# 지식 증류 기법을 활용한 BERT 분석모델 경량화

Make Bert Model smaller and faster by applying distillation techniques

2021년

서강대학교 정보통신대학원

데이터사이언스 전공

장호섭

# 지식 증류 기법을 활용한 BERT 분석모델 경량화

Make Bert Model smaller and faster by applying distillation techniques

지도교수 구 명 완

이 논문을 공학석사 학위논문으로 제출함

2021년 1월 일

서강대학교 정보통신대학원

데이터사이언스 전공

장호섭

장호섭의 공학 석사학위 청구논문을 인준함

2021년 1월 일

주심 ㊞

부심 ㊞

부심 ㊞

# 목 차

# 표 목 차

# 그 림 목 차

# ABSTRACT

As pre-trained models with large scale become more prevalent in natural language processing, corporates, and research instutions struggle to operate these large deep learning models under constrained hardware budget. Due to the rapid increase in size of text datasets, large scale pre-trained models and heavy operating costs incurred remain challenging to effectively implements natural language models to the workplace. To solve these practical problems, various model compression techniques prove that it is possible to reduce the size of BERT model, while retaining most of language model capabilities.

In this paper, the size of distilled BERT model is 5-10 times smaller than original BERT model. The result of study shows that the accuracy of classification with BERT model is 96% and with distilled BERT model is 92%. In performance perspective, BERT classification takes 36 minutes on prediction, yet distilled BERT classification takes 5 minutes and 30 seconds. Considering trade-off between accuracy and performance, distilled BERT model is relatively efficient to implement on corporates, college and research institute.

# 초 록

최근, 대용량 데이터를 기반으로 학습된 Pre-trained 모델들이 자연어 처리에서 높은 성능을 보여 기업, 학교, 연구 기관에서 BERT, GPT 등 최신 pre-trained 모델들을 실제 서비스에 도입하기 시작했다. 하지만, 점점 사이즈가 커지는 최신 모델들과 비정형 데이터는 실제 운영하는데 높은 비용, 학습/추론 성능 이슈 등 실용적인 문제점이 나오기 시작했다. 이를 개선하기 위해서, 모델의 크기를 줄이면서 최대한 성능을 보존하는 모델 압축 방법론들이 나타나기 시작했다. 이번 논문에서는, 기존 BERT 모델의 가중치들을 효율적으로 새로운 모델에 전이할 수 있는 모델 압축 기법인 d지식 증류 기법을 활용해서 기존 모델들과 비교한다.

이번 논문 내에서, 지식 증류 기법을 적용한 BERT 모델은 기존 BERT 모델과 비교해 모델의 사이즈가 5-10배 작아졌다. 실험 결과, 기존 BERT 모델의 정확도는 96%, 압축된 모델은 92%의 긍부정 분류 정확도를 보였지만 성능 측면으로는 BERT는 36분 그리고 압축된 BERT 모델은 5분 30초로 7배 빠른 성능을 보여주는 것을 확인했다. 정확도와 성능 trade-off 관계를 고려하면 전체적으로는 지식 증류 기법이 적용된 BERT 모델이 기존 BERT 모델들과 비교 시, 기업, 학교, 연구소들과 같이 하드웨어, 성능 등 실용적인 측면을 고려하는 곳에서는 더욱 적합한 모델으로 판단이 된다.

# 제1장 연구 배경과 목적

## 1.1. 연구 배경

실제 기업에서는 방대한 데이터를 처리해야 되기 때문에 정확도도 중요하지만, 성능에 대한 이슈가 늘 발생한다. 실제 예시로, 프로젝트에서 소셜 데이터를 수집해서 긍부정 분류 방법론을 적용해 결과를 산출해내는 모델을 생성해서 파일럿에서 좋은 결과를 만들었지만, 실제 유저들에게 오픈을 했을 때, 하드웨어의 과부하와 실제 모델이 돌아간다고 해도 시간이 너무 오래 걸리는 이슈가 생겼다. 대다수의 유저들은 떠나갔고 남은 유저들마저도 모델의 처리속도가 느리기 때문에, 사용 빈도가 매우 낮다는 피드백을 받았다. 이러한 이슈 때문에, 실제 기업에서는 단순하게 모델의 정확성만 높은 것 보다는 정확도와 성능이 균등하게 높은 효율성이 좋은 모델을 선호한다.

본 논문에서는, 실제 기업에서 활용할 수 있는 해당 도메인 뉴스 데이터를 수집해서 분석 모델에 적합한 데이터로 전처리를 진행한 후, 정확도와 효율성을 고려한 분석 모델을 만들고 이를 활용하여 긍부정 분석을 진행하는 것을 목표로 한다.

## 1.2. 연구 문제

본 논문에서는 도메인 뉴스 데이터를 수집해서 모델의 정확성을 올리고 모델 압축 방법론을 활용하여 모델 수행 시간을 최적화 시킨다. 데이터는 긍정, 부정, 중립의 라벨링 작업을 거친 후, pretrained 된 모델을 학습시켜서 지도 학습을 진행한다. 예측 정확도와 처리 성능을 최적화시키는 모델을 구축하는 것을 목표로 한다.

실험 데이터는 자동차 관련 데이터를 활용하여 훈련 데이터, 검증 데이터, 테스트 데이터의 비율을 8:1:1 비율로 분리해서 사용했다. 모델 압축 방법론과 도메인 뉴스 데이터로 학습시킨 분석 모델을 기존 분석 모델들과 비교, 검증한다. 연구에 앞서 기존에 연구되었던 관련 연구들과 비교한다.

본 논문의 구성은 다음과 같다. 2장에서는 논문의 기초가 되는 이론적 배경에 대해 설명을 한다. 3장에서는 뉴스 데이터 수집 및 긍부정 분류 분석 모델을 개발하며 4장에서는 연구의 실험 결과에 대해 설명하고 평가한다. 마지막 5장에서는 본 연구의 결론과 한계점, 그리고 추후 연구 방향에 대하여 제시한다.

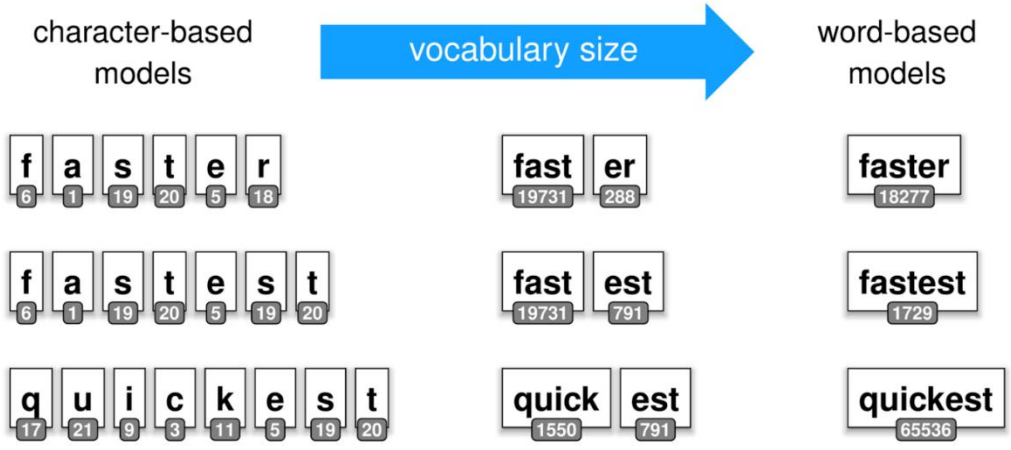
# 제2장 이론적 배경

## 2.1. BERT 모델

BERT는 Bidirectional Encoder Representations from Transformer의 약자로 Google AI Language에 있는 연구원에 의해 만들어진 알고리즘이다. 자연어 추론, Question Answering 등 다양한 자연처리 분야에서 GPT 시리즈와 압도적인 성능을 보이고 있다. BERT의 핵심 기술은 Transformer 구조에 bidirectional training 적용시켜서 만든 언어 모델로 위키피디아, 책 등 대용량 말뭉치 데이터로부터 준 지도 학습(semi-supervised learning)으로 미리 학습한다. 그 후, 자연어처리 특정 하위 분야 문제에 지도학습(supervised learning)으로 모델을 구성하는 방법으로 미리 학습된 모델(pre-trained model)을 사용하여 미리 학습된 파라미터들을 분석 데이터셋에 적합하게 재 학습(Fine-tuning) 해서 분석 모델의 정확도를 높인다. 대부분 자연어처리 하위 분야 문제(Downstream Tasks)에서 리더보드 1위를 차지하는 높은 성능을 보이는 모델로 KoBERT, Albert, Bert Multilingual 등 BERT를 기반으로 하는 파생 모델들이 계속해서 만들어지고 있다.

## 2.2. 토큰화 (Tokenization)

텍스트 분석에서 단어를 식별하고 추출하는 토큰화 (Tokenization)는 분류 모델 생성 전에 선행되어야 하는 작업이다. 텍스트 데이터를 학습한 모델의 크기는 단어의 개수에 영향을 많이 받는다. 그렇기 때문에 RNN 에 이용되는 vocabulary, word embedding 벡터의 종류는 제한이 있다. 하지만 vocabulary 개수가 제한되면 임베딩 벡터로 표현하지 못하는 단어가 생긴다. 이를 해결하기 위해서, BERT 의 토큰화 방법론으로 활용되는 Word Piece Model (WPM)이 만들어졌다.




[그림 1] Word Piece Model 예시

WPM 은 언어에 따라서 달라지는 형태소 분석 대신에 통계적인 특성을 사용하여 기본 단위를 추출하는 방법으로, 음절을 기준으로 하여

음절과 음절의 동시에 등장하는 빈도가 높으면 두 음절을 합치면서 새로운 어휘를 생성한다. 생성된 어휘 중에서 가장 높은 likelihood 를 보이는 어휘를 어휘 사전에 추가한다. 결론적으로, WPM 방식을 사용하면 최소 크기는 하나의 음절이고 최대 크기는 하나의 어절이 된다.

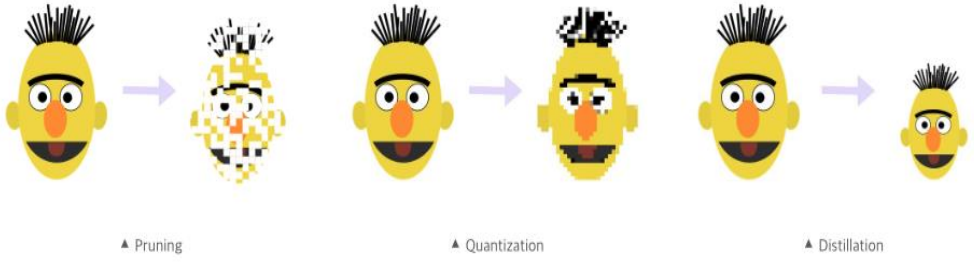
## 2.3. BERT 모델 압축 방법론

이미지, 텍스트 분야 내에서 딥러닝 분야가 점차 확산되면서 점점 더 성능이 좋은 딥러닝 알고리즘을 구축하는 것이 Google, Facebook 등 다양한 IT 기업과 딥러닝을 연구하는 대학교들의 중요한 연구과제이다. 딥러닝 알고리즘의 사이즈를 크게 해서 정확도를 올리는 것이 가장 보편적으로 사용되고 있는 방법이다. 하지만 단순하게 레이어를 두텁게 쌓아서 정확도를 올리는 방법은 높은 정확도를 보이고 있지만, 실용적인 문제(메모리, 과적합 등)가 이슈가 되면서 효율성(efficiency)에 대한 니즈가 나타나기 시작했다.

| Model | Hidden Size | Parameters | RACE (Accuracy) |
|---|---|---|---|
| BERT-large (Devlin et al., 2019) | 1024 | 334M | 72.0% |
| BERT-large (ours) | 1024 | 334M | 73.9% |
| BERT-xlarge (ours) | 2048 | 1270M | 54.3% |

[그림 2] BERT 모델 별 파라미터 및 정확도 비교

위 그림은, 단순하게 모델의 크기를 늘려서 모델을 만든 케이스지만 오히려 과도한 파라미터와 레이어 수의 증가는 과적합으로 인해 모델의 정확도가 저하될 수 있는 문제점도 존재한다. 이러한 문제점을 해결하기 위해서, 효율적인 모델 압축 방법론들에 포커스가 맞춰지고 있다. 모델 압축은 다양한 방법을 이용하여 딥러닝 모델의 성능을 유지하면서 크기를 줄이는 것을 목표로 하고 있다. 현재 사용되는 모델 압축 방법론은 대략적으로 5가지가 있다.




[그림 3] 압축 모델 방법론 예시

1. 가지치기 (Pruning)

주요 가중치를 제외한 비교적 작은 가중치 값을 모두 0으로 치환하여 네트워크의 모델 크기를 줄이는 기술이다. 학습 후에 불필요한 부분을 제거하는 방식으로 어텐션, 레이어 제거 등 다양한 방법을 사용해서 모델을 압축한다. Pruning 방법론은 크게 Neuron Pruning과

Weight Pruning 방법론으로 나눠진다.

2. 가중치 분해 (Weight Factorization)

   가중치 행렬을 분해하여 두 개의 작은 행렬의 곱으로 근사하는 방법이다.

3. 양자화 (Quantization)

   양자화는 원래 연속된 아날로그 값을 연속적이지 않은 디지털 값으로 변환하는 것을 말하며, 정해진 단위값으로 근사 시켜서 통일시키는 것을 일컫는다. 이로 인해, 양자화를 진행하면 정보의 손실이 일어나 정밀도가 떨어지지만, 데이터의 크기가 줄어드는 장점이 있다. 자연어 처리에도 동일하게 이론을 접목시켜, 부동 소수점 값을 잘라내서 더 적은 비트만을 사용해서 모델 압축을 진행한다.

4. 가중치 공유 (Weight Sharing)

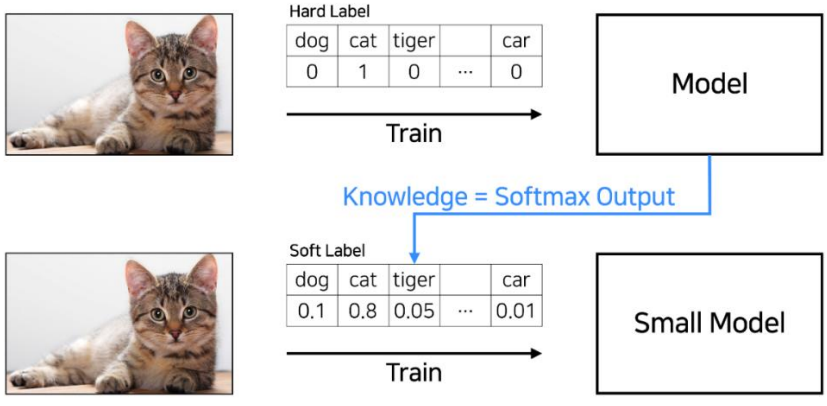
   모델의 일부 가중치를 다른 파라미터들과 공유하는 방식으로, ALBERT는 BERT self-attention layer과 같은 가중치 행렬들을 사용하고 있다.

5. 지식 증류 (Knowledge Distillation)

   이번 논문에서는 지식 증류 (Knowledge Distillation) 방법론을 활용한 긍부정 분류를 적용해서 기존의 다른 알고리즘과 성능, 정확도를 비교하는 것을 목표로 한다.

## 2.4. DISTILLATION (지식 증류 기법)

기존 pre-trained 된 모델을 Teacher 모델으로 가정을 하고 teacher 모델을 토대로 더 작은 모델 즉 student 모델을 생성하는 모델 압축 방법론을 지식 증류 기법(Knowledge Distillation)이라고 부른다. 지식 증류(Knowledge Distillation)은 Hinton이 작성한 논문 내에서 "압축 기술로 Student 모델이 Teacher 모델의 행동을 재생산한다" 라고 정의하고 있다. 기존 데이터에는 이미지와 텍스트에 대한 라벨링이 되어 있으며 이를 Hard Label이라고 부른다. 정답을 충분히 학습시킨 Teacher 모델을 Student 모델에 가중치를 전달하면 대부분 결과도 그 정답과 일치할 것이다.




[그림 4] Softmax Output 결과값 예시

단순하게 Hard Label 형식으로 Cat을 1로 지정하고 나머지는 0

으로 지정을 해서 학습을 하지만, 다양한 데이터셋을 학습시켜서 만들어진 실제 모델이 만들어낸 결과 분포를 살펴보면, 모델에서 Softmax를 통해서 산출되어서 나온 결과는 정답 이외에도 다른 물체에 대한 정보를 담고 있다. 이러한 결과 분포는 실제 이미지에 대한 해석을 더 풍부하게 해주는 효과가 있다. 이렇게 정보가 묻어 나오는 것이 마치 석유의 부산물들이 증류탑에서 나오는 양상과 유사하기 때문에 지식 증류, Knowledge Distillation라고 부른다.

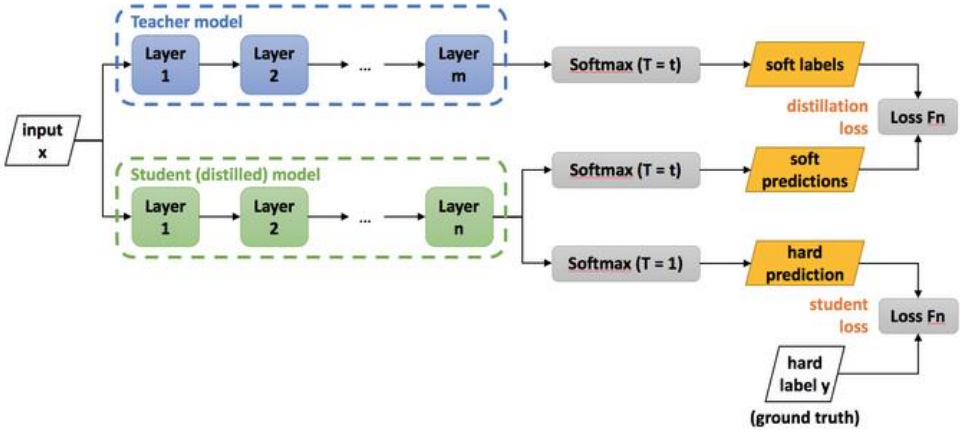
Hinton et al. 논문에서 제시하는 지식 증류의 핵심은 Teacher 모델이 산출한 Softmax output을 새로 학습을 진행하면 간접적으로 Teacher 모델이 학습한 바를 반영하게 되므로 더 효율적으로 모델을 학습시킬 수 있으며, Student 모델은 효율적으로 Teacher의 모델의 웨이트를 학습하고 사이즈 또한 상대적으로 작은 규모로 구성될 수 있다.

| Model | **Score** | CoLA | MNLI | MRPC | QNLI | QQP | RTE | SST-2 | STS-B | WNLI |
|---|---|---|---|---|---|---|---|---|---|---|
| ELMo | 68.7 | 44.1 | 68.6 | 76.6 | 71.1 | 86.2 | 53.4 | 91.5 | 70.4 | 56.3 |
| BERT-base | 77.6 | 48.9 | 84.3 | 88.6 | 89.3 | 89.5 | 71.3 | 91.7 | 91.2 | 43.7 |
| DistilBERT | 76.8 | 49.1 | 81.8 | 90.2 | 90.2 | 89.2 | 62.9 | 92.7 | 90.7 | 44.4 |

[그림 5] BERT와 지식증류 기법 적용한 BERT 모델의 정확도 비교

BERT에 대해서도 지식 증류 기법을 동일하게 적용할 수 있다. Huggingface의 Sanh 논문은 원 BERT 모델에 대해서 훈련시킨 다음, 원 모델이 예측한 Masked LM 분포를 이용하여, 레이어 수가 절반인 새

모델을 학습시킨 결과, 파라미터 수가 절반임에도 불구하고 원 BERT의 성능과 대등한 성능을 보였으며, 심지어 CoLA, SST-2 등 몇몇 작업에 대해서는 더 우수한 성능을 보였다. 즉, pretrain 단계에서 BERT 모델의 출력은 해당 단어에 대한 언어 모델 그 이상의 정보를 가지고 있다는 것을 의미한다.




[그림 6] 지식증류 기법 모델의 구성

Distilled BERT 모델을 생성할 때, 크게 2개의 트랙으로 진행을 한다.

1. Ground Truth와 Student 모델을 학습시키면서 모델 분류 결과 차이를 Cross entropy로 loss를 계산
2. Teacher 모델과 Student 모델의 output logit을 Softmax로 변환한 값의 차이를 Cross entropy로 loss를 계산
3. 1번과 2번의 Weighted Average (가중평균) 값을 산출해서 최종

loss 값 산출

Teacher 모델에서 나오는 Softmax output의 Loss값과 Student 모델에서 나오는 실제 값과 예측 값의 Loss값을 계속 줄여 나가면서 모델의 성능을 높이고 있다.

# 제3장 텍스트 분류 모델 개발

## 3.1. 지식 증류 모델 개발

새로운 압축 모델을 생성하기 위해서는 텍스트 데이터셋이 필요하다. 뉴스를 기반으로 하는 긍부정 모델을 생성하기 때문에, 정확도를 높이기 위해서 AI HUB에서 제공하는 문어체 뉴스 데이터 80만건을 학습에 적용한다. 제목, 내용 데이터를 같은 sentence 데이터로 합쳐서 구성을 한 후, sentence로 구성된 데이터를 양자화(binarize) 시키고 각 토큰들을 모델 내 단어로 변형을 시킨 후 binarized_text 파일로 저장을 한다. 그리고 각 토큰의 빈도 수를 token_counts 파일에 저장을 한다.

BERT의 파생 모델 중 한국어를 베이스로 만든 모델들은 다수 존재한다. Kobert, Hanbert, Kcbert 등 다양한 파생 알고리즘이 존재하지만 이번 논문에서는 SNS, 트위터 기반의 데이터를 기반으로 만들어진 Kcbert를 기반으로 지식 증류 기법을 적용해서 압축된 Kcbert 버전의 모델을 생성한다. Kcbert를 Teacher 모델로 선정하고 student 모델은 Teacher 모델과 동일하게 내부 파라미터 세팅 정보인 training config를 설정한 후, distillation 코드를 실행시켜서 지식 증류 기법이 적용된 student 모델을 생성한다.

Distilled 코드를 실행할 때는, BertForSequenceClassification으로

pretrained 모델을 가져오는 것이 아니라, Distiled 버전인 DistilBertForSequence- Classification으로 pretrained 모델을 호출한 후, 동일하게 코드를 실행한다.

## 3.2. 긍부정 모델 분류기 생성

긍부정 분석을 위해서는 뉴스 데이터 5000건을 수집해서 각각 긍정, 부정, 중립 라벨링을 생성한다. 라벨링을 생성한 후, 해당 데이터를 훈련, 검증, 테스트 데이터 각각 8:1:1 비율로 분류하여 사용한다.

데이터 크롤링 코드로 자동차 산업 관련 뉴스 데이터 5천개를 수집한 후, 데이터를 긍정, 부정, 상관없음(중립)으로 라벨을 부여했다. 긍정, 부정, 중립 데이터의 비율은 각각 1:1:3으로 구성하고 있다. 라벨링을 부여하는 작업이 필수적인 이유는 감성사전 구축을 통해 라벨링을 부여하면 정확도가 떨어지는 이슈가 있어 시간이 오래 걸리지만 개별 데이터를 라벨링을 부여하는 방법을 적용해서 진행한다.

[표 1] 긍부정 분류 데이터 비율

| 인덱스 | 긍부정 분류 | 데이터 수 |
|---|---|---|
| 0 | 중립 | 3000 |
| 1 | 긍정 | 1000 |
| 2 | 부정 | 1000 |

긍부정 모델은 총 3가지의 긍정, 부정, 중립 라벨링 분류를 해서 진행을 했다. BERT 모델 내의 Pre-trained 된 모델들을 가져와서 BertForSequenceClassification (다중 분류)를 할 수 있는 패키지를 가져와서 긍부정 분류를 수행한다. BERT의 파생 모델들은 동일한 조건 하에서 테스트해서 결과를 비교한다.

# 제4장 실험 및 평가

## 4.1. 실험 환경

시스템 구현은 윈도우 64비트 운영체제에서 Python 프로그래밍 언어를 기반으로 진행하였다. 아래 표는 본 실험에서 사용한 개발환경 및 주요 라이브러리이다.

[표 2] 개발 환경 및 라이브러리

| | | |
|---|---|---|
| 개발환경 | 운영체제 | 64비트 운영 체제, x64 기반 프로세서 |
| | CPU | AMD Ryzen 9 3900X 12-Core Processor<br>3.79 GHz |
| | GPU | GeForce GTX 2080 Ti |
| | RAM | 32GB |
| 개발언어 및 라이브러리 | 언어 | Python 3.7.3 |
| | Framework | Tensorflow 1.14.0 |
| | | PyTorch 1.2.0 |

## 4.2. 실험 결과

[표 3] Kcbert 와 Distil Kcbert Training configuration 비교

| TRAINING CONFIGURATION | KCBERT | DISTIL KCBERT |
|---|---|---|
| ACTIVATION | gelu | gelu |
| DIMENSION | 3072 | X |
| LAYER | 12 | Y |
| HEADS | 12 | 12 |
| VOCAB SIZE | 30000 | 30000 |
| MAX POSITION EMBEDDING | 300 | 300 |
| INITIALIZER_LENGTH | 0.02 | 0.02 |

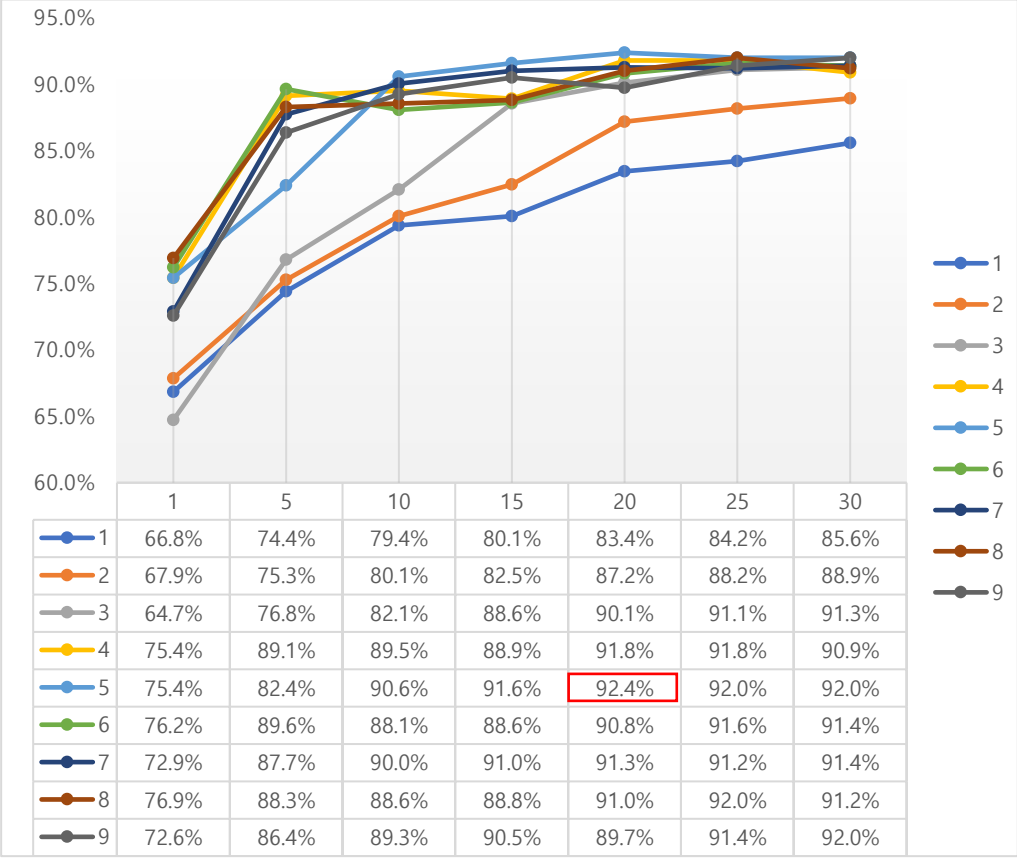
학습 전, KCBERT와 DISTIL KCBERT Training configuration 내에서 파라미터 세팅을 한다. ACTIVATION, HEADS, VOCAB SIZE 등 일부 파라미터들은 동일하게 세팅을 하지만, DIMENSION, LAYER 숫자를 변환하며 파라미터 개수 조정을 하면서 총 9개의 모델을 생성을 했다.

[표 4] Distil Kcbert 의 파라미터 개수의 변화에 따른 소요 시간 측정

| 모델 INDEX | 레이어 숫자 | DIMENSION | 파라미터 개수 | 예측 시간 (초) |
|---|---|---|---|---|
| 1 | 1 | 192 | 26,555,376 | 128 |
| 2 | 1 | 768 | 27,440,688 | 136 |
| 3 | 1 | 3072 | 30,981,936 | 177 |
| 4 | 3 | 192 | 31,878,000 | 292 |
| 5 | 3 | 768 | 34,533,936 | 330 |
| 6 | 3 | 3072 | 45,157,680 | 475 |
| 7 | 6 | 192 | 39,861,936 | 548 |
| 8 | 6 | 768 | 45,173,808 | 649 |
| 9 | 6 | 3072 | 66,421,296 | 882 |

Hidden layer와 Hidden Dimension의 개수를 변경하면서 총 9가지의 지식 증류 기법 모델 생성하면서 모델 별 소요시간을 분석했다. 디멘젼의 개수보다 레이어의 개수가 증가할 수록 모델의 예측 소요시간이 급증하는 패턴을 보인다. 성능을 고려할 시, 상대적으로 레이어의 비중을 낮게, DIMENSION의 비중을 높게 모델 생성 필요하다.

[표 5] Distil 모델 9 개의 Epoch 별 정확도 변화 비교




| | 1 | 5 | 10 | 15 | 20 | 25 | 30 |
|---|---|---|---|---|---|---|---|
| 1 | 66.8% | 74.4% | 79.4% | 80.1% | 83.4% | 84.2% | 85.6% |
| 2 | 67.9% | 75.3% | 80.1% | 82.5% | 87.2% | 88.2% | 88.9% |
| 3 | 64.7% | 76.8% | 82.1% | 88.6% | 90.1% | 91.1% | 91.3% |
| 4 | 75.4% | 89.1% | 89.5% | 88.9% | 91.8% | 91.8% | 90.9% |
| 5 | 75.4% | 82.4% | 90.6% | 91.6% | 92.4% | 92.0% | 92.0% |
| 6 | 76.2% | 89.6% | 88.1% | 88.6% | 90.8% | 91.6% | 91.4% |
| 7 | 72.9% | 87.7% | 90.0% | 91.0% | 91.3% | 91.2% | 91.4% |
| 8 | 76.9% | 88.3% | 88.6% | 88.8% | 91.0% | 92.0% | 91.2% |
| 9 | 72.6% | 86.4% | 89.3% | 90.5% | 89.7% | 91.4% | 92.0% |

가장 높은 정확도를 보여주는 모델은 5번으로 20번의 epoch 실행 시 임계점에 도달해 가장 높은 정확도를 보였다. 5번의 파라미터인 레이어 3개, 디멘젼 768개 이상 사용해서 모델을 구성하면 데이터들이 overfitting이 발생하고 있다는 점을 알수 있다. 지식증류기법 모델의 예측 정확도의 한계점이 존재해서 92.4%의 정확도를 달성했다.

긍부정 모델은 Transformer을 기반으로 하는 BERT 모델을 사용했

으며 BERT모델의 Pre-trained 모델 중 다국어 모델을 지원하는 BERT multilingual 모델, 한국어 모델에 특화되어있는 Kobert, Kcbert, 경량화 모델인 Albert와 논문에서 구축한 kcbert 기반의 지식 증류 기법이 적용된 모델을 비교하는 것을 목적으로 한다. 실험 결과 및 평가는 동일한 데이터 셋을 긍부정 분류를 진행했을 때 정확도와 성능을 비교한다.

[표 6] BERT 알고리즘 별 정확도/성능 비교

| 알고리즘 | 긍부정 분석 | 총 소요시간 | 비고 |
|---|---|---|---|
| KOBERT | 96.9% | 36분 15초 | SNS 데이터 기반 학습 모델 |
| KCBERT_base | 96.7% | 36분 20초 | 뉴스, 위키 등 텍스트 데이터 기반 학습 모델 (SKT 주요 모델) |
| BERT_multilingual | 94.3% | 42분 30초 | 다국어 목적으로 만든 BERT 기본 모델 |
| distil kcbert (7번 모델) | 92.4% | 5분 30초 | Kcbert에 지식증류 기법을 도입한 모델 |
| Albert_Base | 89.2% | 10분 | BERT 대표 경량화 모델 |

정확도와 성능은 5000개의 뉴스 데이터셋을 긍정, 부정 혹은 중립으로 분류하는 모델로 평가를 진행했다. Kcbert, Kobert 등이 96%대의

높은 정확도가 나왔지만 소요시간은 36분으로 BERT multilingual보다 전체적으로 정확도, 성능 측면에서는 좋은 결과가 나왔다. 경량화 모델인 Albert는 89%의 정확도로 성능 측면에서는 기존 BERT 모델 대비 낮지만 소요시간 측면에서는 10분대로 3배 빠른 스피드를 보여줬다. 마지막으로, Kcbert에 지식증류 기법을 적용한 7번 모델은 92.4%의 준수한 정확도와 총 5분 30초가 소요되며 기존 Kcbert 대비 7배 빠른 스피드를 보여줬다. 3%의 정확도 차이가 나지만 성능 측면에서 Kcbert와 Distil Kcbert는 크게 차이가 나기 때문에, 효율성인 측면으로 고려했을때는 Distil Kcbert가 더 좋음을 확인할 수 있다.

# 제5장 결론 및 논의

지식 증류 기법 코드를 적용시켜서 새로운 student 모델을 생성하면, 소폭 정확도가 떨어지지만 성능 측면으로는 크게 향상하는 것을 볼 수 있다. 또한, 지식 증류 기법 코드를 학습시킬 때, 도메인 관련 데이터로 학습을 시킬 시 정확도의 저하를 어느 정도 막을 수 있다. 결론적으로, 실용적인 측면에서는 지식 증류 기법이 적용된 student 모델은 속도 측면에서 기존 BERT 모델 대비 압도적인 성능을 보이고 있어 실시간 텍스트 데이터 전처리 혹은 신속하게 텍스트 분류 결과를 산출해야 되는 기업들에게는 필수적인 요소라고 생각한다.

향후 과제로는 현재 일하고 있는 기업에서 하드웨어를 지원받아서 teacher 모델을 student 모델로 지식 증류 기법을 적용할 시, 사용하는 도메인 데이터를 단순하게 뉴스 관련 데이터만 사용하는 것이 아닌, SNS와 위키 데이터 등 다양한 텍스트 데이터를 수집해서 정확도를 올릴 예정이다. 또한, 단순하게 distillation 방법론 뿐만 아니라, Pruning, Quantization 등 다양한 모델 압축 방법론을 적용해서 최적의 모델을 구축할 예정이다.

# 참고 문헌

[1] Vinctor Sanh, Lysandre Debut, Julien Chaumond, Thomas Wolf, “DistilBert, a distilled version of BERT: smaller, faster, cheaper and lighter”, arXiv:1910.01108, 2019.

[2] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, “Distilling the Knowledge in a Neural Network”, arXiv:1503.02531, 2015.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”, arXiv:1810.04805, 2018.

[4] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, Qun Liu, “TinyBERT: Distilling BERT for Natural Language Understanding”, arXiv:1909.10351, 2019.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, “Attention Is All You Need”, arXiv:1706.03762, 2017.

[6] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer, “Deep contextualized word representations”, arXiv:1802.05365, 2018.

[7] 국립국어원 표준국어대사전 [Online]. Available:

https://stdict.korean.go.kr (downloaded 2020, Dec. 29)

[8] 전창욱, 최태균, 조중현, 신성진, “텐서플로 2와 머신러닝으로 시작하는 자연어처리”, pp. 366-499, 2019.

[9] 이용주, 문용혁, 박준용, 민욱기, “경량 딥러닝 기술 동향 (기술동향자료)”, 한국전자통신연구원, pp. 40-50, 2019.

[10] 옥정우, 노병희, "감성사전 구축과 SVM을 이용한 온라인 뉴스 댓글의 부정 및 긍정 성향 분석", 한국통신학회 학술대회논문집, pp. 1189-1190, 2019.

[11] 김동성, 이상원, 김현건, 김종우, "온라인 뉴스를 활용한 인공지능에 대한 사회적 여론의 감성 분석 연구", *한국지능정보시스템학회 학술대회 논문집*, pp. 75-76, 2018.

[12] SKTBrain/KoBERT [Online]. Available: https://github.com/SKTBrain/KoBERT (downloaded 2020, Dec. 30)

[13] Beomi/KcBERT [Online]. Available : https://github.com/Beomi/KcBERT (downloaded 2020, Dec. 30)

[14] Distillation github code, huggingface [Online]. Available: https://github/com/huggingface/transformers/tree/master/examples/distillation (downloaded 2020, Dec 30)