썸네일과 영상 제목을 활용한

유튜브 카테고리 분류

3조

염예진 이유경 이영송 김민석 서지완

[목차]

1. 프로젝트 선정 이유

딥러닝 팀 프로젝트의 주제를 선정하는 과정에서 수업에서 배운 이미지, 텍스트 데이터 두가지 모두 활용해보고 싶다는 공통 의견이 있었습니다. 이미지와 텍스트 특성을 모두 가지고 있는 데이터를 생각해봤더니 평소에 자주 접할 수 있는 유튜브 영상의 이미지와 영상 제목이 떠올랐습니다. 유튜브 컨텐츠는 해당 컨텐츠가 어떤 카테고리에 속하는지 유튜브에서 만든 알고리즘에 따라 카테고리 분류가 자동으로 지정되는 것으로 판단했습니다. 하지만 실제로 유튜브에 접속해서 영상을 확인해보니 동영상 컨텐츠의 카테고리 분류가 올바르게 되지 않은 경우가 다소 발견되었습니다. 해당 수업을 들으며 이미지를 이용한 분류, 텍스트를 이용한 분류 즉, 이미지와 텍스트를 따로 사용해 분류를 하는 예제를 접해왔습니다. 저희는 이 두 가지의 특성을 추출, 결합해서 딥러닝 모델을 구축한다면, 좀 더 정확한 카테고리 분류를 할 수 있다고 생각했습니다. 따라서 3조의 주제는 "썸네일 이미지와 영상 제목을 활용한 유튜브 카테고리 분류"로 프로젝트를 진행하게 되었습니다.

2. 데이터 수집

"cooking, economy, game, movie, pets, politics, sports" 총 7개의 카테고리를 직접 지정했습니다. 카테고리별 데이터셋 개수가 약 5000개가 되도록 각각 채널을 선정한 뒤, "https://www.youtube.com/"에서 데이터를 수집했습니다. 조회수 예측, 카테고리 분류 2가지 주제로 데이터 분석을 진행할 예정이었기에 "채널명, 동영상 제목, 카테고리, 구독자 수, 썸네일 이미지, 영상 길이, 업로드 날짜, 조회수, 좋아요 수, 싫어요 수"를 수집했습니다. 하지만, 데이터 전처리 부분에서 시간이 많이 소요되었기에 "채널명, 동영상 제목, 카테고리, 썸네일 이미지" 총 4가지 column만 사용하기로 했습니다.

약 30,000개의 데이터를 수집했는데, 학습이 너무 느려서 데이터를 약 13,000개로 축소했습니다. 하지만 축소하는 과정에서 특정 카테고리에서만 데이터를 제거해서 카테고리별 데이터 개수가 비대칭이었습니다. 프로젝트 마지막 단계에 발견했기 때문에 해당 내용은 후에 다시 보완하기로 했습니다. 그리고 텍스트와 이미지를 함께 사용해 모델 평가를 하기 위해 train과 test를 8:2로 나누었습니다.

3. 데이터 전처리

(1) 텍스트 데이터

토큰화를 진행하기 전, 특수문자, 이모티콘, 영어, 숫자를 노이즈로 판단하고 제거했습니다. 영상 제목에 한국어만 사용한 이유는 한국어 영상 제목과 영어 영상 제목은 동일한 의미이기 때문에 한국어만 사용하기로 결정했습니다. 또한, 코퍼스 고빈도어 상위 100개, 채널명, 눈으로 확인할 수 있는 단어들 3가지를 포함해 불용어사전을 정의했습니다. "https://bab2min.tistory.com/544"에서 정리해둔 코퍼스 빈도수 상위 100개를 사용했습니다.

영상 제목은 okt 명사, kkma 명사, soynlp의 word기반, soynlp의 noun기반 총 4가지의 형태소분석기를 이용했습니다. Word embedding은 skip-gram 방식 및 window를 4로 지정한 word2vec, embedding된 N-gram의 합인 fasttext 2가지 방식을 사용했습니다. 따라서 총 8가지 word embedding데이터로 1차 성능 평가를 통해 최종 분류모델에 사용할 형태소 분석기 및 word embedding방식을 정할 예정입니다.

(2) 이미지 데이터

"https://i.ytimg.com/vi/al52wJgMGyI/hqdefault.jpg?sqp=oaymwEZCNACELwBSFXyq4qpAwsIARUAAIhCGAFwAQ==&rs=AOn4CLB4oUvLbTvMbUGV0BCQE-95N_oXsw" 썸네일 이미지의 데이터입니다. Keras의 preprocessing을 이용해서 shape를 축소하고 컬러 채널을 정규화했습니다. 썸네일 이미지 데이터의 사이즈는 (360, 480, 3)에서 (64, 64, 3)으로 축소했습니다.

4. 모델링

(1) 텍스트 모델링

총 8가지의 단어 임베딩 데이터로 모델을 평가했습니다. 모델은 Logistic Regressin, SVC 두 가지를 사용했습니다. 두 모델의 정확도를 정리한 표는 아래와 같습니다.

		Accuracy	
형태소분석기	단어 임베딩	Logistic Regression	SVC
Soynlp(word)	Word2vec	0.738043	0.774467
	Fasttext	0.766740	0.791391
Soynlp(noun)	Word2vec	0.741316	0.781596
	Fasttext	0.78899	0.810791
Kkma	Word2vec	0.807678	0.838317
	Fasttext	0.840901	0.860465
okt	Word2vec	0.825279	0.847584
	Fasttext	0.843494	**0.858364**

Kkma_fasttext의 SVC 정확도가 가장 높았지만, Kkma 형태소 분석기의 속도가 상대적으로 느린 편이었습니다. 따라서 다음으로 높은 정확도를 보이는 Okt의 Word2vec, fasttext를 최종 모델의 특성 추출값으로 사용했습니다.

(2) 이미지 모델링

CNN모델의 경우, Conv2D input 사이즈 (64, 64, 3)로 시작하여 3개의 레이어와 활성함수, pooling, dropout을 적절히 쌓았습니다. Activation은 Relu로 쌓았고, 2x2 Maxpooling2D, 25%의 Dropout의 층으로 모델링했습니다. 그 결과 정확도가 약 0.81로 도출되었습니다.

전이학습 (VGG16)의 경우, input 사이즈 (64, 168, 3)으로 시작하여 1개의 vgg16 레이어와 Flatten, 3개의 Dense 층을 쌓았습니다. VGG16 층을 쌓기 전, trainable=False로 설정했습니다. 그 결과 정확도가 약 0.82로 도출되었습니다.

(3) 텍스트 + 이미지 모델링

영상제목으로부터 특성을 추출하고, 이미지로부터 특성을 추출했습니다. 영상 제목으로부터 특성을 추출한 결과, Okt의 Word2vec와 Okt의 Fasttext의 shape는 문장의 개수 13625개, 차원 수 100개의 형태로 구성되어 있습니다. 영상 이미지로부터 특성을 추출한 결과, CNN모델 특성 추출의 shape는 (13685, 7200)이고, VGG16모델 특성 추출 shape는 (13685, 5120)으로 구성되어 있습니다.

VGG16 + Word2vec, VGG16 + Fasttext, CNN + Word2vec, CNN + Fasttext 총 4번의 모델링을 실시했습니다. Dropout은 40%, Dense층의 Regularizer_L2는 0.01, activation은 Relu, 모델 컴파일할 때 optimizer은 nadam을 사용했습니다. 모델의 Loss와 Accuracy는 다음과 같습니다.

	Loss	Accuracy
VGG+w2v	0.852	87.52%
VGG+Fxt	0.78	88.74%
CNN+Fxt	0.7706	83.56%
CNN+w2v	0.8205	83.70%

5. 결과 해석

7가지의 카테고리 중, 게임과 스포츠 카테고리 분류를 혼동하는 경우가 잦았습니다. 해당 카테고리를 혼동하는 이유를 생각해보았을 때, 스포츠 게임을 하는 컨텐츠라면 우리의 분류 모델이 스포츠 카테고리인지 게임 카테고리인지 혼동할 수도 있다고 생각했습니다. 데이터가 더 많거나 지속적으로 학습을 한다면 더욱 정확한 분류가 있을 수 있다고 생각했습니다.

6. 보완점

(1) 데이터셋 축소 과정

데이터를 30,000개를 축소하는 과정에서 실수가 있었기에 카테고리별 데이터의 개수가 비대칭이었습니다. 코드를 수정해서 카테고리별 2,000개씩 랜덤 추출로 30,000개에서 약 14,000개로 데이터셋을 다시 축소했습니다.

(2) 카테고리 예측

이미지, 텍스트만 이용해서 분류했을 때 각 카테고리별 precision과 이미지와 텍스트의 특성을 결합해서 분류했을 때 precision을 확인해보기로 했습니다. 더불어 분류가 잘못된 이미지나 텍스트에 대해 예시를 몇 가지 찾아보고, 이유가 무엇인지 생각해보기로 했습니다. (2-1), (2-2), (2-3)은 카테고리별 precision, recall, f1-score, accuracy를 정리한 표이고, (2-4)는 분류가 잘못된 이미지와 텍스트에 대한 예시를 정리했습니다.

(2-1) 이미지만 이용해서 분류하기 VGG16

	precision	recall	F1-score	Accuracy
Cooking	0.79	0.84	0.81	
Economy	0.77	0.76	0.76	
Game	0.58	0.70	0.63	
Movie	0.67	0.59	0.63	0.72
Pets	0.81	0.84	0.82	
Politics	0.72	0.69	0.70	
Sports	0.678	0.60	0.64	

(2-2) 텍스트만 이용해서 분류하기 SVC

	precision	recall	F1-score	Accuracy
Cooking	0.92	0.89	0.91	
Economy	0.93	0.87	0.90	
Game	0.56	0.82	0.66	
Movie	0.84	0.78	0.81	0.82
Pets	0.89	0.75	0.82	
Politics	0.90	0.85	0.87	
Sports	0.84	0.77	0.80	

(2-3) 텍스트 + 이미지 결합해서 분류하기

	precision	recall	F1-score	Accuracy
Cooking	0.94	0.91	0.93	0.86
Economy	0.90	0.92	0.91	
Game	0.81	0.79	0.80	
Movie	0.82	0.80	0.81	
Pets	0.95	0.91	0.93	
Politics	0.83	0.89	0.86	
Sports	0.77	0.79	0.78	

(2-4) 분류가 잘못된 예시

Case 1 : 이미지로만 분류했을 때 카테고리 잘못 예측한 경우

이미지로만 분류했을 때 카테고리를 잘못 예측한 경우는 2781개 중 450개입니다. 아래 사진은 정치를 게임으로 예측했거나, 스포츠를 정치로 예측한 경우입니다.

Case 2 : 텍스트로만 분류했을 때 카테고리 잘못 예측한 경우

텍스트로만 분류했을 때 카테고리를 잘못 예측한 경우는 2781개 중 231개입니다. 아래 사진은 영화를 애견으로 예측했거나, 애견을 게임으로 예측한 경우입니다.

Case 3 : 이미지와 텍스트 종합해서 분류했을 때 카테고리 잘못 예측한 경우

텍스트로만 분류했을 때 카테고리를 잘못 예측한 경우는 2781개 중 124개입니다. 아래 사진은 실제는 스포츠 카테고리인데 텍스트는 게임으로, 이미지는 애견으로, 종합했을 때는 정치로 예측했으며, 실제는 스포츠인데 세가지 전부 게임으로 예측한 예시입니다.

7. 결론

저희는 본 프로젝트를 통해 썸네일 이미지와 영상 제목을 활용한 유튜브 카테고리 분류 모델링을 실시했습니다. 이미지만 이용해서 VGG16, CNN으로 모델링을 했을 때, 이미지로만 카테고리 분류를 정확하게 하지 못한다는 어려움이 있었습니다. 텍스트만 이용해서 SVC, LR으로 모델링을 했을 때, 전처리 과정에서 단어 사전을 더욱 정확하게 만들지 못해서 토큰화가 잘 이루어지지 않았다는 점, 제목에 핵심 내용이 없어서 텍스트만으로는 카테고리 분류 정확도가 낮다는 어려움이 있었습니다. 이를 보완하고자 텍스트와 이미지의 특성값을 결합해 4가지 모델링을 실시했습니다. 그 결과, 이미지, 텍스트를 따로 사용해서 카테고리를 분류했을 때보다 정확도가 더 높아진 이점이 있었습니다. 하지만 사람이 눈으로 확인했을 때, 카테고리 분류를 명확하게 할 수 있지만 세세한 부분을 조정하지 못해서 모델링이 예측을 하지 못한 아쉬운 점도 있었습니다. 이러한 부분은 영화 채널 자체의 정보나 기타 추가 요인을 통해 모델을 보완할 수 있을 것으로 생각합니다. 그리고 총 7개의 카테고리를 지정했는데, 채널 특성상 하나의 카테고리에 속하는 경우는 드물다고 생각합니다. 따라서 카테고리 지정, 분류 방식도 변화를 준다면 모델을 보완할 수 있을 것이라고 생각합니다.