



BERT4REC

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding(2019)

BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer(2019)

BERT

개요

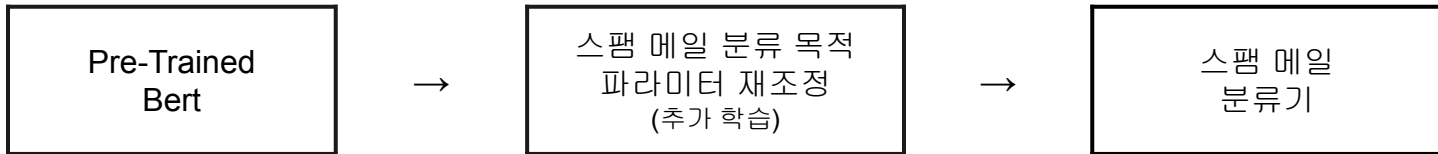
- Transformer(이하 Trm) 인코더를 통해 구현, 레이블이 없는 텍스트 데이터로 사전 훈련된 언어모델
- 다른 작업에 대해서 파라미터 재조정을 위한 추가 훈련 과정을 통해 특정 Task를 수행

- Fine-Tuning vs Feature-based

Fine-Tuning: 입력 임베딩부터 Layer까지 모든 파라미터를 재조정

Feature-based: 입력 임베딩은 제외하고 Layer에 있는 모든 파라미터를 재조정

- Bert는 레이블이 없는 방대한 데이터로 사전 훈련된 모델을 구성(Fine-Tuning)
- 레이블이 있는 다른 작업에서 추가 훈련, 하이퍼파라미터 재조정



BERT

크기

- Bert는 Trm 인코더를 쌓아올린 구조

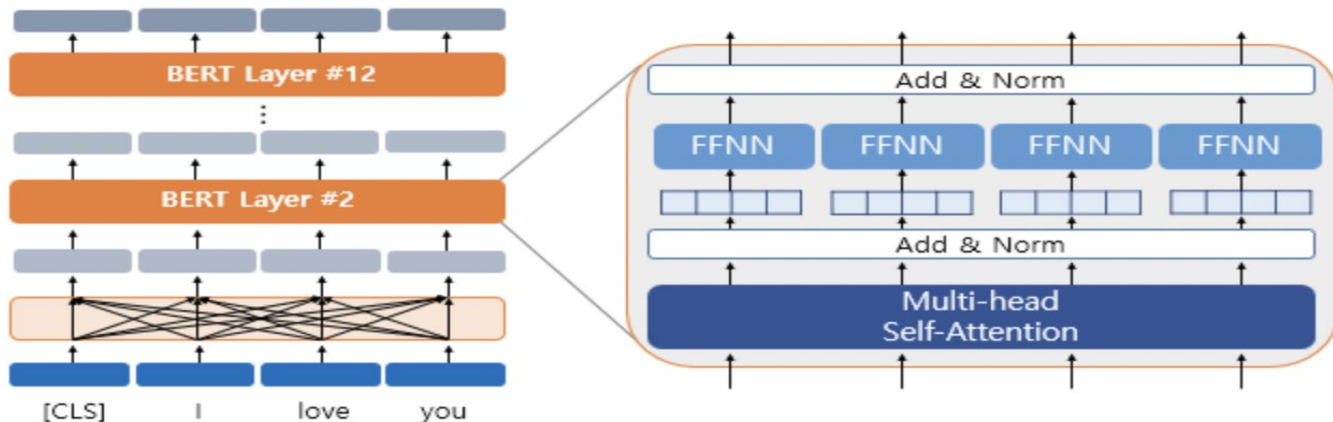
모델	Layer 수	임베딩 벡터 차원	셀프 어텐션 헤드 수
Bert-Base	12	768	12
Bert-Large	24	1,024	16

- 초기 Trm보다 큰 네트워크
- Bert-Base는 기존 Open AI GPT-1과 하이퍼 파라미터가 동일(성능 비교가 용이)
- Bert-Large는 최대 성능을 보여주기 위해 만들어진 모델

BERT

문맥을 반영한 임베딩

- 입력: 임베딩 층을 거친 768차원의 벡터(Base 모델 기준) / 출력: 문맥 정보를 모두 반영한 임베딩 벡터
- Bert Layer 1개의 동작은 Trm 인코더와 동일
- 입력 > self-attention > Feed Forward Neural Network > 출력
- 가장 큰 특징은 **self-attention**이 모든 단어들을 참고하여 문맥을 반영한 출력 임베딩 생성



BERT



WordPiece(단어 임베딩)

- Bert는 단어를 잘개 쪼개는 **WordPiece** 서브워드 토크나이저 사용
- 자주 등장하는 단어는 그대로 단어집합에 추가하고,
자주 등장하지 않는 단어는 더 작은 서브워드로 분리하여 단어 집합에 추가

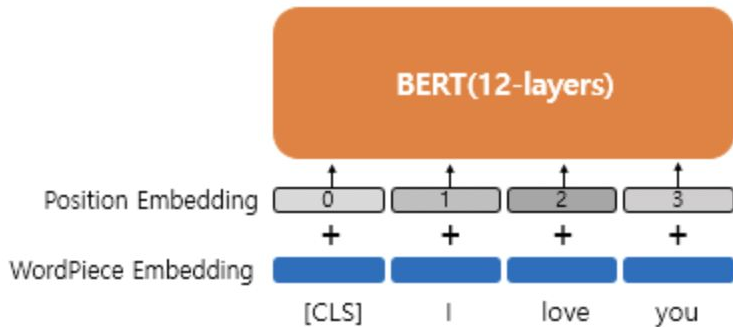
예를 들어 **embeddings**라는 단어 입력

1. 해당 단어는 단어 집합에 존재하지 않음
2. 단어 집합에 존재하는 서브워드로 분리 필수
3. **embeddings**를 ['em', '##bed', '##ding', '#s']로 분리
4. 각 서브워드를 모델 임베딩 벡터 차원에 맞춰서 임베딩 후 입력

BERT

포지션 임베딩

- 단어의 위치 정보를 표현
- 기존 Trm에서는 \sin , \cos 함수를 사용하여 위치에 따라 다른 값을 가지는 행렬을 생성
단어 벡터들과 더하는 방법
- Bert에서는 학습을 통해서 포지션 임베딩을 구함
- Bert에서는 최대 512개의 포지션 임베딩(0 ~ 51



BERT



사전 훈련

- Bert는 BookCorpus(8억 단어)와 Wikipedia(25억 단어)로 학습
- MLM과 NSP 두 가지의 훈련을 진행

1. MLM(Masked Language Model)

- 온전한 문장에서 단어 일부를 **Masking**하고 **Masking**된 단어를 예측
- 어텐션 연산이 모든 단어들에 대해 이루어지기 때문에 **Bert**는 양방향성 언어 모델

2. NSP(Next Sentence Prediction)

- 어떤 문장이 주어졌을 때 다음에 이어질 문장이 어떤 것인지 예측하기 위해
- 연결시킨 두 개의 문장이 이어진 두 문장인지 아닌지 판별

BERT



MLM

- 인공 신경망에 들어가는 입력 테스트 15%의 단어를 Masking
- 인공 신경망은 Masked 단어들을 예측
- 더 정확하게 설명하면 15%의 Masked 단어들 중
 - 80% 단어: Masking
 - 10% 단어: 랜덤으로 단어를 변경
 - 10% 단어: 동일하게 유지

정리

1. 전체 85% 단어: MLM 학습에 사용되지 않음
2. 나머지 15% 중 80%인 전체의 12%는 Masking, 이후 원래 단어를 예측
3. 나머지 15% 중 10%인 전체의 1.5%는 랜덤으로 단어 변경, 원래 단어 예측
4. 다른 1.5% 단어는 기존 단어로 유지, 그러나 모델은 변경된 단어인지는 모르기 때문에 단어 예측

BERT

NSP: QA(Question Answer), NLI(Natural Language Inference)와 같은 Task를 해결하기 위함

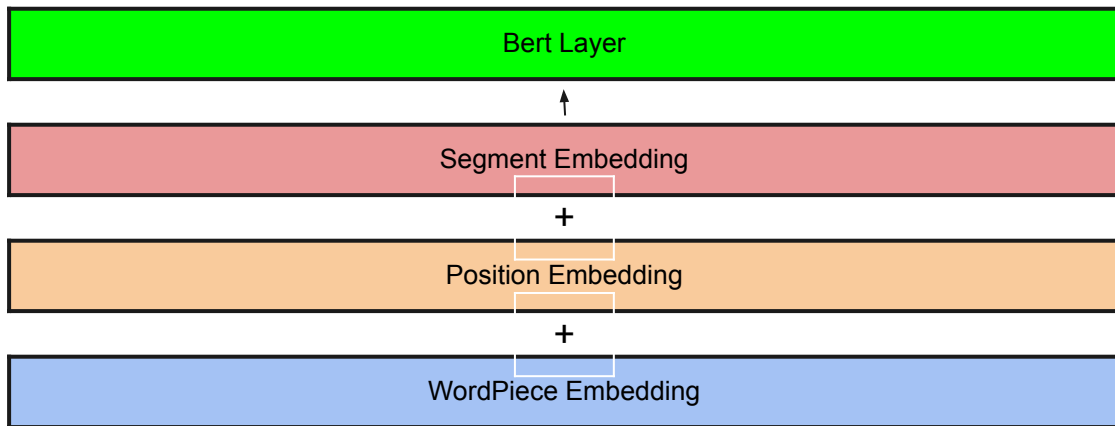
- Bert는 두 개의 문장을 준 뒤, 두 문장이 이어지는 문장인지 아닌지 판별
- 5:5비율로 이어진 두 문장(IsNextSentence)과 랜덤으로 이어붙인 두 문장을 주고 훈련(NotNextSentence)
- 문장 구분은 입력에서 [SEP] 토큰을 이용, 모든 문장 끝에 넣어줌
- 위의 이진 분류 결과는 [CLS] 토큰 출력 위치에서 연결 문장인지 여부 반환
- MLM과 NSP는 각각 학습하는 것이 아닌 동시에 학습, Loss를 더하여 학습

output	IsNextSentence or NotNextSentence		MLM Classifier					
position	0	1	2	3	4	5	...	511
subword	[CLS]	em	[Mask]	##ing	#s	[SEP]	...	[PAD]

BERT

세그먼트 임베딩

- 문장 구분을 위해 또다른 임베딩 층 구성(문장 임베딩)
- **Bert**의 입력은 3가지의 합
 - **세그먼트 임베딩**: 두 개의 문장(실제로는 두 개의 문서 혹은 단락이 될 수도 있음)을 구분하기 위한 임베딩
 - **포지션 임베딩**: 512개의 위치 정보를 구분하기 위한 임베딩
 - **단어 임베딩**: 30,522개의 단어를 구분하기 위한 임베딩



BERT



어텐션 마스크

- BERT가 [PAD] 토큰에 불필요하게 마스킹하지 않도록하기 위함
- 1: [CLS] ~ [SEP], 0: [PAD]로 값이 할당되어지며, 해당 값으로 할당된 시퀀스가 입력
- 아래는 embeddings라는 단어가 입력으로 주어졌을 때의 예

position	0	1	2	3	4	5	...	511
subword	[CLS]	em	##bed	##ing	#s	[SEP]	...	[PAD]
attention mask	1	1	1	1	1	1	0	0