Week3-3 과제(Attention Is All You Need 리뷰)

논문 : Attention Is All You Need

저자 : Ashish Vaswani et al.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA

Abstract

▼ 요약

어텐션 기제를 통한 인코더 디코더 연결을 한 트랜스포머 소개. 기계 번역에서 성능 우수하고 병렬화가 용이해서 학습 시간이 적게 소모. 높은 BLEU 스코어 달성하고 일반화 성능도 또한 우수

이전까지 주요한 시퀀스 변환 모델은 인코더와 디코더를 포함한 복잡한 RNN 혹은 CNN 모델들이다.

- 어텐션 기제를 사용하고 인코더 디코더를 연결한 모델이 최고 성능을 보이는 변환 모델이었다. (아마도 seq2seq 계열을 말하는 듯)

저자는 어텐션 기제만 사용한 간단한 네트워크 구조인 트랜스포머 제시

- 트랜스포머 장점 : 성능도 좋고 병렬화도 가능하고 학습시간도 많이 줄어든다.
  - WMT 2014 Englishto-German translation task : 2017년 당시 최고 성능인 BLEU 점수 28.4 달성
  - WMT 2014 English-to-French translation task : 단일 모델로 BLEU 점수 41.8 달성
    - 8개의 GPU를 사용해서 3.5일 학습함. (학습 비용 축소)
  - 영어 유권자 구문 분석에 트랜스포머를 성공적으로 적용해서 트랜스포머의 일반화 성능이 높은 것을 확인

▼ **Transduction 더 알아보기**

- Transductive learning : 일반화가 없는 학습, 특정 데이터셋에만 최적화된 모델을 생성하는 과정
  - 예 : 추천 시스템 - 일반적인 추천모델을 설계할 필요 없고 특정 분야나 상품 혹은 회사에 고유한 추천모델을 설계한다. 당사에서만 잘 돌아가는 추천 모델을 구성한다.
- Induction learning : 일반화 학습, 이미지 및 텍스트 등을 학습시켜 일반화된 모델을 생성하는 과정
- (Sequence) Transduction : 시퀀스를 다른 형태의 시퀀스로 번역, 변환을 의미한다. 또 입력(영어)과 출력(불어) 시퀀스의 성질이 다르다.
  - 예 : seq2seq 은 입력 시퀀스의 각 토큰들을 인코더 디코더 구조를 통해 출력 토큰을 생성한다.
- Autoencoder : 입력과 출력 시퀀스의 성질이 같음
- Transducer : 어떤 신호를 다른 신호로 변환시키는 도구

1 Introduction

▼ 요약

어텐션 기제는 입력과 출력 시퀀스의 거리에 상관 없이 시퀀스 간 의존성을 모델링할 수 있고 순환 신경망과 같이 쓰임. 트랜스포머는 순환을 피하고 입력과 출력 사이의 전역 의존성을 알아내기 위해 전적으로 어텐션 기제를 사용한다. 8개의 P100 GPU에서 12시간만 훈련해도 더 많은 병렬화와 높은 성능에 도달 가능.

RNN, LSTM, GRU가 시퀀스 모델링과 언어 번역과 언어 모델링과 같은 변환 문제의 SOTA로 자리매김하고 있었다.

~~재귀 언어 모델들과 인코더 디코더 구조~~

2 Background

트랜스포머에서 이것은 평균 주의 가중 위치로 인해 유효 해상도가 감소하지만, 섹션 3.2에 설명된 대로 MHA로 대응. Self-attention는 시퀀스의 표현을 계산하기 위해 단일 시퀀스의 다른 위치와 관련된 어텐션 기제. Self-attention는 독해, 추상적 요약, 텍스트 수반 및 학습 과제 독립적인 문장 표현을 포함한 다양한 과제에 성공적으로 사용.

3 Model Architecture

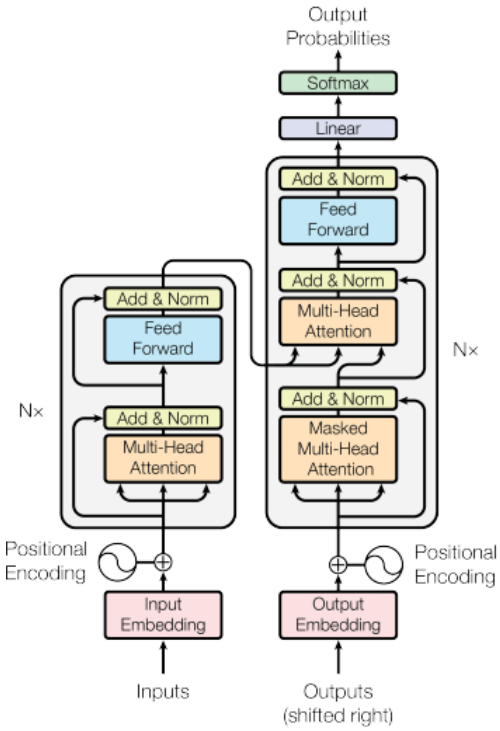

Figure 1: The Transformer - model architecture.

3.1 Encoder and Decoder Stacks

Encoder

- N = 6
- 생성된 결과의 차원은 512.

Decoder

- N = 6
- [질문] This masking, combined with fact that the output embeddings are offset by one position, ensures that the predictions for position i can depend only on the known outputs at positions less than i.

3.2 Attention

어텐션 함수는 쿼리, 키, 값 및 출력이 모두 벡터인 출력에 쿼리와 키 값 쌍 집합을 매핑. 출력은 각 값에 할당된 가중치가 해당 키와 쿼리의 compatibility 함수에 의해 계산되는 값의 가중치 합으로 계산.

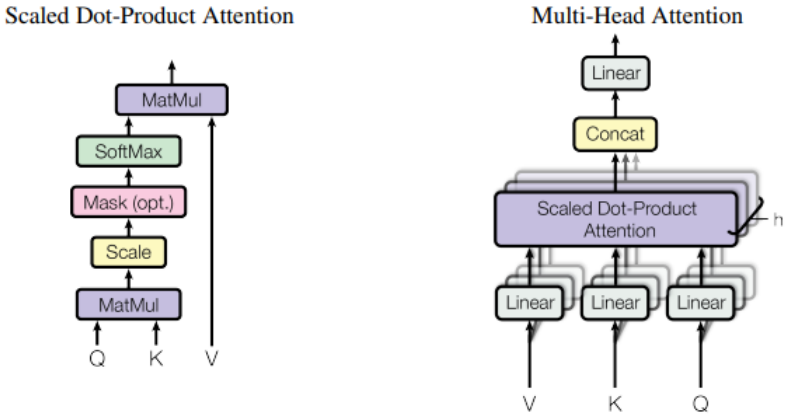

Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

3.2.1 Scaled Dot-Product Attention

Q, K, V 출력 행렬 :

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

3.2.2 Multi-Head Attention

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

Where the projections are parameter matrices $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ and $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$.

In this work we employ $h = 8$ parallel attention layers, or heads. For each of these we use $d_k = d_v = d_{\text{model}}/h = 64$. Due to the reduced dimension of each head, the total computational cost is similar to that of single-head attention with full dimensionality.

3.2.3 Applications of Attention in our Model

MHA 사용되는 레이어

1. encoder-decoder attention레이어
2. encoder 안에 self-attention 레이어
3. decoder 안에 self-attention 레이어

3.3 Position-wise Feed-Forward Networks

each of the layers in our encoder and decoder contains a fully connected feed-forward network

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \quad (2)$$

While the linear transformations are the same across different positions, they use different parameters from layer to layer. Another way of describing this is as two convolutions with kernel size 1. The dimensionality of input and output is $d_{\text{model}} = 512$, and the inner-layer has dimensionality $d_{ff} = 2048$.

- two linear transformations with a ReLU activation in between

3.4 Embeddings and Softmax

위치 정보가 없기 때문에 위치 정보를 주임하기 위한 위치 임베딩 사용. 사인, 코사인 함수로 위치 인코딩.

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

더 긴 시퀀스로 추론할 수 있기 때문에 사인 함수 사용

3.5 Positional Encoding

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

4 Why Self-Attention

Self-Attention 사용으로 얻는 세 가지 이점

1. 레이어 당 계산 복잡도 감소
2. 병렬화될 수 있는 계산의 양
3. 장기 의존성 문제 해결

5 Training

5.1 Training Data and Batching

학습 데이터

1. standard WMT 2014 English-German dataset, 4.5만 문장 쌍, BPE 인코딩, 약 37000 토큰의 소스-타겟 단어를 공유.
2. larger WMT 2014 English-French dataset, 36만 문장과 32000 워드 피스 토큰.

배치 방법

- 문장 쌍은 배치 당 유사한 시퀀스 길이가짐.
- 각 학습 배치마다 25000 소스 토큰과 25000 타겟 토큰으로 구성된 문장 쌍을 포함

5.2 Hardware and Schedule

학습에 엔비디아 GPU P100 개 사용.

- Base 모델은 학습 스텝마다 0.4초 소요. 100,000 스텝을 학습하였고 학습하는데 12시간 소요.
- Larger 모델은 학습 스텝마다 1.0초 소요. 300,000 스텝을 학습하였고 학습하는데 3.5일 소요

5.3 Optimizer

아담 옵티마이저 사용

- B1 = 0.9
- B2 = 0.98
- 오차 = 10^(-9)

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

This corresponds to increasing the learning rate linearly for the first $warmup\_steps$ training steps, and decreasing it thereafter proportionally to the inverse square root of the step number. We used $warmup\_steps = 4000$.

5.4 Regularization

Residual Dropout

- 서브 레이어 인풋과 정규화되기 전에 각 서브 레이어에 드롭아웃 적용
- 양 인코더, 디코더 스택에 임베딩과 위치 인코딩의 합을 드롭아웃 적용
- Base 모델의 드롭아웃 비율 = 0.1

Label Smoothing

라벨 스무딩 value = 0.1

질문 : This hurts perplexity, as the model learns to be more unsure, but improves accuracy and BLEU score.

6 Results

6.1 Machine Translation

WMT 2014 English-to-German translation task

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-to-French translation task

- The Transformer (big) model trained for English-to-French used dropout rate Pdrop = 0.1, instead of 0.3

For the base models, we used a single model obtained by averaging the last 5 checkpoints, which were written at 10-minute intervals. For the big models, we averaged the last 20 checkpoints. We used **beam search** with a **beam size of 4** and **length penalty α = 0.6** [38]. These hyperparameters were chosen after experimentation on the development set. We set the maximum output length during inference to **input length + 50, but terminate early when possible** [38].

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

A

- 계산량 유지하면서 어텐션 헤드 개수와 K, V 차원 조정
- 단일 헤드보다는 다중 헤드가 높은 성능을 달성하지만 헤드가 너무 많아도 성능 하락

B

- Key 를 줄이면 모델 품질 저하
- This suggests that determining compatibility is not easy and that a more sophisticated compatibility function than dot product may be beneficial.

C, D

- 큰 모델 성능이 더 좋음
- 드롭아웃은 오버피팅 방지에 좋음

E

- we replace our sinusoidal positional encoding with learned positional embeddings [9], and observe nearly identical results to the base model.

6.3 English Constituency Parsing

트랜스포머가 다른 작업으로 일반화할 수 있는지 평가하기 위해 영어 유권 구문 분석에 대한 실험을 수행. 영어 유권 구문 분석은 출력이 입력보다 긴 편.

- BerkleyParser 말뭉치로 준지도 학습

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

트랜스포머가 BerkeleyParser를 능가

7 Conclusion

본 연구에서는 인코더-디코더 아키텍처에서 가장 일반적으로 사용되는 반복 레이어를 다중 헤드 셀프 어텐션으로 대체하는 Transformer 제시. 기계 번역에서 빠른 훈련과 높은 성능 달성.

트랜스포머를 텍스트말고도 이미지, 오디오, 비디오와 같은 큰 입출력을 효율적으로 처리하기 위해 로컬 어텐션 기제를 연구할 예정. 생성(아마도 디코딩)을 덜 순차적으로 만들기 위한 것 역시 다른 연구 목표.

본 연구의 모델은 https://github.com/tensorflow/tensor2tensor 에서 확인 가능.

Attention Visualizations

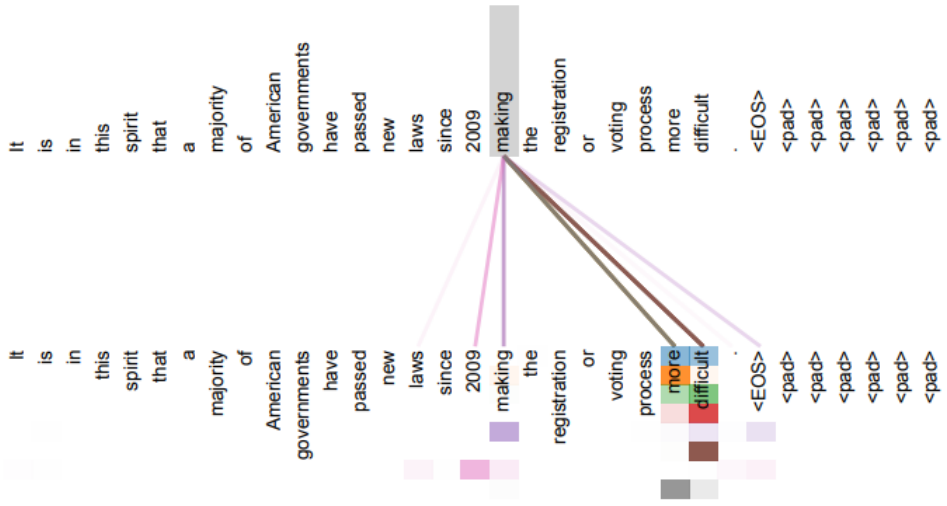

Figure 3: An example of the attention mechanism following long-distance dependencies in the encoder self-attention in layer 5 of 6. Many of the attention heads attend to a distant dependency of the verb 'making', completing the phrase 'making...more difficult'. Attentions here shown only for the word 'making'. Different colors represent different heads. Best viewed in color.