Cross-Domain Recommendation via Preference Propagation GraphNet

Cheng Zhao[1], Chenliang Li[2*], Cong Fu[3]
1. State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan, China, zhaocheng_whuer@whu.edu.cn
2. School of Cyber Science and Engineering, Wuhan University, Wuhan, China, cllee@whu.edu.cn
3. State Key Lab of CAD&CG, Zhejiang University, Hangzhou, China, fc731097343@gmail.com

ABSTRACT

Recommendation can be framed as a graph link prediction task naturally. The user-item interaction graph built within a single domain often suffers from high sparsity. Thus, there has been a surge of approaches to alleviate the sparsity issue via cross-domain mutual augmentation. The SOTA cross-domain recommendation algorithms all try to bridge the gap via knowledge transfer in the latent space. We find there are mainly three problems in their formulations: 1) their knowledge transfer is unaware of the cross-domain graph structure. 2) their framework cannot capture high-order information propagation on the graph. 3) their cross-domain transfer formulations are generally more complicated to be optimized than the unified methods. In this paper, we propose the Preference Propagation GraphNet (PPGN) to address the above problems. Specifically, we construct a Cross-Domain Preference Matrix (CDPM) to model the interactions of different domains as a whole. Through the propagation layer of PPGN, we try to capture how user preferences propagate in the graph. Consequently, a joint objective for different domains is defined, and we simplify the cross-domain recommendation into a unified multi-task model. Extensive experiments on two pairs of real-world datasets show PPGN outperforms the SOTA algorithms significantly.

KEYWORDS

Recommendation Systems; Graph Neural Network; Deep Learning

ACM Reference Format:

Cheng Zhao, Chenliang Li, Cong Fu. 2019. Cross-Domain Recommendation via Preference Propagation GraphNet. In *Proceedings of 28th ACM International Conference on Information and Knowledge Management (CIKM'19).* ACM, New York, NY, USA, Article 4, 4 pages. https://doi.org/10.1145/3357384.3358166

*Chenliang Li is the corresponding author.

CIKM'19, November 3–7, 2019, Beijing, China

ACM ISBN 978-1-4503-6976-3/19/11…$15.00
https://doi.org/10.1145/3357384.3358166

1 INTRODUCTION

Recommendation system is an important area and has a wide range of applications in data mining and machine learning. The interactions between users and items can be formulated as a graph in nature, while recommending new items for users can be regarded as the link prediction task. The same group of users may be involved in different domains, e.g., book recommendation and movie recommendation. It is widely admitted that the graph generated from one single domain may suffer from severe sparsity and cold start problem. Augmenting the system with multiple domains, i.e., superimposing cross-domain user-item interaction graphs has attracted wide research interests [8].

Several Cross-Domain Recommendation (CDR) approaches have been proposed and achieved promising performance, such as NeuMF+ [6], CoNet [6], and NATR [3]. They all model the inter-domain interactions through latent-space-level knowledge transfer. NeuMF+ achieves CDR by simply sharing user embeddings between two NeuMF [5]. CoNet conducts knowledge transfer on latent embedding space based on a cross-stitch network. NATR also leverages latent knowledge transfer, while they focus on protecting user privacy via not sharing user information.

In our study, we find there are several problems with such formulations. Firstly, they process each domain separately and rely on the embedding-level information sharing or latent feature mapping to transfer the knowledge (e.g., the CoNet uses two basic networks to model two different domains). The main drawback of such frameworks is that the knowledge transfer models the interactions between different domains implicitly and is unaware of the structure information bridging the two domains. Secondly, we believe that the items of different domains may share similar properties. As illustrated in Fig. 1, the users who like the movie *Ironman* may all be interested in *DC* or *Marvel* comic books. There are implicit properties shared among *Ironman*, *DC*, and *Marvel* comics. Through different hops of transitions, we can make connections on such properties. Consequently, we can propagate the users' preference on the high-order level graph. However, the formulation of existing works can not capture such high-order relationships. Thirdly, they have to implement their approaches through transfer learning, which is complicated to optimize.

To address the above problems, we propose a novel graph-based method, namely Preference Prorogation GraphNet (PPGN). We define the Cross-Domain Preference Matrix (CDPM) to maintain the cross-domain interactions. Taking CDPM and randomly initialized embeddings as input, we optimize the network parameters and the embeddings by jointly minimizing the recommendation error from different domains. The propagation layer of the network is used to

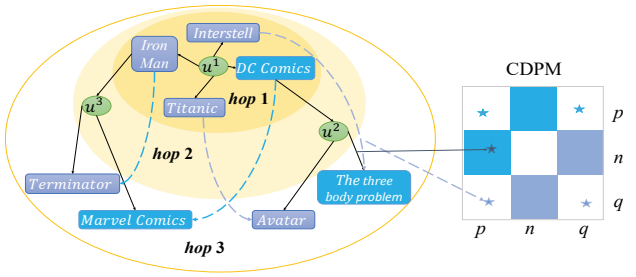

Figure 1: An illustration of the preference propagation via the joint interaction graph. Items in different colors are from different domains. The graph is further reconstructed as the sparse Cross-Domain Preference Matrix (CDPM) on the right, which can be processed by the model immediately. Solid lines denote the known user behaviors, while dotted lines denote the potential recommendation. Through multi-hop propagation, our model can capture the transitions of the user preference and make better predictions.

capture the high-order information transition over the joint graph. Besides, the resulting unified model is easy to optimize. Our contributions can be summarized as follows: i) proposing a novel deep graph-based network to facilitate cross-domain recommendation and address the limitations of previous approaches; ii) outperforming the existing works significantly on real-world datasets.

2 METHODOLOGY

We propose the Preference Propagation GraphNet (PPGN) to address the limitations of existing methods, i.e., capture the multi-hop user preference propagation (Fig. 1), explicitly model the cross-domain interactions and preserve the structure information. Figure 2 illustrates the architecture overview of PPGN, which mainly contains two parts: 1) the graph convolution and propagation module; 2) the knowledge integration and prediction module.

Definitions and Notations. We study the overlapping users' cross-domain recommendation on two domains $\mathcal{D}_a, \mathcal{D}_b$, which can be naturally generalized to multi-domain settings. Let n, p, q denote the number of users, items in $\mathcal{D}_a$ and items in $\mathcal{D}_b$ respectively.

2.1 Graph Convolution and Propagation

The key to capture the high-order user-item relationships across domains on the superimposed graph are to model the knowledge flowing along the observed links (user-item interactions) and maintain the structure information of the graph itself. Following the Graph Convolution Network (GCN) [7], we adopt the graph convolution and propagation layer in this module.

In GCN, the graph convolution and propagation layer takes the graph adjacent matrix and node embedding as inputs. The embeddings are node-wise randomly initialized. To jointly and directly model the cross-domain interactions, we define the Cross-Domain Preference Matrix (CDPM) as follows:

$$\mathbf{A} = \begin{cases} 1, & \text{if interactions (user } u\text{, item } i_a \text{ or } i_b\text{) is observed;} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{(p+n+q)\times(p+n+q)}$. Different from common collaborative filtering matrix, CDPM merges the interactions between users and items in both domains, which can propagate the information across different domains directly.

To avoid gradient vanishing or explosion in the training process, we add a self-loop diagonal matrix $\mathbf{I} \in \mathbb{R}^{(p+n+q)\times(p+n+q)}$ to $\mathbf{A}$, and carry out a normalization process using a degree matrix $\mathbf{D} \in \mathbb{R}^{(p+n+q)\times(p+n+q)}$, so we get $\hat{\mathbf{A}}$, $\hat{\mathbf{A}} = \mathbf{D}^{-1}(\mathbf{A} + \mathbf{I})$. Next, following [5, 10], we map the ID of u, i_a and i_b to the embeddings $e^u \in \mathbb{R}^{d_0}$, $e^{i_a} \in \mathbb{R}^{d_0}$ and $e^{i_b} \in \mathbb{R}^{d_0}$, where d_0 is the initial embedding size. We establish the following embedding table to represent the initial latent factors of both users and items in $\mathcal{D}_a, \mathcal{D}_b$.

$$\mathbf{E_0} = \left[\underbrace{e_1^{i_a}, \cdots, e_p^{i_a}}_{\mathcal{D}_a \text{ items embeddings}}, \overbrace{e_1^{u}, \cdots, e_n^{u}}^{\text{users embeddings}}, \underbrace{e_1^{i_b}, \cdots, e_q^{i_b}}_{\mathcal{D}_b \text{ items embeddings}} \right]^T, \tag{2}$$

Then, we feed $\mathbf{E_0}$ and $\hat{\mathbf{A}}$ through multiple graph convolution and propagation layers, with size-decreasing convolution kernels. We apply at least three layers of the convolution and propagation to ensure the breadth of cross-domain preference propagation. The decreasing of kernel sizes is used because we do not care about nodes that are too far from each other in high-level scope. The propagation process can be expressed as: $\mathbf{E_l} = \sigma(\hat{\mathbf{A}}\mathbf{E_{l-1}}W_l + b_l)$, where $l \geq 3$ indexes the layers, $W_l \in \mathbb{R}^{d_{l-1}\times d_l}$, $b_l \in \mathbb{R}^{d_l}$ are the trainable parameters. d_l, d_{l-1} are the size of current layer and previous layer respectively. σ is the Relu activation function.

Through l_n times of propagation, we can get multiple embedding matrices ranging from $\mathbf{E_0} \in \mathbb{R}^{(p+n+q)\times d_0}$ to $\mathbf{E_n} \in \mathbb{R}^{(p+n+q)\times d_n}$. Then we get the global latent embeddings $\mathbf{E} \in \mathbb{R}^{(p+n+q)\times\sum_{d_0}^{d_n}}$ by concatenation, which mixes interactions of user-item at different scopes, and makes full use of the process of preference propagation. Then we rearrange $\mathbf{E}$ into three parts of embedding matrices: $\mathbf{E_a^i} \in \mathbb{R}^{p\times\sum_{d_0}^{d_n}}$, $\mathbf{E^u} \in \mathbb{R}^{n\times\sum_{d_0}^{d_n}}$ and $\mathbf{E_b^i} \in \mathbb{R}^{q\times\sum_{d_0}^{d_n}}$, respectively representing $\mathcal{D}_a$ item embeddings, user embeddings and $\mathcal{D}_b$ item embeddings, which will be used in the next module:

$$\mathbf{E} = \left[\mathbf{E_0}, \mathbf{E_1}, \cdots, \mathbf{E_{l_n}}\right] = \left[\mathbf{E_a^i}, \mathbf{E^u}, \mathbf{E_b^i}\right]^T \tag{3}$$

2.2 Knowledge Integration and Prediction

After getting the latent embeddings of users (i.e., e^u) and items (i.e., e^{i_a} and e^{i_b}), we feed tuples of (e^{i_a}, e_u, e^{i_b}) to the multi-layer feedforward networks. In details, we combine e^{i_a}, e_u, and e_u,e^{i_b} as the inputs of two multi-layer perceptrons to get recommendation predictions on training samples, namely $\hat{r}_{ua}$,$\hat{r}_{ub}$ between users and items in both domains.

$$\begin{aligned} & e_{ua}^0 = [e^{i_a}, e^u], e_{ub}^0 = [e^{i_b}, e^u] \\ & e_{ua}^1 = \sigma(W_{ua}^1 e_{ua}^0 + b_{ua}^1), e_{ub}^1 = \sigma(W_{ub}^1 e_{ub}^0 + b_{ub}^1) \\ & \cdots \\ & e_{ua}^L = \sigma(W_{ua}^L e_{ua}^{L-1} + b_{ua}^{L-1}), e_{ub}^L = \sigma(W_{ub}^L e_{ub}^{L-1} + b_{ub}^{L-1}) \\ & \hat{r}_{ua} = \phi^a(e_{ua}^L), \hat{r}_{ub} = \phi^b(e_{ub}^L) \end{aligned} \tag{4}$$

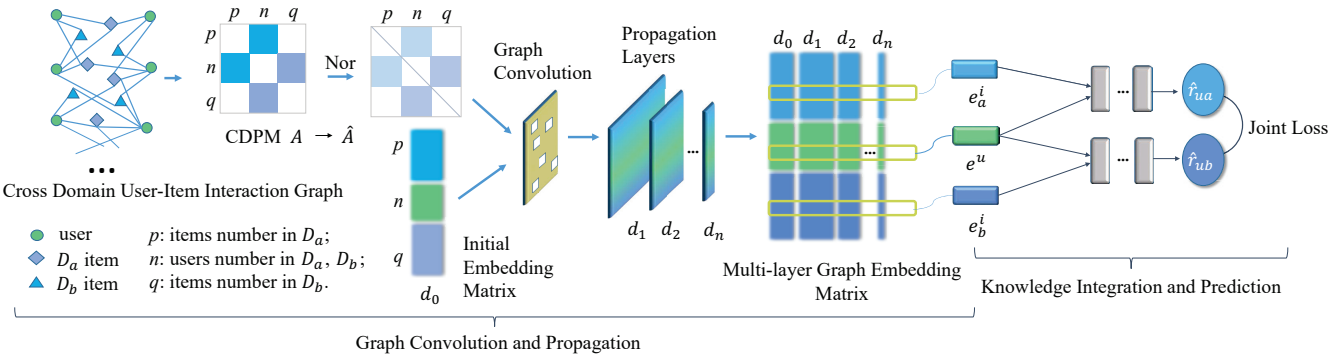

Figure 2: The network architecture of PPGN.

where $W_{ua}, b_{ua}/W_{ua}, b_{ub}$ are the trainable transformation matrices and biases. L is the total number of layers. ϕ^a, ϕ^b are two one-layer perceptrons to map e^L_{ua}, e^L_{ub} to two scalars $\hat{r}_{ua}, \hat{r}_{ub}$.

The goal of PPGN is to improve the prediction performance on both domains via jointly learning. Naturally, the loss function of PPGN $\mathcal{L}$ is designed as a joint cross-entropy loss from recommendation prediction of both domains, namely $\mathcal{L}_{ua}$ and $\mathcal{L}_{ub}$, and with a regularization term $\mathcal{L}_{reg}$:

$$\begin{aligned}\mathcal{L} &= \mathcal{L}_{ua} + \mathcal{L}_{ub} + \mathcal{L}_{reg} \\ &= -\sum_{(i_a,u,i_b)\in T} r_{ua}\log\hat{r}_{ua} + (1-r_{ua})\log(1-\hat{r}_{ua}) \\ &+ r_{ub}\log\hat{r}_{ub} + (1-r_{ub})\log(1-\hat{r}_{ub}) + \lambda\sum|\Theta|\end{aligned} \tag{5}$$

where T denotes the training dataset including positive and negative samples, r_{ua} and r_{ub} denote the corresponding labels, λ denotes the regularization coefficient, and Θ denotes all the trainable parameters. We use *Adam* as the optimizer to update the parameters.

2.3 Training Strategies

Considering the size of $\hat{\mathbf{A}}$ is generally huge, always up to hundreds of thousands, it's hard to conduct matrix multiplication between $\hat{\mathbf{A}}$ and $\mathbf{E_{l-1}}$ in the graph convolution layer in one go. For the sake of scalability, we propose to split $\hat{\mathbf{A}}$ into rows to get multiple sub-matrices $\hat{\mathbf{A}}_\mathbf{i}$ and perform multiplication operation with $\mathbf{E_{l-1}}$ respectively, then concatenate the results back to one matrix:

$$\begin{aligned}\hat{\mathbf{A}} &= \left[\hat{\mathbf{A}}_\mathbf{0}, \hat{\mathbf{A}}_\mathbf{1}, \cdots, \hat{\mathbf{A}}_\mathbf{sn}\right] \\ \hat{\mathbf{A}}\mathbf{E_{l-1}} &= \left[\hat{\mathbf{A}}_\mathbf{0}\mathbf{E_{l-1}}, \cdots, \hat{\mathbf{A}}_\mathbf{sn}\mathbf{E_{l-1}}\right]\end{aligned}$$

Our PPGN requires data inputs in forms of (i_a, u, i_b), where i_a and i_b can be positive or negative samples, and the ratio between them is $1:\eta(\eta>1)$. To solve this sample imbalance problem, we apply a weighting strategy to the loss function as follow,

$$\begin{aligned}\mathcal{L}' = &-\sum_{(i_a,u,i_b)\in T} \alpha\left(r_{ua}\log\hat{r}_{ua} + (1-r_{ua})\log(1-\hat{r}_{ua})\right) \\ &+ \beta\left(r_{ub}\log\hat{r}_{ub} + (1-r_{ub})\log(1-\hat{r}_{ub})\right) + \lambda\sum|\Theta|\end{aligned} \tag{6}$$

Table 1: Statistics of the two pairs of datasets

Datasets	# users	# items	# ratings	density
Books	37,388	269,301	1,254,288	0.012%
Movies and TV	37,388	49,273	792,319	0.043%
CDs and Vinyl	5,331	55,848	376,347	0.126%
Digital Music	5,331	3,563	63,303	0.333%

Table 2: Performance comparison on two pairs of datasets. The best results are highlighted in boldface. And if the result of PPGN-IP is better than other baselines, it'll be underlined.

Metrics	Dataset	BPRMF	NeuMF	NeuMF+	CoNet	SCoNet	PPGN-IP	PPGN
HR@10	Books	.3654	.4300	.4291	.5223	.5141	.4594	**.5770**
	Movies and TV	.4538	.5665	.5605	.6460	.6465	.5689	**.6909**
	CDs and Vinyl	.5532	.6421	.6655	.7539	.7547	.7668	**.7839**
	Digital Music	.4742	.5322	.5991	.7179	.7205	.7492	**.7874**
MRR@10	Books	.1543	.2241	.2249	.3273	.3261	.1835	**.3280**
	Movies and TV	.2034	.2775	.2742	.3651	.3829	.2498	**.3869**
	CDs and Vinyl	.2742	.3092	.3593	.4735	.4875	.4192	**.5012**
	Digital Music	.1431	.1549	.2472	.3855	.3878	.4112	**.4388**
NDCG@10	Books	.2365	.2725	.2724	.3396	.3370	.2470	**.3574**
	Movies and TV	.2654	.3445	.3416	.4060	.4210	.3164	**.4249**
	CDs and Vinyl	.3532	.3933	.4303	.5227	.5291	.5020	**.5697**
	Digital Music	.2045	.2432	.3297	.4436	.4603	.4911	**.5147**

$$\alpha = \begin{cases}\eta, & \text{if } r_{ua}=1; \\ 1, & \text{if } r_{ua}=0.\end{cases} \quad \beta = \begin{cases}\eta, & \text{if } r_{ub}=1; \\ 1, & \text{if } r_{ub}=0.\end{cases} \tag{7}$$

where α and β are the weight values determined by the labels of input set, which speeds up the training process.

3 EXPERIMENTS

Datasets. In the experiments, we evaluate our model on two pairs of real-world cross-domain datasets from Amazon-5cores[1] [4] (each user or item has at least five ratings), including *Books & Movies and TV*; *CDs and Vinyl & Digital Music*. We extract the overlapping users in both domains, and show the detailed statistics in Table 1.

Baselines. To demonstrate the effectiveness, we compare our proposed **PPGN** with the following methods, (a) **BPRMF** [10], a classical single-domain model using matrix factorization with BPR loss;

[1] http://jmcauley.ucsd.edu/data/amazon/

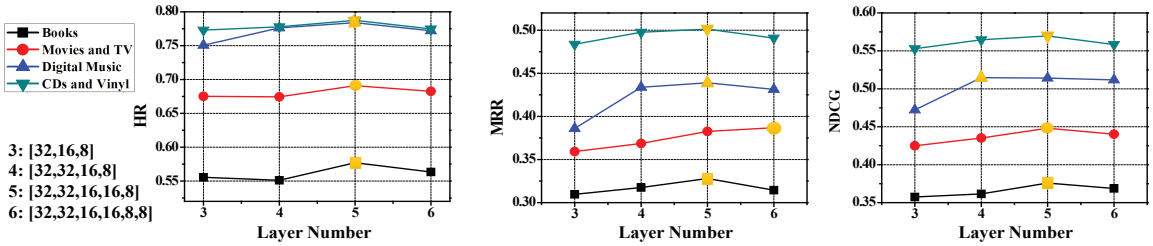

Figure 3: Impact of GCN layer number on two pairs of datasets. The detail layer units of each structure are list on the left.

(b) **NeuMF** [5], a single-domain deep model using neural collaborative network; (c) **NeuMF+**, a cross-domain deep model modified from NeuMF by conducting multi-task training and sharing user embeddings; (d) **CoNet** [6], a cross-domain deep model based on a cross-unit neural network; (e) **SCoNet** [6], a modified version of CoNet with sparsity-induced regularization. Due to **NATR** [3] focusing on protecting the user privacy by not sharing user information, we choose not to evaluate it. We also compare a variant of our PPGN by replacing the multilayer perceptrons with embedding inner product, named **PPGN-IP**.

Training Details. The negative/positive sample ratio η is set as 4. The initial embedding size is 8. The MLP layers in all deep models are $[64, 32, 16, 8]$, and in PPGN we try multiple layer numbers in graph convolutional network ranging from 3 to 6. We also set a message dropout rate of 0.2 in all layers with the regularization coefficient set as 0.1. And the learning rate is 0.001.

Evaluation. We adopt the widely used leave-one-out method to perform the evaluation. The sampled negative items number is set as 99. Following [5], we adopt hit ratio ($HR@k$), mean reciprocal rank ($MRR@k$) and normalized discounted cumulative gain ($NDCG@k$) to evaluate the performances of all the models, where k is 10.

Results and analysis. Table 3 shows the overall performances, from which we can have the following observations:(1) PPGN outperforms all the baselines significantly, demonstrating it can generate high-quality recommendations via precisely capturing the propagation of the user preference; (2) PPGN-IP achieves relatively good results, especially in terms of HR and NDCG. This is because, via preference propagation and structure information preserving, the learned embeddings already contain rich information to reconstruct the users' interests. However, no doubt the non-linear mapping does a better job (PPGN outperforms PPGN-IP).

Figure 3 shows the impact of the number of graph convolution and propagation layers. The marks with yellow color indicate the best models, mostly 5 layers, which shows that the expressibility increases with the layer number till a summit.

4 RELATED WORK

Existing researches for CDR mainly consist of methods as below. For non-neural approaches, EMCDR [9] defines a mapping method to transfer the user factors by matrix factorization from source domain to target domain. For deep models, DCDCSR framework [12] is designed for generating a mixed rating matrix to solve cross-domain and cross-system problems. CoNet and SCoNet [6] is proposed to train a deep cross-stitch network for both domains simultaneously. NATR [3] shares only the item information for user privacy. Meanwhile, NGCF [11] utilizes the graph neural network to enhance collaborative filtering in single-domain recommendation. Some aspect-based recommendation researches [1, 2] also inspire us to investigate for more possibilities on CDR in the future.

5 CONCLUSION

We propose a novel deep graph-based approach to model high order user-item relationships on the interaction graph for the cross-domain recommendation system, namely **PPGN**. The strengths of the proposed method include simplicity, effectiveness, and high-expressibility. Extensive experiments have verified our motivation for modeling user preference propagation on graph structures and shown PPGN outperforms the SOTA algorithms significantly.

ACKNOWLEDGMENTS

This research was supported by National Natural Science Foundation of China (No. 61872278). Chenliang Li is the corresponding author.

REFERENCES

[1] Zhiyong Cheng, Ying Ding, Xiangnan He, Lei Zhu, Xuemeng Song, and Mohan S. Kankanhalli. 2018. Aˆ3NCF: An Adaptive Aspect Attention Model for Rating Prediction. In *IJCAI*. ijcai.org, 3748–3754.
[2] Zhiyong Cheng, Ying Ding, Lei Zhu, and Mohan S. Kankanhalli. 2018. Aspect-Aware Latent Factor Model: Rating Prediction with Ratings and Reviews. In *WWW*. ACM, 639–648.
[3] Chen Gao, Xiangning Chen, Fuli Feng, Kai Zhao, Xiangnan He, Yong Li, and Depeng Jin. 2019. Cross-domain Recommendation Without Sharing User-relevant Data. In *WWW*. 491–502.
[4] Ruining He and Julian McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *WWW*. 507–517.
[5] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *WWW*. 173–182.
[6] Guangneng Hu, Yu Zhang, and Qiang Yang. 2018. CoNet: Collaborative Cross Networks for Cross-Domain Recommendation. In *CIKM*. 667–676.
[7] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR (Poster)*.
[8] Bin Li, Xingquan Zhu, Ruijiang Li, Chengqi Zhang, Xiangyang Xue, and Xindong Wu. 2011. Cross-Domain Collaborative Filtering over Time. In *IJCAI*. 2293–2298.
[9] Tong Man, Huawei Shen, Xiaolong Jin, and Xueqi Cheng. 2017. Cross-Domain Recommendation: An Embedding and Mapping Approach. In *IJCAI*. 2464–2470.
[10] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*. 452–461.
[11] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. SIGIR, 165–174.
[12] Feng Zhu, Yan Wang, Chaochao Chen, Guanfeng Liu, Mehmet A. Orgun, and Jia Wu. 2018. A Deep Framework for Cross-Domain and Cross-System Recommendations. In *IJCAI*. 3711–3717.