Attention Is All You Need

1. Introduction

- RNN 계열(LSTM, GRU)은 **본질적으로 순차적 계산**을 요구하여 병렬화가 어려움
- 긴 시퀀스에서 **메모리 제약 문제**
- Attention은 성능을 크게 개선했으나, 대부분 **여전히 RNN 위에 결합된 보조 모듈**

→ Transformer는 순환 구조를 제거하고 **self-attention만으로 입력–출력 의존성을 모델링**

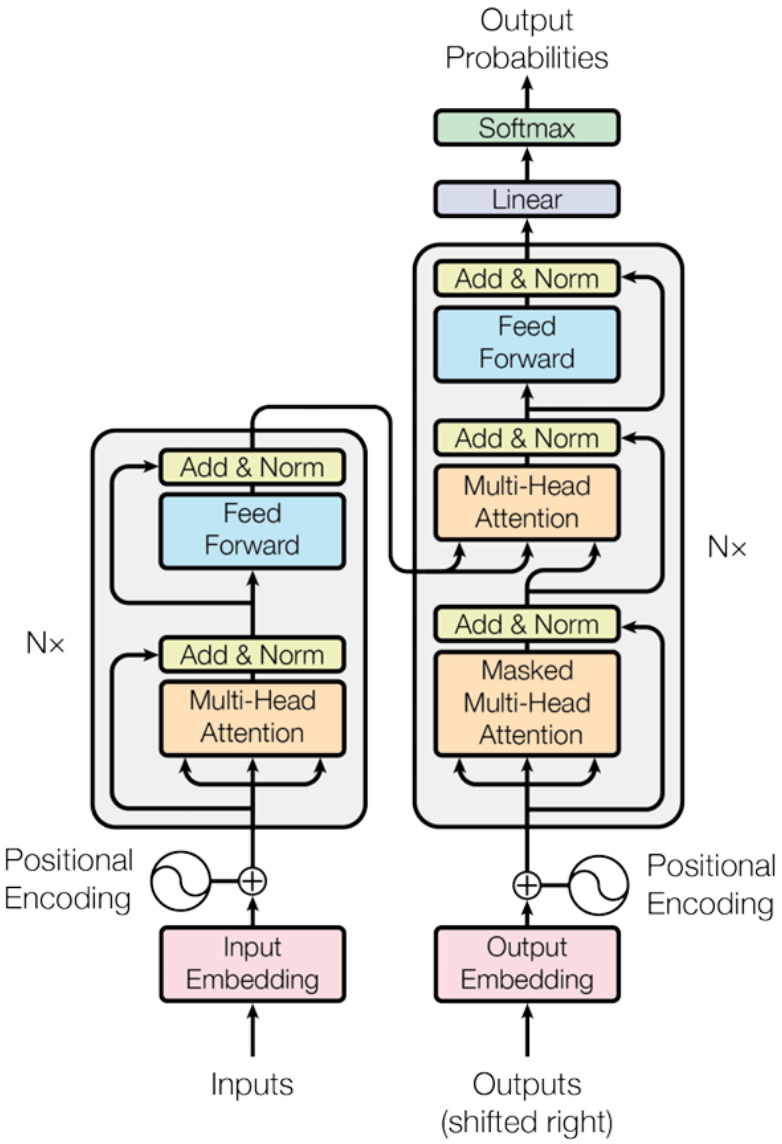
- 병렬 연산 효율 증가
- 학습 시간 단축
- 번역 성능 향상

2. Background

- CNN 기반 모델 (ByteNet, ConvS2S) 은 병렬화는 가능하지만,
  - 두 위치 간 의존성을 학습하기 위해 여러 층을 쌓아야 함 → **경로 길이 증가**
- Self-attention은 이전에도 QA, 요약, 문장 표현 학습 등에 적용된 바 있으나, Transformer에서 **이를 시퀀스 변환(transduction) 모델의 핵심 구조로 전면 채택**

3. Model Architecture

Transformer는 표준적인 **Encoder–Decoder 구조**를 따름



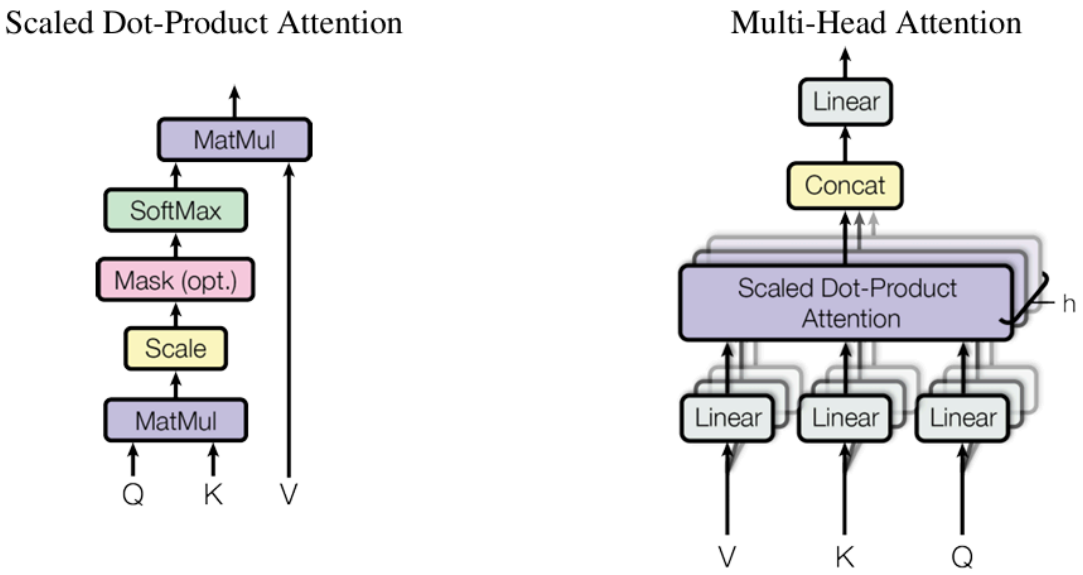
Encoder

- 6개의 동일한 layer로 구성
- 각 layer는 두 개의 sub-layer 포함:
  1. Multi-Head Self-Attention
  2. Position-wise Feed-Forward Network (FFN)
- 각 sub-layer마다 **Residual + LayerNorm** 적용
- 모든 표현 차원: (d_{model} = 512)

Decoder

- 6개의 layer로 구성

- Encoder의 두 sub-layer에 **Encoder–Decoder Attention** 추가
- **Masked Self-Attention** 적용 → 미래 정보가 유출되는 것을 방지 (autoregressive 조건 보장)
- Residual + LayerNorm 적용



Scaled Dot-Product Attention

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

- 스케일링 $(\sqrt{d_k})$은 softmax 포화 방지 및 안정적 학습을 위함
- Dot-product attention은 additive attention보다 **효율적이고 빠름**

Multi-Head Attention

- 하나의 큰 attention 대신, 서로 다른 선형 변환을 거친 **8개의 attention head**를 병렬 수행
- 각 head는 서로 다른 표현 공간을 학습하여 **다양한 관계를 포착**

- 총 계산량은 단일 head와 유사하도록 설계

Transformer에서의 Attention 활용

- **Encoder–Decoder Attention**: Decoder가 입력 전체를 참조
- **Encoder Self-Attention**: 각 위치가 다른 모든 위치와 상호작용
- **Decoder Masked Self-Attention**: 이전 위치만 참조하도록 제한

Position-wise Feed-Forward Networks

각 위치에 동일하게 적용

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

Positional Encoding

Transformer에는 순환 구조가 없으므로 **위치 정보 주입이 필요**.

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

- 상대 위치 관계를 선형 변환으로 표현 가능
- 학습 길이보다 긴 시퀀스에도 외삽 가능
- 학습형 위치 임베딩과 성능이 유사함을 실험적으로 확인

---

4. Advantage

1. **계산 복잡도**

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

2. **병렬화 가능성**
    - Self-attention은 병렬 연산 가능
    - RNN은 본질적으로 순차적
3. **장거리 의존성 경로 길이**
    - Self-attention: 어떤 두 위치 간에도 **상수 경로**
    - CNN: 여러 층이 필요
    - RNN: (O(n)) 단계 필요

→ Attention 분포가 해석 가능 (문법·의미 구조 반영)

---

5. Training

(1) Training Data & Batching

- WMT 2014 EN–DE (4.5M 문장), BPE 토큰화(37k 어휘)
- EN–FR는 더 큰 데이터셋 사용(36M 문장)
- 배치는 비슷한 길이 문장끼리 묶음

(2) Hardware & Schedule

- 다중 GPU 환경에서 학습
- Base 모델과 Big 모델을 구분하여 실험

(3) Optimizer

- Adam 사용
- Warmup 후 cosine-like decay 학습률 스케줄 적용

(4) Regularization

- Residual Dropout 적용
- **Label Smoothing** 사용 → BLEU 개선

---

6. Results

	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

(1) Machine Translation

- EN→DE, EN→FR 모두에서 SOTA
- **Transformer(Big)** 모델이 가장 높은 BLEU 달성
- Beam search + length penalty 사용

(2) Model Variations

- Attention head 수, key dimension, dropout, positional encoding 방식 변경 실험
  - 너무 적거나 많은 head는 성능 저하
  - 큰 모델일수록 성능 향상

- Dropout이 과적합 방지에 중요
- Sinusoidal vs Learned positional encoding 성능 유사

---

7. Conclusion

- Transformer는 **완전한 Attention 기반 모델**로 RNN을 대체할 수 있음을 입증
- 번역에서 SOTA 달성 및 높은 학습 효율 확보
- 향후 방향:
  - 이미지/음성/비디오 등 멀티모달 확장
  - 긴 시퀀스를 위한 제한적(local) attention 연구
  - 비순차적 생성 방식 탐구