을 학습하기 어렵다.

5. Training

5.1 Training Data and Batching

- **훈련 데이터**: WMT 2014 English-German 데이터셋(450만 문장 쌍, 37000 토큰의 Byte-Pair Encoding 어휘 사용)과 WMT 2014 English-French 데이터셋(3600만 문장, 32000 토큰의 word-piece 어휘 사용)으로 훈련되었다.
- **배치**: 대략적인 시퀀스 길이에 따라 문장 쌍을 함께 묶었으며, 각 훈련 배치는 약 25000개의 소스 토큰과 25000개의 타겟 토큰으로 구성되었다.
- **5.2 Hardware and Schedule**
  - **하드웨어**: 8개의 NVIDIA P100 GPU를 사용하여 모델을 훈련했다.

- **훈련 시간**:
  - 기본 모델(Base model): 각 훈련 단계는 약 0.4초가 소요되었으며, 총 100,000단계(12시간) 동안 훈련되었다.
  - 대형 모델(Big model): 각 훈련 단계는 1.0초가 소요되었으며, 총 300,000단계(3.5일) 동안 훈련되었다.

5.3 Optimizer

- **최적화**: Adam optimizer 사용했으며, $\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 10^{-9}$로 설정되었다.
- **학습률**: 훈련 과정 동안 학습률을 변경했으며, 다음 공식에 따라 적용되었다:

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

  - `warmup_steps = 4000`으로 설정하여, 초기 `warmup_steps` 동안 학습률이 선형적으로 증가하고 그 후에는 스텝 수의 역제곱근에 비례하여 감소한다.

5.4 Regularization

- **Residual Dropout**: 각 서브 레이어의 출력에 Dropout이 적용되었으며, 임베딩과 Positional Encoding의 합에도 적용되었다. 기본 모델의 Dropout 비율은 $P_{drop} = 0.1$이었다.
- **Label Smoothing**: $\epsilon_{ls} = 0.1$의 Label Smoothing이 사용되었으며, 이는 Perplexity를 증가시키지만 정확도와 BLEU 점수를 향상시켰다.

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**	$2.3 \cdot 10^{19}$	

- **WMT 2014 English-German**
  - 기존 최고 성능 모델보다 BLEU 점수 2.0 이상 향상
  - Transformer (big) 모델: BLEU 28.4
  - 학습 시간: 3.5일 / 8 GPU(P100)
  - Base 모델도 모든 기존 단일 모델, ensemble을 능가
- **WMT 2014 English-French**
  - Transformer (big): BLEU 41.0 (새로운 단일 모델 최고점)

    (Table 2에는 41.8, 본문에서는 41.0)
  - 학습 비용: 기존 모델의 1/4 이하 FLOPs
- **Checkpoint averaging**
  - checkpoint 중 마지막 n개를 평균하여 최종 모델을 계산함
  - Beam Search: 번역 결과를 생성할 때 가능한 후보들을 여러 개 유지하며 탐색하는 기법.
    - 여기서는 beam size = 4 → 매 타임스텝마다 가장 가능성 높은 4개의 시퀀스를 유지
  - Length Penalty $\alpha = 0.6$
    - 너무 짧은 문장을 선호하지 않도록 점수에 길이에 따른 패널티를 줌
    - α 값이 클수록 긴 문장을 더 선호
  - 번역 생성 중 가능한 최대 길이는 입력 길이 + 50으로 제한
  - 번역이 빨리 끝날 수 있으면 조기 종료

6.2 Model variations

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

열 이름	의미
N	인코더 및 디코더의 층 수 (layer 수)
d_model	모델의 임베딩 차원 (토큰 표현 벡터 크기)
d_ff	피드포워드 네트워크의 내부 차원 (확장 후 줄이는 부분)
h	Multi-head attention의 head 수
d_k , d_v	각 attention head의 key와 value의 차원
P_drop	Dropout 비율 (regularization 용도)
ε_ls	Label smoothing 계수 (loss 안정화 목적)
train steps	훈련 iteration 수 (K = 1,000회)

- (A) Multi-head 수 및 Key/Value 차원 변화
  - parameter 수가 같을 때, head가 많아진다고 성능이 향상되지는 않음
  - 최적의 성능 : h=8, 16
- (B) Key 차원 변화
  - Key 차원의 감소는 성능 하락으로 이어짐
- (C) 모델 크기 변화

    - 큰 모델이 더 성능이 높음
- (D) Dropout 및 Label Smoothing 변화
    - Dropout은 overfitting을 피하는 데 도움이 됨
- (E) 위치 임베딩 변화
    - 거의 변화 없음

6.3 English Constituency Parsing

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

- 번역 외의 자연어처리 task 수행
- WSJ(Wall Street Journal) Penn Treebank
- Discriminative
    - 40K training 문장
    - 4-layer Transformer, d_{model}=1024
    - Transformer는 기존 모델들과 유사하거나 더 나은 성능을 보임
- Semi-supervised
    - 라벨이 없는 데이터도 일부 활용
    - SOTA 성능을 보임

작은 데이터셋에서, task에 특화된 tuning 없이도 범용적으로 높은 성능을 보임

7. Conclusion

Transformer는 최초로 recurrent layer를 사용하지 않고 attention만으로 모델을 구현함

기존 모델 대비 속도도 빠르고, WMT task에서 기존 모델들과 ensemble보다 더 높은 성능을 보임

모델 구조가 간단하고, 확장성이 좋아 추후 다른 modality로의 적용 가능성이 높음(image, audio, video 등)