



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

2022년 2월

박사학위논문

# 텍스트-이미지 임베딩 기반의 영상 콘텐츠 유사도 측정 방법

조선대학교 대학원

컴퓨터공학과

홍 택 은

# 텍스트-이미지 임베딩 기반의 영상 콘텐츠 유사도 측정 방법

A Method of Video Contents Similarity Measurement  
based on Text-Image Embedding

2022년 2월 25일

조선대학교 대학원

컴퓨터공학과

홍 택 은

# 텍스트-이미지 임베딩 기반의 영상 콘텐츠 유사도 측정 방법

지도교수 김 판 구

이 논문을 공학박사학위신청 논문으로 제출함.






2021년 10월

조선대학교 대학원

컴퓨터공학과

홍 택 은

## 홍택은의 박사학위논문을 인준함

심사위원장	조선대학교	교수	양 희 덕	
심사위원	조선대학교	교수	정 호 엽	
심사위원	William Paterson University	교수	임 기 호	
심사위원	가천대학교	교수	최 창	
심사위원	조선대학교	교수	김 판 구	

2022년 1월

조선대학교 대학원

# 목 차

## ABSTRACT

<b>제1장 서론</b> .....	<b>1</b>
제1절 연구 배경 및 목적 .....	1
제2절 연구 방법 및 내용 .....	4
<b>제2장 관련 연구</b> .....	<b>6</b>
제1절 콘텐츠 추천 관련 기존 연구 .....	6
제2절 텍스트-이미지 임베딩 .....	10
1. 텍스트 임베딩 .....	13
2. 이미지 임베딩 .....	21
제3절 Triplet Ranking Loss .....	25
<b>제3장 영상 콘텐츠 유사도 측정 방법</b> .....	<b>28</b>
제1절 제안하는 텍스트-이미지 임베딩 모델 .....	28
1. 텍스트 네트워크 .....	29
2. 이미지 네트워크 .....	33
3. 텍스트-이미지 네트워크 .....	35
제2절 개선된 Triplet Ranking Loss .....	38

제4장 실험 및 결과 .....	40
제1절 데이터 셋 .....	40
제2절 텍스트-이미지 임베딩 모델의 성능 평가 .....	45
1. 모델 구성을 위한 텍스트와 이미지 임베딩 성능 평가 .....	47
2. 제안한 텍스트-이미지 임베딩 모델의 성능 평가 .....	50
제3절 텍스트-이미지 임베딩 시각화 및 추천 방법 .....	57
제5장 결론 및 제언 .....	59
참고문헌 .....	63

## 그림 목 차

그림 1. 메타데이터를 강요하는 OTT 화면의 예 .....	8
그림 2. 화제성이 높은 콘텐츠를 추천하는 OTT 화면의 예 .....	9
그림 3. 일반적인 텍스트-이미지 임베딩 모델 .....	10
그림 4. Word2Vec에서 중심 단어와 주변 단어 구성의 예(window size=2) .....	13
그림 5. Word2Vec의 CBOW(왼쪽)과 Skip-Gram(오른쪽)의 구조 .....	14
그림 6. Recurrent Neural Network(RNN)의 구조 .....	15
그림 7. Long-Short Term Memory(LSTM)의 구조 .....	16
그림 8. BERT의 Masked Language Model(MLM) 구조 .....	18
그림 9. BERT의 Next Sentence Prediction(NSP) 구조 .....	19
그림 10. Residual block의 구조 .....	21
그림 11. InceptionV3 모듈의 구조 .....	22
그림 12. EfficientNet의 구조 .....	23
그림 13. Triplet Ranking Loss의 학습 과정 .....	25
그림 14. Triplet Ranking Loss의 업데이트 경우의 수 .....	26
그림 15. 제안하는 Text-Image Embedding 모델의 구조 .....	28
그림 16. 텍스트-이미지 모듈 .....	35
그림 17. 텍스트-이미지 네트워크 .....	35
그림 18. 기존 Triplet Ranking Loss의 그래프 .....	39
그림 19. 제안하는 Triplet Ranking Loss의 그래프 .....	39
그림 20. 타겟 영화와 유사한 영화의 수집 방법 .....	41
그림 21. 영화ID-영화 제목 데이터셋의 일부 .....	42
그림 22. 영화ID-줄거리 데이터셋의 일부 .....	42
그림 23. 영화ID 폴더-포스터, 스틸컷 데이터셋의 일부 .....	42
그림 24. validation 셋을 이용한 모델 성능 개선 방법 .....	43
그림 25. 실험에 사용하는 데이터셋의 구성 .....	44
그림 26. Recall 설명을 위한 Confusion Matrix .....	45
그림 27. 학습하지 않는 텍스트-이미지 임베딩 방법 .....	50



그림 28. K가 3인 K-NN의 의사결정 예 .....	51
그림 29. Contrastive Loss의 학습 예 .....	52
그림 30. VSE++ 모델의 구조 .....	53
그림 31. 제안하는 방법과 기존 방법의 성능 평가(Recall@20) .....	55
그림 32. 텍스트-이미지 임베딩의 시각화 결과 .....	57

## 표 목 차

표 1. 사전학습 된 Word2Vec으로 문장을 임베딩하는 과정 .....	29
표 2. 기본적인 RNN으로 문장을 임베딩하는 과정 .....	30
표 3. 기본적인 LSTM으로 문장을 임베딩하는 과정 .....	31
표 4. 사전학습된 KoBERT으로 문장을 임베딩하는 과정 .....	31
표 5. 사전학습된 ResNet151, InceptionV3, EfficientNet b7을 이용한 이미지 임베딩 .....	33
표 6. 각 모듈의 kernel과 pooling의 설정 .....	36
표 7. 타겟 영화와 유사한 영화를 구분하기 위해 정의한 장르 .....	41
표 8. 학습에 사용한 하이퍼파라미터 .....	47
표 9. 각 임베딩 방법에 따른 Recall@1의 결과 .....	47
표 10. 각 임베딩 방법에 따른 Recall@10의 결과 .....	48
표 11. 각 임베딩 방법에 따른 Recall@20의 결과 .....	48

# ABSTRACT

## A Method of Video Contents Similarity Measurement based on Text-Image Embedding

Taekeun Hong

Advisor : Prof. Pankoo Kim, Ph.D

Department of Computer Engineering

Graduate School of Chosun University

Owning smart devices, such as smartphones and tablet PCs has been steadily increasing. Further, increased media-based content viewing during the COVID-19 pandemic period enhanced smart device handling time. Particularly, Over-The-Top video service (OTT) viewing has increased significantly. Several OTT viewing-related analyses are being conducted.

The content recommendation system is the most important element that influences the user satisfaction and continued subscription to an OTT service. Users might exhibit inconvenience due to unclear and noncomprehensive topic or category viewing suggestions. Therefore, gaining and maintaining user trust and satisfaction with the content recommendations is important.

However, most of the content recommendation systems require user and content metadata to provide the content or trending recommendations. Even most studies related to content recommendation systems analyze user and content metadata. Analyzing the video content as the recommendation basis is inadequate.

Therefore, we propose a text-image embedding-based video content similarity assessment method for direct content analysis. General text-image embedding method embeds and combines text and image features for accurate depiction of the embedded class relationships. General text-image embedding is primarily used for image-text retrieval, image captioning, and visual question answering.

However, the content analysis studies that derive class relationship similarities through text-image embedding are insufficient. Therefore, we propose a text-image embedding-based video content recommendation method.

The proposed text-image embedding model consists of text, image, and text-image networks. The individual text and image networks embed text and images, while the text-image network combines the text and image embedding and learns the video content relationships.

The text network embeds the input text and uses a pre-trained model, such as Word2Vec, KoBERT, basic RNN, or LSTM. The image network embeds the input images and employs a pre-trained model, such as ResNet, InceptionV3, and EfficientNet. Experiments were conducted using each text and image embedding models, and the best method was selected. The KoBERT and EfficientNet exhibited the best embedding of the text and images, respectively. Therefore, these methods were used to embed text and image in corresponding networks.

The text-image network combines the embedded text and images and learns the video content similarity. This network consists of a flattened layer, fully-connected layer, L2-normalization, and five modules composed of 1D convolution, 1D max pooling, and dropouts. The content similarity in the proposed text-image network was achieved using an improved triplet ranking loss.

The triplet ranking loss aims to predict the relative distance of the given data by learning from the vectorized text or images. It learns the data similarity (close to) or dissimilarity (far from) to the target, by constituting target-like and target-unlike datasets.

As the existing triplet ranking loss updates when the distance value is positive, negative numbers do not affect the learning. If zero distance occurs frequently, the update may not work well and result in improper learning. As the negative numbers could be significant, they could be included in the learning.

Therefore, we propose an improved triplet ranking loss for training the proposed text-image embedding model. The improved triplet ranking loss employs a nonlinear sigmoid function as the existing method. The equation was modified to enable the triplet ranking loss update with a proper value, even when using the sigmoid function with a negative distance value.

Comparative evaluation was performed with metric learning-based loss functions, using the learning method as in the improved triplet ranking loss method. Contrastive loss, the existing triplet ranking loss, and VSE++ functions were used in the comparative evaluation. K-NN serves the same purpose as metric learning in machine learning. Therefore, it was included in the comparison.

The results showed that training the proposed text-image embedding model with the improved triplet ranking loss method achieved superior performance than the existing methods. Therefore, the text-image embedding structural model and the improved triplet ranking loss model are suitable for measuring the video content similarity and learning, respectively.

The dimension reduction and visualization were examined and genre-based

clustering was observed. Therefore, the content recommendations can be user preference genre-based. Therefore, presenting the underlying rationale for the category and topic recommendations, and improving user satisfaction is possible with OTT services.

# 제1장 서론

## 제1절 연구 배경 및 목적

스마트폰과 태블릿PC와 같은 스마트 기기의 보유율이 꾸준히 증가하고 있으며, 코로나의 여파로 인해 스마트 기기를 이용한 미디어 콘텐츠 이용률과 이용 시간도 크게 증가하고 있다[1]. 특히 동영상제공서비스(OTT)를 이용하는 비율이 크게 증가했으며, 주로 YouTube, NETFLIX, 왓챠, Wavve, NAVER TV 등을 이용하는 것으로 나타났다[2]. OTT 사용자의 증가에 따라 OTT를 이용할 때 체감하는 다양한 요소를 분석하는 것이 중요해졌으며, 이와 관련된 연구들이 활발하게 수행되고 있다.

특히, 이용 만족도와 지속 사용 의도에 미치는 요소를 분석하는 연구가 수행되고 있다. OTT 사용자의 이용 만족도와 지속 사용 의도에 영향을 끼치는 요소는 콘텐츠 다양성, 요금제 적절성, 추천 시스템, N스크린 서비스, 몰아보기 기능 등이 있다[3]. OTT의 콘텐츠 추천 시스템에서 저항에 영향을 미치는 요인에 관한 연구가 수행되었으며, 그 결과 추천 정확성에 따라 이용자의 만족도에 긍정적인 평가와 신뢰감을 증가시켜 지속적인 이용에 긍정적인 영향을 줄 수 있는 것으로 밝혀졌다[4].

그러나 대부분의 추천 시스템은 사용자의 메타 데이터를 강요하여 추천 서비스를 제공하거나 화제성이 높은 콘텐츠를 추천한다는 문제점이 존재한다. 추천 서비스에서 추천하는 카테고리 명의 생성 기준과 카테고리 주제가 명확하지 않고 포괄적인 경우 사용자는 불편함을 느낄 수 있기 때문에 콘텐츠 추천에 대한 근거를 통해 신뢰감과 이용 만족도를 이끌어 내야 한다[5].

이와 같은 행태는 콘텐츠를 직접적으로 분석하고 그 내용을 기반으로 추천하지 않기 때문이며, 콘텐츠를 구성하는 데이터를 분석하여 이를 기반으로 사용자

에게 서비스를 제공하는 것은 매우 중요하다.

기존의 추천 시스템은 크게 협업 필터링과 콘텐츠 기반 필터링으로 나눌 수 있다. 하지만 대부분의 연구가 협업 필터링과 관련된 연구이며, 콘텐츠 기반 필터링의 연구라고 하더라도 대부분이 콘텐츠 자체의 분석이 아닌 콘텐츠의 메타 데이터를 이용하는 경우가 대부분이다.

텍스트와 이미지 정보를 함께 분석하기 위한 방법으로 텍스트-이미지 임베딩 방법을 이용한다. 텍스트-이미지 임베딩은 텍스트와 이미지를 결합하기 위한 방법으로 임베딩 된 콘텐츠 사이의 관계를 잘 표현하는 것을 목표로 한다.

따라서 본 논문에서는 콘텐츠를 직접적으로 분석하기 위한 방법으로 콘텐츠를 구성하는 텍스트와 이미지를 함께 이용하여 콘텐츠 간 유사도를 측정하고 추천하기 위한 텍스트-이미지 임베딩 모델과 개선된 Triplet Ranking Loss를 제안한다.

텍스트-이미지 임베딩은 일반적으로 텍스트 네트워크와 이미지 네트워크, 이를 결합하기 위한 텍스트-이미지 네트워크로 구성된다[6]. 텍스트 네트워크는 입력되는 텍스트를 임베딩하기 위한 방법으로 Word2Vec[7], Recurrent Neural Network(RNN)[8], Long-Short Term Memory(LSTM)[9], Bidirectional Encoder Representations from Transformers(BERT)[10] 등으로 구성 될 수 있다. 이미지 네트워크는 입력되는 이미지의 특징을 추출하기 위한 방법으로 ResNet[11], InceptionV3[12], EfficientNet[13] 등으로 구성할 수 있다. 마지막으로 텍스트와 이미지를 결합하기 위한 텍스트-이미지 네트워크는 분류기, Softmax[14], Triplet Ranking Loss[15, 23] 등으로 구성될 수 있다.

Triplet Ranking Loss는 벡터 간 관계를 고려하는 Metric Learning을 기반으로 주어진 데이터의 상대적인 거리를 예측하기 위해 벡터화된 텍스트나 이미지로부터 학습하는 것을 목적으로 한다. 타겟 데이터와 유사한 데이터, 유사하지 않은 데이터로 구성하여 유사한 데이터는 타겟과 가깝도록, 유사하지 않은 데이터는 타겟과 멀도록 학습한다.



기존 Triplet Ranking Loss는 거리 값이 음수인 경우 학습에 영향을 미치지 않기 때문에 업데이트가 잘 되지 않고 학습이 정상적으로 되지 않을 수 있다. 하지만 음수도 분명히 어떠한 의미를 가지고 있기 때문에 학습에 포함하는 것이 적절하므로 개선된 Triplet Ranking Loss를 제안하여 학습에 이용한다.

## 제2절 연구 방법 및 내용

본 논문은 영상 콘텐츠 유사도 측정을 위한 텍스트-이미지 임베딩 모델과 개선된 Triplet Ranking Loss를 제안하며, 이를 통해 도출한 텍스트-이미지 임베딩을 통해 콘텐츠 간 유사도를 측정하는 방법을 소개한다.

제안하는 방법은 텍스트-이미지 임베딩 모델과 개선된 Triplet Ranking Loss로 텍스트-이미지 임베딩 모델은 텍스트 네트워크, 이미지 네트워크, 이를 결합하기 위한 텍스트-이미지 네트워크로 구성된다. 개선된 Triplet Ranking Loss는 텍스트-이미지 네트워크에서 텍스트와 이미지 임베딩을 학습하여 콘텐츠의 유사성 학습에 이용한다.

본 논문에서 분석하는 콘텐츠는 영화를 대상으로 하고 있으며, 텍스트 데이터로 줄거리를 사용하고 이미지 데이터로 스틸컷과 포스터를 사용한다. 제안하는 텍스트-이미지 임베딩 모델을 통해 영화 콘텐츠를 분석하여 영화 간 유사함이 반영된 임베딩을 도출하고 이를 통해 사용자에게 추천하기 위한 방법에 대해 설명한다.

제안한 텍스트-이미지 모델의 임베딩 결과를 통해 제안한 방법을 평가하고 그 결과를 통해 타당성을 검증한다. 타당성을 보이기 위한 방법으로 추천 시스템에서 사용되는 성능 검증 방법인 Recall을 이용한다. 텍스트-이미지 임베딩 결과를 시각화하고 실제 영화를 추천하는 방법에 대해 기술한다.

본 논문의 구성은 다음과 같다.

제2장 관련 연구에서는 콘텐츠 추천과 관련된 기존 연구에 대해 설명하고 문제 제기를 통해 해결하고자 하는 문제와 본 논문에서 제안하는 방법에 기반하는 텍스트-이미지 임베딩과 Triplet Ranking Loss에 관해 설명한다.

제3장에서는 본 논문에서 제안하는 텍스트-이미지 임베딩 모델과 개선된 Triplet Ranking Loss에 관해 기술한다.

제4장에서는 실험 결과로 텍스트-이미지 임베딩 모델의 성능 평가와 텍스트-이미지 임베딩의 시각화 및 추천 방법을 기술한다.

마지막으로 5장에서는 본 연구에 대한 전체적인 결과를 요약하고, 향후 연구를 제시하며 마무리한다.

## 제2장 관련 연구

스마트 기기의 보유율 증가와 코로나 여파로 인해 스마트 기기를 이용한 미디어 콘텐츠 이용률과 이용 시간이 증가하고 있다. 이에 따라 OTT 이용률이 증가하고 있으며, 이용 만족도와 지속 사용 의도에 영향을 주는 요소 중 추천 시스템의 중요성이 대두되고 있다. 따라서 본 장에서는 기존의 콘텐츠 추천 연구를 통해 문제점과 개선 방법에 대해 기술하고 본 논문에서 제안하는 방법의 토대가 되는 텍스트-이미지 임베딩과 Triplet Ranking Loss에 관해 설명한다.

### 제1절 콘텐츠 추천 관련 기존 연구

추천 시스템은 정보 필터링 기술의 일종으로 사용자가 원하는 아이템을 찾을 수 있게 도와주는 기술이다. 정보가 늘어나고 다양한 온라인 콘텐츠 플랫폼이 증가함에 따라 사용자의 선택지가 매우 다양해졌지만, 너무 많은 선택지로 인해 취향에 맞는 것을 찾기 위해 훨씬 많은 시간을 소비하고 있다. 따라서 개인의 요구를 충족시켜줄 수 있는 만족스러운 추천 시스템의 중요성이 증가하고 있다 [16, 17].

추천 시스템은 크게 협업 필터링과 콘텐츠 기반 필터링으로 나눌 수 있다. 협업 필터링은 사용자와 아이템을 기반으로 추천하는 방법이다. 협업 필터링에서 사용자 기반 추천은 취향이 비슷한 사용자들을 모아 아이템을 추천해주는 방법이며, 아이템 기반 추천은 관심이 있는 아이템을 기반으로 그 아이템과 연관이 높은 다른 아이템을 추천하는 방법이다. 콘텐츠 기반 필터링은 아이템 자체를 분석하여 추천해주는 방법이며, 협업 필터링과 콘텐츠 기반 필터링을 함께 사용하는 하이브리드 필터링 방법이 있다.

기존의 콘텐츠 추천 시스템은 대부분 사용자와 콘텐츠의 관계를 모델링하여 개인화된 경험 제공을 목표로 하는 협업 필터링과 관련된 연구가 진행되었다. 협업 필터링은 데이터 희소성과 활동 이력이 부족한 사용자에서 메타데이터 부족에 따른 cold-start와 사용자 수가 증가함에 따라 계산량이 기하급수적으로 증가한다는 문제점이 존재한다. 그뿐만 아니라 오래전의 활동 이력이 고려되지 않는 장기 의존성에 대한 문제와 콘텐츠 사이의 특성을 파악하는 것이 어렵다는 점도 존재한다.

따라서 콘텐츠의 정보를 분석하여 추천하는 콘텐츠 기반 필터링 방법을 사용하는 것이 콘텐츠 사이의 특성 및 관계를 정확히 파악할 수 있다. 기존의 영상 콘텐츠를 추천하기 위한 콘텐츠 기반 필터링 방법은 콘텐츠의 메타데이터인 장르, 평점 등과 사용자의 관계를 함께 분석했다[17, 18, 19, 20].

협업 필터링에 기반한 행렬 분해 방법론과 콘텐츠 기반 필터링에서 사용되는 콘텐츠의 메타데이터 정보인 배우, 감독, 장르를 결합한 하이브리드 추천 기법을 제안하는 연구가 있었다. 행렬 분해 방법을 이용하여 사용자의 각 콘텐츠에 대한 선호도 점수를 산출하고 사용자의 배우, 감독, 장르 선호도를 반영하여 보정된 선호도 점수를 계산했다[17].

콘텐츠 기반 필터링 방법에 기반하여 사용자가 선호하는 장르를 기반으로 추천시스템을 구축하는 연구가 있었다. 특정 장르에 대한 사용자의 영화 평점과 영화 장르의 행렬로 내적 결과를 도출하고 유클리디안 거리를 통해 추천 대상 영화를 결정했다[18].

영화의 다양한 메타데이터를 포함하는 Movie Genome의 개념을 이용하여 콘텐츠의 오디오, 이미지, 장르, 태그를 포함하는 메타데이터 특징을 제안하며, 아이템 기반 최근접 이웃 방법으로 영화를 추천하고 애플리케이션으로 구현하는 연구가 있었다[19].

영화의 메타데이터에서 영화와 평점의 정보를 이용한 순환 신경망 기반 영화 추천 모델 개선을 위한 가중치 결합 기법과 모델을 제안하는 연구가 있었다. 유사도 기반의 타겟 벡터와 임베딩, 프로젝션 행렬의 가중치 결합을 극복하는 방법을 통해 기존의 영화 추천 모델의 성능을 개선했다[20].

앞서 설명한 대로 콘텐츠 기반 필터링 추천 시스템이라 할지라도 콘텐츠와 관련된 메타데이터를 분석하는 연구가 대부분이며, 콘텐츠의 실제 내용을 분석하는 연구는 미비한 실정이다. 하지만 메타데이터는 콘텐츠의 내용을 정확히 반영하지 못하기 때문에 적절한 추천이 이뤄지지 못하는 경우가 많다. 콘텐츠를 구성하는 텍스트, 이미지와 같은 데이터를 분석하고 이를 기반으로 사용자에게 추천 서비스를 제공하는 것이 더욱 정확한 결과를 기대할 수 있다.

그뿐만 아니라 현재 서비스 중인 OTT의 콘텐츠 추천 시스템은 사용자에게 메타데이터를 강요하거나 화제성이 높은 콘텐츠를 추천해주는 경우가 대부분이다. 그림 1과 그림 2는 사용자에게 메타데이터를 강요하는 경우와 화제성이 높은 콘텐츠를 추천하는 OTT 화면의 예이다.

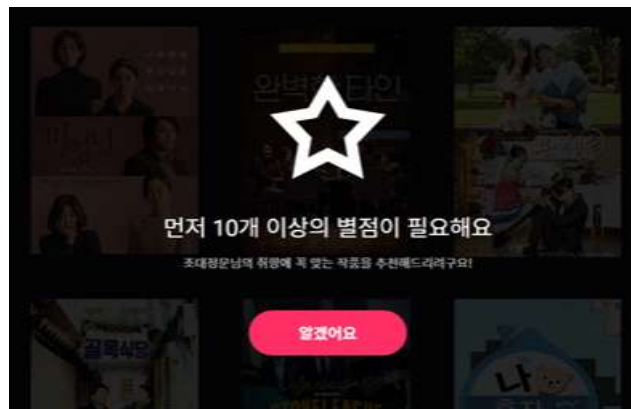


그림 1. 메타데이터를 강요하는 OTT 화면의 예

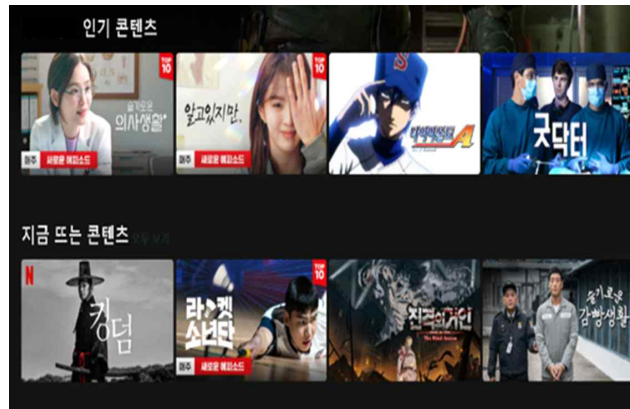


그림 2. 화제성이 높은 콘텐츠를 추천하는 OTT 화면의 예

따라서 본 논문에서는 콘텐츠를 구성하는 텍스트와 이미지 정보를 함께 분석하기 위한 텍스트-이미지 임베딩 방법을 제안하며, 이를 통해 콘텐츠 사이의 유사도를 측정하고 사용자에게 추천하는 방법을 제시한다.

## 제2절 텍스트-이미지 임베딩

텍스트와 이미지 정보를 함께 분석하기 위한 방법으로 텍스트-이미지 임베딩 방법이 존재한다. 텍스트와 이미지는 다른 도메인의 데이터이므로 결합하는데 어려움이 따른다. 텍스트-이미지 임베딩은 특징 추출의 관점에서 다른 도메인의 데이터 결합을 수행한다. 일반적으로 텍스트 데이터를 임베딩하기 위한 텍스트 네트워크와 이미지 특징 추출을 위한 이미지 네트워크, 이 둘을 결합하고 관계를 학습하기 위한 텍스트-이미지 네트워크로 구성된다. 그림 3은 일반적인 텍스트-이미지 임베딩 모델의 구성이다.

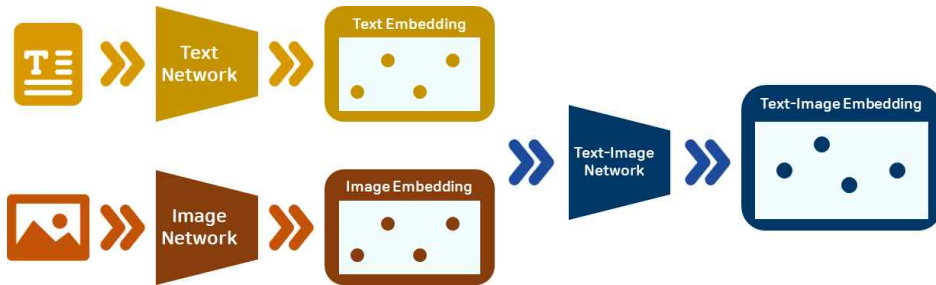


그림 3. 일반적인 텍스트-이미지 임베딩 모델

텍스트 네트워크에서는 텍스트의 특징을 추출하는 텍스트 임베딩을 수행한다. 기존에 연구되어 온 Word2Vec[7], RNN[8], LSTM[9], KoBERT[10]와 같은 모델의 구성으로 학습에 활용할 수 있으며, 특징 추출기를 이용하여 텍스트의 특징을 추출할 수 있다.

이미지 네트워크에서는 이미지의 특징을 추출하는 이미지 임베딩을 수행한다. 기존에 연구되어 온 이미지 분류에 사용되는 ResNet[11], InceptionV3[12], EfficientNet[13]과 같은 모델의 구성으로 학습에 활용할 수 있으며, 사전학습된 모델을 특징 추출기를 이용하여 이미지의 특징을 추출할 수 있다.

텍스트와 이미지 네트워크의 임베딩 방법은 입력 데이터와 해결하고자 하는 문제에 따라 달라질 수 있다.



마지막으로 텍스트-이미지 네트워크에서 텍스트와 이미지 특징을 임베딩하여 두 도메인의 특징을 결합 및 학습한다. 이 과정에서 도메인의 각 클래스 사이의 관계를 학습하며, 해결하고자 하는 문제의 목적에 적합한 Loss 함수를 활용한다. 이때, 일반적으로 Metric Learning 기반의 Loss 함수가 사용된다.

일반적인 텍스트-이미지 임베딩은 텍스트와 이미지를 임베딩으로 결합하고 클래스들 사이의 관계를 잘 표현하는 것을 목표로 한다. 주로 해결하고자 하는 문제는 이미지-텍스트 검색, 이미지 캡셔닝, Visual Question Answering(VQA)이다[21].

일반적인 텍스트-이미지 임베딩 방법으로 Visual Sementic Embedding++(VSE++)가 대표적이다[22]. VSE의 목표는 쿼리에 대한 검색 결과를 도출하는 것인데 쿼리가 캡션일 경우 검색 결과는 이미지이고 쿼리가 이미지인 경우 검색 결과는 캡션이 되는 작업이다. VSE++는 VSE를 개선한 방법으로 텍스트와 이미지 정보를 함께 사용하기 위해 텍스트와 이미지 임베딩을 제안한 초기의 연구이다. VSE++는 텍스트 네트워크를 Gated Recurrent Unit(GRU)[23]로 구성하고 이미지 네트워크를 VGG19[24] 또는 ResNet151[11]를 사용한다. 기존의 Triplet Ranking Loss는 이미지의 관계 학습에 주로 사용되었으나 VSE++는 이미지와 텍스트의 벡터 결합과 관계 학습에 Triplet Ranking Loss를 사용했다.

그 외의 도메인 특징을 학습하기 위한 방법으로 Contrastive Loss[25]가 있다. Metric Learning 관점에서 임베딩된 텍스트와 이미지는 행렬 또는 벡터와 같기 때문에 텍스트-이미지 임베딩에 활용할 수 있다.

텍스트-이미지 임베딩은 서로 다른 도메인인 텍스트와 이미지의 특징을 결합하기 위한 방법으로 콘텐츠를 구성하는 텍스트와 이미지를 결합하고 콘텐츠 사이의 관계를 학습하기 위한 적절한 방법이다. 하지만 기존의 텍스트-이미지 임베딩은 콘텐츠의 내용을 분석하여 콘텐츠 간 관계를 도출하는 연구는 미비한 실정이다.

따라서 본 논문에서는 콘텐츠 분석에 텍스트-이미지 임베딩을 활용하기 위한 방법을 제안한다. 기존의 텍스트-이미지 방법을 콘텐츠 간 관계를 도출하는 작업에 이용해보고 제안한 방법과 비교 실험하여 기존 텍스트-이미지 임베딩 방법과 제안하는 방법의 성능을 평가한다.

## 1. 텍스트 임베딩

텍스트 임베딩은 사람이 일상생활에서 사용하는 말, 문자 등과 같은 자연어를 컴퓨터를 통하여 처리하기 위한 자연어처리 방법 중 하나이며, 텍스트를 벡터로 표현하기 위한 방법이다. 텍스트 임베딩은 처리하고자 하는 텍스트의 단위에 따라 단어 임베딩, 문장 임베딩, 문서 임베딩 등으로 구분될 수 있으나 일반적으로 자연어에서 가장 기초 단위인 단어를 이용한다.

전통적인 텍스트 임베딩 방법으로 one-hot-vector, n-gram, bag of words 등이 있으나 단어의 빈도수를 기반으로 하기 때문에 주변 단어에 대한 정보와 단어 관계 분석에 어려움이 있으며, 등장하는 단어에 따라 벡터의 차원이 함께 증가하기 때문에 차원의 저주에 빠지게 되고 비효율적이다.

Word2Vec[7]은 단어를 벡터로 표현하기 위한 방법인 단어 임베딩 방법 중 하나로 비슷한 위치에 등장하는 단어는 비슷한 의미를 가진다는 가정을 가지고 주변 단어를 통해 중심 단어를 파악하는 것을 목적으로 하는 Neural Network Language Model(NNLM)이다.

Word2Vec은 n의 크기를 갖는 window를 이용하여 중심 단어와 주변 단어를 구성한다. 예를 들어 window의 크기가 2이고 “나는 건강을 위해 등산을 한다”라는 문장이 있을 경우 중심 단어와 주변 단어는 그림 4와 같이 구성된다.

	중심 단어	주변 단어
나는 건강을 위해 매주 등산을 한다	[1, 0, 0, 0, 0, 0]	[0, 1, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0]
나는 건강을 위해 매주 등산을 한다	[0, 1, 0, 0, 0, 0]	[1, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0], [0, 0, 0, 1, 0, 0]
나는 건강을 위해 매주 등산을 한다	[0, 0, 1, 0, 0, 0]	[1, 0, 0, 0, 0, 0], [0, 1, 0, 0, 0, 0] [0, 0, 1, 0, 0, 0], [0, 0, 0, 1, 0, 0]
⋮	⋮	⋮
나는 건강을 위해 매주 등산을 한다	[0, 0, 0, 0, 0, 1]	[0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 1, 0]

그림 4. Word2Vec에서 중심 단어와 주변 단어 구성의 예(window size=2)

그림 4의 왼쪽은 예문으로 window의 크기에 따라 구성되는 중심 단어와 주변 단어의 구성을 나타내고 있으며, 오른쪽은 구성된 중심 단어와 주변 단어를 bag of words를 통해 임베딩 하는 과정을 보인다. Word2Vec[7]은 이렇게 구성된 중심 단어와 주변 단어를 입력과 출력으로 하는 NNLM 기반의 구조를 가지며, Continuous Bag of Words(CBOW)와 Skip-Gram의 방식으로 나뉜다. 그림 5는 CBOW와 Skip-Gram 기반의 Word2Vec 구조이다.

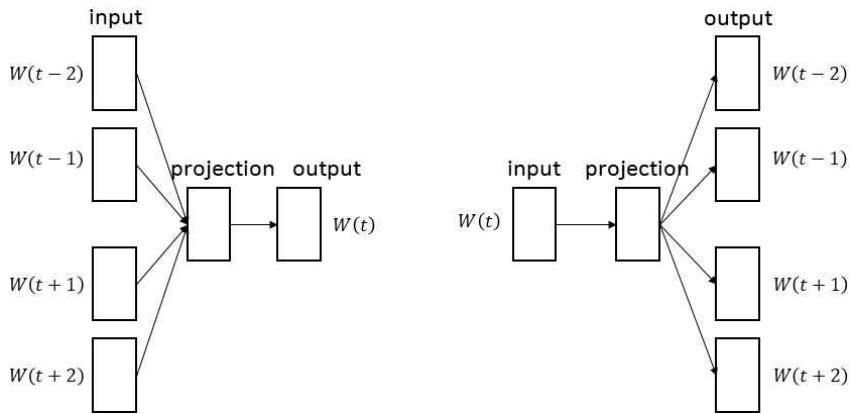


그림 5. Word2Vec의 CBOW(왼쪽)과 Skip-Gram(오른쪽)의 구조

그림 5의 왼쪽은 Word2Vec의 CBOW 기반의 모델이다. Input layer, Projection layer, Output layer로 구성되며,  $W$ 는 단어,  $W(t)$ 는 중심 단어이다. CBOW 기반의 모델은 주변 단어인  $W(t \pm n)$ 를 입력으로 받고 projection layer를 통해서 가중치를 업데이트한다. 여기서  $n$ 은 window의 크기이다. CBOW는 주변 단어를 통해 중심 단어를 예측한다. 그림 5의 오른쪽은 Word2Vec의 Skip-Gram 기반의 모델로 CBOW와 반대로 중심 단어를 통해 주변 단어를 예측한다.

딥러닝의 성능이 개선되면서 텍스트 분석을 위한 다양한 모델들이 등장했고 Recurrent Neural Network(RNN)는 그중 가장 기본이 되는 모델 중 하나이다. RNN은 순서를 가진 시계열 데이터를 학습할 때 주로 사용되며, 기존의 Deep Neural Network(DNN)과 달리 이전의 상태를 고려하여 처리한다. 그렇기 때문에 순차적으로 등장하는 텍스트의 정보를 처리하는 데 적합하고 널리 활용되었다.

RNN[8]은 hidden state를 기반으로 모델이 구성된다. 시계열 데이터의 순서와 같은 구조의 hidden state로 학습하는 재귀적인 구조를 갖는다. 현재 hidden state에서 학습한 결과를 다음 hidden state로 전달하는데 이전의 결과를 새로운 결과와 현재의 상태를 입력으로 학습하는 형태이다. 그림 6은 RNN[8]의 구조이다.

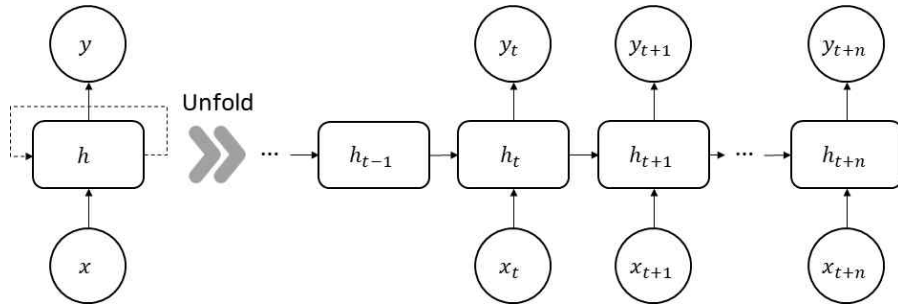


그림 6. Recurrent Neural Network(RNN)의 구조

그림 6의 왼쪽은 기본적인 hidden state를 포함하는 RNN cell의 구조이며, 오른쪽은 RNN cell이 재귀적으로 학습하는 과정을 입력 데이터의 순서에 따라 표현한 그림이다. 여기서  $x$ 는 입력이고  $y$ 는 출력,  $h$ 는 hidden state이다. RNN cell에서 현재 상태의 hidden state인  $h_t$ 는 이전의 hidden state인  $h_{t-1}$ 과 현재의 입력인  $x_t$ 를 입력으로 하고  $y_t$ 를 출력한다. 출력  $y_t$ 는 이전 hidden state인  $h_{t-1}$ 를 고려한 입력  $x_t$ 에 대한 결과이다.  $h_t$ 와  $y_t$ 의 수식은 다음과 같다.

$$h_t = \tanh(W_h h_{t-1} + W_x x_t + b_h) \quad (1)$$

$$y_t = W_y h_t + b_y \quad (2)$$

수식 1은 현재 상태의 hidden state인  $h_t$ 를 계산하기 위한 수식이며,  $\tanh(x)$ 는  $x$ 에 대한 활성화 함수인 하이퍼볼릭탄젠트이고  $W$ 는 가중치,  $b$ 는 편향이다.  $y_t$ 는 이전의 hidden state인  $h_{t-1}$ 과 현재의 입력인  $x_t$ 의 결과이며, 수식 2와 같다. 일반적으로  $y_t$ 에 Softmax를 이용하여 결과를 도출한다.

하지만 깊은 계층의 RNN을 구성하기 위해 많은 RNN cell을 사용하게 될 경

우 학습하기 위한 파라미터의 수가 기하급수적으로 증가하게 되고 먼 과거부터 계산해온 hidden state가 정상적으로 고려되지 않아 기울기 소실 문제가 발생할 수 있다. 이를 개선하기 위한 방법으로 Long-Short Term Memory(LSTM)[9]가 제안되었다. RNN에서 발생하는 기울기 소실 문제는 학습 과정에서 가중치가 계속 곱해짐에 따른 문제이다. LSTM은 이를 해결하기 위해 RNN의 hidden state 구조에 cell state를 추가하는 구조를 통해 기울기 소실 문제를 극복했다. 그림 7은 LSTM[9]의 구조이며, 식 3, 4, 5, 6, 7, 8은 LSTM을 구성하는 gate와 cell의 식이다.

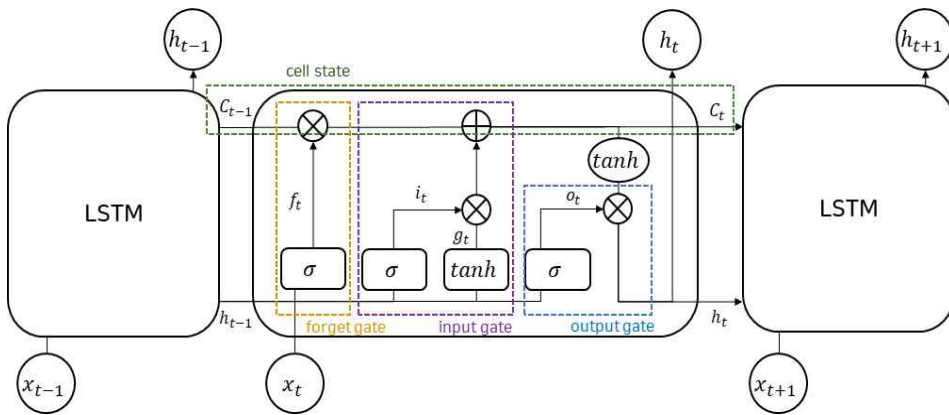


그림 7. Long-Short Term Memory(LSTM)의 구조

$$f_t = \sigma(W_h h_{t-1} + W_x x_t + b_h) \quad (3)$$

$$i_t = \sigma(W_h h_{t-1} + W_x x_t + b_h) \quad (4)$$

$$o_t = \sigma(W_h h_{t-1} + W_x x_t + b_h) \quad (5)$$

$$g_t = \tanh(W_h h_{t-1} + W_x x_t + b_h) \quad (6)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \quad (7)$$

$$h_t = o_t \odot \tanh(c_t) \quad (8)$$

forget gate는 과거 정보를 일부 소실시키기 위한 gate로 그림 7에서 노란색 점선 박스 부분에 해당한다. 이전 hidden state인  $h_{t-1}$ 과 입력인  $x_t$ 를 받아 sigmoid 함수를 취한다. sigmoid 함수의 출력값은 0에서 1 사이이며, 그 값이 0과 가까울수록 과거 정보를 소실시키고 1과 가까울 경우 과거 정보를 보존한다. forget gate는 식 3과 같이 표현할 수 있으며,  $W$ 는 가중치  $h_{t-1}$ 은 이전 hidden state의 값,  $x_t$ 는 입력값,  $b$ 는 편향이다.

input gate는 현재 입력으로 들어오는 정보를 기억하기 위한 gate로 그림 7에서 보라색 점선 박스 부분에 해당한다. hidden state인  $h_{t-1}$ 과 입력인  $x_t$ 를 받아 활성화 함수인 sigmoid 함수를 취한 값과 같은 입력을 받아 활성화 함수인 하이퍼볼릭탄젠트를 취한 값에 element-wise 곱의 연산 결과가 출력된다. input gate는 식 4와 식 6에 대한 element-wise 연산인  $i_t \odot g_t$ 으로 표현할 수 있다. 식 4에서  $W$ 는 가중치,  $h_{t-1}$ 은 이전 hidden state의 값,  $x_t$ 는 입력 값,  $b$ 는 편향이다. 식 6에서  $\tanh(x)$ 는 활성화 함수인 하이퍼볼릭탄젠트이고  $W$ 는 가중치  $h_{t-1}$ 은 이전 hidden state의 값,  $x_t$ 는 입력값,  $b$ 는 편향이다.

output gate는 forget gate와 같은 구성과 같은 수식을 가지며, 그림 7에서 파란색 점선 박스 부분에 해당한다. output gate는 현재 상태인  $h_t$ 의 값을 구할 때 쓰인다.

앞서 설명한 것과 같이 LSTM에서 가장 핵심이 되는 부분인 cell state는 그림 7에서 초록색 점선 부분으로  $C$  와 같다. cell gate는 과거의 정보를 일부 소실시키기 위한 forget gate, 이전 cell gate의 값을 element-wise 연산한 값, input gate의 출력값을 더하는 것으로 구성된다. 이를 통해 상당히 오래된 과거의 정보를 비교적 잘 전파 할 수 있으며, 기울기 소실 문제를 해결할 수 있다. 현재의 cell gate 값은 수식 7과 같이 나타낼 수 있다.

RNN[8]과 RNN을 기반으로 하는 LSTM[9]은 hidden state를 기반으로 하여

과거의 상태를 고려하는 모델로 일부 모듈이 재귀적으로 동작하는 비교적 단순한 구조를 갖기 때문에 목적에 따라 다양한 구조로 변형이 가능하다. 이에 따라 인코더, 디코더로도 사용이 가능하며, 문제를 해결하기 위한 작업에 따라 철자 단위의 언어 모델, 이미지 캡셔닝, 감정분류, 기계번역 등에도 활용할 수 있다.

자연어에 해당하는 텍스트를 분석하기 위한 다양한 NNLM이 고안되었으나 자연어처리의 고질적인 문제인 텍스트 데이터의 부족에 따른 성능 저하를 개선하기 어려운 점이 많았다. BERT[10]는 사전학습과 전이학습을 아이디어로 우수한 성능 개선을 이뤄냈다. BERT[10]는 사전학습을 기본으로 대규모 언어 모델을 구축하고 필요와 작업의 목적에 맞게 전이학습으로 우수한 성능을 보이는 모델이다. BERT는 입력 단계에서 token embedding, segment embedding, position embedding을 통해 문장을 임베딩한다. token embedding을 통해 문장의 시작과 문장의 끝을 특수 토큰을 통해 구별한다. segment embedding은 각 단어가 몇 번째 문장인지 마킹한다. position embedding은 transformer[26]에서 사용되는 방법으로 각 토큰의 위치를 마킹한다. BERT[10]는 문장 임베딩을 위해 비지도학습 모델인 Masked Language Model(MLM)과 Next Sentence Prediction(NSP)을 사용한다.

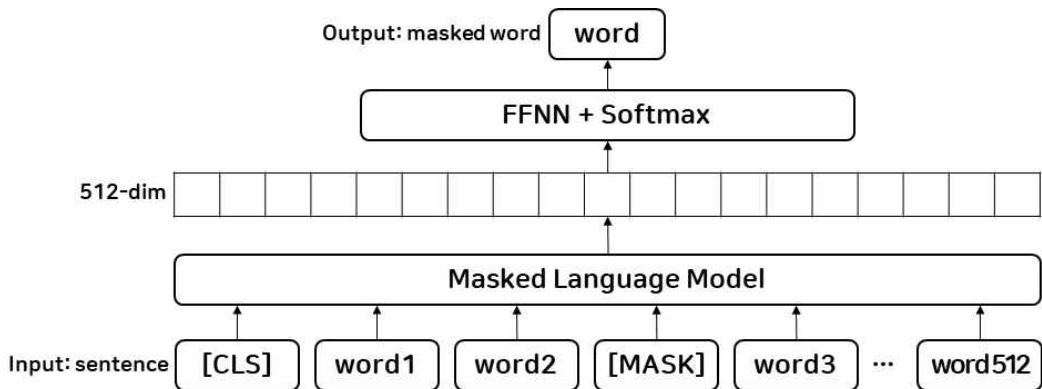


그림 8. BERT의 Masked Language Model(MLM) 구조

그림 8은 BERT에서 MLM의 구조를 나타낸다. 문장을 입력으로 하며, 문장은 512개 이하의 단어로 구성되어야 한다. 문장에서 단어의 일부를 Mask 토큰으로



바꾸고 Mask로 가려진 단어를 잘 예측할 수 있도록 학습하는 과정을 거친다. 일반적으로 문장의 단어 중 15% 정도를 masking 한다. MLM을 통해 추출된 문장 단어의 특징은 Feed-Forward Neural Network(FFNN)와 Softmax layer를 거쳐 masking 된 단어를 예측하게 된다. 이를 통해 문맥을 파악하는 능력을 갖게 되고 문맥에 따라 가장 적합한 단어를 알 수 있다.

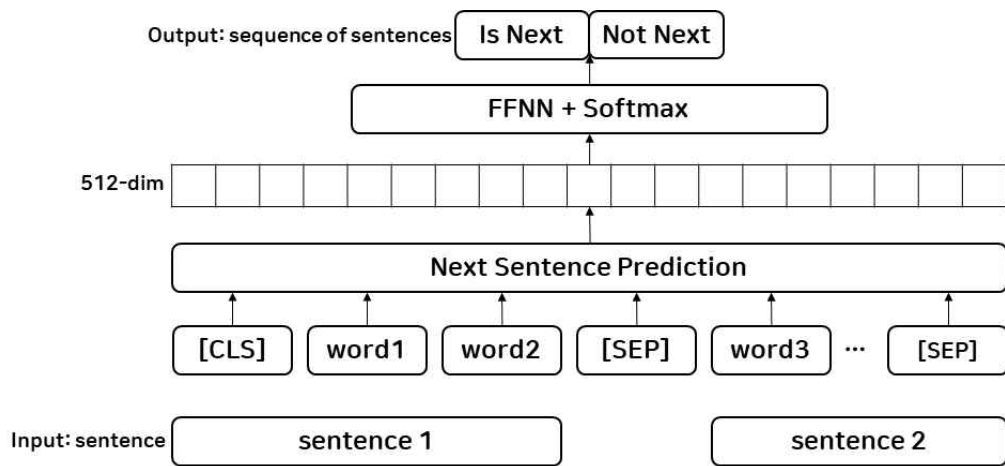


그림 9. BERT의 Next Sentence Prediction(NSP) 구조

그림 9는 BERT에서 NSP의 구조를 나타낸다. 자연어는 단어의 순서에 따라 문장을 구성하고 문장의 순서에 따라 문단 또는 문서로 구성되기 때문에 Q&A 시스템이나 자연어 추론에서 문장과 문장 간의 순서와 관계가 중요하다. 두 개의 문장을 입력으로 하고 중심 문장 다음에 주변 문장이 등장하는지를 예측하기 위해 Next Sentence Prediction을 거치며, 그 결과인 문장의 특징은 FFNN과 Softmax layer를 거치는 학습 과정을 수행한다. 두 문장의 순서를 예측하는 방식이며, 이를 통해 두 문장 사이의 관계 및 순서를 학습하게 된다.

앞서 설명한 Word2Vec[7], RNN[8], LSTM[9], BERT[10]는 분석하는 규모가 다르기는 하나 일반적으로 텍스트 임베딩에 자주 활용되는 모델이다. 그뿐만 아니라 최근 자연어처리를 하는 딥러닝 모델은 텍스트의 특징을 추출하기 위한 인코더와 같은 형태로 사전학습 모델을 활용하는 추세이다.

따라서 본 논문에서는 텍스트 네트워크에 적용하기 위해 Word2Vec[7]과 BERT[10]는 이전 연구로 공개된 사전학습 모델을 이용하며, BERT는 한국어 버전인 KoBERT[10]를 이용한다. RNN[8]과 LSTM[9]은 기본적인 형태로 학습하여 각 방법을 텍스트 임베딩으로 활용한다. 각 방법을 통해 제안하는 모델의 텍스트 임베딩 네트워크로 사용하고 가장 성능이 좋은 텍스트 임베딩 방법을 선정한다.

## 2. 이미지 임베딩

이미지 임베딩은 이미지의 특징을 추출하는 과정으로 고차원으로 구성된 이미지의 픽셀을 저차원적 특성 벡터 변환을 통해 이미지를 분석하기 위한 기초적인 단계이다. 이미지 임베딩은 이미지를 컴퓨터가 처리할 수 있는 형태로 변경하고 이미지의 관계를 고려한다는 부분에서 기본적인 목적은 텍스트 임베딩과 같다.

이미지 분류 모델 중 하나인 Convolutional Neural Network(CNN)이 등장한 이후에 Deep Neural Network(DNN)를 이용한 이미지처리의 성능이 비약적으로 증가했다. 그에 따라 DNN 구조의 다양한 이미지처리 방법들이 제안되었으며, 그 성능도 지속적으로 증가하여 사람을 능가하는 작업도 등장했다.

ResNet[11]은 VGG의 구조를 기반으로 하며, Convolutional layer를 추가하여 깊게 만들었다. 하지만 네트워크를 깊게만 만든다고 좋은 성능을 보이지 않기 때문에 Residual block을 제안하여 깊이는 네트워크는 깊지만 성능은 개선할 수 있는 방식을 취했다[24]. 그림 10은 Residual block의 구조이다.

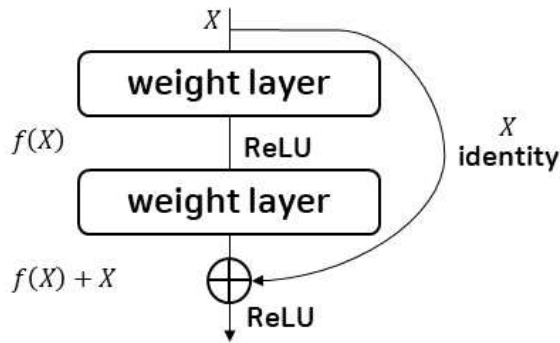


그림 10. Residual block의 구조

기존의 방법은 연결된 layer를 연속적으로 통과하면서 입력  $X$ 의 출력인  $f(X)$ 를 구하기 위한 학습을 했지만 ResNet은 입력  $X$ 와 입력  $X$ 에 대한  $f(X)$ 의 연산인  $f(X)+X$ 을 통해  $f(X)$ 가 0이 되는 방향으로 학습한다. 이러한 방식으로

깊은 층을 갖는 모델일지라도 연산을 할 때 layer를 생략하는 방식으로 연산량을 줄였을 뿐만 아니라 성능까지 개선되는 것을 보였다. ResNet은 Residual block을 기반으로 하여 크기가 다른 여러 개의 kernel로 구성된 Convolutional layer로 네트워크를 구성했다.

InceptionV3[12]는 기본적으로 CNN의 Convolutional layer에서 발생하는 수많은 연산을 줄이면서 성능을 극대화하는 것을 목표로 한다. 특히 kernel의 크기가 커질수록 연산량이 늘어나기 때문에 이러한 문제를 해결하고자 kernel의 크기를 줄이면서 성능은 개선할 수 있는 모듈을 제안했다. 그림 11은 InceptionV3 모듈의 구조이다.

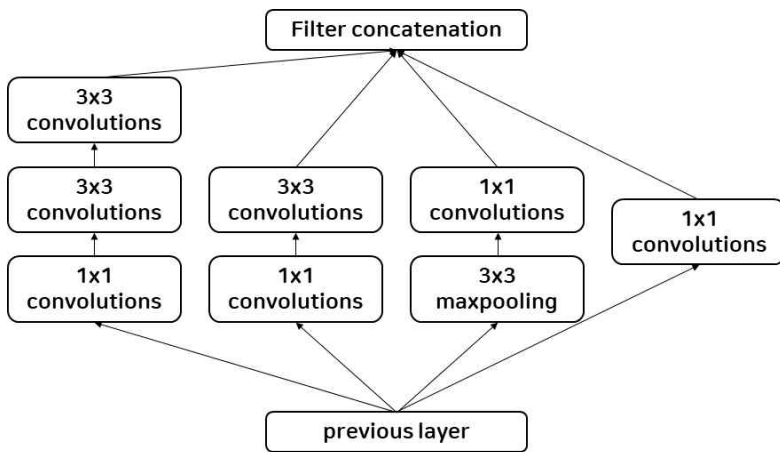


그림 11. InceptionV3 모듈의 구조

Inception은 V1부터 V4까지 버전을 업데이트해왔다[27, 28, 29]. 기본적으로 InceptionV1에서 제안한 구조를 따르면서 내부 구조의 convolutional layer의 kernel 크기와 구성을 달리하면서 성능을 점점 개선해왔다. InceptionV3는 InceptionV2와 같은 구조를 갖지만, optimizer를 변경하고 factorized 7x7로 모듈의 구조를 약간 변경하고 마지막 Fully Connected layer에 Batch Normalization(BN)을 하는 BN-auxiliary를 통해 성능을 개선했다.

일반적으로 모델의 성능을 높이기 위해 모델의 깊이, 모델의 너비, 입력 이미지의 해상도를 조절한다. 하지만 이 세 가지를 한꺼번에 조절하여 최적의 상태를 찾는 것은 쉽지 않은 문제이며, 성능 향상에 어려움이 있었다. EfficientNet[18]은 이 세 가지가 최적화될 수 상태를 찾는 방법을 제안한다. 모델의 깊이를 조절하는 방식은 모델이 갖는 layer의 개수에 변화를 주는 것이다. 모델의 너비를 조절하는 방식은 모델의 layer가 갖는 convolution kernel의 개수나 크기에 변화를 주는 것이다. 해상도가 높은 이미지는 모델의 깊이를 깊게 하고 모델의 너비를 넓게 하는 것이 양질의 이미지 특징을 추출할 수 있기 때문에 중요하다. 이와 같은 이유를 기반으로 모델의 깊이와 모델의 너비, 이미지의 해상도를 균등하게 조절할 수 있는 방법을 제안했다. 제안한 방법을 통해 기존의 CNN 기반 모델에 적용했다. 그림 12는 EfficientNet의 구조이다.

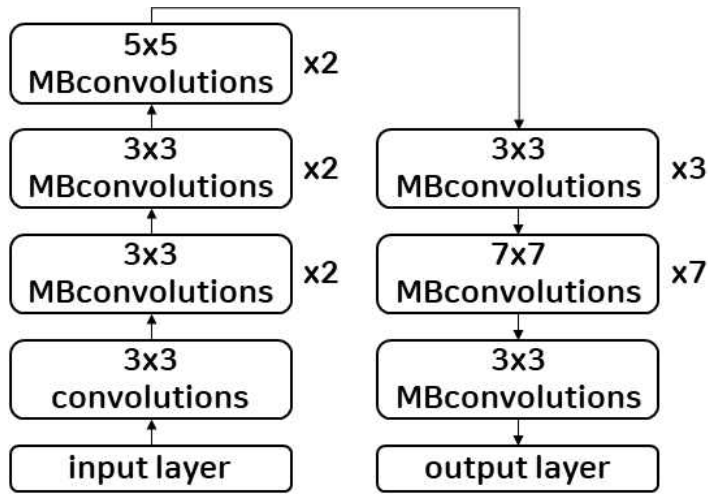


그림 12. EfficientNet의 구조

EfficientNet은 모바일에서 사용 가능한 최적의 구조를 갖는 모델을 찾기 위한 모델인 MNasNet[30]의 구조를 기반으로 하며, MBconvolutions는 Inverted bottleneck을 갖는 MobileNet V2[31]에서 쓰는 기본 모듈이다. MNasNet은 모바일 환경에서 사용 가능한 모델과 관련된 연구였기 때문에 지연속도를 고려하였지만 EfficientNet은 이를 고려하지 않았다.

앞서 설명한 ResNet[11], InceptionV3[12], EfficientNet[13]은 이미지 분류를 위해 제안된 모델이지만 사전학습된 모델로 이미지 임베딩에 자주 활용된다. 자연어처리를 위한 텍스트 임베딩과 같이 이미지처리를 위한 이미지 임베딩에서도 이미지의 특징을 추출하기 위한 인코더와 같은 형태로 사전학습 모델을 활용하는 추세이다. 따라서 본 논문에서는 이미지 네트워크에 적용하기 위해 이전 연구로 공개되어 있는 사전학습된 ResNet[11], InceptionV3[12], EfficientNet[13]을 이미지 임베딩으로 활용한다. 각 방법을 통해 제안하는 모델의 이미지 임베딩 네트워크로 사용하고 가장 성능이 좋은 이미지 임베딩 방법을 선정한다.

### 제3절 Triplet Ranking Loss

Triplet Ranking Loss[15, 23]는 주어진 데이터의 상대적인 거리를 예측하는 것이 목표이며, 벡터화된 텍스트 임베딩이나 이미지의 특징으로부터 학습을 수행하기 때문에 Metric Learning이라고도 한다.

Triplet Ranking Loss는 학습에 3개의 데이터를 한 쌍으로 이용한다. 기준이 되는 데이터를 Anchor, 나머지 2개의 데이터를 Negative와 Positive라고 정의한다. Anchor와 Positive, Anchor와 Negative의 거리를 측정하여 Positive는 더욱 가깝게 학습하고 Negative는 더욱 멀게 학습하는 것이 목표이다. 그림 13은 Triplet Ranking Loss의 학습 예이다.



그림 13. Triplet Ranking Loss의 학습 과정

그림 13의 왼쪽에서 보는 것과 같이 파란색 점인 Anchor와 주황색 점인 Negative, 초록색 점인 Positive가 있을 때, 학습을 통해 오른쪽과 같이 Positive인 초록색 점을 Anchor와 가깝게 배치하고 Negative인 주황색 점을 멀리 배치한다. Triplet Ranking Loss의 수식은 아래와 같다.

$$Loss = \max(0, m + d(f(x_a), f(x_p)) - d(f(x_a), f(x_n))) \quad (9)$$

수식 9에서,  $x_a$ 는 anchor이고  $x_p$ 는 positive,  $x_n$ 은 negative,  $m$ 은 마진이며,  $d(x)$ 는 거리 측정 함수이다. 여기서  $d(x)$ 는 유클리디안, 코사인 거리 함수 등이 될 수 있다. 따라서  $x_a$ 와  $x_n$ 의 거리 값이  $x_a$ 와  $x_p$ 의 거리 값보다 크면 Loss는 음의 값을 갖게 되는데 최대값이 0이 되도록 정의되었기 때문에 Loss를 0으로

업데이트 하게 되고 이 경우는 학습하지 않는다.

반대로  $x_a$ 와  $x_p$ 의 거리 값이  $x_a$ 와  $x_n$ 의 거리 값보다 클 경우  $m$ 을 고려하여  $m$ 의 사이값으로 들어오는 경우와 들어오지 않는 경우를 판단하고 Loss를 계산한다[15].

앞서 설명한 대로 Triplet Ranking Loss를 업데이트하는 과정에서 발생하는 3가지 경우의 수가 발생한다. 이는  $x_a$ 와  $x_p$ ,  $x_a$ 와  $x_n$ 의 거리에 따라 발생하는 것이며, 업데이트 경우는 Easy Negatives, Semi-Hard Negatives, Hard Negatives가 있다.

그림14는 Triplet Ranking Loss의 3가지 업데이트 경우의 수를 나타내며, 수식 10은 Easy Negatives, 수식 11은 Semi-Hard Negatives, 수식 12는 Hard Negatives이다.



그림 14. Triplet Ranking Loss의 업데이트 경우의 수



$$d(f(x_a), f(x_n)) > d(f(x_a), f(x_p)) + m \quad (10)$$

$$d(f(x_a), f(x_p)) < d(f(x_a), f(x_n)) < d(f(x_a), f(x_p)) + m \quad (11)$$

$$d(f(x_a), f(x_n)) < d(f(x_a), f(x_p)) \quad (12)$$

그림 14에서 파란색 점은  $x_a$ 이고 초록색 점은  $x_p$ 이다.  $x_n$ 의 위치에 따라 Loss 값이 달라질 수 있으며, 그 경우는 초록색 원에 포함되는 Easy Negatives, 노란색 원에 포함되는 Semi-Hard Negatives, 주황색 원에 포함되는 Hard Negatives이다.

수식 10는 그림 14의 초록색 원에 포함되는 Easy Negative 상태로 최초 상태에서  $x_a$ 를 기준으로  $x_n$ 가  $x_p$ 보다 멀리 있기 때문에 더 이상 학습할 필요가 없기 때문에 Loss가 0이 되고 업데이트 되지 않는 상황이다.

수식 11는 그림 14의 노란색 원에 포함되는 Semi-Hard Negatives 상태이며,  $x_a$ 를 기준으로  $x_n$ 가  $x_p$ 보다 멀리 있지만  $m$ 의 값에 따라  $x_a$ 와  $x_n$ 의 거리가 달라질 수 있다. 따라서 이 경우의 Loss는  $m$ 의 값에 따라 달라지며,  $m$ 보다는 작은 값으로 업데이트 된다.

수식 12는 그림 14의 주황색 원에 포함되는 Hard Negatives 상태이며,  $x_a$ 를 기준으로  $x_n$ 가  $x_p$ 보다 가까이에 있기 때문에, Loss는  $x_a$ 와  $x_p$ 의 거리 값만큼 업데이트 된다.

본 논문에서는 학습 과정에서 사용되는 Triplet Ranking Loss를 개선하기 위한 방법을 제안한다. 기존의 방법은 거리 측정을 통해 도출된 값이 0 또는 음수일 때 아예 업데이트를 하지 않는 경우가 발생한다. 이런 상황이 빈번하게 발생하면 업데이트가 잘 되지 않고 학습이 정상적으로 되지 않는 경우가 발생할 수 있다. 음수의 경우에도 그 값이 지니는 의미가 있기 때문에 이를 학습에 활용하기 위한 방법을 제안한다.

### 제3장 영상 콘텐츠 유사도 측정 방법

본 논문에서는 영상 콘텐츠의 유사도 측정을 위한 텍스트-이미지 임베딩 모델을 제안한다. 제안하는 방법은 기존의 텍스트-이미지 임베딩 모델의 구성과 같이 텍스트를 임베딩하는 텍스트 네트워크와 이미지를 임베딩하는 이미지 네트워크, 임베딩 된 텍스트와 이미지의 관계를 고려하여 임베딩하는 텍스트-이미지 네트워크로 구성된다.

#### 제1절 제안하는 텍스트-이미지 임베딩 모델

본 장에서는 영상 콘텐츠 유사도 측정을 위해 제안하는 텍스트-이미지 임베딩 모델의 구성과 개선된 Triplet Ranking Loss에 관해 기술한다. 2절 관련 연구에서 기술한 텍스트-이미지 임베딩과 Triplet Ranking Loss를 기반으로 하며, 제안하는 모델의 세부적인 구성과 Triplet Ranking Loss의 개선 방법에 관해 설명한다. 그림 15는 본 논문에서 제안하는 텍스트-이미지 임베딩 모델의 구성이다.

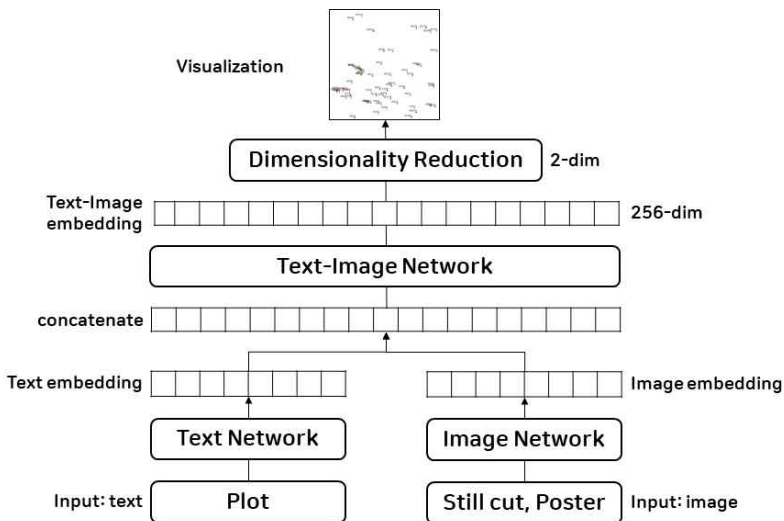


그림 15. 제안하는 Text-Image Embedding 모델의 구조

## 1. 텍스트 네트워크

본 절에서는 제안하는 텍스트-이미지 임베딩 모델에서 텍스트를 임베딩 하는 텍스트 네트워크의 구성에 관해 기술한다. 일반적으로 텍스트 인코더와 같은 역할은 하는 텍스트 임베딩으로 Word2Vec[7], RNN[8], LSTM[9], KoBERT[10]를 이용하여 텍스트 네트워크를 구성해 보고 그 결과를 통해 성능이 가장 좋은 모델을 본 논문에서 제안하는 텍스트-이미지 임베딩 모델의 텍스트 네트워크로 선정한다. Word2Vec[7]과 KoBERT[10]는 기존 연구를 통해 공개된 사전학습 모델을 이용하고 RNN[8]과 LSTM[9]은 기본적인 형태로 네트워크를 구성하여 텍스트 임베딩에 사용한다.

텍스트 네트워크에서 Word2Vec로 임베딩을 하기 위해 기존에 공개된 사전학습 모델을 이용한다[32]. 자연어처리와 관련된 토픽 모델링과 말뭉치 학습에 사용되는 파이썬 라이브러리인 gensim으로 사전학습 되었으며, 사전학습된 Word2Vec의 출력 vector size는 200이고 학습에 사용된 말뭉치의 크기는 339M, 단어의 개수는 30,185개이다. 표 1은 사전학습된 Word2Vec으로 문장을 임베딩하는 과정을 보인다.

표 1. 사전학습 된 Word2Vec으로 문장을 임베딩하는 과정

입력 문장	나는 건강을 위해 매주 운동을 한다
형태소 구문 분석 & 어간 추출	‘나’, ‘는’, ‘건강’, ‘을’, ‘위해’, ‘매주’, ‘운동’, ‘을’, ‘하다’
단어 임베딩 embedding shape(9, 200)	[[0.978328 -2.1541088 ... 1.232299 -2.092169] [0.01761385 0.83592516 ... -0.38992476 0.21850778] ... [-2.4837196 1.109513 ... -0.6130726 -0.10444362] [1.3277844 -1.4225535 ... -0.62285185 1.3685807]]
문장 임베딩 embedding shape(200)	[-0.29741368 0.39047575 ... 0.05722494 0.1159771]

입력 문장을 형태소 구문 분석을 통해 단어로 분해하고 어간 추출을 통해 단어의 원형으로 변형해준다. Word2Vec[7]의 경우 학습에 사용한 단어를 말뭉치 사전으로 구축하는데 말뭉치 사전에 없는 경우 임베딩을 할 수 없다. 그다음으로 각 단어를 임베딩한다. 표 1의 예에서는 어간 추출로 추출한 단어의 수가 9개이고 사전학습으로 출력하는 vector size가 200이기 때문에 9 x 200의 크기를 갖는 벡터가 반환된다. 각 단어 벡터를 문장 벡터로 변환하기 위해 각 벡터의 열을 기준으로 평균을 취해준다.

텍스트 네트워크에서 RNN[8]을 사용하기 위해 기본적인 형태의 RNN 구조를 구성하여 인코더와 같이 사용했다. RNN[8]에 문장을 입력하기 위해 전체 문서에 대한 단어 사전을 구축하고 bag-of-words 한다. 20개의 RNN cell로 구성했으며, 각 문장에서 최대 문장의 길이로 입력 크기를 설정했고 최대 문장의 길이보다 짧은 문장은 zero padding으로 입력 크기를 맞췄다. 활성화 함수로는 하이퍼볼릭 탄젠트를 사용했고 출력의 크기는 768이다. 표 2는 RNN으로 임베딩한 결과를 보인다.

표 2. 기본적인 RNN으로 문장을 임베딩하는 과정

입력 문장	나는 건강을 위해 매주 운동을 한다
문장 임베딩 embedding	[[2.80887983e-03 - 6.37548603e-03 2.34103063e-04 ...
shape(768)	3.87605908e-03 - 5.97368693e-03 -5.66468108e-04]]

텍스트 네트워크에서 LSTM[9]을 사용하기 위해 기본적인 형태의 LSTM 구조를 구성하여 인코더와 같이 사용했다. forget, input, output gate에서 사용하는 활성화 함수는 sigmoid, sell state의 활성화 함수는 하이퍼볼릭탄젠트로 구성했다. 입력 문장의 처리와 LSTM cell의 개수, 출력의 크기는 RNN[8]과 동일하게 진행했다. 표 3은 LSTM[9]으로 임베딩한 결과를 보인다.

표 3. 기본적인 LSTM으로 문장을 임베딩하는 과정

입력 문장	나는 건강을 위해 매주 운동을 한다
문장 임베딩 embedding shape(768)	[[ -6.8857765e-04 - 8.7323471e-04 -3.7422928e-03 ... 3.9630355e-03 - 4.6416780e-04 2.7927137e-03]]

텍스트 네트워크에서 KoBERT를 사용하기 위해 기존에 공개된 사전학습 모델을 이용한다[33]. 입력 문장을 토큰나이저 방법 중 하나인 WordPiece[34]를 통해 토큰화하고 bag-of-words와 같은 방법으로 단어를 input ids로 변환한다. 사전학습된 KoBERT의 출력 vector size는 768이고 학습에 사용된 말뭉치의 크기는 문장 5M, 단어 54M, 단어의 개수는 8,002이다. 최대 입력을 512개의 단어까지 받으며, 출력의 크기는 768이다. 표 4는 사전학습된 KoBERT[33]로 임베딩한 결과를 보인다.

표 4. 사전학습된 KoBERT으로 문장을 임베딩하는 과정

입력 문장	나는 건강을 위해 매주 운동을 한다
토큰화	[CLS], ‘_나는’, ‘_건강’, ‘_을’, ‘_위해’, ‘_매주’, ‘_운동’, ‘_을’, ‘_한다’, [SEP]
input ids	[2, 1375, 882, 7088, 3567, 1999, 3514, 7088, 4965, 3]
문장 임베딩 embedding shape(768)	[-0.030031124129 - 0.031172504648 -0.18609851598, ... 0.042988702654 - 0.045441471040 -0.068088270723]

표 4의 입력 문장이 토큰화된 결과에서 특수 토큰인 [CLS]와 [SEP]가 등장한다. [CLS]는 문장의 시작을 알리는 특수 토큰이고 [SEP]는 문장을 구분해주는 특수 토큰이다. 토큰화 결과를 보면 언더바(\_)가 있는 단어가 존재하는데 언더바는 뒤에 등장하는 단어와 결합되는 경우에 구분하기 위함이다. 언더바가 있는 단어의 언더바를 제거하고 뒤의 단어를 붙이게 되면 토큰화 전의 구성으로 복원된다. 예를 들어 ‘\_건강’, ‘\_을’과 같이 토큰화가 되었을 경우 ‘\_건강’에서 언더바를 제거하고 뒤에 오는 단어인 ‘\_을’을 합친 ‘건강을’이 원래의 구성이다. 언더바가 있

는 단어 뒤에 언더바가 있는 단어가 올 경우 이 두 단어는 결합의 여지가 없는 단독적인 단어로 토큰화된 것이다. 토큰화 후에 사전학습된 KoBERT[33]의 단어 사전을 통해 입력을 id로 바꿔주고 임베딩을 통해 768의 크기를 갖는 벡터로 임베딩한다.

지금까지 설명한 Word2Vec[7], RNN[8], LSTM[9], KoBERT[10]를 이용한 텍스트 임베딩 방법을 텍스트 네트워크로 구성하여 본 논문에서 제안한 텍스트-이미지 임베딩을 수행한다. 텍스트-이미지 임베딩 결과가 가장 좋은 텍스트 임베딩 방법을 본 논문의 텍스트 네트워크로 선정한다.

## 2. 이미지 네트워크

본 절에서는 제안하는 텍스트-이미지 임베딩 모델에서 이미지를 임베딩 하는 이미지 네트워크의 구성에 대해 기술한다. 다른 도메인 간 특징을 결합하여 데이터를 함께 분석하기 위한 연구가 활발하게 이뤄짐에 따라 일반적으로 성능이 좋다고 알려진 방법을 특징 추출기로 하는 사전학습된 모델이 다수 존재한다. 일반적으로 이미지 분류에서 좋은 성능을 내는 것으로 알려진 ResNet[11], Inception V3[12], EfficientNet[13]을 특징 추출기로 하는 사전학습된 모델을 이용하여 이미지 네트워크를 구성하고 그 결과를 통해 가장 성능이 좋은 모델을 본 논문에서 제안하는 텍스트-이미지 임베딩 모델의 이미지 네트워크로 선정한다.

파이썬 기반의 딥러닝 라이브러리인 Keras[35]에서 제공하는 사전학습된 ResNet, InceptionV3, EfficientNet 모델을 이용했다. 이는 1,000개의 클래스로 구성된 이미지넷의 데이터를 이용하여 학습된 모델이다. ResNet은 151개의 layer로 구성된 버전인 ResNet151를 사용했고 EfficientNet은 b7 버전을 사용했다. 표 5는 이미지 임베딩의 내용이다.

표 5. 사전학습된 ResNet151, InceptionV3, EfficientNet b7을 이용한 이미지 임베딩

사전 학습모델	input shape	output shape	이미지 임베딩
ResNet151	224, 224	1, 2048	[2.0103593 1.1456257 ... 3.3114839 3.5019507]
InceptionV3	299, 299	1, 2048	[0.9541025 2.0347521 ... 0.05512767 0.48773468]
EfficientNet b7	600, 600	1, 2560	[0.3998461 0.13490929 ... 0.01077013 0.0847197]

ResNet151의 이미지 입력 모양은 224 x 224이고 출력 크기는 2048이며, InceptionV3의 이미지 입력 모양은 229 x 299이고 출력 크기는 ResNet151과 같다. EfficientNet b7의 이미지 입력 모양은 600 x 600이고 출력 크기는 2560이다.

지금까지 기술한 내용은 이미지 임베딩은 이미지 하나를 임베딩하는 것에 해

당한다. 본 논문에서 사용하는 이미지 데이터인 스틸컷과 포스터 이미지의 개수는 영화마다 개수가 다르다. 예를 들어 스틸컷과 포스터의 개수가 23개인 영화가 있을 때 ResNet151로 이미지 특징을 추출하게 될 경우 출력 모양은 (이미지 개수 x 출력 크기)인 (23 x 2048)이 된다. 따라서 각 영화에 해당하는 이미지 특징을 1차원 벡터로 만들기 위해 각 벡터의 열을 기준으로 평균을 취해준다.



### 3. 텍스트-이미지 네트워크

텍스트-이미지 네트워크는 2절과 3절에서 설명한 결과인 텍스트 임베딩과 이미지 임베딩을 결합 및 학습하는 절차이다. 텍스트와 이미지 벡터를 concatenate를 통해 결합해준다. 그다음 개선된 Triplet Ranking Loss로 학습하기 위한 네트워크를 구축한다. 텍스트-이미지 네트워크는 1D Convolution과 1D MaxPooling, Dropout layer로 구성된 텍스트-이미지 모듈을 여러 개 배치하고 마지막에 Fully Connected Layer와 Flatten layer, L2-Normalization layer로 구성한다. 텍스트-이미지 모듈은 그림 16, 텍스트-이미지 네트워크는 그림 17과 같다.

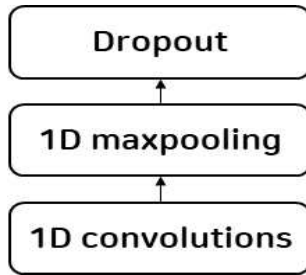


그림 16. 텍스트-이미지 모듈

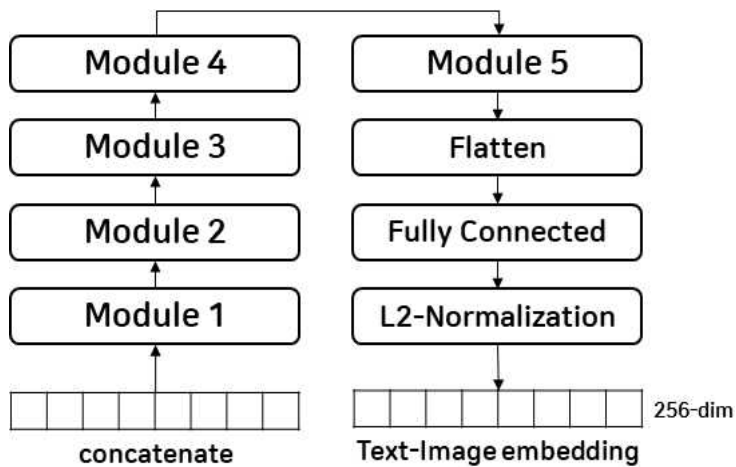


그림 17. 텍스트-이미지 네트워크

텍스트와 이미지 네트워크에서 임베딩한 결과를 concatenate로 결합하여 텍스트-이미지 네트워크의 입력으로 사용한다. 텍스트 임베딩과 이미지 임베딩 방법에 따라 출력되는 벡터의 크기가 모두 다르기 때문에 텍스트와 이미지를 결합한 concatenate의 벡터 크기 또한 다르다. 1D Convolution과 1D MaxPooling, Dropout layer로 구성된 모듈은 총 5개로 구성된다. 각 모듈의 구성은 같지만, Convolution의 kernel크기와 개수는 다르게 구성했고 MaxPooling의 크기와 strides는 2, Dropout은 0.3으로 설정했다. 표 6은 각 모듈의 kernel과 pooling의 설정 정보이다.

표 6. 각 모듈의 kernel과 pooling의 설정

	kernel size	num of kernel	kernel strides	pooling size	pooling strides
Module 1	32	128	1	2	2
Module 2	64	256	1	2	2
Module 3	128	512	1	2	2
Module 4	64	1024	1	2	2
Module 5	16	2048	1	2	2

모듈 뒤에 따르는 Flatten을 통해 각 Module의 kernel 개수에 따라 발생한 벡터들을 합쳐 1차원 벡터로 만들어 주고 Fully connected layer를 통해 모든 네트워크가 연결될 수 있도록 하고 출력의 크기를 256으로 조절해준다. 마지막으로 L2-Normalization을 통해 정규화를 수행한다. L2 정규화는 학습에서 발생하는 Loss의 범위가 너무 커질 경우 모델의 학습에 방해 요소가 되거나 과적합 되는 것일 수 있기 때문에 발생하는 Loss를 조절해주는 방법이다. L2-Normalization의 수식은 다음과 같다.

$$L_2 = \sqrt{\sum_i^n x_i^2} = \sqrt{x_1^2 + x_2^2 + x_3^2 + \dots + x_n^2} \quad (13)$$

수식 13은  $n$ 차원 좌표평면에서 벡터의 크기를 계산하는 방법이며, 유클리디언 거리 측정 공식과 같아서 유클리디언 정규화로 불리기도 한다. Loss에 대해 L2

정규화를 수행한다고 하면  $x_n$ 은  $n$ 번째에서 발생하는 Loss 값이다. 수식 13을 이용하여 발생하는 Loss 값들을 정규화하여 너무 큰 값이 나오더라도 그 값을 일정하게 조절해준다.

본 절에서 설명한 텍스트-이미지 네트워크를 통해 도출한 각 임베딩 벡터를 결합하고 텍스트-이미지 임베딩을 수행한다. 텍스트-이미지 임베딩에서 사용하는 제안하는 개선된 Triplet Ranking Loss는 다음 절에서 설명한다.

## 제2절 개선된 Triplet Ranking Loss

기존의 Triplet Ranking Loss[15, 23]는 거리 측정을 통해 도출된 값이 0 또는 음수일 때 아예 업데이트하지 않는 경우가 발생한다. 이런 상황이 빈번하게 발생하면 업데이트가 잘 안 되고 학습이 정상적으로 되지 않는 경우가 발생할 수 있다. 따라서 본 논문에서는 비선형 함수를 기반으로 하는 개선된 Triplet Ranking Loss를 제안한다.

개선된 Triplet Ranking Loss에서 개선에 중점을 두는 부분은 두가지이다. 첫 번째로, 거리 측정의 값이 음수일 경우도 업데이트가 가능하도록 하는 것이다. 음수의 값도 의미가 있는 것이 분명하기 때문에 업데이트에 활용하고자 한다. 두 번째는 거리 측정값이 너무 작은 음수 값으로 업데이트되지 않게 하는 것이다. 너무 작은 음수 값을 가질 경우 영향이 미비하여 정상적으로 업데이트되지 않고 오히려 연산량만 증가하여 학습에 방해가 될 수 있기 때문이다.

개선된 Triplet Ranking Loss에서 Anchor와 Positive, Anchor와 Negative의 거리를 구하는 방식은 일치하며, 거리 측정값에서 0 또는 양수인 최대값만 사용하는 부분을 비선형 함수인 sigmoid를 이용하여 변형했다. 식 15는 본 논문에서 제안하는 개선된 Triplet Ranking Loss이다.

$$Z = d(f(x_a), f(x_p)) - d(f(x_a), f(x_n)) \quad (14)$$

$$Loss = \frac{Z}{1 + e^{-Z}} = Z \cdot \sigma(Z) \quad (15)$$

식 14는 앞서 설명한 Triplet Ranking Loss에서 Anchor와 Positive, Anchor와 Negative의 거리를 구하는 방법에서 마진을 제외한 수식이며,  $x_a$ 는 Anchor,  $x_p$ 는 Positive,  $x_n$ 은 Negative,  $d(x)$ 는 유클리디안, 코사인 거리와 같은 거리 측정 함수로 수식의 구성도 같다. 기존의 Triplet Ranking Loss와 달리 음수를 고려하

기 위해 비선형 함수인 sigmoid를 곱하는 형태로 변형하였다. 제안한 Triplet Ranking Loss가 Anchor와 Positive, Anchor와 Negative의 거리가 음수일 경우 Loss에 미치는 영향을 그래프를 통해 설명한다. 그림 18은 기존의 Triplet Ranking Loss이고 그림 19는 개선된 Triplet Ranking Loss이다.

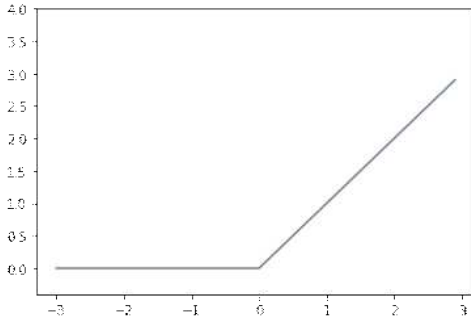


그림 18. 기존 Triplet Ranking Loss의 그래프

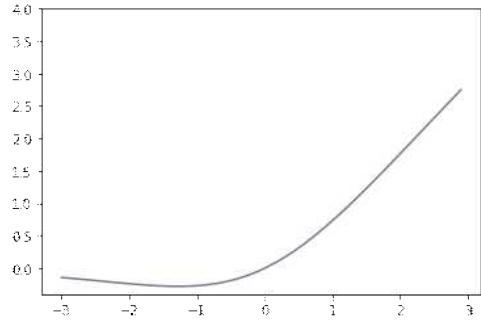


그림 19. 제안하는 Triplet Ranking Loss의 그래프

Triplet Ranking Loss의 값을 표현한 그래프인 그림 18에서 알 수 있듯  $Z$ 가 음수인 경우 아무 값도 고려하지 않지만 제안하는 Triplet Ranking Loss는  $Z$ 가 양수인 경우 기존 Triplet Ranking Loss와 비슷한 값을 업데이트하고 음수인 경우에도 어느 정도 값을 보존하면서 업데이트하는 것을 알 수 있다. 그뿐만 아니라 점점 더 작아지더라도 Loss 값이 함께 작아지는 것이 아니라 조금씩 증가하며 그 의미를 보존할 수 있을 뿐만 아니라 계속해서 작은 값으로 업데이트되는 것을 방지한다. 이는 그림 19에서 확인 할 수 있다.

콘텐츠의 유사도를 측정하기 위한 텍스트-이미지 임베딩 모델을 업데이트하기 위해 제안한 개선된 Triplet Ranking Loss를 이용한다.

## 제4장 실험 및 결과

본 장에서는 콘텐츠 분석에 텍스트와 이미지를 함께 사용하기 위한 데이터 셋 수집과 그 구성에 관해 기술하며, 앞서 설명한 텍스트-이미지 임베딩 모델 구성을 위한 실험과 제안한 모델의 성능을 보인다. 본 논문에서 실험한 환경으로 운영체제는 Ubuntu Server 18.04.5 LST, RAM은 128G, CPU는 Intel(R) Xeon(R) Silver 4210R CPU @ 2.40GHz 2개를 이용했고 GPU는 NVIDIA RTX A5000 4개, 프로그래밍 언어로 Python 3.7.5, 주요 라이브러리로 gensim 3.8.1, Pytorch 1.8.1, Tensorflow 2.7.0을 사용했다.

### 제1절 데이터 셋

본 논문에서 대상으로 하는 콘텐츠 데이터는 영화로 각 영화의 내용을 가장 잘 반영 할 수 있는 텍스트와 이미지 데이터를 수집하기 위한 방법을 기술한다. 영상 콘텐츠에서 핵심이 되는 부분을 키 프레임으로 추출하여 이미지로 사용하고 이미지의 설명문을 텍스트로 구성하는 것이 영화의 내용을 가장 잘 반영하는 텍스트-이미지 데이터 구성 방법으로 볼 수 있다. 하지만 이러한 내용의 공개된 데이터는 존재하지 않기 때문에 직접 구성해야 하며, 수작업으로 작업하는 것은 너무 많은 시간과 비용이 소모되기 때문에 불가능하다. 그뿐만 아니라 영상 콘텐츠의 키 프레임을 추출하는 것과 키 프레임의 이미지에 적합한 설명문을 생성하는 것은 각각의 연구 토픽으로 다뤄지고 있을 만큼 어려운 문제로 본 연구의 데이터 셋 구성에 활용하기는 어려운 부분이 존재한다. 따라서 본 논문에서는 포털 사이트에서 제공하는 영화 정보를 이용한다. 포털사이트에서 제공하는 내용은 정보 제공이 목적이며, 제작자와 협의로 제공되는 정보이기 때문에 수집 가능한 정보 중에서 영화의 내용을 가장 잘 반영한다고 할 수 있다.

데이터는 네이버 영화에서 2021년 8월 2일의 평점 순 영화랭킹의 2,000개를 타겟 영화로 한다. 유사한 영화는 타겟 영화를 대상으로 장르가 비슷하거나 같은 경우로 구분했으며, 네이버에서 제공하는 영화 정보 중 관련 영화를 참고했다. 하지만 네이버에서 제공하는 관련 영화 중에서도 비슷하지 않은 영화가 다수 포함되어 있기 때문에 이를 구분하기 위해 관련 영화 중에서 장르가 같은 영화만 유사한 영화로 정의했다. 비슷한 장르끼리 묶어 같은 장르로 취급하여 10개로 구분했다. 표 7은 구분한 장르를 나타낸다.

표 7. 타겟 영화와 유사한 영화를 구분하기 위해 정의한 장르

드라마, 멜로/애정/로맨스	판타지, SF	애니메이션	모험	가족
공포, 스릴러, 미스터리	느와르, 전쟁, 액션	다큐멘터리	코미디	뮤지컬

수집한 영화 데이터 중 포스터와 스틸컷의 개수가 20개 이하이거나 아예 존재하지 않는 영화는 데이터 셋에서 제외했고 타겟 영화와 유사한 영화는 4,978개를 수집했다. 수집하는 데이터는 영화 제목, 영화 ID, 줄거리, 포스터, 스틸컷이다. 그림 20은 타겟 영화와 유사한 영화 수집 방법을 보인다.

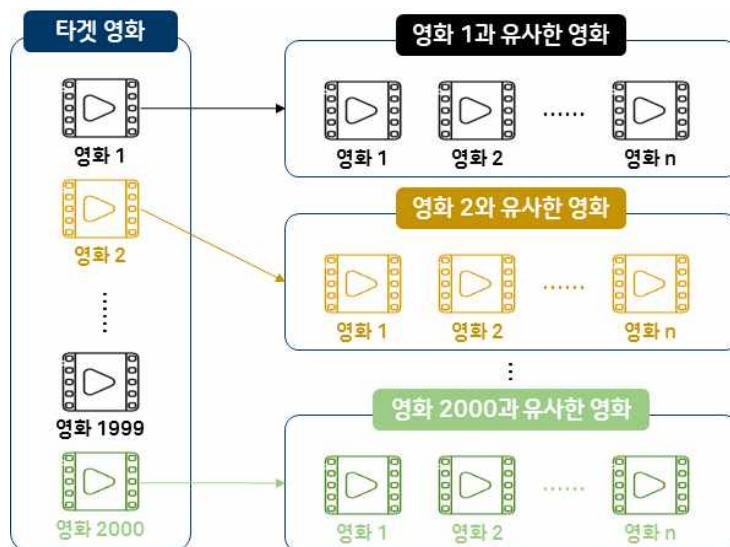


그림 20. 타겟 영화와 유사한 영화의 수집 방법

데이터 수집에는 파이어폭스, 인터넷 익스플로러, 크롬 등과 같은 인터넷 브라우저를 컨트롤 할 수 있는 파이썬 라이브러리인 Selenium을 활용했다.

수집한 데이터에서 영화 ID-영화 제목, 영화 ID-줄거리는 csv파일 형태로 구성했고 포스터와 스틸컷은 url을 수집한 후 이미지를 다운하여 영화 ID 폴더-포스터, 스틸컷 이미지로 구성했다. 그림 21과 그림 22는 영화 ID-영화 제목과 영화 ID-줄거리 csv파일의 일부이며, 그림 23은 영화 ID 폴더-포스터, 스틸컷 이미지의 일부이다.

	10117	라퐁
1	205285	와인 패밀리
2	17796	알 포스티노
3	11315	달콤 쌉사름한 초콜릿
4	31231	말레나
5	101248	베스트 오퍼
6	19127	스타 메이커
7	186613	작은 아씨들
8	29309	달타냥의 딸
9	39584	코러스
10	72007	오션스
11	64170	연노은 우연
12	29059	파아니스트의 전설
13	152655	달링
14	146372	시크릿 레터
15	38835	티미널

그림 21. 영화ID-영화 제목 데이터셋의 일부

	10117	13살 소녀, 남만의 도시 파리에 오다! 파리에 전학 온 첫 날, 13살 소녀 백(소피 마르
1	205285	일만 하며 바쁘게 살아온 캐나다의 자동차 회사 CEO 마크. 문득 남은 인생을 어떻게
2	17796	작은 섬 칼라 디스토에 오게 된 시인 네루다, 어부의 아들 마리오는 그의 도착으로
3	11315	페드로(Pedro. 마르코 레오나르디 분)와 리타(Rita. 루미 카바조스 분)는 서로 사랑하는
4	31231	2차 대전이 한창인, 햇빛 찬란한 지중해의 작은 마을. 매력적인 말레나. 걸어갈 때면
5	101248	최고가로 미술품을 낙찰시키는 세기의 경매사이자 예술품의 가치를 알아보는 완벽하
6	19127	2차대전이 종전되고 패전국이 된 이탈리아의 생활은 말이 아니다. 모두가 희망도 없
7	186613	배우가 되고 싶은 첫째 메그(엠마 왓슨) 작가가 되고 싶은 둘째 조(시얼샤 로넌) 음악
8	29309	<삼총사>에 등장하는 달타냥의 딸 엘로이즈가 아버지의 뒤를 이어 왕위 찬탈을 노
9	39584	2차 세계대전 직후 프랑스 작은 기숙사 학교 토요일마다 하염없이 야박을 기다리는
10	72007	즐거움, 환상, 신비로움이 늘 함께하는 바다! 지금껏 단 한번도 보지 못한 새로운 바다
11	64170	절고 부유한 보석 세공사인 아다처 부인의 집에 들어가기 위해 기존의 가정부를 사
12	29059	1900년, 유럽과 미국을 오가는 버지니아 호에서 태어나 평생을 바다 위에서 살아온
13	152655	세상 부러울 것 없는 아름다운 귀족 로빈과 다이애나. 하지만 바이러스 감염으로 로
14	146372	죽음만큼 사랑했던 연인, 어느 날 갑자기 사라져 버린 연인. 세상은 그를 죽었다 한다
15	38835	동유럽 작은 나라 '크로코지아'의 평범한 남자 빅터 나보스키(통 윙크스). 뉴욕 일성의

그림 22. 영화ID-줄거리 데이터셋의 일부

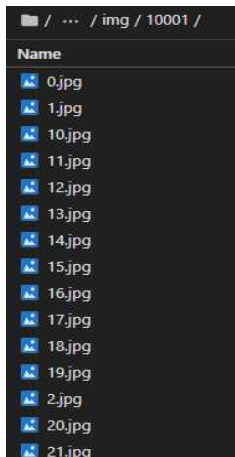


그림 23. 영화ID 폴더-포스터, 스틸컷 데이터셋의 일부



수집한 데이터셋은 학습과 테스트에 활용하기 위해 train, test, validation 셋으로 나누었다. train 셋은 학습에 사용하고 test 셋은 학습된 모델의 성능을 검증할 때 사용하며, validation 셋은 모델의 학습이 잘 되었는지 확인하기 위한 셋이다. validation 셋을 통해 학습하는 과정에서 발생할 수 있는 과적합과 적절한 하이퍼파라미터, 모델의 네트워크 및 layer를 찾을 수 있다. 그림 24는 validation 셋을 이용하여 모델의 성능을 개선하기 위한 방법에 대한 설명이다.

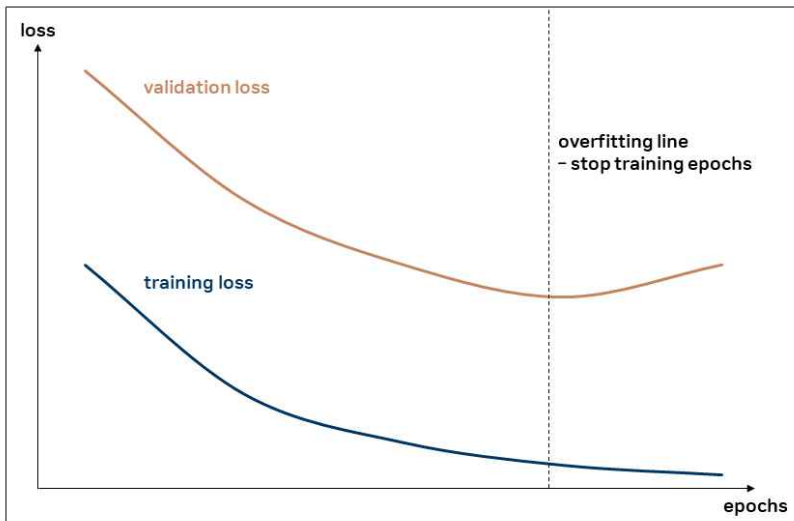


그림 24. validation 셋을 이용한 모델 성능 개선 방법

그림 24에서 training loss는 train 셋을 이용하여 모델을 학습할 때 발생하는 loss 곡선이고 validation loss는 train 셋을 통해 학습된 모델을 validation set으로 학습할 때 발생하는 곡선이다. training loss만 봤을 때는 loss가 점점 감소하며 모델이 정상적으로 학습되는 것처럼 보이지만 validation loss를 봤을 때는 어느 시점 이후 loss가 다시 증가하는 것을 알 수 있다. 이 시점을 기준으로 과적합으로 판단할 수 있다. 과적합은 모델이 train 셋에 대해 과도하게 학습되는 것으로 다른 data에서는 합리적인 성능을 내지 못하는 상태이다. 따라서 과적합 발생 직전의 epochs까지만 학습하는 것이 모델의 성능이 최적의 상태가 된다. 뿐만 아니라 위 그림과 같은 최적의 곡선의 상태를 찾기 위해 모델의 하이퍼파라미터와 네트워크를 구성하는 layer를 조절한다.

따라서 본 논문에서는 train, test, validation 셋을 활용하기 위해 타겟 영화와 유사한 영화에서 각 10%를 validation 셋으로 하고 그 나머지 유사한 영화와 타겟 영화를 train 셋과 test 셋으로 구성했다. 그림 25는 수집한 데이터 셋을 나타낸다.

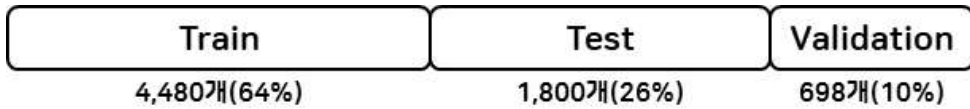


그림 25. 실험에 사용하는 데이터셋의 구성

전체 데이터 셋에서 train 셋은 4,480개, test 셋은 1,800개, validation 셋은 698개로 구성했으며, 전체 데이터 셋에서 차지하는 비율은 각각 64%, 26%, 10%이다.

## 제2절 텍스트-이미지 임베딩 모델의 성능 평가

본 절에서는 앞서 설명한 텍스트-이미지 임베딩 모델의 성능 평가를 수행한다. 텍스트 네트워크를 구성할 수 있는 Word2Vec, RNN, LSTM, KoBERT와 이미지 네트워크를 구성할 수 있는 ResNet151, InceptionV3, EfficientNet b7을 사용하여 가장 성능이 좋은 임베딩 방법을 찾는다. 또한, 가장 성능이 좋은 텍스트와 이미지 임베딩 방법을 제안하는 텍스트-이미지 임베딩 모델의 텍스트와 이미지 네트워크에서 사용하고 기존의 텍스트-이미지 임베딩 방법과 성능을 비교하여 본 논문에서 제안한 방법의 타당성을 보인다.

가장 성능이 좋은 텍스트와 이미지 네트워크의 임베딩 방법을 찾기 위해 텍스트 네트워크에서 사용할 Word2Vec, RNN, LSTM, KoBERT와 이미지 네트워크에서 사용할 ResNet151, InceptionV3, EfficientNet b7을 사용 가능한 모든 조합으로 구성한다. 텍스트와 이미지 네트워크에 뒤따르는 텍스트-이미지 네트워크는 앞서 설명한 5개의 텍스트-이미지 모듈과 Flatten layer, Fully connected layer, L2-Normalization으로 구성된다.

텍스트-이미지 임베딩 방법에 따른 성능을 평가하기 위해 Recall을 척도로 사용한다. Recall은 검색, 추천 시스템에서 주로 사용되며, 모델의 성능을 평가하는 척도 중 하나이다. True Positive, False Positive, False Negative, True Negative의 경우의 수를 갖는 Confusion Matrix를 이용하는 척도이다. Confusion Matrix는 그림 26과 같다.

Confusion Matrix		Real Answer	
		True	False
Model Predict	True	True Positive	False Positive
	False	False Negative	True Negative

그림 26. Recall 설명을 위한 Confusion Matrix

Recall은 실제 정답이 True인 것 중에서 모델 예측이 True라고 예측한 것의 비율을 나타낸다. 따라서 True Positive와 False Negative의 합 중에서 True Positive의 비율이다. Recall은 모델이 예측한 결과가 얼마나 정확한지 파악할 수 있는 지표이다. Recall의 수식은 16과 같다.

$$Recall(R) = \frac{TP}{TP+FN} \quad (16)$$

수식 16에서  $TP$  는 True Positive이고  $FN$  은 False Negative이다. 검색, 추천 시스템은 하나의 결과를 추천하는 것보다 여러 개의 결과를 추천하는 것을 필요로 한다. 이럴 경우 Recall을 이용한 Recall at k를 이용하며, Recall@k로 표현한다. 모델이 k개의 정답을 예측할 때 실제 정답이 몇 개인지를 나타내는 지표이며, k개의 예측 중에 모델의 예측이 정답인 개수 / 모델이 예측 가능한 정답의 개수로 계산할 수 있다. 예를 들어 k가 20이고 모델이 예측 가능한 정답이 20개, 20개의 예측 중에서 모델의 예측이 정답인 개수가 10개라면  $10 / 20$ 이므로 50%의 Recall을 갖는다. 본 논문에서는 이와 같은 방법으로 테스트 셋인 해당 영화 1,800개의 Recall@k를 구해서 평균을 취한다.

## 1. 모델 구성을 위한 텍스트와 이미지 임베딩 성능 평가

제안하는 텍스트-이미지 임베딩 모델에서 텍스트와 이미지 네트워크를 구성하는 임베딩 모델을 선정하기 위해 각 임베딩 방법을 네트워크로 구성하여 실험을 진행한다. 가장 성능이 좋은 임베딩 방법을 도출하고 제안하는 텍스트-이미지 임베딩 모델의 텍스트와 이미지 네트워크의 임베딩 방법으로 선정한다. 텍스트 네트워크에 사용하는 임베딩 방법은 Word2Vec, RNN, LSTM, KoBERT이며, 이미지 네트워크에 사용하는 임베딩 방법은 ResNet151, InceptionV3, EfficientNet b7이다. 텍스트와 이미지 네트워크 다음에는 텍스트-이미지 네트워크로 구성한다. 표 8은 학습에 사용한 하이퍼파라미터를 나타낸다.

표 8. 학습에 사용한 하이퍼파라미터

optimizer	learning rate	activation function	loss	epochs	batch size	dropout	normalization
Adam	0.0003	ReLU	제안한 Triplet Ranking Loss	200	512	0.3	L2

표 8에 기술된 하이퍼파라미터를 통해 제안한 텍스트-이미지 임베딩 모델을 학습하였으며, 텍스트와 이미지 임베딩 방법에 따라 학습한 결과는 recall@1, recall@10, recall@20으로 평가한다. 표 9, 표 10, 표 11은 recall@1, recall@10, recall@20으로 각 임베딩 방법을 성능 평가한 결과이다.

표 9. 각 임베딩 방법에 따른 Recall@1의 결과

	Word2Vec	RNN	LSTM	KoBERT
ResNet151	21.8	31.7	35.5	36.8
InceptionV3	25.1	34.4	39.1	39.3
EfficientNet b7	29.5	40.9	39.9	<b>41.5</b>

표 10. 각 임베딩 방법에 따른 Recall@10의 결과

	Word2Vec	RNN	LSTM	KoBERT
ResNet151	44.5	51.1	51.7	57.4
InceptionV3	41.4	51.6	52.5	61.1
EfficientNet b7	50.9	52.5	55.1	<b>62.8</b>

표 11. 각 임베딩 방법에 따른 Recall@20의 결과

	Word2Vec	RNN	LSTM	KoBERT
ResNet151	59.9	68.9	72.2	75.2
InceptionV3	61.4	76.1	75.3	78.8
EfficientNet b7	66.4	72.6	79.1	<b>81.9</b>

텍스트와 이미지 네트워크 선정을 위해 실험한 결과 Recall@1에서 RNN-EfficientNet b7, LSTM-InceptionV3, LSTM-EfficientNet b7, KoBERT-InceptionV3, KoBERT-EfficientNet b7이 거의 비슷한 성능을 보였지만 KoBERT-EfficientNet b7이 41.5%로 가장 좋은 성능을 보였다. Recall@1은 모델이 추천 가능한 항목 중에서 1개의 항목만 추천했을 때 정답을 추천할 확률이다.

Recall@10에서는 KoBERT-InceptionV3, KoBERT-EfficientNet b7이 비슷한 성능을 보였지만, KoBERT-EfficientNet b7이 62.8%로 가장 좋은 성능을 보였다. Recall@10은 모델이 추천 가능한 항목 중에서 10개의 항목을 추천했을 때 정답을 추천할 확률이다.

Recall@20에서는 LSTM-EfficientNet b7, KoBERT-InceptionV3, KoBERT-EfficientNet b7이 비슷한 성능을 보였지만 KoBERT-EfficientNet b7이 81.9%로 가장 좋은 성능을 보였다. Recall@20은 모델이 추천 가능한 항목 중에서 20개의 항목을 추천했을 때 정답을 추천할 확률이다.

Recall@1, Recall@10, Recall@5를 통해 성능을 검증한 결과 세 지표 모두 KoBERT-EfficientNet b7을 이용하는 것이 가장 좋은 것으로 나타났다. 따라서 본 논문에서 제안하는 텍스트-이미지 임베딩 모델의 텍스트 네트워크와 이미지 네트워크의 임베딩 방법으로 KoBERT와 EfficientNet b7을 사용한다.

## 2. 제안한 텍스트-이미지 임베딩 모델의 성능 평가

본 논문에서 제안한 방법과 기존 방법의 성능을 비교 평가한다. 비교 평가에 사용되는 기존 방법은 ① 학습하지 않는 텍스트-이미지 임베딩 방법, ② K-Nearest Neighbor(K-NN)[36], ③ Contrastive Loss[25], ④ Triplet Ranking Loss-Hard Negatives[23], ⑤ Triplet Ranking Loss-SemiHard Negatives[23], ⑥ VSE++[22]이다.

첫 번째 비교 방법인 ① 학습하지 않는 텍스트-이미지 임베딩 방법은 본 논문에서 사용하는 텍스트 네트워크인 KoBERT와 이미지 네트워크인 EfficientNet b7으로 구성은 같으나 텍스트 임베딩과 이미지 임베딩을 concatenate 한 후 Fully Connected Layer를 통해 256의 크기를 갖는 벡터로 텍스트-이미지 임베딩만 수행하며, 학습은 하지 않는다. 학습하지 않는 텍스트-이미지 임베딩 방법의 구성은 그림 27과 같다.

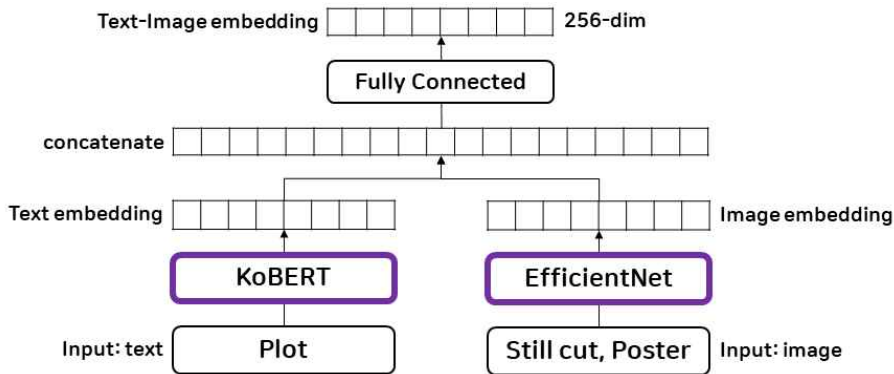


그림 27. 학습하지 않는 텍스트-이미지 임베딩 방법

두 번째 방법인 ② K-NN[36]은 지도학습 방법 중 하나로 기존 데이터를 기반으로 입력 데이터 주변의 데이터를 살펴보고 더 많은 데이터가 포함되어있는 카테고리로 분류하는 방법이다. 입력 데이터의 카테고리를 분류하기 위해 주변의 데이터를 몇 개 살펴볼 것인지를 변수 K를 통해 결정하며, 이는 사용자가 임의로 설정할 수 있다. 일반적으로 K는 홀수로 설정하여 어떤 경우에도 의사결정을



내릴 수 있도록 하며,  $K$ 의 값이 너무 클 경우 노이즈에 강하지만 분류의 경계가 명확하지 않을 수 있고  $K$ 의 값이 너무 작을 경우 노이즈에 민감해질 수 있다. 그림 28은  $K$ -NN[36]에서  $K$ 의 값이 3인 예이다.

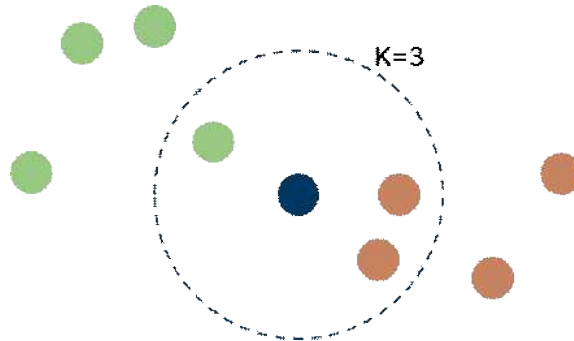


그림 28.  $K$ 가 3인  $K$ -NN의 의사결정 예

그림 28에서  $K$ 는 3이고 입력 데이터는 파란색이다. 입력 데이터 주변 데이터 3개를 살펴보면 주황색 데이터 2개, 초록색 데이터 1개이기 때문에 입력 데이터는 주황색 데이터와 같은 카테고리로 결정된다.

$K$ -NN[36]은 모든 데이터에 대한 거리를 측정하기 때문에 각 데이터의 차원이 너무 클 경우 연산이 오래 걸리고 제대로 된 분류가 수행되지 않을 수 있다. 따라서, 본 논문의 실험에서 텍스트 임베딩과 이미지 임베딩을 concatenate 한 후 PCA를 통해  $x$ ,  $y$  축의 값을 갖도록 차원 축소한 결과를  $K$ -NN[36]에 이용했다.  $K$ 는 11로 설정했다.

세 번째 방법인 ③ Contrastive Loss[25]는 네 번째와 다섯 번째 성능 비교 방법인 Triplet Ranking Loss[23]가 제안되기 이전에 사용되었던 방법이다. Triplet Ranking Loss[23]는 Anchor와 Positive, Negative를 동시에 고려하지만, Contrastive Loss[25]는 Anchor와 Positive, Anchor와 Negative를 각각 고려하며, Margin을 기준으로 Positive와 Negative의 경계를 설정한다. 그림 29는 Contrastive Loss[25]의 학습 예이다.

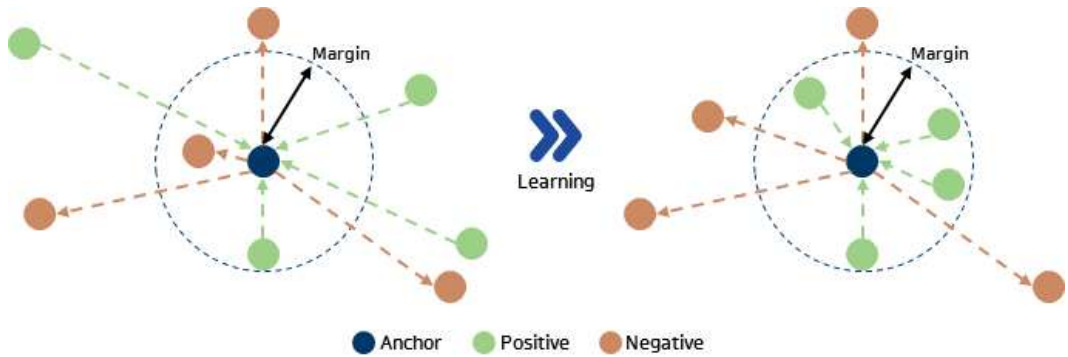


그림 29. Contrastive Loss의 학습 예

그림 29에서 왼쪽 그림은 Anchor와 Positive, Negative가 배치된 예를 보이며, Margin과 관계없이 데이터가 배치된 것을 알 수 있다. 오른쪽 그림은 학습 후의 모습이며, Anchor를 기준으로 Margin의 경계에 따라 Positive와 Negative가 적절히 배치된 것을 알 수 있다.

Contrastive Loss[25]는 Margin을 통해 Anchor와 Positive, Negative의 경계만 나눌 뿐 Positive와 Negative의 관계는 고려하지 않는다. Contrastive Loss[25]에서 Anchor와 Positive, Negative 사이의 거리 측정은 앞서 설명한 유클리디안 거리와 같다. 수식 17과 수식 18은 Contrastive Loss 함수[25]이다.

$$Z = d(f(x_a), f(x_k)) \quad (17)$$

$$Loss = (1 - Y) \frac{1}{2} (Z_w)^2 + (Y) \frac{1}{2} \{\max(0, m - Z_w)\}^2 \quad (18)$$

수식 17에서  $x_a$ 는 Anchor이고  $x_k$ 는 다른 대상이며,  $x_k$ 는 Positive 또는 Negative가 될 수 있다.  $d(i, j)$ 는  $i$ 와  $j$ 의 거리를 나타내며,  $d$ 는 거리 측정 함수로 유클리디언 또는 코사인 거리와 같은 방법이 될 수 있다.

수식 18에서  $Y$ 는  $x_a$ 와  $x_k$ 에 대한 판별자로  $x_k$ 가 Positive일 경우 0, Negative일 경우 1의 값을 갖는다. 따라서  $x_k$ 가 Positive일 경우 좌측 향으로 Loss 값이 업데이트되고 Negative일 경우 우측 향으로 Loss 값이 업데이트된다.  $1/2$ 은 정규화를 위함이며,  $Z$ 는 Anchor와  $x_k$ 의 거리,  $Z_W$ 는  $Z$ 에 대한 가중치,  $m$ 은 마진이다. 좌측 향은 Positive인  $x_k$ 가 Anchor인  $x_a$ 와 점점 가까워지도록 Loss 값이 업데이트 된다. 우측 향은 Negative인  $x_k$ 가 마진의 값보다  $x_a$ 와 가까이 있으면 마진의 밖으로 업데이트하고, 마진의 값보다 멀리 있으면 Loss 값을 0으로 설정하여 업데이트하지 않는다. 본 논문에서 제안한 텍스트-이미지 임베딩 구조에 Contrastive Loss 함수를 적용하여 실험한다.

네 번째와 다섯 번째 방법인 ④ Triplet Ranking Loss-Hard Negatives[23]와 ⑤ Triplet Ranking Loss-SemiHard Negatives[23]는 본 논문에서 제안하는 텍스트-임베딩 모델과 같은 구조를 갖지만 Loss 함수만 Triplet Ranking Loss의 Hard Negatives와 Semi-Hard Negatives를 사용한다.

여섯 번째 방법인 ⑥ VSE++[15]는 기존의 텍스트-이미지 임베딩 방법으로 텍스트 임베딩 방법은 GRU를 사용하고 이미지 임베딩 방법은 ResNet151을 사용하며, 각 임베딩을 concatenate하고 Fully Connected Layer를 통해 256의 크기를 갖는 벡터를 출력한다. 그림 30은 VSE++ 모델의 구조이다.

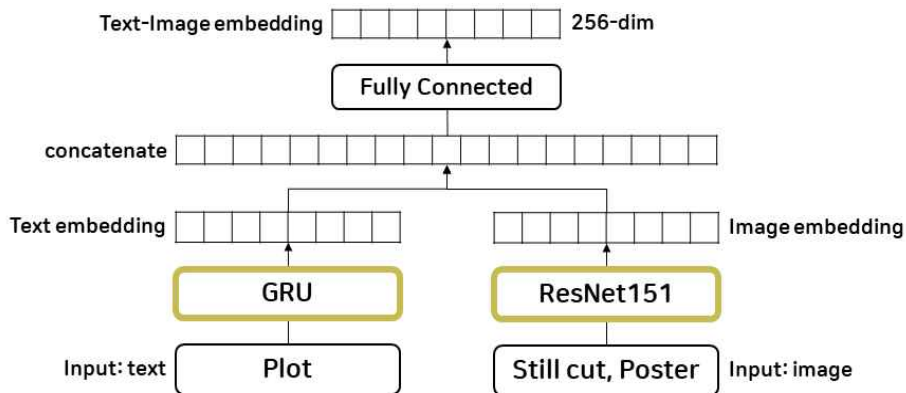


그림 30. VSE++ 모델의 구조

VSE++[15] 모델을 학습하는데 사용하는 함수는 Hard Negatives를 기반으로 하는 Triplet Ranking Loss이다. 수식 17은 VSE++에서 사용하는 Loss 함수이다.

$$l_{MH}(i,d) = \max[\alpha + s(i,\hat{c}) - s(i,c)]_+ + \max[\alpha + s(\hat{i},c) - s(i,c)]_+ \quad (17)$$

수식 17에서  $\alpha$ 는 마진,  $[x]_+$ 는  $\max(x,0)$ 이다.  $i$ 와  $c$ 는 관련 있는(Positive) 이미지와 쿼리이다.  $\hat{i}$ 와  $\hat{c}$ 는 관련 없는(Negative) 이미지와 쿼리이다.  $s(x,y)$ 는 데이터  $x$ 와  $y$ 가 있을 때  $x$ 와  $y$ 의 유사점수이다. 수식 17에서 이미지와 쿼리 쌍인  $(i,c)$ 가 주어졌을 때 Loss 함수의 결과는 관련 없는(Negative) 쌍인  $s(i,\hat{c})$ 와  $s(c,\hat{i})$ 의 최대값만 취하게 된다. 따라서 최소값은 무시하게 되고 최대값만 업데이트에 사용한다.

제안한 텍스트-이미지 임베딩 모델의 성능을 비교하기 위해 앞서 설명한 ① 학습하지 않는 텍스트-이미지 임베딩 방법, ② K-NN[36], ③ Contrastive Loss[25], ④ Triplet Ranking Loss-Hard Negatives[23], ⑤ Triplet Ranking Loss-SemiHard Negatives[23], ⑥ VSE++[15]와 Recall@20으로 성능을 비교한다. 그림 31은 제안한 모델과 기존의 방법과의 성능을 비교한다.

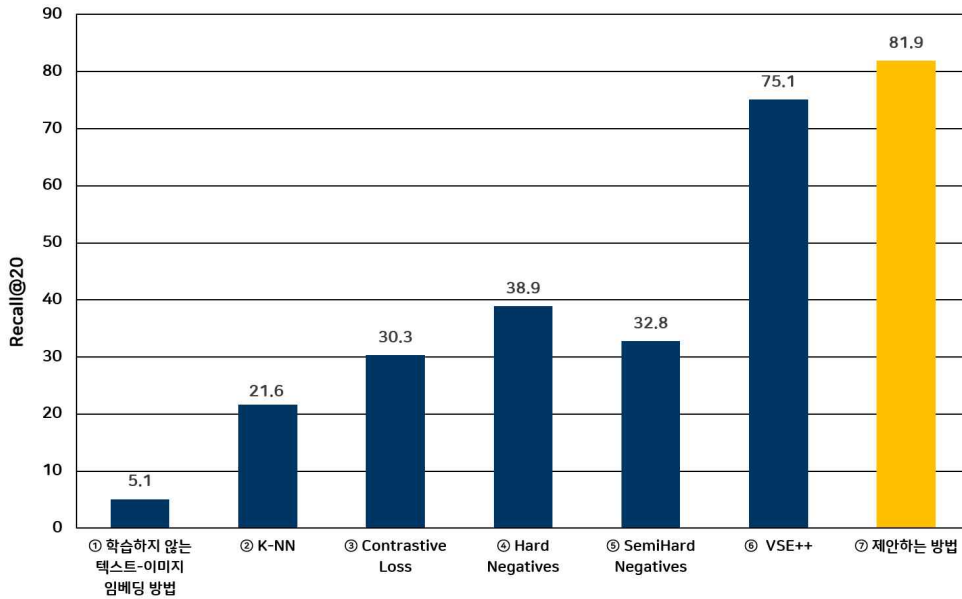


그림 31. 제안하는 방법과 기존 방법의 성능 평가(Recall@20)

각 방법의 성능 평가 결과 ① 학습하지 않는 텍스트-이미지 임베딩 방법이 5.1, ② K-NN 방법이 21.6, ③ Contrastive Loss 방법이 30.3, ④ Triplet Ranking Loss-Hard Negatives 방법이 38.9, ⑤ Triplet Ranking Loss-SemiHard Negatives 방법이 32.8, ⑥ VSE++ 방법이 75.1, 본 논문에서 제안하는 텍스트-이미지 임베딩 방법이 81.9를 보인다. 이를 통해 제안하는 방법이 가장 좋은 성능을 보이는 것을 알 수 있다.

① 학습하지 않은 텍스트-이미지 임베딩 방법은 최초로 본 논문의 방법론을 설계하고 테스트 실험을 진행하는 과정에서 시각화 결과가 꽤 유의미하게 도출되었다. 비슷한 위치에 배치된 영화를 실제로 확인해 보았을 때 유사한 경우가 상당수 발견되어 학습하지 않고 영화의 텍스트와 이미지 정보를 임베딩 한 경우도 좋은 결과가 나올 수 있을 것 같다는 기대를 했다. 따라서 실제 성능 평가 방법으로 검증해 보기 위해 비교 실험에 학습하지 않은 텍스트-이미지 임베딩 방법을 포함했다. 하지만 Recall을 통한 성능 평가 결과는 좋지 않은 것으로 도출됐다. 사전학습된 텍스트와 이미지 임베딩 모델을 통해 각 데이터를 임베딩하고 결합한 후 벡터의 크기를 줄이기만 했기 때문으로 보인다.

② K-NN 방법은 알려진 대로 학습을 통해 모델을 구축하는 것이 아니라 기존 데이터를 기반으로 입력 데이터 주변의 데이터를 통해 카테고리를 분류하는 방법이므로 클래스 간 관계를 이해하는 것이 제한적이고 텍스트와 이미지의 특징을 고려하지 않기 때문에 좋은 결과로 도출되지 않은 것으로 보인다.

③ Contrastive Loss, ④ Triplet Ranking Loss-Hard Negatives, ⑤ Triplet Ranking Loss-SemiHard Negatives, ⑥ VSE++는 벡터화된 텍스트나 이미지의 특징으로부터 학습을 수행하는 Metric Learning의 종류로 볼 수 있다.

③ Contrastive Loss의 성능 평가 결과 비교적 낮은 성능을 보이지만 ⑤ Triplet Ranking Loss-SemiHard Negatives와 비슷한 성능을 보인다. ⑥ Triplet Ranking Loss-Hard Negatives 방법과 ⑤ Triplet Ranking Loss-SemiHard Negatives 방법은 본래 이미지 도메인에 대한 관계를 학습하기 위해 정의된 Loss 함수이다. 따라서, 한 종류의 도메인에서는 좋은 결과를 보이지만 다른 도메인을 함께 학습하는 데는 다소 부적절한 것으로 보인다. 또한, SemiHard Negatives가 Hard Negatives보다 좋지 않은 성능을 보이는데 이는 학습 과정이 Hard Negatives보다 복잡한 조건으로 학습을 하기 때문에 그에 따라 발생하는 Loss 값 또한 일률적이지 못해서 모델 학습에 영향을 끼친 것으로 보인다.

④ VSE++ 방법은 텍스트-이미지 임베딩에 사용되는 모델로써 다른 기존의 방법 보다는 좋은 결과가 나왔다. 학습 방법은 Metric Learning으로 제안하는 방법과 유사하지만 VSE++의 작업 목적은 텍스트-이미지 검색 또는 이미지-텍스트 검색이기 때문에 본 논문의 작업 목적인 콘텐츠 간 유사도 측정과는 다른 점이 있다.

제안하는 방법은 비교군인 기존의 방법들보다 좋은 성능을 보였다. 그에 따라, 본 논문의 목적인 콘텐츠의 유사도 측정을 위한 텍스트-이미지 임베딩 모델과 모델 학습을 위한 개선된 Triplet Ranking Loss는 목적에 적합하게 잘 설계 및 학습되었다고 할 수 있다.

### 제3절 텍스트-이미지 임베딩 시각화 및 추천 방법

본 절에서는 앞서 설명한 제안하는 텍스트-이미지 임베딩 모델로 테스트 데이터를 시각화하여 임베딩 결과를 확인하고 유사한 것으로 보이는 일부의 콘텐츠를 확인한다. 시각화를 위해 텍스트-이미지 임베딩 결과를 PCA로 차원 축소했다. 텍스트-이미지 임베딩 결과의 벡터 크기를 256에서 2로 축소하고 이를 x, y 좌표로 하여 좌표평면상에 나타낸다. 그림 32는 테스트 데이터 셋을 시각화한 모습이다.

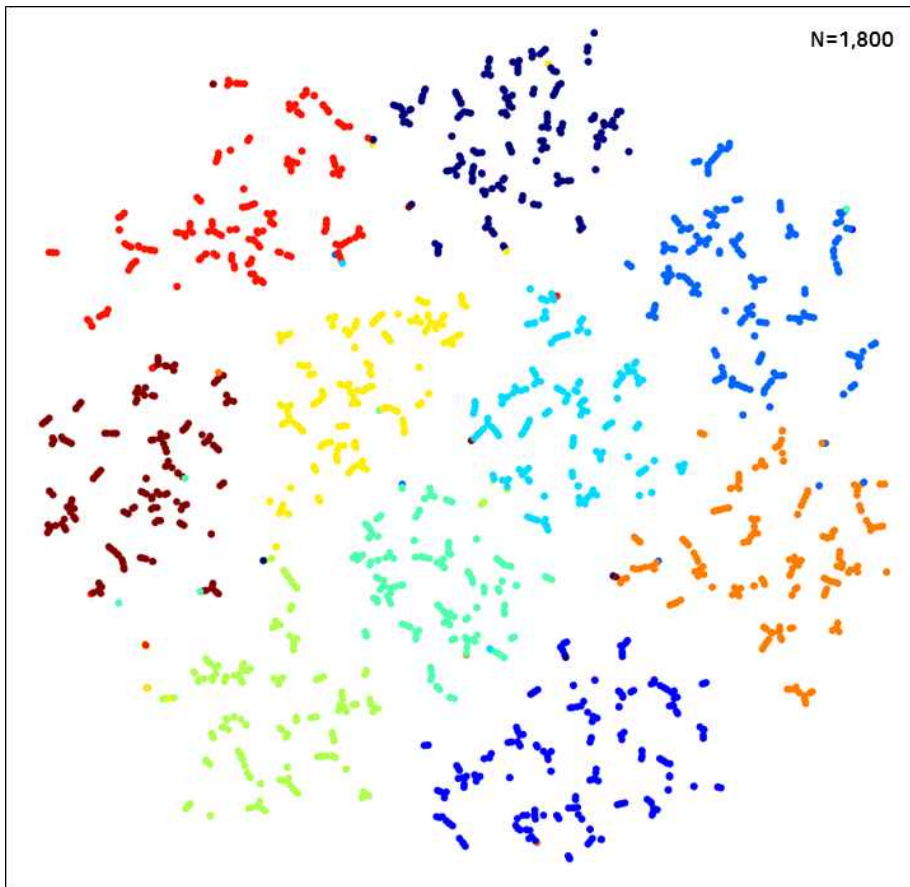


그림 32. 텍스트-이미지 임베딩의 시각화 결과

그림 32의 점은 테스트 셋인 1,800개의 영화이며, 장르를 기준으로 한 유사한 영화의 색이 같다. 이는 데이터 구성에서 유사한 영화를 수집할 때 포털사이트에서 제공하는 관련 영화 중 장르가 같은 영화를 수집했기 때문에 발생한 결과로 보인다.

학습된 텍스트-이미지 임베딩 모델을 이용하여 테스트 데이터 셋을 임베딩하고 차원 축소를 통해 시각화한 결과는 비교적 유사한 영화끼리 잘 군집 되어 있는 것을 볼 수 있다. 시각화했을 때 비교적 군집이 잘 되었다는 것은 본 논문에서 제안하는 텍스트-이미지 임베딩 모델이 잘 학습되었고 그 결과 또한 우수하다고 할 수 있다.

하지만 드문드문 잘못 군집 된 영화들이 존재하는 것을 볼 수 있다. 이러한 현상은 다른 장르이고 유사하지 않은 영화임에도 플롯의 내용과 스틸컷, 포스터의 내용이 유사한 경우 발생한 것으로 보인다. 또한 유사한 영화를 수집할 때 포털사이트에서 제공하는 관련 영화 중 유사하지 않음에도 등장하는 영화가 일부 수집된 것으로 보인다.

제안하는 영상 콘텐츠의 유사도 측정을 위한 텍스트-이미지 임베딩 방법은 콘텐츠의 내용을 직접 분석한 결과이기 때문에 사용자가 시청한 콘텐츠를 기반으로 그와 유사한 영상 콘텐츠를 정확하게 추천할 수 있다. 이와 같은 방법으로 추천하는 방식은 기존의 OTT에서는 사용하지 않는 방식일 뿐만 아니라 기존 추천 시스템 연구에서도 미비한 부분이었다.

그뿐만 아니라, OTT 사용 만족도에 큰 영향을 끼치는 추천 카테고리 명과 주제에 대한 근거를 명확하게 제시함으로써 사용자의 서비스 이용 만족도를 개선할 수 있을 것으로 보인다.



## 제5장 결론 및 제언

스마트폰과 태블릿PC 같은 스마트 기기의 보유율이 꾸준히 증가하고 코로나의 여파로 인해 최근 몇 년 사이에 스마트 기기를 이용한 콘텐츠 이용률과 이용 시간이 크게 증가하고 있다. 이에 따라 OTT를 이용하여 콘텐츠를 시청하는 사용자가 느끼는 이용 만족도와 지속 사용 의도에 관한 연구도 활발하게 진행되고 있다. 사용자가 OTT를 이용할 때 느끼는 만족도와 지속 사용 의도와 연관된 요소로는 콘텐츠 다양성, 요금제 적절성, 추천 시스템, N스크린 서비스, 몰아보기 기능 등이 있으나 그 중 추천 시스템이 가장 큰 요인으로 분석되었다.

그뿐만 아니라 OTT의 콘텐츠 추천에 따른 서비스 이용에서 저항을 느끼게 되는 요인에 관한 연구에서는 추천 정확성에 따라 사용자의 만족도에 긍정적인 평가와 신뢰감이 증가하고 그에 따라 지속적인 이용에 긍정적인 영향을 줄 수 있다는 분석을 했다.

앞서 기술한 것처럼 추천 시스템은 사용자의 만족도 및 지속사용 의도에 중요한 요소로 작용하지만, 대부분의 추천 시스템은 사용자에게 메타 데이터를 강요하거나 화제성이 높은 콘텐츠를 추천하며, 추천하는 카테고리 명의 생성 기준과 주제가 명확하지 않고 포괄적이기 때문에 사용자가 느끼는 불편함이 크다.

기존의 추천 시스템 연구는 크게 협업 필터링과 콘텐츠 기반 필터링으로 나눌 수 있다. 협업 필터링은 사용자 기반 추천과 아이템 기반 추천으로 나눌 수 있으며, 사용자와 아이템 사이의 관계를 고려하여 추천하는 시스템이다. 콘텐츠 기반 필터링은 아이템 자체를 분석하여 추천하는 방법이다.

기존의 콘텐츠 추천 시스템 연구는 대부분 사용자와 콘텐츠의 관계를 모델링하여 개인화된 경험을 제공하는 협업 필터링과 관련된 연구가 대부분이다. 협업 필터링은 데이터 희소성 문제와 활동 이력이 부족한 사용자의 경우 메타데이터 부족에 따른 cold-start 문제가 발생한다. 또한, 사용자 수에 따른 연산량이 기하급수적으로 증가하므로 콘텐츠를 추천하기 위한 방법으로 활용하기 어렵다.

따라서 콘텐츠의 정보를 분석하여 추천하는 콘텐츠 기반 필터링 방법이 가장 효과적이지만 콘텐츠의 내용을 온전히 분석하는 연구는 미비하며, 대부분 사용자와 아이템의 메타데이터를 이용한다.

따라서 본 논문에서는 영상 콘텐츠의 내용을 직접적으로 분석하기 위해 텍스트와 이미지를 함께 사용하는 방법인 텍스트-이미지 임베딩 방법을 활용한다. 텍스트-이미지 임베딩은 텍스트와 이미지 정보를 함께 분석하기 위한 방법으로 다른 도메인 데이터의 특징추출을 통한 벡터화 및 결합으로 클래스 간의 관계를 학습하기 위한 모델이다.

하지만 대부분의 텍스트-이미지 임베딩 모델은 이미지-텍스트 검색, 이미지 캡셔닝, VQA와 같은 문제 해결에만 사용되었기 때문에 콘텐츠 분석과 관련된 연구는 미비한 실정이다.

따라서 본 논문에서는 콘텐츠 분석에 적합한 텍스트-이미지 임베딩 모델과 제안한 모델을 학습하기 위한 개선된 Triplet Ranking Loss를 제안한다.

제안하는 텍스트-이미지 임베딩 모델은 텍스트와 이미지 네트워크, 텍스트-이미지 네트워크로 구성된다. 텍스트와 이미지 네트워크는 텍스트와 이미지를 임베딩하고 텍스트-이미지 네트워크는 텍스트와 이미지 임베딩 결과를 결합하고 콘텐츠의 관계를 학습한다.

텍스트 네트워크에서는 사전학습된 Word2Vec, KoBERT, 기본적인 RNN, LSTM 중 하나를 이용하며, 이미지 네트워크에서는 사전학습된 ResNet, Inception V3, EfficientNet 중 하나를 이용한다. 텍스트와 이미지 임베딩 방법은 각각의 방법을 모두 사용하여 실험을 진행하고 그중 가장 성능이 뛰어난 방법을 선정한다. 그 결과 텍스트와 이미지 임베딩에서 KoBERT와 EfficientNet을 사용하는 것이 가장 뛰어난 성능을 보였으며, 텍스트와 이미지 네트워크에서 이 방법으로 임베딩 한다.

텍스트-이미지 네트워크는 임베딩 된 텍스트와 이미지를 결합하고 콘텐츠의 유사함을 학습하기 위한 네트워크로 1차원 Convolution과 1차원 MaxPooling, Dropout으로 구성된 5개의 모듈과 Flatten layer, Fully Connected layer, L2-Normalization으로 구성된다. 각 모듈은 kernel의 크기와 개수를 달리 설정했다. 제안한 텍스트-이미지 임베딩 네트워크에서 콘텐츠 간 유사도를 학습하기 위해서 개선된 Triplet Ranking Loss를 제안한다.

기존 Triplet Ranking Loss는 거리 값이 0 이상인 경우만 업데이트 하기 때문에 음수는 학습에 영향을 미치지 않는다. Triplet Ranking Loss 값으로 0이 빈번하게 발생하게 된다면 업데이트가 잘 안되고 학습이 정상적으로 되지 않을 수 있다. 그뿐만 아니라, 음수도 분명히 어떠한 의미가 있기 때문에 학습에 포함하는 것이 적절하다.

따라서 본 논문에서 텍스트-이미지 임베딩 모델의 학습을 위해 개선된 Triplet Ranking Loss를 제안한다. 개선된 Triplet Ranking Loss는 기존의 방법에 비선형 함수인 sigmoid를 결합하였으며, 그를 통해 Loss의 값이 음수인 경우에도 적절한 값으로 업데이트될 수 있게 수식을 변경했다.

개선된 Triplet Ranking Loss와 같은 학습 방법을 가진 Metric Learning 기반의 Loss 함수들과 비교 평가를 수행한다. 비교 평가에 사용한 함수는 Contrastive Loss, 기존의 Triplet Ranking Loss, VSE++이다. 머신러닝에서 Metric Learning과 같은 목적을 갖는 K-NN도 비교 항목에 포함시켰다.

개선된 Triplet Ranking Loss로 제안하는 텍스트-이미지 임베딩 모델을 학습한 결과 기존의 방법들보다 우수한 성능을 보였다. 이로 인해 영상 콘텐츠 유사도 측정을 위한 텍스트-이미지 임베딩 모델의 구조와 이를 학습하기 위한 개선된 Triplet Ranking Loss가 적합함을 보였다.

최종적으로 차원 축소 및 시각화 결과를 살펴보면 장르를 기반으로 군집화가 된 것을 확인할 수 있으며, 이와 같은 결과를 토대로 사용자가 시청한 콘텐츠와 유사한 콘텐츠 장르를 근거로 추천할 수 있을 것으로 보인다.

이에 따라 추천하는 카테고리 명과 주제에 대한 근거를 제시하고 사용자의 서비스 이용 만족도를 개선할 수 있을 것으로 보이며, 그동안 제대로 연구되지 않았던 콘텐츠 기반 추천 시스템이 가능성을 보였다.

## 참고문헌

- [1] 신지형, 김윤화. “KISDI STAT REPORT 2020년 한국미디어패널 조사결과 주요 내용”, 『정보통신정책연구원ICT데이터사이언스연구본부』, 21-01호, 2021.
- [2] 정용찬, 김윤화. “2020 방송매체 이용행태 조사“, 『방송통신위원회』, 2020.
- [3] 정용국, 장위, “구독형OTT 서비스특성이이용자만족과지속사용의도에 미치는 영향: 넷플릭스이용자를대상으로“, 한국콘텐츠학회논문지, 제20권, 제12호, pp. 123-135, 2020.
- [4] 빠오탄탄, 김현, ”OTT 서비스 콘텐츠 추천 시스템 수용 저항에 영향을 미치는 요인 : 넷플릭스 이용자를 중심으로“, 방송통신연구, 여름호, pp. 9-46, 2021.
- [5] 김지현, 하희정, 김서희, 정영욱, ”OTT 서비스 콘텐츠 추천 사용자 경험 분석 - 넷플릭스 사례를 중심으로“, Journal of Integrated Design Research, 제20권, 제2호, pp. 73-87, 2021.
- [6] Yan Gong, Georgina Cosma, Hui Fang, “On the Limitations of Visual-Semantic Embedding Networks for Image-to-Text Information Retrieval”, Imaging. vol.7 issue.8, July 2021.
- [7] Tomax Mikolov, Greg Corrado, Kai Chen, Jeffrey Dean, “Efficient Estimation of Word Representations in Vector Space”, ICRL 2013, January 2013.
- [8] Pengfei Liu, Xipeng Qiu, Xuanjing Huang, “Recurrent Neural Network for Text Classification with Multi-Task Learning”, IJCAI 2016, July 2016.

- [9] Hasim Sak, Andrew Senior, Françoise Beaufays, “Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling”, INTERSPEECH 2014, September 2014.
- [10] 이상아, 장한솔, 백연미, 박수지, 신호필. “소규모 데이터 기반 한국어 버트 모델”, 『정보과학회논문지』, 제47권, 제7호, pp.682-692, 2020.
- [11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, “Deep Residual Learning for Image Recognition”, CVPR 2016, June 2016.
- [12] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, “Going deeper with convolutions”, CVPR 2015, June 2015.
- [13] Mingxing Tan, Quoc V. Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks”, PMLR 2019, June 2019.
- [14] Weiyang Liu, Yandong Wen, Zhiding Yu, Meng Yang, “Large-Margin Softmax Loss for Convolutional Neural Networks”, International Conference on Machine Learning, pp. 507 - 516, 2016.
- [15] Fartash Faghri, David J. Fleet, Jamie Ryan Kiros, Sanja Fidler, “VSE++: Improving Visual-Semantic Embeddings with Hard Negatives”, BMVC 2018, September 2018.
- [16] 송경우, 문일철, “추천시스템 최근 연구 동향 및 향후 연구 방향 소개”, 정보과학회지, 제39권, 제3호, pp. 16-23, 2021.
- [17] 백서인, 민대기, “콘텐츠 선호 모형을 결합한 행렬 분해 기반 영화 추천시스템”, 대한산업공학회지, 제47권, 제3호, pp. 280-288, 2021.

- [18] Reddy, S. R. S., Nalluri, S., Kuniseti, S., Ashok, S., and Venkatesh, B., “Content-based Movie Recommendation System Using Genre Correlation”, Smart Intelligent Computing and Applications, pp. 391-39, 2019.
- [19] Yashar Deldjoo, Markus Sched, Mehdi Elahi, “Movie Genome Recommender: A Novel Recommender System Based on Multimedia Content”, CBMI 2019, September 2019.
- [20] 권명하, 공성인, 최용석, “임베딩을 활용한 순환 신경망 기반 추천 모델의 성능 향상 기법”, 정보과학회논문지, 제45권, 제7호, pp.659-666, 2018.
- [21] Yan Gong, Georgina Cosma, Hui Fang, “On the Limitations of Visual-Semantic Embedding Networks for Image-to-Text Information Retrieval”, Imaging. vol.7 issue.8, July 2021.
- [22] Fartash Faghri, David J. Fleet, Jamie Ryan Kiros, Sanja Fidler, “VSE++: Improving Visual-Semantic Embeddings with Hard Negatives”, BMVC 2018, no.22, September 2018.
- [23] Florian Schroff, Dmitry Kalenichenko, James Philbin, “FaceNet: A unified embedding for face recognition and clustering”, CVPR 2015, pp. 815-823, June 2015.
- [24] Karen Simonyan, Andrew Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition”, ICLR 2015, May 2015.
- [25] Raia Hadsell, Sumit Chopra, Yann LeCunn, “Dimensionality Reduction by Learning an Invariant Mapping”, CVPR 2006, June 2006.
- [26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, “Attention Is All You Need”, NIPS 2017, December 2017.

- [27] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, “Going Deeper with Convolutions”, CVPR 2014, June 2015.
- [28] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, “Rethinking the Inception Architecture for Computer Vision”, CVPR 2016, pp. 2818–2826, June 2016.
- [29] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, Alex Alemi, “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning”, AAAI 2017, pp. 4278–4284, February 2017.
- [30] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, Quoc V. Le, “MnasNet: Platform-Aware Neural Architecture Search for Mobile”, CVPR 2019, pp. 2815–2823, June 2019.
- [31] Andrew G. Howard, Menglong Zhu, Hartwig Adam, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications”, ArXiv abs/1704.04861, 2017.
- [32] Pretrained Word2Vec model, “<https://github.com/Kyubyong/wordvectors>”  
(Last Access 2021.11.22.)
- [33] Pretrained KoBERT model, “<https://github.com/SKTBrain/KoBERT>”  
(Last Access 2021.11.22.)
- [34] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, et al., “Google’s Neural Machine Translation System: Bridging the Gap between Human and Machine Translation“, arXiv:1609.08144, pp. 1–23, September 2016.



- [35] Pretrained ResNet, InceptionV3, EfficientNet model, “<https://keras.io/api/applications/>” (Last Access 2021.11.22.)
  
- [36] Pdraig Cunningham, Sarah Jane Delany, “k-Nearest Neighbour Classifiers - A Tutorial”, ACM Computing Surveys, vol.54, issue.6, article No.128, pp.1-25, July 2022.