좌충우돌 감성분석 BERT 미세조정 분석

박정원[O], 모현수[*], 김정민[*]
[O]금오공과대학교 컴퓨터공학과,
[*]금오공과대학교 컴퓨터공학과
e-mail: {kine1956[O], grmo8408[*], qhrtj0517[*]}@naver.com

Sentiment Analysis BERT Models Challenge

Park-Jung-Won[O], Mo-Hyun-Su[*], Kim-Jeong-Min[*]
[O]Dept. of Computer Engineering, Kumoh National Institute of Technology,
[*]Dept. of Computer Engineering, Kumoh National Institute of Technology

● 요 약 ●

텍스트에 나타나는 감성을 분석하는 NLP task 중 하나인 감성분석에 자주 사용되는 한국어와 외국어 데이터들에 대해 다양한 BERT 모델들을 적용한 결과를 고성능 순서로 정리한 사이트(Paper with code)와 Github를 통해 준수한 성능을 보이는 BERT 모델들을 분석하고 실행해보며 성능향상을 통한 차별성을 가지는 것이 목표이다.

키워드: 감성 분석(Sentiment Analysis), BERT 모델, 성능 향상(Performance Improvement)

I. Introduction

현재 감성분석 BERT 모델에 자주 사용 되는 외국어/한국어 데이터 총 3종류를 기반으로 준수한 성능을 보이는 여러 가지 BERT을 분석하고 실행해보며 성능 향상을 목표로 한다. 외국어 데이터의 경우 IMDB(Internet Movie DataBase)[1]와 Amazon review를 사용했고 한국어 데이터는 NSMC(Naver Sentiment Movie Corpus)[2]를 사용했다.

해당 논문에 사용한 BERT 모델은 Github과 Sentiment Analysis | Papers With Code[3]를 참고하였다.

Table 1. 실험에 사용한 데이터와 BERT 모델

Data set	사용한 BERT Model
IMDB	NB-weighted-BON +dv-cosine
	Ktrain(SKT)
	TF-Hub
NSMC	Bert-Base-Multilingual- Cased
	Kobert
Amazon Review	Bert-Lager(UDA)
	Bert-Base-Uncased

II. Preliminaries

2.1 관련 연구

한국어 정보처리 학술에서는 관련연구로서 의견들이 제시되고 있다. 그 중에 문맥 표현 중 하나인 ELMo에 대하여 소개하고 한국어 데이터 중 NSMC를 이용하여 ELMo기법을 적용하여 CNN과 RBB을 학습하여 표현을 만들었고 NSMC를 해결하기 위해 문맥 표현 ELMo에 기반한 딥 러닝 모델을 제안한 것이 있다. 이 실험에서는 RNN 인코더 모델에 비해 89.24%의 성능을 보였다.[10]

감성분석 task에 대한 Bert-FineTuning 방식으로는 HuggingFace의 transfomers 라이브러리를 이용해서 TF-IDF를 통한 벡터화와 Naive Bayes 분류기를 결합해서 기존 모델보다 10%의 성능 향상을 이뤄낸 사례가 있다.[11]

III. The Proposed Scheme

3.1 IMDB데이터를 위한 BERT 모델들

- NB-weighted-BON + dv-cosine[1]

IMDB 데이터의 경우 벡터 데이터에 대해 내적 대신 cosine similarity 사용 시 정확도가 향상하는 점에서 Naïve bayes weighted bag of n-gram vector를 조합하여 IMDB 데이터에 대한 최고 성능을 보여줬다.

- Ktrain [2]

ktrain은 Tensorflow Keras에서 경량 wrapper로 신경망을 구축, 훈련, 디버그 및 배포하는 데 도움이 되는 라이브러리이다.

훈련 모델은 fit_onecycle로 1cycle learning rate policy로 훈련의 전반에 대해 학습 속도를 증가하고 후반에 감소하며 논문의 권장 사항을 기반으로 최대 2e-5의 학습률을 사용했다.

- TF-Hub [3]

Tensorflow Hub와 Keras를 사용한 기초적인 전이 학습 어플리케이션으로 pre-trained 된 텍스트 임베딩 모델을 Tensorflow Hub에 있는 사전 훈련된 텍스트 임베딩 모델로 사용했다. 차원의 크기 20 ~ 128, vocab 크기 1백만 개 이하를 사용한 모델들을 사용할 수 있다.

3.1.2 NSMC 데이터에 대한 BERT 모델 미세조정

- Bert-Base-Multilingual- Cased [4]

마스킹 된 언어 모델링 (MLM) 목표를 사용하여 영어에 대한 사전 훈련 된 모델이며12- layer, 768- hidden, 12- head, 110M개의 매개 변수를 가지고 있습니다.

- Kobert(SKTBrain) [5]

한글로 된 위키 + 뉴스 데이터를 기반으로 학습한 결과를 통해 한글과 관련된 task들에 대해 대체로 높은 성능을 보여주는 모델이다.

3.2 Amazon 리뷰 데이터를 위한 BERT모델들

- Bert-Larger (UDA) [6]

Unsupervised Data Augmentation(UDA)는 준지도학습 중 예측이 입력 값의 작은 변화에 민감하게 반응하지 않도록 강제하는 방법과 지도학습에서 사용되는 방법에 레이블이 존재하는 데이터에 변화를 줘 원본 데이터와 같은 레이블을 가지는 새로운 데이터를 만드는 방법, 이 두 방법을 결합한 것이다.

- Bert-Base-Uncased [7]

마스킹 된 언어 모델링 (MLM) 목표를 사용하여 영어에 대한 사전 훈련된 모델이며 12-layer, 768- hidden, 12-head, 110M개의 매개 변수를 가지고 있습니다.

3.3 BERT 미세조정 결과 분석

Table 2. 실험결과 비교

Data set	사용한 BERT Model	실행 결과	표기된 성능
IMDB	NB-weighted-BON +dv-cosine	97%	97%
	Ktrain(SKT)	94%	94%
	TF-Hub	85%	87%
NSMC	Bert-Base-Multilingual- Cased	87%	87%
	Kobert	89%	89%
Amazon Review	Bert-Lager(UDA)	88%	97%
	Bert-Base-Uncased	82%	x

정리된 표에 나타나는 BERT 모델들의 표기된 성능에 도달하기 위해 Ktrain[4]의 경우는 실험 방법 중 각 epoch와 batch 사이즈 조절하였으며, batch 사이즈를 조절했을 경우 큰 향상은 없었으나 미세한 향상은 일어났고, epoch 값을 조절하였을 때에는 성능향상의 기미가 확실히 보였다. 그래서 epoch 값과 batch 사이즈를 동시에 변경하였을 때, 시너지 효과까지는 아니지만 성능향상 효과는 하나의 값만 바꿨을 경우보다는 향상되었다. TF-hub[3]의 경우 Ktrain 문제 해결과 동일한 방식과 레이어 부분의 함수 모델을 교체하는 방식을 시도하였지만 도달하지 못했고 BERT-Large(UDA)의 경우 Tensorflow와 Pytorch의 두 모델이 존재하였는데, 우선 Tensorflow 같은 경우 환경조건 자체가 옛날 버전을 호환하여 예기치 못한 오류들이 많이 일어나 실행하지 못하였다. 그래서 Pytorch을 사용한 모델을 시도해 보았으며, 이는 실행은 무사히 되었다. 하지만 Amazon 데이터와 BERT 모델의 방대한 크기로 인한 실행 환경 제약과 설정을 달리 하였을 때, 사용되는 메모리 양이 초과되어 제대로 된 실험이 불가능하였으며 표기된 성능에 미치지 못했다.

IV. Conclusions

감성분석에 자주 쓰이는 데이터와 BERT 모델들을 돌려보면서 실행을 위한 기본 설정, 추가적인 라이브러리 설치 그리고 실행과정에서 생기는 자잘한 오류들을 해결하면서 BERT 모델 실행에 대한 방식을 이해했지만 성능 향상에 대한 방안을 제대로 모색하지 못해서 명확한 성능향상을 이뤄내지 못한 점이 아쉬웠다.

REFERENCES

[1] IMDBDataSet, https://keras.io/ko/datasets/#imdb

[2] NSMC DataSet, https://github.com/e9t/nsmc

[3] Sentiment Analysis | Papers With Code, https://paperswithcode.com/task/sentiment-analysis

[3] NB-weighted-BON + dv-cosine Github, https://github.com/tanthongtan/dv-cosine,Tan Thongtan and Tanasanee Phienthrakul, "Sentiment Classification using Document Embeddings trained with Cosine Similarity" on Computational Linguistics: Student Research Workshop page 407 ~ 414, July 28 - August 2, 2019.

[4] Ktrain, https://github.com/amaiya/ktrain

[5] TF-hub, https://github.com/tensorflow/docs-l10n

[6] Bert-base-multilingual, https://mccormickml.com/2019/07/22/BERT-fine-tuning/

[7] Kobert, https://github.com/SKTBrain/KoBERT

[8] Bert-larger(UDA), https://github.com/google-research/uda, https://github.com/SanghunYun/UDA_pytorch

[9] Bert-base-Uncased, https://huggingface.co/bert-base-uncased

[10] Cheoneum Park, Geonyeong Kim, Hyunsun Kim, Changki LeeKangwon National University, Contextualized Embedding-based Korean Movie Review Sentiment Analysis, pp.75-78(1-4), 2018

[11] Tutorial: Fine tuning BERT for Sentiment Analysis, https://skimai.com/fine-tuning-bert-for-sentiment-analysis/