NLP 파트 - BERT - # 36

BERT : Pre-trainig of Deep Bidirectional Transformers for Language Understanding

Abstract

- **BERT : Bidirectional Encoder Representations form Transformer**
  - *"Attention is all you need(Vaswani et al., 2017)"* 에서 소개한 Transformer 구조를 활용한 Language Representation에 관한 논문
  - 대용랑 unlabeled data로 모델을 미리 학습 시킨 후, 특정 task를 가지고 있는 labeled data로 transfer learning(전이 학습)을 하는 모델
  - BERT 이전의 모델의 접근 방식은 *shallow bidirectional* (얕은 양방향성) 또는 *unidirectional* (단방향성)
  - BERT는 모델 자체의 fine-tuning을 통해 해당 task의 *State-Of-The-Art(SOTA)* 달성

1. Introduction

1.1 pre-trained language representation 적용 방식

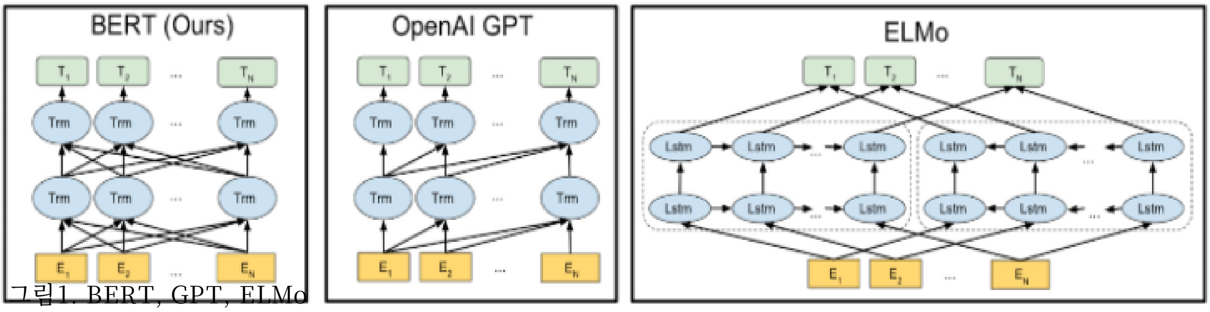

그림1. BERT, GPT, ELMo

- **feature-based approach**
  - pre-train 후 단어에 대한 embedding 벡터 값이 고정
  - 임베딩은 그대로 두고 그 위에 레이어만 학습 하는 방법
  - ex. ELMo

- **fine-tuning approach**
  - pre-train 후 단어에 대한 embedding 벡터 값이 고정되지 않음
  - 임베딩까지 모두 업데이트하는 기법
  - ex. OpenAI GPT

- **한계점**
  - 일반적인 langauge model(ELMo, GPT)은 단방향(unidirectional) 혹은 얕은 양방향(shallow bidirectional)
  - 이전 Language Model은 left-to-right 또는 rigt-to-left 구조를 사용하여 모든 토큰을 오직 이전 토큰만 고려한다는 문제

1.2 BERT의 pre-training 방법론

- **Masked Language Model(MLM)**
  - MLM은 input에서 무작위하게 몇개의 token을 mask 하고 이를 Transformer 구조에 넣어서 주변 단어의 context만을 보고 mask된 단어를 예측하는 모델

- **Next Sentence Prediction(NSP)**
  - 두 문장을 pre-training시에 같이 넣어줘서 두 문장이 이어지는 문장인지 아닌지 예측하는 것

2. Related Work

- ELMo, OpenAI GPT와 같은 모델 존재

3. BERT

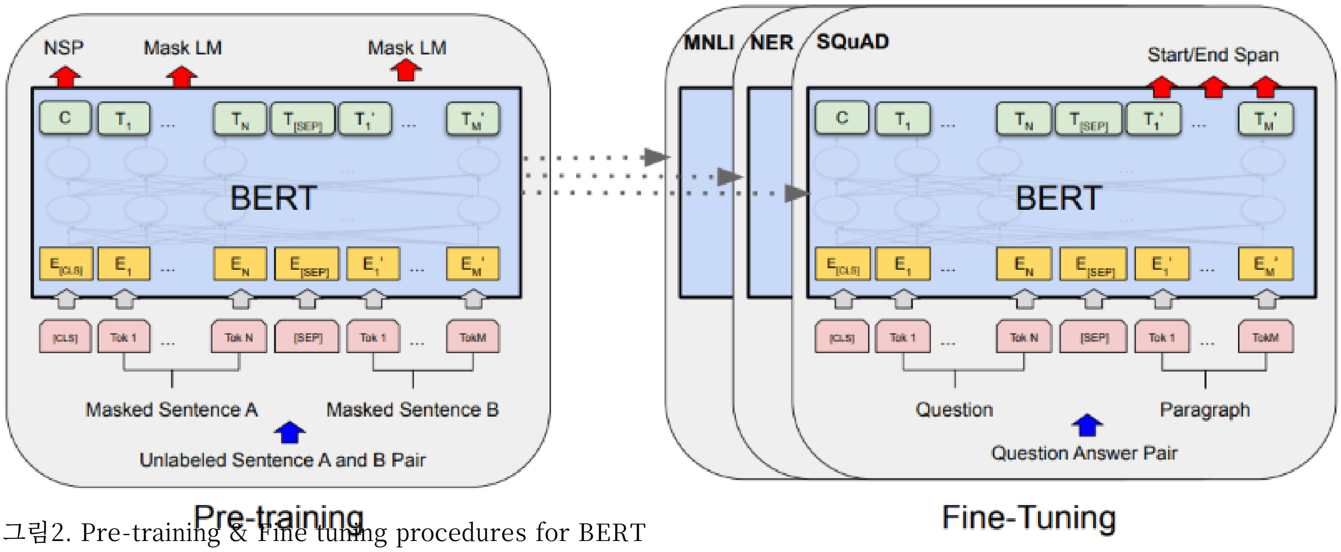

그림2. Pre-training & Fine tuning procedures for BERT

- BERT의 아키텍처는 Transformer를 사용하지만, pre-training과 fine-tuning시의 아키텍처를 조금 다르게하여 Transfer Learning(전이학습)을 용이하게 함

3.1 Model Architecture

- BERT는 transformer 중에서도 encoder 부분만을 사용
- BERT는 모델의 크기에 따라 base 모델과 large 모델을 제공

```
# BERT_base : L=12, H=768, A=12, Total Parameters = 110M
# BERT_large : L=24, H=1024, A=16, Total Parameters = 340M
# L : 인코더 layer 수, H : 은닉 유닛, A : self-attention heads 수
```

3.2 Input Representation

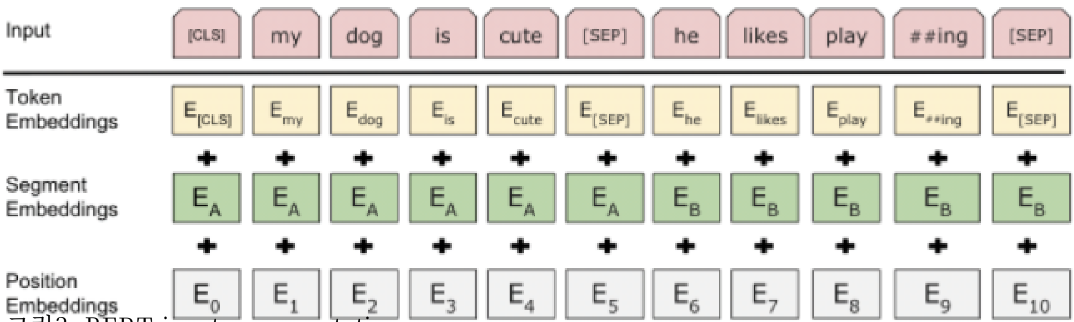

그림3. BERT input representation

3.2.1 3가지 embedding

sentence A : Paris is a beautiful city
sentence B : I love Paris

- **Token embedding**
  - [CLS] 토큰은 첫 번째 문장의 시작 부분에만 추가
  - [SEP] 토큰은 모든 문장의 끝에 추가

```
tokens = [Paris, is, a, beautiful, city, I, love, Paris]
tokens = [[CLS], Paris, is, a, beautiful, city, I, love, Paris]
tokens = [[CLS], Paris, is, a, beautiful, city, I, love, Paris, [SEP]]
```

- **Segment embedding**
  - Segment embedding은 주어진 두 문장을 구별하는데 사용

```
tokens = [[CLS], Paris, is, a, beautiful, city, [SEP], I, love, Paris, [SEP]]
```

- **Position embedding**
  - 데이터를 직접 입력하기 전에 문장에서 단어(토큰)의 위치에 대한 정보를 제공
  - Position embedding 레이어를 사용해 문장의 각 토큰에 대한 위치 임베딩 출력
  - E0, E1 ⋯ , E10

3.2.2 WordPiece

Let us start pretraining the model

- **WordPiece Tokenizer**
  - WordPiece Tokenizer를 통해 개별 단어가 pre, ##train, ##ing와 같은 하위 단어로 분할
  - 단어가 어휘 사전에 없으면 그 단어를 하위 단어로 분할해 하위 단어가 어휘 사전에 있는지 확인
  - 어휘 사전 이외(OOV : Out-Of-Vocabulary)의 단어를 처리하는데 효과적

```
tokens = [let, us, start, pre, ##train, ##ing, the, model]
```

3.3 Pre-training BERT

3.3.1 Task #1: Masked Language Modeling(MLM)

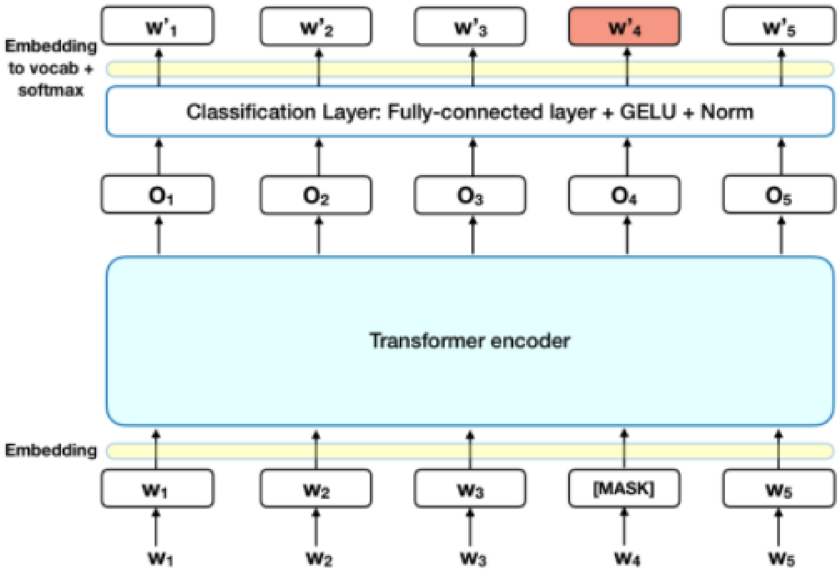

그림4. BERT Masked Language Model

- 주어진 입력 문장에서 전체 단어의 15%를 무작위로 마스킹하고 마스크된 단어를 예측하도록 모델을 학습
- 마스크된 단어를 예측하기 위해 모델은 양방향으로 문장을 읽고 마스크된 단어를 예측

```
tokens = [[CLS], Paris, is, a, beautiful, city, [SEP], I, love, Paris, [SEP]]
tokens = [[CLS], Paris, is, a, beautiful, [MASK], [SEP], I, love, Paris, [SEP]]
```

- 사전학습 시 위와 같은 방식으로 토큰을 마스킹하면 사전학습과 파인 튜닝 사이에 불일치 발생
- 문제를 극복하기 위해 15% 토큰에 대해 **80-10-10%** 규칙 적용
  - 15% 중 80%의 토큰(실제 단어)을 [MASK] 토큰으로 교체

  ```
  tokens = [[CLS], Paris, is, a, beautiful, [MASK], [SEP], I, love, Paris, [SEP]]
  ```

  - 15% 중 10%의 토큰(실제 단어)을 임의의 토큰(임의 단어)으로 교체

  ```
  tokens = [[CLS], Paris, is, a, beautiful, love, [SEP], I, love, Paris, [SEP]]
  ```

  - 15% 중 10%의 토큰은 어떤 변경도 하지 않음

  ```
  tokens = [[CLS], Paris, is, a, beautiful, city, [SEP], I, love, Paris, [SEP]]
  ```

3.3.2 Task #2: Next Sentence Prediction(NSP)

- NSP task에서는 BERT에 두 문장을 입력하고 두 번째 문장이 첫 번째 문장의 다음 문장인지 예측(이진 분류 작업)
- NSP task를 수행함으로써 두 문장 사이의 관계를 파악하여 Question & Answering과 같은 down-stream task에 유용

  sentence A : She cooked pasta
  sentence B : It was delicious

```
input  =  [[CLS],  She,  cooked,  ##ed,  [MASK],  [SEP],  It,  was,  [MASK],  [SEP]]
label  =  isNext
```

  sentence C : Turn the radio on
  sentence D : She bought a new hat

```
input  =  [[CLS],  Turn,  the,  [MASK],  on,  [SEP],  She,  bought,  a,  new,  [MASK],  [SEP]]
label  =  notNext
```

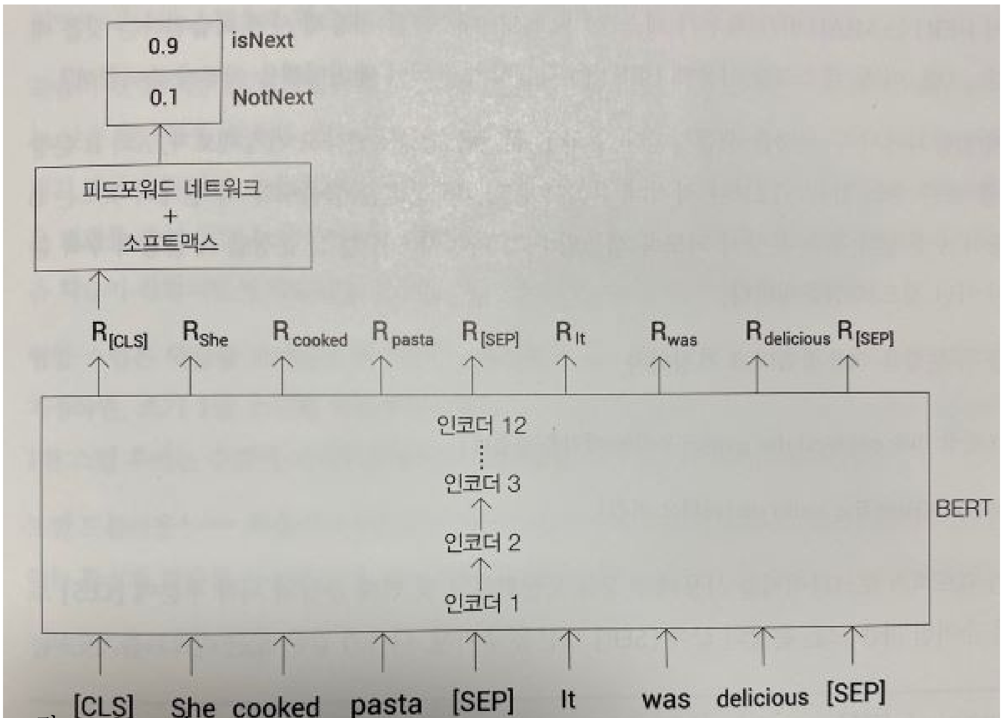

그림5. NSP task

- [CLS] 토큰 표현을 가져와 softmax 함수를 사용해 피드포워드 네트워크에 입력
- 문장 쌍이 isNext인지 notNext인지에 대한 확률값 반환

3.4 Fine-tuning BERT

- **Single Text Classification**
  - 하나의 텍스트에 대한 텍스트 분류 유형
  - 영화 리뷰 감성 분류, 로이터 뉴스 분류 등과 같이 입력된 문서에 대해서 분류를 하는 유형

- **Tagging**
  - 하나의 텍스트에 대한 태깅 작업
  - 대표적으로 문장의 각 단어에 품사를 태깅하는 품사 태깅 작업과 개체를 태깅하는 개체명 인식 작업

- **Text Pair Classification or Regression**
  - 텍스트의 쌍에 대한 분류 또는 회귀 문제
  - 텍스트의 쌍을 입력으로 받는 대표적인 태스크로 자연어 추론(Natural language inference)
  - 자연어 추론 문제란, 두 문장이 주어졌을 때, 하나의 문장이 다른 문장과 논리적으로 어떤 관계에 있는지를 분류

- **Question Answering**
  - 질문과 본문을 입력받으면, 본문의 일부분을 추출해서 질문에 답변하는 질의 응답
  - 태스크의 대표적인 데이터셋 SQuAD(Stanford Question Answering Dataset)v1.1

4. Experiments

4.1 GLUE

System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
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	79.9	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	88.1	91.3	45.4	80.0	82.3	56.0	75.2
BERT$_{\text{BASE}}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
BERT$_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

그림6. GLUE results

4.2 SQuAD v1.1

System	Dev		Test	
	EM	F1	EM	F1
Leaderboard (Oct 8th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
#1 Single - nlnet	-	-	83.5	90.1
#2 Single - QANet	-	-	82.5	89.3
Published				
BiDAF+ELMo (Single)	-	85.8	-	-
R.M. Reader (Single)	78.9	86.3	79.5	86.6
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
Ours				
BERT$_{\text{BASE}}$ (Single)	80.8	88.5	-	-
BERT$_{\text{LARGE}}$ (Single)	84.1	90.9	-	-
BERT$_{\text{LARGE}}$ (Ensemble)	85.8	91.8	-	-
BERT$_{\text{LARGE}}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
BERT$_{\text{LARGE}}$ (Ens.+TriviaQA)	**86.2**	**92.2**	**87.4**	**93.2**

그림7. SQuAD v1.1 results

4.3 SQuAD v2.0

System	Dev		Test	
	EM	F1	EM	F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	86.3	89.0	86.9	89.5
#1 Single - MIR-MRC (F-Net)	-	-	74.8	78.0
#2 Single - nlnet	-	-	74.2	77.1
Published				
unet (Ensemble)	-	-	71.4	74.9
SLQA+ (Single)	-		71.4	74.4
Ours				
BERT$_{\text{LARGE}}$ (Single)	78.7	81.9	80.0	83.1

그림8. SQuAD v2.0 results

4.4 SWAG

System	Dev	Test
ESIM+GloVe	51.9	52.7
ESIM+ELMo	59.1	59.2
OpenAI GPT	-	78.0
$BERT_{BASE}$	81.6	-
$BERT_{LARGE}$	**86.6**	**86.3**
Human (expert)†	-	85.0
Human (5 annotations)†	-	88.0

그림9. SWAG results

출처

BERT 논문 https://arxiv.org/abs/1810.04805
구글 BERT의 정석

파일

BERT논문.pdf	757 KB	2022-03-14	신재영
clipboard-202203141600-86u23.png	110 KB	2022-03-14	신재영
clipboard-202203141603-xvfww.png	84.2 KB	2022-03-14	신재영
clipboard-202203141603-i21sy.png	41.8 KB	2022-03-14	신재영
clipboard-202203141604-v0dbf.png	80.8 KB	2022-03-14	신재영
clipboard-202203141605-zfqbu.png	127 KB	2022-03-14	신재영
clipboard-202203141605-gt2aw.png	70 KB	2022-03-14	신재영
clipboard-202203142227-o2bxv.jpg	259 KB	2022-03-14	신재영
clipboard-202203142311-f5ihy.png	122 KB	2022-03-14	신재영
clipboard-202203150000-st6zg.png	26.1 KB	2022-03-14	신재영