

NAVER CONNECT 부스트캠프 AI Tech

NLP 기초 프로젝트

문장 간 유사도 측정 Semantic Text Similarity(STS)

Member

김주원_T5056, 강민재_T5005, 김태민_T5067, 신혁준_T5119 윤상원_T5131

1. 프로젝트 개요

프로젝트 주제	Semantic Text Similarity (STS) : 두 텍스트가 얼마나 유사한지 판단하는 NLP Task
프로젝트 구현내용	1. Hugging Face 의 Pretrained 모델과 STS 데이터셋을 활용해 두 문장의 0 과 5 사이의 유사도를 측정하는 AI 모델을 구축 2. 리더보드 평가지표인 피어슨 상관 계수(Pearson Correlation Coefficient ,PCC)에서 높은 점수(1 에 가까운 점수)에 도달할 수 있도록 데이터 전처리, 증강, 하이퍼 파라미터 튜닝을 진행
개발 환경	GPU : Tesla V100 서버 5 개 (RAM32G) / K80, T4, and P100 랜덤 할당(RAM52G) / GeForce RTX 4090ti 로컬 (RAM 24GB), Rtx3060ti 8gb 로컬 2 대 (RAM 8 GB) 개발 Tool : PyCharm, Jupyter notebook, VS Code [서버 SSH 연결], Colab Pro +, wandb
협업 환경	Github Repository : Baseline 코드 공유 및 버전 관리, issue 페이지를 통한 실험 진행 Notion : STS 프로젝트 페이지를 통한 역할분담, 아이디어 브레인 스토밍, 대회관련 회의 내용 기록 SLACK, Zoom : 실시간 대면/비대면 회의
프로젝트 구조 및 사용 데이터셋의 구조도	<pre> level1_semantictextsimilarity-nlp-11 ├── wandb_tuning.py # wandb sweep을 이용한 hyperparameter tuning을 위한 모델 학습 코드. ├── train.py # hyperparameter 재현용 모델을 train하는 코드로 infer를 위해 모델을 구축하는 코드 ├── infer.py # 튜닝된 모델을 이용한 최종적인 예측값을 출력 하는 코드 ├── data │ ├── dev.csv # STS dev 데이터 셋 : 550개 │ ├── test.csv # STS test 데이터 셋 : 1,100개 │ ├── Augmented_train_data_v1.csv. # STS train data 증강 데이터셋 : 15910개 │ │ │ │ │ │ # 구성 : under sampling + swap sentence + copied sentence + uniform distribution │ ├── Augmented_train_data_v3.csv. # STS train data 증강 데이터셋 : 18460개 │ │ │ │ │ │ # 구성 : under sampling + swap sentence + copied sentence + uniform distribution │ │ │ │ │ │ # + Synonym Replacement(품사 선택(명사, 조사) 교체) │ ├── train.csv # STS train 데이터 셋 : 9,324개 │ ├── .gitignore │ └── Readme.md </pre>

2. 프로젝트 팀 구성 및 역할

- 대부분의 팀원들이 첫 NLP 도메인의 프로젝트만큼 명확한 기준을 가지고 업무를 구분한 것보다 다양한 인사이트를 기르기 위해 데이터 전처리부터 모델 튜닝까지 End-to-End 로 경험하는 것을 목표로 하여 협업을 진행했다. 따라서 각자 튜닝할 모델을 할당하여 하이퍼 파라미터 튜닝을 하고 데이터 전처리, 증강 등 본인의 아이디어를 구현하되 서로의 내용이 겹치지 않도록 분업을 하여 프로젝트를 진행했다.

이름	역할
강민재	모델 튜닝 (electra-kor-base , koelectra-base-v3-discriminator), 데이터 증강 (back translation / switching sentence pair /임의 글자 삽입 및 제거), 데이터 전처리 실험 (레이블 정수화 및 노이즈 추가), Ensemble 실험 (output 평균, 표준편차 활용), EDA (글자수 기반 데이터 분포 분석)
김태민	Hugging Face 기반 Baseline 코드 작성 , Task 에 적합한 모델 Search 및 분배 , 모델 실험 총괄 , 데이터 전처리 실험 (Random Token Masking , Label Random Noise, Fill Random Token Mask, Source Tagging), Custom Loss 실험 (Binary Cross Entropy + Focal Loss), 모델 튜닝 (xlm-roberta-large, electra-kor-base), 모델 Ensemble
김주원	모델 튜닝 (kobigbird-bert-base, electra-kor-base), EDA (라벨 분포 데이터 분석), EDA 기반 데이터 증강 아이디어 제시 , 데이터 증강 (Easy Augmented Data SR 증강), 팀 협업 프로세스 관리 (Github 팀 관리 + 팀 Notion 페이지 관리) , Custom Loss 실험 (RMSE)
윤상원	모델 튜닝 (koelectra-base-finetuned-nsmc, KR-ELECTRA-discriminator 모델 튜닝), 데이터 증강 (label rescaling, 단순 복제 데이터 증강, 어순 도치 데이터 증강, under sampling + swap sentence + copied sentence + uniform distribution + random noise), 모델 Ensemble
신혁준	모델 튜닝 (KR-ELECTRA-discriminator, mdeberta-v3-base-kor-further) 데이터 증강 (맞춤법 교정 증강,EDA(Easy Data Augmentation) SR(Synonym Replacement)품사 선택(명사, 조사) 교체 + swap sentence + copied sentence, Data Distribution), 데이터 전처리 실험 (맞춤법 교정)

3. 프로젝트 수행 절차 및 방법

팀 협업을 위해 프로젝트 관련 Ground Rule 을 설정하여 프로젝트가 원활하게 돌아갈 수 있도록 팀 규칙을 정했으며, 날짜 단위로 간략한 목표를 설정하여 협업을 원활하게 진행할 수 있도록 계획을 하여 진행했다.

3.1 협업 관련 Ground Rule

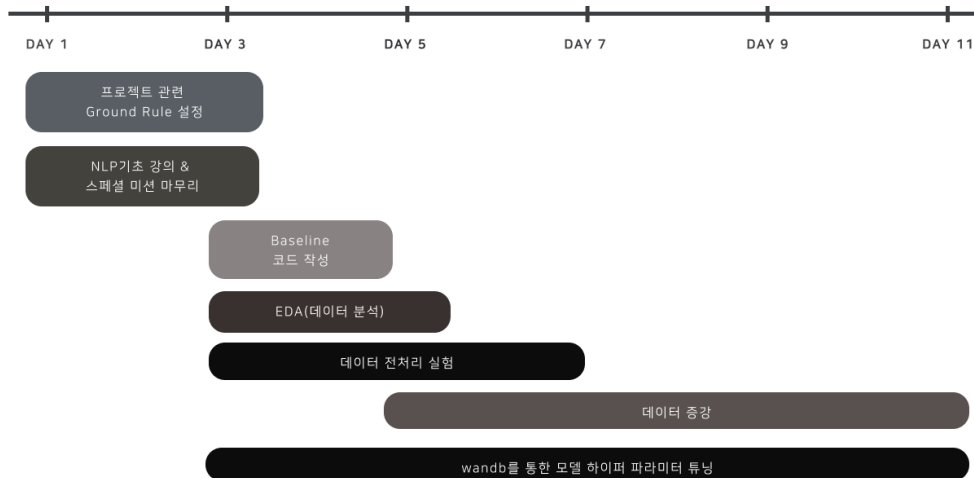
- a. **실험 관련 Ground Rule:** 본인 실험 시작할 때, Github issue 에 “[score 점수(없다면 --)] 모델이름, data = 데이터 종류, 전처리 종류, 데이터 증강 종류”양식으로 issue 를 올린 뒤 실험을 시작한다.
- b. **Commit 관련 Ground Rule:** git commit & push 는 한번 실험할 때 마다 진행한다. 코드 수정 내용, 점수, 관련된 issue 가 들어가도록 commit 하고 개인 branch 에 push 한다.
- c. **Submission 관련 Ground Rule:** 각 사람별로 하루 submission 횟수는 2 회씩 할당한다. 추가로 submission 을 하고 싶으면 SLACK 단체 톡방에서 해당 날짜에 submission 계획이 없는 혹은*횟수가 남은 사람에게 물어봐서 여유가 된다면 추가 submission 가능하다.
- d. **Notion 관련 Ground Rule:** 원활한 아이디어 브레인스토밍과 분업을 위해 회의를 할 경우 노선에 기록하며, 아이디어를 팀회의를 통해 실험을 팀원에게 할당한다.

3.2 프로젝트 진행 Time line

- (1~3 일차): NLP 기초 대회 관련 대회 강의 및 스페셜 미션 완료 & 협업 관련 Ground Rule 설정
- (3~4 일차): STS Baseline 코드 완성 & EDA(Exploratory Data Analysis) & 전처리/증강 관련 아이디어 회의
- (5~14 일차) : 전처리, 데이터 증강, 하이퍼 파라미터 튜닝 등 아이디어 구현 및 실험 진행

세부적인 대회 진행 절차는 아래와 같다.

Product RoadMap



4. 프로젝트 수행 결과

4.1. 순위

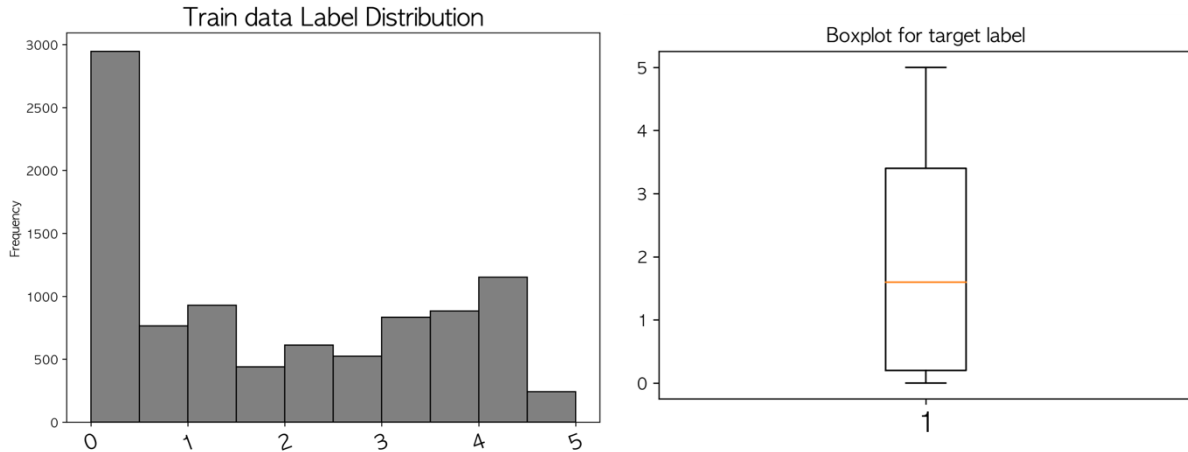
순위	분류	점수(Pearson Correlation)	제출 횟수
1	Public Score (대회 진행)	0.9367	64
2	Private Score(최종)	0.9403	64

4.2. EDA(Exploratory Data Analysis)

주어진 데이터의 문장의 길이 관련, 라벨 분포와 관련된 데이터 분석을 진행했고, 이를 통해 문제 해결 전략을 수립하였다.

4.2.1 라벨 분포 관련 EDA

데이터의 분포에 대한 편향을 확인해보기 위해 target column 인 Label 에 대한 데이터 편향을 조사하였고, Label 구간을 1 로 잡아 0,0~1,1~2,2~3,3~4,4~5,5 의 데이터들은 어떤 특징을 가지고 있는지 데이터 분석을 통해 파악해보았다.

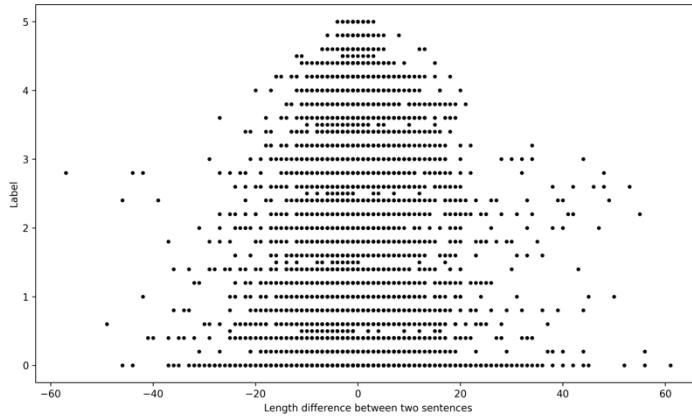


우선 데이터 분포에 대해서 살펴보면 우측 box-plot 을 보면 데이터 Label 의 평균은 0~5 의 중간 값인 2.5 가 아닌 2 아래인 1.8 정도에 형성된 것을 볼 수 있다. 실제로 값을 파악해봤을 때 평균은 1.8, 중간 값은 1.6, 하위 25% 값은 0.2 이하로 많은 데이터가 0 에 가까운 값으로 치우친 것을 알 수 있다. 또한 좌측처럼 Label 분포를 막대 그래프로 그렸을 때 0 이하인 값이 매우 많고, 5 값이 매우 적은 것을 알 수 있다. 따라서 분포가 불균형하고 한 쪽 Label 에 치우쳐 있으므로 class imbalance 와 유사하게 편향이 생겨 현재 충분히 많지 않은 데이터 셋에서 학습이 잘 안 될 가능성이 있다. 따라서 모델에 학습을 시켰을 때 성능이 좋지 않다면, 편향을 줄여주는 방법으로 데이터 증강을 하거나, under sampling 을 하면 성능의 개선 여지가 있다.

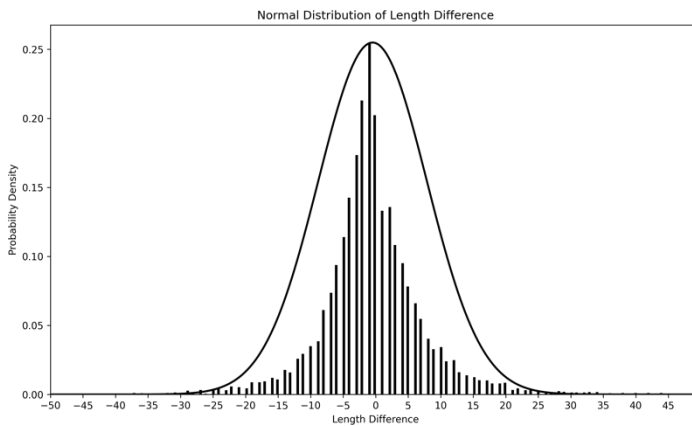
추가로 데이터 증강이나 전처리를 할 때 참고할 수 있도록 Label 별 데이터 특징을 아래와 같이 살펴 보았다.

예시 Label	sentence1	sentence2	속한 Label 구간	특징
5.0	그냥재미없음;;;	그냥 재미 없음.;	5.0	문장 부호의 갯수가 다르거나, 띄어쓰기가 다르거나, 영어의 대소문자만 다르고 문장이 거의 일치한다.
4.2	앗 제가 접근권한이 없다고 뜬니다;;	오, 액세스 권한이 없다고 합니다	4.0~4.9	한, 두 단어가 다르고 뜻이 거의 일치한다.
3.8	전 암만 찍어도 까만 하늘.. πππ	암만 찍어도 하늘은 까맣다.. πππ	3.0~3.9	여러 개의 단어가 다르지만, 전체적인 맥락의 뜻은 거의 일치한다.
2.4	주택청약조건 변경해주세요	주택청약 무주택기준 변경해주세요.	2.0~2.9	비슷한 주제를 담고 있으나, 전체적인 뜻이 다르다.
1.0	알콜중독자가 기초생활수급자?	기초생활수급자 장애인을 위하여	1.0~1.9	비슷한 소재가 쓰인다. 전체적인 맥락과 뜻은 다르다.
0.6	이렇게 귀여운 쥐들은 처음이네요. ㅎㅎㅎ	이렇게 지겨운 공포영화는 처음..	0.1~0.9	몇개의 문구, 단어가 일치할 수 있지만 주제, 맥락이 다르다.
0	뿌듯뿌듯 하네요!!	꼬옥 실제로 한번 봐어요 뿌뿌뿌~!~!	0	대부분 같은 단어도 존재하지 않으며, 주제, 맥락이 많이 다르다.

4.2.2 문장 길이관련 EDA



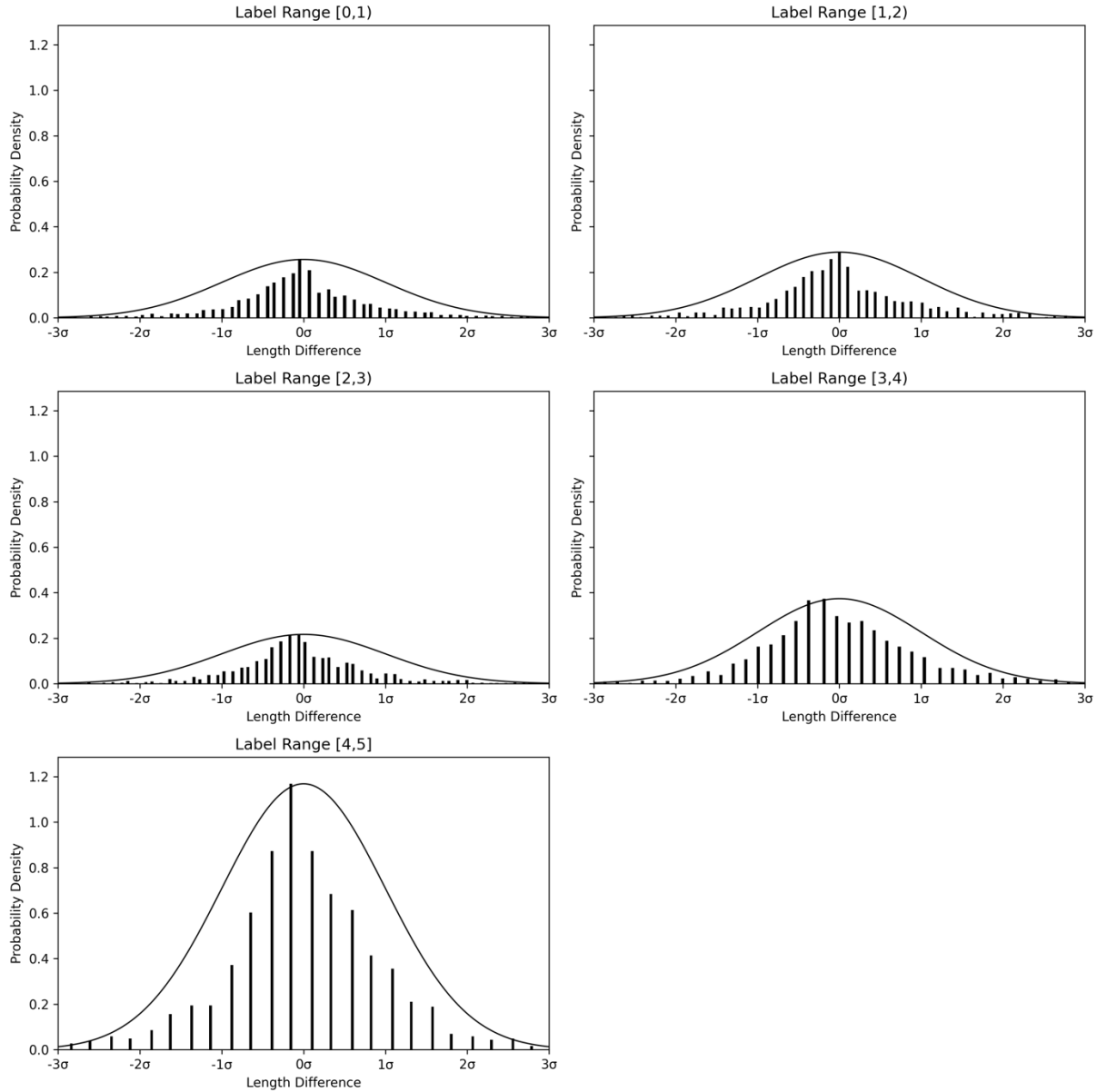
이전 EDA 과정에서 하나의 feature 에만 집중해서 분석을 진행하였고, 특별한 인사이트를 발견하기 어려웠다. 그래서 산점도를 그려 두 개 이상의 feature 를 동시에 분석하여 유의미한 패턴을 발견하려 하였고, 그 결과 새로운 인사이트를 발견할 수 있었다. x 축은 1 번 문장과 2 번 문장의 길이의 차이를 나타내고, y 축은 두 문장의 유사도를 나타낸다. 높은 점수일수록 데이터의 표본이 부족하다는 것을 알 수 있고, 특히 주목할 점은 문장 쌍의 길이 차이가 모든 레이블에서 정규분포를 따른다는 것을 확인할 수 있다.



정확한 확인을 위해서 y 축을 확률밀도로 나타내었고, 문장 쌍의 길이 차이에 대한 분포가 정규분포를 따름을 더욱 명확하게 확인할 수 있다.

두 figure 를 종합적으로 이해해보았을 때, 모든 레이블에서 Length difference 가 정규분포를 따르기 때문에, 데이터 전처리 또는 증강 과정에서 특정 레이블에서 분포가 불균형해진다면 해당 레이블에서는 Length difference 에 대한 편향이 발생할 수 있다고 판단하였다.

NLP 에서 모델이 길이에 의한 편향을 학습할 수 있다는 사실은 여러 연구를 통해 입증되었고, 널리 알려진 사실이다. 예를 들어, Kenton Murray 의 논문[1]에는 NMT 모델의 번역 결과가 문장의 길이에 의존한다는 사실이 언급되었다. 또한, Sarthak Jain 의 연구[2]에서는 Attention 모듈의 성능은 인정하지만, 문장이 의미가 아닌 다른 feature 에서 무언가를 학습한다는 사실을 지적하기도 했다. 이러한 점에서 미루어 볼 때, 만약 위와 같이 모든 레이블 범위에서, length difference 가 같은 분포를 따르지 않고, 특히 정규분포를 따르지 않게 하는 이상치가 다수 존재하게 된다면 학습 과정에서 이에 대한 편향이 발생할 것임을 예측할 수 있다. 따라서 어떤 증강 또는 전처리에 의한 성능 저하가 발생하였다면, 이러한 편향이 발생했을 가능성을 염두에 두어야 한다는 결론을 내렸다.



4.2.3 가설

train 데이터 셋은 4.2.1 의 라벨 분포 EDA 에 근거하여 데이터의 Target column 인 Label 이 0 인 값이 상대적으로 매우 많고(전체 데이터의 약 22.72%) 5 의 값이 부족한(전체 데이터의 약 0.98%) 불균형한 분포를 갖고있다. 따라서 Label 이 imbalance 함으로 모델 학습이 자주 등장하는 라벨에 대해 편향되는 부정적인 영향을 미칠 수 있다고 판단했다. 따라서 이러한 문제를 해결하기 위해 데이터 증강을 이용하여 라벨의 분포를 보다 균등하게 맞추게 되면 모델의 예측 성능이 향상될 것이라고 가설을 세웠다. 다만, Label 별로 문장 길이의 차이는 균등 분포는 아니지만 가우시안 분포를 따르고 있으므로, 학습 시 sentence1 과 sentence2 의 문장 차이에 대한 편향은 발생하지 않을 것으로 판단되었다. 따라서 Label 분포를 고르게 하기 위한 데이터 증강을 할 때에 길이에 대한 편향이 생기지 않도록 데이터 증강을 진행하기로 계획했다.

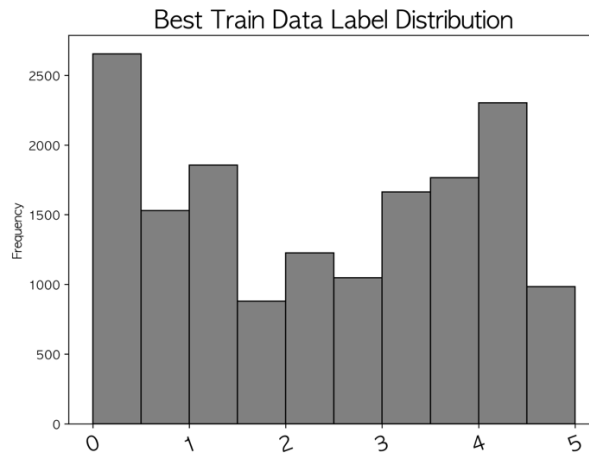
4.3. EDA 를 기반으로 한 데이터 증강

4.3.1 분포를 고려한 단순 복제 증강

라벨 분포를 균등 분포로 맞추기 위해 라벨 값을 1 단위로 나누어, 0 부터 1 사이의 값의 데이터는 oversampling 하지 않고, 나머지 구간의 데이터의 개수를 0 부터 1 사이의 데이터 개수와 같아지도록 oversampling 하였다. oversampling 기법으로는 문장의 의미가 훼손되지 않게 하기 위해서 단순히 데이터를 복사하여 추가하는 방식을 사용했다. 실험 결과 기존 raw dataset 보다 pearson 점수가 상승하여 유의미한 증강 기법임을 확인했다.

4.3.2 Down sampling+swap sentence+copied sentence+uniform distribution 데이터 증강

여러 실험을 통해 불용어를 문장 중간에 삽입하거나 인접한 단어의 어순을 바꾸어 문장을 변형하면 의미 변화가 생기기 때문에 문장의 의미가 변형되고 데이터의 정보가 변질되어 학습에 부정적인 영향을 끼치게 되었다는 것을 확인하였다. 따라서 최대한 문장의 구조를 수정하지 않고 데이터를 증강하여 라벨 균등 분포를 맞추기 위해 아래와 같은 방법들을 사용하였다.



a. under sampling : Label 값이 0 점이 데이터가 지나치게 많아, 단순 복제 방식으로 oversampling 하는 것이 아닌, 문장을 수정하지 않고 새로운 데이터를 추가하여 1 점대 이상의 데이터를 증강하는 방식으로는 개수를 균등하게 맞춰주기 어렵다고 판단하여, 2119 개의 0 점 데이터 중 1000 개만 추출하여 under sampling 을 진행하였다. 문장의 길이 변화가 모델의 학습에 미치는 영향을 줄이기 위해 0 점 데이터의 총 문장 길이 분포와 추출한 데이터의 총 문장 길이 분포가 유사하도록 1000 개의 데이터를 추출하였다. 이후, 사용되지 않은 0 점 데이터의 일부는 copied sentence 증강 기법에 사용되었다.

b. swap sentence : STS task 에 사용되는 BERT 계열의 모델의 경우, 데이터 셋 내의 sentence_1 과 sentence_2 의 순서를 바꾸게 되면 유의미한 차이를 만들 수 있을 것이라고 가설을 세웠다. 이 가설에 대한 근거로는 문장의 위치가 바뀌면서 segment embedding 과 positional embedding 의 값이 달라져, 새로운 데이터를 학습하는 효과를 낼 수 있을 것이라고 판단했기 때문이다. 마찬가지로, 해당 증강 기법도 라벨의 균등 분포를 위해 증강하는 것이므로 0 점대의 데이터를 제외한 나머지 데이터에 적용하여 데이터의 양을 2 배로 늘려주었다.

c. copied sentence : 논문[3]에서 영감을 받아 응용한 기법으로, 다른 데이터에 비해 양이 많이 부족한 Label 값이 5 점인 데이터를 증강하기 위한 방법으로 under sampling 시에 선택되지 않은 데이터 중 약 500 개의 데이터를 가져와 같은 문장을 sentence_1 과 sentence_2 에 넣고 Label 값을 5 로 주어 새로운 데이터를 생성하였다. 이 때 선택된 500 개의 데이터는 전체 데이터의 분포를 따르도록 sampling 하였다. 위와 같은 방식을 사용하면 문장의 의미 훼손을 최소화하고, 문장의 의미가 거의 일치하는 Label 5 점에 해당하는 데이터의 패턴을 모델이 학습하기 용이해질 것이라고 판단하였다.

4.3.3 Easy Data Augmentation Synonym Replacement(품사 선택 교체)+swap sentence+copied sentence

논문[4]을 참고하여 Easy Data Augmentation(EDA)기법인 Synonym Replacement(SR)을 적용했다. KoEDA 라이브러리를 사용하여 문장에서 임의의 단어를 선택하여 동의어로 교체했다. 하지만 KoEDA 라이브러리는 문장의 단어를 임의로 변경했기 때문에 생성된 두 문장의 의미가 원본의 두 문장과 달라졌고, 원본 데이터의 문장 유사도 점수와 생성된 데이터의 문장 유사도가 달라지게 되어 성능향상을 얻을 수 없었다. 따라서 직접 EDA 의 SR 데이터 증강 라이브러리를 구현하였고 방법은 아래와 같다.

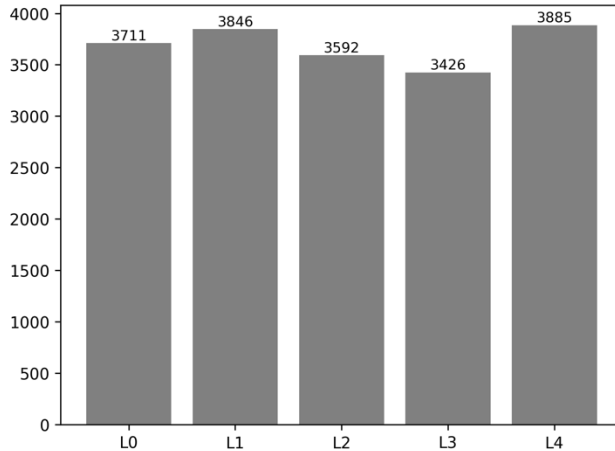
-a. 문제점 해결을 위한 가설 설정: 문장의 위치를 임의로 변경하는 KoEDA 라이브러리와 다르게 두 문장에서 모두 같은 단어가 존재할 때만 동의어로 교체하여 데이터를 증강하면 원본 데이터의 문장 유사도가 증강된 데이터에서도 보존된다는 가설을 세운다. 이 가설을 활용하여 구현할 때 아래의 원칙을 지켜 증강하도록 구현했다.

첫번째로 비교되는 두 문장의 같은 단어를 같은 단어로 교체한다. **두번째**로 원본 문장의 문장 구조를 최대한 보존한다. 문장에서 임의로 단어를 선택하여 임의로 교체하거나 문장 구조가 바뀐다면 비교되는 두 문장의 문장 유사도 점수가 변동될 수 있다. 따라서 두 문장에 모두 존재하는 단어를 동일하게 교체하면 원본 데이터의 문장 유사도를 유지하여 데이터를 증강할 수 있을 것이다.

예시	sentence1	sentence2	Label
원본 문장	아버지가 방에서 주무신다.	아버지가 방에서 주무십니다.	5.0
변경 문장	아버지가 안방에서 주무신다.	아버지가 안방에서 주무십니다.	5.0

-b. 구현

1. SR 데이터 제작: 원본 문장은 오타와 비문이 많기 때문에 맞춤법 검사로 정제한 데이터를 사용하고, 두 비교 문장에서 형태소 분석을 통해 같은 단어이며 품사가 명사인 단어를 교체 후보로 정하고, 동의어 Wordnet 에 있으면 두 문장 모두 명사를 교체했다. Wordnet 은 KoEDA 내부에서 찾았으며 KAIST 에서 제작한 동의어 사전[5]을 활용했고, 교체할 동의어 명사 끝음절의 받침 여부에 따라서 문장의 조사를 교체했다. 총 5188 Dataset 을 생성했다.



2. Label Uniform Distribution: 원본 데이터를 모두 사용하며 Label Distribution 을 균등하게 맞추기 위해서 추가적인 증강을 했다. 원본 데이터에 Label 이 0~1 사이인 Dataset 은 39.8%로 차지하는 비율이 높다. 따라서 생성한 L0 Dataset 은 사용하지 않았다. 전체 중 비율이 낮은 Label 1~3 사이인 데이터와 Label 이 5.0 인 데이터 셋을 증강했다. Swap Sentence 기법을 사용하여 Label 이 0~1 사이인 값을 증강했고, Label 이 5.0 인 dataset 은 Copied Sentence 기법을 사용하여 Label 이 0 인 데이터 1000 개를 이용해 증강했다. 최종적으로 18460 개의 기존의 2 배 크기의 데이터셋을 구축했으며, Label 구간 크기가 1 인 분포는 약 20% 수준으로 이전 Label Distribution 을 적용한 best.csv 데이터보다 균등한 분포의 dataset 을 만들었다.

snunlp/KR-ELECTRA-discriminator 모델에서 Hyper-parameter Tuning 적용 후 Evaluation Pearson Correlation 0.9363, Public 0.9265, Private 0.9354 으로 팀의 단일 모델 중 최고 점수를 받았다. 이전 Label Distribution 을 적용한 best.csv 를 snunlp/KR-ELECTRA-discriminator 모델에서 학습했을 때 Public 0.9216, Private 0.9280 으로 성능이 향상됨을 확인했다.

4.4. 모델 탐색 및 선정

4.4.1 모델 Research



Hugging Face 의 pre-trained 모델을 선택하여 transfer learning 을 진행했고, competition 의 STS 의 Task 를 올바르게 수행할 수 있을까라는 관점에서 첫번째 기준을 만들어 모델을 일차적으로 선별했다. Sentence Similarity 기반의 모델이 아닌 Text Classification 모델을 이용했는데. 이유는 다음과 같다. Hugging Face 의 Sentence Similarity Task 는 왼쪽의 그림[6]처럼 여러 문장끼리의 유사성을 판단하여 그사이의 거리를 재는 것이라고 파악했고 이는 주로 검색 및 클러스터링, 문서 분류 작업에 유용할 거라고 판단할 수 있었다. 따라서 두 개의 문장의 유사도를 구하는 Regression 에는 적합하지 않을 것이라고 추측했으며, 예상과 일치하게 해당 모델들은 대회 Task 에 높은 성능을 보이지 않는 것을 확인했다. 따라서 Text Classification 에 Label 의 개수가 1 인 모델들을 선택하여 Regression Task 에 활용했다.

1 차 기준에 따라 총 5 가지 모델(monologg/koelectra-base-v3-discriminator, lighthouse/mdeberta-v3-base-kor-further, monologg/koelectra-base-finetuned-nsmc, monologg /kobigbird-bert-base, kykim/electra-kor-base)을 이용해 튜닝을 진행했는데, 위 모델들 중 Public Score 가 0.921 이상인 세가지 모델(kykim/Electra-kor-base, snunlp/KR-ELECTRA-discriminator, XML-Roberta-large)을 앙상블 모델의 후보군으로 선정했다.

그리고 해당되는 모델들은 다음과 같은 측면에서 앙상블에 적합하다고 생각하여 모든 모델을 앙상블에 활용했다.

- a. **kykim/electra-kor-base** : 데이터의 경우 단어의 하위집합에 대해서 토크나이징이 수행된다. 이는 단어 수준에 대해서 문장을 이해하여 결과값을 도출하게 된다.
- b. **snunlp/KR-ELECTRA-discriminator**: snunlp 의 ELECTRA 모델은 34GB 의 한국어 데이터 세트와 30000 개의 형태소 기반 단위 토큰을 사용하였기에 같은 ELECTRA 모델이지만 서로 다른 부분을 특징을 파악할 것이라고 생각되어 선정하였다.
- c. **xlm-roberta-large**: XML 의 모델은 100 개의 언어와 2.5TB 라는 데이터 세트와 훈련된 모델이다. 비록 이 데이터 세트에 한국어가 적더라도 전체적인 언어의 문맥 이해는 ELECTRA 모델보다 높을 것이며 Transformers 모델 특성상 좋은 품질의 많은 양의 데이터라면 성능은 지속해서 올라감으로 한국어가 데이터 셋이 조금이어도 가장 큰 데이터 세트를 사용한 모델을 선정하였다.

4.4.2 모델 앙상블

앙상블은 두가지 방식을 이용하였다. 일반 평균과 가중 평균을 이용한 방식인데, 일반평균의 방식으로 각각의 3 개 모델의 결과값의 평균을 구하여 특정 모델에 편향되지 않도록 모델 앙상블을 진행했다. 가중평균은 모델별로 가중치를 주어 평균을 내보았다. 가장 큰 점수의 모델에 비교적 높은 가중치를 주고 최하 점수의 모델에 낮은 가중치를 두어 가장 높은 모델의 큰 영향을 줄 수 있도록 조절하였다. 최종적으로는 일반 평균을 통해 만든 앙상블 모델을 선택하여 제출하였고 pearson correlation 이 public 0.9367, private 0.9403 점수를 얻을 수 있었다.

5. 자체 평가 의견

5.1. 잘한 점

- 논문과 잘 알려진 기법을 찾아보고 실제로 적용해보았다. 근거를 가지고 실험을 하고 성능이 향상된 이유와 실패한 이유를 찾으며 팀원 모두 능동적으로 모델을 개선했다.
- Github issue 와 Notion 을 활용하여 각자의 실험 내용과 결과를 공유하며, 불필요한 자원이나 시간의 낭비를 막고, 효율적으로 실험을 진행하였다.

- 데이터 전처리와 증강에 대한 다양한 기법들을 브레인스토밍을 통해 공유하고, 최대한 많은 실험을 통하여 각 기법이 왜 효과적이지, 효과적이지 못한지를 경험을 통해 이해하고, 각자 공부한 이론적 근거를 나누며 태스크에 대한 이해도를 높였다.

5.2 실험에서 실패했으나 유의미한 실험

a. BERT Masking: BERT의 Masking 기법을 채택함으로써 마치의 CV의 Cutout 처럼 동작하는 가설을 세웠다. 입력의 2 문장에 대해서 각각 랜덤한 Masking을 적용하여 Overfilling을 방지하는 방식으로 적용하였었다. 하지만, CV 데이터와 다르게 NLP는 데이터가 조금만 다르더라도 전체적인 context 의미가 변질하며 또한 모델이 Maskfill을 학습하는 여지를 줌으로써 실패했다.

b. Label Noise: epoch마다 데이터 세트의 label 값에 소수점 단위의 노이즈를 주어 전체적인 Overfilling을 방지하였다. 하지만, Label의 Noise를 적당히 주는 파라미터는 찾기가 어려웠다. 데이터의 분포가 일치하지 않을 경우는 잘 동작하였지만 분포를 조정하자 Noise로 인해 데이터 세트의 분포가 불안정해져 학습에 악영향을 주었다.

b. Back Translation: 시도해 본 다양한 증강 기법들 중에 가장 큰 성능 향상을 기대했던 것은 역번역(Back-translation)이었다. 하지만 단순히 역번역을 통한 데이터 증강으로는 유의미한 효과를 볼 수 없었다. 번역 이전에 불필요한 특수 문자 제거, 띄어쓰기, 맞춤법 교정 등의 전처리가 선행되지 않았기 때문임을 일차적인 원인이라고 해석하였다. 데이터 대부분이 영화 리뷰, 슬랙 채팅 등 구조화되지 않은 구어체의 텍스트를 기반으로 하기 때문에 번역 과정에서 정보 손실이 크게 일어나는 경우도 상당 수 존재했다. 또한, 레이블 불균형 문제를 해결하지 않은 채로 증강을 적용하였기 때문에, 성능 향상을 기대하기 어렵다는 결론에 도달했다. 증강한 데이터를 조금 더 효과적으로 사용하기 위해서는 각 레이블에 대한 분포를 uniform distribution을 따르게 하고, 데이터를 사전에 정제하여 역번역 과정에서 정보 손실을 최소화할 수 있는 샘플을 대상으로 증강 기법을 적용해야 할 것이라고 생각하였다.

5.3 프로젝트 협업 프로세스에서 아쉬웠던 점

- 전반적으로 대부분의 팀원이 처음 경험하는 딥러닝 대회이고, 대회 성적에 집중한 나머지 베이스라인 코드에 대한 이해도가 조금 부족했고, 코드 리팩토링 과정에서 시간을 투자하지 못하였다.

- Huggingface trainer를 사용하여 하이퍼파라미터나 모델을 튜닝하였기 때문에 조금 더 low-level에서 모델을 수정하는 접근법은 구상만 하고 실제로 실험하지 못한 방법들이 있다.

5.4 프로젝트를 통해 배운점 또는 시사점

- 사전 학습된 모델을 우리에게 맞는 task로 전이 학습하여 예측 값을 얻는 방법을 몸소 익힐 수 있었다.

- wandb를 이용해 하이퍼 파라미터 튜닝을 하며 다양한 사용 방법들을 익힐 수 있었다.

- 딥러닝 모델을 개발하는 대회가 어떻게 진행되는지 전반적인 프로세스를 이해할 수 있었고, 이러한 문제를 협업을 통해 해결해 나가는 경험을 얻을 수 있었다.

- 널리 알려진 데이터 전처리 및 증강 기법의 효과는 태스크와 데이터에 의존적이며, 상황에 따라 어떤 기법을 사용할지에 대한 인사이트를 터득할 수 있었다.

6. 참고 문헌(References)

[1] Kenton Murray and David Chiang, Correcting Length Bias in Neural Machine Translation, 2018

[2] Sarthak Jain, Byron C. Wallace, Attention is not Explanation, 2019

[3] Park, Chanjun, Kim, Kuekyeng, Lim, Heuseok, Optimization of Data Augmentation Techniques in Neural Machine Translation, 2019, Annual Conference on Human and Language Technology(p258-261)

[4] Jason Wei and Kai Zou, EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks, 2019, EMNLP-IJCNLP(p 6382-6388)

[5] KAIST Wordnet 동의어 사전, <https://github.com/catSirup/KorEDA>

[6] Research Gate, https://www.researchgate.net/figure/Visualization-of-clusters-obtained-via-HDBSCAN-on-sentence-embeddings-Each-cluster_fig2_338683513

NAVER CONNECT 부스트캠프 AI Tech

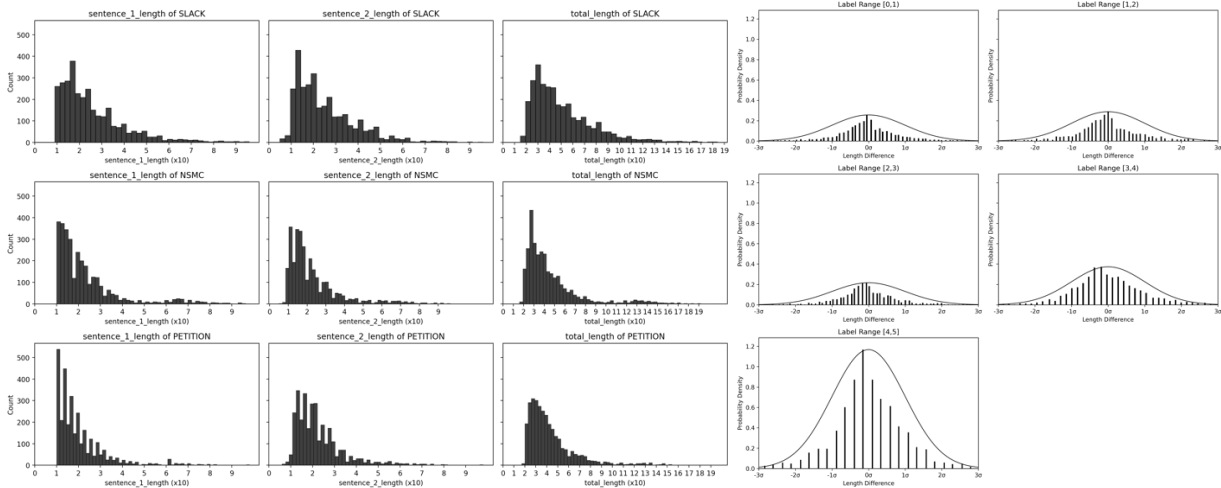
NLP 기초 프로젝트

개인 회고

Member

김주원_T5056, 강민재_T5005, 김태민_T5067, 신혁준_T5119 윤상원_T5131

EDA



Source 별 문장 길이의 분포가 모두 유사함을 확인하여, source 에 의한 편향의 발생 확률이 낮을 것이라고 판단하였다. Label 에 sample 의 문장 간의 길이 차이가 정규 분포를 따른다는 점을 확인하여, 데이터 전처리 및 증강 결과가 이 분포에서 크게 벗어나게 되면, length difference 에 의한 편향이 발생할 것임을 유추할 수 있다. 특히 특정 레이블에서 그러한 현상이 발생하면, 해당 레이블에 대한 예측에 취약해질 것이라고 판단하였다.

Data Preprocessing

0.2 단위로 되어있는 레이블을 0.5 단위로 조정하거나, 1 단위로 정수화하였으나 성능 향상은 미미했다. 그리고 앙상블 과정에서는 이러한 전처리가 오히려 정보 손실로 이어질 수 있다고 판단하여 사용하지 않았다.

Data Augmentation

- Sentence Swap
- Remove or insert Random Character
- Remove Random Space
- Back translation

일반적으로 제시되는 Easy Data Augmentation 기법인 임의 단어 삽입, 제거, 순서 교체, 동의어 교체는 한국어 문장을 대상으로 수행하기는 어렵다. 정교한 전처리가 선행되지 않으면 오히려 문장의 의미가 크게 훼손될 것이라고 판단하였고, 적절한 증강 기법을 탐색하던 중 역번역을 고려하게 되었다. 파파고 API 를 활용하여 문장을 영어로 번역한 후 다시 한국어로 번역하였고, 성능을 평가해보았다. 안타깝게도 모델 성능이 개선되지는 않았는데, 그 원인을 분석해보면 다음과 같다. 그 중 처음 두 가지 원인은 앞서 언급한 다른 증강 기법에도 적용되는 문제점이다.

1. 레이블 불균형 문제를 해결하지 않은 채로 데이터를 증강하였다. 레이블이 uniform distribution 을 이루지 않으면 그로 인한 bias 가 학습되므로 이 점을 미리 해결하였어야 했다고 판단하였다.
2. 레이블을 조정하지 않았다. 임의 글자 반복, 삭제와 더불어 번역 과정에서 정보 손실을 고려하여 레이블을 정교하게 조정해 주어야 하지만, 시간적인 한계로 일관된 기준을 세우기가 어려웠다. 따라서 단순히 원본 샘플의 레이블을 그대로 사용하였다. 이 과정에서 문장의 의미는 변형되었는데, 레이블은 바뀌지 않았기 때문에 모델의 학습에 방해가 되었을 것이라고 판단하였다.

3. 데이터 전처리가 이루어지지 않은 채로 번역을 수행하였다. 문장 자체가 문법적으로 옳은 표현과 사전에 등재된 단어만을 사용한 것이 아니기 때문에 기계번역 과정에서 문장의 의미가 훼손될 수밖에 없다. 예를 들어, "비양심적군복무자입니다."라는 문장은 띄어쓰기가 제대로 되지 않은 문장이며, "전 암만 짝어도 까만하늘.. ππ"라는 문장의 이모티콘은 "울면서"라고 번역이 되었다. 따라서 역번역 데이터가 효과적으로 학습이 되려면 사전에 번역의 질을 높일 수 있는 전처리를 적용해야 할 것이라고 결론지었다.

Ensemble

단순 평균을 계산하는 것 외에 다른 모델 앙상블 기법을 시도하였는데, 방법은 다음과 같다.

앙상블에 사용한 모델의 각 test sample 에 대한 예측값의 평균과 표준편차를 구하였다. 각 샘플에 대한 예측값의 평균에서 1.04σ 범위(70%) 밖에 있는 예측값은 이상치로 판단하여 앙상블 대상에서 제외하였다. 위 범위 내에 있는 예측값을 대상으로 평균을 계산하였다.

이 방법을 사용한 이유는, 예측값 중 이상치로 판단될 수 있는 값에 의해 평균의 함정에 빠지지 않기 위해서였다. 그러나, 특별한 성능 향상을 거두지는 못하였다. 조금 더 정교한 범위 조정과 산술 평균 이외 다른 평균 혹은 대푯값을 계산하는 방법을 사용하면 앙상블 성능이 나아질 거라고 기대된다. 제출 횟수의 한계로 시도해보지 못한 일부 실험을 다음에 시도해봐야겠다고 생각했다.

피드백

팀원 대부분이 AI 관련 프로젝트가 처음이다 보니, 특별히 역할을 분담하지 않고 전 과정에 걸쳐 각자 실험을 기획하고 진행하였다. 그러다 보니, github 와 notion 을 사용하여 필요에 따라 협업하고 프로세스를 공유하였음에도 불구하고, 서로 진행한 실험 내용을 명확히 파악하지 못할 때가 있었다. 다음 프로젝트에서는 조금 번거롭더라도 구체적으로 실험 과정을 정리하고 공유할 수 있도록 신경 써야겠다는 생각이 들었다. 어느 정도 모델 구현에 익숙해지면, 분업을 통해서 효율적으로 프로젝트 과정이 흘러갈 수 있도록 현업과 비슷한 실험 환경을 구성해보고 싶다는 생각도 들었다.

연구 과정에서는 제출 횟수의 제한으로 인해, 뚜렷한 성능 향상이 기대될 것이라고 판단되지는 않는 다양한 실험을 진행하지 못했다. 예를 들어, 모델이 내놓은 예측값을 사용하여 데이터를 증강하는 방법, 다양한 앙상블 기법들을 실험해보고 싶었으나, 제출 횟수 이외에도 시간적 한계 때문에 모든 의문을 풀지는 못했다. 다만 이런 문제점도, 프로젝트 관리를 체계화하면 해결될 문제라고 생각한다. 일례로, 뒤늦게 레이블 불균형을 깨달아서 일부 증강 기법을 재시도한 적이 몇 차례 있는데, 실험 과정을 사전에 잘 정리하면 이런 시간 낭비를 없앨 수 있을 것이다.

김주원_T5056 캠퍼

팀내에서 나의 역할

이번 NLP 기초 프로젝트에서는 전체 팀장을 맡아서 팀의 협업 관리하는 Project Manager 의 역할과 모델(monologg/kobigbird-bert-base)의 하이퍼 파라미터를 튜닝하고 데이터를 분석(EDA)하는 AI 엔지니어의 역할을 맡았다. 먼저 Project Manager 로서 팀이 하나의 방향으로 진행하기 위해서 Notion 페이지를 만들어 팀의 Ground Rule 을 공지하고, 팀 페이지를 만들어 아이디어를 공유하고 분업할 수 있는 환경을 조성했다. 이후에서 Github 을 팀원들이 자유롭게 활용할 수 있도록 팀원들의 branch 를 만들고 디렉토리 환경의 초기 설정하고, 익숙하지 못한 팀원들도 잘 사용할 수 있도록 가이드라인을 만들어 배포했다. 이후 AI 엔지니어로서 주어진 모델의 하이퍼파라미터를 wandb 로 튜닝하고, EDA 를 통해 데이터 증강을 하여 모델의 성능을 개선하는 등의 노력을 하였고, 팀이 public 1 등, private 2 등을 할 수 있도록 기여했다.

AI 개발자로 성장을 하기위해 시도했던 모험들(어떤 것을 도전했는가? 어떤 것들을 익혔는가?)

AI 개발자로 성장을 하기 위해서는 모델을 다루는 기술과 데이터를 읽는 기술(Data Literacy) 필요하다고 판단했다. 이번 프로젝트를 통해서 모델의 하이퍼 파라미터를 튜닝하는 법을 확실하게 파악하고, 데이터를 분석하고 분석한 기법을 적용하여 성능 향상 혹은 성능 향상이 이루어지지 않더라도 왜 해당 기법이 실패했는지 논리적으로 추론할 수 있는 힘을 기르고자 했다. 따라서 먼저 모델의 하이퍼 파라미터 튜닝을 올바르게 하기 위해 과거에 학습했던 Andrew Ng 의 Hyperparameter Tuning 방법을 정리하고, 적용함으로써 더 나은 방법을 발견하고자 했다. 강의에 나왔던 내용은 먼저 Train data 에 최대한 Fitting 할 수 있도록 학습하고, 이다음에 과적합을 줄이는 방식으로 실험을 해야 한다고 했다. 처음에는 wandb 로 무작정 하이퍼 파라미터로 튜닝하여 학습하였지만, 다시 학습 내용을 정리한 뒤 train data 를 더 잘 학습하게 하는 하이퍼 파라미터를 먼저 튜닝하고 고정된 다음 overfitting 을 감소시킬 수 있는 하이퍼 파라미터를 튜닝하는 방식을 활용했고, 모델을 올바르게 튜닝하여 튜닝 전보다 훨씬 더 좋은 모델의 성능을 확인할 수 있었다. (피어슨 계수 0.05 증가) 추가로 데이터에 Label 의 분포에 대한 분석을 진행했다. Label 에 대한 데이터를 분석하며 해당 데이터 Label 이 0 의 데이터가 많고, 5 데이터가 적은 imbalance 한 부분을 확인하고, 팀원들에게 발표를 진행했다. 이를 팀원들이 본인의 아이디어에 적용하여 데이터 증강을 진행했고, 최종적으로 분포를 고려한 여러 가지 데이터 셋에서 큰 성능 향상이 이루어져 팀이 올바른 방향으로 진행될 수 있도록 도움을 줬다는 점에서 큰 보람을 느낄 수 있었다. 이렇게 모델의 하이퍼 파라미터 튜닝 방법, 데이터 분석하는 방법을 실습해보며 AI 개발자로 한 걸음 나아갈 수 있었던 의미 있는 시간이었다.

함께 일하기 위한 동료가 되기 위해 기울였던 노력들


대부분 팀원들이 처음으로 참여하는 NLP 도메인 대회이므로 이번에 우리 팀에서 프로젝트 목표로 삼은 부분은 모든 팀원들이 각자 데이터 전처리부터 모델의 하이퍼 파라미터 튜닝에 이르기까지 End-to-End 로 경험하기였다. 하지만, 본인의 아이디어가 충분히 논리적인 근거를 갖추었음에도 자신감이 없어서 구현에 망설이거나 결과가 나오지 않아 힘들어하는 모습의 팀원도 있었다. 팀장으로서 팀원이 끝까지 아이디어를 구현할 수 있도록 독려했고 결국 해당 캠퍼가 아이디어를 무사히 구현하여 결국 모델의 성능향상이 이루어지는 부분을 보았다. 또한, 팀이 원활하게 아이디어가 나오고 대화하는 분위기가 진행되도록 나의 입장에서 모델을 평가하는 것이 아니라 아이디어를 제시한 사람의 입장에서 최대한 생각하고 해당 아이디어를 더 잘 구현할 수 있도록 아이디어를 제시해보았다. 이렇게 서로의 아이디어가 구현되어 하나의 결과물을 만들고, 또한 아이디어와 아이디어가 쌓여 새로운 아이디어가 탄생하는 것을 보면서 협업을 통해 팀이 성장하고 좋은 결과를 내는 모습을 볼 수 있었다.

부딪힌 한계, 개선해야할 부분

우선 팀장으로 팀 매니징을 함께 하다 보니 협업 관리를 제대로 하지 못해 아쉬움이 남는다. 초기 설정이나 그라운드 룰 설정은 잘해 두었지만, 해당 부분을 지속해서 체크하지 못해 대회의 끝으로 갈수록 Github 의 코드 버전 관리가 제대로 안 되어있는 부분이 있었다. 해당 부분은 대회에 모델링이나 데이터 분석에 매달려 밤을 새우는 등 일상생활의 루틴을 지키지 못해 프로젝트를 진행될 수록 에너지를 잃고, 점검해야 할 시간에 점검하지 못한 부분이 컸다. 따라서 페이스를 잘 조절하여 체력적으로 준비된 상태에서 프로젝트를 진행할 수 있도록 매일 매일 점검해야겠다.

그리고 두 번째로는 팀원들의 입장에서 생각하려고 했지만, 내 관점에서 생각해서 이해를 못 한 부분이 있었다. Back Translation 관련 이야기였는데 결론적으로 서로가 비슷한 이야기였지만, 내가 이해하지 못해 이야기가 빙글빙글 돌았다. 따라서 상대방이

이야기했을 때 먼저 상대방의 의견을 바르게 이해했는지 체크하고 다시 말하는 방식을 습관화해야겠다고 생각했다. 마지막으로 내가 내용을 너무 어렵고 길게 장황하게 말하는 부분이 있는 것 같은데, 조금 더 압축적으로 쉽게 팀원들이 알아들을 수 있도록 이야기해야겠다는 다짐을 하게 됐다.

 김태민_T5067 캠퍼

1. 나는 내 학습 목표를 달성하기 위해 무엇을 어떻게 했는가?

이번 대회에서는 직접 Huggingface 에서 기존의 CV 모델들 말고 NLP 모델들을 사용하고 이를 자유롭게 튜닝하는 방법을 배우고 싶었다. 팀원이 모두 대회는 처음이다 보니 전체적인 Baseline 설계와 모델 리서치를 통해 자유롭게 모델들을 튜닝하고 데이터 쪽보단 직접 각종 딥러닝 테크닉들을 적용했다. 팀원들에게는 Baseline 에서 각각의 실험 모델들을 선정해주며 대회 유경험자로서 대회 전반을 이끌어냈다.

2. 마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

마주친 한계는 기존 CV 데이터의 대해서 전처리를 많이 수행하였지만 NLP 데이터의 전처리는 잘 경험해본 적이 없어 예상대로 어느 순간부터 데이터에 의한 모델의 한계점이 눈에 띄게 늘어났다. 데이터를 전처리하기 위해 각종 방법이 동원되었지만 정작 모델 앙상블과 딥러닝 테크닉을 말씀으로써 데이터의 전처리에서는 팀원들에게 맡기며 크게 다른 것은 없었다. 비록 이번 대회가 NLP 의 모델부터 자유롭게 다루어 보자는 생각이었지만 데이터에 대한 실습이 미약했던 것으로 보인다.

3. 한계/교훈을 바탕으로 다음 프로젝트에서 시도해보고 싶은 점은 무엇인가?

현재로서 다음 프로젝트는 팀원 모두가 기본적으로 모델을 사용할 줄 알고 Huggingface 라는 라이브러리도 익숙해졌으므로 기존 Baseline 인 pytorch lightning 으로 조금 더 low-level 에서의 튜닝과 모델 부분과는 다르게 직접적으로 NLP 데이터를 전처리하고 다루어 보고 싶다. 또한 각종 딥러닝 테크닉들이 CV 를 기반으로 두고 NLP 에 적용했는데 이는 잘 적용되지 않아서 NLP 에서의 테크닉을 조금 더 발전시키고 싶다.

4. 나는 어떤 방식으로 모델을 개선했는가?

모델의 개선 측면은 기본적으로 토큰라이저에 고유 token 을 추가시키는 방법 이외에 모델의 레이어를 직접적으로 수정하지는 않았다. 다만 각 모델의 토큰라이저를 보며 low-level~high-level 에서 나오는 모델의 결과값을 보며 앙상블 모델을 선정하여 높은 성능을 거두었다.

5. 내가 해본 시도 중 어떠한 실패를 경험했는가? 실패의 과정에서 어떠한 교훈을 얻었는가?

- **Directional_label_noise** : Overfitting 을 방지하기 위해 label 의 중간값을 기준으로 낮은 값은 낮게 높은 값은 크게 label 에 노이즈를 주었다. 이는 초기 데이터셋(분포가 맞지 않음)에 대하여 효과가 있었지만 추후 분포가 맞춰지면서 오히려 방해하는 역할을 하였다. 각종 noise 는 기본적으로 데이터셋의 분포를 맞추고 즉 데이터의 전처리가 끝난 후 딥러닝 테크닉이 들어가는 게 맞다는 교훈을 얻어냈다.
- **Source Tagging** : source 별로 각각의 학습이 이루어질 수 있도록 각 데이터에 맞는 source tagging 을 [SEP] 기준으로 붙여 인풋으로 넣어줬다. 하지만 이는 성능에 악영향을 주었으며 아마 source 대로 label 이 편향되는 효과를 주었기 때문이라고 예상한다. 교훈으로 한쪽으로 편향될 수 있는 딥러닝 모델에 대해 다시 유의점을 느낄 수 있었다.
- **Fill Random Token Mask** : 이 또한 단어의 다양성 증가와 Overfitting 을 방지 그리고 CV 의 Cutmix 처럼 동작하기 위해 도입되었다. 각 문장의 단어를 Random 하게 Masking 하고 이 Masking 된 곳에 Fill-Mask 모델을 통하여 유의어로 교체하는 방식을 사용하였다. 하지만 한국어의 특성상 완전히 완전히 다른 의미로 변질되는 경우가 많아 실패하였다. 아마 명사나 동사 부분만 태깅 후 이 부분만 교체하는 방식을 사용하면 나아지지 않을까라는 아쉬움이 존재한다. 교훈으로는 NLP 는 단어를 바꾼다는게 CV 에서의 단순하게 픽셀값을 바꾸는 것과는 의미가 완전히 달라진다는 교훈을 얻어냈다.
- **Combine Custom Loss(Binary Cross Entropy + Focal Loss)** : Autoencoder 에서 recon Loss 는 MSE 가 많이 쓰이지만 BCE 가 성능이 더 좋다는 논문이 있다. 따라서 *On denoising autoencoders trained to minimise binary cross-entropy [1]* 논문에 의거하여 NLP 에 도입하여 기존 MSE 를 BCE 로 교체하며 데이터 분포 불균형을 해결하기 위해 Focal loss 와 결합하여 Custom loss 를 만들었지만 NLP 에서의 BCE loss 의 영향과 label 갯수가 1 이다 보니 Focal Loss 가 어떤 영향을 미치는지 완전히 파악되지 않아 하이퍼 파라미터 조정 실패로 돌아가게 되어 전체적으로 학습이 불안정해졌다. 교훈으로 각 Loss 의 역할을 좀 더 상세히 알고 만약 Custom 으로 제작 시에는 완벽히 두 로스를 이해하고 있어야겠다는 생각이 들었다. 다만 직접 BCE 만 따로 쓰거나 Focal loss 만 따로 쓰는 작업 보다 두 로스를 합쳐 학습이 돌아갔다는 점에서 큰 의미가 있었다고 생각한다.

6. 협업 과정에서 잘된 점/아쉬웠던 점은 어떤 것이 있는가?

팀원들이 대회 경험이 없다 보니 대회를 끌고 나갔는데 모두 잘 따라와 주고 적극적으로 참여하였다. 나를 제외한 모두가 처음임에도 불구하고 각각 지시한 일들과 파트를 모두 열심히 수행하였다. 비록 기법의 결과가 악영향을 미쳐도 상심하지 않고 열심히 다음 실험을 이어 나가는 점에서 팀원 모두가 잘 수행하여 최종 2 등을 기록할 수 있던 것 같다.

아쉬운 점으로는 내 대회 운영 방식이 Bottom-up 방식이지만 NLP 대회는 처음이다 보니 방향성이 중구난방으로 생겨 Bottom-up 방식이 제대로 운영되지 않았다. 결과적으로 대회에서 할 내용을 지시한 내가 많이 아쉽다. 다음 프로젝트부터는 기존의 대회를 진행했던 것처럼 Bottom-up 방식으로 대회를 진행하고 주체적인 역할도 좋지만 팀원으로서의 파트도 맡고 싶다.

참고 문헌 :

[1] Antonia Creswell, Kai Arulkumaran, Anil A. Bharath, *On denoising autoencoders trained to minimise binary cross-entropy*, 2017

신혁준_T5119 캠퍼

모델 개선을 위해 시도한 테크닉

맞춤법 교정 전처리와 데이터 증강

맞춤법 교정 데이터로 증강해 성능을 개선했다. 토큰화에서 맞춤법이 틀린 문장은 잘못된 단위로 토큰을 나눈다. BERT 계열 모델은 Position Embedding 에서 토큰의 순서를 학습한다. 문장에서 쓰인 의미와 다른 토큰이 입력으로 들어가면 학습에 방해된다고 판단했다. 이를 근거로 토큰화가 의미 단위로 잘 이루어지게 맞춤법을 교정했다. 하지만 원본 데이터보다 성능이 낮아졌다. 원인으로는 Pre-Train 모델에서 Input 데이터인 nsmc 데이터를 학습했고 nsmc 데이터를 맞춤법 교정하면서 Pre-Train 모델의 효과가 떨어졌다고 추측했다. 아이디어를 얻어 전처리가 아닌 문장 구조가 유사한 맞춤법 교정 데이터를 증강 데이터로 사용했다. 결과는 원본보다 성능이 올랐다. 유사한 문장 구조의 데이터가 추가되어 Position Embedding 에서 토큰의 순서를 잘 학습했다고 추측했다.

명사, 조사 교체 데이터 증강과 Label Distribution

명사와 조사만 동의어로 교체하여 증강에 성공했다. 팀원이 데이터 증강을 시도했으나 성능 개선에 실패했다. KoEDA 가 임의의 단어로 교체함을 실패 원인으로 개인적인 분석했다. 임의가 아닌 단어를 지정해서 교체하면 효과적인 증강을 이룰 수 있다고 가정하고 이에 맞는 가설을 세웠다. 동의어 사전을 구하기 어려웠는데 KoEDA 내부에 사전이 있을 것으로 판단해 내부에서 Wordnet 파일을 찾았다. 동의어는 용언의 활용, 수식언으로 인한 의미 변화 등을 통제하기 어려워 명사만 교체했다. 앞선 실험에서 Position Embedding 단계에서 유사한 문장구조가 늘어나면 학습이 잘된다는 결론을 얻었고 문장의 구조를 최대한 유지했다. 명사 뒤에 조사가 있을 경우 조사의 성질을 이용해 바꾸었다. 조사는 앞말의 종성에 따라서 달라진다. 명사의 끝음절의 유니코드가 28 로 나누어떨어지면 받침이 없는 점을 이용해 조사를 변경했다.

이전의 Label Distribution 의 Undersampling 을 Oversampling 으로 개선했다. 원본 데이터의 모든 데이터를 사용했다. 명사, 조사 교체 증강 데이터와 Label Distribution 으로 이전 모델보다 성능이 개선됐다.

아쉬운 점은 Ablation Study 이다. 명사, 조사 교체 증강 중 어떤 테크닉을 이용한 데이터가 명확하게 모델의 성능을 높였는지 정확히 알기 어려웠다. 다만 전체 18460 Dataset 에서 약 28%가 명사, 조사 교체 증강 데이터이고, 이전 실험의 점수 증가 폭에 비해 크기에 유의미하다고 판단했다.

피드백

전처리와 문장 증강 실험의 결과와 과정을 토의해 문제점을 찾아내는 역할을 했다. 또한 이를 반영하여 모델이 개선되도록 노력했다. 중요한 점은 모델 성능 개선이 아닌 논리적인 설명과 실패 원인을 분석할 수 있는 능력임을 깨달았다. 논리적인 근거가 있는 실험이 실패했으면 원인을 분석하여 충분히 성능을 높일 수 있음을 이번 명사, 조사 증강 데이터 제작을 통해서 경험했다. 이전 실험에서 성공 요인을 다시 적용하고 실패 요인을 개선하니 빠르게 모델이 개선되었다.

소통으로 팀원에게 믿음을 주어야 한다. 노력을 들인 실험에서 결과를 얻지 못한 경우가 많다. 실험이 논리적으로 타당하지만 실패하면 낙담이 더 크다. 이때 힘을 낼 수 있게 하는 것은 팀원들의 응원과 지지이다. 구현이 길어지고 성능에 대한 확신이 없어질 때 팀원들은 끝까지 응원해주었다. 포기하지 않고 끝까지 실험할 수 있었고, 대회 마지막 날에 높은 성능의 모델을 완성해냈다. 나 또한 팀원들의 실패한 논리적인 실험에서 같이 문제점을 찾아보고 개선안을 제안하며 팀원에게 자신에 대한 믿음을 주려 노력했다.

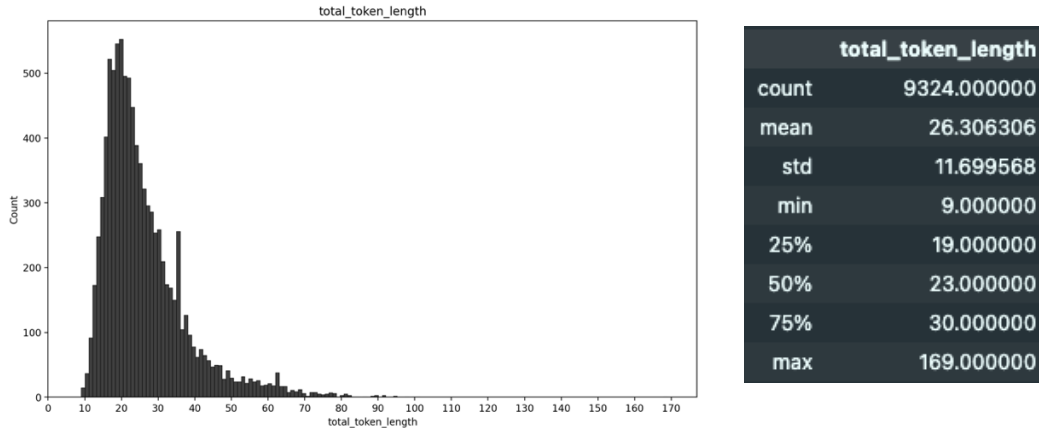
아쉬운 점은 전처리에 몰두한 나머지 모델을 전반적으로 다루고 설계하지 못했다. 첫 대회여서 시간에 쫓겨 모델과 테크닉을 적용한 시간이 적었다. 다음 대회에는 Low Level 에서 전반적인 모델 설계를 해야겠다. 개인적으로는 Ablation Study 가 완벽하지는 않은 듯하다. 하지만 짧은 대회 기간과 GPU 의 제한, 제출 횟수의 제한으로 어려움이 있었으므로 양호한 수준으로 팀원과 결론 내렸다.

윤상원_T5131 캠퍼

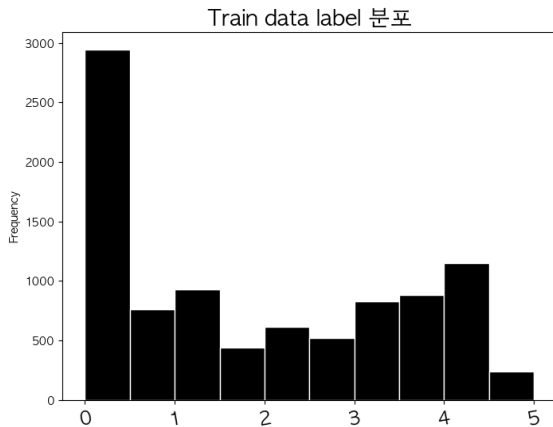
EDA(Exploratory data analysis)

문장 길이 분석

아래는 학습 데이터 셋의 sentence 1 과 sentence 2 를 합쳐 토큰화한 길이의 분포이다. 최대 길이가 169 이므로, 토큰나이징 시 max_length 의 값을 기존 512 에서 256 으로 줄여 padding 토큰의 양을 줄일 수 있었다. 그로 인해 연산량이 2 배가량 빨라졌다.



라벨 분포 분석



- 학습 데이터셋의 라벨 분포이다.

- 0 점부터 0.5 점 사이의 데이터가 지나치게 많고, 4.5 점부터 5 점 사이의 데이터가 적어 데이터의 불균형이 존재한다. 라벨이 불균형한 데이터로 모델을 학습하게 되면 자주 등장하는 라벨에 대해 편향되게 예측하는 등 모델 학습에 부정적인 영향을 미칠 수 있다.

- 따라서 이를 해결하기 위해 우리 팀에서는 데이터를 전처리 및 증강하여 라벨의 분포를 균등하게 맞추어 모델의 성능을 높이려는 다양한 실험을 진행하였다.

Data Preprocessing & Data Augmentation

어순 도치 증강

학습 데이터의 라벨 분포를 균등 분포로 맞춰주기 위해 시도했던 방법이다. 상대적으로 데이터의 수가 부족한 라벨 값이 1 이상 5 이한 데이터의 문장에서 이웃한 단어의 위치를 서로 바꾸어 새로운 데이터를 만들어주는 방식였으나 모델의 예측 성능은 하락하였다. 그 이유는 어순 도치가 문장의 의미를 훼손시켜 학습 과정에서 부정적인 영향을 미쳤다고 판단하여 해당 기법은 사용하지 않았다.

단순 복제 증강

어순 도치 증강이 문장의 의미를 훼손한다는 문제점을 보완하기 위해 사용한 방법이다. 새로운 데이터를 추가하지 않고도 단순히 같은 데이터를 추가하여 라벨의 분포를 균등하게 맞추기만 해도 데이터 불균형이 해소되므로, 모델의 예측 성능이 향상될 것이라는 가설을 기반으로 진행되었다. 라벨 값을 1 단위로 나누어, 1 점 이상의 단위 데이터의 양이 0 점 미만의 데이터의 양과 같아지도록 데이터를 단순 복제하여 증강하였다. 결론적으로 원본 데이터로 학습한 모델보다 해당 데이터로 학습한 모델의 예측 성능이 향상되었으나, 과적합 등의 문제로 해당 기법은 사용되지 않았다.

Under Sampling + Swap Sentence + Copied Sentence

데이터 셋의 문장을 변형하지 않으면서 반복되지 않는 데이터를 추가하여 증강하기 위해 아래와 같은 방법들을 사용하여 라벨의 분포를 균등하게 맞춰주었다.

- **Under Sampling**

학습 데이터의 1/3 가량이 라벨 값이 0 점인 데이터이므로, 나머지 구간의 데이터를 문장 변형 없이 증강하는 방식으로는 개수를 맞춰주기 어렵다고 판단하였다. 따라서 0 점 데이터 중 1000 개만 추출하여 학습에 사용하고 나머지 사용되지 않는 데이터의 일부는 copied sentence 에서 활용했다. 문장 길이 분포의 변화가 모델의 학습에 미치는 영향을 줄이기 위해 전체 0 점 데이터 문장 길이 분포와 추출한 데이터의 문장 길이 분포가 유사하도록 추출하였다.

- **Swap Sentence**

STS task 에 사용되는 BERT 계열의 모델의 경우, 데이터 셋 내의 sentence_1 과 sentence_2 의 순서를 바꾸게 되면 유의미한 차이를 만들 수 있다고 가설을 세웠다. 이 가설에 대한 근거는 문장의 위치가 바뀌면서 segment embedding 과 positional embedding 의 값이 달라져, 새로운 데이터를 학습하는 효과를 낼 수 있을 것이라고 생각했다.

- **Copied Sentence**

논문[1]에서 영감을 받아 응용한 기법으로, 5 점짜리 데이터를 증강하기 위해 고안해 낸 방법이다. 이 기법을 사용하면 원본 학습 데이터의 문장을 그대로 사용함과 동시에 문장의 의미가 거의 일치하는 5 점 데이터의 패턴을 모델이 학습하기 쉬워질 것이라고 가설을 세웠다. under sampling 기법에서 버려진 데이터의 일부를 가져와 같은 문장을 sentence 1 과 sentence 2 에 넣고 라벨 값을 5 점으로 주어서 새로운 데이터 500 개를 생성하였다.

해당 데이터로 모델을 학습시킨 결과 우리 팀에서 사용했던 모든 모델에서 예측 성능이 향상되어, 유의미한 증강 기법임을 확인할 수 있었다.

보완할 점

코드 관리 측면

딥러닝에서 처음 경험하는 대회였다 보니 코드 모듈화와 같은 리팩토링에 전혀 신경쓰지 못한 부분이 아쉬웠다. 이로 인해 실험을 진행할 때에도 코드의 어느 부분을 수정해야될지 종종 헷갈렸다. 다음 대회에서는 효율적인 실험 진행과 가독성을 위해 코드 리팩토링을 진행해야겠다.

실험 진행 측면

이번 대회에서는 하이퍼 파라미터 튜닝, 전처리, 증강 등의 실험에서 모두 seed 를 42 만 사용하여 실험했다. 이렇게 실험할 경우 실제로는 성능이 향상됨에도 seed 가 잘못 걸려 성능이 향상되는것을 관찰할 수 없을 수도 있다. 따라서 다음 대회에서는 같은 실험에 대해 여러 seed 를 사용해 평균낸 값으로 성능이 향상되었는지 확인해 볼 것이다.

모델 측면

이번 대회에서는 사전 학습된 모델의 구조를 크게 수정할 기회가 없었다. 비록 성능이 떨어질지 몰라도, 이번 대회에서 시도했던 것보다 더 low level 에서 모델의 구조를 원하는 방식으로 변형해봐야겠다.

Reference :

[1] Park, Chanjun, Kim, Kuekyeng, Lim, Heuseok, Optimization of Data Augmentation Techniques in Neural Machine Translation, 2019, Annual Conference on Human and Language Technology(p258-261)