

Neural Collaborative Filtering

소프트웨어학부 2017012333 이수아

CONTETNS

배경지식

- Recommendation system
- Collaborative filtering
- explicit / implicit Dataset

연구주제

요약

결론

Recommendation system





수많은 온라인 서비스 중에서 사용자의 취향을 파악하고 상품을 추천해 주는 시스템
즉, 사용자가 아직 소비하지 않은 아이템 (ex. 영화) 중 선호할 만한 것을 예측하는 것



Collaborative filtering

추천 시스템에 사용되는 대표적인 알고리즘

사용자의 과거 평점 데이터 만으로 아직 평점을 남기지 않은 아이템들에 대한 평점을 예측하는 기법

	M1	M2	M3	M4	M5
	?	1	?	3	?
	1	?	4	?	3
	3	1	?	?	1
	4	?	5	4	4

? : 사용자가 아직 평점을 남기지 않은 아이템

주어진 평점 데이터를 사용하여
예측 평점이 높은 아이템을 추천해주는 방식

배경지식

explicit Dataset

선호와 비선호를 명확하게 구분해준 데이터 셋

호불호에 따라 평점 매기는 것이 대표적인 예시

implicit Dataset

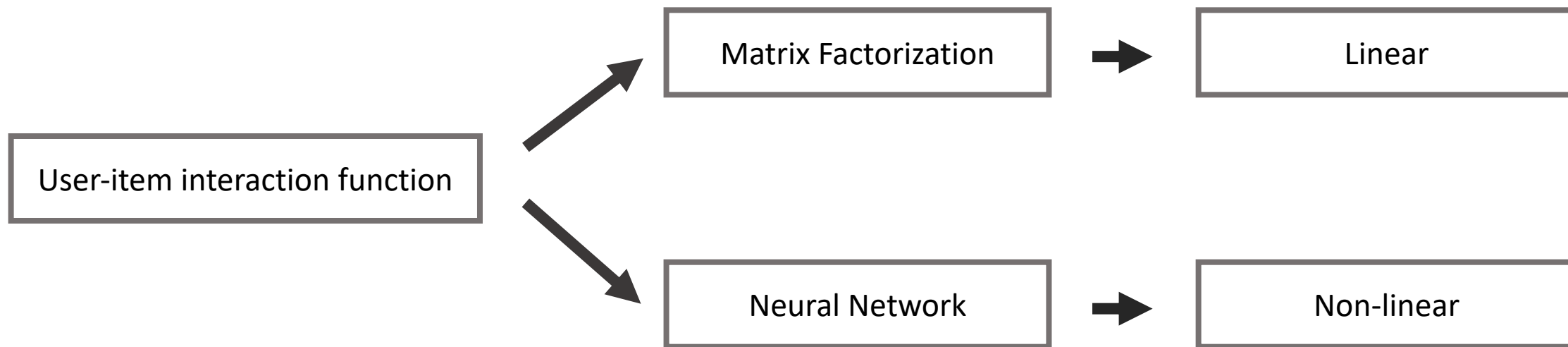
선호와 비선호의 구분 없이 행동의 빈도수만 기록한 데이터 셋

Abstract

기존 MF에 기반한 Collaborative filtering은 user-item 공간의 latent feature들의 **inner product**를 통해 두 관계를 표현
user와 item간의 관계를 학습함에 있어 기존의 liner 방식에 기반한 MF의 한계를 지적

Inner product ➡ Neural achitecture = Neural Collaborative Filtering(NCF)

Introduction



PRELIMINARIES

$$Y_{m,n} = \begin{pmatrix} y_{1,1} & y_{1,2} & \cdots & y_{1,n} \\ y_{2,1} & y_{2,2} & \cdots & y_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ y_{m,1} & y_{m,2} & \cdots & y_{m,n} \end{pmatrix}$$

M, N : # users and items

Y : user-item matrix

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

0은 상호작용이 없는 것이지, 해당 item을 비선호 한다는 의미는 x

Matrix factorization

$$\begin{array}{ccc} Y(\text{user} - \text{item}) & P(\text{user}) & Q(\text{item}) \\ \begin{bmatrix} y_{1,1} & \cdots & y_{1,n} \\ \vdots & \ddots & \vdots \\ y_{m,1} & \cdots & y_{m,n} \end{bmatrix} & = & \begin{bmatrix} p_{11} & \cdots & p_{1k} \\ \vdots & \ddots & \vdots \\ p_{m1} & \cdots & p_{mk} \end{bmatrix} \begin{bmatrix} q_{11} & \cdots & q_{1n} \\ \vdots & \ddots & \vdots \\ q_{k1} & \cdots & q_{kn} \end{bmatrix} \\ m \times n & & m \times k \quad k \times n \end{array}$$

$$\text{where } \mathbf{p}_u = [p_{u1}, \dots, p_{uk}], \quad \mathbf{q}_i = [q_{1i}, \dots, q_{ki}]$$

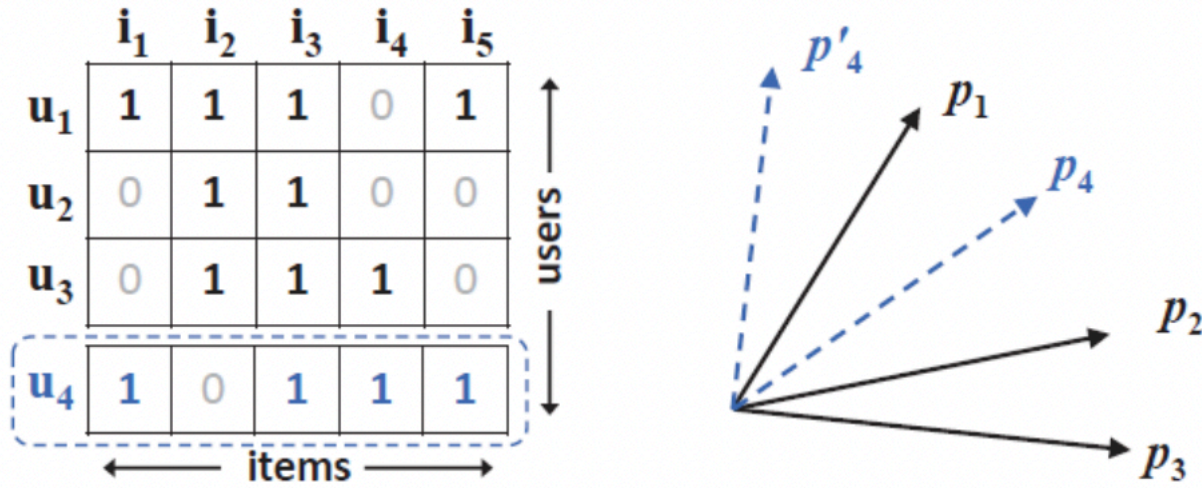
p_u, q_i : latent vector for user u and item i

K : dimension of latent space

저차원의 행렬 2개로 분해하여 표현하는 방법

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

Limitation of MF



Jaccard coefficient : $s_{ij} = \frac{|R_i| \cap |R_j|}{|R_i| \cup |R_j|}$

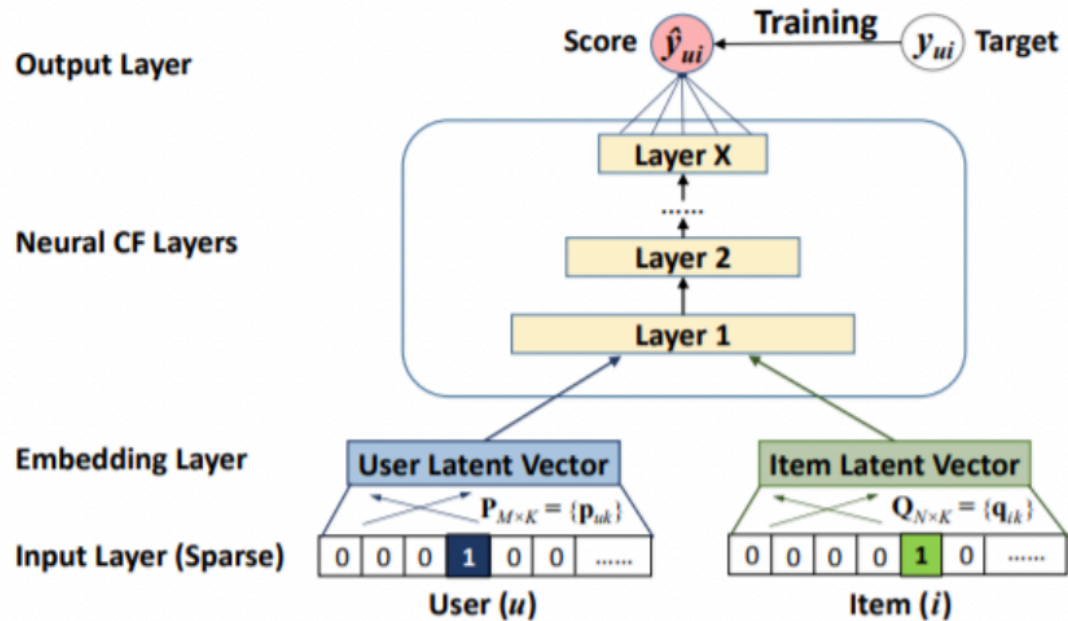
R_u : set of items for user u

Jaccard coefficient를 사용하여 user i, j 의 유사도 측정

$$s_{23}(0.66) > s_{12}(0.50) > s_{13}(0.40)$$

$$s_{41}(0.60) > s_{43}(0.40) > s_{42}(0.20)$$

NEURAL COLLABORATIVE FILTERING



Input layer: one-hot encoding vector

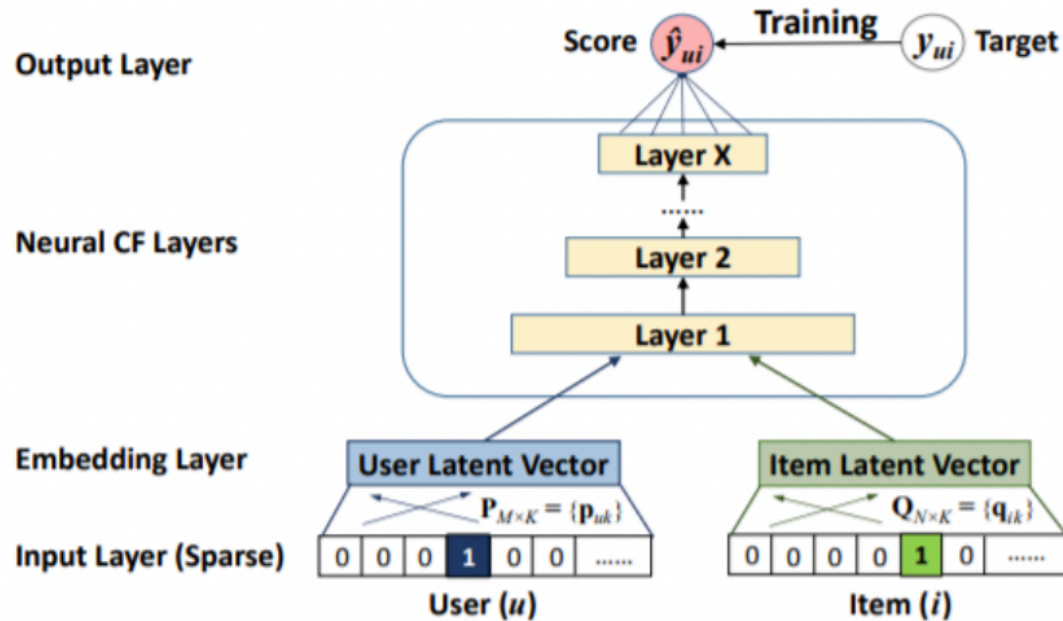
embedding layer: sparse vector \rightarrow dense vector

User latent vector = $P^T v_u^U$

Item latent vector = $Q^T v_i^I$

P, Q : latent factor matrix ; v : one-hot vector

NEURAL COLLABORATIVE FILTERING



Neural CF Layers: concatenate → DNN

Output Layer: $0 \leq \hat{y}_{u,i} \leq 1$

$$\hat{y}_{ui} = f(P^T v_u^U, Q_u^T | P, Q, \Theta_f) = \phi_{out} \left(\phi_x \left(\dots \phi_2 \left(\phi_1 (P^T v_u^U, Q^T v_i^I) \right) \dots \right) \right)$$

ϕ_x : mapping function of x-th neural network

ϕ_{out} : use logistic or probit function

Learning NCF

likelihood function

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

$$y_{u,i} = 0 \text{ or } 1$$

$$0 \leq \hat{y}_{u,i} \leq 1$$

Loss function(BCEloss와 동일)

$$L = -\log p(\mathcal{Y}, \mathcal{Y}^- | P, Q, \Theta_f)$$

$$= - \sum_{(u,i) \in \mathcal{Y}} y_{u,i} \log \hat{y}_{u,i} - \left(\sum_{(u,j) \in \mathcal{Y}^-} (1 - y_{u,i}) \log (1 - \hat{y}_{u,j}) \right)$$

$$= - \left(\sum_{(u,i) \in \mathcal{Y} \cup \mathcal{Y}^-} (y_{u,i} \log \hat{y}_{u,i} + (1 - y_{u,i}) \log (1 - \hat{y}_{u,i})) \right)$$

L을 최소화 하는 parameter 찾는다!

Generalized Matrix Factorization(GMF)

MF는 NCF의 특별한 케이스

User latent vector $p_u = P^T v_u^U$

Item latent vector $q_i = Q^T v_i^I$

NCF layer

$$\phi_1 = p_u \circ q_i$$

output layer

$$\hat{y}_{ui} = a_{out}(h^T(p_u \circ q_i))$$

a_{out} : activation function

h^T : edge weights of the output layer

MF

a_{out} = identity function,

$$h^T = [1, \dots, 1]_{1 \times k}$$

GMF

a_{out} = sigmoid function,

$$h^T = [h_1, \dots, h_k]_{1 \times k}$$

Multi-Layer Perceptron (MLP)

non-linear하고 flexible하기 때문에 보다 복잡한 관계를 표현할 수 있다.

$$\mathbf{z}_1 = \phi_1(\mathbf{p}_u, \mathbf{q}_i) = \begin{bmatrix} \mathbf{p}_u \\ \mathbf{q}_i \end{bmatrix}, \quad \phi_1 : p_u, q_i \text{ concatenate}$$

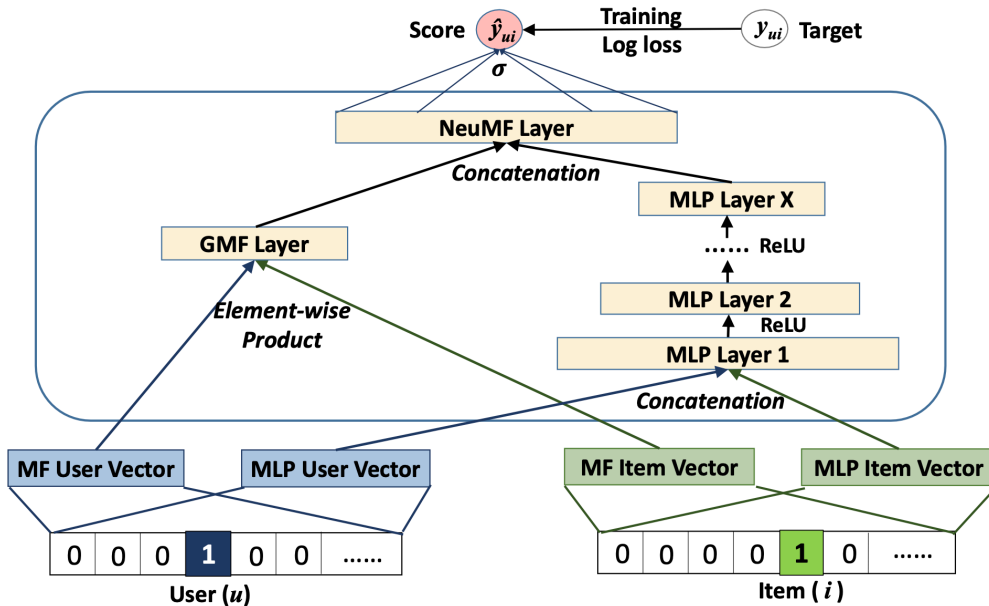
$$\phi_2(\mathbf{z}_1) = a_2(\mathbf{W}_2^T \mathbf{z}_1 + \mathbf{b}_2),$$

.....

$$\phi_L(\mathbf{z}_{L-1}) = a_L(\mathbf{W}_L^T \mathbf{z}_{L-1} + \mathbf{b}_L),$$

$$\hat{y}_{ui} = \sigma(\mathbf{h}^T \phi_L(\mathbf{z}_{L-1})),$$

Fusion of GMF and MLP



서로 다른 embedding layer 사용

$$\phi^{GMF} = \mathbf{p}_u^G \odot \mathbf{q}_i^G,$$

$$\phi^{MLP} = a_L(\mathbf{W}_L^T(a_{L-1}(\dots a_2(\mathbf{W}_2^T \begin{bmatrix} \mathbf{p}_u^M \\ \mathbf{q}_i^M \end{bmatrix} + \mathbf{b}_2)\dots)) + \mathbf{b}_L),$$

$$\hat{y}_{ui} = \sigma(\mathbf{h}^T \begin{bmatrix} \phi^{GMF} \\ \phi^{MLP} \end{bmatrix}),$$

두 vector 차원이 다를 수 있음!

Neural matrix factorization(NeuMF)

: user-item간의 interaction 표현하기 위해 MF의 linearity 와 MLP의 non-linearity를 결합한 모델

Experiment

MovieLens, Pinterest 두개의 데이터로 학습

Pinterest 데이터의 경우 20개 이상 핀을 본 사용자만 데이터에 포함

처음 hyper parameter를 조정하기 위해 사용자 당 하나의 데이터만 추출해서 데이터셋을 만들었음

하나의 positive 당 4개의 negative sample을 뽑아서 학습에 사용

Ranked list의 성능은 Hit Ratio(적중률)과 Normalized Discounted Cumulative Gain(nDCG)으로 판단

GMF, MLP, NeuMF를 **ItemPop**, **ItemKNN**, **BPR**, **eALS** 방법을 사용하여 비교

Experiment

RQ1) 제안된 NCF 방법이 최첨단 implicit collaborative filtering method를 능가하는가?

RQ2) 우리가 제안한 optimization framework(log loss with negative sampling)는 recommendation task에서 어떻게 작동하나?

RQ3) hidden units의 더 깊은 layers가 user-item interaction data로부터 학습하는 데 더 도움이 되는가?

RQ1)

제안된 NCF 방법이 최첨단 implicit collaborative filtering method를 능가하는가?

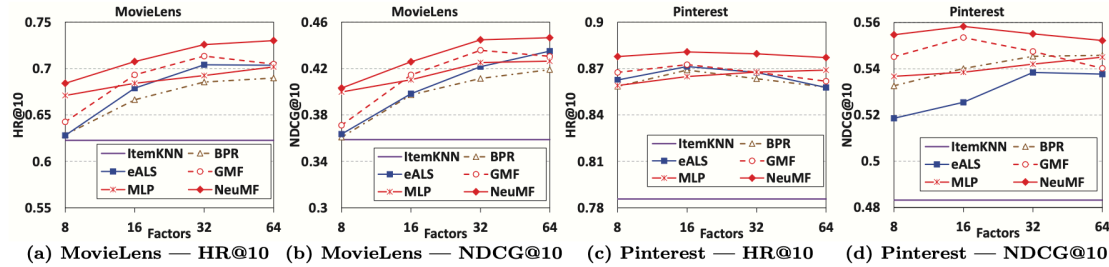


Figure 4: Performance of HR@10 and NDCG@10 *w.r.t.* the number of predictive factors on the two datasets.

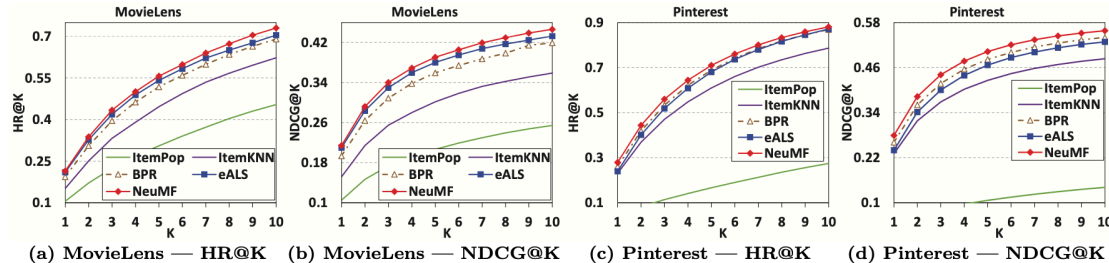


Figure 5: Evaluation of Top- K item recommendation where K ranges from 1 to 10 on the two datasets.

NeuMF가 eALS와 BPR을 크게 능가하여 최고의 성능을 달성

RQ2)

우리가 제안한 optimization framework(log loss with negative sampling)는 recommendation task에서 어떻게 작동하나?

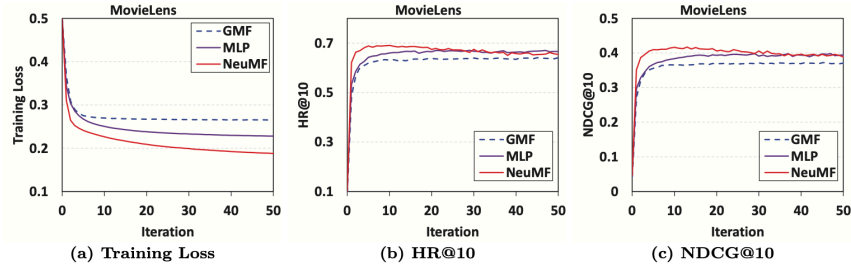


Figure 6: Training loss and recommendation performance of NCF methods *w.r.t.* the number of iterations on MovieLens (factors=8).

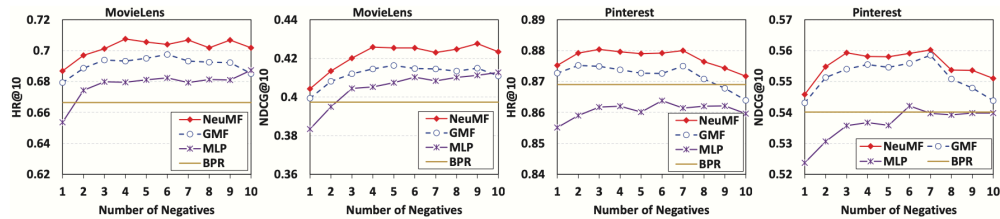


Figure 7: Performance of NCF methods *w.r.t.* the number of negative samples per positive instance (factors=16). The performance of BPR is also shown, which samples only one negative instance to pair with a positive instance for learning.

BCELoss

3에서 6 사이가 optimal

RQ3)

hidden units의 더 깊은 layers가 user-item interaction data로부터 학습하는 데 더 도움이 되는가?

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

MLP에서 레이어를 늘리면 더 잘 학습 ➡ DNN이 적절한 효과를 냄

Conclusion

NeuMF = GMF + MLP

GMF는 MF를 일반화한 모델, MLP는 DNN(deep neural network) 모델

➡ 논문에서 제시된 NCF(neural collaborative framework)로 표현 가능

collaborative filtering의 핵심(user와 item의 상호작용 모델링)을 놓치지 않으면서 성능은 높은 방법

➡ Linear space에 기반한 기존 모델들이 갖는 한계를 DNN을 도입해 해결할 수 있었기 때문

나아가 DNN에만 의존한 것이 아닌 두 모델을 통합함으로써 더 큰 성능 향상을 보일 수 있었다.

기존의 여러 모델들(Neural Tensor Network, Wide & Deep learning)과 아이디어는 비슷하지만

collaborative filtering의 아이디어를 실현한다는 점에서 가장 큰 contribution을 갖는다.

끝