Bert
(Bidirectional Encoder Representation from Transformers)

이현재

Contents

Introduction

- Unsupervised Pre-training
  - Unlabeled data로 사전 학습
- Supervised Fine-tuning
  - 사전 학습된 모델의 parameter로 초기화한 후 down-stream task의 labeled data로 parameter를 tuning
  - 각각의 down-stream task(하위)는 동일한 pre-trained parameter로 초기화하며 개별적인 fine-tuned model을 가진다
  - 정의 : 다른 작업에 대해서 parameter 재조정을 위한 추가 훈련 과정
- Bert가 높은 성능을 얻을 수 있었던 이유
  - 레이블이 없는 방대한 데이터로 사전 훈련된 모델을 가지고, 레이블이 있는 다른 작업(Task)에서 추가 훈련과 함께 하이퍼 파라미터를 재조정하여 이 모델을 사용하면 성능이 높게 나오는 기존의 사례들을 참고
  - BERT는 특정 과제를 하기 전 사전 훈련 Embedding을 통해 특정 과제의 성능을 더 좋게 할 수 있는 언어 모델
    - pre-trained language representation을 down-stream tasks에 적용하기 위한 두 가지 approach가 있다. feature-based approach와 fine-tuning approach이다.
    - **feature-based approach**는 pre-trained representation을 추가적인 features로 task-specific architectures에 포함한다. (ex. ELMo)
    - **fine-tuning approach**는 모든 Pre-trained parameters를 down-stream tasks에 학습시키고, 최소한의 task-specific parameters를 도입한다. (ex. OpenAI GPT)

Introduction

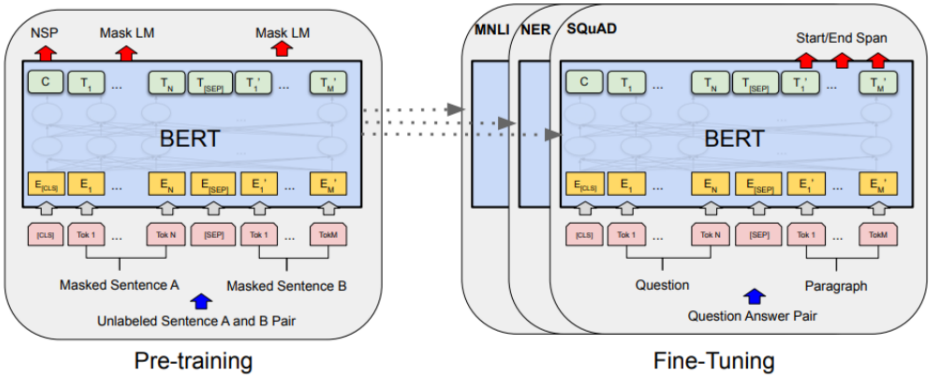
- 양방향성 포함
  - ‘Bidirectional’에서 볼 수 있듯이 Bert 모델은 양방향성 특성을 가지고 있다.
  - Bert 이전의 대부분의 모델은 문장이 존재하면 왼쪽에서 오른쪽으로 진행하여 문맥(Context)를 파악하는 방법을 취했기 때문에 전체 문장을 파악하는데 있어서 제한될 수 밖에 없는 한계점을 가지고 있다.
  - (저는 이해가 안되지만) **'나는 하늘이 예쁘다고 생각한다'**라는 문장을 이해할 때, 단순히 **'하늘'**이라는 명사를 정해놓고 **'예쁘다'**라는 표현을 사용하지는 않습니다. **'예쁘다'**를 표현하고 싶어서 **'하늘'**이라는 명사를 선택했을 수도 있습니다. 즉, 앞에서 뒤를 볼 수도 있지만 뒤에서 앞을 보는 경우도 충분히 이해할 수 있어야 전체 맥락을 완전히 파악할 수 있다.

단방향성, 양방향성

  - (저는 이 예시로 이해했습니다.) 긍정적인 의미의 ‘잘했다’와 부정적인 의미의 ‘잘했다’를 파악하기 위해서는 앞뒤 맥락을 확인해야 한다.
  - A : “나 오늘 주식 올랐어 “ B : “진짜 잘했다” (긍정)
  - A : “나 오늘 주식으로 망했어” B : “진짜 잘했다” (부정)

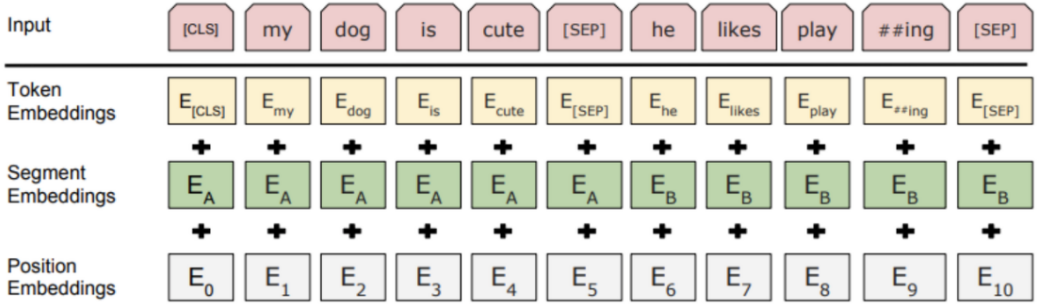
Bert

- 기본 구조 : Transformer의 Encoder를 쌓아 올린 구조
  - Transformer Encoder 층의 수를 L, d_model의 크기를 D, 셀프 어텐션 헤드의 수를 A라고 하였을 때,
  - BERT-Base : L=12, D=768, A=12 : 110M개의 파라미터 (이 부분만 중점적으로 다루겠습니다.)
  - BERT-Large : L=24, D=1024, A=16 : 340M개의 파라미터
- 모델 학습 과정
  - Bert를 사용 x : 분류를 원하는 데이터 → LSTM, CNN 등의 머신러닝 모델 → 분류
  - Bert를 사용 : 관련 대량 corpus → Bert → 분류를 원하는 데이터 → LSTM, CNN 등의 머신러닝 모델 → 분류
  - Bert는 대량의 코퍼스를 Encoder가 Embedding하고(언어모델링), 이를 전이하여 Fine-tuning하고 Task를 수행
  - 대량 코퍼스로 Bert 언어모델을 적용하고, Bert 언어 모델 출력에 추가적인 모델(RNN, LSTM 등 DNN까지도)을 쌓아 원하는 Task를 수행



언어모델링(Pre-training), NLP Task(FIne-tuning)

Bert의 Input representation



Input Representation - BERT paper

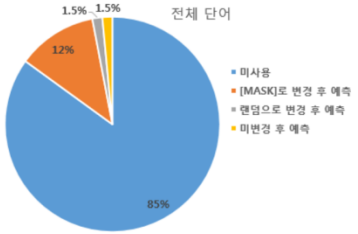
- Token Embedding
  - Word Piece Embedding 방식 사용 : 자주 등장하면서 가장 긴 길이의 sub-word를 하나의 단위로 만듦. 자주 등장하는 단어(sub-word)는 그 자체가 단위가 되고, 자주 등장하지 않는 단어(rare word)는 더 작은 sub-word로 쪼개짐. 이렇게 하게 되면 이전에 자주 등장하지 않은 단어를 모두 Out-Of-Vocabulary(OOV)로 처리하여 모델링의 성능을 저하했던 문제를 해결 가능
  - CLS : 입력 받은 모든 문장의 시작 토큰으로 삽입, 이 토큰은 Classification task에서는 사용되지만 그렇지 않을 경우에는 무시
  - SEP : 첫 번째 문장과 두 번째 문장을 구분. 각 문장의 끝에 삽입 (문장을 구분하기 위해)
- Segment Embedding
  - Segment Embedding을 통해 앞뒤 문장을 더욱 쉽게 구별할 수 있도록 도와줌
  - 토큰으로 나누어진 단어들을 다시 하나의 문장으로 만들고 첫 번째 [SEP] 토큰까지는 0으로 그 이후 [SEP] 토큰까지는 1 값으로 마스크를 만들어 각 문장들을 구분

Bert의 Input representation

- Positional Embedding
  - Transformer 구조에서도 사용된 방법으로 각 토큰의 위치를 알려주는 Embedding
  - 최종적으로 세 가지 Embedding을 더한 Embedding을 Input으로 사용

Bert의 Pre-training

- Bert는 문장 표현을 학습하기 위해 두 가지 Unsupervised 방법을 사용
  - Masked Language Model (MLM)
  - Next Sentence Model
- Masked Language Model (MLM)
  - 문장에서 단어 중의 일부를 Mask 토큰으로 바꾼 뒤, 가려진 단어를 예측하도록 학습. 이 과정에서 Bert 모델은 문맥을 파악하는 능력을 기르게 된다.
  - **Mask 토큰만을 예측하는 pre-training을 진행**
  - 입력 텍스트의 15%의 단어를 다음과 같은 규칙으로 Masking을 진행
    - 80%의 단어들은 Mask로 변경 ex) The man went to the store → The man went to the [MASK]
    - 10%의 단어들은 랜덤으로 단어가 변경 ex) The man went to the store → The man went to the dog
    - 10%의 단어들은 동일하게 둔다.
    - 이렇게 하는 이유는 Mask만 사용할 경우에는 Mask 토큰이 Fine-tuning 단계에서는 나타나지 않으므로 사전 학습 단계와 Fine-tuning 단계에서의 불일치가 발생하는 문제가 있을 수도 있기 때문에.
    - 이 문제를 완화하기 위해서 랜덤으로 선택된 15%의 단어들의 모든 토큰을 Mask로 사용하지 않는다.



https://wikidocs.net/115055 예시는 여기에서 참고하세요...!

Bert의 Pre-training

- Next Sentence Prediction (NSP)
  - 두 문장의 관계를 이해하기 위해 Bert의 학습 과정에서 두 번째 문장이 첫 번째 문장의 바로 다음에 오는 문장인지 예측하는 방식
  - 문장 A와 B를 이어 붙이는데, B는 50% 확률로 관련 있는 문장(isNext label) 또는 관련 없는 문장(NotNext label)을 사용
  - 이 두 문장이 실제 이어지는 문장인지 아닌지를 [CLS] 토큰의 위치의 출력층에서 이진 분류 문제를 풀도록 한다. [CLS] 토큰은 BERT가 분류 문제를 풀기 위해 추가된 특별 토큰이다.
  - Bert가 언어 모델 외에도 다음 문장 예측이라는 태스크를 학습하는 이유는 Bert가 풀고자 하는 태스크 중에서는 QA(Question Answering)나 NLI(Natural Language Inference)와 같이 두 문장의 관계를 이해하는 것이 중요한 task들이 있기 때문

Bert fine-tuning code : https://ebbnflow.tistory.com/164?category=895676

Bert의 Fine-tuning

하나의 텍스트에 대한 텍스트 분류 유형의 Fine-tuning 기법 : https://wikidocs.net/115055

- Task 혹은 모델에 따라서 parameter를 재조정
- Supervised learning

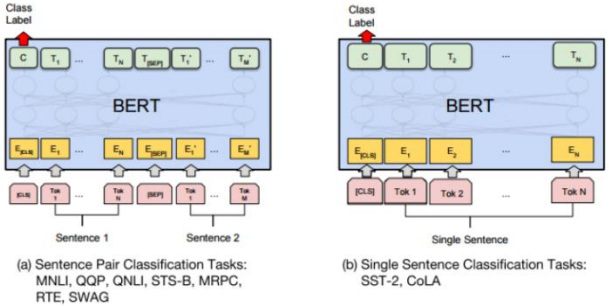


Figure 4: Illustrations of Fine-tuning BERT on Different Tasks.

(a), (b) : sequence level task
(c), (d) : token level task

(a) : 두 개의 문장에 대한 분류 task
(b) : 하나의 문장에 대한 분류 task
(c) : 질문과 응답에 대한 task
(d) : 각 토큰에 대해 분류 task

이렇게 task에 따라 fine-tuning을 한다

Bert의 한계

- Bert는 일반 NLP 모델에서 잘 작동하지만, Bio, Science, Finance 등 특정 분야의 언어 모델에 사용하려면 잘 적용되지 않는다고 한다. 사용 단어들이 다르고 언어의 특성이 다르기 때문이라고 한다. 따라서 특정 분야에 대해 Bert를 적용하려면 특정 분야의 특성을 수집할 수 있는 언어데이터들을 수집하고, 언어 모델 학습을 추가적으로 진행해주어야 한다.

- 한국어와 같이 복잡한(?) 언어에 대해서는 적용하기 힘들다는 이야기...!

Q&A