JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 24, No. 11, pp. 2831-2840, Nov. 2023

Check for updates

KoBERT, KoGPT-2, KoBART 활용 및 하이퍼파라미터 최적화를 진행한 리뷰 감성분석 애플리케이션 구현

이 민 아[1] · 박 연 지[2] · 나 준 영[1] · 손 채 봉[3*]
[1]광운대학교 전자통신공학과 석사과정 [2]광운대학교 전자통신공학과 박사과정 [3*]광운대학교 전자통신공학과 부교수

Implementation of Review Sentiment Analysis Application Using KoBERT, KoGPT-2, and KoBART Optimized Hyperparameters

Min-A Lee[1] · Yeon-Ji Park[2] · Jun-Yeong Na[1] · Chae-Bong Sohn[3*]

[1]Master's Course, Department of Electronics & Communications Engineering, Kwangwoon University, Seoul 01897, Korea
[2]Ph.D Course, Department of Electronics & Communications Engineering, Kwangwoon University, Seoul 01897, Korea
[3*]Professor, Department of Electronics & Communications Engineering, Kwangwoon University, Seoul 01897, Korea

[요 약]

응용 프로그램 배포 플랫폼에서 제공되는 사용자 리뷰와 별점은 애플리케이션의 다운로드 횟수에 큰 영향을 미치기 때문에, 개발자들은 리뷰를 통해 사용자들의 피드백을 받아들이고 애플리케이션을 업데이트한다. 그러나 사용자가 원하는 것을 알기 위해서는 리뷰를 모두 읽어야만 하는 불편함이 있다. 이를 개선하기 위해 리뷰 데이터셋을 분석하고, 그 결과를 개발자에게 보여주려고 한다. 데이터셋을 정제한 후, 모델의 하이퍼파라미터 변경을 통한 파인튜닝을 진행하였다. 카카오톡과 인스타그램 리뷰를 크롤링해 초기 데이터셋을 생성하고, KoBERT와 KoGPT-2, KoBART 모델을 사용한 감성분석을 진행하였다. 정제한 데이터셋으로 각 모델 별 재학습을 진행해 보았고, 모델의 하이퍼파라미터를 변경해보며 학습을 진행하였다. 초기 데이터로 진행한 감성분석의 정확도가 약 74%가 나온 반면, 데이터 정제와 모델의 하이퍼파라미터 보정 후 정확도가 약 89%로 약 15% 증가함을 볼 수 있다. 그 후 감성분석 성능이 가장 높은 모델을 사용하여 리뷰를 선택해 참고할 수 있게 하고자 애플리케이션을 개발하였다. 해당 애플리케이션을 사용함으로써 개발자가 사용자의 만족도를 높이는 방향으로 업그레이드하도록 도움을 줄 것이라 기대한다.

[Abstract]

User reviews and ratings available on application distribution platforms have a significant impact on the number of downloads an application receives, so developers rely on reviews to get feedback from users and update their applications. However, it is inconvenient to read all the reviews to know what users want. To improve this, we want to analyze the review dataset and show the results to developers. After cleaning the dataset, we proceeded to fine-tune the model by changing the hyperparameters. We created an initial dataset by crawling KakaoTalk and Instagram reviews, and conducted sentiment analysis using KoBERT, KoGPT-2, and KoBART models. We retrained each model with the purified dataset and changed the hyperparameters of the models to improve the learning. While the accuracy of sentiment analysis with the initial data was about 74%, we can see that the accuracy increased by about 15% to about 89% after data purification and model hyperparameter correction. We then developed an application to select and reference reviews using the model with the highest sentiment analysis performance. By using this application, we hope to help developers upgrade to improve user satisfaction.

색인어 : 감정분석, 자연어처리, KoBART, KoBERT, KoGPT-2

Keyword : KoBART, KoBERT, KoGPT-2, Natural Language Processing, Sentiment Analysis

http://dx.doi.org/10.9728/dcs.2023.24.11.2831

Received 22 September 2023; **Revised** 01 November 2023
Accepted 20 November 2023

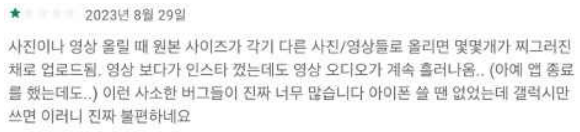
***Corresponding Author; Chae-Bong Sohn**

Tel: +※ 개인정보 표시제한

E-mail: chsohn@kw.ac.kr

Ⅰ. 서 론

특정 카테고리 내에서 대부분의 애플리케이션들은 비슷한 서비스를 제공하는 것을 찾아볼 수 있다. Google Play 또는 Apple AppStore와 같은 응용 프로그램 배포 플랫폼에서 등급과 리뷰 형식으로 제공되는 사용자의 피드백은 애플리케이션의 다운로드 횟수에 영향을 미친다[1]. 즉, 사용자 피드백 요소들이 다른 사용자가 여러 비슷한 애플리케이션들 중 한 가지를 선택할 수 있도록 하는 데에 주요 영향을 미칠 수 있다는 것을 의미한다. 특히 한국 소비자들은, 미국 소비자들보다 애플리케이션을 구매하는 데 소비자 평점을 더 중요시한다는 연구결과가 존재한다[2]. 이런 부분들로 인하여, 개발자들은 애플리케이션을 사용자가 만족할 수 있도록 끊임없이 업데이트 하는 것이 필요하다. 비슷한 사례로, 유튜브에서 소비자의 리뷰를 활용하여 영상에 대한 의견을 분류한 연구가 존재한다[3]. 그림 1은 구글 플레이 스토어에서 사용자 피드백 요소 중 일부인 애플리케이션에 대한 사용자의 별점과 리뷰이다.



* I attached a picture to show the Korean review.

그림 1. '구글 플레이 스토어 애플리케이션'에 대한 사용자 별점 등급 및 리뷰

Fig. 1. Starred rating and review of a user on 'Google Play Store application'

1-1 연구 배경

Google Play 또는 Apple AppStore와 같은 응용 프로그램 배포 플랫폼에서 등급과 리뷰 형식으로 제공되는 사용자의 피드백은 애플리케이션의 다운로드 횟수에 영향을 미친다[1]. 그림 1은 구글 플레이 스토어에서 사용자 피드백 요소 중 일부인 애플리케이션에 대한 사용자의 별점과 리뷰이다. 이러한 리뷰들이 다른 사용자들에게 있어서 비슷한 애플리케이션들 중 하나의 애플리케이션 선택 시 많은 영향을 끼친다. 특히 한국 소비자들은, 미국 소비자들보다 애플리케이션을 구매하는 데 소비자 평점을 더 중요시한다는 연구결과가 존재한다[2]. 이런 부분들로 인하여, 개발자들은 사용자가 만족할 수 있도록 애플리케이션을 끊임없이 업데이트 하는 것이 필요하다. 비슷한 사례로, 유튜브에서 소비자의 리뷰를 활용하여 영상에 대한 의견을 분류한 연구가 존재한다[3].

본 논문에서는 BERT(Bidirectional Encoder Representations from Transformers), GPT(Generative Pre-trained Transformer), BART(Bidirectional and Auto-Regressive Transformers) 대신 KoBERT(Korean Bidirectional Encoder Representations from Transformers), KoGPT-2(Korean Generative Pre-trained Transformer-2), KoBART(Korean Bidirectional and Auto-Regressive Transformers)를 사용하였다. 그 이유는 BERT, GPT2, BART를 사용하였을 때보다 KoBERT, KoGPT-2, KoBART를 사용하였을 때 감성분석의 정확도가 높게 나오기 때문이고, 긍정/요청/부정으로 나누는 감성분석 정확도를 높여야 개발자들에게 실질적인 도움이 될 수 있기 때문이다. 이는 BERT, GPT2, BART가 영어 데이터셋을 사용하여 사전학습된 모델인 반면, KoBERT, KoGPT-2, KoBART는 한국어 데이터셋을 사용하여 사전학습된 모델이기 때문에 발생한 결과로 보여진다. 따라서 본 논문에서는 KoBERT, KoGPT-2, KoBART를 채택하였다.

본 논문에서는 정제하지 않은 데이터셋과 정제한 데이터셋을 사용하였을 때 모델의 테스트 정확도를 비교하고, 하이퍼파라미터 변경을 통한 파인튜닝이 정확도를 얼마나 높일 수 있는지 알아본다. 그 후 가장 높은 정확도를 보인 모델을 사용하여 애플리케이션을 구현한다.

데이터셋은 긍정/요청/부정 세 가지로 분류했다. 애플리케이션 개발자가 리뷰를 볼 때, 어떤 점 때문에 긍정적/부정적으로 평가하는지, 어떤 점이 개선되었으면 하는지를 중점으로 생각할 것이라 판단해 긍정/요청/부정 3가지로 나누어 감성분석을 시도하였다.[4]

1-2 연구 내용

데이터셋을 긍정/요청/부정으로 분류하는 감성분석의 정확도를 증가시키기 위해 크게 두 가지 방법을 사용하였다. 첫번째는 데이터셋 자체를 정제하는 방법이고, 두번째는 각 모델의 하이퍼파라미터 변경을 통한 파인튜닝을 진행하는 것이다.

먼저 초기 데이터셋을 구하기 위해 '구글 플레이 스토어' 웹사이트에서 카카오톡과 인스타그램의 리뷰를 크롤링했다. 그리고 1~2점을 부정, 3점을 요청, 4~5점을 긍정으로 분류한 후, KoBERT, KoGPT-2, KoBART 모델을 이용하여 감성분석했다. KoBERT, KoGPT-2, KoBART 세 모델 모두 SKT에서 개발한 모델로, Transformer 기반의 모델을 비교하기 위해 세 모델을 채택했다.

그 후 데이터셋을 정제하는 방법을 사용했다. 데이터셋 정제의 방법으로는, 데이터셋을 긍정, 요청, 부정 세가지 경우로 나눈 후 리뷰가 5자 이내로 짧은 경우, 한글의 자/모음만 존재하는 경우, 맞춤법이 틀린 경우, 내용은 부정적인 리뷰인데 긍정적인 별점을 받거나 그 반대인 경우, 결측값을 제거하는 방법을 사용하였다.

마지막으로 하이퍼파라미터 변경을 통한 파인튜닝 방법을 사용했다. 정제한 데이터셋을 그대로 사용하고, 하이퍼파라미터 중 배치사이즈와 러닝레이트를 변경해가며 최상의 정확도를 내는 하이퍼파라미터를 찾았다. 하이퍼파라미터 중 배치사

이즈와 러닝레이트 조절을 시도한 이유는, 배치사이즈는 조절하는 값에 따라 발생하는 노이즈의 양이 달라지기 때문에 모델의 감성분석 정확도에 영향을 끼칠 것이라 생각했기 때문이고, 러닝레이트는 값에 따라 로스가 얼마나 잘 수렴하는지 달라지기 때문에 감성분석 정확도에 영향을 끼칠 것이라 생각했기 때문이다. 배치사이즈는 16과 32 두 가지를 사용하였고, 러닝레이트는 1e-5, 3e-5, 5e-5 세 가지를 사용해 각 모델 당 총 6번의 실험을 진행하였다.

그 후 감성분석 성능이 가장 잘나온 모델을 이용하여 애플리케이션에 적용하였다. 애플리케이션에서는 원하는 리뷰의 그룹을 선택할 시 관련 리뷰들을 보여줄 수 있는 시스템을 제안한다.

데이터셋 정제만 진행했을 때의 감성분석 결과는 데이터셋 정제를 진행하지 않았을 때의 감성분석 결과인 KoBERT 75.0%, KoGPT-2 73.9%, KoBART 77.8%에 비해 약 10.9%, 8.4%, 9.1% 증가한 KoBERT 85.9%, KoGPT-2 82.3%, KoBART 86.9%의 정확도를 보여주었다. 이에 하이퍼파라미터 정제까지 시도한 결과, KoBERT 85.9%, KoGPT-2 85.2%, KoBART 89.2%의 정확도까지 오른 것을 확인할 수 있었다.

1-3 논문의 구성

본 논문의 구성은 1장 서론으로 시작하여, 2장에서는 BERT, GPT, BART, KoBERT, KoGPT-2, KoBART 모델과 이 모델들의 근간이 되는 Transformer에 대해 설명하고, BERT, GPT, BART 모델로 한국어 감성분석한 결과를 보여준다. 3장에서는 데이터셋 정제 후 모델 학습에 대한 과정과 결과를 소개하며, 4장에서는 하이퍼파라미터 변경에 따른 모델 학습 결과를 소개한다. 5장에서는 애플리케이션의 설계와 구현 방법을 서술하며, 마지막 장에서는 결과를 기반으로 향후 실험을 발전시킬 방향을 제안한다.

II. 관련연구

2-1 Transformer

트랜스포머는 기존에 존재하던 아키텍처들과는 달리, CNN(Convolutional Neural Networks)이나 RNN(Recurrent Neural Network)의 구조 없이 어텐션 메커니즘으로 이루어진 구조를 띈다([5], 그림 2). 이전까지는 성능 개선을 위하여 새로운 구조가 개발되어 나왔다면, 트랜스포머 이후엔 트랜스포머를 개선한 BERT[6], GPT[7], BART[8] 등의 모델들이 나왔다.

트랜스포머 이전에 나왔던 RNN, LSTM(LongShort-Term Memory)은 연산을 위해 그 전의 결과 값이 현재의 입력 값과 같이 입력되어야하는 구조였다[9]. 따라서 이전 연산이 끝나기 전에는 다음 연산을 할 수 없다는 단점이 있었다. 트랜스포머는 이런 면을 보완하기 위하여 어텐션 메커니즘을 중점으로 구성되었다.

트랜스포머는 크게 인코더와 디코더로 이루어져있다. 인코더는 입력 문장을 임베딩해 문맥 정보를 추출하고, 이것을 이용하여 중요한 특징을 추출하는 역할을 한다. 인코더 층은 여러 개로 구성되며, 각 층은 자체적으로 어텐션과 피드포워드 신경망(feedforward neural network)으로 구성되어 있다. 디코더는 출력 문장을 생성하는 역할을 한다. 인코더로부터 얻은 문맥 정보와 이전에 생성된 단어를 활용해 다음 단어를 예측하고, 생성한다. 디코더도 인코더와 마찬가지로 여러 층으로 구성되며, 각 층은 어텐션과 피드포워드 신경망으로 구성된다.

트랜스포머의 핵심 아이디어는 셀프 어텐션(self-attention)으로, 입력 시퀀스 내 다른 위치의 단어 간의 상호 관계를 계산하여 중요한 정보를 추출하는 메커니즘이다. 셀프 어텐션은 각 단어의 임베딩 벡터를 활용하여 현재 단어와 다른 단어들 간의 유사도를 계산하는 방식이다.

그림 2의 트랜스포머를 이용하여 GPT와 BERT, BART 모델이 제작되었다.

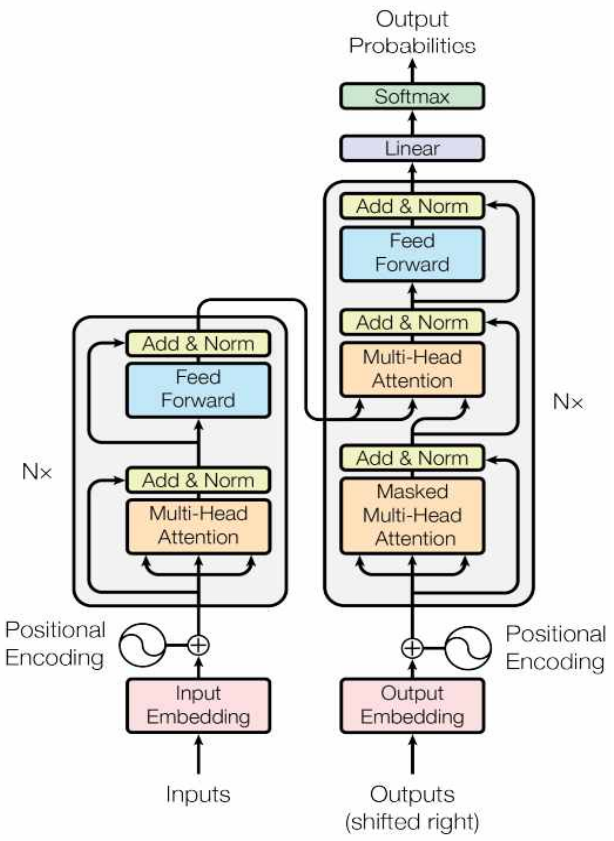


그림 2. 트랜스포머의 구조[5]
Fig. 2. Structure of Transformer[5]

2-2 GPT

GPT는 Generative Pre-trained Transformer의 약어로, 트랜스포머 구조에서 인코더 부분을 생략하고 출력 문장을 생성하는 역할을 하는 디코더만을 발전시켜 만든 모델이다(그림 3). 따라서 주어진 텍스트나 문맥을 기반으로 다음 단어나 문장을 예측, 생성할 수 있는 모델로, 텍스트 생성에 좀 더 특화되어있다. GPT는 문장 생성뿐만 아니라 파인튜닝 과정을 거쳐 코드 작성 및 학습, 텍스트를 다른 스타일로 변환시키기, 데이터 분석 등에도 적용할 수 있다.

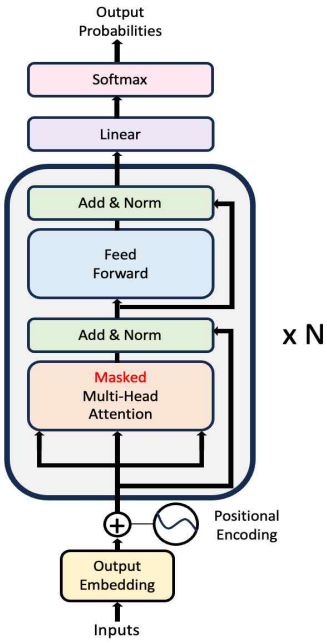


그림 3. 디코더의 구조
Fig. 3. Structure of decoder

2-3 BERT

BERT는 Bidirectional Encoder Representations from Transformers의 약어로, GPT의 단방향 예측을 보완하기 위하여 트랜스포머의 디코더 부분을 무시하고 인코더 부분만을 발전시켜 만들었다(그림 4). BERT는 한 문장에서 특정 단어를 가리고 그것을 예측하기 위해 문장 분석 학습을 진행하였다. 이 학습은 데이터에 대한 라벨링이 필요 없어서, 웹상에 존재하는 각종 문장을 긁어와서 학습 재료로 사용할 수 있었다. 따라서 더욱 저렴한 데이터 가공 비용으로 방대한 학습량을 달성할 수 있었다. 이런 방식으로 많은 데이터를 학습하여 하이퍼 파라미터값을 생성해 놓았고, 각각의 독립적인 분류, 추론, 문장비교, 질문 대답 등의 태스크에서 간단한 레이어를 추가하고, 적은 데이터와 학습시간으로 파인튜닝만 거쳐도 각 태스크 별 SOTA(State-of-the-art) 모델을 뛰어넘는 성능을 보여주었다.

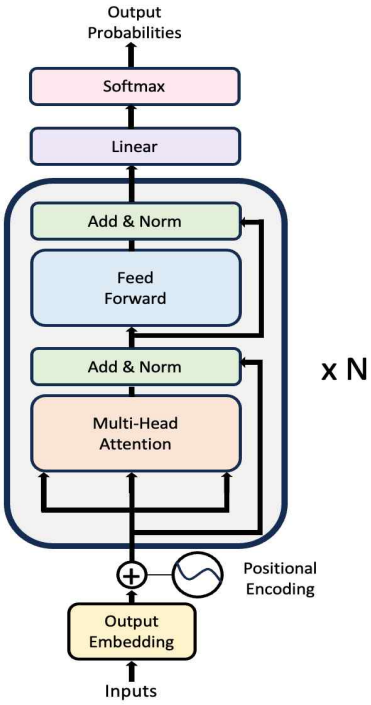


그림 4. 인코더의 구조
Fig. 4. Structure of Encoder

2-4 BART

BART는 Bidirectional and Auto-Regressive Transformers의 약어로, facebook AI(Artificial Intelligence)에서 개발된 시퀀스 투 시퀀스 모델이다. BERT의 구조와 GPT의 구조를 합친 구조를 띄고 있기 때문에 트랜스포머와 구조가 유사하다. 이 점으로 인하여 자연어 생성, 자연어 이해 부분 둘 다 강세를 보인다. 구조만 보았을 때엔 트랜스포머와 비슷한 아키텍처를 기반으로 하지만, 훈련 방법과 오브젝트 평션에 차이가 있다. BART는 입력 시퀀스의 일부를 무작위로 마스킹한 후, 마스킹된 부분을 오리지널 시퀀스로 예측하는 방법으로 학습되었다. 학습방법은 BERT와 비슷하지만 BART는 전체 시퀀스를 대상으로하는 시퀀스 투 시퀀스 모델로 학습된다는 점에서 차이가 있다.

2-5 BERT, GPT-2, BART의 한국어 감성분석 결과

표 1은 BERT, GPT2, BART를 학습할 때 사용한 파라미터이고, 표 2는 BERT, GPT2, BART의 학습 결과이다.

이를 통해 앞으로 보여줄 KoBERT, KoGPT-2, KoBART의 한국어 감성분석 결과보다 BERT, GPT-2, BART의 한국어 감성분석 결과가 현저히 떨어짐을 보여준다. 이는 BERT, GPT-2, BART는 영어 데이터셋을 사용하여 학습한 반면, KoBERT, KoGPT-2, KoBART는 한국어 데이터셋을 사용하여 학습한 결과라고 볼 수 있다.

표 1. BERT, GPT2, BART 모델 학습 진행 시 사용한 파라미터
Table 1. Parameters used when training the BERT, GPT2, BART model

Parameters	Value
max length	64
classes	3
batch size	32
optimizer	AdamW
epochs	100
learning rate	5e-5

표 2. BERT, GPT2, BART의 학습 결과
Table 2. Training outcomes for BERT, GPT2, and BART

Model	Test accuracy
BERT	66.78%
GPT	57.89%
BART	67.52%

2-6 KoBERT

KoBERT는 Korean Bidirectional Encoder Representations from Transformers의 약어로, 기존의 BERT가 영어를 기반으로 학습된 모델이어서, 한국어에서의 성능 한계가 존재했다. KoBERT[10]는 이러한 성능의 한계를 뛰어넘기 위해, SKT에서 개발한 한국어 BERT 모델이다. KoBERT는 뉴스나 위키피디아 등등에서 수집한 수백만개의 한국어 문장을 가지고 만든 대규모 말뭉치를 학습하였다. 한국어는 불규칙한 언어 변화가 있다는 언어의 특징이 있어서, 이 부분을 반영하기 위하여 데이터 기반 토큰화 기법을 추가하여 성능 향상을 이끌었다. KoBERT는 다양한 딥러닝 API를 지원하므로, 다양한 분야에서 언어 이해 서비스 확산에 기여하고 있다.

2-7 KoGPT-2

KoGPT-2[11]는 Korean Generative Pre-trained Transformer2의 약어로, SKT에서 발표한 한국어 GPT이며 머신러닝 알고리즘을 활용해 입력된 샘플 텍스트를 문법적, 정보, 구문론적 등의 일관성을 갖춘 텍스트로 생성하는 자연어처리 모델이다. GPT-2[12]와 마찬가지로 트랜스포머의 디코더만 사용한 구조이며, 한국어 위키피디아, 뉴스, 나무위키 등 다양한 데이터로부터 추출한 152M의 문장으로 학습되었다. 자주 쓰는 이모지 등을 추가해 인식 성능을 높였다. 오픈소스 기반의 GPT-2 모델을 한국어로 학습한 KoGPT-2는 질문에 대한 응답 생성, 챗봇, 문장 완성 등 한국어 해석이 필요한 여러 애플리케이션의 머신러닝 성능을 향상에 기여하고 있다. 모델을 파인튜닝시키면 감성 분류 모델도 얻을 수 있다. 따라서 본 논문에서는 KoGPT-2를 파인튜닝해 텍스트 감성분석에 사용한다.

2-8 KoBART

KoBART[13]는 Korean Bidirectional and Auto-Regressive Transformers의 약어로, KoBERT, KoGPT-2에 이어 SKT에서 세 번째로 발표한 한국어버전의 모델이다. BART와 마찬가지로 트랜스포머의 인코더-디코더 구조를 띄고 있으며, 시퀀스투시퀀스 구조를 사용한다. 사전학습에 한국어 위키피디아, 뉴스, 책, 모두의 말뭉치 등 이전보다 더 다양한 문장으로 학습되었다. KoBART는 트랜스포머의 인코더-디코더 구조를 그대로 띄고 있으므로, BART와 마찬가지로 자연어 생성과 자연어 이해 둘 다에 강세를 보이고 있다.

Ⅲ. 데이터셋 정제와 모델 학습 결과

3-1 초기 데이터셋

보정되지 않은 데이터셋을 사용할 때와 정제한 데이터셋을 사용할 때의 테스트 정확도 비교를 위해 초기 데이터셋을 준비하고, 초기 데이터셋으로도 모델의 학습을 진행했다.

준비한 데이터셋은 한국 사람들에게 많이 쓰이는 애플리케이션인 '카카오톡'과 '인스타그램'을 크롤링한 데이터셋을 사용했다. 파이썬 기반의 크롤러를 사용하여 구글 플레이스토어 웹사이트에서 '카카오톡', '인스타그램' 애플리케이션 리뷰들을 가져왔다. 2023년 8월 기준 최신 리뷰들을 사용하였고, 리뷰와 별점 컬럼만 추출하였다. 감성의 분류는 총 3가지로, 긍정 / 요청 / 부정으로 나누었다. 그 이유는 애플리케이션의 개발자들이 어떤 면에서 애플리케이션을 긍정적으로 평가하는지, 어떤 면에서 애플리케이션을 부정적으로 평가하는지, 어떤 점이 개선되었으면 하는지를 가장 중요하게 고려할 것이라고 생각했기 때문이다. 이용한 데이터셋의 개수인 4,500개 중 2,250개는 카카오톡 리뷰, 2,250개는 인스타그램 리뷰로 사용하였다. 전체 데이터 4,500개 중 80%인 3,600개는 학습 데이터셋으로, 20%인 900개는 테스트 데이터셋으로 사용하였다.

3-2 데이터셋 정제

리뷰 중 특수기호, 숫자, 외국어, 말이 되지 않는 한글(예를 들어 “ㅎㅋ”, “ㅏㅏㅏ” 등의 자음,모음만 존재하는 경우)들과 결측값을 제거하는 과정을 거쳤고, 5글자 이내의 리뷰(예를 들어 “굳”, “좋아” 등)를 제거하여 단순히 별점을 주기 위해 쓴 리뷰들을 제거하였다.

긍정에는 칭찬의 말이 들어간 문장들을 넣었고, 중립에는 “~해결해 주세요”, “~고쳐 주세요”, “~해주세요” 등등 부탁의 언어가 들어간 문장들을 넣었다. 부정엔 부정적인 말들을

넣었다. 데이터 전처리를 하지 않고 학습했을 때와 같이 전체 데이터의 80%에 해당하는 3,600개는 학습 데이터셋으로, 전체 데이터의 20%에 해당하는 900개는 테스트 데이터셋으로 각각 저장하였다.

표 3은 긍정, 중립(요청) 부정 데이터셋으로 사용한 개수이다.

표 3. 학습과 테스트에 사용한 긍정, 중립(요청), 부정 데이터셋의 개수

Table 3. Number of positive, neutral (request) and negative datasets used for training and testing

	Positive	Neutral (requested)	Negative
Train dataset	1,200	1,200	1,200
	Positive	Neutral (requested)	Negative
Test dataset	300	300	300

3-3 모델 학습 결과

정제하지 않은 데이터셋을 사용하였을 때에는 그림 5와 같이, KoBERT의 학습 정확도 = 0.9920, 테스트 정확도 = 0.7495, KoGPT-2의 학습 정확도 = 0.9473, 테스트 정확도 = 0.7388, KoBART의 학습 정확도 = 0.9992, 테스트 정확도 = 0.7779가 나왔다는 것을 확인할 수 있다.

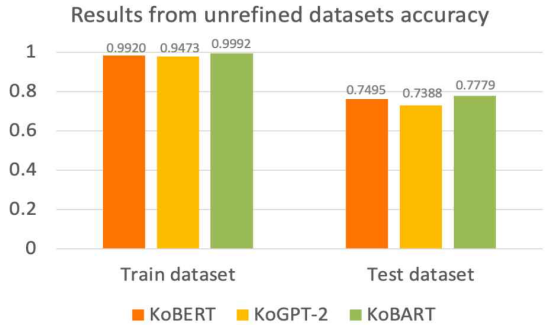


그림 5. 정제하지 않은 데이터셋으로 모델을 학습한 결과

Fig. 5. The result of training the model with unrefined datasets

반면 정제한 데이터셋을 사용했을 시 그림 6과 같이 KoBERT의 학습 정확도 = 0.9992, 테스트 정확도 = 0.8587으로 테스트 정확도가 약 10.9%가 증가했고, KoGPT-2의 학습 정확도 = 0.9883, 테스트 정확도 = 0.8232로 테스트 정확도가 약 8.44% 증가했고, KoBERT의 학습 정확도 = 0.9819, 테스트 정확도 = 0.8692로 테스트 정확도가 약 9.1%가 증가했다는 것을 알 수 있다.

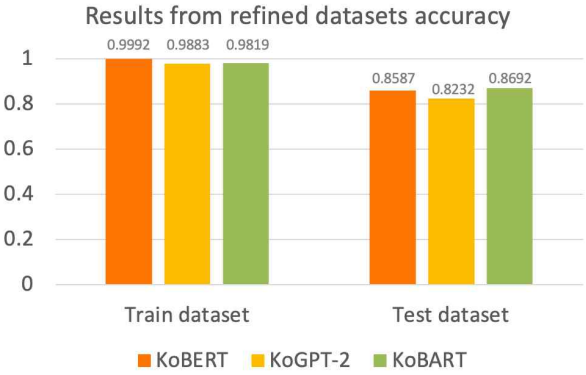


그림 6. 정제한 데이터셋으로 모델을 학습한 결과

Fig. 6. The result of training the model with refined datasets

표 4는 KoBART, KoBERT 와 KoGPT-2 모델 학습을 했을 시 사용한 파라미터이다. 정제하지 않은 데이터셋을 사용할 때 보다 정제한 데이터셋을 사용했을 때 각각 7.8%, 7.62%, 9.1%가 증가하여 데이터셋 정제가 의미 있었다는 것을 보여준다.

KoBART와 KoBERT가 KoGPT-2보다 더 나은 성능을 보인 것을 알 수 있는데, 이는 KoBART와 KoBERT는 양방향으로 문맥을 파악할 수 있는 능력이 있기 때문에 자연어 이해와 자연어 분류에 강한 반면, KoGPT-2는 단방향으로 진행되는 자연어 생성에 강하기 때문에 자연어 분류에서 정확도가 비교적 떨어지는 것을 볼 수 있다.

표 4. 정제된 데이터셋으로 KoBART, KoBERT, KoGPT-2 모델 학습 진행 시 사용한 파라미터

Table 4. Parameters used when training the KoBART, KoBERT, KoGPT-2 model with refined datasets

Parameters	Value
max length	64
classes	3
batch size	32
optimizer	AdamW
epochs	100
learning rate	5e-5

Ⅳ. 모델의 파인튜닝과 모델 학습 결과

4-1 변경할 하이퍼파라미터 설정

데이터셋 정제에서 그치지 않고, 모델의 하이퍼파라미터를 변경해 보았을 시 더 나은 결과가 나올 수 있을 지 실험해보았다. 변경할 하이퍼파라미터로는 배치 사이즈와 러닝레이트를 선택했다. 배치사이즈는 조절하는 값에 따라 발생하는 노

이즈의 양이 달라지기 때문에 모델의 감성분석 정확도에 영향을 끼칠 것이라 생각했기 때문이고, 러닝레이트는 값에 따라 로스가 얼마나 잘 수렴하는지 달라지기 때문에 감성분석 정확도에 영향을 끼칠 것이라 생각했기 때문이다. 배치사이즈는 16과 32 두 가지를 사용하였고, 러닝레이트는 1e-5, 3e-5, 5e-5 세 가지를 사용해 각 모델 당 6번, 총 18번의 실험을 진행하였다.

4-2 모델 학습 결과

KoBERT, KoGPT-2, KoBART의 하이퍼파라미터 변경 후 학습한 결과를 정리한다. 표 5는 KoBERT의 학습결과, 표 6은 KoGPT-2의 학습 결과, 표 7은 KoBART의 학습결과이다.

결과에 따르면, KoBART 모델을 사용하며 배치사이즈 16, 러닝레이트 3e-5를 사용하였을 때 테스트 정확도가 약 89.2 가장 높은 것을 확인할 수 있다. 따라서 애플리케이션 구현 시 KoBART 모델의 하이퍼파라미터를 배치사이즈 16, 러닝레이트 3e-5로 변경하여 사용하였다.

표 5. KoBERT의 하이퍼파라미터 변경 시 학습 결과
Table 5. Training outcomes for changing hyperparameters in KoBERT

<Batch size 16>					
Learning rate	Test loss	Test accuracy	Precision	Recall	F1 score
1e-5	1.0121	0.7992	0.8026	0.8041	0.8033
3e-5	0.8881	0.8022	0.8121	0.8124	0.8122
5e-5	0.9221	0.8179	0.8188	0.8203	0.8195
<Batch size 32>					
Learning rate	Test loss	Test accuracy	Precision	Recall	F1 score
1e-5	0.9295	0.8259	0.8300	0.8303	0.8301
3e-5	0.7998	0.8541	0.8565	0.8541	0.8550
5e-5	0.9139	0.8587	0.8591	0.8604	0.8597

표 6. KoGPT-2의 하이퍼파라미터 변경 시 학습 결과
Table 6. Training outcomes for changing hyperparameters in KoGPT-2

<Batch size 16>					
Learning rate	Test loss	Test accuracy	Precision	Recall	F1 score
1e-5	0.8096	0.8519	0.8530	0.8519	0.8523
3e-5	0.8495	0.8254	0.8257	0.8254	0.8217
5e-5	1.003	0.8122	0.8102	0.8122	0.8099
<Batch size 32>					
Learning rate	Test loss	Test accuracy	Precision	Recall	F1 score
1e-5	1.0547	0.8486	0.8508	0.8486	0.8481
3e-5	0.8887	0.8508	0.8522	0.8508	0.8512
5e-5	0.9845	0.8232	0.8228	0.8232	0.8228

표 7. KoBART의 하이퍼파라미터 변경 시 학습 결과
Table 7. Training outcomes for changing hyperparameters in KoBART

<Batch size 16>					
Learning rate	Test loss	Test accuracy	Precision	Recall	F1 score
1e-5	0.8302	0.8621	0.8639	0.8621	0.8630
3e-5	0.8272	0.8923	0.8980	0.8291	0.8621
5e-5	1.0453	0.8208	0.8219	0.8247	0.8233
<Batch size 32>					
Learning rate	Test loss	Test accuracy	Precision	Recall	F1 score
1e-5	1.0029	0.8392	0.8401	0.8516	0.8459
3e-5	0.8783	0.8457	0.8477	0.8523	0.8500
5e-5	0.8922	0.8692	0.8714	0.8699	0.8706

V. 애플리케이션 구현

5-1 구현 환경

표 8은 본 논문에서 실험에 사용된 시스템 환경의 세부사항을 나타내고, 표 9는 애플리케이션 구현 시 사용한 KoBART의 하이퍼파라미터를 나타낸다.

표 8. 구현 환경
Table 8. Implementation environments

System environments	
CPU	AMD Ryzen 5 5600X 6-Core Processor
RAM	32GB
VGA	NVIDIA GeForce RTX 3060
OS	Ubuntu 20.04
TOOL	Python 3.8.16

표 9. 애플리케이션 구현 시 사용한 KoBART의 하이퍼파라미터
Table 9. KoBART's Parameters used when application implementation

Parameters	Value
max length	64
classes	3
batch size	16
optimizer	AdamW
epochs	100
learning rate	3e-5

애플리케이션은 Android Studio 2022.3.1과 Java 언어, KoBART 모델을 사용하여 개발하였다. 사용자가 보낸 URL (Uniform Resource Locator)은 Ubuntu 20.04에 Apache 2.4.57를 설치하여, apache 내에서 URL 내의 리뷰 크롤링,

크롤링한 리뷰의 데이터셋 전처리 후 학습된 KoBART에 넣어 결과를 뽑아내었다. 그림 7을 통하여 핸드폰에서 작동 시 모습을 확인할 수 있으며, 그림 8을 통하여 사용자가 애플리케이션에 본인이 원하는 URL을 붙여넣는 순간부터의 프로그램 동작 플로우차트를 확인할 수 있다.

* I attached a picture to show the Korean review classification result on a mobile phone.

그림 7. 핸드폰에서의 애플리케이션 작동 예시

Fig. 7. Example of application operation on a mobile phone

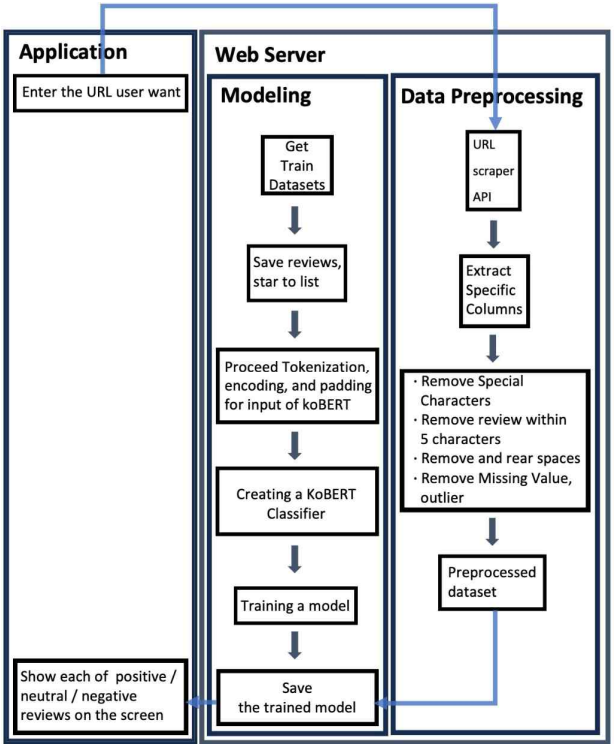


그림 8. 전반적인 시스템 동작 플로우차트

Fig. 8. Flowchart of overall system

사용자가 애플리케이션에 긍정 / 중립(요청) / 부정으로 분류하여 보고 싶은 리뷰가 있는 URL을 붙여 넣으면, 웹서버에 존재하는 'Data preprocessing' 파일이 받는다. 해당 파일에서 'URL scraper API(Application Programming Interface)'를 이용하여 리뷰를 불러오고, 리뷰와 별점의 컬럼만 따로 추출한다. 추출한 컬럼을 가지고 특수문자, 외국어, 한국어 자 · 모음만 존재하는 경우를 제거한다. 그 이후 5글자 이내의 리뷰를 제거하고, 앞 · 뒤 공백을 제거한 후 마지막으로 결측 값을 제거하는 데이터 전처리 과정을 거친다.

전처리 과정을 거친 데이터셋은 학습 후 저장된 모델을 통하여 긍정 / 중립(요청) / 부정으로 나누어지고, 웹서버에서 각 분류에 속한 리뷰들이 애플리케이션 화면으로 보여지게 된다.

Ⅵ. 결 론

본 논문은 애플리케이션의 중요한 피드백 요소인 별점과 리뷰에서 발생할 수 있는 모호성을 제거하고자, 개발자가 참고할 만한 긍정/요청/부정 세 가지로 경우를 나누어 감성분석을 진행하고 해당 결과를 애플리케이션에 구현하여 개발자들이 애플리케이션을 업그레이드 할 때 도움이 되고자 작성되었다.

데이터셋의 정제가 테스트 정확도에 얼마나 큰 영향을 미치는 지 확인하기 위해 정제하지 않은 데이터셋과 정제한 데이터셋을 각각 학습시켜 결과를 비교분석하였다. KoBERT는 정제하지 않은 데이터셋을 사용했을 때의 테스트 정확도가 75.0%였던 것에 비해, 정제한 데이터셋을 사용했을 때 85.8%로 약 10.8% 증가한 것을 확인할 수 있었고, KoGPT는 정제하지 않은 데이터셋을 사용했을 때의 테스트 정확도가 73.9%였던 것에 비해, 정제한 데이터셋을 사용했을 때 82.3%로 약 8.4% 증가한 것을 확인할 수 있었다. 또, KoBART는 정제하지 않은 데이터셋을 사용했을 때의 테스트 정확도가 77.9%였던 것에 비해, 정제한 데이터셋을 사용했을 때 86.9%로 약 9.0% 증가한 것을 확인할 수 있었다.

감성분석의 정확도를 끌어올리기 위해서 각 모델의 하이퍼파라미터 변경이 얼마나 영향을 미치는지 알아보기 위해, 하이퍼파라미터 중 배치사이즈와 러닝레이트의 값을 조정하며 테스트 정확도를 최대로 끌어올렸다. KoBERT는 배치사이즈 32, 러닝레이트 5e-5일 때 테스트 정확도가 85.9%로 가장 높았고, KoGPT-2는 배치사이즈 16, 러닝레이트 1e-5일 때 정확도가 85.2%로 가장 높았다. KoBART는 배치사이즈 16, 러닝레이트 3e-5일 때 정확도가 86.9%로 가장 높았다.

애플리케이션에는 세 모델 중 성능이 가장 좋게 나온 KoBART를 사용하여 개발자가 별점과 상관없이 긍정/요청/부정 중 원하는 리뷰를 볼 수 있게 구현했다.

Transformer 기반의 한국어 분류 모델들을 비교하기 위

해 KoBERT, KoGPT-2, KoBART를 사용하였으나 KoBART 이후에도 많은 한국어 분류 모델들이 나오고 있으므로, 다른 모델을 사용하여 학습을 진행할 시 프로그램을 효과적으로 발전시킬 수 있을 것으로 보인다.

감사의 글

본 연구는 2020년도 중소벤처기업부의 중소기업기술혁신개발사업(수출지향형)사업 지원에 의한 연구임 [S2879452]
본 연구는 과학기술정보통신부 및 정보통신기획평가원의 대학ICT연구센터육성지원사업의 연구결과로 수행되었음 (IITP-2023-2020-0-01846)
관련 부처에 감사드립니다.

참고문헌

[1] H.-J. Kang, “A Study on Analysis of Intelligent Video Surveillance Systems for Societal Security,” *Journal of Digital Contents Society*, Vol. 17, No. 4, pp. 273-278, August 2016. https://doi.org/10.9728/dcs.2016.17.4.273

[2] C. Song, B. C. Kim, and K. Park, “Impact of Consumer Rating on Mobile Application Sales: A Comparison between Korean and American Consumers,” *Korean Management Review*, Vol. 43, No. 5, pp. 1493-1518, October 2014.

[3] S. Park, H. Yang, M. Choe, M. Ha, K. Chung, and M. Koo, “Sentimental Analysis of YouTube Korean Comments Using KoBERT,” in *Proceedings of Korea Software Congress (KSC 2020)*, Online, pp. 1385-1387, December 2020.

[4] B. Pang, L. Lee, and S. Vaithyanathan, “Thumbs Up?: Sentiment Classification Using Machine Learning Techniques,” in *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing (EMNLP ’02)*, Philadelphia: PA, pp. 79-86, July 2002. https://doi.org/10.3115/1118693.1118704

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, ... and I. Polosukhin, “Attention is All You Need,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS ’17)*, Long Beach: CA, pp. 6000-6010, December 2017.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding,” arXiv:1810.04805, May 2019. https://doi.org/10.48550/arXiv.1810.04805

[7] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, “Improving Language Understanding by Generative Pre-Training,” 2018.

[8] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, ... and L. Zettlemoyer, “BART: Denoising Sequence-to-Sequence Pre-Training for Natural Language Generation, Translation, and Comprehension,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, pp. 7871-7880, July 2020. https://doi.org/10.18653/v1/2020.acl-main.703

[9] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Computation*, Vol. 9, No. 8, pp. 1735-1780, November 1997. https://doi.org/10.1162/neco.1997.9.8.1735

[10] SK Telecom. SKT Open Source KoBERT [Internet]. Available: https://sktelecom.github.io/project/kobert/.

[11] SK Telecom. SKT Source KoGPT2 [Internet]. Available: https://sktelecom.github.io/project/kogpt2/.

[12] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, “Language Models Are Unsupervised Multitask Learners,” *OpenAI Blog*, Vol. 1, No. 8, 9, 2019.

[13] GitHub. SKT-AI/KoBART: Korean BART [Internet]. Available: https://github.com/SKT-AI/KoBART.

이민아(Min-A Lee)

2022년 : 광운대학교 화학과 (학사)

2022년～현 재: 광운대학교 대학원 전자통신공학과 (석사과정 재학중)
※관심분야 : 인공지능, 자연어처리, 신호처리 등

박연지(Yeon-Ji Park)

2020년 : 광운대학교 컴퓨터소프트웨어학과 (학사)

2020년～현 재: 광운대학교 대학원 전자통신공학과 (석박통합과정 재학중)
※관심분야 : 인공지능, 자연어처리, 컴퓨터비전 등

나준영(Jun-Yeong Na)

2022년 : 광운대학교 전자통신공학과 (학사)

2022년～현 재: 광운대학교 대학원 전자통신공학과 (석사과정 재학중)
※관심분야 : 인공지능, 컴퓨터비전, 헬스케어, 블록체인 등

손채봉(Chae-Bong Sohn)

1993년 : 광운대학교 전자공학과 (학사)
1995년 : 광운대학교 대학원 전자공학과 (석사)
2006년 : 광운대학교 대학원 전자공학과 (박사)

1991년～1993년: 삼성전자 소프트웨어 멤버십 1기
2013년～2017년: 광운대학교 정보통신처 처장
2020년～2021년: 광운대학교 학생복지처 처장/인재개발원 원장
2006년～현 재: 광운대학교 전자통신공학과 교수
2022년～현 재: 광운대학교 대학원 방위사업학과 학과장
2023년～현 재: 광운대학교 중소기업산학협력센터 센터
2023년～현 재: 광운대학교 벤처스타트업아카데미 사업단장
※관심분야 : 인공지능, 자연어처리, 컴퓨터비전, 임베디드시스템 등