KUBIG
DATA SCIENCE & AI

한국어 문장 관계 분류 프로젝트

DL 1팀 | 19기 안태림 20기 기광민 20기 장건호

KUBIG
DATA SCIENCE & AI

CONTENTS

KUBIG
DATA SCIENCE & AI

01. 프로젝트 소개

01. 주제 선정

- ✓ DL Basic Study에서 학습한 내용을 토대로 도전할만한 주제와 팀원의 관심사를 종합해 선정
- ✓ premise 문장을 참고해 hypothesis 문장이 참인지(Entailment), 거짓인지(Contradiction), 혹은 참/거짓 여부를 알 수 없는 문장인지(Neutral)를 판별하는 알고리즘 작성

Data

Natural Language Inference Dataset - Train dataset: 24998개 문장 쌍 / Test dataset: 1666개 문장 쌍

Premise → Hypothesis → label

01. 목표

1. 한국어 문장 관계 분류 모델의 성능 높이기

Premise	Hypothesis	label
재개발 정비사업 원주민들에 대한 종합대책을 마련하는 등 원주민 보호에 적극 나섰다.	원주민들은 종합대책에 만족했다.	Neutral
마지막 경기씬은 록키1보다 더 대단했다.	록키1에도 경기씬이 있다.	Entailment
빌은 그의 환자이던 루 네이단슨의 부고 소식을 접하게 되며, 그 집에 찾아가게 된다.	빌은 우연히 루 네이단슨의 집에 찾아가게 된다.	Contradiction

2. 적용하기

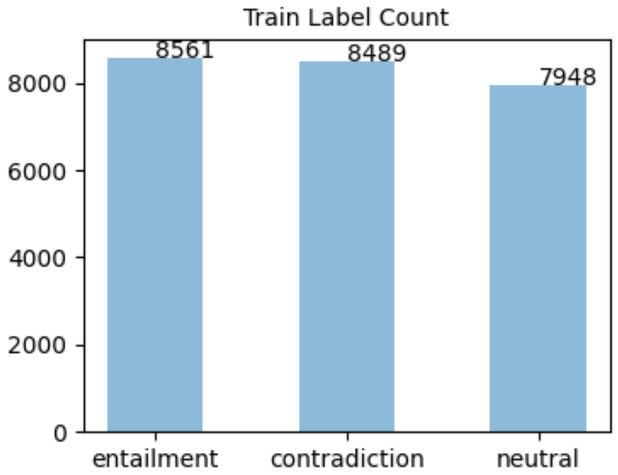
Premise	Hypothesis	label
쿠빅은 고려대학교 데이터 사이언스 및 인공지능 학회이다.	쿠빅 못생겼다	neutral
요즘 두바이 초콜릿이 품절 대란이다. 난리도 아니다.	두바이 초콜릿은 인기가 사그러들었다.	contradiction
쿠빅 학회원들은 열심히 준비한 프로젝트를 쿠빅 콘테스트에서 발표한다.	사실 쿠빅 학회원들은 콘테스트에 놀고 먹으러 왔다.	contradiction
고려대학교 학식은 매우 맛있다.	고려대학교 학생들은 학식을 먹지 않는다	contradiction

KUBIG
DATA SCIENCE & AI

02. 전처리 및 이론 소개

02. 데이터 전처리

데이터 불균형 확인



한글 이외의 단어 제거

```
train['premise'] = train['premise'].str.replace('[^ㄱ-ㅎㅏ-ㅣ가-힣 0-9]', '', regex=True)
test['hypothesis'] = test['hypothesis'].str.replace('[^ㄱ-ㅎㅏ-ㅣ가-힣 0-9]', '', regex=True)
```

이후 Tokenizing 시 Padding을 위한 문장 최대 길이 확인

```
Max Premise Length:  90
Min Premise Length:  19
Mean Premise Lenght:  45.406552524201935

Max Hypothesis Length:  103
Min Hypothesis Length:  5
Mean Hypothesis Lenght:  24.924433954716378
```

02. 데이터 증강

문맥을 고려한 한국어 텍스트 데이터 증강 (Korean Text Augmentation Considering Context, K-TACC)

Overview

- `BERT_augmentation` : BERT based 모델을 활용하여, 의미상 자연스러운 토큰을 삽입하거나 대체하는 형식으로 문장 augmentation 수행
- `Adverb_augmentation` : 부사를 그 부사의 뜻풀0
- 기존 EDA(Easy Data Augmentation), AEDA(A 장 생성 가능

KorEDA

이 프로젝트는 EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks 를 한국어로 쓸 수 있도록 wordnet 부분만 교체한 프로젝트 입니다.

wordnet은 KAIST에서 만든 Korean WordNet(KWN) 을 사용했습니다.

K-TACC 실험 결과를 참고하여 높은 성능을 보인 KorEDA Random Swap, KorEDA Random Synonym Replacement, Bert Random Masking Insertion, Bert Random Masking Replacement를 시도하여 비교

02. 데이터 증강

KorEDA Random Swap

무작위로 문장 내에서 두 단어를 선택하고 위치를 바꾼다.

예시: [사람들은 이순신을 뛰어난 장군으로 생각한다]

사람들은 생각한다 이순신을 뛰어난 장군으로

Bert Random Masking Insertion

문장에 [mask] 토큰을 추가하고 이를 복원한다.

사람들은 이순신을 뛰어난 조선의 장군으로 생각한다

KorEDA Random Synonym Replacement

문장에서 랜덤으로 stop words가 아닌 n 개의 단어들을 선택해 임의로 선택한 동의어들 중 하나로 바꾼다.

사람들은 이순신을 위대한 장군으로 생각한다

Bert Random Masking Replacement

특정 단어를 masking한 뒤 다시 복원한다.

사람들은 이순신을 위대한 전략가로 생각한다

02. Pretrained model

KLUE-RoBERTa large

한국어 대규모 데이터셋으로 학습된 BERT/RoBERTa 기반의 모델

Hugging face의 transformers 패키지를 통해서 호출

KoElectra base

Generator(G), discriminator(D)를 이용한 Replaced Token Detection 기반 모델

KLUE-RoBERTa large와 견줄만한 accuracy

KoGPT2 (skt/kogpt2-base-v2)

문장 생성에 최적화된, 한국어 디코더 모델

낮은 accuracy로 모델 선택에서 제외

02. 주요 기법

Softmax로 앙상블 구현

앙상블 시도하려 했으나 용량 제한 문제로 어려움

➔ 각 단일 모델에 Softmax 적용 후 label별 확률값을 종합하여 최종 결과를 도출하는 방식으로 앙상블 구현

➔ 다양한 기준으로 가중치를 상이하게 부여 (예시. 단일 모델들의 Accuracy)

```
dataloader = DataLoader(test_dataset, batch_size=16, shuffle=False)

model.eval()
output_pred = []
output_prob = []

for i, data in enumerate(tqdm(dataloader)):
    with torch.no_grad():
        outputs = model(
            input_ids=data['input_ids'].to(device),
            attention_mask=data['attention_mask'].to(device),
            token_type_ids=data['token_type_ids'].to(device)
        )
    logits = outputs[0]
    prob = F.softmax(logits, dim=-1).detach().cpu().numpy()
    logits = logits.detach().cpu().numpy()
    result = np.argmax(logits, axis=-1)

    output_pred.append(result)
    output_prob.append(prob)

pred_answer, output_prob = np.concatenate(output_pred).tolist(), np.concatenate(output_prob, axis=0).tolist()
print(pred_answer)
```

KUBIG
DATA SCIENCE & AI

03. Train & Test

03. 모델링

Hugging Face Hub의 Transformers 라이브러리 -> Pretrained Tokenizer와 Model 불러와서 적용

KLUE-RoBERTa large

```python
from transformers import AutoModelForSequenceClassification, AutoTokenizer, AutoConfig

tokenizer = AutoTokenizer.from_pretrained("klue/roberta-large")

config = AutoConfig.from_pretrained("klue/roberta-large")
config.num_labels = 3

model = AutoModelForSequenceClassification.from_pretrained("klue/roberta-large", config=config)
```


```
RobertaConfig {
  "_name_or_path": "klue/roberta-large",
  "architectures": [
    "RobertaForMaskedLM"
  ],
  "attention_probs_dropout_prob": 0.1,
  "bos_token_id": 0,
  "classifier_dropout": null,
  "eos_token_id": 2,
  "gradient_checkpointing": false,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 1024,
  "id2label": {
    "0": "LABEL_0",
    "1": "LABEL_1",
    "2": "LABEL_2"
  },
  "initializer_range": 0.02,
  "intermediate_size": 4096,
  "label2id": {
    "LABEL_0": 0,
    "LABEL_1": 1,
    "LABEL_2": 2
  },
  "layer_norm_eps": 1e-05,
  "max_position_embeddings": 514,
  "model_type": "roberta",
  "num_attention_heads": 16,
  "num_hidden_layers": 24,
  "pad_token_id": 1,
  "position_embedding_type": "absolute",
  "tokenizer_class": "BertTokenizer",
  "transformers_version": "4.42.4",
  "type_vocab_size": 1,
  "use_cache": true,
  "vocab_size": 32000
}
```

03. 모델링

Hugging Face Hub의 Transformers 라이브러리 -> Pretrained Tokenizer와 Model 불러와서 적용

KoELECTRA base

```
from transformers import ElectraTokenizer, ElectraForSequenceClassification

koelectra_tokenizer = ElectraTokenizer.from_pretrained("monologg/koelectra-base-discriminator")
koelectra_model = ElectraForSequenceClassification.from_pretrained("monologg/koelectra-base-discriminator", num_labels=3)
```


```
ElectraConfig {
  "_name_or_path": "monologg/koelectra-base-discriminator",
  "architectures": [
    "ElectraForPreTraining"
  ],
  "attention_probs_dropout_prob": 0.1,
  "classifier_dropout": null,
  "embedding_size": 768,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "id2label": {
    "0": "LABEL_0",
    "1": "LABEL_1",
    "2": "LABEL_2"
  },
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "label2id": {
    "LABEL_0": 0,
    "LABEL_1": 1,
    "LABEL_2": 2
  },
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "electra",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "pad_token_id": 0,
  "position_embedding_type": "absolute",
  "summary_activation": "gelu",
  "summary_last_dropout": 0.1,
  "summary_type": "first",
  "summary_use_proj": true,
  "transformers_version": "4.42.4",
  "type_vocab_size": 2,
  "use_cache": true,
  "vocab_size": 32200
}
```

03. Train & Test

Hugging Face Trainer Class를 사용하여 사전 학습된 모델을 Fine-Tuning하고 Train 진행

```python
from transformers import TrainingArguments, Trainer, EarlyStoppingCallback

training_ars = TrainingArguments(
    output_dir='./result', # 학습 결과물을 저장할 디렉토리를 지정한다.
    num_train_epochs=2, # 학습 에폭 수를 지정한다.
    per_device_train_batch_size=32, # 각 디바이스당 학습 배치 크기를 지정한다.
    save_total_limit=5,
    save_steps=500, # 일정한 스텝마다 모델 가중치를 저장하는 주기를 지정한다.
    evaluation_strategy='steps', # 매 eval_steps마다 평가
    eval_steps = 500,
    logging_steps=500, # 일정한 스텝마다 로깅을 수행하는 주기
    load_best_model_at_end = True
)

trainer = Trainer(
    model=model, # = AutoModelForSequenceClassification.from_pretrained('klue/roberta-large', config=config)
    args=training_ars, # 학습 설정을 담고 있는 TrainingArguments 객체. 학습에 관련된 설정들을 조정하고 제어하는데 사용된다.
    train_dataset=train_dataset,
    eval_dataset=eval_dataset,
    tokenizer=tokenizer, # 모델의 토크나이저(tokenizer) 객체. 텍스트 데이터를 모델이 이해할 수 있는 형태로 변환하는데 사용된다.
    compute_metrics=compute_metrics,
)
```

```python
trainer.train()
model.save_pretrained('./result/best_model')
```

[1514/1514 50:08, Epoch 2/2]

Step	Training Loss	Validation Loss	Accuracy
500	0.550400	0.401850	0.870200
1000	0.303400	0.365086	0.892200
1500	0.173700	0.325390	0.903200

03. Train & Test

```
dataloader = DataLoader(test_dataset, batch_size=16, shuffle=False)

model.eval()
output_pred = []
output_prob = []

for i, data in enumerate(tqdm(dataloader)):
    with torch.no_grad():
        outputs = model(
            input_ids=data['input_ids'].to(device),
            attention_mask=data['attention_mask'].to(device),
            token_type_ids=data['token_type_ids'].to(device)
        )
    logits = outputs[0]
    prob = F.softmax(logits, dim=-1).detach().cpu().numpy()
    logits = logits.detach().cpu().numpy()
    result = np.argmax(logits, axis=-1)

    output_pred.append(result)
    output_prob.append(prob)

pred_answer, output_prob = np.concatenate(output_pred).tolist(), np.concatenate(output_prob, axis=0).tolist()
print(pred_answer)
```

```
100%|██████████| 105/105 [00:22<00:00,  4.67it/s][1, 2, 0, 1, 0, 2, 2, 2, 0, 1, 1, 0, 1, 0, 2, 2, 2, 2, 1, 2, 2, 2, 2, 0, 1,
```

```
def num_to_label(label):
    label_dict = {0: "entailment", 1: "contradiction", 2: "neutral"}
    str_label = []

    for i, v in enumerate(label):
        str_label.append([i,label_dict[v]])

    return str_label

answer = num_to_label(pred_answer)
print(answer)
```

```
[[0, 'contradiction'], [1, 'neutral'], [2, 'entailment'], [3, 'contradiction'], [4, 'contradiction'], [5, 'neutral'], [6, 'neutral'], [7
```

03. 시행착오

학습 데이터 & 검증 데이터 분할 이전에 데이터 증강

➔ 비슷한 문장들이 검증 데이터에 들어가기 때문에 학습에서는 Accuracy가 높게 나왔지만 데이콘 Public Score은 하락

★ 분할 이후에 데이터 증강

Epoch 횟수 늘리기

➔ 오버피팅으로 인한 Validation Loss 증가 및 데이콘 Public Score 하락

~~★ early stopping patience을 통한 과적합 방지~~

★ 단순 epoch수 감소

[4000/5425 2:15:20 < 48:14, 0.49 it/s, Epoch 5/7]

Step	Training Loss	Validation Loss	Accuracy
500	0.613600	0.455595	0.836129
1000	0.379800	0.448421	0.863226
1500	0.295500	0.354767	0.875000
2000	0.196200	0.442780	0.895484
2500	0.156100	0.400232	0.894516
3000	0.111000	0.492232	0.902097
3500	0.082700	0.462592	0.905161
4000	0.057500	0.538779	0.911290

03. 시행착오

Tokenizing시 Max Length 256으로 설정

➔ GPU 제한 및 시간 초과로 진행에 어려움 존재

★ 문장의 최대 길이인 103으로 조정

Step	Training Loss	Validation Loss	Accuracy
500	0.586600	0.416747	0.852586
1000	0.343900	0.418555	0.868448
1500	0.226600	0.509217	0.880862
2000	0.104500	0.496500	0.884828

[2271/2271 1:14:49, Epoch 3/3]

Step	Training Loss	Validation Loss	Accuracy
500	0.601200	0.377758	0.866800
1000	0.333900	0.438089	0.863000
1500	0.217600	0.371832	0.883000
2000	0.096900	0.456151	0.895800

데이터 증강 방식

➔ 증강 비율 2배로 (4200개 -> 8400개) 올릴 시 데이콘 Public Score 하락

★ 기존 증강 비율 유지

[1514/1514 48:57, Epoch 2/2]

Step	Training Loss	Validation Loss	Accuracy
500	0.573200	0.433706	0.854200
1000	0.305400	0.394685	0.886800
1500	0.173900	0.337312	0.900000

➔ 단일 증강 VS 복합 증강

★ 동일한 조건하에 복합 증강 시 데이콘 Public Score이 가장 높게 나옴

[1514/1514 50:08, Epoch 2/2]

Step	Training Loss	Validation Loss	Accuracy
500	0.550400	0.401850	0.870200
1000	0.303400	0.365086	0.892200
1500	0.173700	0.325390	0.903200

KUBIG
DATA SCIENCE & AI

04. 결론

04. 인사이트

[1514/1514 15:08, Epoch 2/2]

Step	Training Loss	Validation Loss	Accuracy
500	0.647400	0.490143	0.811200
1000	0.390500	0.456896	0.843400
1500	0.275400	0.437847	0.851400

[1514/1514 50:08, Epoch 2/2]

Step	Training Loss	Validation Loss	Accuracy
500	0.550400	0.401850	0.870200
1000	0.303400	0.365086	0.892200
1500	0.173700	0.325390	0.903200

KoELECTRA base와 KLUE-RoBERTa large에 각각 0.4와 0.6의 앙상블 가중치를

설정하였을 때 가장 높은 점수를 얻을 수 있었음

Public 0.863 / Private 0.853

데이콘 Public Score 기준 상위 11프로

1050752	submission_ensemble.csv edit		2024-08-25 19:23:13	0.863 0.8535414166

- Train Data 증강을 통한 학습 성능 향상

- 2개의 모델을 결합해 분류 성능 향상

- 하이퍼 파라미터 fine-tuning을 통해 모델 성능 최적화

04. 인사이트

직접 적용하기 결과

<1>

Premise : 쿠빅은 고려대학교 데이터 사이언스 및 인공지능 학회이다.

Hypothesis : 쿠빅 못생겼다.

label : neutral

prediction : neutral

<2>

Premise : 요즘 두바이 초콜릿이 품절 대란이다. 난리도 아니다.

Hypothesis : 두바이 초콜릿은 인기가 사그러들었다.

label : contradiction

prediction : contradiction

index	label
0	neutral
1	contradiction
2	neutral
3	contradiction

04. 인사이트

직접 적용하기 결과

<3>

Premise : 쿠빅 학회원들은 열심히 준비한 프로젝트를 쿠빅 콘테스트에서 발표한다.

Hypothesis : 사실 쿠빅 학회원들은 콘테스트에 놀고 먹으러 왔다.

label : contradiction

prediction : neutral

<4>

Premise : 고려대학교 학식은 매우 맛있어서 많이 먹는다.

Hypothesis : 고려대학교 학생들은 학식을 먹지 않는다

label : contradiction

prediction : contradiction

index	label
0	neutral
1	contradiction
2	neutral
3	contradiction

04. Discussion

Test data 사례

<1>

Premise : '식목일이자 절기상 봄농사를 시작하는 청명인 5일 전국이 포근해 봄 기운을 물씬 느낄 수 있겠다.'

Hypothesis : 5일은 절기상 봄농사를 시작할 때이나 평소보다 쌀쌀할 예정이다.

prediction : contradiction

<2>

Premise : '현재 장성택의 매형인 전영진 쿠바 주재 북한 대사와 조카인 장용철 말레이시아 대사도 조선민주주의인민공화국 본국으로 소환된 것으로 전해졌다.'

Hypothesis : 전영진과 장용철은 조산민주주의인민공화국 출신이다.

prediction : neutral

04. Limitation

GPU 제한으로 인한 어려움

보다 더 다양한 모델과 데이터셋 구성에 대한 실험 진행의 어려움

Trainer 파라미터 제한

TrainingArguments 및 Trainer 내 파라미터들인 gradient accumulation, weight_decay, optimizer, learning rate 등에 대해서 더 많은 탐색 필요

지속적인 Accuracy 오류

KLUE-RoBERTa large에 기존데이터 + 3가지 2800개씩 증강 (8200개) 한 데이터를 통한 학습에서 오류 발생

1400개씩 증강 (4200개) 했을 때는 문제 발생 x

➔ 데이터가 커진 만큼 배치 사이즈나 학습률의 조정이 필요했을 것으로 예상

KUBIG
DATA SCIENCE & AI

Thank You