sentiBERT

- BERT의 변형, 효과적으로 감정과 의미를 파악.
- 이진 구문 분석 트리를 통해 문맥화된 표현 속 의미와 구성을 파악
- sentiBERT는 문장 수준에서 감성 분류에 효과적인 기법

구문 단위로 SST로 표현된 감성 구성은

관련 작업처럼 다른 감성 분석 작업으로 바뀔 수 있다.

<예시>

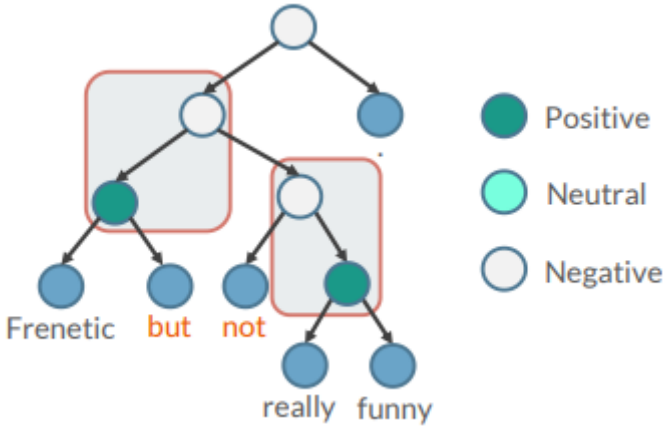
```
"Frenetic vut not really funny"
```

이 문장은 but이라는 단어로 인해 두 문장으로 나눌 수 있다. 감정의 변화를 나타내는 but

게다가, not이라는 단어는 really funny의 감성을 바꾼다(변화를 준다.)

이러한 부정과 대조의 타입은 문장들이 복잡할 때 다루기 어렵다.

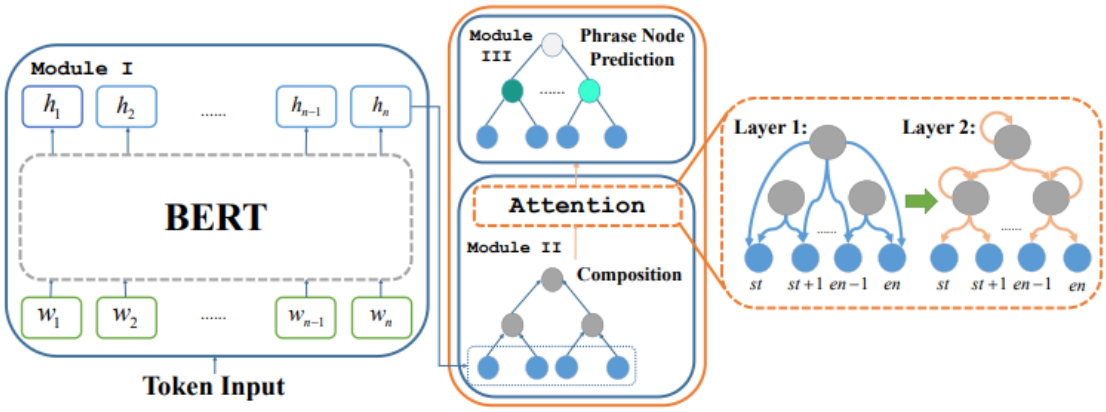
구문, 그리고 통사적으로 결합되는 방식(but)의 의미에 따라 결정된다.



- 파란색 노드 - 토큰 노드
  구문 노드의 색깔은 구문의 감정을 표현
- 빨간색 박스 - 감정은 자식노드에서 부모 노드로 변화(부정과 대조적 표현으로 인해)

-> 재귀적으로 구성된 트리 구조가 훨씬 더 구성적 감성 의미를 잘 파악한다.

sentiBERT는 재귀적 네트워크과 BERT모델을 기반으로 하여 향상되었다



SentiBERT의 구조
모듈1: BERT 인코더
모듈2: 어텐션 메커니즘에 기반한 의미 구성 모듈
모듈3: 구문 단위의 감성 해석기

의미 구성 모듈은 어텐션 기반의 네트워크로 형성된 두 개의 층으로 구성

첫번째 층 - 토큰에 기반한 각 구문의 표현을 생성
두번째 층 - 자식을 기반으로 첫번째 층으로부터 얻어낸 구문 표현을 정제

SST로 학습한 SentiBERT는 트위터 감성 분석과 감성 강도 분류, 문맥적 감성 탐색만큼 다른 관련 작업으로 변환될 수 있다

구성적 문장 감성트랜스포머 기반의 신경망 모델로 설계됨

SentiBERT는 세 개의 모듈로 구성

1. BERT
2. 어텐션 네트워크에 기반한 의미 구성 모듈
3. 이 모듈은 효과적인 구문 표현을 얻는것을 목표로 한다. 문맥화된 표현과 구성 파싱트리에 의해 레벨의 어텐션 메커니즘으로 설계
    1. 토큰에 어텐션
    2. 자식 노드에 어텐션
4. 구문과 문장단위의 감성 해석기

BERT는 입력된 문장에 대한 문맥화된 표현을 만들어주는 근간이다.

SentiBERT의 학습 목표

1. 마스크드 언어 모델
   몇개의 텍스트들이 마스크되어 모델이 그 텍스트들을 예측하도록 학습
   이 목적은 오리지널 BERT 모델에서 문맥적인 정보를 파악하도록 학습하는 것이다.
2. 구문 노드 예측
   앞서 언급한 구문 표현을 기반으로 구문 단위의 감성 레이블을 예측하기 위해 모델을 학습시킨다.

-> 이것은 SentiBERT가 구성적 감성 의미를 파악하도록 학습하게 한다.

문장 단위의 감성과 감정 분류 작업에서 파인 튜닝을 했을 때, 그 목적은 오리지널 BERT에서 [CLS]에 타겟팅하는 것보다 트리 루트가 알맞게 레이블됐다.

transferability
SST로 학습된 구성적 감성 의미는 다른 작업으로 변화될 수 있다.

마스크드 언어 모델(Masked Language Model)

입력 테스트의 단어 집합의 15%의 단어를 랜덤으로 마스킹한다.

마스킹

- 원래의 단어가 무엇이었는지 모르게 한다는 의미

인공 신경망에게 마스킹 된 단어들을 예측하도록 한다.

문장 중간에 구멍을 뚫어놓고, 구멍에 들어갈 단어들을 예측하게 하는 식

fine tuning

다른 작업에 대해서 파라미터 재조정을 위한 추가 훈련 과정

코퍼스(corpus)

자연어 데이터

- 조사나 연구 목적에 의해서 특정 도메인으로부터 수집된 텍스트 집합을 말한다.

토큰화(Tokenization)

자연어 처리에서 크롤링 등으로 얻어낸 코퍼스 데이터가 필요에 맞게 전처리되지 않은 상태라면, 해당 데이터를 사용하고자하는 용도에 맞게 토큰화(tokenization) & 정제(cleaning) & 정규화(normalization)하는 일을 하게 된다.

토큰화

- 주어진 코퍼스(corpus)에서 토큰(token)이라 불리는 단위로 나누는 작업
- 보통 의미있는 단위로 토큰을 정의

1. 단어 토큰화(Word Tokenization)

토큰의 기준을 단어로 하는 경우

의미를 갖는 문자열도 단어에 포함

입력 - Time is an illusion. Lunchtime double so!

출력 - "Time", "is", "an", "illustion", "Lunchtime", "double", "so"

보통 토큰화 작업은 단순히 구두점이나 특수문자를 전부 제거하는 정제(cleaning) 작업을 수행하는 것만으로 해결되지 않는다.

구두점이나 특수문자를 전부 제거하면 토큰이 의미를 잃어버리는 경우가 발생하기도 한다.

심지어 띄어쓰기 단위로 자르면 사실상 단어 토큰이 구분되는 영어와 달리, 한국어는 띄어쓰기만으로는 단어 토큰을 구분하기 어렵다.

토큰화에서 고려해야할 사항

1. 구두점이나 특수 문자를 단순 제외해서는 안된다.
2. 줄임말과 단어 내에 띄어쓰기가 있는 경우

2. 문장 토큰화(Sentence Tokenization)

갖고있는 코퍼스 내에서 문장 단위로 구분하는 작업 -> 문장 분류

정제(Cleaning)

갖고 있는 코퍼스로부터 노이즈 데이터를 제거

정제 작업은 토큰화 작업에 방해가 되는 부분들을 배제시키고 토큰화 작업을 수행하기 위해서 토큰화 작업보다 앞서 이루어지기도 하지만, 토큰화 작업 이후에도 여전히 남아있는 노이즈들을 제거하기 위해 지속적으로 이루어지기도 한다.

1. 규칙에 기반한 표기가 다른 단어들의 통합

2. 대, 소문자 통합

3. 불필요한 단어의 제거

4. 정규 표현식

Language Model

언어 모델

-> 주어진 문장, 단어를 바탕으로 단어에 확률을 부여하는 모델

Recurrent Neural Network

순환 신경망(RNN) - 자연어 처리에 대표적으로 사용되는 인공신경망

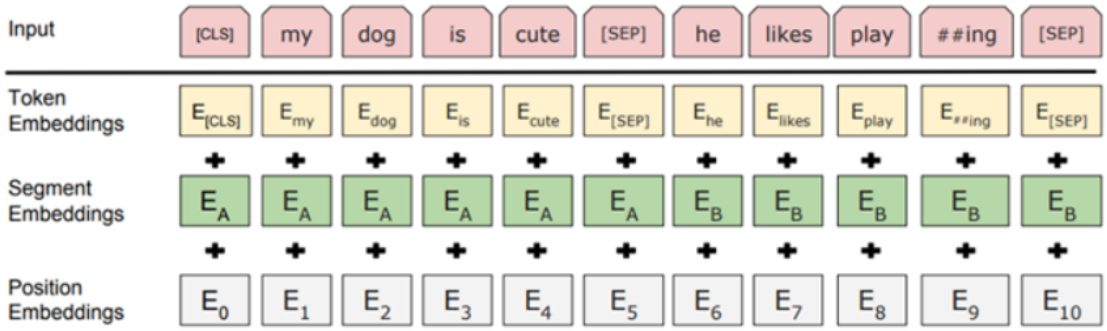
RNN은 입력과 출력을 시퀀스 단위로 처리하는 시퀀스(Sequence) 모델

BERT

사전 학습된 대용량의 레이블링 되지 않는 데이터를 이용하여 언어 모델(Language Model)ㅇ을 학습하고 이를 토대로 특정 작업(문서 분류, 질의 응답, 번역 등)을 위한 신경망을 추가하는 전이 학습 방법

사전 학습 모델

- 상대적으로 적은 자원만으로도 충분히 자연어 처리의 여러 일 수행 가능



<BERT의 input representation>

1. Token Embeddings
   - Word piece 임베딩 방식을 사용
   - Word Piece 임베딩: 자주 등장하면서 가장 긴 길이의 sub-word를 하나의 단위로 만든다.
   - 이전에 자주 등장하지 않은 단어를 전부 Out-of-vocabulary(OOV)로 처리 -> 모델링 성능 저하 문제를 해결
   - 입력받은 모든 문장의 시작으로 [CLS] 토큰이 주어진다.
   - [CLS] 토큰: 모델의 전체 계층을 다 거친 후 토큰 시퀀스의 결합된 의미를 가지게 된다.

     -> 여기에 간단한 classifier을 붙이면, 단일 문장, 또는 연속된 문장을 분류할 수 있다.

     분류 작업이 아니라면 이 토큰을 무시

     [SEP] 토큰: 문장의 구분을 위해 문장의 끝에 사용
2. Segment Embeddings
   - 토큰으로 나누어진 단어들을 다시 하나의 문장으로 만들고,
   - 첫 번째 [SEP] 토큰까지는 0으로 그 이후 [SEP] 토큰까지는 1값으로 마스크를 만들어 각 문장들을 구분한다.
3. Position Embeddings
   - 토큰의 순서를 인코딩

     -> BERT는 transformer의 encoder를 사용하는데, Transformer는 Self-Attention 모델을 사용

     Self-Attention - 입력의 위치에 대해 고려X -> 입력 토큰의 위치 정보를 주어야 함

     그래서 Transformer에서는 Sigmoid함수를 이용한 Positional encoding을 사용

BERT를 이용한 자연어 처리

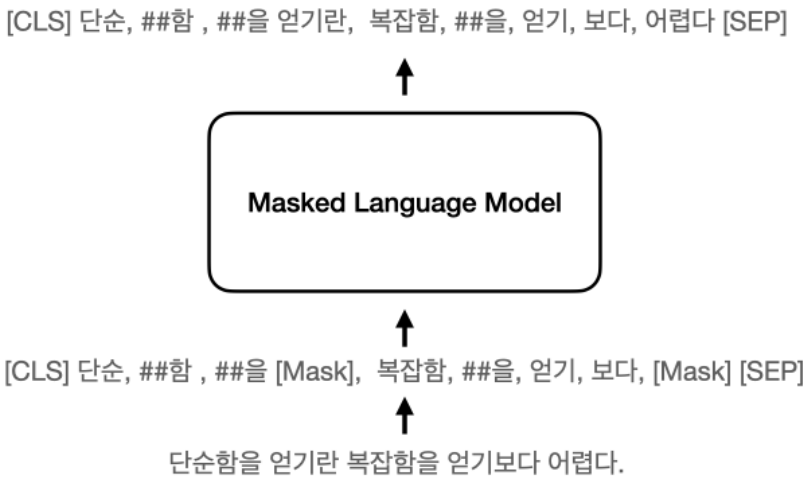
1. Pre-training

    - 거대 Encoder가 입력 문장들을 임베딩하여 언어를 모델링
2. fine-tuning
    - fine-tuning하여 여러 자연어 처리 Task를 수행

Transformer 기반의 BERT

- BERT는 MLM과 NSP를 위해 Transformer를 기반으로 구성
- BERT는 Transformer의 인코더-디코더 중 인코더만 사용

MLM(Masked Language Model)



- 일련의 단어가 주어지면 그 단어를 예측하는 작업

  입력에서 무작위하게 몇 개의 토큰을 마스킹하고 이를 Transformer 구조에 넣어 주변 단어의 맥락으로 마스킹된 토큰만 예측
- BERT에서 MLM이 수행되는 과정은 토큰 중 15%는 무작위로 [MASK] 토큰으로 바꿈
    - 80%는 토큰을 [MASK]로 바꾸고 10%는 토큰을 무작위 단어로 바꾼다.
    - [MASK] 토큰만을 예측하는 pre-training 작업을 수행
- BERT가 문맥을 파악하는 능력을 길러내게 한다.

NSP(Next Sentence Prediction)

- 두 문장의 관계를 이해하기 위해

  BERT의 학습 과정에서 두 번쨰 문장이 첫 번째 문장의 바로 다음에 오는 문장인지 예측하는 방식
- MLM이 적용된 예시

$\text{Input} =$ `[CLS] the man went to [MASK] store [SEP]`

`he bought a gallon [MASK] milk [SEP]`

$\text{Label} =$ `IsNext`

$\text{Input} =$ `[CLS] the man [MASK] to the store [SEP]`

`penguin [MASK] are flight ##less birds [SEP]`

$\text{Label} =$ `NotNext`