

## 자연어처리 응용 과제

# Transformer 기반 사전 학습 모델을 이용한 감정 분석: IMDB 영화 리뷰 데이터셋을 활용한 DistilBERT 모델의 미세 조정 및 평가

건국대학교 메타버스융합전공 박사과정

손우람 (202371380)

2024년 6월 13일

### 과제 소개

이 과제는 Hugging Face의 transformers 라이브러리와 IMDB 영화 리뷰 데이터셋을 사용하여 감정 분석을 수행하는 방법을 탐구합니다. 본 연구에서는 DistilBERT 모델을 활용하여 텍스트 데이터를 분석하고, 긍정적, 부정적 감정을 분류하는 방법을 다룹니다. IMDB 영화 리뷰 데이터셋을 이용해 모델을 미세 조정(fine-tuning)하고, 이를 통해 감정 분석의 성능을 향상시키는 과정을 설명합니다.

### 목표

- Transformer 기반의 사전 학습된 언어 모델인 DistilBERT를 이해하고 활용하는 방법을 학습한다.
- IMDB 영화 리뷰 데이터셋을 사용하여 감정 분석 모델을 구축하고, 모델의 성능을 평가한다.
- 모델 미세 조정(fine-tuning) 과정을 통해 감정 분석의 정확도를 향상시킨다.

### 세부 내용

- 데이터셋 로드 및 전처리: Hugging Face의 datasets 라이브러리를 사용하여 IMDB 영화 리뷰 데이터셋을 로드하고, 학습 및 평가 데이터셋으로 분할합니다.
- 모델 및 토큰라이저 로드: DistilBERT 모델과 토큰라이저를 로드하여 텍스트 데이터를 처리할 준비를 합니다.
- 데이터 토큰화: 데이터셋을 모델에 맞게 토큰화하고, 필요한 형식으로 변환합니다.
- 모델 미세 조정: Trainer를 사용하여 사전 학습된 DistilBERT 모델을 IMDB 데이터셋에 맞춰 미세 조정합니다.
- 평가 및 결과 분석: 모델을 평가하고, 새로운 텍스트 데이터를 통해 감정 분석을 수행하여 결과를 확인합니다.

## 필요 패키지 및 버전

프로젝트를 실행한 환경:

- Python 3.9.5
- macOS Sonoma (14.5)
- Apple M1 Pro, 16GB

필요한 파이썬 패키지와 버전:

- Python: 3.9.5
- transformers: 4.18.0 이상
- datasets: 1.18.0 이상
- accelerate: 0.21.0 이상
- evaluate: 0.3.0 이상
- torch: 1.10.0 이상

## 패키지 설치 명령어

터미널에서 다음 명령어를 실행하여 필요한 패키지를 설치합니다:

```
pip install transformers>=4.18.0
pip install datasets>=1.18.0
pip install accelerate>=0.21.0
pip install evaluate>=0.3.0
pip install torch>=1.10.0
```

## 코드 설명 - Hugging Face의 transformers 라이브러리를 이용한 감정 분석

아래 코드는 Hugging Face의 transformers 라이브러리를 사용하여 감정 분석을 수행합니다. 이 코드에서는 사전 학습된 감정 분석 모델을 로드하고, 주어진 텍스트에 대해 감정 분석을 수행한 후 결과를 출력합니다.

```
from transformers import pipeline

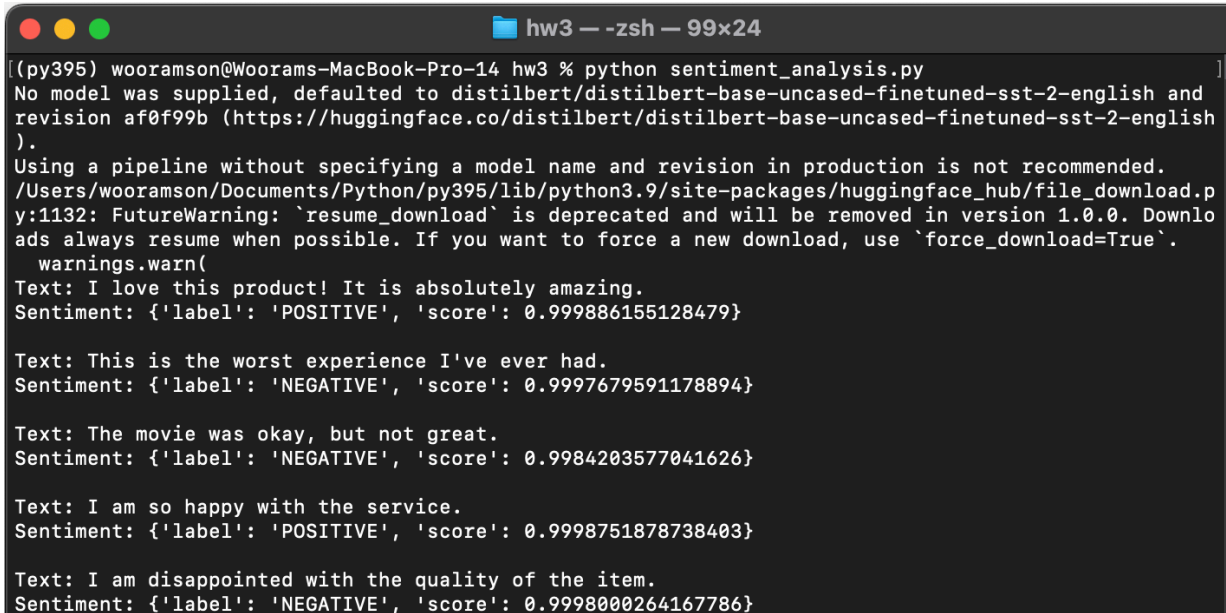
# 감정 분석 파이프라인 로드
sentiment_pipeline = pipeline("sentiment-analysis")

# 분석할 텍스트 정의
texts = [
    "I love this product! It is absolutely amazing.",
    "This is the worst experience I've ever had.",
    "The movie was okay, but not great.",
    "I am so happy with the service.",
    "I am disappointed with the quality of the item."
]

# 감정 분석 수행
results = sentiment_pipeline(texts)

# 결과 출력
for text, result in zip(texts, results):
    print(f"Text: {text}")
    print(f"Sentiment: {result}\n")
```

코드 실행 결과:



```
hw3 — -zsh — 99x24
[(py395) wooramson@Woorams-MacBook-Pro-14 hw3 % python sentiment_analysis.py
No model was supplied, defaulted to distilbert/distilbert-base-uncased-finetuned-sst-2-english and
revision af0f99b (https://huggingface.co/distilbert/distilbert-base-uncased-finetuned-sst-2-english
).
Using a pipeline without specifying a model name and revision in production is not recommended.
/Users/wooramson/Documents/Python/py395/lib/python3.9/site-packages/huggingface_hub/file_download.p
y:1132: FutureWarning: `resume_download` is deprecated and will be removed in version 1.0.0. Downlo
ads always resume when possible. If you want to force a new download, use `force_download=True`.
warnings.warn(
Text: I love this product! It is absolutely amazing.
Sentiment: {'label': 'POSITIVE', 'score': 0.999886155128479}

Text: This is the worst experience I've ever had.
Sentiment: {'label': 'NEGATIVE', 'score': 0.9997679591178894}

Text: The movie was okay, but not great.
Sentiment: {'label': 'NEGATIVE', 'score': 0.9984203577041626}

Text: I am so happy with the service.
Sentiment: {'label': 'POSITIVE', 'score': 0.9998751878738403}

Text: I am disappointed with the quality of the item.
Sentiment: {'label': 'NEGATIVE', 'score': 0.999800264167786}
```

### 코드 설명 - IMDB 영화 리뷰 데이터셋을 활용한 DistilBERT 모델의 미세 조정 및 평가

이 코드는 Hugging Face의 transformers와 datasets 라이브러리를 사용하여 IMDB 영화 리뷰 데이터셋에 대한 감정 분석을 수행합니다. 코드는 크게 다섯 부분으로 나뉩니다: 데이터 로드 및 전처리, 모델 및 토크나이저 로드, 데이터 토큰화, 모델 미세 조정, 감정 분석 수행 및 결과 출력.

#### 1. 데이터셋 로드 및 전처리

먼저, IMDB 영화 리뷰 데이터셋을 로드하고, 학습 및 평가 데이터셋으로 분할합니다.

```
import torch
from transformers import DistilBertTokenizerFast, DistilBertForSequenceClassification,
Trainer, TrainingArguments
from datasets import load_dataset
import evaluate

# 데이터셋 로드
dataset = load_dataset('imdb')

# 데이터셋을 학습 및 평가 세트로 분할
train_dataset = dataset['train'].shuffle(seed=42).select(range(2000))
test_dataset = dataset['test'].shuffle(seed=42).select(range(500))
```

#### 2. 모델 및 토크나이저 로드

DistilBERT 모델과 해당 토크나이저를 로드합니다.

```
# 토크나이저 및 모델 로드
tokenizer = DistilBertTokenizerFast.from_pretrained('distilbert-base-uncased')
model = DistilBertForSequenceClassification.from_pretrained('distilbert-base-uncased')
```

#### 3. 데이터 토큰화

텍스트 데이터를 모델이 처리할 수 있는 형식으로 변환합니다.

```
# 데이터셋을 토큰화
def tokenize_function(examples):
    return tokenizer(examples['text'], padding="max_length", truncation=True)

train_dataset = train_dataset.map(tokenize_function, batched=True)
test_dataset = test_dataset.map(tokenize_function, batched=True)
```

```
train_dataset = train_dataset.remove_columns(['text'])
test_dataset = test_dataset.remove_columns(['text'])

train_dataset.set_format('torch')
test_dataset.set_format('torch')
```

#### 4. 모델 미세 조정

모델을 학습시키기 위해 필요한 설정을 정의하고, Trainer를 사용하여 모델을 미세 조정합니다.

```
# 평가 메트릭 정의
metric = evaluate.load('accuracy')

def compute_metrics(eval_pred):
    logits, labels = eval_pred
    predictions = torch.argmax(logits, dim=-1)
    return metric.compute(predictions=predictions, references=labels)

# 학습 인자 정의
training_args = TrainingArguments(
    output_dir='./results',
    eval_strategy="epoch",
    learning_rate=2e-5,
    per_device_train_batch_size=8,
    per_device_eval_batch_size=8,
    num_train_epochs=3,
    weight_decay=0.01,
)

# Trainer 정의
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=train_dataset,
    eval_dataset=test_dataset,
    compute_metrics=compute_metrics,
```

```
)  
  
# 모델 학습  
trainer.train()
```

## 5. 감정 분석 수행 및 결과 출력

미세 조정된 모델을 사용하여 감정 분석을 수행하고, 결과를 출력합니다.

```
# 학습된 모델로 감정 분석 파이프라인 생성  
from transformers import pipeline  
sentiment_pipeline = pipeline("sentiment-analysis", model=model, tokenizer=tokenizer)  
  
# 분석할 텍스트 정의  
texts = [  
    "I love this product! It is absolutely amazing.",  
    "This is the worst experience I've ever had.",  
    "The movie was okay, but not great.",  
    "I am so happy with the service.",  
    "I am disappointed with the quality of the item."  
]  
  
# 감정 분석 수행  
results = sentiment_pipeline(texts)  
  
# 결과 출력  
for text, result in zip(texts, results):  
    print(f"Text: {text}")  
    print(f"Sentiment: {result}\n")
```

## 주요 부분 설명

- 데이터셋 로드 및 전처리: `load_dataset` 함수를 사용하여 IMDB 데이터셋을 로드하고, 학습과 평가를 위한 데이터셋으로 분할합니다.
- 모델 및 토큰라이저 로드: `DistilBertTokenizerFast`와 `DistilBertForSequenceClassification`을 사용하여 모델과 토큰라이저를 로드합니다.
- 데이터 토큰화: 텍스트 데이터를 모델에 맞게 토큰화하여 입력 형식으로 변환합니다.

- 이 과정을 통해 최신 NLP 기법을 사용하여 감정 분석을 수행하는 방법을 배우고, 실제 데이터셋을 활용하여 모델을 학습시키고 평가하는 방법을 익힐 수 있습니다.

터미널을 열고 파일이 저장된 디렉터리로 이동한 후, 다음 명령어를 입력하여 스크립트를 실행합니다:

```
(py395) woramson@Woramson-MacBook-Pro-14 hw3 % python sentiment_analysis_imdb.py  
Some weights of DistilBertForSequenceClassification were not initialized from the model checkpoint  
at distilbert-base-uncased and are newly initialized: ['classifier.bias', 'classifier.weight', 'pre  
_classifier.bias', 'pre_classifier.weight']  
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and  
inference.  
{'eval_loss': 0.3697955310344696, 'eval_accuracy': 0.858, 'eval_runtime': 11.1558, 'eval_samples_pe  
r_second': 44.82, 'eval_steps_per_second': 5.647, 'epoch': 1.0}  
{'loss': 0.3536, 'grad_norm': 30.76290512084961, 'learning_rate': 6.666666666666667e-06, 'epoch': 2  
.0}  
{'eval_loss': 0.36677539348602295, 'eval_accuracy': 0.89, 'eval_runtime': 10.7978, 'eval_samples_pe  
r_second': 46.306, 'eval_steps_per_second': 5.835, 'epoch': 2.0}  
{'eval_loss': 0.4181872010231018, 'eval_accuracy': 0.888, 'eval_runtime': 10.6824, 'eval_samples_pe  
r_second': 46.806, 'eval_steps_per_second': 5.898, 'epoch': 3.0}  
{'train_runtime': 458.0583, 'train_samples_per_second': 13.099, 'train_steps_per_second': 1.637, 't  
rain_loss': 0.2854630889892578, 'epoch': 3.0}  
100%|██████████████████████████████████████████████████████████████████████████████| 750/750 [07:38<00:00, 1.64it/s]  
Text: I love this product! It is absolutely amazing.  
Sentiment: {'label': 'LABEL_1', 'score': 0.9929469227790833}  
  
Text: This is the worst experience I've ever had.  
Sentiment: {'label': 'LABEL_0', 'score': 0.9813004732131958}  
  
Text: The movie was okay, but not great.  
Sentiment: {'label': 'LABEL_0', 'score': 0.9665337204933167}  
  
Text: I am so happy with the service.  
Sentiment: {'label': 'LABEL_1', 'score': 0.9883363842964172}  
  
Text: I am disappointed with the quality of the item.  
Sentiment: {'label': 'LABEL_0', 'score': 0.9888933300971985}
```

이번 과제에서는 Hugging Face의 transformers와 datasets 라이브러리를 사용하여 IMDB 영화 리뷰 데이터셋을 기반으로 감정 분석을 수행하는 방법을 탐구했습니다. 이를 통해 사전 학습된 DistilBERT 모델을 미세 조정(fine-tuning)하여 감정 분석의 정확도를 높이는 과정을 다뤘습니다.

- 사전 학습된 모델 활용: Hugging Face의 transformers 라이브러리를 사용하여 DistilBERT와 같은 사전 학습된 모델을 쉽게 로드하고 사용할 수 있음을 배웁니다.

2. 데이터셋 로드 및 전처리: `datasets` 라이브러리를 통해 IMDB 영화 리뷰 데이터셋을 로드하고, 이를 학습 및 평가 데이터로 나누어 처리하는 방법을 익혔습니다.
3. 모델 미세 조정: `Trainer` 클래스를 활용하여 모델을 미세 조정하고, 주어진 데이터셋에 맞게 모델을 학습시켜 성능을 최적화하는 방법을 학습했습니다.
4. 평가 및 결과 분석: 미세 조정된 모델을 사용하여 새로운 텍스트 데이터에 대해 감정 분석을 수행하고, 그 결과를 해석하는 방법을 배웠습니다.

### 제한점(Limitations):

1. 데이터 의존성: 모델의 성능은 학습에 사용된 데이터셋의 품질과 다양성에 크게 의존합니다. IMDB 데이터셋은 영화 리뷰에 특화되어 있어, 다른 도메인이나 언어로의 일반화에는 한계가 있을 수 있습니다.
2. 연산 자원 요구: 대규모 모델을 미세 조정하고 평가하는 과정은 상당한 연산 자원과 시간이 필요합니다. 특히, GPU가 없는 환경에서는 처리 속도가 느려질 수 있습니다.
3. 모델 편향성: 사전 학습된 모델은 학습에 사용된 데이터의 편향성을 반영할 수 있습니다. 이는 특정 의견이나 감정을 과도하게 반영하거나 왜곡할 수 있는 위험을 내포하고 있습니다.
4. 한계적인 이해 능력: 현재의 감정 분석 모델은 텍스트의 표면적 의미를 분석하는 데 강하지만, 맥락적 의미나 복잡한 감정 상태를 이해하는 데는 한계가 있습니다.

이번 과제를 통해 최신 NLP 기술과 Hugging Face의 도구들을 활용하여 텍스트 데이터를 분석하고, 감정을 분류하는 방법을 익혔습니다. 이는 감정 분석, 고객 피드백 분석, 소셜 미디어 모니터링 등 다양한 실생활 문제에 적용할 수 있는 유용한 기술입니다. 또한, 사전 학습된 모델을 미세 조정함으로써 특정 도메인에 맞게 모델의 성능을 최적화할 수 있는 방법을 이해하게 되었습니다.



## 참고 문헌

1. Hugging Face Transformers 라이브러리:  
<https://huggingface.co/docs/transformers/index>
  - a. GitHub 레포지토리: <https://github.com/huggingface/transformers>
2. Datasets 라이브러리: <https://huggingface.co/docs/datasets/>
  - a. GitHub 레포지토리: <https://github.com/huggingface/datasets>
3. Evaluate 라이브러리: <https://huggingface.co/docs/evaluate/index>
  - a. GitHub 레포지토리: <https://github.com/huggingface/evaluate>
4. PyTorch: <https://pytorch.org/docs/stable/index.html>
  - a. GitHub 레포지토리: <https://github.com/pytorch/pytorch>
5. IMDB 영화 리뷰 데이터셋
  - a. Hugging Face Datasets 페이지: <https://huggingface.co/datasets/imdb>
6. Hugging Face 모델 허브
  - a. 모델 허브: <https://huggingface.co/models>
7. Hugging Face 파이프라인 API
  - a. 공식 문서: [https://huggingface.co/docs/transformers/main\\_classes/pipelines](https://huggingface.co/docs/transformers/main_classes/pipelines)
8. Hugging Face 튜토리얼: <https://huggingface.co/learn/nlp-course/chapter1>