접수번호: AI0149

「통계데이터 인공지능 활용대회」코드설명 보고서

Ⅰ. 구동환경 및 정보

1. 사용언어 : Python
2. 클라우드 : Google Cloud Platform (GCP) - Debian Linux OS (Virtual Machine)
    a. GPU : NVIDIA Tesla A100 1개 & CPU : Intel cascade lake 1개
3. 학습 및 테스트에 소요되는 시간 :
    a. 10 Fold Training : 1 Fold (1시간) * 10 = 총 10시간 학습
    b. Test Data Inference : 1 checkpoint (20초) * 10 = 총 3분 20초

Ⅱ. 코드 구성 방법

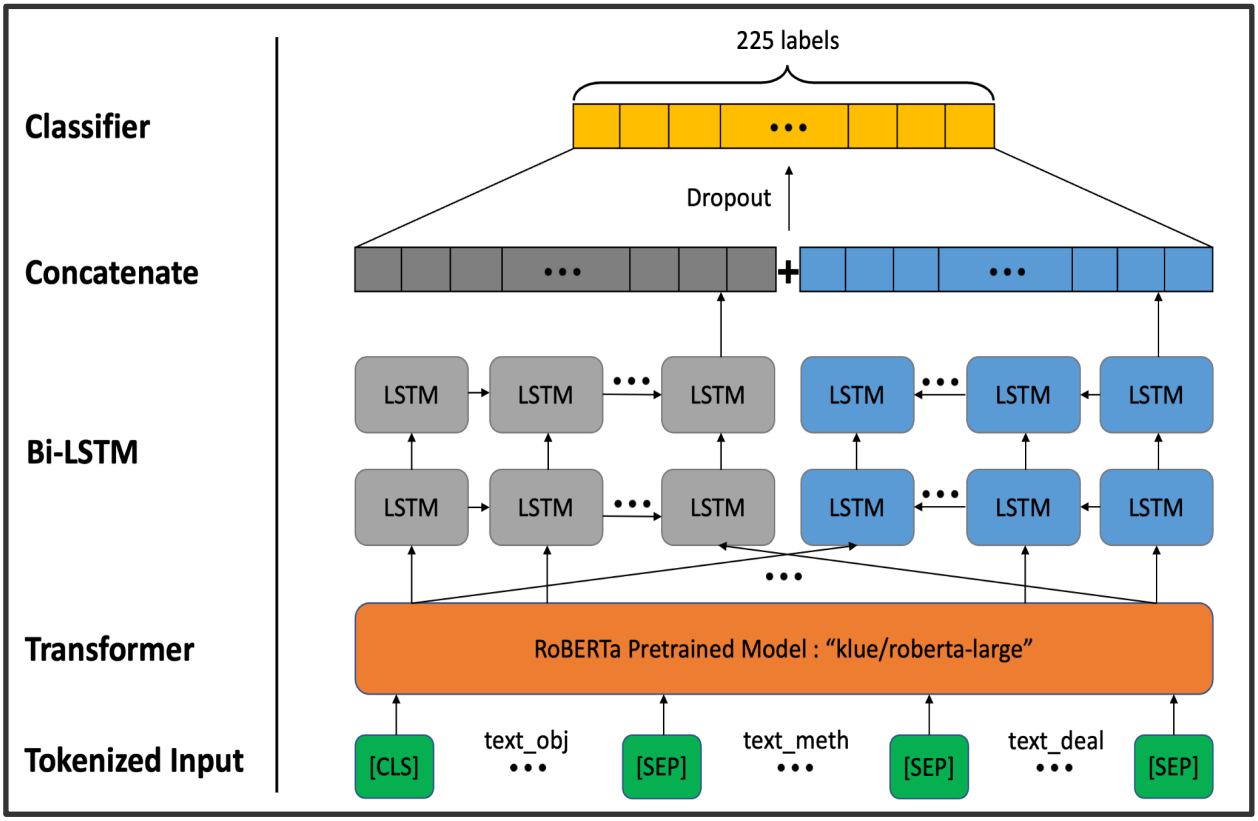
- data : 대분류, 중분류, 소분류의 label을 숫자와 mapping한 pickle 파일(학습을 위해서) + 대분류와 중분류, 중분류와 소분류의 포함관계를 딕셔너리 형태로 mapping한 pickle 파일(추론을 위해서)이 존재하는 폴더.
- input : 학습할 데이터와 추론할 데이터가 존재하는 폴더.
- utils
    - preprocessor.py : data에 있는 text obj, mthd, deal 부분을 하나의 문장으로 변형.
    - encoder.py : preprocessor를 통해서 하나 문장이 된 데이터를 tokenizer를 통해서 토큰화 및 인코딩.
    - scheduler.py : 학습할 때의 learning rate를 어떻게 schedule 한건지 결정.
- arguments.py : Pretrained Model 경로, 데이터 관련 설정, 모델 학습 파라미터(learning rate, epoch, batch size), 저장 경로 등을 argument 형태로 제공.
- model.py : transformer 라이브러리에서 roberta모델을 상속받아서 다양한 헤드(lstm, cnn)를 추가한 모델을 제공.
- train_kfold.py : 학습할 데이터를 sklearn 라이브러리에서 제공하는 StratifiedKFold 함수를 이용하여 arguments.py에 지정된 args에 따라 모델을 학습.
- trainer.py : transformers 라이브러리에서 제공하는 trainer 클래스를 상속받아서 R-drop 논문에서 제시한 학습 방법을 구현한 클래스 제공.
- inference.py : 모델을 학습하고 만들어진 K개의 checkpoint를 앙상블 (soft voting, hard voting) 한 결과물을 제작.
- running.sh : 최종 제출물을 만들기 위해서 학습, 추론할 때 사용한 명령어.

Ⅲ. 예측 모델의 흐름도 및 개요

학습 방향 : 소분류가 정해지면 중분류, 대분류가 정해지기 때문에, 학습 데이터에 존재하는 소분류 225개를 분류하는 Multi-Class Classification Task로 접근.

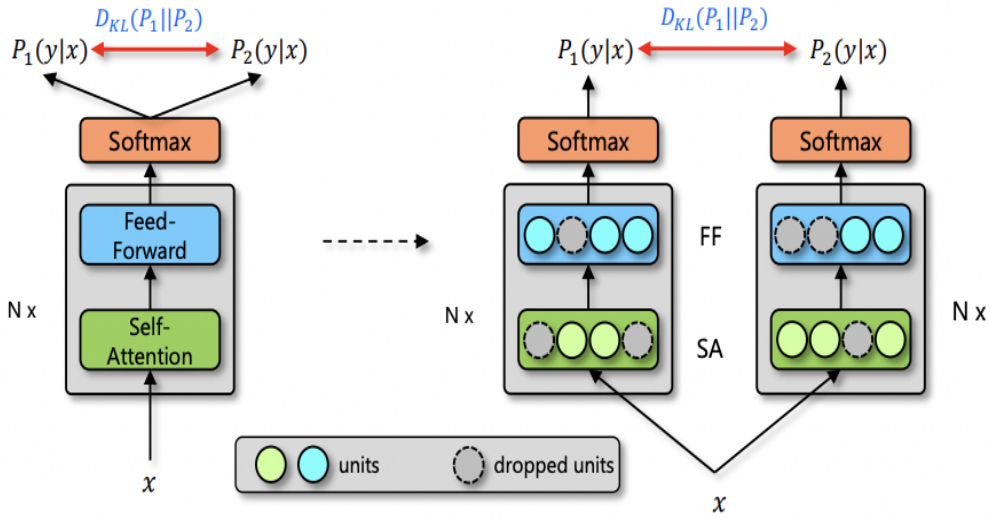
1. 데이터 변형
    a. KoSpacing 라이브러리를 활용해서 기존 데이터에 띄어쓰기 작업을 진행.
2. 데이터 전처리
    a. text_obj, text_mthd, text_deal 3개를 각각의 사이에 [SEP]를 집어넣어서 하나의 문장으로 변형.
    b. 학습 데이터에 사업 대상, 사업 방법, 사업 취급품목 column에 결측치 존재하는 경우가 있는데, 추론 데이터에서도 결측치가 있는 것으로 확인되어 학습과 추론에서 동일한 환경을 구성하기 위해 결측치가 있는 경우에는 공백으로 처리.
3. 데이터 토큰화 및 인코딩
    a. 하나의 문장이 된 데이터를 klue/roberta-large tokenizer로 토큰화 및 정수 인덱스로 인코딩.
    b. Dynamic Padding을 적용하여 학습 속도를 개선.
4. 모델 학습

    Pre-trained weight : **" klue/roberta-large "** 를 활용, 전체적인 모델 개요는 다음과 같다.



    a. 모델 구조 : 기본적인 Roberta 모델 구조에 LSTM Layer를 추가(RobertaLSTM) - 모든 문장의 구조가 대상, 방법, 취급품목 순서로 구성되어 있어 Bi-LSTM을 이용할 경우 이러한 순서 관계를 앞뒤로 잘 파악할 수 있을 것이라고 판단. 또한 문장의 길이가 짧기 때문에 Long term dependency의 문제가 발생하지 않을 것이라 판단되어 해당 방법을 적용.
    b. 목적 함수 : R-drop[1] 방법을 활용해서 모델을 학습 - Dropout으로 인해서 생기는 편향을 줄임으로써 모델의 일반화 성능을 높임.

[1] R-drop : https://proceedings.neurips.cc/paper/2021/file/5a66b9200f29ac3fa0ae244cc2a51b39-Paper.pdf



c. Optimizer 및 Scheduler : Optimizer의 경우 AdamW를 사용하고 Scheduler의 경우 Linear warmup scheduler를 사용.

d. Out Of Fold(OOF) : Stratified KFold를 이용하여 라벨 간 밸런스를 유지하고 총 10 fold를 진행 - 10가지 경우의 서로 다른 90000개의 훈련 데이터, 10000개의 검증 데이터로 10개의 모델을 제작 및 성능 검증(추론 할 때 10개의 모델을 앙상블).

5. 추론 과정

a. data 폴더에 존재하는 pickle 파일을 활용하여 소분류에 따라 중분류, 대분류를 Mapping.

b. 10 fold 학습을 통해서 생긴 10개의 checkpoint들을 soft-voting을 하여 test data 추론 결과를 csv 파일로 생성 및 제출.

IV. 모델 학습 파라미터

Epoch	3 (10548 Steps)	FP 16(Mixed precision)	True
Learning Rate	3e-5	Weight Decay	1e-3
Warmup ratio	0.05 (5% of total steps)	AdamW Beta1, Beta2	(0.9, 0.999)
Train Batch Size	256	AdamW Epsilon	1e-8
Eval Batch Size	128	Dropout	0.1
Seed	42		

신청자명	소속/직위/팀명	아주대학교/학부생/메타몽	성명	이기성
	휴대전화	010-9117-4691	전자우편	eternityk0716@gmail.com
제출일	2022-04-14 (목요일)			