[논문리뷰]Bert: pre-training of deep bidirectional transformers for language understanding

https://www.semanticscholar.org/reader/df2b0e26d0599ce3e70df8a9da02e51594e0e992

KEY WORD

- Encoder
- down-stream task

> 🥁 먼저 해결해야 할 작업을 Upsteam task라 부르고
> 다음에 최종적으로 해결하고자 하는 작업을Downstream task라고함

- feature-based
- fine-tuning
- deep bidirectional language model
- MLM과 NSP

목차

Abstract

▼ 본문

우리는 **트랜스포머의 양방향 인코더 표현**(**B**idirectional **E**ncoder **R**epresentations from **T**ransformers)을 의미하는 **BERT**라는 새로운 언어 표현 모델을 소개한다. 최근 언어 표현 모델들과 달리, BERT는 모든 레이어에 양방향 문맥에서 공동으로 조절함으로써 라벨이 없는 텍스트로

부터 깊은 양방향 표현들을 사전 훈련하도록 디자인되었다. 이에 대한 결과로, 사전 훈련된 BERT 모델은 작업별 구조 수정을 크게 할 필요 없이(without substantial task-spcific architecture modification) 질문 답변과 언어 추론과 같은 넓은 분야의 작업에서 state-of-the-art 모델을 만들기 위해 그냥 하나의 출력 레이어만 추가만 해도 파인튜닝될 수 있다. BERT는 컨셉적으로 간단하며 실질적으로(empirically) 파워풀하다. 이것은 GLUE 점수를 80.5%(7.7% 절대적 향상), MultiNLI 정확도를 86.7%, SQuAD v1.1 질문 답변 테스트 F1을 93.2(1.5 point 절대적 향상) 그리고 SQuAD v2.0 테스트 F1에서 83.1(5.1 point 절대적 향상) 등을 포함해 열 한개의 자연어 처리 작업에서 새로운 state-of-the-art 결과를 얻는다.

Abstract

We introduce a new language representation model called **BERT**, which stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. Unlike recent language representation models (Peters et al., 2018a; Radford et al., 2018), BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.

BERT is conceptually simple and empirically powerful. It obtains new state-of-the-art results on eleven natural language processing tasks, including pushing the GLUE score to 80.5% (7.7% point absolute improvement), MultiNLI accuracy to 86.7% (4.6% absolute improvement), SQuAD v1.1 question answering Test F1 to 93.2 (1.5 point absolute improvement) and SQuAD v2.0 Test F1 to 83.1 (5.1 point absolute improvement).

BERT(Bidirectional Encoder Representations from Transformer)는 이름 그대로 Transformer의 Encoder를 활용한 Language model

기존 트랜스포머 단점: 단방향 예측. 한정된 down stream task. 번역 기능만 가능한 한정적인 기능

Bert란 어떤모델?

unlabeled data로 부터 **pre-train**을 진행 한 후, 이를 특정 **downstream task(with labeled data)**에 **fine-tuning(transfer learning)**을 하는 모델

Bidirectional LSTM이나 **ELMo**와 같은 모델에서 **bidirectional이란 키워드를 사용**하긴 했으나, BERT에서는 **deep** bidirectional의 **deep**을 더욱 강조하여 기존의 모델들과의 차별성을 강조

Introduction

▼ 본문내용

언어 모델을 사전 훈련하는 것은 많은 자연어 처리 작업 향상에 효과적이라고 알려졌다. 여기에는 문장 간의 관계를 전체적으로 분석하여 예측하는 것을 목표로 하는 자연어 추론 및 의역(paraphrasing)과 같은 문장 수준의 과제와, 토큰 수준에서 미세한 산출물을 산출하기 위해 모델이 요구되는 명명된 실체 인식 및 질문 답변과 같은 토큰 수준의 과제가 포함된다.

다운 스트림 과제를 위해 사전 훈련된 언어 표현들을 적용하는 것에는 두 가지 전략이 존재한다. feature-based와 미세조정(fine-tuning)이다. ELMo와 같은 feature-based 접근 방식은 사전 훈련된 표현들을 추가 기능으로 포함하는 작업별 구조를 사용한다. 생성적 사전 훈련된 트랜스포머(Generative Pre-trained Transformer, GPT)와 같은 미세 조정 접근 방식은 최소한의 작업별 파라미터를 도입하고, 모든 사전 훈련된 파라미터를 간단하게 미세조정함으로써 다운스트림 과제에 대해 훈련한다. 이 두가지 접근 방식은 사전 훈련동안 같은 목적 함수(objective function)을 공유하며, 여기서 일반적인 언어 표현을 학습하기 위해 단방향 언어 학습 모델을 사용한다. 우리는 현재 기술들은 특히 미세 조정 접근 방식에 대해 사전 훈련된 표현들의 힘을 제한한다고 주장한다. 가장 큰 한계는 표준 언어 모델들이 단방향이며 이것은 사전 훈련 동안 사용될 수 있는 구조의 선택에 한계가 있다. 예를 들어 OpenAI GPT에서, 작가들은 모든 토큰이 트랜스포머의 셀프 어텐션 레이어에서 이전 토큰에만 참조를 확인할 수 있는 '왼쪽에서 오른쪽으로' 구조를 사용한다. 이러한 제한은 문장 수준의 업무에 최적이 아니며, 질문 답변과 같은 토큰 수준의 작업을 위한 미세 조정 기반의 접근 방법을 적용할 때 매우 좋지 않을 수 있는데, 이 때 양방향에서 문맥을 통합하는 것이 중요하다.

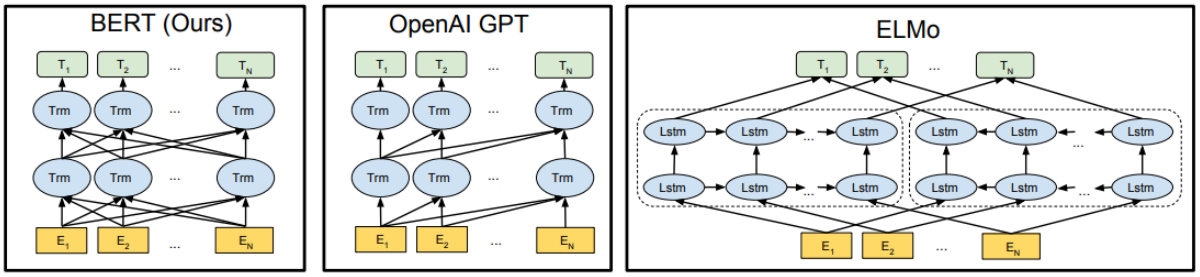
이 논문에서, BERT: **B**idirectional **E**ncoder **R**epresentations from **T**ransformers를 제안함으로써 미세 조정 기반 접근 방법을 향상시킨다. BERT는 Cloze 업무에 의해 영감을 받은 사전 훈련을 목적으로 하는 '마스킹된 언어 모델(masked language model, MLM)'을 사용함으로 이전에 언급된 단방향성 제약을 완화시킨다. 마스킹된 언어 모델은 무작위로 입력으로부터의 토큰 일부를 마스킹하고 목적은 오직 그 맥락 안에서 마스킹된 단어 기반으로 원래의 단어 ID를 예측하는 것이다. '왼쪽에서 오른쪽으로'의 언어 모델 사전 훈련과 달리, MLM의 목적은 표현이 왼쪽과 오른쪽 맥락을 융합할 수 있게 하며, 이것은 우리가 깊은 양방향 트랜스포머를 사전 훈련할 수 있게 한다. 마스킹된 언어 모델 외에도 우리는 사전 훈련된 텍스트 쌍 표현을 공동으로 하는 "다음 문장 예측" 작업 또한 사용한다. 이 논문의 기여는 다음과 같다.

- 우리는 언어 표현을 위한 양방향 사전 훈련의 중요성을 증명한다. 사전 훈련을 위해 단방향 언어 모델을 사용한 Radford et al. (Improving language understanding with unsupervised learning. 2018)과 달리 BERT는 깊은 양방향 표현을 사전 훈련이 가능하도록 마스킹된 언어 모델을 사용한다. 이것은 또한 독립적으로 훈련된 '왼쪽에서부터 오른쪽으로'와 '오른쪽에서부터 왼쪽으로' 언어 모델의 얕은 결합을 사용한 Peters et al. (Deep contextualized word representations. 2018a)와 대조적이다.

- 우리는 사전 훈련된 표현이 많은 고도로 설계된(heavily-engineered) 작업별 아키텍처의 필요성이 줄어든다는 것을 보여준다. BERT는 문장 수준 및 토큰 수준 작업으로 이루어진 대규모 집합에서 state-of-the-art 성능을 달성해 많은 작업별 구조를 능가하는 최초의 미세 조정 기반 표현 모델이다.
- BERT는 11개의 NLP 작업에서 최첨단으로 발전시켰다

BERT는 양질의 pre-trained language representation를 얻는 것 과 **down-stream task로의 손쉬운 fine-tuning**에 중점을 둠

- 기존 down stream task에 pre-trained language representation를 적용하는 방법



1. **feature based approach**

기존의 input에 pretrained representation를 feature로서 추가하여 함께 사용하는 방법 ⇒ 임베딩은 그대로 두고 그 위의 레이어만 학습하는 방법

▼ 예시(ELMo)

task-specific architecture를 사용하고, pre-trained representation을 추가적인 feature로 사용

ELMo가 **순방향 언어 모델과 역방향 언어 모델을 모두 사용**하기 때문에 **Bidirectional lanuage model**이라고 생각할 수 있지만, **ELMo는 각각의 단방향(순방향,역방향) 언어모델의 출력값을 concat⇒트랜스포머가 아닌 LSTM을 사용해서 사용하기 때문에 하나의 모델 자체는 단방향**이다. 단방향으로 두번 실행하는 구조임. 이것이 바로 BERT에서 강조하는 **deep bidirectional과의 차이점**

2. fine-tuning approach

fine-tuning based approach는 최소한의 task-specific parameter만을 추가하고, 모든 pretrained parameter를 down-stream task에서 fine-tuning 하는 방법

⇒ 모든것을 미세하게 업데이트. 임베딩 까지 모두 업데이트

▼ 예시(OpenAI의 gpt)

최소한만의 task-specific feature를 사용한다. downstream task에 활용하기 위해선 그저 사전 학습된 모든 parameter를 fine-tuning하면 된다.

단방향만 고려하기 때문에 다음단어 예측은 잘 하지만 그 뒤 혹은 앞을 알 수 없음. 디코더만 사용

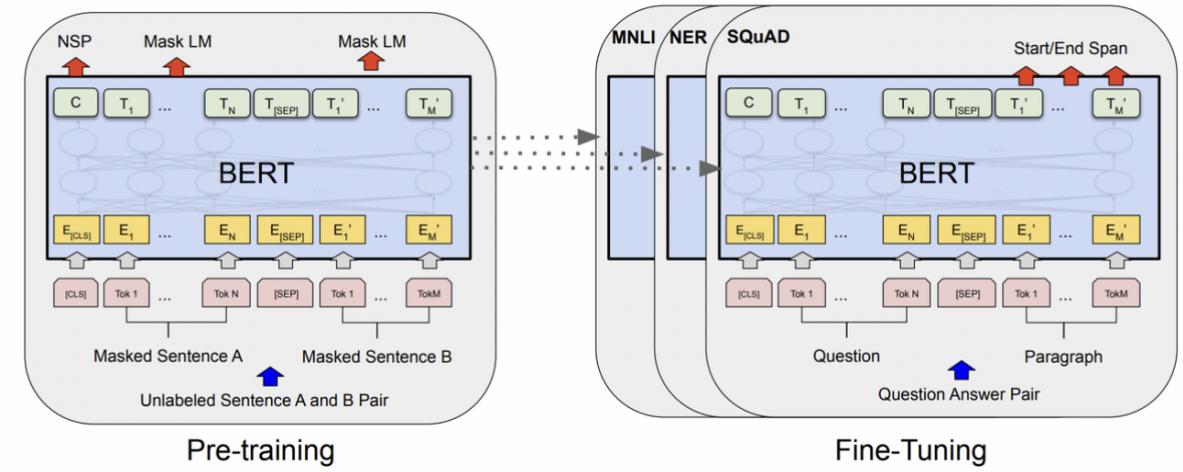
이 두 방식의 공통적인 문제점은 standard language model들이 unidirectional하다는 것 →MLM로 해결

Related Work

▼ 본문내용

사전 훈련하는 일반적인 언어 표현에는 긴 역사가 있으며, 우리는 간단하게 가장 많이 쓰이는 접근법들을 이 섹션에서 리뷰한다

BERT



Bert 자체를 학습시키는 방법

pre-training 단계, fine tuning 단계로 두가지 단계로 구분한다.

Pre-training part: **다양한 pre-training tasks의 unlabled data를 활용해 초기 파라미터를 설정**

Fine-tuning part: 이를 바탕으로 학습된 모델을 **downstream tasks의 labled data를 이용해 fine-tuning**.

⇒ 즉 레이블이 없는 방대한 데이터로 사전 훈련된 모델을 만든 후 , 레이블이 있는 다른 작업에서 추가 훈련과 함께 하이퍼파라미터를 재조정하여 이 모델을 사용하여 높은 성능을 얻음.

이때, 각각의 downstream task에 대한 pre-trained parameter가 같더라도 각 task는 각각의 fine-tuned model을 가진다.

또한, **pre-trained architecture와 fine-tuned 된 downstream architecture에는 큰 구조적 차이가 없으며,** 이는 BERT의 독특한 특징인 **다양한 task에 걸친 unified architecture**라는 점에서 기인한다.

- **Model Architecture (모델 구조)**

Bert의 모델 아키텍처는 **multi-layer bidirectional Transformer encoder**이다.

즉, **양방향 Transformer encoder를 여러 층 쌓은 것**

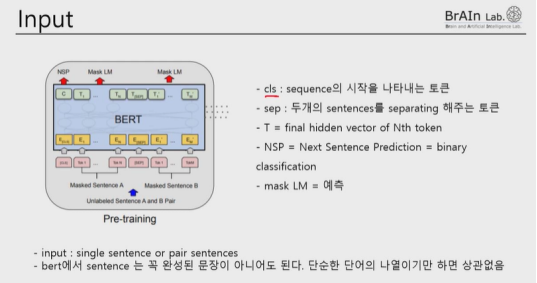
BERT base의 경우 L = 12, H = 768, A = 12로 총 110M개의(약 1억1천만) 파라미터를 사용하였고,

BERT large의 경우 L = 24, H = 1024, A = 16으로 총 340M개의(약 3억4천만) 파라미터를 사용하였다.

Input/Output Representations

Input 구조

$$h_0 = UW_e + W_p + W_s$$



T: 인코더의 히든레이어 다 거치고 마지막으로 나온 벡터

입력: 문장을 끊어서, 혹은 문단 단위로 넣어도 됨(단어의 나열이기만 하면됨)

Transformer의 방법을 그대로 따르기 때문에 역시 **Token embedding** 후에 **Positional embedding**을 더하여 줍니다.

여기서 포인트는 Ws(sequence embedding)인데 각각의 문장을 구분해 주기 위한 **Segment embedding**을 더해 줍니다.

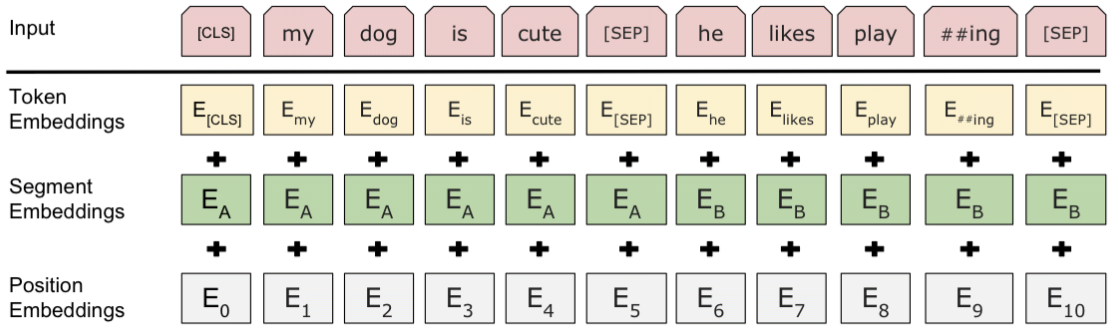
저자들은 Segment embedding를 **학습 가능**한 parameter로 설정 (해당 token이 어느 sentence에 포함되어있는지에 대한 embedding으로 생각하면 쉬울듯)

BERT가 다양한 down-stream tasks에 잘 적용되기 위해선 input representation이 애매하지 않아야 한다.

하나의 문장 혹은 한 쌍의 문장을 하나의 토큰 시퀀스로 분명하게 표현해야한다.

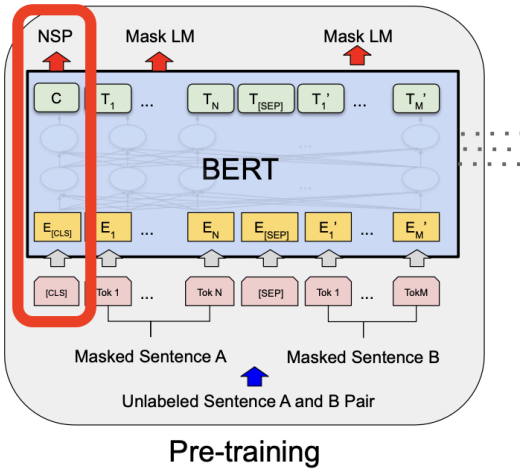
> 이때 'sentence'는, 언어학적인 sentence 즉 문장의 의미 뿐만 아니라 인접한 텍스트들의 임의의 범위라는 뜻도 포함

이를 위해 **BERT에서는 *총 3가지의 Embedding vector*를 합쳐서 input으로 사용**한다



- **pre-training 과정 (MLM,NSP)**

BERT의 Pre-training 과정에서는 크게 **두가지 unsupervised-tasks**가 사용되는데, 바로 **Masked language model(MLM)**과 **Next Sentence Prediction(NSP)**이다.



Pre-training

MLM (Masked Language Model)

input에서 token들을 랜덤하게 mask해서 model이 masked된 단어를 문맥(context)으로부터 추측하게 하는 pre-training objective

즉, 전체 문장을 reconstruction 하며 학습하지 않고, 일정 비율(본 논문은 15%)로 Masking된 임의의 단어 토큰만을 예측하며 학습함

이 방법은 방향을 갖는 조건부확률로서 언어모델을 구성하지 않음. 따라서 보다 원활한 **bidirectional pre-training**을 가능하게 해줌

단점:

pre-training 과정에서는 Masked Token이 존재하지만, 실제 **Fine-tuning 과정에서는 Masked Token이 존재하지 않음 (**pre-training 단계와 fine-tuning 단계 사이에 **mismatch**가 존재)

→pre-trained 단계에선 잘 되지만 실제 fine tuning 시 반영 잘 안됨

해결법: Masking 과정에서 Generalization을 위한 Trick을 사용 → Masking 하도록 정해진 토큰에 대하여
확률적인 variation을 주는 것

논문에서는 i번째 토큰이 마스킹할 토큰으로 정해졌을 때 80프로는 마스킹하고 10프로는 랜덤 토큰으로 변경, 10프로는 기존으로 유지함

▼ 해결책 적용한 MLM예시

[80%] my dog is hairy → my dog is [mask]
[10%] my dog is hairy → my dog is apple

[10%] my dog is hairy →my dog is hairy

아까 15%로 마스크 할 확률로 선정된 후 8:1:1 비율로 또 선정되는것.

비율을 이렇게 설정한 이유는 여러번의 실험을 통해 결정

Masking Rates			Dev Set Results		
MASK	SAME	RND	MNLI	NER	
			Fine-tune	Fine-tune	Feature-based
80%	10%	10%	84.2	95.4	94.9
100%	0%	0%	84.3	94.9	94.0
80%	0%	20%	84.1	95.2	94.6
80%	20%	0%	84.4	95.2	94.7
0%	20%	80%	83.7	94.8	94.6
0%	0%	100%	83.6	94.9	94.6

Table 8: Ablation over different masking strategies.

NSP (Next Sentence Prediction)

이진분류 0,1 클래스로 분류함.

주어진 두개의 문장이 이어지는 문장인지([IsNext]) 아닌지([NotNext])를 판단하는 **BInary classification** 작업

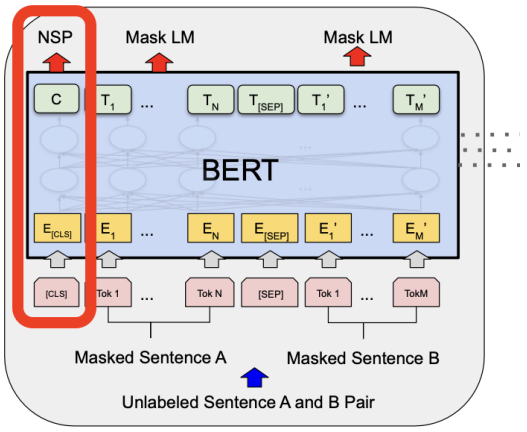
▼ 동작원리

50%는 다음에 오는 문장 (Next), 50%는 랜덤문장 (NotNext)

if 10개의 문장이 있다고 생각하자

이 과정에서는 첫번째 토큰인 [CLS] 토큰을 활용하여 output layer에서 [IsNext], [NotNext]를 예측하도록 학습합니다.

class 토큰에서만 일어남

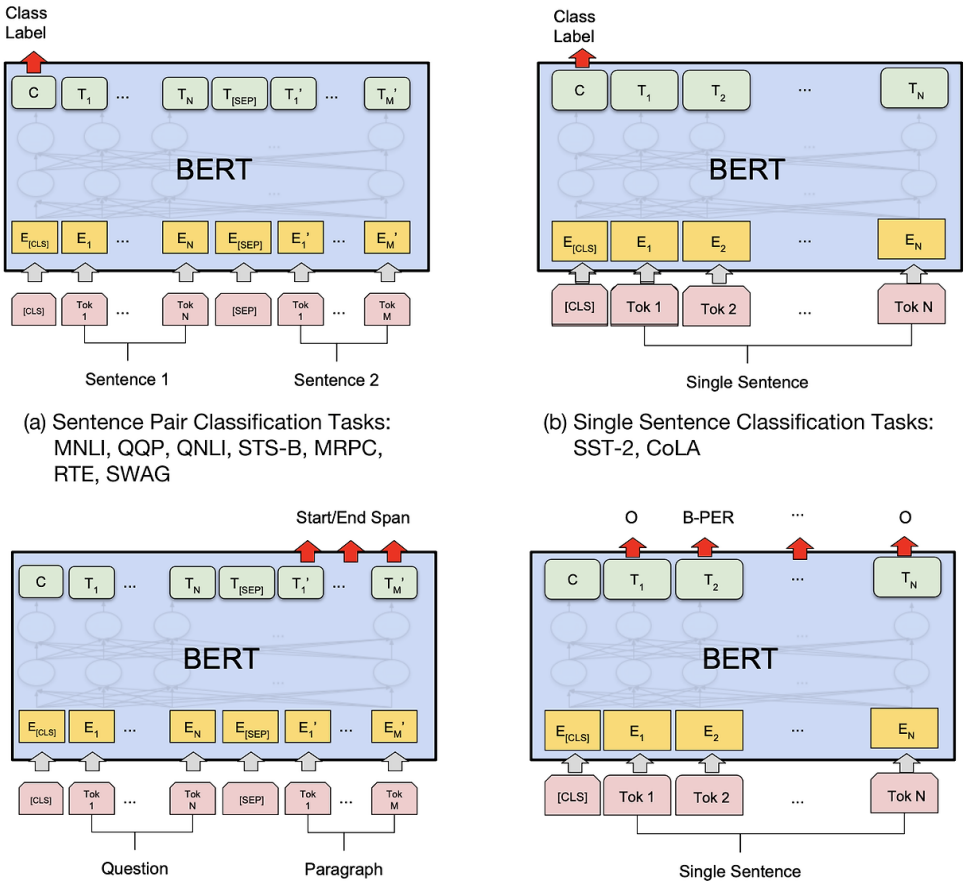


Pre-training

CLS가 첫 번째 센스가 시작하기 전에 와야 되는 토큰이라고 의미적으로 그렇게 설명했는데, 실제로 이 토큰이 가지고 있는 정보는 0 과 1에 관한 binary classification에 관한 정보를 가지고 있음.

그래서 이 토큰도 마찬가지로 이 버트 모델을 지나면서 학습이 됨. 그래서이 마지막으로 나오는 이제 클래스 토큰은 0과 1 라벨에 대한 확률 분포를 가지고 있음

Fine-tuning 방법



(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

(b) Single Sentence Classification Tasks: SST-2, CoLA

(c) Question Answering Tasks: SQuAD v1.1

(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

classification task는 NSP(next sentence prediction) 방법 그대로 [CLS] 토큰의 representation을 output layer로 넘겨주어 학습

(a): sentence 2개받아서 classification

(b): single sentence classification→ 문장 긍정부정 구분

토큰 단위의 down-stream task:토큰의 representation을 output layer로 넘겨주는 형태로 학습

(c) QA

(d) single sentence 형태소 분석

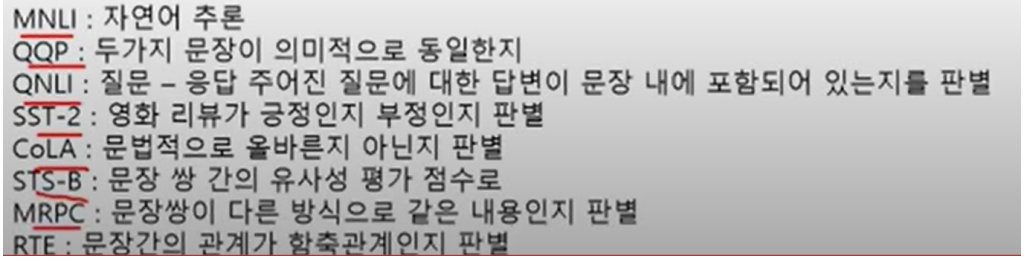
Experiments

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$BERT_{BASE}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$BERT_{LARGE}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

실험은 BERTBASE, BERTLARGE 두가지 버전으로 진행

BERTBASE: (L=12, H=768, A=12, Total Parameters=100M) →GPT와 같은 파라미터수

BERTLARGE: (L=24, H=1024, A=16, Total Parameters=340M)



모든 task에서 성능이 좋은것을 표현

Conclusion

주요 기여는 이러한 심층적인 양방향 아키텍처 구조를 찾는 데 더욱 일반적이며, 사전 훈련된 동일한 모델을 해외에서 NLP 작업 세트를 지속적으로 처리할 수 있도록 허용