논문 발표

Transformer

발표자 황동현

INDEX

제목

Attention is All you need

Attention이 전부다

Attention만 있으면 된다

기존의 순환 신경망(RNN)이나 합성곱 신경망(CNN)
모델들에서 복잡한 구조

➔ **Attention 메커니즘만**으로도
충분히 효과적인 시퀀스 변환 모델을 만들 수 있음

Abstract

개요

1. 기존의 복잡한 RNN이나 CNN 대신, **Attentions 메커니즘만을 사용**하여 더 단순한 네트워크 아키텍처인 **Transformer**를 제안

1. 두 가지 기계 번역 작업에서, 이 모델이 **더 우수한 성능과 병렬화 가능성, 훈련 시간 단축**을 보여줌

2. WMT 2014 영어-독일어 번역 작업에서 **28.4 BLEU** (기존 결과보다 2 BLEU 향상)
   영어-프랑스어 번역 작업에서 **41.8 BLEU** (단일 모델 최고 점수)

➔최고 모델들의 훈련 비용보다 **훨씬 적은 비용으로** 뛰어난 결과를 달성
➔대규모 및 제한된 훈련 데이터 모두에서 **영어 구성 구문 분석 작업에** 성공적으로 적용
➔다른 작업에도 **잘 일반화된다는** 것을 입증

01 Introduction | 소개

1. 순환 신경망의 중요성
- **RNN, LSTM, Gated RNN**은 시퀀스 모델링 및 변환 문제에서 최첨단 접근 방식으로 자리 잡았습니다.

2. 순차적 계산의 한계
- **RNN 모델은** 본질적으로 순차적인 계산을 요구하여 **긴 시퀀스**에서는 병렬화가 어렵고, **메모리 제약**으로 인해 배치 크기가 제한됩니다.

3. Attention 메커니즘의 역할
- Attention 메커니즘은 **시퀀스의 거리와 상관없이** 종속성을 모델링할 수 있도록 도와주며, 이는 다양한 작업에서 중요한 역할을 합니다.

4. Transformer의 제안
- **순환을 배제**하고 전적으로 Attention 메커니즘에 의존하는 Transformer 모델을 제안합니다. 이는 병렬화를 크게 증가시키며, **짧은 시간** 내에 높은 번역 품질을 달성할 수 있습니다.

02 Background | 배경

1. 순차적 계산을 줄이기 위한 기존 모델
- Extended Neural GPU, ByteNet, ConvS2S는 **합성곱 신경망**을 사용하여 병렬로 계산하지만, 위치 간 **거리에 따라 연산 수가 증가**합니다.

2. 트랜스포머의 혁신
- 트랜스포머는 위치 간 **거리에 관계없이 상수 연산 수로** 줄이며, **Multi-Head Attention**으로 유효 해상도 감소 문제를 해결합니다.

3. 셀프 Attention의 역할
- 셀프 Attention은 **시퀀스 내 위치 간 관계를 계산**하며, 독해, 요약 등 다양한 작업에서 사용됩니다.

4. 엔드 투 엔드 메모리 네트워크
- **순환 Attention 메커니즘**을 사용하며, **간단한 질문 응답 및 언어 모델링 작업**에서 성능을 보입니다.

5. 트랜스포머의 독창성
- 트랜스포머는 시퀀스-정렬 RNN이나 합성곱 없이 **셀프 Attention만으로** 입력 및 출력 표현을 계산하는 첫 번째 변환 모델입니다.

하지만 이러한 RNN에 기반한 seq2seq 모델에는 크게 두 가지 문제가 있습니다.
첫째, 하나의 고정된 크기의 벡터에 모든 정보를 압축하려고 하니까 정보 손실이 발생합니다.
둘째, RNN의 고질적인 문제인 기울기 소실(vanishing gradient) 문제가 존재합니다.

1. 어텐션(Attention)의 아이디어

어텐션의 기본 아이디어는 디코더에서 출력 단어를 예측하는 매 시점(time step)마다, 인코더에서의 전체 입력 문장을 다시 한 번 참고한다는 점입니다. 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 됩니다.

트랜스포머(Transformer)는 2017년 구글이 발표한 논문인 "Attention is all you need"에서 나온 모델로 기존의 seq2seq의 구조인 인코더-디코더를 따르면서도, 논문의 이름처럼 어텐션(Attention)만으로 구현한 모델입니다. 이 모델은 RNN을 사용하지 않고, 인코더-디코더 구조를 설계하였음에도 번역 성능에서도 RNN보다 우수한 성능을 보여주었습니다.

출처 : 위키독스 (https://wikidocs.net/22893)

03 Model Architecture

모델 구조

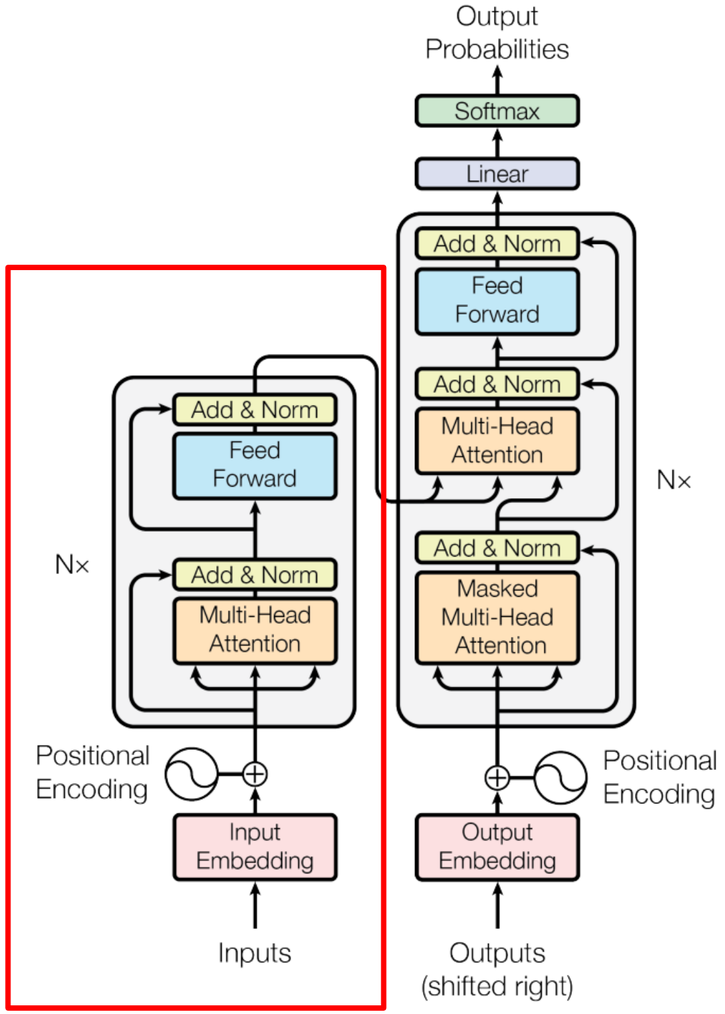


Figure 1: The Transformer - model architecture.

인코더

1. **Input Embedding**: 입력 심볼이 입력 임베딩 레이어를 통해 벡터로 변환됩니다.
2. **Positional Encoding**: 위치 정보를 추가하여 순서의 중요성을 반영합니다.
3. **N개의 인코더 레이어**: 인코더는 여러 개의 동일한 레이어로 구성됩니다.
   - 각 레이어는 다음과 같은 두 가지 하위 레이어로 구성됩니다.
     1) **Multi-Head Attention**: 여러 개의 어텐션 헤드를 사용하여 서로 다른 부분의 입력 시퀀스 간의 관계를 병렬로 학습합니다.
     2) **Feed Forward**: 완전 연결 레이어를 통해 비선형 변환을 적용합니다.
     3) **Add & Norm**: 각 하위 레이어의 출력에 원래 입력을 더하고(normalization) 정규화합니다.

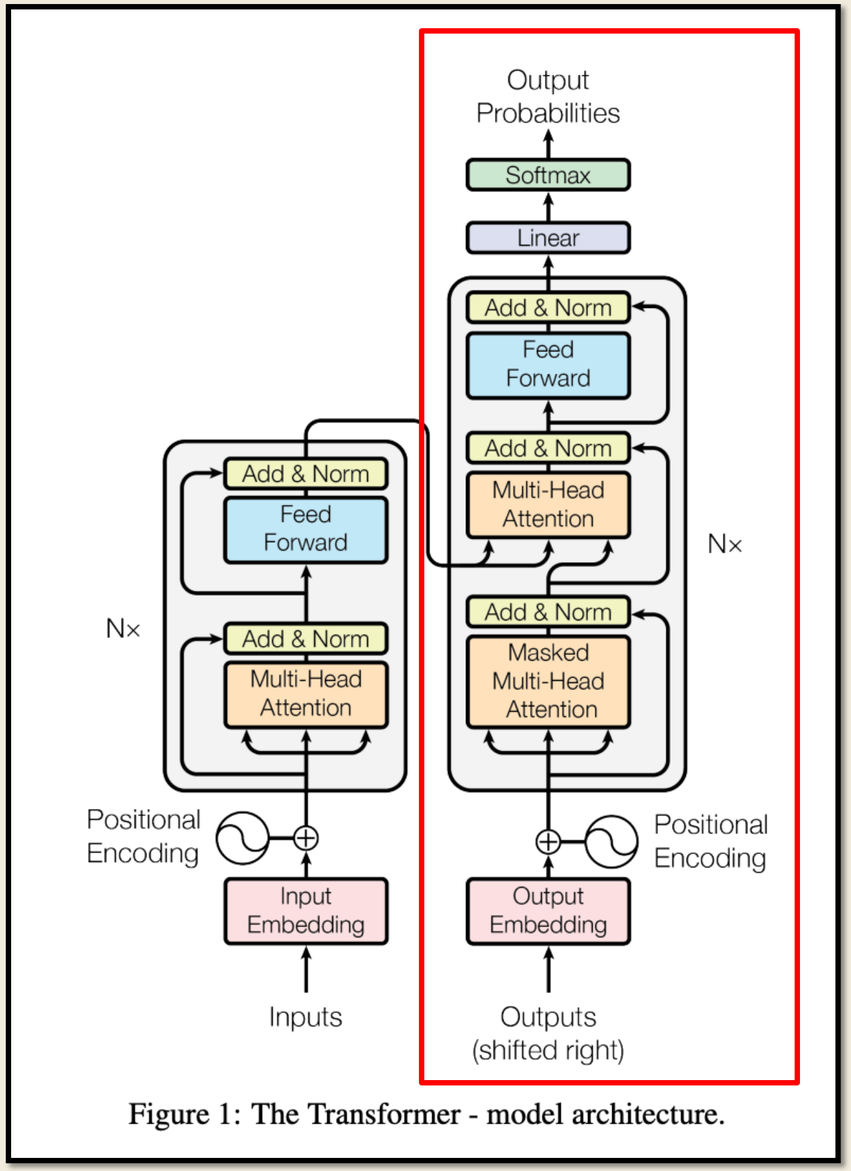


Figure 1: The Transformer - model architecture.

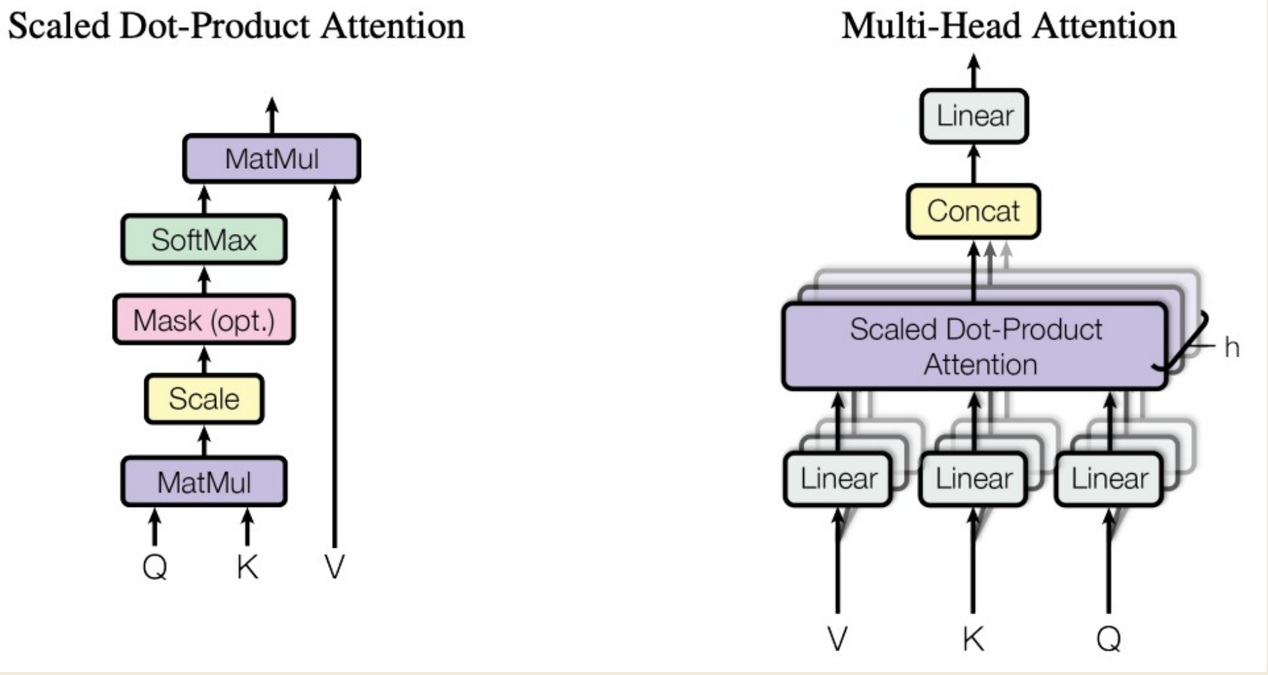
디코더

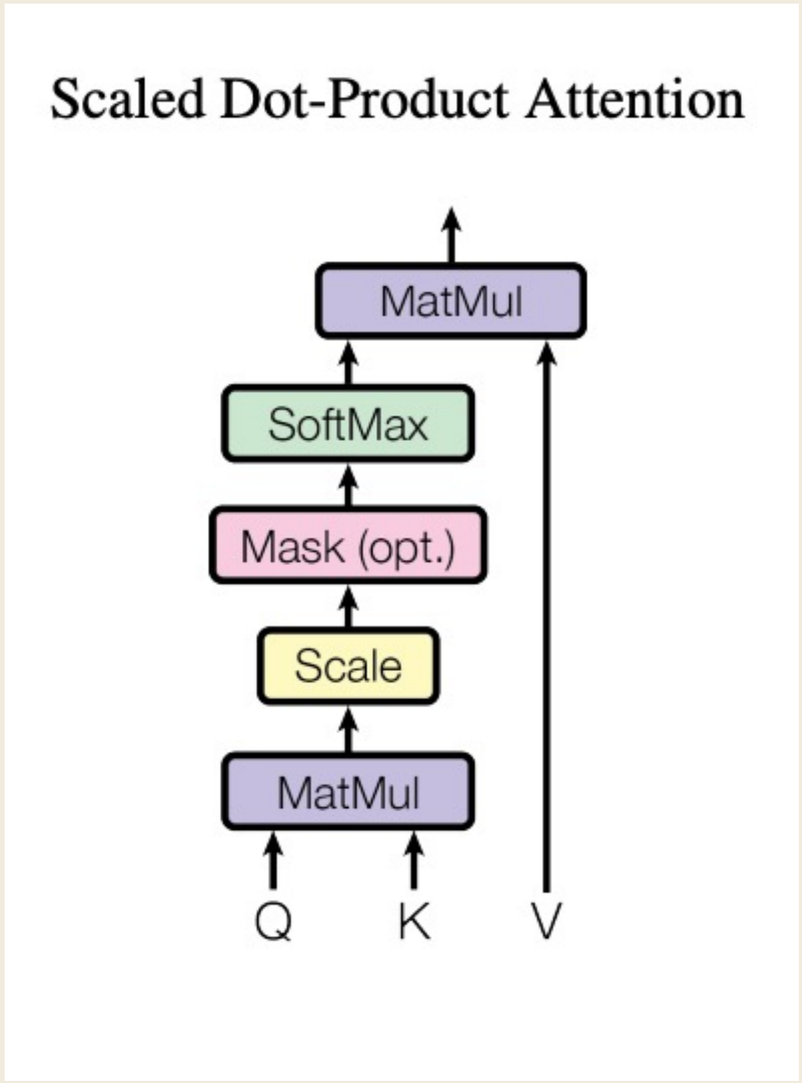
1. **Output Embedding**
   - 출력 심볼이 출력 임베딩 레이어를 통해 벡터로 변환됩니다.
2. **Positional Encoding**
   - 위치 정보를 추가하여 순서의 중요성을 반영합니다.
3. **N개의 디코더 레이어**: 디코더도 여러 개의 동일한 레이어로 구성됩니다.
   - 각 레이어는 다음과 같은 세 가지 하위 레이어로 구성됩니다.
     1) **Masked Multi-Head Attention**: 현재 위치 이후의 정보가 보이지 않도록 마스크된 어텐션을 적용하여 출력 시퀀스를 한 요소씩 생성합니다.
     2) **Multi-Head Attention**: 인코더의 출력과 디코더의 입력을 기반으로 어텐션을 적용합니다.
     3) **Feed Forward**: 완전 연결 레이어를 통해 비선형 변환을 적용합니다.
     4) **Add & Norm**: 각 하위 레이어의 출력에 원래 입력을 더하고(normalization) 정규화합니다.

3.2 어텐션

1. **어텐션 함수 정의**: 쿼리와 키-값 쌍의 집합을 출력으로 매핑하는 함수입니다.
2. **벡터 표현**: 쿼리, 키, 값, 출력은 모두 벡터입니다.
3. **가중합 계산**: 출력은 값들의 가중합으로 계산되고, 가중치는 쿼리와 키의 호환성 함수로 계산됩니다.

- Query (Q): 현재 찾고자 하는 정보의 벡터.
- Key (K): 각 입력의 특성을 나타내는 벡터.
- Value (V): 최종 출력에 반영될 입력의 정보 벡터.





3.2.1 스케일드 닷 프로덕트 어텐션

1) **입력**: Q (쿼리), K (키), V (값).

2) **행렬 곱셈** (MatMul): Q와 K를 곱합니다.

3) **스케일링** (Scale): 앞서 곱한 결과를 스케일링합니다.

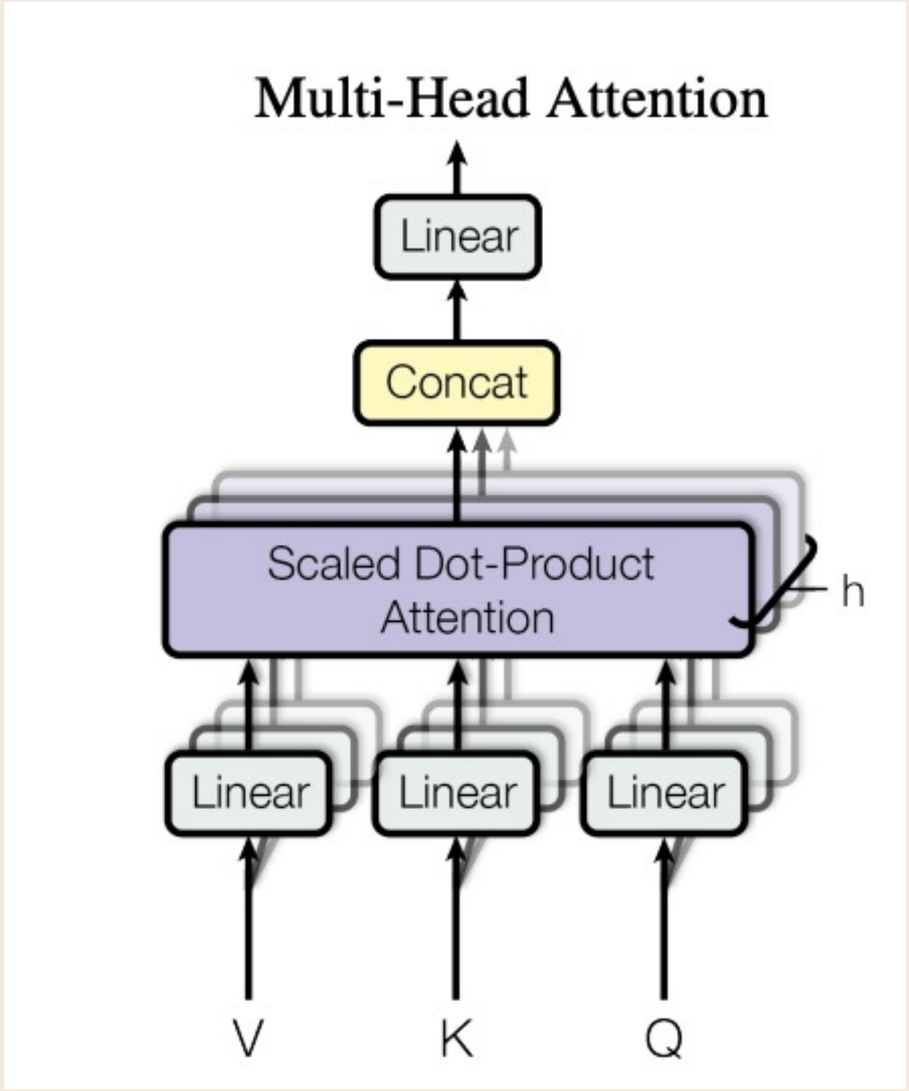
4) **마스킹** (Mask, 선택적): 필요한 경우 마스킹을 적용합니다.

5) **소프트맥스** (SoftMax): 가중치를 계산하기 위해 소프트맥스를 적용합니다.

6) **행렬 곱셈** (MatMul): 소프트맥스 결과와 V를 곱합니다.

7) **출력**: 최종 어텐션 값.

-> 입력된 데이터의 중요한 부분을 강조하고, 덜 중요한 부분을 무시할 수 있게 함으로써 모델이 더 중요한 정보를 더 많이 학습하도록 돕습니다.



3.2.2 멀티-헤드 어텐션

1) **입력**: Q (쿼리), K (키), V (값).

2) **선형 투영** (Linear): Q, K, V 각각을 여러 헤드로 나누어 선형 투영합니다.

3) **스케일드 닷 프로덕트 어텐션**: 각 헤드마다 스케일드 닷 프로덕트 어텐션을 병렬로 수행합니다.

4) **병합** (Concat): 모든 헤드의 출력을 병합합니다.

5) **최종 선형 투영** (Linear): 병합된 출력을 다시 선형 투영하여 최종 어텐션 값을 얻습니다.

6) **출력**: 최종 어텐션 값.

-> 입력 데이터를 다양한 시각에서 동시에 처리함으로써 모델의 표현력을 증가시킵니다. 이는 복잡한 상호 관계를 더 잘 이해하고 다양한 패턴을 학습하는 데 도움을 줍니다.

3.2.3 모델에서 어텐션의 응용

1. 인코더-디코더 어텐션 레이어

- 쿼리는 이전 디코더 레이어에서 오고, 키와 값은 인코더의 출력에서 옵니다.
- 디코더의 모든 위치가 입력 시퀀스의 모든 위치에 주목할 수 있게 합니다.

2. 셀프 어텐션이 포함된 인코더

- 키, 값, 쿼리가 모두 인코더의 이전 레이어 출력에서 옵니다.
- 인코더의 각 위치는 이전 레이어의 모든 위치에 주목할 수 있습니다.

3. 셀프 어텐션이 포함된 디코더

- 디코더의 각 위치가 해당 위치까지의 모든 위치에 주목할 수 있게 합니다.

3.3 위치별 피드-포워드 네트워크 Position-wise Feed-Forward Networks

1. 구성 요소: 두 개의 선형 변환과 그 사이의 ReLU 활성화로 구성됩니다.

2. 매개변수: 각 레이어마다 다른 매개변수를 사용합니다.

3. 차원: 입력과 출력의 차원은 d_model = 512, 내부 레이어의 차원은 d_ff = 2048입니다.

목적

1. 두 개의 완전 연결층 (fully connected layer)으로 구성되어 **비선형적인 정보 변환**을 수행하여 모델의 정보 처리 능력을 향상시킵니다.

2. Transformer 모델은 셀프 어텐션 레이어를 반복적으로 사용하기 때문에 **기울기가 소실되**는 문제가 발생하는데 셀프 어텐션 레이어 사이에 위치하여 **정보 흐름을 끊어주는 역할**을 수행하여 모델의 매개변수가 업데이트되지 않아 기울기를 소실하는 문제를 해결합니다.

➔ 모델의 정보 처리 능력을 향상시키고 안정성을 강화하여

더욱 정확하고 유창한 번역 결과를 생성

레이어 타입의 효율성 비교

복잡도 순차연산 최대경로길이

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

- 각 레이어가 주어진 문제에 얼마나 적합한지 평가할 때 유용한 정보를 제공합니다.

1) Self-Attention: 복잡도가 높지만 순차 연산이 적고 경로 길이가 짧아 빠르게 전체 시퀀스를 처리할 수 있습니다.

2) Recurrent: 복잡도가 낮지만 순차 연산이 많아 전체 시퀀스를 처리하는 데 더 오래 걸릴 수 있습니다.

3) Convolutional: 복잡도와 순차 연산 모두에서 균형을 잡고 있습니다.

4) Self-Attention (제한된): 복잡도와 경로 길이 모두에서 일정한 이점을 제공합니다.

04

Why
Self-Attetion

왜
셀프-어텐션인가

비교

Self-Attention

Recurrent

Convolution

기준

계층당 총 계산 복잡도

병렬화 가능한 계산량
(최소 순차 연산 수)

장기 의존성 간의 경로 길이

Self-Attention

➔ **모든 위치를 일정한** 수의 순차적으로 실행된 연산으로 연결

➔ 시퀀스 길이 n이 표현 차원 d보다 작을 때 계산 **복잡도 면에서 순환 층보다 빠름**

➔ 장기 시퀀스의 경우 성능을 향상시키기 위해 입력 시퀀스에서 출력 위치를 중심으로
이웃 크기 r만 고려하는 방식으로 **제한 가능** Self-Attention (restricted)

05 Training 훈련

WMT 2014 영어-독일어

약 450만 개 문장 쌍

약 37,000개의 공유 어휘

WMT 2014 영어-프랑스어

약 3600만 개 문장

32,000개의 단어 조각 어휘

하드웨어 및 일정

8개의 NVIDIA P100 GPU가
장착된 하나의 기계

베이스 모델 훈련 시간
각 단계 0.4초
100,000단계 (약 12시간)

큰 모델 훈련 시간
각 단계 .10초
300,000단계 (약 3.5일)

옵티마이저

Adam 옵티마이저

06

Results

훈련 결과

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 영어-독일어 번역 작업

-> 큰 트랜스포머 모델(Transformer (big))이
이전 최고 모델을 2.0 BLEU 이상 능가.

-> 새로운 최고 BLEU 점수: 28.4.

→ 기본 모델도 모든 이전 모델과 앙상블을 능가

→ 훈련 비용은 훨씬 저렴.

WMT 2014 영어-프랑스어 번역 작업

-> 큰 모델이 41.0 BLEU 점수를 달성, 이전 최고 단일 모델을 능가.

-> 훈련 비용은 이전 최고 모델의 1/4 이하.

07 Conclusion 결론

1. Transformer

- 전적으로 Attention에 기반한 최초의 시퀀스 변환 모델.
- 인코더-디코더 아키텍처에서 순환 층을 Multi-Head, Self-Attention으로 대체.

2. 번역 작업 성과

- Transformer는 순환 층이나 합성곱 층 기반 아키텍처보다 훈련 속도가 훨씬 빠름.
- WMT 2014 영어-독일어 및 영어-프랑스어 번역 작업에서 새로운 최첨단 성과 달성.
- 영어-독일어 번역 작업에서 최상위 모델은 이전의 모든 앙상블 모델을 능가.

3. 미래 계획

- Attention 기반 모델을 다른 작업에 적용할 계획.
- 텍스트 이외의 입력 및 출력 양식 문제로 Transformer를 확장할 계획.
- 이미지, 오디오 및 비디오와 같은 대규모 입력 및 출력을 효율적으로 처리하기 위해 제한된 Attention 메커니즘 조사할 계획.
- 생성 과정을 덜 순차적으로 만드는 것도 연구 목표 중 하나.

감사합니다

THANK YOU