⌨️

Transformer

1. Abstract

기존의 시퀀스 변환 모델들과 달리, 본 논문에서 제안하는 Transformer는 RNN이나 CNN을 사용하지 않고, 오직 Attention 메커니즘만을 활용하는 방식으로 설계되었다. 실험 결과, Transformer는 높은 성능을 유지하면서도 병렬 처리가 가능해 학습 속도가 기존 모델들보다 훨씬 빠르다는 것이 확인되었다.

2. Introduction

전통적인 순환 신경망(RNN, LSTM, GRU 등)은 입력 데이터를 순차적으로 처리하는 구조를 갖는다. 즉, 이전 시점의 상태(*h*t−1)를 이용해 현재 상태(*h*t)를 계산하는 방식이다. 이러한 특성으로 인해 RNN 기반 모델들은 병렬 처리가 어려워지고, 시퀀스 길이가 길어질수록 학습 속도가 저하되는 문제가 발생한다.

반면, Attention 메커니즘은 입력 시퀀스 내 단어 간 거리에 상관없이 상호 의존성을 효과적으로 학습할 수 있도록 해준다. 기존 연구들은 Attention을 RNN과 함께 사용했지만, Transformer는 RNN을 완전히 배제하고 Attention만으로 모델을 구성한 점에서 차별점을 가진다.

특히, Transformer는 기존 Attention과 달리 **Self-Attention** 메커니즘을 활용한다. 일반적인 Attention과 Self-Attention의 차이점은 이후에 보다 상세히 다룰 예정이다.

3. Model Architecture

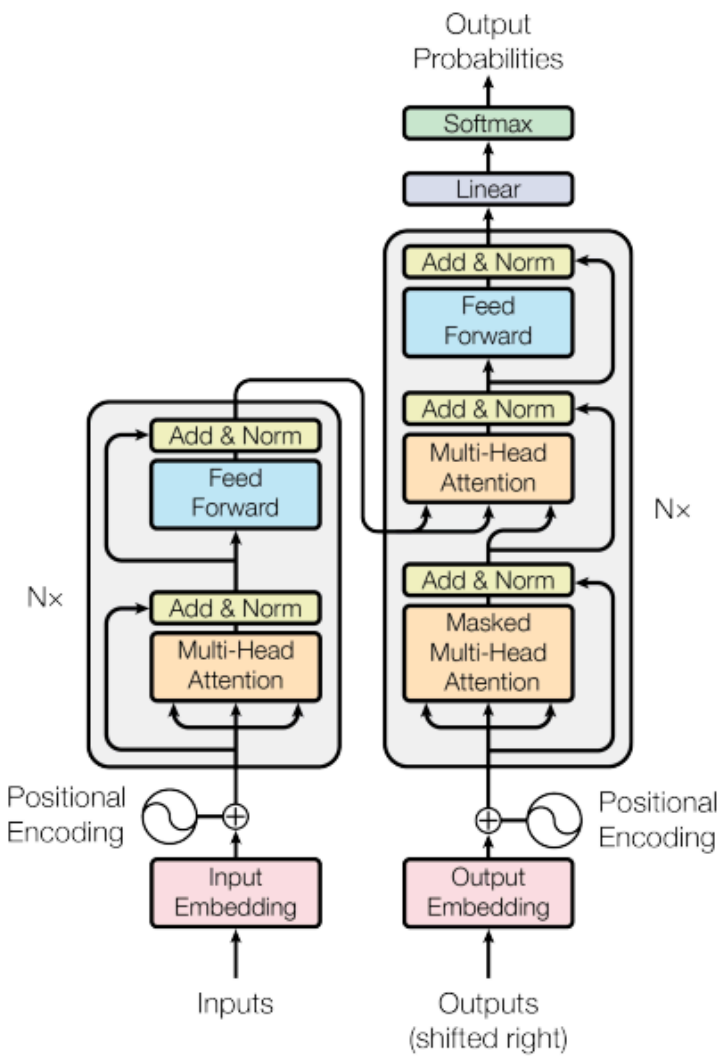


Figure 1: The Transformer - model architecture.

위 그림은 Transformer의 전체 아키텍처를 나타낸다. 모델은 Self-Attention과 Point-wise Fully Connected Layer로 구성되며, 왼쪽은 인코더(Encoder), 오른쪽은 디코더(Decoder)에 해당한다.

3.1 Positional Embedding

Transformer는 입력 데이터를 한꺼번에 처리하는 구조이므로, 기존의 RNN처럼 단어 순서를 자연스럽게 반영할 수 없다. 이를 해결하기 위해 **Positional Encoding**을 사용하여 단어의 위치 정보를 임베딩 벡터에 추가한다.

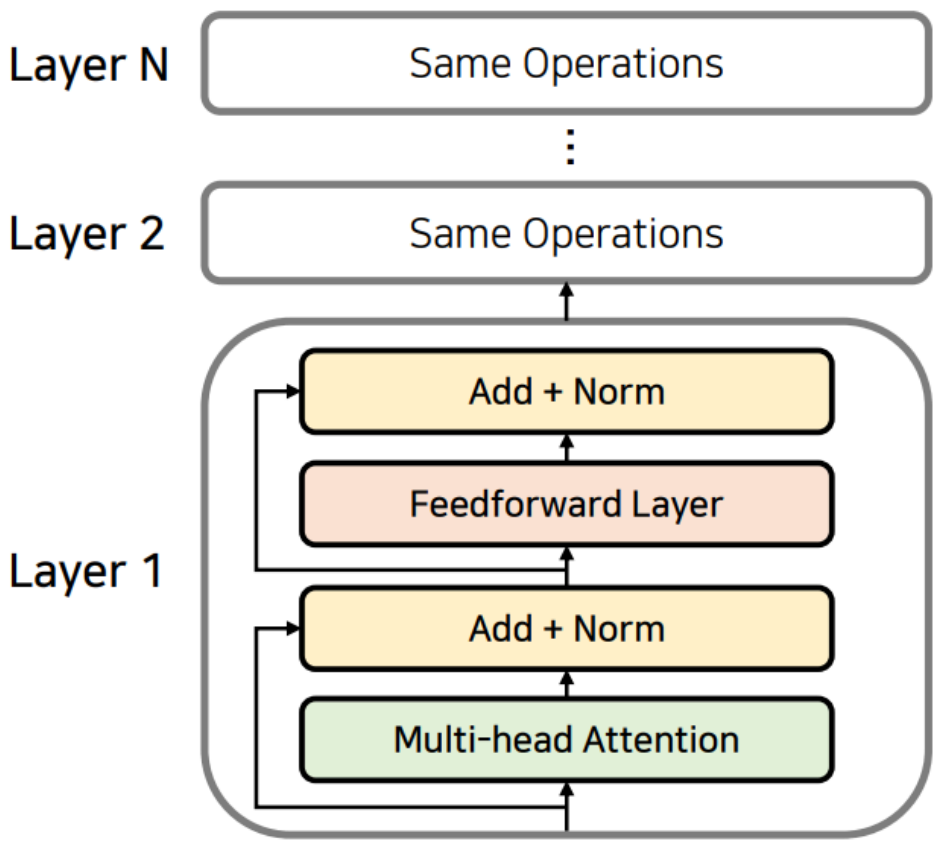
Positional Encoding은 다음과 같은 주기 함수로 정의된다.

$$PE(pos, 2i) = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right)$$

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

이 방식을 활용하면 단어의 상대적 위치를 벡터 형태로 표현할 수 있어, RNN 없이도 문장 내 순서를 반영할 수 있다.

3.2 Attention

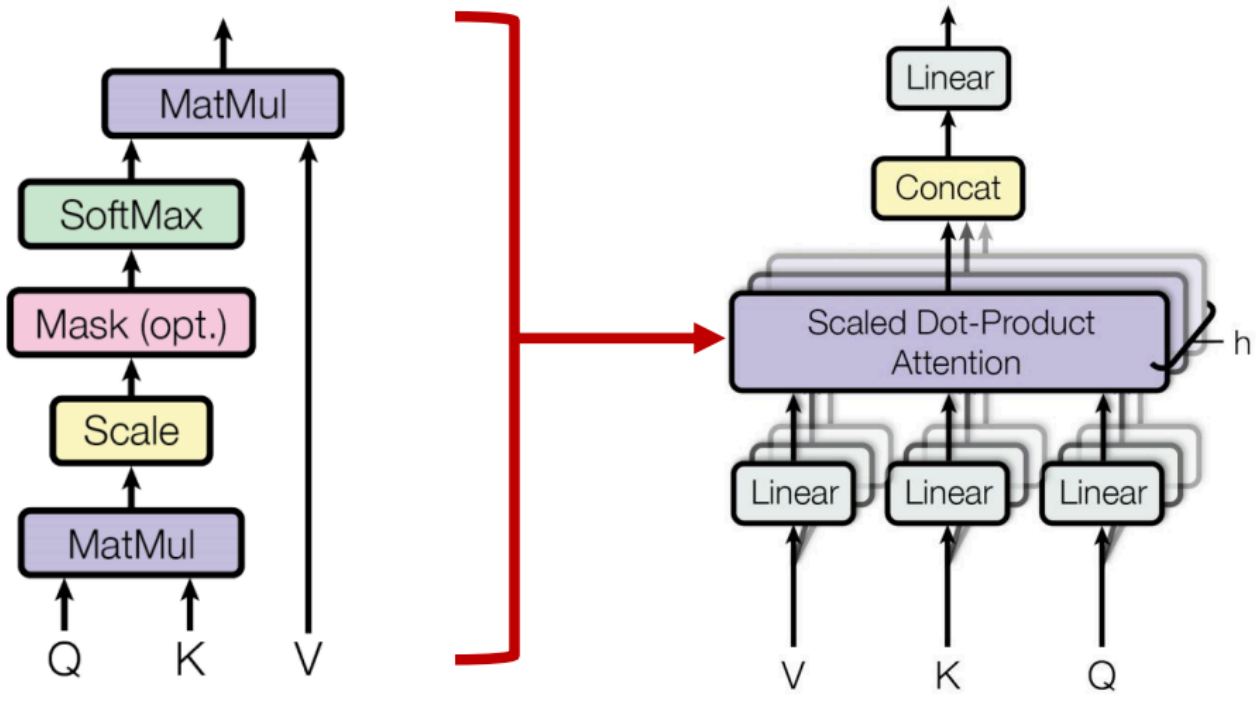


위 그림은 Transformer의 인코더 구조를 나타낸다. 각 인코더 레이어는 동일한 구조로 구성되며, 논문에서는 6개의 레이어를 쌓는 방식을 사용했다.

입력 데이터는 임베딩 과정을 거친 후, Residual Connection 방식으로 Multi-Head Attention을 수행하고, 정규화(Normalization)를 거친 후 피드포워드 신경망(Feedforward Layer)을 통과한다. 각 레이어는 서로 다른 파라미터를 가진다.

3.2.1 Self Attention

Multi-head attention



left : Scaled dot-product attiontion, right : Multi-head attention

Self-Attention에서는 각 단어를 Query(Q), Key(K), Value(V)로 변환해 문장 내 다른 단어들과 얼마나 관련이 있는지 계산한다. 예를 들어, "I love you"라는 문장에서 "I"를 기준으로 나머지 단어들과의 연관성을 계산할 수 있다.

어텐션 값은 다음 수식으로 계산된다.

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

3.2.2 Multi-head attention

기본 Self-Attention을 확장한 형태가 **Multi-Head Attention**

$$MultiHead(Q, K, V) = Concat(head_1, \ldots, head_h)W_O$$

$$\text{where } head_i = Attention(QW_{Q_i}, KW_{K_i}, VW_{V_i})$$

즉, 여러 개의 어텐션 헤드를 사용하여 정보를 다양한 관점에서 학습하고, 최종적으로 결합하는 방식이다. 논문에서는 h=8개의 헤드를 사용했으며, 각 head는 64차원을 가진다.

$$d_k = d_v = d_{model}/h = 64$$

3.3 Position-wise feed forward network

Transformer에서는 총 세 가지 유형의 어텐션을 사용한다.

1. 인코더-디코더 어텐션 (Encoder-Decoder Attention)
    - 디코더에서 생성되는 쿼리(Query)와 인코더의 출력 값을 키(Key), 밸류(Value)로 사용한다.
    - 이를 통해 디코더가 입력 문장의 모든 단어에 집중할 수 있도록 돕는다.
2. 인코더 셀프 어텐션 (Encoder Self-Attention)
    - 인코더의 각 단어가 문장 내 모든 단어와 관계를 맺을 수 있도록 한다.
3. 마스킹된 디코더 셀프 어텐션 (Masked Decoder Self-Attention)
    - 디코더가 현재까지 생성된 단어만을 참고하도록 하여, 정답 단어를 미리 보지 않도록 방지한다.

4. Conclusion

Transformer는 기존의 RNN, CNN 기반 모델과 달리 순차적 계산 없이 Self-Attention만을 활용하는 구조를 채택하여 학습 속도와 성능을 크게 향상시켰다. 또한, 병렬 처리가 가능해 대량의 데이터를 다룰 때 유리한 특징을 가진다.