석 사 학 위 논 문

DeepFM-Transformer: DeepFM과 NLP Transformer 결합모델 기반의 추천 시스템

김 형 우

제주대학교 대학원

컴퓨터 공학과

2024년 2월

DeepFM-Transformer: DeepFM과 NLP Transformer 결합모델 기반의 추천 시스템

이 논문을 컴퓨터공학 석사 학위논문으로 제출함

김 형 우

제주대학교 대학원

컴퓨터 공학과

지 도 교 수 이 상 준

김형우의 컴퓨터공학 석사 학위논문을 인준함

2023년 11 월

심사위원장	이 상 준	이상준
위 원	곽 호 영	곽호영
위 원	변 영 철	변영철

DeepFM-Transformer: DeepFM과 NLP Transformer 결합모델 기반의 추천 시스템

김 형 우

제주대학교 대학원 컴퓨터 공학과

요 약

제품 설명데이터는 제품을 구매하는 데 매우 중요한 요인이기에, 추천 시스템에서 성능을 높일 수 있는 데이터이다. 기존에 좋은 성능을 보여준 Factorization Matrix 기반 모델은 데이터 특성 간 상호작용 가능하여 추천 시스템에서 좋은 성능을 보여주지만, NLP 모델이 아니기에 제품 설명 텍스트 데이터를 다른 특성과 조합하여 추천하기가 어렵다.

본 논문에서는 제품 설명 텍스트 데이터를 자연어 처리하고, 다른 특성과 함께 조합하여 학습하는 추천 모델을 제안한다. 제안된 모델인 DeepFM-Transformer는 특성 간 상호작용을 학습하는 DeepFM 모델과 제품 특성 텍스트 데이터를 NLP 하는 Transformer 모델을 결합한 추천 모델이다. 연구 결과 Factorization Matrix 기반 모델인 DeepFM에 비하여 평점 예측 RMSE에서 0.00715, xDeepFM에 비하여 0.01503의 성능 향상을 보여, 제품의 설명이 추천 성능 향상에 영향을 미친다는 것을 밝혀 내었다.

키워드: 추천시스템, 제품 설명, DeepFM, Transformer, Self-Attention, NLP, 딥러닝, 신경망

DeepFM-Transformer: A Recommendation system based on DeepFM and NLP Transformer combined model

Hyeong-Woo Kim

(Supervised by professor Sang-Joon Lee)

Department of Computer Engineering

The Graduate School

Jeju National University

Abstract

Product description data is a very important factor in purchasing a product, so it is data that can improve performance in a recommendation system. Factorization Matrix based models show good performance in recommendation systems by enabling interaction between data features, but since they are not NLP models, it is difficult to recommend product description text data in combination with other features.

In this paper, we propose recommendation modeling that learns by processing product description text data in natural language and combining it with other features. The proposed model, DeepFM-Transformer, is a recommendation model that combines the DeepFM model, which learns interactions between features, and the Transformer model, which processes natural language text data of product features. As a result of the study, it showed a performance improvement of 0.00715 in rating prediction RMSE compared to the Factorization Matrix-based model DeepFM and 0.01503 compared to xDeepFM,

revealing that the product description has an effect on improving recommendation performance

Keywords- DeepFM-Transformer, Recommendation system, product description, DeepFM, Transformer, Self-Attention, NLP, deep learning, neural network.

목 차

표 목차

그림 목차

Ⅰ. 서 론

1. 연구 배경

온라인으로 누구나 자신의 제품을 소개할 수 있고, 검색엔진으로 쉽게 많은 제품을 접할 수 있게 됨에 따라, 자신에게 맞는 제품을 구매하기 어려워지고 있어, 추천 시스템의 중요성이 커지고 있다.

온라인에서 제품을 구매할 때 모르는 제품에 대하여, 글자로 된 설명을 읽지 않고 구매하는 사람은 많지 않을 것이다. 제품 설명은 고객이 제품을 선택할 수 있는 범위를 좁혀주고 가장 좋은 제품을 찾는 데 도움을 주므로 구매 결정을 내리는 데 중요한 역할을 한다. Salsify의 조사에 따르면 고객의 87%가, Field Agent Digital Shopper Report의 조사에 따르면 고객의 82%가 제품 설명을 매우 중요하게 여기며 제품을 구매할 때 가장 중요한 요인이라 생각한다고 했다 [1][2].

하지만 제품 설명에는 개인에 대한 정보가 없기 때문에, 제품 설명만으로는 올바른 개인화 추천을 하기가 어렵다.

또한 텍스트 데이터이기 때문에, NLP를 거쳐야 보다 나은 추천이 가능하다. 이런 이유로 제품 설명데이터는 추천을 위한 많은 정보가 있음에도 불구하고, 딥러닝 자연어 처리 기술이 이전보다 매우 발전한 지금의 시대에도 사용자의 제품 리뷰 데이터를 사용한 추천 기술은 많으나, 제품 설명만으로 추천하는 기술은 드문 현실이다.

또한 개인화 추천에서는 제품의 특성(카테고리, 가격 등)과, 사용자의 특성(나이, 성별 등)이 매우 중요한 데이터이다. 이뿐 아니라 Factorization Machines에서는 제품과 사용자 특성 간의 상호작용을 사용하여, 기존의 추천 방식인 협업필터링, 행렬 분해 방식 추천보다 추천의 정확도를 높였다 [3].

앱 시장에서 연구한 결과, 사람들이 식사 시간에 음식 배달을 위한 앱을 다운

로드하는 경우가 많다는 것을 발견했으며, 이는 앱 카테고리와 시간 이 두가지 특성 간의 (order-2) 상호작용이 추천에 중요한 신호로 사용될 수 있고, 또 남자 청소년들은 슈팅게임과 RPG게임을 좋아하는 것으로 나타났는데, 이는 앱 카테고리, 사용자 성별, 연령 이 세 가지 특성의 (order-3) 상호작용이 추천에 중요한 역할을 한다는 의미이다 [4].

맥주를 사는 30대 남성은 기저귀를 구매한다 (order-4)과 같은 특성 간의 상호작용은 월마트 데이터 분석가를 통해서 밝혀 졌지만 [5], 이 보다 더 고차원의 특성 간 상호작용은 전문가도 밝혀내기 힘들다.

wide & deep [6] 에서는 특성 간 저차원의 상호작용으로 해당 사용자의 구매한 아이템과 유사한 아이템을, 고차원의 상호작용으로는 보다 일반적인 아이템을 추천하는 방식을 조합하여 기존의 방식보다 더 좋은 성능을 내기도 하였다. wide & deep은 저차원과 고차원의 특성 간 상호작용을 조합한 모델이지만, 특성의 고차원 상호작용을 위해서는 전문가의 데이터 특성 공학이 필요한 모델이다.

DeepFM 모델은 데이터 특성 공학이 없이 딥러닝 학습이 가능하고, 저차원과 고차원의 특성을 조합하여 wide & deep 보다 나은 성능을 보인 SOTA 모델이다. 하지만 NLP 모델이 아니기 때문에 제품 설명을 다른 특성과의 상호작용에 모델링을 할 수가 없다 [4].

본 논문에서는 DeepFM 과 NLP 모델인 Transformer의 Encoder 그리고 이 둘을 조합하는 신경망을 사용하여, 제품의 설명을 NLP한 벡터값과 사용자와 제품 특성의 저차원 및 고차원 상호작용을 조합하여 추천할 수 있는 모델을 제안한다.

2. 연구 구성

본 논문의 구성은 다음과 같다. 2장 관련 연구에서 본 연구에 필요한 self-attention, Fasttext, Wide&Deep, DeepFM 등을 살펴본다. 3장에서는 본 논문에서 제안하는 DeepFM-Transformer의 아키텍처 및 모델에 대한 설명, 학습 방법 등에 대하여 설명한다. 4장에서는 실험을 위해 데이터 셋의 분포도 비교와

데이터 전처리 및 비교 모델별 실험하는 방법에 관해서 설명하고 결과를 도출한다. 5장에서는 결론 및 향후 연구에 대하여 논의한다.

Ⅱ. 관련연구

1. Self-Attention

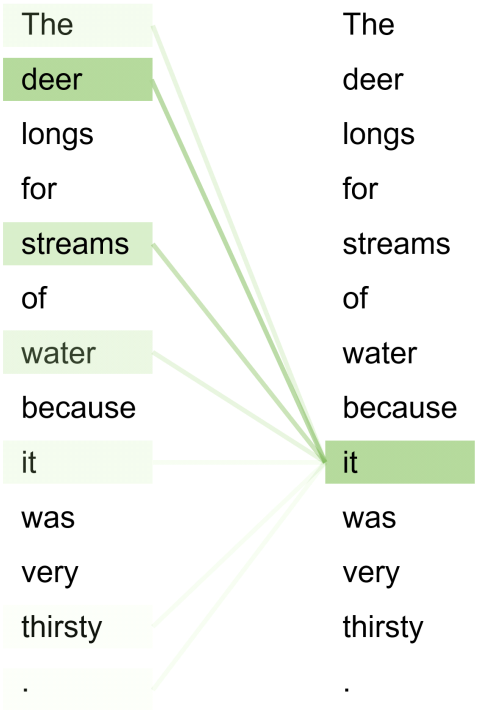
기존 NLP모델의 경우 LSTM을 기반으로 순차적으로 처리하기 때문에 속도가 늦고 텍스트를 읽을 때 장거리 맥락을 추론하는 데 어려움이 있다는 단점이 있다[7].

Self-Attention은 이런 단점으로 보완하여 순차적 처리가 아닌 입력값들을 병렬로 처리하여 속도를 향상시키고, 단어 간의 연관성을 계산하여 문장의 맥락을 이해하도록 하는 딥러닝 모델이다.

해당 모델은 아래의 셀프 어텐션 계산식을 통해 단어들을 임베딩 값으로 변환하여 단어 간의 조합과 중요도, 유사도를 학습할 수 있다[8].

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}}) \quad (1)$$

그림 1의 예시 문장을 번역하면 '사슴은 시냇물을 찾아 헤매었다. 왜냐하면 그것은 너무 목이 말랐기 때문이다.'라는 의미가 된다. 그런데 여기서 그것(it)에 해당하는 것이 시냇물(streams)일지 사슴(deer)일지 판단하는 것은, 사람인 경우 목마른 주체가 사슴인 것을 쉽게 알 수 있지만 기계는 그렇지 않다. 하지만 셀프 어텐션은 입력 문장 내의 단어들끼리 유사도를 구함으로써 그것(it)이 사슴(deer)과 연관되었을 확률이 높다는 것을 찾아낸다.



[그림 1] 셀프 어텐션의 단어 유사도

2. Fasttext

Fasttext는 단어를 벡터로 표현하는 방식으로 미콜로프 [9]가 제안하였다. 기존의 단어 임베딩 방식인 Word2vec와 큰 차이점은 Fasttext는 '서브 워드(Subword)' 개념을 사용한다는 것이다. Fasttext는 단어를 서브 워드라는 '분할된 단어'의 합으로 표현하고 분할된 단어별로 벡터 표현을 학습시킨다. 영어 단어를 서브 워드로 분할 하는 경우 기본적으로 3~6문자로 분할 된다. 단어 where을 예로 들자면 단어의 시작과 끝을 나타내는 기호 '<' 또는 '>'를 붙여 <where>로 하고, 이것을 3~6문자로 구분하면 다음과 같다.

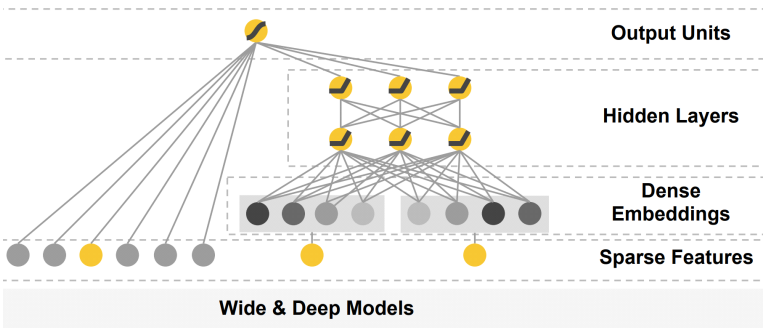
- 3문자: <wh, whe, her, ere, re>
- 4문자: <whe, wher, here, ere>
- 5문자: <wher, where, here>
- 6문자: <where, where>

5 + 4 + 3 + 2 = 14가지 서브 워드가 되었다. 14가지 서브 워드의 벡터 표현 합으로 단어 where가 표현된다 [10].

서브 워드를 사용하는 이유는, 기존 word2vec가 알 수 없는 단어(unknown word)에 취약했기 때문이다. 단어를 벡터로 표현할 때 알 수 없는 단어를 벡터로 표현하는 문제가 있다. 단어의 토큰화 수행 시 <unknown>이라는 토큰으로 표기할 수는 있지만, 그렇게 되면 알 수 없는 서로 다른 단어들이 전부 <unknown>의 벡터로 표현되어, 자연어 처리 정확도가 떨어진다. 서브 워드는 이러한 알 수 없는 단어의 벡터 표현 문제를 해결하는 방법이다. 벡터 표현을 학습하지 않은 알 수 없는 단어를 서브 워드로 나누어 다른 단어로 학습하면 서브 워드의 합으로 알 수 없는 단어의 벡터 표현도 얻을 수 있다 [10].

3. Wide & Deep

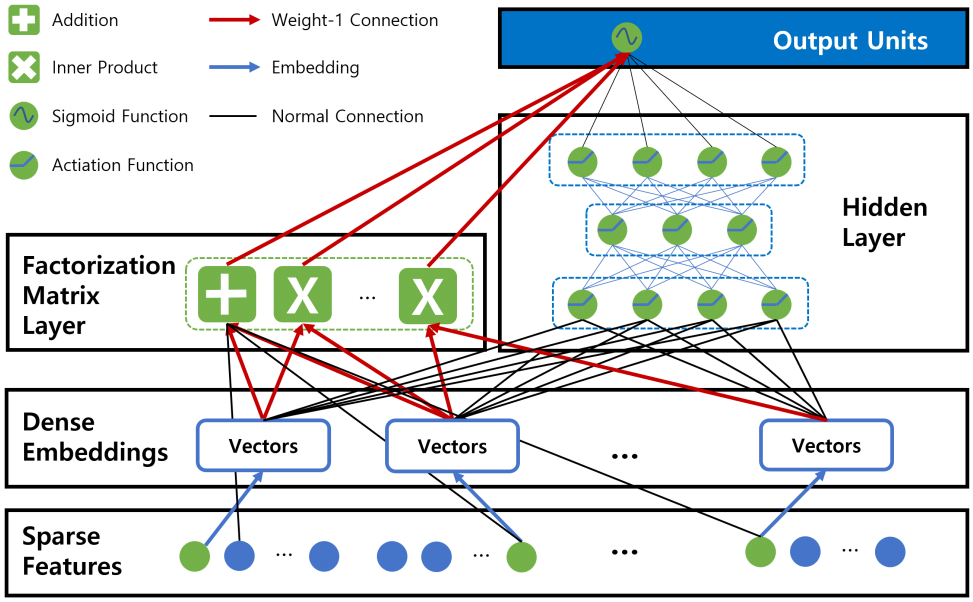
구글에서 발표한 모델로 구성은 Wide 부분과 Deep 부분으로 되어 있다. 그림 2의 왼쪽 Wide 부분에서는 '암기'라고도 표현하며 사용자의 행위 등을 학습하는 부분이며, 오른쪽 Deep부분은 '일반화'라고도 하며, 딥러닝 구조를 사용하여 사용자의 행위를 일반화하는 것을 학습하는 부분이다. 기억과 일반화를 조합하여 추천 정확도를 향상 시킨 모델로 해당 모델을 구글 플레이 추천에 사용하였으며, 앱 설치 수가 증가했다고 하였다. 하지만 이 모델은 사용자의 행위를 전문가의 특성공학을 거쳐야 학습 데이터로 사용할 수 있는 단점이 있다 [6].



[그림 2] Wide & Deep 아키텍처

4. DeepFM

DeepFM은 사용자 및 제품 등의 특성들에 대하여 저차원 및 고차원 상호작용을 모두 학습하는 것을 목표로 하는 Factorization-Machine 기반 신경망이다. DeepFM은 동일한 입력을 공유하며, 그림 3의 왼쪽 Factorization Matrix Layer에 해당하는 FM 컴포넌트와 오른쪽 Hidden Layer에 해당하는 Deep 컴포넌트의 두 가지 요소로 구성된다. Wide & Deep과 다르게 데이터가 특성 공학을 하지 않고 바로 모델의 입력으로 사용할 수 있다 [4].

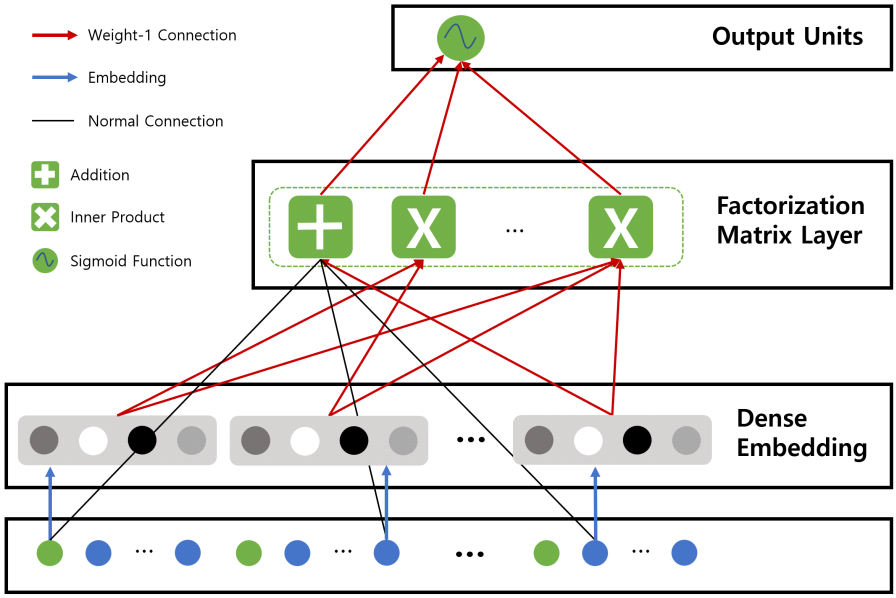


[그림 3] DeepFM 아키텍처

1) FM Component

원-핫 인코딩 방식은 대부분이 0이고 해당하는 항목에만 1로 구성되는 매우 희박한 행렬이다. 그러므로 해당 데이터 사용 시 성능이 저하가 나타나거나 행렬 분해 방식을 사용해야 한다. 하지만 FM 컴포넌트는 데이터 세트가 희박한 경우에도 이전 방식보다 훨씬 더 효과적으로 특성 간의 상호작용을 할 수 있다. FM은

특성 i, j의 잠재 벡터 V_i와 V_j의 내적을 통해서 계산하며, 훈련 데이터에 전혀 또는 거의 나타나지 않는 특성 간 상호작용을 잘 학습할 수 있는 모델이다. 그림 4는 FM 컴포넌트의 아키텍처 이다 [4].



[그림 4] FM 아키텍처

2) Deep Component

Deep Component는 신경망으로 구성된 특성 간 고차원 상호작용을 학습하는 부분이다. Wide & Deep와 다르게 전문가의 특성공학이 필요하지 않기 때문에, Deep Component는 모든 특성의 저차원 임베딩을 신경망의 입력으로 활용할 수 있다 [4].

Ⅲ. DeepFM-Transformer

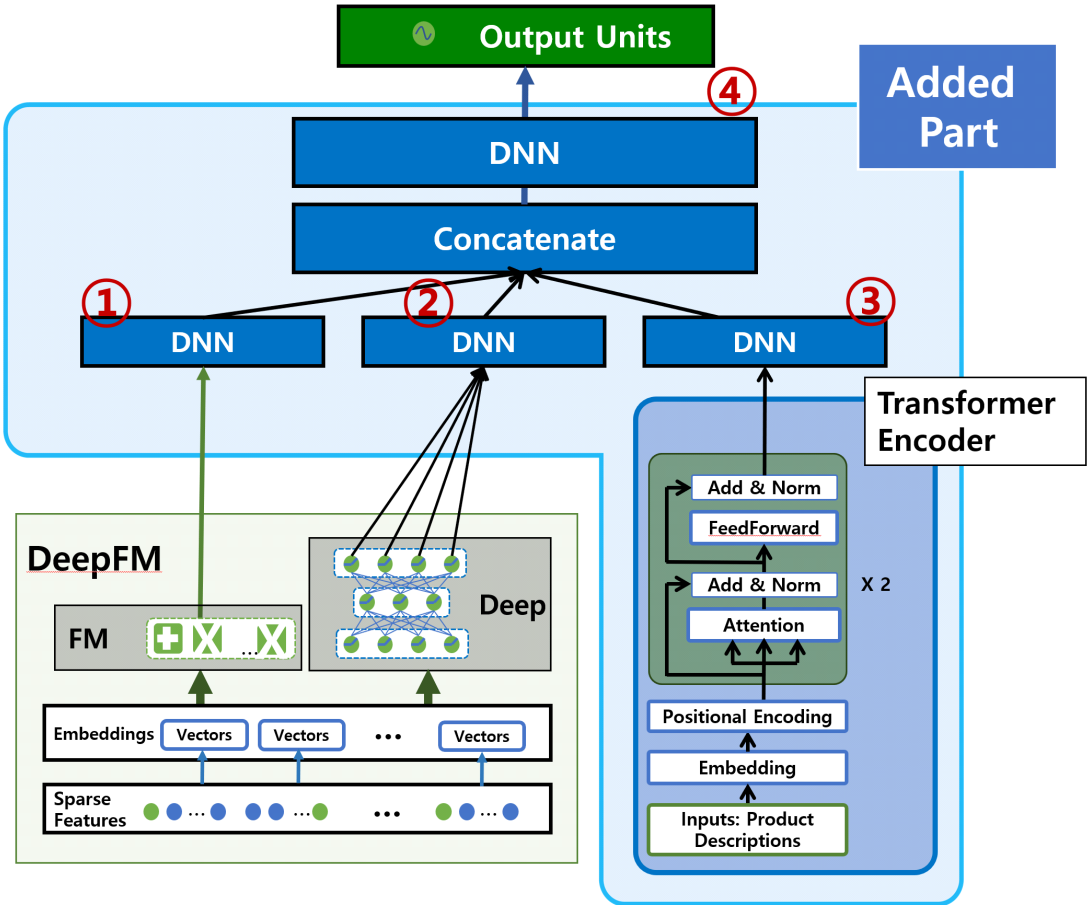
본 논문에서 제안하는 DeepFM-Transformer는 사용자와 제품 특성의 저차원 및 고차원 상호작용과 제품의 설명 텍스트 데이터를 NLP 한 후 생성된 벡터를 조합하여 추천하는 모델이다.

DeepFM-Transformer의 아키텍처는 그림 5에서 볼 수 있듯이, 사용자와 제품 특성의 저차원 및 고차원 상호작용을 처리하는 DeepFM과 제품의 설명 텍스트 데이터를 NLP하는 Transformer Encoder와 두 모델의 출력값을 합치고(concatenate) 훈련하는 DNN과 Output Unit으로 구성되어 있다.

입력값은 사용자와 제품의 각 특성을 레이블링한 수치 데이터 $x = [x_1, x_2, \dots, x_j, d_1, d_2, \dots, d_k]$ 이며, x_j는 일반 특성의 레이블링 된 수치 데이터이고, d_k는 설명 텍스트 문장의 단어를 수치로 표현한 데이터이다. 출력을 표현하는 수식은 수식 2와 같다.

$$\hat{y} = sigmoid(DNN(yFM + yDeep + yTransformer)) \quad (2)$$

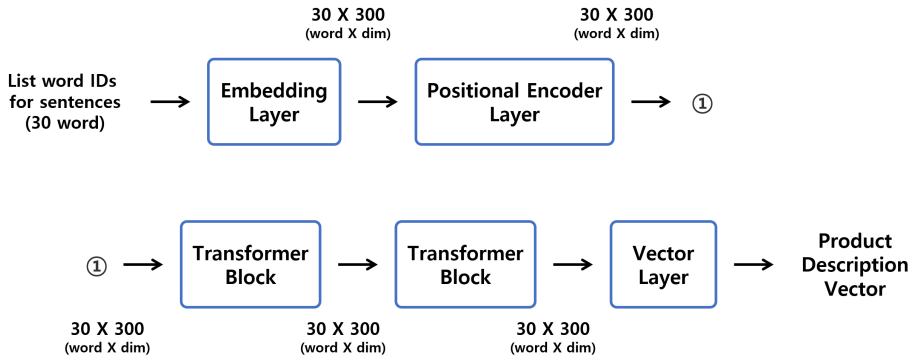
1. Transformer Encoder



[그림 5] DeepFM-Transformer 아키텍처

본 논문에서는 Lian, J. 등이 제안하는 Transformer의 Encoder를 사용한다 [8]. 본 논문에서 Transformer Encoder의 목적은 제품의 설명을 다른 특성들의 저차원 및 고차원 상호작용한 벡터와 조합하여 최고의 결과를 낼 수 있는 최적의 벡터를 만드는 것이다.

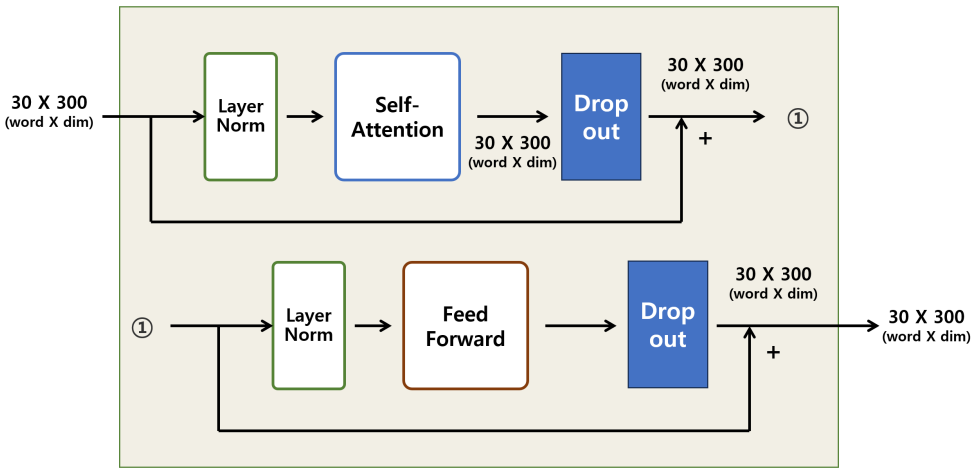
Transformer Encoder의 구성 방법은 그림 6과 같이 문장의 단어 ID리스트를 임베딩하고, 단어의 순서를 인코딩 한 후 그림 7에서 볼 수 있는, Layer Norm, Self-Attention 그리고 Feed-Forward Layer로 구성된 Transformer 블록을 통과한다. 그리고 문장의 벡터를 생성하는 Vector Layer를 거쳐 제품 설명 벡터를 생성한다 [8][10].



[그림 6] Transformer Encoder의 학습 순서

Lian, J.등이 제안한 Transformer Encoder는 Multi-Head Attention이 핵심 역할을 하지만, 본 논문에서는 ogawa yutaro [10]가 제안한 Sing Self-Attention을 기반으로 한 Transformer Block을 사용한다. Transformer Block은 그림 7과 같은 아키텍처로 구성되며, 임베딩 된 입력값을 정규화하고, Self-Attention에서 각 단어의 Attention 점수를 계산하고 Dropout층을 거친 결과에 임베딩 입력값을 더하여 ① 값을 계산한다. 이어서 해당 값의 정규화를 시행하고 2개의 Layer로 구성된 FeedForward 신경망을 거치고, Dropout층을 거쳐서 원래 입력과 같은 형태의 벡터값을 생성한다 [8][10].

본 논문에서는 Transformer Block을 2개를 연속으로 구성하였다 [10].



[그림 7] Transformer Block 모듈의 구성도

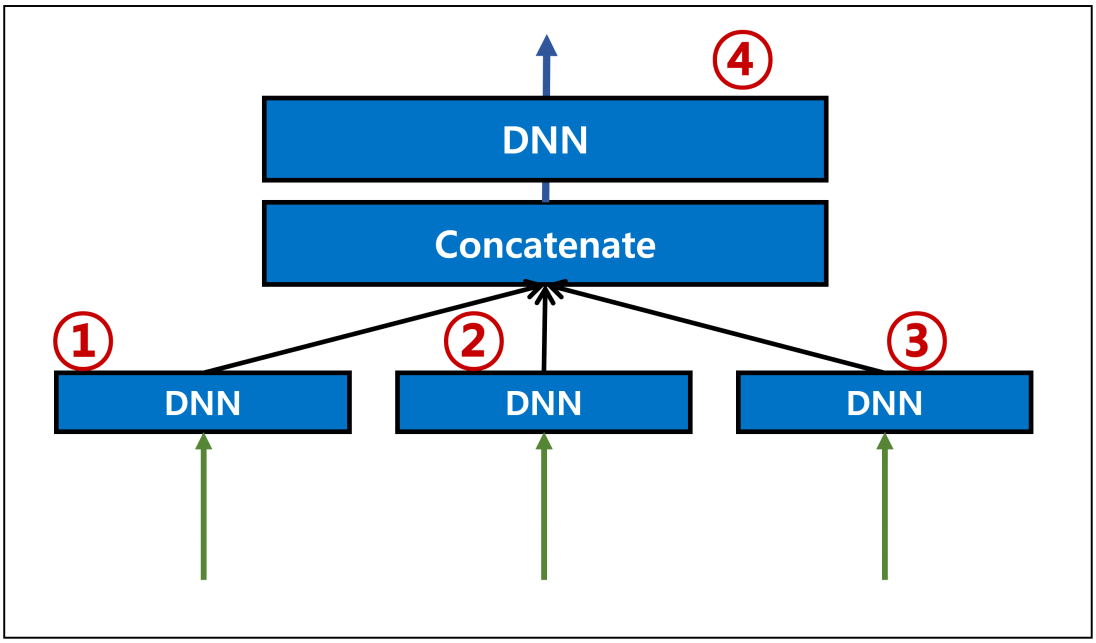
Transformer Block의 출력값은 그림 6에서의 마지막 Layer인 Vector Layer에서 첫 번째 단어의 특징 벡터(1 x 300)를 꺼내서 사용한다. 모든 단어의 특징 벡터를 사용해도 되지만, 레코드별로 문장 길이가 짧은 문장일 경우 문장 후반에 패딩 처리되어 있다. 그래서 의미 있는 전체의 문장 길이가 서로 다르므로, 문장의 첫 번째 단어의 특징 벡터를 사용하였다. 훈련을 통하여 손실을 계산하고, 역 전파하여, 최적의 벡터가 생성되도록 학습하였다 [10].

2. DeepFM

그림 5에서 볼 수 있듯이 DeepFM 부분은 입력층, 임베딩층, FM 부분과 Deep 부분으로 구성하였다. 입력값은 원-핫 인코딩된 희소행렬을 사용할 수도 있고, 레이블링 된 밀집 행렬을 사용할 수도 있다. FM 부분에서는 특성들의 저차원 상호작용을, Deep 부분에서는 특성들의 고차원 상호작용을 학습하였다 [4].

3. Added Part

Added Part는 본 논문에서 제안하는, 특성과 제품 설명데이터를 조합하여 상호작용을 학습하는 모델의 핵심적인 부분이다. 그림 5에서와 같이 Transformer Encoder의 출력과 DeepFM에서 Deep part의 출력과 FM part의 출력을 합쳐서 새로운 벡터를 생성하고, 해당 벡터를 신경망에서 학습하여 최종 결과를 출력하였다. 기존의 DeepFM에서와 차이점은 FM의 출력을 Deep의 출력과 바로 결합하지 않고 3개의 층으로 구성된 신경망을 거치도록 하였다. Added Part의 구성도는 그림 8과 같다.

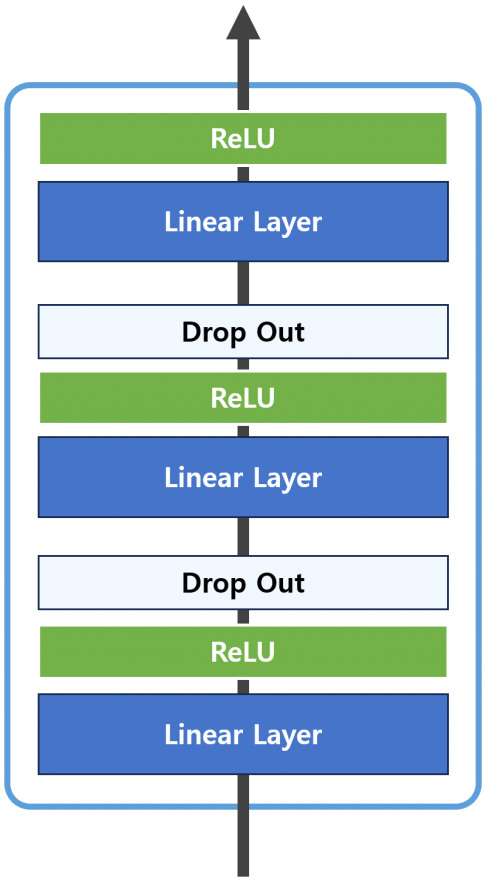


[그림 8] 본 논문에서 제안하는 새로운 딥러닝 모듈

1) FM과 연결된 DNN

FM의 출력을 입력으로 받는 DNN은 그림 5, 8 에서 ①에 해당하는 DNN 이다.

해당 DNN의 구성은 그림 9와 같으며, 3개의 Linear Layer로 되어 있다. 첫 번째와 두 번째 층에서는 ReLU와 DropOut을 사용하였으며, 세 번째 층에서는 ReLU만 사용하였다. 해당 DNN의 출력은 그림 5의 Concateate로 보내진다. FM의 출력 크기는 Batch x 1의 사이즈여서 정규화를 거치지 않았다.

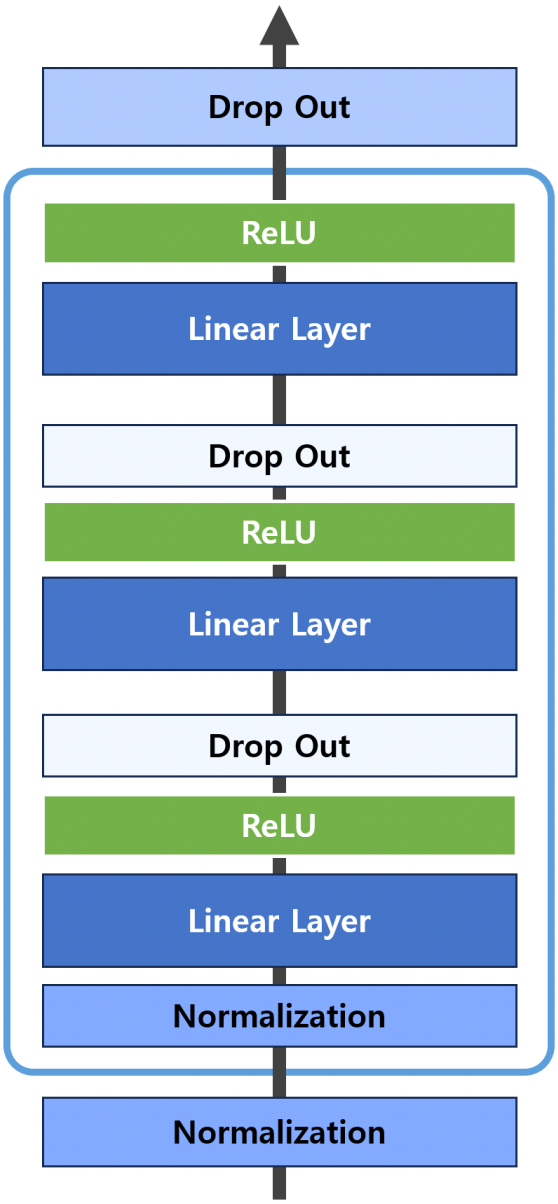


[그림 9] FM과 연결된 DNN 구성도

2) Deep과 연결된 DNN과 Transformer와 연결된 DNN

그림 10은 DeepFM과 Transformer의 Encoder에 연결된 DNN 구성도이다. 해당 DNN은 그림 5의 ②, ③에 해당하는 DNN으로써 이 둘은 같은 구조로 구성하였다. 해당 DNN은 DeepFM과 Transformer의 Encoder의 출력을 입력으로 받고 있다.

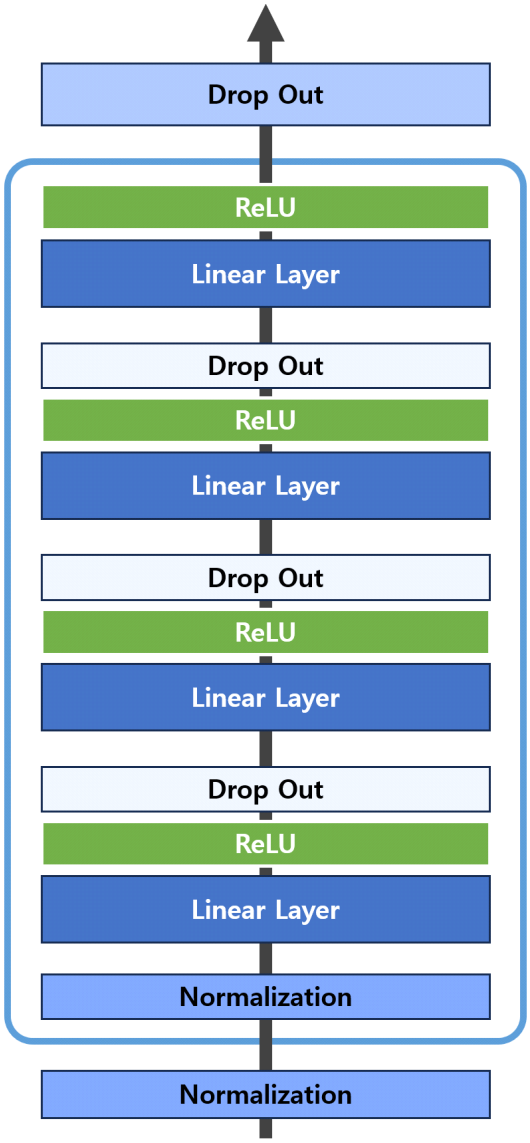
그림 10에서 볼 수 있듯이 DNN 전에 정규화 층이 있고, DNN의 출력에 Dropout 층이 있다. DNN 내부에는 3개의 Linear Layer로 구성되어 있으며, 입력값을 정규화하는 층이 있고, 첫 번째와 두 번째 층에서는 ReLU와 DropOut을 사용하였으며, 세 번째 층에서는 ReLU만 사용하였다. 그리고 해당 DNN 출력은 역시 그림 5의 Concateate로 보내진다.



[그림 10] DeepFM과 Transformer에 연결된 DNN 구성도

3) 최종 DNN

최종 DNN은 FM, DeepFM 그리고 Transformer의 Encoder의 출력을 하나의 벡터로 합친 값을 입력으로 받아 최종적인 Output Unit으로 출력값을 보내는 역할을 한다.



[그림 11] 최종 DNN 구성도

그림 8에서 ④에 해당하는 DNN으로 그림 11에서 볼 수 있듯이 해당 DNN도 DeepFM과 Transformer에 연결된 DNN과 같이 DNN 모듈 전에 정규화 층이 있고, DNN의 출력에 DropOut층이 있어 전체를 DropOut한다. DNN 내부에는 4개의 Linear Layer로 구성되어 있으며, 입력값을 정규화 하는 층이 있고, 첫 번째와 두 번째 층 그리고 세 번째 층에는 ReLU와 DropOut을 사용하였으며, 네 번째 층에서는 ReLU만 사용하였다. 그리고 해당 DNN 출력은 그림 5의 Output Unit으로 보내진다.

Ⅳ. 실 험

1. 실험 설정

1) 데이터 셋

실험을 위해서는, 평점, 사용자 정보, 영화정보, 영화 설명 텍스트 데이터 셋이 필요하였다. 평점, 사용자 정보, 영화정보를 위해서 MovieLens-1M 데이터 셋과, 제품 설명데이터에 해당하는 영화 설명데이터와 영화감독 특성을 사용하기 위해 IMDb Top 1000 데이터 셋, 이 둘을 합쳐서 생성한 MovieLens-IMDB 데이터를 사용하였다.

MovieLens 데이터 셋의 경우 추천 시스템에서 가장 유명한 데이터 셋 중 하나로 2014년 140,000회 이상 다운로드 되었고 구글 스칼라 기준 Movielens를 언급한 논문은 7,500건 이상이 있다. 1998년에 처음 출시된 MovieLens 데이터 셋은 사람들의 영화에 대한 선호도를 표현하는데, 특정 시간에 영화에 대한 평점(별점 0~5개)을 표현하였다 [11][12]. MovieLens에는 영화에 대한 평가 수에 따라 100K, 1M, 10M, 20M, 25M, 5가지 형태의 데이터 셋이 존재하는데, 각각의 데이터 셋은 특성이 조금씩 다르다. 본 논문에서는 사용자의 특성과 영화의 특성을 사용하기 위해 MovieLens-1M의 데이터 셋을 사용하였다.

IMDb Top 1000 데이터는 영화, TV 프로그램, 비디오 게임과 배우, 감독 및 기타 영화 산업 전문가에 대한 정보와 통계를 포함하는 온라인 데이터베이스이며, 팬들이 영화와 프로그램의 세계를 탐색하고 무엇을 볼지 결정할 수 있도록 돕기 위해 고안된 영화, TV 및 유명인 콘텐츠에 대해 세계에서 가장 인기 있고 권위 있는 사이트인 IMDb에서 가장 인기 있는 영화 1,000편에 대한 정보가 있는 데이터 셋이다 [13][14].

1.1) Row 데이터

그림 12, 13, 14는 Movielens-1M의 3가지 파일을 Pandas 라이브러리를 사용하여 표 형식으로 읽어온 것이다.

사용자 데이터의 경우 5가지의 특성(user_id, gender, age, occupation, zipcode)이 존재하고 총 6,040개의 데이터로 이루어져 있다. 영화 데이터의 경우 3가지의 특성(movie_id, title, genre)이 존재하고 총 3,883개의 데이터로 이루어져 있다. 평점 데이터의 경우 4가지의 특성(user_id, movie_id, rating, time)이 존재하고 총 1,000,209개의 데이터로 이루어져 있다.

	user_id	gender	age	occupation	zipcode
0	1	F	1	10	48067
1	2	M	56	16	70072
2	3	M	25	15	55117
3	4	M	45	7	02460
4	5	M	25	20	55455
...	...	...	...	...	...
6035	6036	F	25	15	32603
6036	6037	F	45	1	76006
6037	6038	F	56	1	14706
6038	6039	F	45	0	01060
6039	6040	M	25	6	11106

6040 rows × 5 columns

[그림 12] Movielens-1M의 사용자 데이터

	movie_id	title	genre
0	1	Toy Story (1995)	Animation|Children's|Comedy
1	2	Jumanji (1995)	Adventure|Children's|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama
4	5	Father of the Bride Part II (1995)	Comedy
...	...	...	...
3878	3948	Meet the Parents (2000)	Comedy
3879	3949	Requiem for a Dream (2000)	Drama
3880	3950	Tigerland (2000)	Drama
3881	3951	Two Family House (2000)	Drama
3882	3952	Contender, The (2000)	Drama|Thriller

3883 rows × 3 columns

[그림 13] Movielens-1M의 영화 데이터

	user_id	movie_id	rating	time
0	1	1193	5	978300760
1	1	661	3	978302109
2	1	914	3	978301968
3	1	3408	4	978300275
4	1	2355	5	978824291
...	...	...	...	...
1000204	6040	1091	1	956716541
1000205	6040	1094	5	956704887
1000206	6040	562	5	956704746
1000207	6040	1096	4	956715648
1000208	6040	1097	4	956715569

1000209 rows × 4 columns

[그림 14] Movielens-1M의 평점 데이터

그림 15 는 IMDb Top 1000 데이터 셋을 Pandas 라이브러리를 사용하여 표 형식으로 읽어온 것이다. IMDb Top 1000 데이터 셋의 경우 16가지의 특성(Poster_Link, Series_Title, Released_Year, Certificate, Runtime, Genre, IMDB_Rating, Overview, Meta_score, Director, Star1, Star2 ,Star3, Star4, No_of_votes, Gross)이 존재하고 총 1,000편의 영화 데이터 셋으로 이루어져 있다.

	title	year	Runtime	Genre	IMDB_Rating	Overview	Meta_score	Director	Star1	Star2	No_of_Votes	Gross
0	The Shawshank Redemption	1994	142 min	Drama	9.3	Two imprisoned men bond over a number of years...	80.0	Frank Darabont	Tim Robbins	Morgan Freeman	2343110	28,341,469
1	The Godfather	1972	175 min	Crime, Drama	9.2	An organized crime dynasty's aging patriarch t...	100.0	Francis Ford Coppola	Marlon Brando	Al Pacino	1620367	134,966,411
2	The Dark Knight	2008	152 min	Action, Crime, Drama	9.0	When the menace known as the Joker wreaks havo...	84.0	Christopher Nolan	Christian Bale	Heath Ledger	2303232	534,858,444
3	The Godfather: Part II	1974	202 min	Crime, Drama	9.0	The early life and career of Vito Corleone in ...	90.0	Francis Ford Coppola	Al Pacino	Robert De Niro	1129952	57,300,000
4	12 Angry Men	1957	96 min	Crime, Drama	9.0	A jury holdout attempts to prevent a miscarria...	96.0	Sidney Lumet	Henry Fonda	Lee J. Cobb	689845	4,360,000
...	...	...	...	...	...	...	...	...	...	...	...	...
995	Breakfast at Tiffany's	1961	115 min	Comedy, Drama, Romance	7.6	A young New York socialite becomes interested ...	76.0	Blake Edwards	Audrey Hepburn	George Peppard	166544	NaN
996	Giant	1956	201 min	Drama, Western	7.6	Sprawling epic covering the life of a Texas ca...	84.0	George Stevens	Elizabeth Taylor	Rock Hudson	34075	NaN
997	From Here to Eternity	1953	118 min	Drama, Romance, War	7.6	In Hawaii in 1941, a private is cruelly punish...	85.0	Fred Zinnemann	Burt Lancaster	Montgomery Clift	43374	30,500,000
998	Lifeboat	1944	97 min	Drama, War	7.6	Several survivors of a torpedoed merchant ship...	78.0	Alfred Hitchcock	Tallulah Bankhead	John Hodiak	26471	NaN
999	The 39 Steps	1935	86 min	Crime, Mystery, Thriller	7.6	A man in London tries to help a counter-espion...	93.0	Alfred Hitchcock	Robert Donat	Madeleine Carroll	51853	NaN

1000 rows × 12 columns

[그림 15] IMDb Top 1000 데이터 셋

실험을 위해 MovieLens-1M 데이터 셋에서는 사용자 아이디, 영화 아이디, 직업, 성별, 연령, 장르 특성을 IMDb 데이터 셋에서는 영화 감독 특성, 영화 설명 데이터 사용하였다.

두 데이터 셋을 병합하기 위하여, title과 year 데이터를 키값으로 사용하였고, MovieLens-1M의 경우 title에서 연도를 추출하여 year라는 파생 변수를 생성하였다.

영화 수는 Movielens-1M의 경우 3,883편, IMDb Top 1000에서는 1,000편이다.

MovieLens-1M에 존재하는 영화와 IMDb에 존재하는 영화의 차이 때문에 최종 201,063 건의 데이터가 생성되었다. 생성된 Movielens-IDMb의 경우 239편의 영화가 있다. 그림 16은 생성된 MovieLens-IMDb 데이터 셋을 보여준다.

	user_id	movie_id	rating	gender	age	occupation	zipcode	title	genre	year	Overview	Director
0	1	1193	5	F	1	10	48067	One Flew Over the Cuckoo's Nest	Drama	1975	A criminal pleads insanity and is admitted to ...	Milos Forman
1	1	914	3	F	1	10	48067	My Fair Lady	Musical	1964	Snobbish phonetics Professor Henry Higgins agr...	George Cukor
2	1	1287	5	F	1	10	48067	Ben-Hur	Action	1959	After a Jewish prince is betrayed and sent int...	William Wyler
3	1	595	5	F	1	10	48067	Beauty and the Beast	Animation	1991	A prince cursed to spend his days as a hideous...	Gary Trousdale
4	1	2398	4	F	1	10	48067	Miracle on 34th Street	Drama	1947	When a nice old man who claims to be Santa Cla...	George Seaton
...	...	...	...	...	...	...	...	...	...	...	...	...
201058	6040	541	4	M	25	6	11106	Blade Runner	Film-Noir	1982	A blade runner must pursue and terminate four ...	Ridley Scott
201059	6040	2028	5	M	25	6	11106	Saving Private Ryan	Action	1998	Following the Normandy Landings, a group of U....	Steven Spielberg
201060	6040	1089	4	M	25	6	11106	Reservoir Dogs	Crime	1992	When a simple jewelry heist goes horribly wron...	Quentin Tarantino
201061	6040	1090	3	M	25	6	11106	Platoon	Drama	1986	Chris Taylor, a neophyte recruit in Vietnam, f...	Oliver Stone
201062	6040	1097	4	M	25	6	11106	E.T. the Extra-Terrestrial	Children's	1982	A troubled child summons the courage to help a...	Steven Spielberg

201063 rows × 12 columns

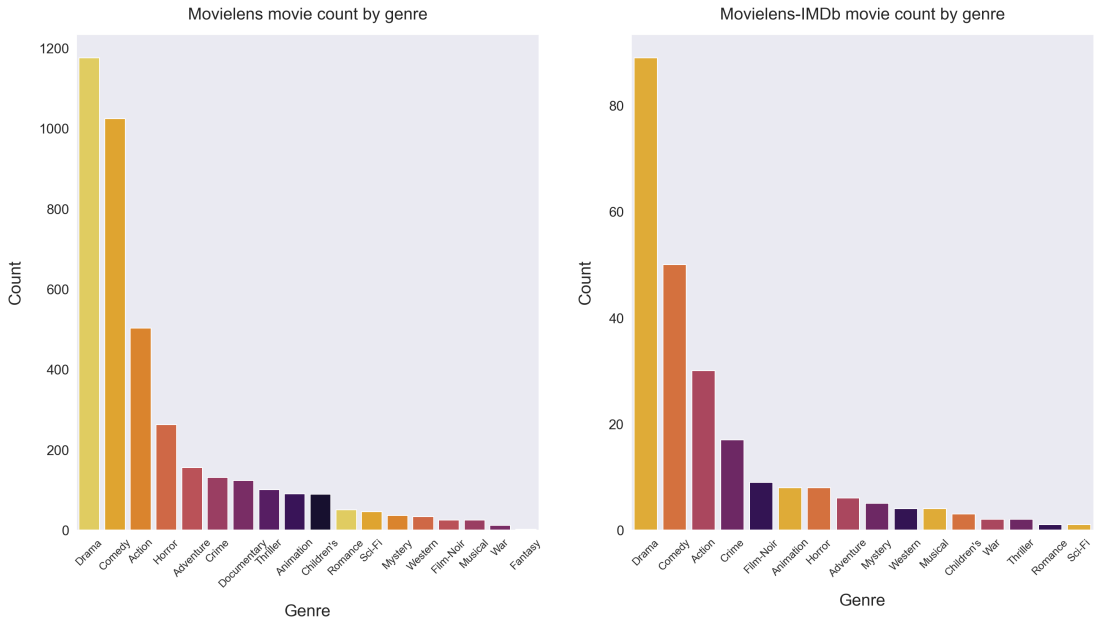
[그림 16] 병합되어 생성된 MovieLens-IMDb 데이터 셋

1.2) 데이터 분포

1.2.1) 영화 데이터의 분포

평점이 적용된 데이터 셋이 아닌, 순수 영화에 대한 MovieLens와 MovieLens_IMDb 두 데이터 셋에 대한 데이터 분포를 먼저 살펴본다.

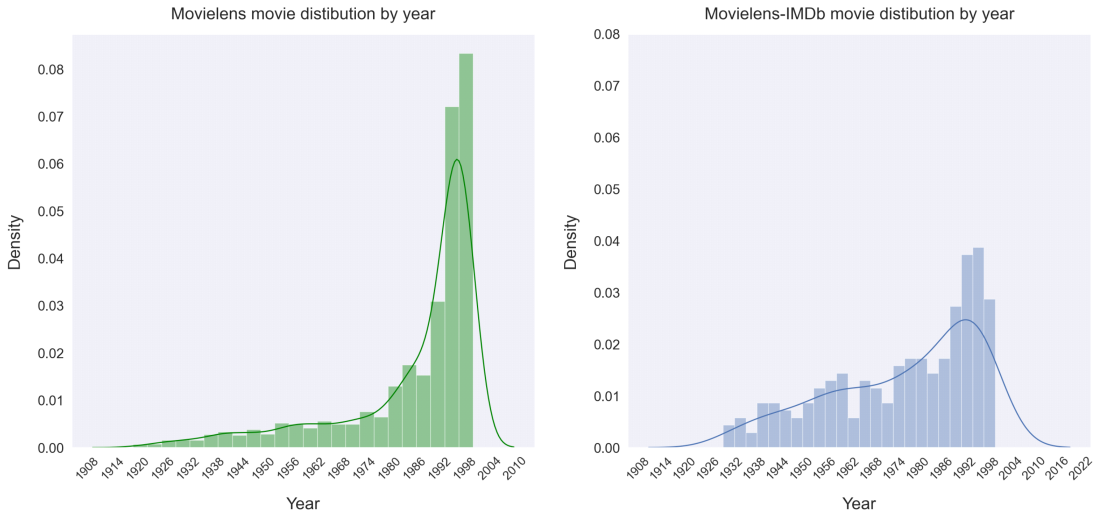
장르의 경우 그림 17에서 볼 수 있듯이 Movielens-IDMb의 영화 장르는 MovieLens에 비해 코미디, 액션, 로맨스가 감소하고 전쟁이 증가한 것으로 나타났다. 그 외의 장르도 조금씩 분포가 다른 것을 볼 수 있다.



[그림 17] Movielens와 Movielens-IMDb의 장르별 영화 편수

연도의 경우 그림 18에서 볼 수 있다. Movielens의 경우 1990년대 중, 후반이 많은 것을 알 수 있고, MovieLens-IDMb역시 1990년대 중, 후반이 영화 편수가 많이 있으나, Movielens에 비하면 절반 정도 분포를 보이며 전체적으로 1980년대 이전의 영화의 분포도 높은 것을 알 수 있다.

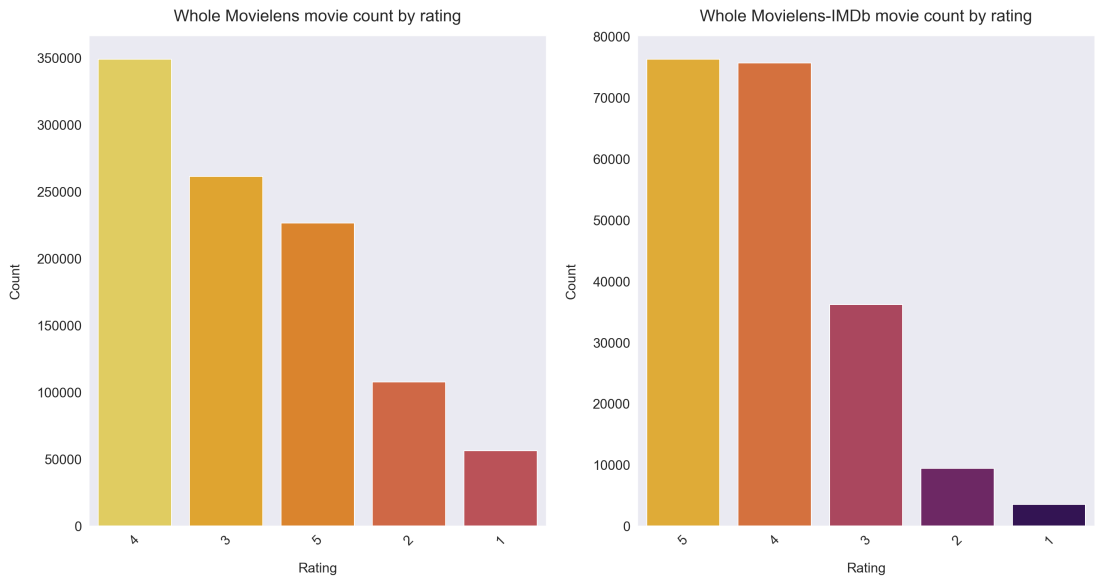
그래프에서는 안 보이지만, Movielens의 영화의 수는 1996, 1995, 1998, 1997, 1999 순으로 분포되었고, Movielens-IDMb의 경우는 1993, 1995, 1996, 1997, 1999 순으로 분포되었다.



[그림 18] Movielens 와 Movielens-IMDb 연도별 영화 편수

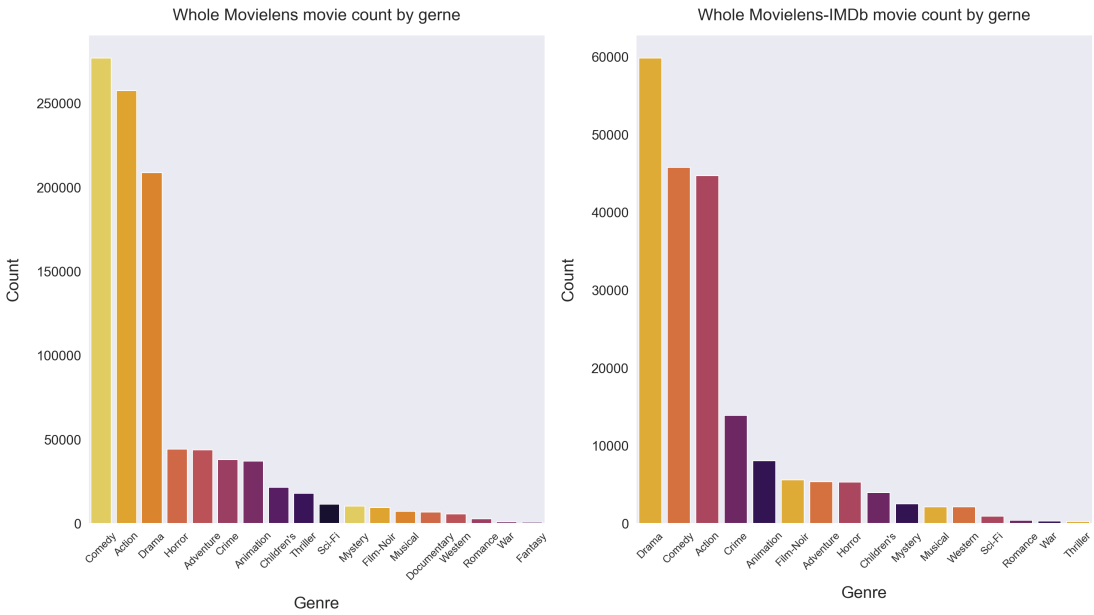
1.2.2) 평점이 포함된 데이터 셋의 데이터 분포

평점이 포함된 전체 데이터에서 Movielens의 평점과 Movielens-IDMb의 평점 분포는 그림 19에서 볼 수 있듯이 Movielens-IMDb의 평점은 MovieLens에 비해 5점은 높게, 3, 2, 1점은 낮게 분포되어 있다.



[그림 19] Movielens와 Movielens-IMDb의 평점별 개수

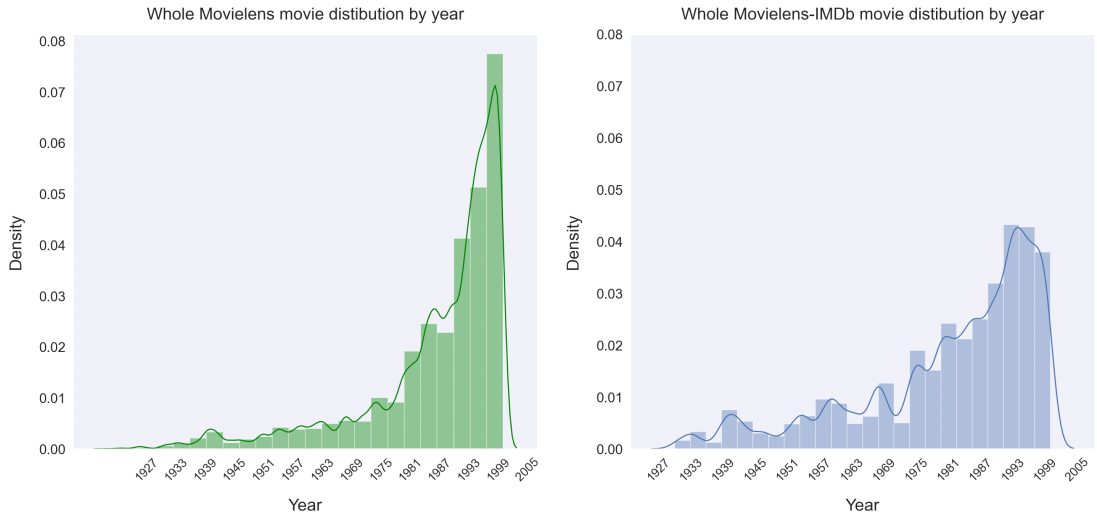
평점이 포함된 전체 데이터에서 장르의 경우 그림 20에서 볼 수 있듯이 Movielens-IDMb 장르는 MovieLens에 비해 코미디, 액션, 로맨스가 감소하고 전쟁이 증가한 것으로 나타났다.



[그림 20] 평점이 포함된 Movielens와 Movielens-IMDb 데이터 셋의 장르별 영화 편수

평점이 포함된 전체 데이터에서 연도의 경우 그림 21에서 볼 수 있듯이, Movielens의 경우 1990년대 중, 후반이 많은 것을 알 수 있고, MovieLens-IDMb 역시 1990년대 중, 후반이 영화 편수가 많이 있으나, Movielens에 비하면 영화 분포가 낮은 것을 알 수 있다. 하지만 이전에 살펴본 평점이 없는 MovieLens-IDMb 영화 데이터에 비해서는 90년대 후반을 제외하면 비슷한 분포를 보인다. 전체적으로 Movielens-IMDb의 1980년대 이전 영화 분포가 Movielens에 비해 높기는 하지만 역시 평점이 없는 MovieLens-IDMb 영화 데이터에 비해서는 낮은 것을 알 수 있다.

그래프에서는 가려져서 안 보이지만, Movielens의 영화의 수는 1996, 1995, 1998, 1997, 1999 순으로 분포되었고, Movielens-IDMb의 경우는 1993, 1995, 1996, 1997, 1999 순으로 분포되었다.



[그림 21] 평점이 포함된 Movielens와 Movielens-IMDb 데이터 셋의 연도별 영화 편수

1.3) 데이터 전처리

1.3.1) 자연어 처리를 위한 전처리

자연어는 사람을 위한 언어이기에 사람은 바로 이해할 수 있으나, 기계는 자연어를 바로 이해할 수가 없고, 숫자 형태로 변경해야 인식을 할 수 있다. 또한 Raw 자연어 데이터의 경우 자연어가 아닌 여러 노이즈가 섞여 있을 수 있다. 예를 들면 웹에서 수집한 문서 데이터의 HTML태그나 메일 데이터라면 헤더가 노이즈라고 볼 수 있다. 텍스트가 아닌 노이즈는 제거해야 한다.

표 1은 일반적인 머신러닝의 자연어 처리 프로세스를 보여준다. 본 논문에서는 해당 처리 프로세스로 자연어 처리를 진행하였다. 또한 일반적인 자연어 처리와 같이 없는 단어는 '<unk>', 패딩 처리는 '<pad>', 문장의 시작은 '<cls>', 문장의 끝은 '<eos>'를 사용하였다.

[표 1] 머신러닝의 자연어 처리 프로세스

1. 데이터 청소	HTML 태그나 메일 헤더 등 글이 아닌 노이즈 제거
2. 정규화	대소문자 통일, 표기 오류 수정, 의미 없는 숫자 제거 등
3. 단어 분할	문장을 단어별로 구분
4. 기본형 변환	단어를 기본형으로 변환
5. 불용어 제거	출현 수가 많은 단어나 조사 등 특별한 의미가 없는 단어 제거
6. 단어의 수치화	단어를 머신러닝으로 다룰 수 있도록 수치로 변경

그림 22는 단어 사전(vocabulary)에 대한 값들을 보여준다. 전처리된 단어를 수치로 변환하기 위해서는 단어 사전을 만들어야 하고, 문장에서 분할된 단어들은 단어 사전에 해당하는 숫자 값으로 변경된다. 그림 22의 첫 줄 오른쪽에 보면, 위에서 설명한 '<unk>', '<pad>', '<cls>', '<eos>' 기호들이 0부터 3까지의 숫자 값이 할당되었음을 볼 수 있다.


```
defaultdict(<bound method Vocab._default_unk_index of <torchtext.legacy.vocab.Vocab object at 0x7f0e0c43b8b0>>, {'<unk>': 0, '<pad>': 1, '<cls>': 2, '<eos>': 3,
'a': 4, '.': 5, 'the': 6, ',': 7, 'to': 8, 'of': 9, 'and': 10, 'his': 11, 'in': 12, 'an': 13, 's': 14, 'is': 15, 'with': 16, 'by': 17, 'on': 18, 'after': 19, 'f
rom': 20, 'as': 21, 'he': 22, 'one': 23, 'their': 24, 'man': 25, 'who': 26, 'that': 27, 'for': 28, 'life': 29, 'her': 30, 'into': 31, 'are': 32, 'new': 33, 'whe
n': 34, 'two': 35, 'up': 36, 'help': 37, 'has': 38, 'friends': 39, 'him': 40, 'war': 41, 'find': 42, 'have': 43, 'young': 44, 'they': 45, 'against': 46, 'but':
47, 'world': 48, 'father': 49, 'them': 50, 'school': 51, 'must': 52, 'old': 53, 'over': 54, 'wife': 55, 'back': 56, 'becomes': 57, 'former': 58, 'years': 59, 'h
imself': 60, 'u': 61, 'run': 62, 'love': 63, 'while': 64, 'american': 65, 'death': 66, 'day': 67, 'was': 68, 'its': 69, 'leads': 70, 'space': 71, 'goes': 72, 'o
ut': 73, 'friend': 74, 'sent': 75, 'york': 76, 'nazis': 77, 'finds': 78, 'murder': 79, 'falls': 80, 'john': 81, 'group': 82, 'at': 83, 'high': 84, 'rebels': 85,
'during': 86, 'three': 87, 'tries': 88, 'other': 89, 'star': 90, 'struggles': 91, 'where': 92, 'power': 93, 'discovers': 94, 'police': 95, 'town': 96, 'woman':
97, 'be': 98, 'time': 99, 'named': 100, 'it': 101, 'more': 102, 'escape': 103, 'best': 104, 'small': 105, 'i': 106, 'darth': 107, 'luke': 108, 'vader': 109, 'wh
ose': 110, 'jones': 111, 'vietnam': 112, 'mission': 113, 'officer': 114, 'about': 115, 'planet': 116, 'city': 117, 'german': 118, 'park': 119, 'year': 120, 'al
l': 121, 'own': 122, 'boy': 123, 'go': 124, 'family': 125, 'connor': 126, 'cyborg': 127, 'toy': 128, 'black': 129, 'suburban': 130, 'childhood': 131, 'son': 13
2, 'down': 133, 'corruption': 134, 'lives': 135, 'set': 136, 'marty': 137, 'before': 138, 'emperor': 139, 'past': 140, 'only': 141, 'sets': 142, 'frustrated': 1
43, 'order': 144, 'again': 145, 'not': 146, 'murdered': 147, 'mob': 148, 'lawyer': 149, 'daughter': 150, 'story': 151, 'will': 152, 'can': 153, 'mother': 154,
```


[그림 22] 단어 사전(vocabulary)의 값들

그림 23은 영화 개요(overview)를 전처리 및 수치로 변환한 데이터 셋이다. tokens 특성을 보면 맨 앞에는 2로 시작됨을 볼 수 있는데 이는 시작을 표현하는 '<cls>'의 단어 사전값이다. 역시 마지막에 1이 많음을 볼 수 있는데 이는 최대 길이로 설정된 길이보다 영화 개요가 짧을 경우 '<pad>'로 패딩 된 값들이다.

	user_id	movie_id	rating	gender	age	occupation	zipcode	overview	director	tokens
0	1	1193	5	F	1	10	48067	A criminal pleads insanity and is admitted to a mental institution, where he rebels against the oppressive nurse and rallies up the scared patients.	Milos Forman	[2, 4, 500, 595, 534, 10, 15, 590, 8, 4, 446, 591, 7, 92, 22, 85, 46, 6, 593, 592, 10, 596, 36, 6, 597, 594, 5, 3, 1, 1]
1	1	914	3	F	1	10	48067	Snobbish phonetics Professor Henry Higgins agrees to a wager that he can make flower girl Eliza Doolittle presentable in high society.	George Cukor	[2, 1438, 1436, 194, 193, 1435, 913, 8, 4, 1439, 27, 22, 153, 325, 1134, 287, 1434, 1433, 1437, 12, 84, 313, 5, 3, 1, 1, 1, 1, 1, 1]
2	1	1287	5	F	1	10	48067	After a Jewish prince is betrayed and sent into slavery by a Roman friend, he regains his freedom and comes back for revenge.	William Wyler	[2, 19, 4, 246, 479, 15, 1329, 10, 75, 31, 354, 17, 4, 267, 74, 7, 22, 1331, 11, 1330, 10, 449, 56, 28, 1038, 5, 3, 1, 1, 1]
3	1	595	5	F	1	10	48067	A prince cursed to spend his days as a hideous monster sets out to regain his humanity by earning a young woman's love.	Gary Trousdale	[2, 4, 479, 1047, 8, 896, 11, 825, 21, 4, 1049, 626, 142, 73, 8, 1051, 11, 1050, 17, 1048, 4, 44, 97, 14, 63, 5, 3, 1, 1, 1]
4	1	2398	4	F	1	10	48067	When a nice old man who claims to be Santa Claus is institutionalized as insane, a young lawyer decides to defend him by arguing in court that he is the real thing.	George Seaton	[2, 34, 4, 952, 53, 25, 26, 1770, 8, 98, 1774, 1771, 15, 1773, 21, 532, 7, 4, 44, 149, 1565, 8, 1478, 40, 17, 1769, 12, 1772, 27, 3]
5	1	2918	4	F	1	10	48067	A high school wise guy is determined to have a day off from school, despite what the Principal thinks of that.	John Hughes	[2, 4, 84, 51, 732, 726, 15, 680, 8, 43, 4, 67, 255, 20, 51, 7, 189, 176, 6, 728, 731, 9, 27, 5, 3, 1, 1, 1, 1, 1]
6	1	2791	4	F	1	10	48067	A man afraid to fly must ensure that a plane lands safely after the pilots become sick.	Jim Abrahams	[2, 4, 25, 583, 8, 585, 52, 584, 27, 4, 587, 295, 588, 19, 6, 586, 324, 589, 5, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
7	1	3105	5	F	1	10	48067	The victims of an encephalitis epidemic many years ago have been catatonic ever since, but now a new drug offers the prospect of reviving them.	Penny Marshall	[2, 6, 1383, 9, 13, 1376, 1377, 1378, 59, 1035, 43, 166, 1375, 687, 550, 7, 47, 227, 4, 33, 209, 1380, 6, 1381, 9, 1382, 50, 5, 3, 1]

[그림 23] 토큰화가 완료된 데이터 셋

1.3.2) 자연어의 벡터화

영화 설명 특성을 자연어 처리하기 위해, 단어의 분산 표현은 영어판 Fasttext (wiki-news300d-1M.vec)를 사용하였다. wiki-news300d-1M.vec는 위키피디아 2017, UMBC 웹베이스 코퍼스 및 statmt.org 뉴스 데이터 셋으로 훈련한 1백만 개의 단어 벡터이다 [9].

한 단어를 표현하는 임베딩 벡터의 차원은 300이고 전체 단어 수는 999,994개이다.

2) 훈련 데이터 셋과 평가 데이터 셋

훈련과 평가를 위해 훈련 데이터 셋(Tranin Data Set)과 평가 데이터 셋(Test Data Set)을 8:2의 비율로 분리했다. 분리를 위해 Scikit-Learn의 train_test_split 함수를 사용했다. 설정값으로는 random_state = 1004로 하였다. 또 훈련 중 검증을 위해, 훈련 데이터 셋과 검증 데이터 셋을 훈련 데이터 셋에서 8:2의 비율로 분리했다.

3) 층 및 매개변수 설정

- 그림 5의 Added Part에서 FM의 출력을 입력으로 받는 ①번 DNN 설정의 경우 - 4개의 층, 매개변수 값: 1, 64, 128, 128.
- Deep의 출력을 입력으로 받는 ②번 DNN 설정 - 4개의 층, 매개변수 값: 128, 128, 128, 128.
- 트랜스포머 인코더의 출력을 입력으로 받는 ③번 DNN 설정 - 4개의 층, 매개변수 값: 128, 128, 128, 128. ②번 DNN 설정과 동일하다.
- 각 DNN의 출력값을 조합한 벡터를 입력으로 받는 최종 ④번 DNN 설정 - 5층, 매개변수 값: 128, 256, 128, 64, 1로 설정하였다.

4) 평가 지표

본 논문에서는 사용자의 평점(5점 만점)을 예측하고, 평균 제곱근 오차(RMSE, Root Mean Squared Error)라는 평가 지표를 사용하여 모델을 평가하였다. 수식 3은 RMSE의 식이다.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i}^{N}(pred_i - target_i)^2} \quad (3)$$

5) 모델 비교

실험에서는 Wide & Deep, DeepFM, xDeepFM [15] 및 DeepFM-Attention 4개의 모델을 비교하였다. Wide & Deep, DeepFM, xDeepFM 3개의 모델에서는 제품 설명(텍스트) 데이터를 제외한 사용자 아이디, 영화 아이디, 직업, 성별, 연령, 장르, 감독 7개의 특성을, DeepFM-Attention에는 영화 설명데이터를 포함한 8개의 특성을 사용하였다.

6) 하이퍼 파라미터 설정

MovieLens-IMDB 데이터 셋을 사용하여 모델을 평가하기 위해 설정한 하이퍼 파라미터는 표 2와 같다.

[표 2] 하이퍼 파라미터 설정값

하이퍼 파라미터	값
Dropout	0
Batch Size	128
Optimizer	Adam
Activation Function	ReLU

Dropout은 0일 때 전체 모델이 전부 가장 좋은 성능이 나왔기 때문에 0으로 설정했다.

7) 컴퓨터 사양

실험을 위해서 사용한 컴퓨터는 개인 PC를 사용하였고, 하드웨어 사양은 표 3과 같다.

[표 3] 실험용 PC의 하드웨어 사양

하드웨어	사 양
CPU	12th Gen Intel(R) Core(TM) i7-12700KF
Mainbord	ASUS PRIME Z690-A
RAM	DDR5 64G
GPU	NVIDIA GeForce RTX 2080 Ti

실험을 위해 사용한 프로그래밍 언어는 Python, 딥러닝 프레임워크는 PyTorch, 라이브러리는 pandas이며 버전은 표 4와 같다.

[표 4] 프로그래밍 언어 및 라이브러리 버전

언어, 프레임워크 및 라이브러리	버전
Python	3.9.18
PyTorch	1.9.0+cu102
pandas	1.4.4

8) 훈련 시간 및 Epoch

본 논문에서 제안하는 DeepFM-Transformer의 경우 한 epoch당 13초의 시간이 걸렸고, 비교 모델인 DeepFM, xDeepFM, Wide&Deep의 경우는 한 epoch당 3초의 시간이 걸렸다. 모델당 20 epoch까지 실험하였고, 대부분 최고성능은 10~12 epoch에서 보여주었다. DeepFM-Transformer의 경우 12 epoch일 때 최고성능을 보여주었다.

2. 성과평가

1) DeepFM-Transformer의 최종 DNN에서 첫 번째 층의 뉴런 수 별 성능

먼저 본 논문에서 제안하는 모델인 DeepFM-Transformer의 최종 DNN에서 첫 번째 층의 뉴런 수를 달리하여 실험하였다. 나머지 층의 뉴런 수는 256, 128, 64, 1로 설정하였다. 표 5에서 해당 성능을 알 수 있는데, 뉴런 수가 128일 때 RMSE 점수가 0.84988로 가장 좋은 성능을 보여주었다.

[표 5] DeepFM-Transformer의 최종 DNN에서 첫 번째 층의 뉴런 수 별 RMSE 점수

	64	128	256	521
RMSE	0.85932	0.84988	0.85797	0.85751

2) 모델별 RMSE 점수

표 6 는 Movielens-IMDb 데이터 셋의 평가(test) 데이터 셋에 대한 실험 모델별 RMSE 점수를 보여준다. 표 6에서 우리가 제안한 모델은 RMSE 점수 0.84988을 달성하여 다른 모델보다 성능이 우수함을 확인 할 수 있다. 제안된 모델은 DeepFM보다 0.00715, xDeepFM보다 0.01503, Wide & Deep보다 0.02098 낮은 RMSE 점수를 달성했다. 전체적인 설정값은 앞서 밝힌 실험 설정에 해당하는 값으로 하였다.

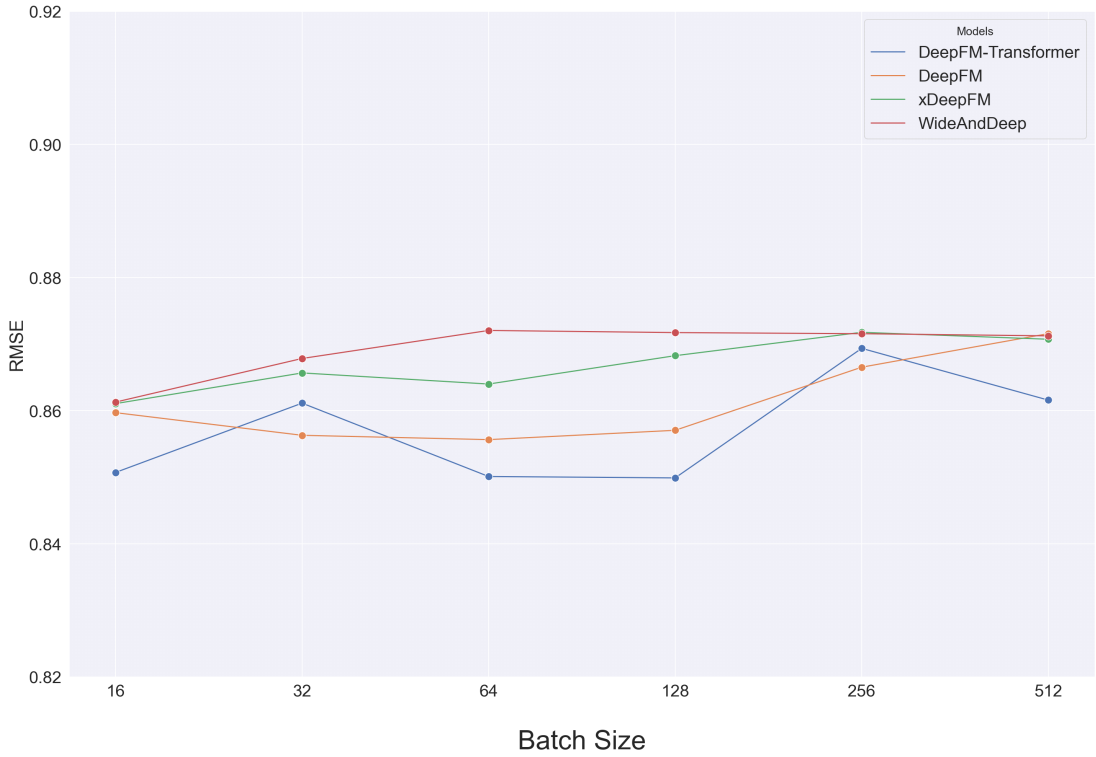
[표 6] Movielens-IMDb 데이터 셋에 대한 모델별 RMSE 점수

	DeepFM-Transformer	DeepFM	xDeepFM	Wide&Deep
RMSE	0.84988	0.85703	0.86491	0.87086

3) 배치 크기 별 각 모델의 RMSE 점수

그림 24는 배치 크기 별 각각의 모델들의 RMSE 값을 보여준다. 배치 크기는 관례에 따라 8의 배수로 하여 [16, 32, 64, 128, 256, 512] 6가지로 실험하였다. 다른 하이퍼 파라미터는 실험 설정에 밝힌 값을 사용하였다.

해당 실험에서 모델별로 가장 성능이 좋은 배치 사이즈가 서로 달랐음을 알 수 있다. 본 논문에서 제시하는 DeepFM-Transformer의 경우 배치 크기가 128일 때 가장 좋은 성능을 보였으며, 64, 16일 때도 전체 모델 중 가장 좋은 성능을 보여주었다. DeepFM의 경우 64일 때 가장 좋은 성능을 보여주었고 512에서 가장 안 좋은 성능을 보여주었다. xDeepFM과 Wide & Deep의 경우 배치 크기가 16일 때 가장 좋은 성능을 보여주었으며, xDeepFM의 경우 배치 크기가 256일 때 Wide & Deep의 경우 배치 크기가 128일 때 가장 안 좋은 성능을 보여주었다. 전체적으로 배치 크기가 256, 512일 때 가장 안 좋은 성능을 보여주었다.



[그림 24] 배치 크기별 실험 모델의 RMSE 값

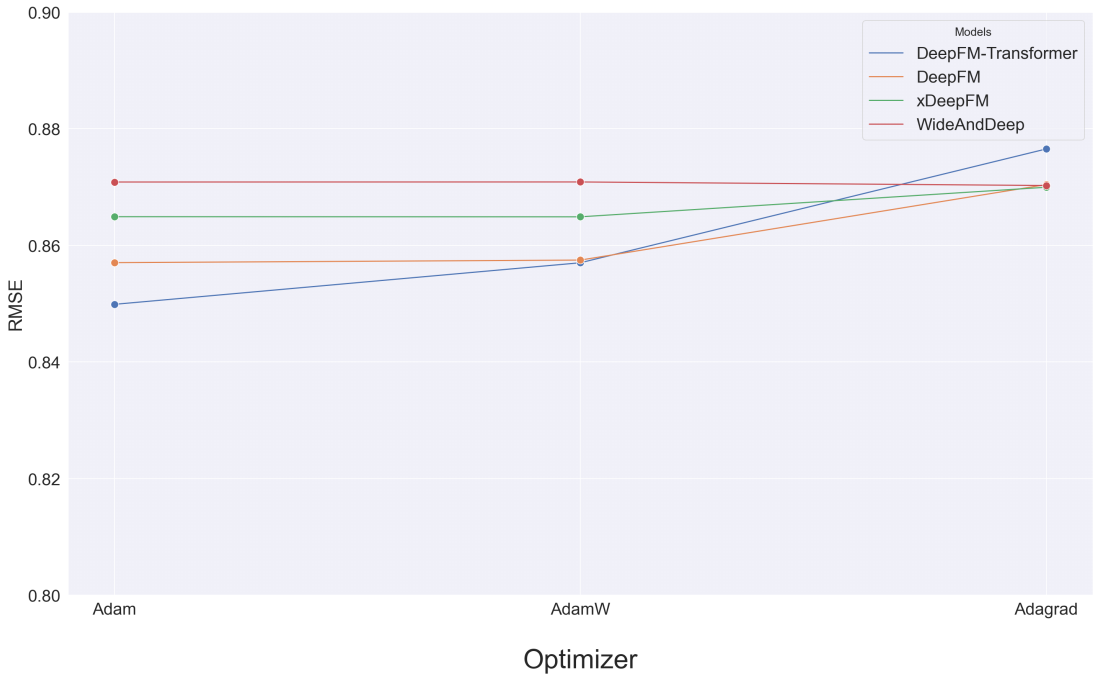
표 7은 배치 크기별 실험 모델의 평균 RMSE 값이다. 배치 크기 별 평균 RMSE에서도 본 논문에서 제안하는 DeepFM-Transformer가 가장 좋은 성능을 보인 것을 알 수 있다. 제안된 모델은 DeepFM보다 0.003998, xDeepFM보다 0.00978, Wide & Deep보다 0.012149 낮은 RMSE 점수를 보였다.

[표 7] 배치 크기별 실험 모델의 평균 RMSE 값

	DeepFM-TransformerT	DeepFM	xDeepFM	Wide&Deep
Mean RMSE	0.857115	0.861113	0.866895	0.869265

4) 옵티마이저 별 각 모델의 RMSE 점수

그림 25는 각 옵티마이저 별 실험 모델들의 RMSE 값을 보여준다. Adam, AdamW 및 Adagrad의 세 가지 옵티마이저가 사용되었다. 전체 모델은 Adam을 사용할 때 가장 좋은 성능을 보인 반면, DeepFM-Transformer와 DeepFM은 Adagrad를 사용할 때 낮은 성능을 보였다. xDeepFM과 Wide & Deep의 경우는 전체적으로 비슷한 RMSE 값을 보여주었다.



[그림 25] 옵티마이저 별 실험 모델의 RMSE 값

V. 결 론

1. 결론

제품 설명데이터에는 제품에 대한 많은 정보가 존재 한다. 본 논문에서는 기존의 Factorization-Machine 기반의 신경망 추천 모델에서 보여주지 못했던, 특성과 제품 설명 텍스트 데이터 간의 상호작용하는 학습을, 텍스트 데이터를 NLP한 벡터와 다른 특성 간의 상호작용이 가능한 신경망을 구성함으로써, 텍스트 데이터도 다른 특성들과 상호작용하는 학습이 가능한 DeepFM-Transformer를 제안하였다.

실험 결과 자연어 처리한 제품의 설명 텍스트 데이터와 사용자, 제품의 특성을 서로 조합하여 추천 모델링한 DeepFM-Transformer가 MovieLens-1M과 IMDB top1000를 병합한 MovieLens-IMDB 데이터 셋에서, Wide & Deep, DeepFM, xDeepFM보다 RMSE에서 더 좋은 성능을 보여주었다. Factorization-Machine 기반의 신경망 모델에서 특성 간 상호작용하는 것과 텍스트 데이터의 조합으로 추천이 가능한 것을 보여준 연구라 할 수 있다.

다양한 추천 시스템(제품, 영화, 음악, 강의 등)에 사용되는 학습 데이터에는 설명데이터가 존재할 가능성이 높기 때문에, 다양한 분야의 추천 연구에도 도움이 될 것으로 생각된다.

2. 향후 연구

본 연구에서는 특성 간의 상호작용과 제품 설명 텍스트 데이터를 조합하는 새로운 형태의 추천 방법을 제시하였다. 본 연구의 방식은 새로운 접근법이라 볼 수 있지만, 본 연구에 사용된 Transformer의 경우 Multi-Head Attention을 적용하지 않은 Sing-Head Attention이었으며, NLP에서 성능이 좋은 최신 모델(Bert,

GPT 등 [16][17])이 있기 때문에, 본 연구에 사용된 Transformer Encoder 부분을 최신 NLP 모델로 교체하여 연구하는 것이 필요할 것으로 보인다. 다만 최신 모델 등은 규모가 크기 때문에 훈련 시간 등을 고려해야 할 것이다.

또한 텍스트 데이터와 다른 특성 데이터의 조합하는 추천이 가능한 만큼, 사용자의 리뷰 데이터를 다른 특성 데이터와 조합하여 추천에 적용하는 연구도 필요할 것으로 보인다.

참고문헌

[1] Edgars Kebbe. (2023, June 9). Why Are Product Description Important?. ecomstrive.com
https://ecomstrive.com/why-are-product-descrption-important/
[2] DesiCity. (2023, March 7). The Importance of Great Product Descriptions in e-Commerce. inkedin.com
https://www.linkedin.com/pulse/importance-great-product-descriptions-e-commerce-desicity
[3] Steffen Rendle. Factorization machines. In 2010 IEEE International conference on data mining, pages 995-1000. IEEE, 2010.
[4] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. Deepfm: a factorizationmachine based neural network for ctr prediction. arXiv preprint arXiv:1703.04247, 2017.
[5] Riskworks. (2012, July 17). Diapers, Beer, and Data Science in Retail. canworksmart.com.
https://canworksmart.com/diapers-beer-retail-predictive-analytics/
[6] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. Wide & deep learning for recommender systems. In Proceedings of the 1st workshop on deep learning for recommender systems, pages 7-10, 2016.
[7] Jack W Rae and Ali Razavi. Do transformers need deep long-range memory. arXiv preprint arXiv:2007.03356, 2020
[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is

all you need. Advances in neural information processing systems, 30, 2017.
[9] Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., & Joulin, A. (2017). Advances in pre-training distributed word representations. arXiv preprint arXiv:1712.09405.
[10] 오가와 유타로. 만들면서 배우는 파이토치 딥러닝. 한빛미디어, 2021.
[11] Harper, F. M., & Konstan, J. A. (2015). The movielens datasets: History and context. Acm transactions on interactive intelligent systems (tiis), 5(4), 1-19.
[12] paperswithcode. MovieLens. paperswithcode.com.
https://paperswithcode.com/dataset/movielens/
[13] imdb. About IMDb. imdb.com.
https://www.imdb.com/pressroom/about/
[14] Tréa Lavery, Editorial Assistant (2017, March). Internet Movie Database (IMDb). techtarget.com.
https://www.techtarget.com/whatis/definition/Internet-Movie-Database-IMDb
[15] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, Guangzhong Sun. (2018). xDeepFM: Combining explicit and implicit feature interactions for recommender systems. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK; pp. 1754-1763.
[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.
[17] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.