Chapter 10.
한국어 언어 모델: KoBERT, KoGPT2, KoBART

발표자: 박채원

22-10-07

들어가기 전

- KoBERT, KoGPT2, KoBART
  - SK텔레콤에서 공개한 오픈소스 언어 모델

10.1 KoBERT

- KoBERT: 한국어에 특화된 BERT
  - 한국어 위키피디아의 약 500만개의 문장과 5400만개의 단어 학습
  - 한국어 위키피디아를 대상으로 sentencePiece를 이용해 토크나이저를 학습함
  - 토크나이저엔 총 8002개의 어휘 존재

- 구글 다국어 BERT의 문제점
  - 구글 BERT의 경우 한국어를 포함한 다국어 모델이 있지만 학습 데이터에 한국어 비중이 낮아 한국어 다운스트림 태스크에서는 낮은 성능을 보임

10.1 KoBERT

- KoBERT 설치
  - pip install git+https://git@github.com/SKTBrain/KoBERT.git@master
  - requirements: Torch==1.9.0, sentencepiece==0.1.96, transformers==4.8.1

- KoBERT 토크나이저
  - from kobert_tokenizer import KoBERTTokenizer
  - tokenizer = KoBERTTokenizer.from_pretrained('skt/kobert-base-v1')
  - Tokenizer.encode("한국어 모델을 공유합니다.") -> [2, 4958, … , 54, 3] (CLS, SEP 토큰 자동 추가)

- KoBERT를 활용해 한국어 문장 표현 얻기
  - 문장을 토큰화해 토큰 아이디를 얻고, 이 아이디를 torch 텐서로 변환한 후 사전학습 된 kobert-base-v1 모델을 이용해 문장에 있는 토큰들의 **표현**을 얻을 수 있다.
  - 위 문장의 경우 torch.Size([1, 7, 768]) 크기의 문장 표현을 얻을 수 있음
  - CLS 토큰의 표현을 입력문장의 표현 벡터로 사용

```
tensor([[[-2.5751e-01, -1.2774e-01,  9.3930e-02,  ...,  1.0787e-01,
           5.4864e-02,  1.2147e-01],
         [-6.1597e-02,  1.6335e-01,  1.9825e-01,  ...,  2.1901e-02,
          -1.1380e-01, -1.0935e-01],
         [ 4.2402e-02,  1.5121e-01, -2.6012e-01,  ...,  2.2399e-01,
          -5.8981e-02, -3.5264e-02],
         ...,
         [ 1.5473e-01, -2.5105e-01,  4.9095e-02,  ...,  1.2206e-01,
          -9.0732e-02, -1.2300e-02],
         [ 3.6820e-02,  4.6772e-04,  1.9415e-02,  ..., -9.5293e-02,
          -6.6260e-02,  6.4011e-02],
         [ 8.1147e-02, -1.4294e-01, -9.7400e-02,  ..., -1.1425e-01,
          -6.2362e-02,  9.8021e-02]],

        [[-2.1518e-01,  7.7403e-02,  1.8227e-01,  ...,  1.0093e-01,
```

10.1 KoBERT

- 예제: 네이버 영화 리뷰 감성 분석
  - 네이버 영화 리뷰 감성 데이터 (NSMC)를 사용해 KoBERT를 파인튜닝
  - 기존 BERT는 긍부정을 분류할 수 있는 분류기를 포함하고 있지 않기 때문에 분류기, 즉 nn.Linear 레이어를 추가해야한다.
    - nn.Linear(hidden_size, num_classes) (*hidden_size=768, num_classes=2)

```
def forward(self, token_ids, valid_length, segment_ids):
    _, pooler = self.bert(input_ids = token_ids, token_type_ids = segment_ids.long(), attention_mask = attention_mask.float().to(token_ids.device))
    return self.classifier(out)
```

  - NSMC를 파인 튜닝하면 KoBERT의 정확도는 90.1%로 구글 다국어 BERT의 정확도 87.5% 보다 높은 성능을 보인다.

10.2 KoGPT2

- KoGPT2: 한국어에 특화된 GPT-2 모델
  - 주어진 텍스트를 기반으로 다름 단어를 잘 예측할 수 있도록 학습된 언어 모델. 특히 문장 생성에 최적화
  - 1억 2500만개의 변수 사용
  - 한국어 위키피디아 외 뉴스, 모두의 말뭉치, 청와대 국민청원 등 약 40GB 이상의 한국어 텍스트로 사전학습 됨
  - BPE(Byte Pair Encoding) 토크나이저로 학습
  - 어휘 크기 51,200 / 대화에 자주 사용되는 이모티콘, 이모지 등도 추가

10.2 KoGPT2

- KoGPT2 사용from transformers import GPT2LMHeadModel
- from transformers import PreTrainedTokenizerFast
- model = GPT2LMHeadModel.from_pretrained('skt/kogpt2-base-v2')

- 법

- KoGPT2 토크나이저 사용법
  - tokenizer = PreTrainedTokenizerFast.from_pretrained("skt/kogpt2-base-v2")
  - tokenizer.encode("~~")

- KoGPT2를 활용해 문장 생성하기
  - 임의의 문장을 토크나이저로 인코딩 후 인코딩 된 아이디를 텐서 취하고 모델의 generate 함수로 생성된 문장의 아이디를 얻는다. 그리고 얻어진 아이디를 토크나이저로 디코딩해 문장을 얻을 수 있다.

10.3 KoBART

- KoBART: 텍스트 인필링 노이즈 함수만을 적용해 BART 모델을 40GB 이상의 한국어 텍스트로 사전 학습시킨 한국어 인코더-디코더 언어 모델
  - 6개의 인코더 레이어와 6개의 디코더 레이어가 개별 스택처럼 쌓인 형태로 구성
  - 모든 레이어는 16개의 어텐션 헤드 사용
  - 피드포워드 네트워크는 768개 차원의 은닉 유닛으로 구성
  - 총 변수의 수는 1억 2000만개

- BART의 노이즈 함수
  - 토큰 마스킹: BERT의 토큰을 마스킹
  - 토큰 삭제: 토큰을 삭제, 어느 위치의 단어가 삭제되었는지 표시하지 않음
  - 문서 회전: 문서 내의 토큰들을 무작위로 배치
  - 문장 섞기: 문서 내의 문장들을 무작위로 배치
  - 텍스트 인필링: 일정 span을 마스크 토큰으로 치환

10.3 KoBART

- KoBART 설치 방법
  - pip install git+https://github.com/SKT-AI/KoBART#egg=kobart

- KoBART 토크나이저
  - from kobart import get_kobart_tokenizer
  - kobart_tokenizer = get_kobart_tokenizer()
  - kobart_tokenizer.tokenize("~~~")

- KoBART 모델 사용
  - from transformers import BartModel
  - from kobart import get_pytorch_kobart_model
  - model = BartModel.from_pretrained(get_pytorch_kobart_model())
  - output = model(input_ids) -> output으로부터 인코더의 출력과 디코더의 출력을 얻을 수 있다.

10.3 KoBART

- 예제: 문서 요약
  - 파인 튜닝 방법은 인코더에 원문을 넣고 디코더 출력에서 요약문을 예측
  - 학습 후 예측 시엔 인코더엔 원문을 입력하고 디코더엔 BOS(Beginning Of Sentence)태그인 <s>만 주고 자동 회귀 방식으로 생성

- 문서 요약 과정
  - 토크나이저로 입력문을 인코딩 한 후 모델의 generate 함수에 토큰 아이디를 입력해 '요약 토큰 아이디'를 생성. 생성된 요약 토큰 아이디를 토크나이저로 디코딩 해 요약문을 얻을 수 있다.

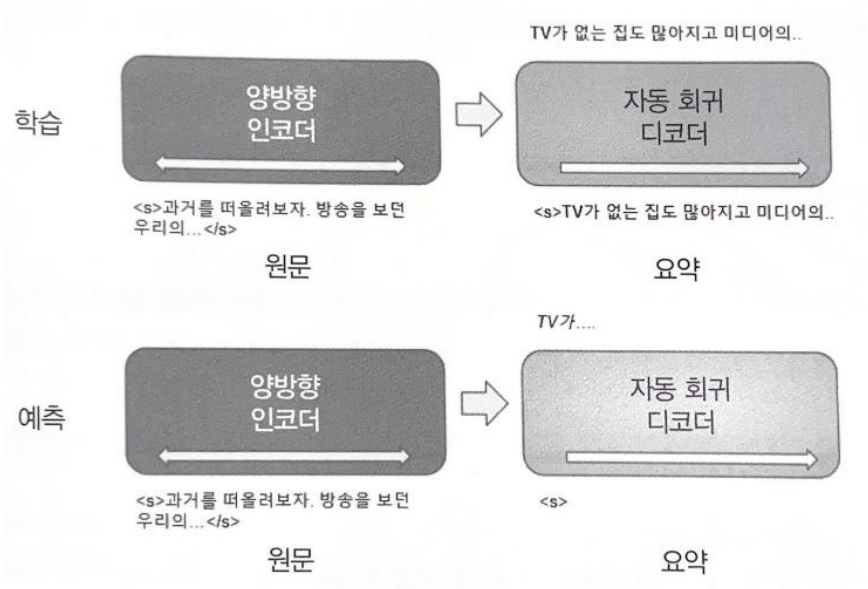

그림 1. 문서 요약 학습 및 예측