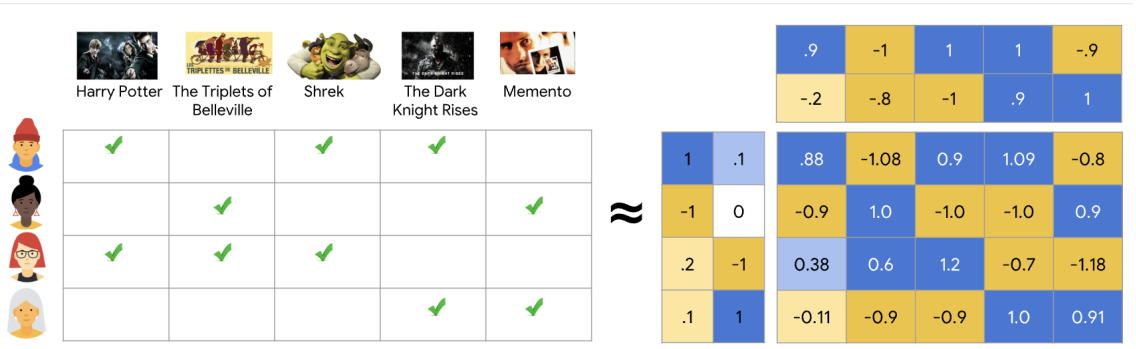
Neural Collaborative Filtering

Background

- Contents Based Filtering
  - 유저가 아이템 A를 좋아할 때, A와 비슷한 특징을 갖는 아이템을 추천해주는 방식
    - ex) 장르가 비슷하거나, 감독이 똑같거나 등등
- Collaborative Filtering
  - 사용자의 행동 양식(평점, 구매이력) 을 기반으로 하여 추천
    - ex) 사람들의 평정을 보고 영화를 볼지 안볼지 결정하는 것과 유사
  - 최근접 이웃 기반(nearest neighbor collaborative filtering)
    - 사용자 기반 (user based collaborative filtering)
      : 사용자의 구매 패턴(평점)과 유사한 사용자를 찾아서 추천 리스트 생성
      (유저 A와 B가 유사하다면, B가 좋아한 상품을 A도 좋아하겠지)
    - 아이템 기반 (item based collaborative filtering)
      : 특정 사용자가 준 점수간의 유사한 상품을 찾아서 추천 리스트 생성
      (Item A와 Item B가 유사한 평점 분포를 가졌다면, B를 좋아한 유저는 A 상품도 좋아하겠지)
  - Latent Based Collaborative Filtering : Matrix Factorization



    - SVD(특이값 분해) 의 응용

    - 주어진 사용자-아이템 상호작용 정보를 이용하여 사용자와 아이템의 잠재적인 특성을 추출한다
    - 새로운 아이템에 대해서도 (사용자와 아이템간의 상호작용이 없는) 예측 평점을 낼 수 있다
    - 차원 축소의 효과가 있다 (메모리 사용량을 줄이고 연산 속도를 향상 시킬 수 있다)

Abstract

- DNN의 활용이 추천 시스템에서 큰 주목을 받지 못했다
- implicit data를 기반으로 한 협업 필터링 문제에 대해 신경망 기반 기술을 개발하고자 노력
- 과거에도 추천에 DNN이 사용된 적은 있으나, 보조하는 정도로 사용되었으며 유저와 아이템의 관계에 대해서는 여전히 MF에 의존적
- 내적을 신경망 아키텍처로 대체함으로써 협업 필터링에 대한 NCF(Neural network-based Collaborative Filtering)이라는 일반적인 프레임워크를 제안
- **비선형성을 강화**하기 위해 NCF 모델링에는 다층 퍼셉트론(multi-layer perceptron)을 활용하여 사용자-아이템 상호작용 함수를 학습

1.Introduction

- Matrix Factorization은 netflix prize 이 후 가장 인기가 많은 추천 방법이 되었다
- 하지만 Matrix Factorization의 한계가 존재한다
    - 내적은 사용자와 아이템의 잠재 요인을 선형적으로 결합하는 방식이기 때문에, 사용자와 아이템 간의 복잡한 상호작용 구조를 제대로 포착하지 못할 수 있다
- 이 논문에서는 비선형성을 줄 수 있는 DNN을 이용하여 협업 필터링에 접근한다
- 수집하기 쉬운 implicit feedback만 사용하였다 (noisy implicit feedback signal)

2. Preliminaries

2-1. Learning from Implicit Data

$$y_{ui} = \begin{cases} 1, & \text{if interaction (user } u\text{, item } i\text{) is observed;} \\ 0, & \text{otherwise.} \end{cases}$$

- Implicit feedback은 상호작용 정보가 관측되었을 경우 1, 관측되지 않은 경우는 0
- 값이 1이면 사용자 u와 항목 i 간에 상호작용이 있다(선호를 의미하지는 않는다)
- 값이 0인 경우에도 u가 i를 좋아하지 않는다는 것을 의미하지는 않는다(사용자가 항목을 인식하지 못하는 것일 수도 있다 - 결측)
- 선호에 대한 노이즈가 많이 포함되어 있으며, 부정적인 피드백이 부족하고 결측 데이터가 많기 때문에 정확한 예측이 어렵다.
  - 따라서 관측되지 않은 항목의 점수를 예측하는 문제가 된다
- 이런 문제를 풀 때 보통 다음과 같은 loss(point wise loss function, pairwise loss function)가 사용되며, 논문에서는 모두를 이용한다

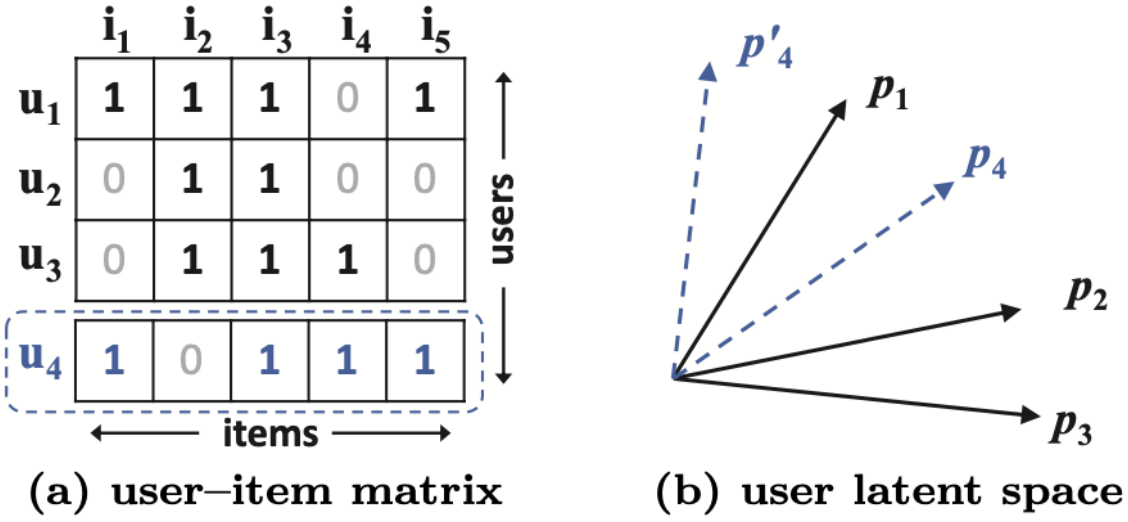
$$L_1 = min\ \frac{1}{2}(\hat{y}_{u,i} - y_{u,i})^2$$
$$L_2 = max(0, f(y_{unobs}) - f(y_{obs}) + \alpha)\ \ s.t\ \ rank(y_{obs}) > rank(y_{unobs})$$

2-2. Matrix Factorization

- MF는 아래 식과 같이 예측 값을 p와 q 벡터의 내적으로 계산하게 된다

$$\hat{y}_{ui} = f(u, i|\mathbf{p}_u, \mathbf{q}_i) = \mathbf{p}_u^T \mathbf{q}_i = \sum_{k=1}^{K} p_{uk} q_{ik},$$

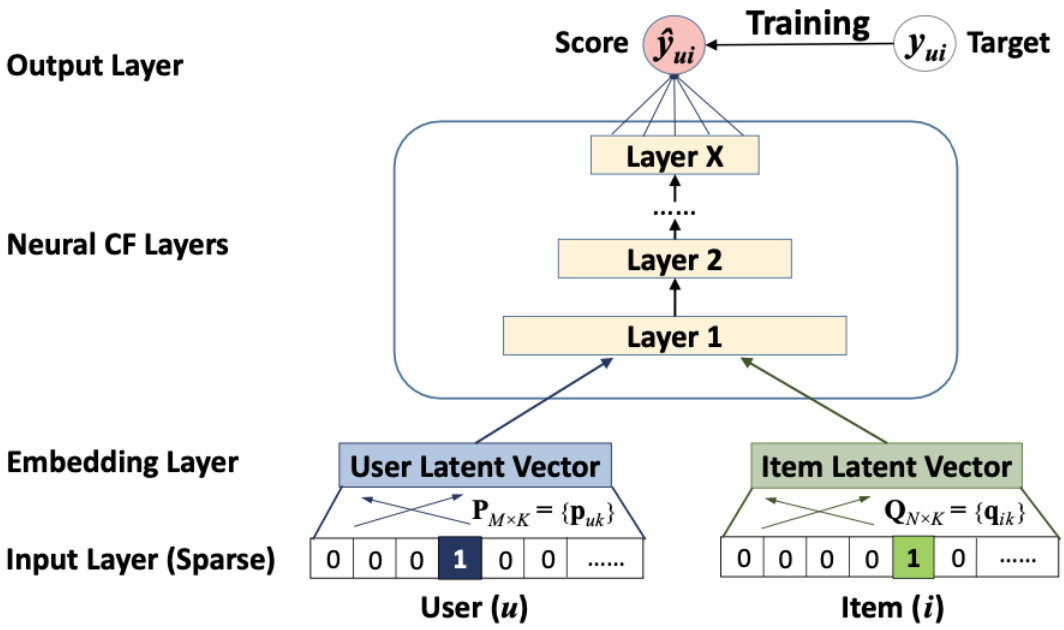
- Matrix Factorization에는 한계가 존재한다



- MF는 사용자와 아이템을 동일한 잠재 공간에 매핑하기 때문에 두 사용자 간의 유사성도 내적으로 측정할 수 있다
- 잠재 벡터 사이의 코사인 유사도를 측정할 수 있다
- (a)에서 s23(0.66) > s12(0.5) > s13(0.4) 임을 알 수 있다 이는 (b)에서도 잘 반영된다
  - 2,3인 가장 유사하고, 그 다음은 1,2이 유사, 그 다음은 1과 3이 유사
- (a)에서 새로운 유저 4가 추가 되었을 때 s41(0.6) > s43(0.4) > s42(0.2) 로 계산된다
  - 4는 1과 제일 유사하며, 그 다음으로 3, 2 순으로 유사하다
- 하지만 (b)에 표현된 벡터를 봤을 때, 4가 3보다 2에 더 가까움을 알 수 있다

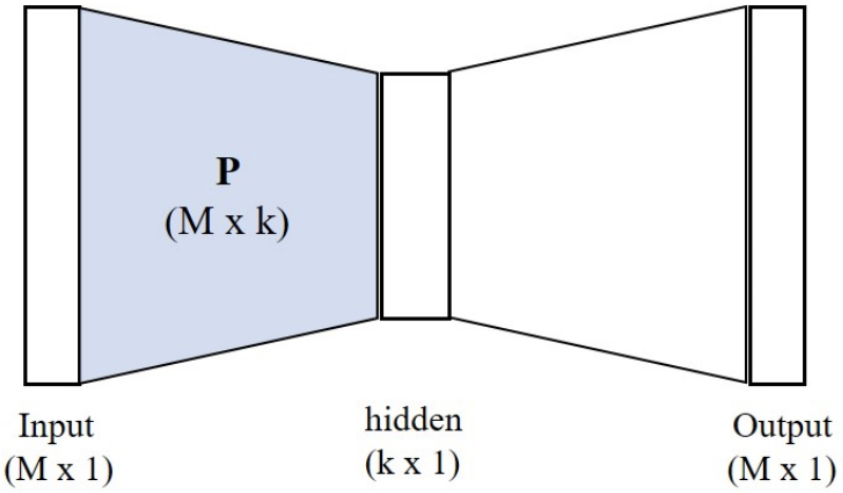
- 저차원 잠재 공간에서 복잡한 사용자-아이템 상호작용을 추정하기 위해 단순하고 고정된 내적을 사용하는 MF의 가능한 한계
- 본 논문에서는 데이터로부터 DNN을 사용하여 상호작용 함수를 학습함으로써 이러한 한계를 해결

3.Neural Collaborative Filtering



3-1. General Framework

- Input one-hot vector -> Embedding layer -> Neural collaborative layer -> Output layer 의 구조
- **input one-hot vector**
  - 순수한 협업 필터링 설정에 초점을 맞추기 때문에, 입력값으로 사용자와 아이템의 신원(identity)한 원핫 인코딩 벡터가 들어간다
- **Embedding layer**
  - 임베딩 레이어는 희소한 표현을 밀집 벡터로 변환하기 위한 fully connected layer, embedding vector값은 MF관점에서 잠재 벡터와 비슷한 맥락



- **Neural collaborative layer**
  - user latent vector 와 item latent vector를 concatenation한 벡터를 input으로 받아 deep neural network 통과하는 단계
- **Output layer**
  - 0과 1 사이의 값을 출력한다

3-1-1. *Learning NCF*

- 이진 데이터로 구성되어있기 때문에 다음과 같은 likelihood function을 얻을 수 있다
  - 동전 던지기와 유사
  - 관측 데이터로 부터 모델을 가장 잘 설명할 수 있는 파라미터 값을 찾는 것
  - likelihood function 최대가 되는 것이 모델을 제일 잘 설명하는 파라미터 값을 찾은 것

$$p(\mathcal{Y}, \mathcal{Y}^- | \mathbf{P}, \mathbf{Q}, \Theta_f) = \prod_{(u,i) \in \mathcal{Y}} \hat{y}_{ui} \prod_{(u,j) \in \mathcal{Y}^-} (1 - \hat{y}_{uj}).$$

- 손실 함수는 해당 likelihood에 negative logarithm을 적용하여 다음과 같다

$$L = -\sum_{(u,i)\in\mathcal{Y}} \log \hat{y}_{ui} - \sum_{(u,j)\in\mathcal{Y}^-} \log(1-\hat{y}_{uj})$$
$$= -\sum_{(u,i)\in\mathcal{Y}\cup\mathcal{Y}^-} y_{ui}\log \hat{y}_{ui} + (1-y_{ui})\log(1-\hat{y}_{ui}).$$

- 해당 값을 최소로 하는 것이 likelihood function 최대
- binary cross-entropy loss와 같은 수식
  - 실제값이 1이고 예측 값이 1이면 L값은 0 , 실제값이 1이고 예측값이 0 이면 -무한

3-2. Generalized Matrix Factorization (GMF)

- MF는 NCF의 특별한 케이스임을 설명

$$\phi_1(\mathbf{p}_u, \mathbf{q}_i) = \mathbf{p}_u \odot \mathbf{q}_i,$$

$$\hat{y}_{ui} = a_{out}(\mathbf{h}^T(\mathbf{p}_u \odot \mathbf{q}_i)),$$

- p와 q를 latent vector라고 했을 때, 이를 element-wise product 한 값에 가중치(h)를 내적하고 activation function을 거친다
- 여기서 h가 uniform vector고 a1이면 MF이 된다
- 논문에서 사용되는 GMF는 a가 sigmoid func이고, h값은 uniform vector가 아닌 값이된다

3-3. Multi-Layer Perceptron (MLP)

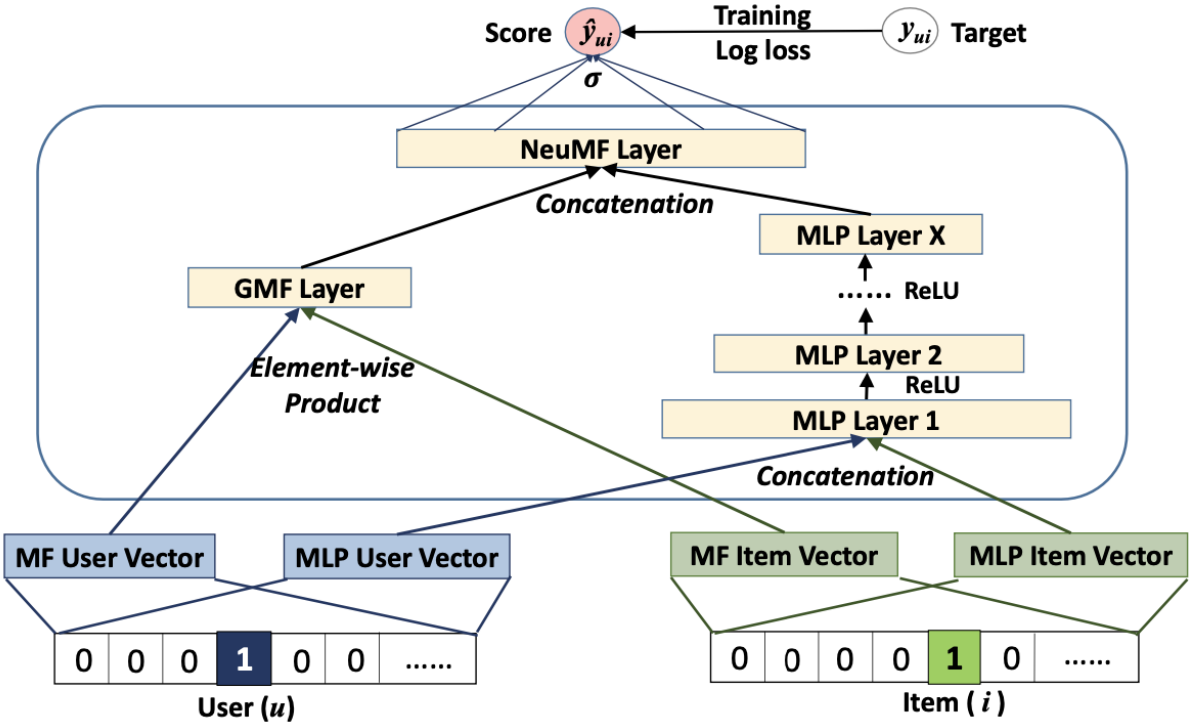
- GMF는 linear하고 fixed한 특징으로 인해 user 와 item간의 복잡한 관계를 표현하지 못함

- MLP는 non-linear하고 flexible 하기 때문에 보다 복잡한 관계를 표현할 수 있다
- user, item embedding vector를 concatenate

$$\mathbf{z}_1 = \phi_1(\mathbf{p}_u, \mathbf{q}_i) = \begin{bmatrix} \mathbf{p}_u \\ \mathbf{q}_i \end{bmatrix},$$
$$\phi_2(\mathbf{z}_1) = a_2(\mathbf{W}_2^T \mathbf{z}_1 + \mathbf{b}_2),$$
$$\cdots\cdots$$
$$\phi_L(\mathbf{z}_{L-1}) = a_L(\mathbf{W}_L^T \mathbf{z}_{L-1} + \mathbf{b}_L),$$
$$\hat{y}_{ui} = \sigma(\mathbf{h}^T \phi_L(\mathbf{z}_{L-1})),$$

- 활성화 함수로는 ReLU 사용

3-4. Fusion of GMF and MLP



$$\hat{y}_{ui} = \sigma(\mathbf{h}^T a(\mathbf{p}_u \odot \mathbf{q}_i + \mathbf{W} \begin{bmatrix} \mathbf{p}_u \\ \mathbf{q}_i \end{bmatrix} + \mathbf{b})).$$

- 각 모델별로 서로 다른 embedding layer를 사용(두 벡터의 차원이 다를 수 있다)
- user-item간의 상호 관계를 표현하기 위해 MF의 linearity 와 MLP의 non-linearity를 결합한 것이 특징(neural matrix factorization**)**

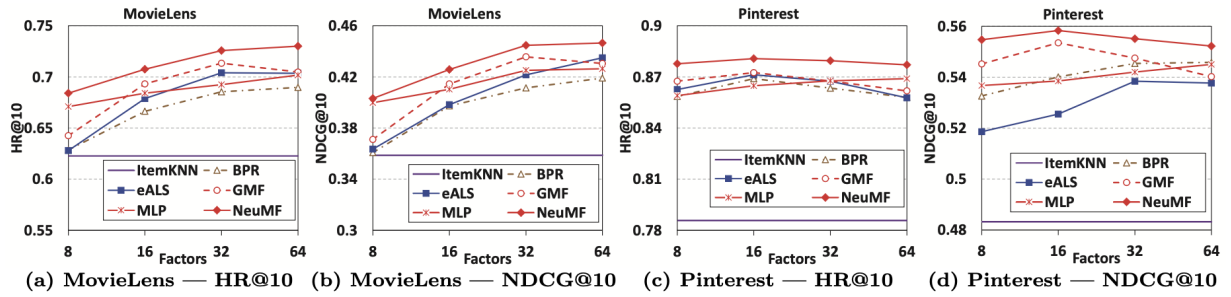
4.Experiments

- **RQ1.** NCF가 그 당시의 SOTA를 능가할 수 있는가**?**
- **RQ2.** 제안한 log loss with negative sampling optimization framework가 추천시스템 task에서 효과가 있는가?
- **RQ3.** user-item interaction 데이터로 학습을 하는 데 깊은 layers가 도움이 되는가?

4-1. Experimental Settings

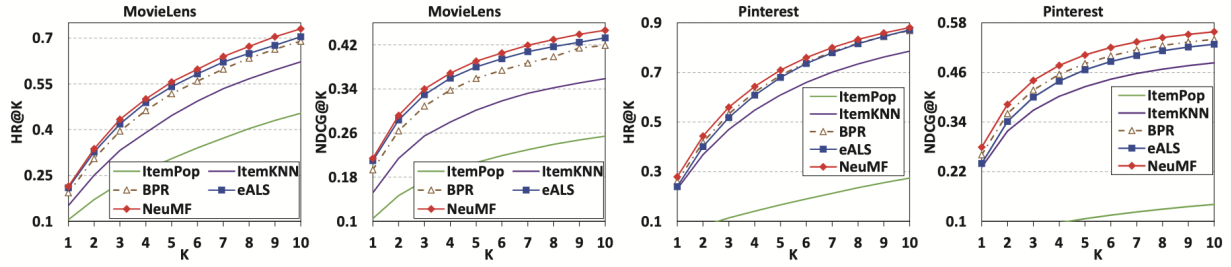
- MovieLens의 ml-1m dataset과 Pinterest의 dataset을 사용
- Baseline Models
  - ItemPop : non-personalized
  - ItemKNN : standard item-based collaborative filtering
  - BPR : pairwise ranking loss를 가지고 MF 모델을 optimize
  - eALS : MF 최신 기술

4-2. Performance Comparison(RQ1)



(a) MovieLens — HR@10 (b) MovieLens — NDCG@10 (c) Pinterest — HR@10 (d) Pinterest — NDCG@10

Figure 4: Performance of HR@10 and NDCG@10 *w.r.t.* the number of predictive factors on the two datasets.

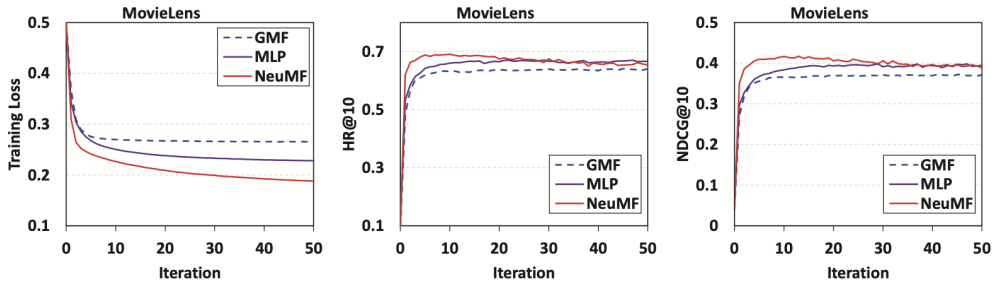


(a) MovieLens — HR@K (b) MovieLens — NDCG@K (c) Pinterest — HR@K (d) Pinterest — NDCG@K

Figure 5: Evaluation of Top-K item recommendation where K ranges from 1 to 10 on the two datasets.

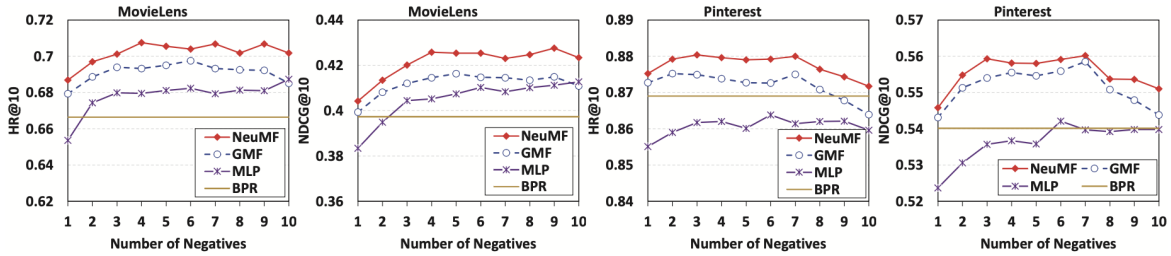
- HR@10 : 10개 중에 몇개를 맞추었는지(hit)를 의미하는 지표입니다.
- NDCG : 랭킹 추천에 많이 쓰이는 지표로, 이상적인 정답 랭킹과 예측 랭킹을 비교하는 지표입니다.
- NeuMF가 모든 부분에 있어 최고 성능을 냈다

4-3. Log Loss with Negative Sampling(RQ2)



(a) Training Loss (b) HR@10 (c) NDCG@10

Figure 6: Training loss and recommendation performance of NCF methods *w.r.t.* the number of iterations on MovieLens (factors=8).



(a) MovieLens — HR@10 (b) MovieLens — NDCG@10 (c) Pinterest — HR@10 (d) Pinterest — NDCG@10

Figure 7: Performance of NCF methods *w.r.t.* the number of negative samples per positive instance (factors=16). The performance of BPR is also shown, which samples only one negative instance to pair with a positive instance for learning.

- 학습이 진행됨에 따라 loss가 지속적으로 감소하고, Loss값도 가장 낮다(implicit data에 대해서 log loss자체가 loss로서 괜찮다는 것을 시사)
- number of negatives(positive instance당 negative instance 수) 에 따른 학습 결과에서도 높은 성능을 보였다
- 다만 number of negatives가 높아짐에 따라 전체적인 성능이 떨어짐을 알 수 있다
- implicit data의 경우 negative data에 노이즈가 있기 때문에 학습에 사용하기 어려움이 있는데 이를 고려하여 성능을 보여주기 위함이 아니였을까

4-4. Is Deep Learning Helpful? (RQ3)

Table 3: HR@10 of MLP with different layers.

Factors	MLP-0	MLP-1	MLP-2	MLP-3	MLP-4
MovieLens					
8	0.452	0.628	0.655	0.671	**0.678**
16	0.454	0.663	0.674	0.684	**0.690**
32	0.453	0.682	0.687	0.692	**0.699**
64	0.453	0.687	0.696	0.702	**0.707**
Pinterest					
8	0.275	0.848	0.855	0.859	**0.862**
16	0.274	0.855	0.861	0.865	**0.867**
32	0.273	0.861	0.863	**0.868**	0.867
64	0.274	0.864	0.867	0.869	**0.873**

- 층이 깊어질수록 성능이 좋은 것을 알 수 있다
- collaborative recommendation에서 deep models을 사용하는 것이 유의미
- 실제로 activation function으로 ReLU 대신 Identity를 써봤더니 성능이 훨씬 떨어졌다
  - non-linear layers가 쌓임에 따라 더 높은 수준의 비선형성이 생긴다
- hidden layer가 없는 MLP-0의 경우 ItemPop 수준의 성능이 나온다. 따라서 은닉층을 통해 적절히 비선형성을 주는 것이 성능에 도움을 준다

5. Related Work

- 초기에는 explicit data에 초점을 맞춘 추천 시스템 연구가 주를 이루었으며, 최근에는 implicit data에 대한 관심이 증가
- 이러한 추세에 맞추어 implicit data갖는 추천 문제를 해결하기 위해 여러 가지 전략과 모델들이 제안되었으며, 이 중 일부는 신경망을 사용하여 구현되었다
- NeuMF가 MF와 MLP를 결합하는 아이디어는 부분적으로 NTN에서 영감을 받았지만, NeuMF는 MF와 MLP가 다른 임베딩 세트를 학습할 수 있는 측면에서 더 유연하다
- 최근 Google은 앱 추천을 위한 Wide & Deep learning 접근 방식은 임베딩에 MLP를 사용하는데, 이는 강력한 일반화 능력을 갖고 있다고 보고된다
- NeuMF는 순수한 협업 필터링 시스템을 위해 DNN을 탐구하는 것에 초점을 맞추었다.

- DNN이 사용자-항목 상호작용을 모델링하는 데 유망한 선택지이다

6. Conclusion

- NCF는 신경망을 사용하는 general한 협업필터링 framework 로서, 간단하게 다양한 모델들을 표현
- MF의 일반화 형태인 GMF와 MLP를 융합하여 NeuCF를 제안하였으며, 기존의 모델에 비해 우수한 성능을 달성