

KUBIG

KU독 & 종아요

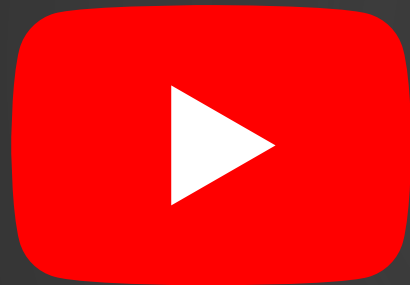
The logo for KU Tube features a red rounded rectangle with a white play button icon on the left. To its right, the text "KU Tube" is displayed in a large, bold, sans-serif font. The letters "KU" are light red, and "Tube" is white.

TABLE OF CONTENTS

[01] Introduce of Project

[02] INTRODUCTION of Data
- Data Crawling

[03] Data Pretreatment

[04] Modeling

01.

Introduce of Project

Introduce of Project

동영상 서비스 이용자 93% "유튜브 시청"...10대는 99.2%

단순한 동영상 플랫폼을 넘어 하나의 사회가 되어가는 유튜브에서 **영상의 조회수**는 곧 사회적 영향력과 개인의 수익과 직결되는 요소이다. 이러한 점에서 모든 유튜버들의 목표는 높은 조회수이고 높은 조회수를 위한 첫 시작은 단연코 **영상의 제목**일 것이다.



Introduce of Project

유튜브 영상의 조회수를 높게 나오기 위해선

- ✓ 어떤 제목이 필요할까?
- ✓ 영상 내용을 담은 제목이어야 하는 것인가?
- ✓ 자극적인 제목이 많은 조회수가 나올까?

>> 영상의 제목과 조회수 사이의 상관관계를 알아보고
제목에 따른 예상 조회수를 예측하는 프로젝트



YouTube

02.

Introduction of Data

데이터 수집





데이터 수집 기준

유튜브 랭킹

통합검색

전체	게임	BJ/인물/연예인	음악/댄스/가수	TV/방송	음식/요리/레시피	패션/미용	뉴스/정치/사회
취미/라이프	IT/기술/컴퓨터	회사/오피셜	교육/강의	영화/만화/애니	키즈/어린이	애완/반려동물	스포츠/운동
국내/해외/여행	자동차	주식/경제/부동산	해외	미분류	검색어	검색	

전체 160건 (현재 1페이지)

순위	이미지	제목	구독자수	View수	Video수	조회수
288		[스포츠/운동] SPOTV	170만	11억8805만	29,623개	6,450
1110		[스포츠/운동] 이스타TV 이스타 이스타TV 이주현 박종윤 려추종윤 허든못볼	56만	8억8029만	5,670개	4,302
406		[스포츠/운동] Shoot for Love 스포터브 Shoot for Love "shoot for love" 축구 스포츠 sports	130만	7억5322만	832개	3,459
1506		[스포츠/운동] 스포츠타임 스포티비 뉴스 SPOTV SPORTSTIME 스포츠타임 NEWS	40만	6억4091만	8,330개	2,875

'유튜브 랭크' 사이트 사용

총 7개의 주제 카테고리 선정

스포츠, 교육, 반려동물, IT, 경제, 정치, 패션

>> 독립적인 내용의 주제 선택

구독자순으로 30명의 유튜버 별 인기 동영상 150개

[제목, 조회수, 업로드 일자, 댓글수, 좋아요수, 구독자수]

>> 인기 동영상 데이터만 사용

= 조회수가 많은 데이터에 대한 예측 가능

+ 30명의 유튜버 별 최신 동영상 200개 추가

>> 전처리 과정에서 중복 데이터 삭제

데이터 수집 크롤링(crawling)

동적 크롤링 Selenium 시도

```
# 이미지가 곧바로 로드 되지 않을 때, 20초간 강제로 기다림
img = WebDriverWait(driver, 20).until(EC.presence_of_all_elements_located((By.XPATH, img_xpath)))
if img is None:
    print(idx, 'img is not loaded.')

# 한 페이지에 약 8개 불러오는 데, 동영상 목록을 추가 불러오기 위해 스크롤 내림
if idx % 8 == 0 :
    driver.execute_script('window.scrollTo(0, 1080);')
    time.sleep(2)
    driver.execute_script('window.scrollTo(0, 1080);')
    time.sleep(2)
    driver.execute_script('window.scrollTo(0, 1080);')
    time.sleep(2)

# 주소, 썸네일, 제목, 조회수, 좋아요수, 댓글수
# 주소를 리스트에 저장
video_url = driver.find_element(By.XPATH, url_xpath)
get_url = video_url.get_attribute('href')
print("1", get_url)
url_list.append(get_url)
```



>> 브라우저를 사용하여 연속적으로 접근하기 때문에
데이터의 수집에 한계가 없지만 속도가 매우 느림

유튜브 API를 사용한 크롤링

```
# 키워드에 채널명 넣기
def get_youtube_video(keyword):
    page_token = ''

    while (True):
        response = youtube.search().list(
            type='video', #비디오만 추출
            q = keyword,
            part = "snippet",
            maxResults = 50,
            regionCode = 'KR', #국가는 한국으로 설정
            order = 'date',
            pageToken = page_token).execute()

        for item in response['items']:
            title = item['snippet']['title'] #영상제목
            date = item['snippet']['publishedAt'] #영상날짜
            thumbnail = item['snippet']['thumbnails']['high']['url'] #썸네일
            channel_name = item['snippet']['channelTitle'] #채널이름
            global df
            df = df.append(
                {'channel_name': channel_name, 'title': title, 'date': date,
                 'thumbnail': thumbnail},
                ignore_index=True)

        if ('nextPageToken' in response):
            page_token = response['nextPageToken']
        else:
            break
```

>> 구글에서 제공하는 API 키를 발급받고
YouTube Data API 이용 가능 - 더욱 편리하고 속도 빠름

데이터 수집

데이터 소개

data							
	title	date	viewCount	likeCount	commentCount	subscriber	category
0	전직 UDT. 특수부대 학원에 가다! (숨참기 8분 30초 ㄷㄷ...?) 달려라...	2019-11-23T08:00:12Z	11610712	62590.0	5187.0	3070000	스포츠_운동
1	110만 유튜브 해외여행 클래스... ㄷㄷ	2019-11-04T08:30:02Z	8770917	54094.0	7249.0	3070000	스포츠_운동
2	푸쉬업 '이렇게' 제발 하지마세요 (팔꿈치 박살)	2019-01-20T10:30:00Z	6101839	65041.0	3316.0	3070000	스포츠_운동
3	북한 잠수함에 권총 한 자루를 들고간 UDT 작전 실화 (with 유병호 준위)	2019-09-02T09:00:08Z	5924894	59601.0	6539.0	3070000	스포츠_운동
4	UDT vs 프로 파이터를 (달려라 김계란 Ep2 팀매드 체육관 인턴 1부)	2019-09-28T08:00:11Z	5774965	39466.0	4191.0	3070000	스포츠_운동
...
58733	엡손 프린터 L3156 집에 프린터는 하나는 꼭 필요하더라고요.	2019-07-03T11:00:01Z	57906	455.0	126.0	220000	IT_컴퓨터
58734	삼성 T7 & T7 Touch 이제 외장 SSD도 지문으로 잠금해제~	2020-07-05T11:45:01Z	57215	750.0	194.0	220000	IT_컴퓨터
58735	주행요금 0원! 이제 약속장소까지 전기차로 갑니다 (SOCAR)	2020-12-03T12:00:02Z	56736	753.0	111.0	220000	IT_컴퓨터
58736	2018 New MacBook Pro 디자인 빼고 다 바꿨어요!! (아 올해는 ...	2018-07-14T03:22:33Z	56594	730.0	216.0	220000	IT_컴퓨터
58737	LG그램은 원래 16인치가 맞았던게 아닐까?	2020-12-20T11:30:35Z	55637	791.0	304.0	220000	IT_컴퓨터

58738 rows x 7 columns

‘주제 별 카테고리’ 열을 추가함

58738 행과 [제목, 조회수, 업로드
일자, 댓글수, 좋아요수, 구독자수,
카테고리]

>> 총 7개의 열의 데이터 프레임 구축!

> 58738 rows * 7 columns

03.

Data Pretreatment

Data Pretreatment

제목 – html 태그 decode 처리 By `html_decode`

```
[ ] 1 def html_decode(s):
    2     """
    3     Returns the ASCII decoded version of the given HTML string. This does
    4     NOT remove normal HTML tags like <p>.
    5     """
    6     htmlCodes = (
    7         ('"', '&#39;'),
    8         ('"', '&quot;'),
    9         ('>', '&gt;'),
   10        ('<', '&lt;'),
   11        ('&', '&amp;')
   12    )
   13    for code in htmlCodes:
   14        s = s.replace(code[1], code[0])
   15    return s
   16
```

Data Pretreatment

제목 - 특수 문자 제거 및 띄어쓰기, 맞춤법 교정

```
[ ] def clean_punc(text, punct, mapping):  
    for p in mapping:  
        text = text.replace(p, mapping[p])  
  
    for p in punct:  
        text = text.replace(p, f' {p} ')  
  
    specials = {'\u200b': ' ', '...': ' ... ', '\ufffd': ' ', 'करना': ' ', 'है': ' '}  
    for s in specials:  
        text = text.replace(s, specials[s])  
  
    return text.strip()  
  
[ ] cleaned_corpus = []  
for sent in df['title']:  
    cleaned_corpus.append(clean_punc(sent, punct, punct_mapping))
```

제목 - 특수 문자 제거 By regular expression



Data Pretreatment

제목 - 띄어쓰기 수정 By pykospacing

cf.) meme까지 교정하는 양상을 보여 맞춤법 교정은 진행하지 않음

```
[ ] from pykospacing import Spacing
    spacing = Spacing()
    spacing("김형호영화시장분석가는'1987'의네이버영화정보네티즌10점평에서언급된단어들을지난해12월27일부터올해1월10일까지통계프로그램R과KoNLP패키지로텍스트마이닝하여분석했다.")
```

'김형호 영화시장 분석가는 '1987'의 네이버 영화 정보 네티즌 10점 평에서 언급된 단어들을 지난해 12월 27일부터 올해 1월 10일까지 통계 프로그램 R과 KoNLP 패키지로 텍스트마이닝하여 분석했다.'

```
[ ] review_preprocessed_list=[]
    for i in basic_preprocessed_corpus:
        a = spacing(i)
        review_preprocessed_list.append(a)
```

Data Pretreatment

제목 - 이모지 변환 By emoji module

```
import emoji

print(emoji.demojize("😄"))
print(emoji.demojize("😜"))
print(emoji.demojize("😮"))
```

:grinning_face_with_big_eyes:

:winking_face_with_tongue:

:zipper-mouth_face:

```
[ ] for i in range(len(df['title'])):
    df['title'][i] = emoji.demojize(df['title'][i])
    df['title'][i] = df['title'][i].replace(":", " ")

/usr/local/lib/python3.7/dist-packages/IPykernel_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy
```

```
for i in df['title']:
    print(i)
```

스트리밍 출력 내용이 있어서 마지막 5000줄이 삭제되었습니다.

역대급 비추월에 반해버렸습니다. 팔린 보신적 있으신가요???

퇴치작전)생태계교란종 거대 괴물쥐 "뉴트리아" 포획성공했습니다

의문의택배)이거 도대체 누가 보낸거죠? 바로 연락주세요..

드디어 수백만원에 불개미군단 천적을 입양했습니다=

퇴치작전)역대급인것 같네요.. 황소개구리 퇴치하려 갔는데 크기가 이상합니다=

태어난지 한달된 니모에게 말미잘을 선물해 줘 보았다.

생달을 통해 날고 산에 물어놓으면 생기는 일..

세상에나..반찬통에 닭간 넣었을뿐인데 가져지옥이 될줄이야...

미쳤네미쳤어..아기여우들이 탄생했어요!! 사장님까지 예쁘심2

우리나라에 서식하는 아프리카 괴물물고기를 잡았습니다!!

어릴적 누구나 키워본 추억의 병아리들 기억하시나요? 그런데 크기가....

참게 잡으려다 바위 틈에 엄청난것을 발견했는데...

심쿵주의하세요!! 갓 태어난 아기라쿤이 안기자마자 하는 행동이...뜨흠ㅠ

구슬작전)자칫하면 싹 죽겠는데요? 고인물에 수만마리가 갇혀있다고 제보받고 있는데...

무 믿으시겠지만 직접만큼 부화장으로 "수익마리 씨뽕키" 부화시켰습니다=

Data Pretreatment

feature – 날짜 경과 일수 (현재 – 업로드 날짜) **date_diff**

```
[ ] 1 df['date'] = pd.to_datetime(df['date'])
    2 df['date'] = df['date'].dt.to_period(freq='D')
    3 df['date'] = df['date'].values.astype('datetime64[D]')

[ ] 1 import datetime
    2 day = datetime.datetime(2022,5,22)
    3 df['date_diff'] = day - df['date']

[ ] 1 df['date_diff']=df['date_diff'].apply(str)
    2 df['date_diff'] = df['date_diff'].str.strip(" days 00:00:00")
```


Data Pretreatment

feature – 조회수 구간 나눠주기 **FOR multiclass classification**

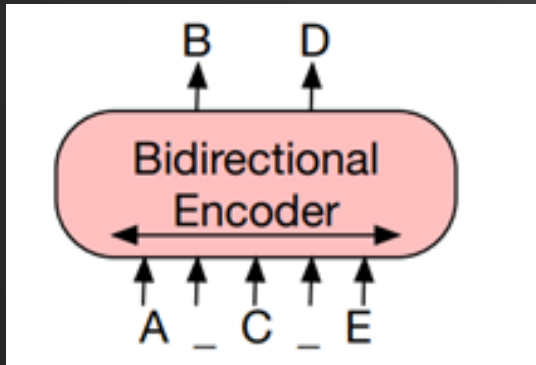
```
[ ] 1 import math # 구간 나눠주기 -> multiclass classification
    2
    3 for index, view in enumerate(df['views_bin']):
    4     if view >=700: df['views_bin'][index] = 700
    5     elif view >=250: df['views_bin'][index] = 250
    6     elif view >=100: df['views_bin'][index] = 100
    7     elif view >=30: df['views_bin'][index] = 30
    8     elif view <30: df['views_bin'][index] = 1
```

- 0이상 ~ 3만 미만: 1
 - 3만 이상 10만 미만 : 30
 - 10만 이상 25만 미만 : 100
 - 25만 이상 70만 미만 : 250
 - 70만 이상 : 700
- 0이상 ~ 3만 미만: 10933개
 - 3만 이상 10만 미만 : 10793개
 - 10만 이상 25만 미만 : 10608개
 - 25만 이상 70만 미만 : 11942개
 - 70만 이상 : 10178
- => **WELL BALANCED**

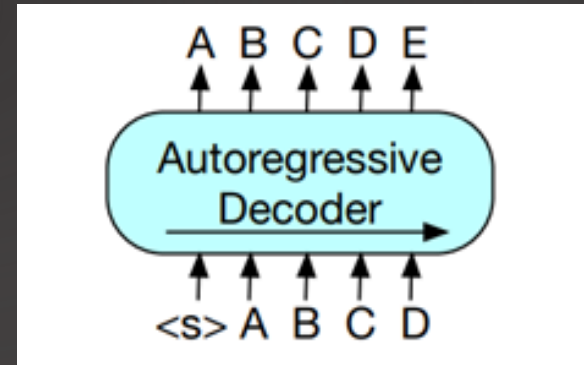
04. Modeling

About BART

BART : BERT와 GPT 모델을 연결해 두 모델의 단점을 보완한 모델

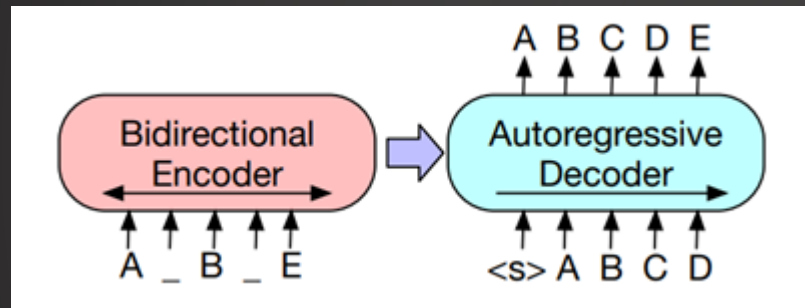


기존의 BERT 모델
-양방향 인코더 모델
(Bidirectional Encoder)

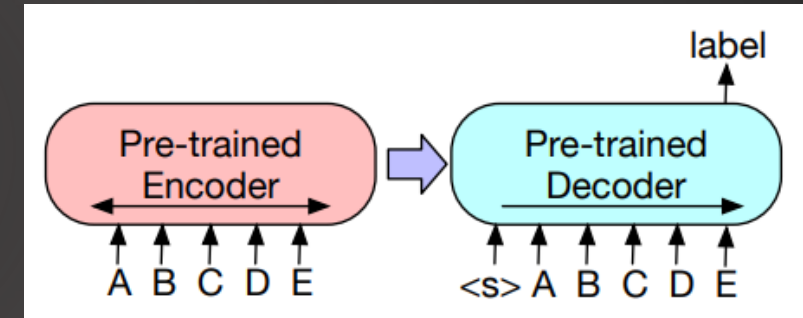


기존의 GPT 모델
-자기회귀성 디코더 모델
(Autoregressive Decoder)

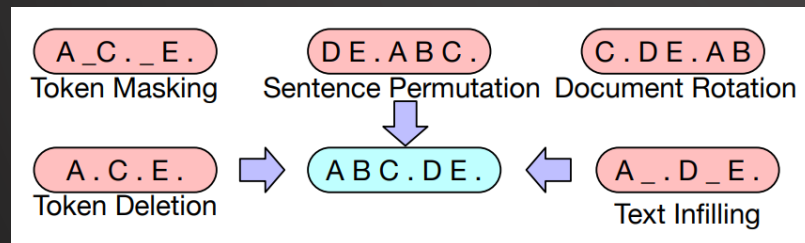
About BART



[BART]



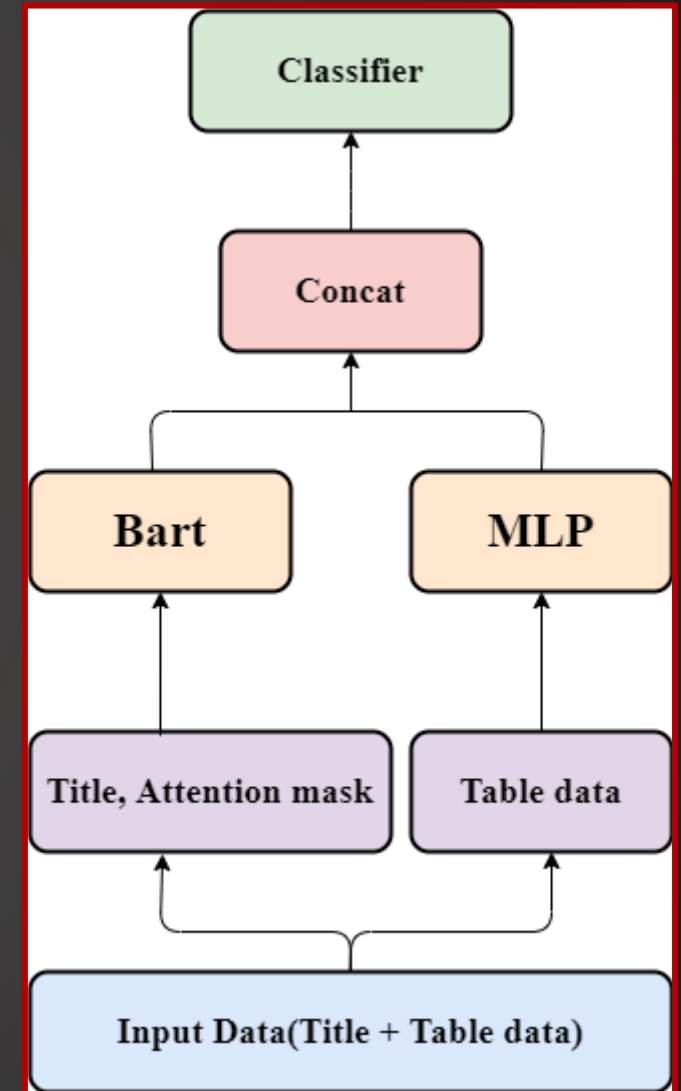
- 본 프로젝트에서는 BART의 final hidden state만 떼어서 classification에 사용
- BART만 이용할 시, final hidden state를 직접 linear classifier에 전달해줌



[BART에 사용가능한 noise 종류]

Model Architecture Modeling

- 텍스트 데이터 + 정형 데이터를 모두 활용하여 예측하는 모델링 진행
- 텍스트 데이터의 처리 : KoBART 모델
- 정형 데이터 : Multi Layer Perceptron



Model Architecture Modeling

Dataset Class 생성

1. kobart tokenizer 사용, input_ids, attention_mask를 키로 받음
2. 정형데이터 table 형태로 저장
3. 최종 dictionary 구축
 - train, validation : input_ids, attention_mask, table, labels(정답)
 - Test : 라벨 제외

```
class Train_Val_Dataset(Dataset):  
  
    def __init__(self, df, max_seq_len=64):  
        self.data = df  
        self.tokenizer = get_kobart_tokenizer()  
        self.max_seq_len = max_seq_len  
  
    def __getitem__(self, index):  
  
        record = self.data.iloc[index]  
        #text, table, label = str(record["preprocessed_title"]), record[['subscriber', 'date_diff', 'category']], int(record["views_bin"])  
        text, table, label = str(record["preprocessed_title"]), record[['category', 'likeCount_scaled', 'date_diff_scaled']], int(record["views_bin"])  
        tokens = (  
            [self.tokenizer.bos_token]  
            + self.tokenizer.tokenize(text)  
            + [self.tokenizer.eos_token]  
        )  
        encoder_input_id = self.tokenizer.convert_tokens_to_ids(tokens)  
        attention_mask = [1] * len(encoder_input_id)  
        if len(encoder_input_id) < self.max_seq_len:  
            while len(encoder_input_id) < self.max_seq_len:  
                encoder_input_id += [self.tokenizer.pad_token_id]  
                attention_mask += [0]  
        else:  
            encoder_input_id = encoder_input_id[: self.max_seq_len - 1] + [  
                self.tokenizer.eos_token_id  
            ]  
            attention_mask = attention_mask[: self.max_seq_len]  
        return {  
            "input_text": np.array(encoder_input_id, dtype=np.int_),  
            "input_table": np.array(table, dtype=np.int_),  
            "attention_mask": np.array(attention_mask, dtype=float),  
            "labels": np.array(label, dtype=np.int_),  
        }
```

Model Architecture Modeling

4. DataLoader 생성 : batch_size = 24

```
1 train_data = Train_Val_Dataset(train_dataset)
2 val_data = Train_Val_Dataset(val_dataset)
3 test_data = TestDataset(test_dataset)
4
5 # batch_size 만큼 데이터 분할
6 train_dataloader = DataLoader(train_data,
7                               batch_size=TRAIN_BATCH_SIZE,
8                               shuffle=True)
9
10 val_dataloader = DataLoader(val_data,
11                             batch_size=TRAIN_BATCH_SIZE,
12                             shuffle=False) #drop_last=True
13
14 test_dataloader = DataLoader(test_data,
15                              batch_size=TEST_BATCH_SIZE,
16                              shuffle=False)
17
```


Model Architecture Modeling

kobart embedding -마지막 layer 값 추출

1. 텍스트 데이터 : 768차원으로

embedding 차원 구성

정형데이터 : 마찬가지로 768차원 가지도록

table_extractor 신경망 구성

2. 둘을 concat

3. 조회수 간격 예측 classifier 클래스 구성

```
1 class FinalModel(nn.Module):
2     def __init__(self, table_dim, num_label): # table_dim은 정형데이터의 column 수라 생각하면 됨
3         super(FinalModel, self).__init__()
4         self.text_extractor = bart
5         self.table_extractor = nn.Sequential(nn.Linear(table_dim, 768//2),
6                                             nn.ReLU(),
7                                             nn.Linear(768//2, 768),
8                                             nn.ReLU())
9         self.classifier = Classifier(768 * 2, num_label)
10
11     def forward(self, x):
12         text_input = x['input_text']
13         attention_input = x['attention_mask']
14         table_input = x['input_table'].float() # (batch_size, table_dim)
15         text_feature = self.text_extractor(text_input, attention_input)['last_hidden_state'] # (batch_size, 768)
16         text_feature = text_feature.view(text_feature.size(0), -1, text_feature.size(-1))[:, -1, :]
17         table_feature = self.table_extractor(table_input) # (batch_size, 768)
18         fusion_input = torch.cat([text_feature, table_feature], dim = 1) # (batch_size, 768 * 2)
19         output = self.classifier(fusion_input)
20         return output
```

```
1 class Classifier(nn.Module):
2     def __init__(self, input_dim, output_dim): # input_dim
3         super(Classifier, self).__init__()
4         self.input_dim = input_dim
5         self.output_dim = output_dim
6         self.linear = nn.Sequential(nn.Linear(input_dim, input_dim//2),
7                                     nn.ReLU(),
8                                     nn.Linear(input_dim//2, input_dim//4),
9                                     nn.ReLU(),
10                                    nn.Linear(input_dim//4, output_dim))
11
12     def forward(self, x):
13         output = self.linear(x)
14         return output
```


Model Training Conclusion

kobart embedding -마지막 layer 값 추출

1. 이모지를 넣은 것 / 넣지 않은 것,
2. 정형데이터(date_diff, 좋아요수, 카테고리)
열 조합 변경
3. Only 텍스트 데이터 + KoBart모델
=> 51% 성능

[한계점]

1. 자연어처리 데이터와 정형데이터를 함께 활용하는 모델은 여전히 자연어처리 모델만을 활용하는 것보다 좋은 성능을 내기 어렵다.
2. 유튜브 데이터 자체가 여러 변수가 존재하며, 누적데이터의 특성을 가지기 때문에 예측이 어렵다.
3. 특정 라벨이 주어진 것이 아니라 임의로 y값의 기준을 설정하여 label을 만들었기 때문에 이 label의 신뢰도가 부족하다. 즉 label이 적절하게 나뉘지지 않아 예측에 어려움이 있다.

KU독 & 좋아요 느낀점

1. 우리의 삶에 있어서 밀접한 내용을 다루고 이를 분석함으로써 데이터분석의 실제 적용 사례를 직접 경험해 봄
2. 정형 데이터 분석을 경험할 수 있었으며 직접 모델을 구성함으로써 딥러닝에 대한 이해도를 높일 수 있었음
3. 여러 한계점들로 인해 처음 계획했던 썸네일을 이용한 분석을 진행하지 못했던 것이 가장 아쉬웠음

Please Subscribe And Like

