Deep Distribution Network: Addressing the Data Sparsity Issue for Top-N Recommendation

Lei Zheng
Department of Computer Science
University of Illinois at Chicago
IL, US
lzheng21@uic.edu

Chaozhuo Li*
Department of Computer Science
Beihang University
Beijing, China
lichaozhuo@buaa.edu.cn

Chun-Ta Lu
Department of Computer Science
University of Illinois at Chicago
IL, US
clu29@uic.edu

Jiawei Zhang
IFM Lab, Department of Computer Science
Florida State University
FL, US
jiawei@ifmlab.org

Philip S. Yu
Department of Computer Science
University of Illinois at Chicago
IL, US
psyu@uic.edu

ABSTRACT

Existing recommendation methods mostly learn fixed vectors for users and items in a low-dimensional continuous space, and then calculate the popular dot-product to derive user-item distances. However, these methods suffer from two drawbacks: (1) the data sparsity issue prevents from learning high-quality representations; and (2) the dot-product violates the crucial triangular inequality and therefore, results in a sub-optimal performance.

In this work, in order to overcome the two aforementioned drawbacks, we propose Deep Distribution Network (DDN) to model users and items via Gaussian distributions. We argue that, compared to fixed vectors, distribution-based representations are more powerful to characterize users' uncertain interests and items' distinct properties. In addition, we propose a Wasserstein-based loss, in which the critical triangular inequality can be satisfied. In experiments, we evaluate DDN and comparative models on standard datasets. It is shown that DDN significantly outperforms state-of-the-art models, demonstrating the advantages of the proposed distribution-based representations and wassertein loss.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Computing methodologies** → *Neural networks.*

KEYWORDS

Sparsity, Recommendation, Distribution

*Indicates Equal Contributions.

ACM Reference Format:
Lei Zheng, Chaozhuo Li, Chun-Ta Lu, Jiawei Zhang, and Philip S. Yu. 2019. Deep Distribution Network: Addressing the Data Sparsity Issue for Top-N Recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3331184.3331330

1 INTRODUCTION

The effectiveness of recommender systems (RS) often relies on how well users' interests or preferences can be understood and user-item interactions can be modeled. However, the data sparsity issue arises when users interacted with a limited number of items, hindering RS from understanding users' intentions. The problem is considered as one of major challenges for RS. Nonetheless, tackling the sparsity issue raises great challenges. Users' interests are diverse, and perceptions of items differ from user to user. This intricate information requires models of high-complexity while training such models needs a large amount of data, which contradicts to the reality of data scarcity.

Recent studies [2, 12, 13] have suggested the importance of learning embeddings, or vectors, for users and items. Although embedding-based models have been proven useful in capturing typical interests of users and general concepts of items, most of existing approaches learn fixed vectors to represent users and items. Arguably, users' behaviors are uncertain, and can be seen as stochastic events sampled from underlying distributions. When a user is modeled with a fixed vector, all actions of the user are considered to be certain and the uncertainty is hardly captured. Moreover, existing well-known Collaborative Filtering (CF) methods, such as matrix factorization [6], mostly use the popular dot-product as a metric, which violates the triangular inequality[4], to calculate user-item similarities. Nevertheless, according to [4, 10], the triangular inequality is a prerequisite for fine-grained setting of users and items.

Probabilistic distributions are classic and fundamental tools for tackling uncertainty and dealing with limited data. As users' actions are uncertain, we can consider them as observed stochastic events governed by underlying distributions of user interests. These

Table 1: Notations

Notation	Description
$\mathcal{U}, \mathcal{I}$	user and item set
$\mathcal{I}_u^+, \mathcal{I}_u^-$	a set of items liked by user u, and all the remaining items without interactions with u
$\mathbf{x}_u, \mathbf{x}_i$	feature vectors for user u and item i
$f_u(:;\Omega^u), f_i(:;\Omega^i)$	two mean networks of user u and item i
$g_u(:;\Pi^u), g_i(:;\Pi^i)$	two covariance networks of user u and item i
$\mathbf{W}_{\text{mean}}^{u,l}, \mathbf{W}_{\text{mean}}^{i,l}$	projection matrices of the l_{th} layer of the mean network of user u and item i
$\mathbf{W}_{\text{cov}}^{u,l}, \mathbf{W}_{\text{cov}}^{i,l}$	projection matrices of the l_{th} layer of the covariance network of user u and item i
$n_{(l)}$	number of neurons of the l_{th} layer

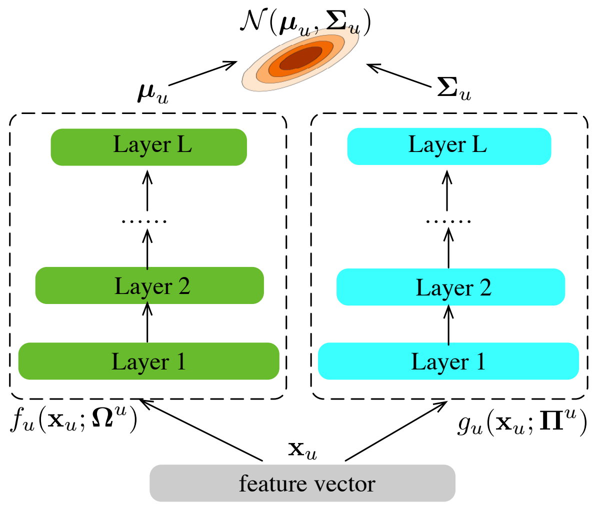
distributions are able to describe how interests of users distribute in the space. As such, in order to power RS with the ability of combating the data sparsity issue with limited data, we propose Deep Distribution Network (DDN) to learn distributions for users and items. Specifically, we associate each user and item with a Gaussian distribution, whose mean and covariance matrix are estimated by deep neural networks, to characterize their interests and properties. Then, instead of calculating the popular dot-product, the Wasserstein distance is utilized to measure the difference between two Gaussian distributions, and the triangular inequality can therefore be satisfied. Finally, a pair-wise loss is proposed to minimize the Wasserstein distance of positive user-item pairs and maximize negative pairs. Our work makes the following contributions:

- **Novelty:** To the best of our knowledge, it is the first work proposing to model users and items by Gaussian distributions via deep architectures for recommendation. We demonstrate that, distributions of users and items can be well modeled to alleviate the data sparsity issue.
- **A Wasserstein Loss:** We propose a Wasserstein loss for recommendation tasks. In the proposed loss, the crucial triangular inequality can be satisfied and therefore, leads to better performances, compared to conventional methods.
- **High Performance:** In the experiments, it is shown that DDN achieves state-of-the-art performances on three benchmark datasets. Specifically, compared to the best performing comparative method, DDN gains **42.4%** and **47.3%** improvements in Hit Ratio@10 and NDCG@10, respectively, averaging on all datasets.

2 PROPOSED MODEL

Let us assume that a user u and an item i are associated with a feature vector $\mathbf{x}_u \in \mathcal{R}^{n_{(0)}\times 1}$ and $\mathbf{x}_i \in \mathcal{R}^{n_{(0)}\times 1}$, respectively (notation $n_{(0)}$ is described in Table 1). A user set and an item set are denoted as $\mathcal{U}$ and $\mathcal{I}$, respectively. For a user $u \in \mathcal{U}$, let $\mathcal{I}_u^+$ denote a set of items liked by user u and $\mathcal{I}_u^-$ denote the remaining items. Important notations are summarized in Table 1.

Instead of deriving vectors of users and items based on their interactions, we aim to learn Gaussian distributions to characterize interests of users and perceptions of items [1]. To do so, as illustrated in Fig. 1, we introduce a mean and a covariance network to learn these two parameters for the users' distribution. And, since users and items are two different types of entities, another two deep models will be built to estimate mean vectors and covariance matrices of items. Please bear in mind that, although these mean



$\mathcal{N}(\boldsymbol{\mu}_u, \boldsymbol{\Sigma}_u)$
$\boldsymbol{\mu}_u$
$\boldsymbol{\Sigma}_u$
Layer L
Layer L
......
......
Layer 2
Layer 2
Layer 1
Layer 1
$f_u(\mathbf{x}_u;\boldsymbol{\Omega}^u)$
$g_u(\mathbf{x}_u;\boldsymbol{\Pi}^u)$
$\mathbf{x}_u$
feature vector

Figure 1: The mean and covariance networks of users. A feature vector $\mathbf{x}_u$ of user u is taken into $f_u(\mathbf{x}_u;\Omega^u)$ and $g_u(\mathbf{x}_u;\Pi^u)$ to learn the mean $\boldsymbol{\mu}_u$ and covariance Σ_u, respectively.

vectors and covariance matrices are also fixed after training, they together describe a probability density function. And, this function describes the sampling probability of each point in a space. This is a key point to distinguish DDN from existing embedding-based methods .

2.1 Mean Networks

To learn a mean vector for user u, we build a mean network to take the user feature $\mathbf{x}_u \in \mathcal{R}^{n_{(0)}\times 1}$ into account, and output a mean vector $\boldsymbol{\mu}_u$ as:

$$\boldsymbol{\mu}_u = \underbrace{\text{elu}\Big(\ldots\text{elu}\big(\mathbf{W}_{\text{mean}}^{u,2}(\text{elu}(}_{L}\mathbf{W}_{\text{mean}}^{u,1}\mathbf{x}_u + \mathbf{b}_{\text{mean}}^{u,1})) + \mathbf{b}_{\text{mean}}^{u,2}\big)\ldots\Big), \quad (1)$$

where $\mathbf{W}_{\text{mean}}^{u,l} \in \mathcal{R}^{n_{(l-1)}\times n_{(l)}}$ and $\mathbf{b}_{\text{mean}}^{u,l} \in \mathcal{R}^{n_{(l)}\times 1}$ are projection matrix and bias vector of the l_{th} layer, respectively; elu is an activation function. We denote the mean network of users as $f_u(:;\Omega^u)$, where $\Omega^u = \{\mathbf{W}_{\text{mean}}^{u,1}, \ldots, \mathbf{W}_{\text{mean}}^{u,L}, \mathbf{b}_{\text{mean}}^{u,1}, \ldots, \mathbf{b}_{\text{mean}}^{u,L}\}$ is a parameter set. Likewise, another mean network, denoted as $f_i(:;\Omega^i)$ parameterized by $\Omega^i = \{\mathbf{W}_{\text{mean}}^{i,1}, \ldots, \mathbf{W}_{\text{mean}}^{i,L}, \mathbf{b}_{\text{mean}}^{i,1}, \ldots, \mathbf{b}_{\text{mean}}^{i,L}\}$, is utilized to derive mean vectors of items.

2.2 Covariance Networks

To learn covariance matrices of users, we establish a L-layer covariance network for estimating the covariance matrix of user i. The diagonal elements of Σ_u is computed as:

$$\boldsymbol{\sigma}_u = \underbrace{\text{elu}\Big(\ldots\text{elu}\big(\mathbf{W}_{\text{cov}}^{u,2}(\text{elu}(}_{L}\mathbf{W}_{\text{cov}}^{u,1}\mathbf{x}_u + \mathbf{b}_{\text{cov}}^{u,1}) + \mathbf{1}) + \mathbf{b}_{\text{cov}}^{u,2}\big) + \mathbf{1}\ldots\Big), \quad (2)$$

where $\mathbf{W}_{\text{cov}}^{u,l} \in \mathcal{R}^{n_{(l-1)}\times n_{(l)}}$ and $\mathbf{b}_{\text{cov}}^{u,l} \in \mathcal{R}^{n_{(l)}\times 1}$ are projection matrix and bias vector of the l_{th} layer, respectively; $\mathbf{1}$ denotes an vector of all ones. Finally, the covariance matrix of user u is given by:

$$\Sigma_u = \text{diag}(\boldsymbol{\sigma}_u) + \mathbf{I}, \quad (3)$$

Table 2: Statistics of Datasets

Dataset	#users	#items	density
MovieLens-1M	6,014	3,706	1.0%
LastFM	1,892	17,632	0.28%
Amazon Video Games	22,996	10,672	0.049%

where $\mathbf{I} \in \mathcal{R}^{n_{(L)} \times n_{(L)}}$ is an identity matrix ensuring Σ_u to be positive semi-definite. The covariance network of users is denoted as $g_u(:;\mathbf{\Pi}^u)$, where $\mathbf{\Pi}^u = \{\mathbf{W}^{u,1}_{\text{cov}}, ..., \mathbf{W}^{u,L}_{\text{cov}}, \mathbf{b}^{u,1}_{\text{cov}}, ..., \mathbf{b}^{u,L}_{\text{cov}}\}$ is a parameter set. Analogously, another covariance network for items is denoted as $g_i(:;\mathbf{\Pi}^i)$, where $\mathbf{\Pi}^i = \{\mathbf{W}^{i,1}_{\text{cov}}, ..., \mathbf{W}^{i,L}_{\text{cov}}, \mathbf{b}^{i,1}_{\text{cov}}, ..., \mathbf{b}^{i,L}_{\text{cov}}\}$ includes all parameters of the network.

Our focus is to model the sparse user-item interaction data with the proposed distribution-based representations, we therefore avoid using additional information, such as user demographics or item textual descriptions, for feature vectors of users and items, even though these information is shown to be helpful for easing the sparsity issue [11]. Instead, $\mathbf{x}_u$ and $\mathbf{x}_i$ are randomly initialized, and then optimized during the training.

2.3 A Wasserstein Loss

Recall that two Gaussian distributions of user u and item i, $\mathcal{N}(\boldsymbol{\mu}_u, \Sigma_u)$ and $\mathcal{N}(\boldsymbol{\mu}_i, \Sigma_i)$, are estimated by mean and covariance networks. Instead of using the dot-product, one can utilize statistical distances to measure the distance between $\mathcal{N}(\boldsymbol{\mu}_u, \Sigma_u)$ and $\mathcal{N}(\boldsymbol{\mu}_i, \Sigma_i)$. In this section we compare two popular distribution distances: Kullback-Leibler (KL) divergence and the p_{th} Wasserstein distance, and propose a Wasserstein based loss for recommendation.

It is easy to verify that the p_{th} Wasserstein distance (W_p) satisfies the triangular inequality, while KL-divergence violates the inequality. As discussed in [4], the satisfaction of the inequality benefits RS for reasoning over intricate user-item relationships, while the violation results in problematic representations of users and items. Moreover, if two distributions are non-overlapping, the Wasserstein distance can still measure the distance between them, while KL-divergence fails and leads to vanishing gradients. Hence, a Wasserstein based loss is proposed as:

$$\begin{aligned}\mathcal{L} = \; & - \sum_{(u,i,i') \in \mathcal{D}} \ln \sigma \{ W_2(\mathcal{N}(\boldsymbol{\mu_u}, \Sigma_{\boldsymbol{u}}), \mathcal{N}(\boldsymbol{\mu}_{i'}, \Sigma_{i'})) \\ & - W_2(\mathcal{N}(\boldsymbol{\mu_u}, \Sigma_{\boldsymbol{u}}), \mathcal{N}(\boldsymbol{\mu_i}, \Sigma_i))\} + \lambda(||\Omega^u||_2^2 + \\ & ||\Omega^i||_2^2 + ||\mathbf{\Pi}^u||_2^2 + ||\mathbf{\Pi}^i||_2^2),\end{aligned} \tag{4}$$

where σ denotes a sigmoid function; the training data $\mathcal{D}$ is created by $\{(u, i, i') | u \in \mathcal{U} \wedge i \in \mathcal{I}_u^+ \wedge i' \in \mathcal{I}_u^-\}$; and λ represents the weight on the regularization terms. Fortunately, the W_2 distance between two Gaussian distributions has an analytical solution as $W_2(\mathcal{N}(\boldsymbol{\mu}_1, \Sigma_1), \mathcal{N}(\boldsymbol{\mu}_2, \Sigma_2)) = ||\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2||_2^2 + Tr(\Sigma_1 + \Sigma_2 - 2 * (\Sigma_1{}^{1/2}\Sigma_2\Sigma_1{}^{1/2})^{1/2})$. Eq. 4 seeks to maximize the Wasserstein distance of a negative pair (u, i') and minimize the distance of a positive pair (u, i). For evaluation, the final recommendation list of items for a user u is given by ranking $W_2(\mathcal{N}(\boldsymbol{\mu_u}, \Sigma_{\boldsymbol{u}}), \mathcal{N}(\boldsymbol{\mu_i}, \Sigma_{\boldsymbol{i}}))$ in an ascending order.

3 EXPERIMENTS

In this section we conduct experiments to anwser the following research questions:

RQ1: Does DDN outperform state-of-the-art methods?
RQ2: Are the distribution-based representations helpful for tackling the data sparsity issue?
RQ3: How does the proposed Wasserstein loss work?
RQ4: Can DDN handle cold-start users in an effective way?

3.1 Experimental Settings

Comparative Models. We compare DDN with five state-of-the-art methods: **ItemKNN** [9], **eALS** [3], **BPR** [8], **NCF** [2] and **CML** [4].
Datasets. We test all methods on three standard datasets: ***MovieLens-1M***, ***LastFM***, and ***Amazon Video Games*** [7]. As in [2], we transform datasets with explicit ratings into implicit data by regarding rating of 5 as positive feedbacks and all others as negative. For each dataset, we select the latest item of each user for testing and the second latest one for validation. All remaining items are for training. The statistics of datasets are shown in Table 2.
Evaluation Protocols. We evaluate all models in two metrics: Hit Ratio@N (HR@N) and NDCG@N. We follow a common strategy as in [2] to avoid heavy computation on evaluating all user-item pairs. For each user i, we randomly sample 999 negative items, and rank them with the single ground-truth item. Based on the rankings of these $1,000$ items, HR@N and NDCG@N can be evaluated.
Paramter Settings. For ItemKNN, we employ the cosine distance to measure item similarities. For eALS and BPR, we search the latent dimensions from $\{8, 16, 32, 64\}$ and $\mathcal{L}_2$ regularization term from $\{0.0001, 0.001, 0.01, 0.1\}$. All hyper-parameters are tuned using the validation set. For DDN, the *Adam* optimizer [5] with the learning rate of 0.001 is adopted.

3.2 Performance Comparison (RQ1 and RQ2)

To anwser RQ1 and RQ2, DDN is compared with five state-of-the-art models on three datasets with different densities. Table 3 shows the performance comparison. Overall, benefiting from the proposed distribution-based representations and Wasserstein loss, DDN beats all comparative methods, and achieves **42.4%** and **47.3%** improvements over the best comparative model in HR@10 and NDCG@10, respectively, averaging on all three datasets. These experiments reveal a number of interesting discoveries: (1) CML yields the second best performances in *MovieLens-1M* and *Amazon Video Games*, demonstrating the importance of the satisfaction of the triangular inequality; (2) Owing to the capability of capturing non-linearities via deep models, NCF defeats other comparative methods in *LastFM*; (3) It is shown that DDN achieves more improvements in a sparser dataset than in a denser one. It is validated that, compared to comparative approaches, DDN can better diminish the negative impacts of the data sparsity issue.

3.3 Effectiveness of the Wasserstein Loss (RQ3)

In order to anwser RQ3, we conduct experiments to compare DDN with DDN-KL, which is a variant of DDN employing the KL-divergence to measure the distances between users and items. Fig. 2 shows the performance comparison between DDN and DDN-KL in *MovieLens-1M*. Overall, when N is vared from 3 to 10, DDN consistently outperforms DDN-KL in HR@N and NDCG@N. Specifically, DDN improves DDN-KL by **21.0%** and **31.0%** in HR@N and NDCG@N, respectively, averaging on N. This experiment shows that, benefiting from the satisfaction of the triangular inequality, the proposed

Table 3: Performance comparison in HR@10 and NDCG@10. The best and second best method are boldfaced and underlined, respectively. ⋆ and ⋆⋆ denote the statistical significance for $p < 0.05$ and $p < 0.01$, respectively, compared to the best baseline.

Dataset	Metric	ItemKNN	eALS	BPR	NCF	CML	DDN	*DDN vs. best*
MovieLens-1M	HR@10	0.038	0.049	0.061	0.081	<u>0.092</u>	**0.128**⋆	39.1%
	NDCG@10	0.021	0.024	0.025	0.039	<u>0.041</u>	**0.058**⋆⋆	41.4%
LastFM	HR@10	0.063	0.101	0.121	<u>0.103</u>	0.101	**0.147**⋆	42.7%
	NDCG@10	0.031	0.035	0.039	<u>0.052</u>	0.050	**0.076**⋆⋆	46.1%
Amazon Video Games	HR@10	0.032	0.041	0.046	0.052	<u>0.055</u>	**0.080**⋆⋆	45.4%
	NDCG@10	0.018	0.019	0.021	0.022	<u>0.034</u>	**0.042**⋆	54.5%

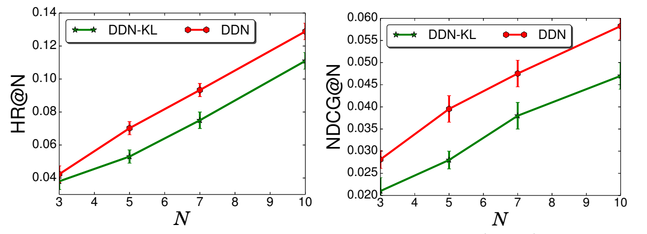


Figure 2: In *MovieLens-1M*, DDN is compared with DDN-KL in terms of HR@N and NDCG@N with N varied from 3 to 10. Errors bars are 1-standard deviation.

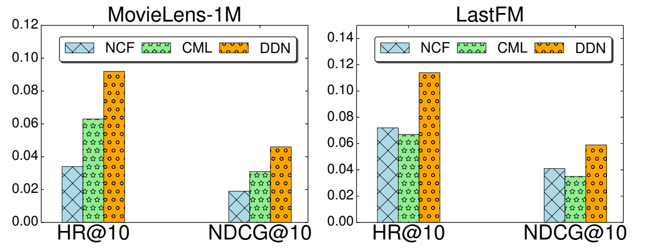


Figure 3: Performance comparison in HR@10 and NDCG@10 under a sparse setting, where each user is associated with only one user-item interaction for training.

Wasserstein loss assists DDN with reasoning over complex user-item relations with limited data.

3.4 Recommending for Cold-start Users (RQ4)

In this section we are curious if DDN can handle cold-start users in an effective way. Therefore, we compare DDN with two strong competitors, NCF and CML, in an extremely sparse setting, where each user is only associated with one item for training, one for validation and one for testing. Fig. 3 shows that, suffering from the *cold-start* problem, the performances of NCF and CML inevitably degrade. However, DDN outperforms NCF and CML in terms of HR@10 and NCDG@10. Specifically, in *MovieLens-1M*, DDN improves CML by **46.0%** and **48.4%**, in HR@10 and NCDG@10, respectively. In *LastFM*, DDN beats NCF by **58.3%** and **43.9%**, in HR@10 and NCDG@10, respectively. Hence, it is demonstrated that, compared with two best performing state-of-the-art baselines, DDN can better handle *cold-start* users.

4 CONCLUSIONS

We present Deep Distribution Network (DDN) to model users and items with Gaussian distributions for Top-N recommendation. Compared to existing approaches learning fixed vectors of users and items, DDN addresses the uncertainty inherent from the data sparsity issue by distribution-based representations. In DDN, each user and item is associated with a Gaussian distribution, whose mean and covariance are estimated by deep neural networks. Experimentally, we show that, compared to fixed vectors, the proposed distribution-based representations can better ease the sparsity issue and handle cold-start users. Additionally, we propose a Wasserstein distance based loss satisfying the triangular inequality, which is crucial for the performances of RS. By comparing DDN with one of its variants, DDN-KL, it is demonstrated that the proposed Wasserstein loss leads to a better performance.

ACKNOWLEDGMENTS

This work is supported in part by NSF through grants IIS-1526499, IIS-1763325, and CNS-1626432, and NSFC 61672313. This work is also partially supported by NSF through grant IIS-1763365.

REFERENCES

[1] Aleksandar Bojchevski and Stephan Günnemann. 2017. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. *arXiv preprint arXiv:1707.03815* (2017).
[2] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017.* 173–182. https://doi.org/10.1145/3038912.3052569
[3] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval.* ACM, 549–558.
[4] Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge Belongie, and Deborah Estrin. 2017. Collaborative metric learning. In *Proceedings of the 26th International Conference on World Wide Web.* International World Wide Web Conferences Steering Committee, 193–201.
[5] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[6] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.
[7] Himabindu Lakkaraju, Julian J McAuley, and Jure Leskovec. 2013. What's in a Name? Understanding the Interplay between Titles, Content, and Communities in Social Media. *ICWSM* 1, 2 (2013), 3.
[8] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence.* AUAI Press, 452–461.
[9] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web.* ACM, 285–295.
[10] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Latent relational metric learning via memory-based attention for collaborative ranking. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web.* International World Wide Web Conferences Steering Committee, 729–739.
[11] Lei Zheng, Bokai Cao, Vahid Noroozi, S Yu Philip, and Nianzu Ma. 2017. Hierarchical collaborative embedding for context-aware recommendations. In *2017 IEEE International Conference on Big Data (Big Data).* IEEE, 867–876.
[12] Lei Zheng, Chun-Ta Lu, Fei Jiang, Jiawei Zhang, and Philip S Yu. 2018. Spectral collaborative filtering. In *Proceedings of the 12th ACM Conference on Recommender Systems.* ACM, 311–319.
[13] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining.* ACM, 425–434.