[논문리뷰] BERT

Abstract

BERT는 Bidirectional Encoder Representations from Transformers 를 뜻하고 다른 LM(language model)들과 달리, 모든 층의 left and right (bidirectional) 방향으로 pretrain합니다.

먼저 pretrained language model을 설명하자면, 두 가지로 나뉩니다. feature-based와 fine-tuning.

feature-based 방식엔 대표적으로 ELMo가 있고, pre-trained representations를 추가해서 task-specific 구조를 사용합니다. ***fine-tuning방식***은 대표적으로 OpenAI GPT가 있고, 모든 pre-trained 파라미터를 fine-tuning해서 최소한의 task-specific 파라미터를 사용합니다. 두 방식 모두 pre-traing할 때 unidirectional language model을 사용합니다.

이 논문에서는 이 fine-tuning 방식을 개선해서 masked language model(MLM)으로 성능을 향상한 BERT를 소개합니다. MLM은 랜덤으로 몇 토큰들을 마스킹하고, 마스킹된 토큰을 예측합니다. 그래서 기존의 left-to-right 구조와 달리, 양방향으로 학습을 하게 되고 이로 인해 정확도가 향상될 것이라고 말합니다. 그리고 next-sentence-prediction 방식 또한 접목시켰습니다.

2 Related Work

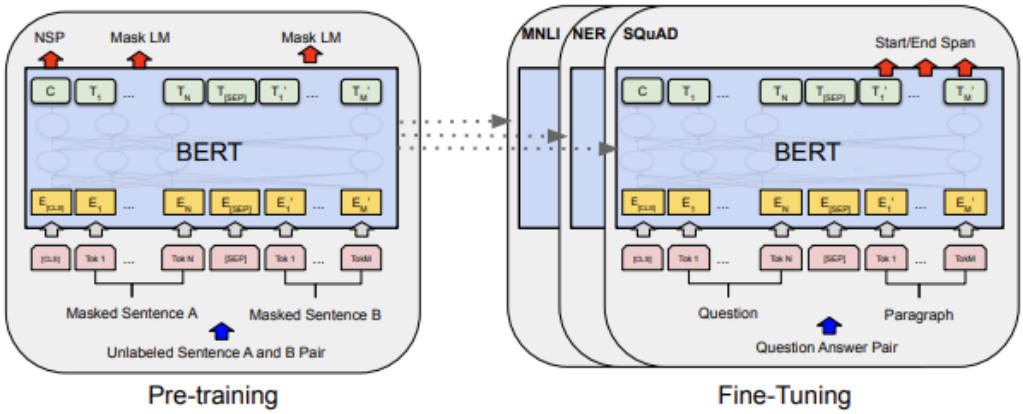
2.1 Unsupervised Feature-based Approaches

Pre-trained word embeddding은 NLP의 중요한 핵심으로 그동안 left-to-right LM을 사용해왔습니다.

ElMo는 left-to-right and right-to-left LM을 통해 NLP를 발전시켜왔고 SOTA성능을 제공해왔습니다.

2.2Unsupervised Fine-tuning Approaches

OpenAI GPT는 적은 파라미터로 학습할 수 있습니다.



3 BERT

Model Architecture

BERT의 모델 구조는 multi-layer bidirectional Transformer encoder 입니다. Transformer의 개념과 구조는 2017년 발표한 Attention is all you need 논문을 참고.

layer 개수: L

hidden size: H

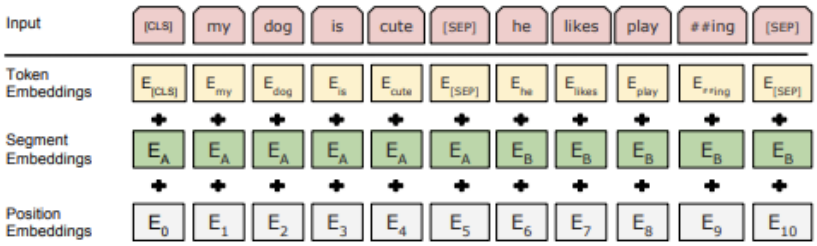
self-attention heads 개수: A

먼저 두 모델의 사이즈를 둔다.

- BERT BASE : L=12,H=768,A=12,totalparameter=110ML=12,H=768,A=12,totalparameter=110M
- BERT LARGE : L=24,H=1024,A=16,totalparameter=240

BERT BASE는 OpenAI GPT와 같은 사이즈로 비교목적. BERT는 bidirectional self-attention하는 반면, GPT는 오직 자신의 왼쪽으로만 토큰을 참조할 수 있는 제한적인 self-attention을 한다.

Input/Output Representations



"sentence"는 인접한 텍스트의 임의의 구간을 가집니다. 단어 임베딩은 WordPiece embeddings을 사용합니다. 쌍으로 구성된 문장을 하나의 "sequence"로 표현하기 위해, 두 문장을 [SEP]이라는 토큰으로 나눕니다. 그리고, 학습된 임베딩을 모든 토큰에 추가합니다. 그래서 주어진 토큰들에 상응하는 토큰들을 summing하고 segment, position embeddings으로 input representation이 생성됩니다.

3.1 Pre-training BERT

pre-train BERT는 두 가지의 unsupervised tasks를 합니다.

Task#1: Masked LM

상식적으로, 양방향 모델이 한방향 모델보다는 성능이 좋습니다. 하지만 일반적 전통적 LM은 모두 한방향입니다. 그래서 양방향 모델을 사용하기 위해서 랜덤으로 몇 input token들을 masking합니다. 이를 "masked LM"(MLM)이라고 칭합니다. 그리고는 그 토큰들을 예측합니다. softmax를 통해.

랜덤으로 15%를 WordPiece tokens를 masking합니다.

단점은, pre-training과 fine-tuning 사이의 mismatch가 생성된다는 것입니다. 왜냐하면 [MASK] 토큰은 fine-tuning때 안 나타납니다. 이를 완화하기 위해서 항상 masked words 를 대체하지 않습니다. (1) [MASK] 토큰 80% (2)랜덤토큰 10%으로 대체, 3)대체하지 않는 토큰 10%.

Task#2: Next Sentence Prediction(NSP)

전체적 문맥을 이해하는 기법입니다 50%는 실제 다음 문장 A(labeled as IsNext), 50%는 corpus의 랜덤 문장(labeled as NotNext).

3.2 Fine-tuning BERT

self-attention 메커니즘을 이용해서 두 문장을 하나의 sequence로 인코딩합니다.

pre-training과 비교해서 fine-tuning은 비용이 더 적습니다.

5 Ablation Studies

Tasks	Dev Set MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
$BERT_{BASE}$	84.4	88.4	86.7	92.7	88.5
No NSP	83.9	84.9	86.5	92.6	87.9
LTR & No NSP	82.1	84.3	77.5	92.1	77.8
+ BiLSTM	82.1	84.1	75.7	91.6	84.9

NSP를 뺐을 때의 성능은 떨어진다는 것을 확인함으로써, NSP의 중요성과 성능을 확인할 수 있습니다.

5.2 Effect of Model Size

Hyperparams				Dev Set Accuracy		
#L	#H	#A	LM (ppl)	MNLI-m	MRPC	SST-2
3	768	12	5.84	77.9	79.8	88.4
6	768	3	5.24	80.6	82.2	90.7
6	768	12	4.68	81.9	84.8	91.3
12	768	12	3.99	84.4	86.7	92.9
12	1024	16	3.54	85.7	86.9	93.3
24	1024	16	3.23	86.6	87.8	93.7

LM(ppl)은 masked LM perplexity(헷갈리는 정도)로 낮을수록 좋음. 그래서 #L과 #H가 점점 커지는(크기가 점점 커질수록) 성능이 좋아진다는 것을 확인할 수 있음.