Learning personalized itemset mapping for cross-domain recommendation

Zhang, Yinan; Liu, Yong; Han, Peng; Miao, Chunyan; Cui, Lizhen; Li, Baoli; Tang, Haihong

2020

Zhang, Y., Liu, Y., Han, P., Miao, C., Cui, L., Li, B. & Tang, H. (2020). Learning personalized itemset mapping for cross-domain recommendation. 2020 International Joint Conference on Artificial Intelligence (IJCAI' 20), 2561-2567. https://dx.doi.org/10.24963/ijcai.2020/355

https://hdl.handle.net/10356/150977

https://doi.org/10.24963/ijcai.2020/355

Learning Personalized Itemset Mapping for Cross-Domain Recommendation

Yinan Zhang[1,2*] , **Yong Liu**[1,3*] , **Peng Han**[4,6] , **Chunyan Miao**[2†] , **Lizhen Cui**[5] ,
Baoli Li[6] and **Haihong Tang**[6]
[1]Alibaba-NTU Singapore Joint Research Institute
[2]School of Computer Science and Engineering, Nanyang Technological University
[3]Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly (LILY)
[4]King Abdullah University of Science and Technology
[5]School of Software & Joint SDU-NTU Centre for Artificial Intelligence Research (C-FAIR), Shandong University
[6]Alibaba Group

Abstract

Cross-domain recommendation methods usually transfer knowledge across different domains implicitly, by sharing model parameters or learning parameter mappings in the latent space. Differing from previous studies, this work focuses on learning the explicit mapping between a user's behaviors (*i.e.,* interaction itemsets) in different domains during the same temporal period. In this paper, we propose a novel deep cross-domain recommendation model, called Cycle Generation Networks (CGN). Specifically, CGN employs two generators to construct the dual-direction personalized itemset mapping between a user's behaviors in two different domains over time. The generators are learned by optimizing the distance between the generated itemset and the real interacted itemset, as well as the cycle consistency loss defined based on the dual-direction generation procedure. We have performed extensive experiments on real datasets to demonstrate the effectiveness of the proposed model, comparing with existing single-domain and cross-domain recommendation methods.

1 Introduction

Recommendation system has recently become a widespread research topic. Existing recommendation methods are usually developed based on collaborative filtering (CF) techniques, *e.g.,* matrix factorization [Liu *et al.*, 2017; Liu *et al.*, 2018]. The quick development of deep learning (DL) also motivates the emergence of various DL-based recommendation methods [Wang *et al.*, 2017; Wang *et al.*, 2019; Wu *et al.*, 2019; Lian *et al.*, 2020a; Lian *et al.*, 2020b]. These DL-based methods can deal with more complex user-item interaction patterns in an end-to-end manner. However, both CF- and DL-based methods usually treat the recommendation problem as a supervised machine learning task, which needs a lot of historical data. As a user may only interact with a small fraction of items, traditional recommendation methods suffer severely from the data-sparsity problem.

To solve this challenge, various cross-domain recommendation methods have been proposed, and most of them are based on transfer learning [Pan, 2016]. The main objective is to transfer knowledge from the source domain with abundant interaction data to the target domain with limited interaction data. The knowledge learned in the source domain can be used as priors to define a regularization loss to improve the recommendation accuracy in the target domain. Moreover, there also exist some application scenarios where all domains suffer from the data sparsity issue. In this kind of scenarios, the recommendation models for different domains are collectively trained, by learning domain-specific factors and domain-sharing factors simultaneously. In those methods mentioned above, the knowledge is implicitly transferred across domains by sharing model parameters [Singh and Gordon, 2008; Hu *et al.*, 2018; Huang *et al.*, 2019] or learning parameter mappings in latent space [Jiang *et al.*, 2016; Man *et al.*, 2017; Li and Tuzhilin, 2020].

In practice, users' interests and states may vary over time. How to quickly capture these changes and recommend most suitable items is fundamentally essential for a recommender system. Moreover, as users usually interact with items in multiple domains, it is evident that users may interact with some domains first when they get a new interest or transfer to a new state. Taking star tracers as an example, many of them may change their idols very frequently. They may become interested in a new idol after watching a movie or TV show. To better understand and support the idol, they would like to purchase relevant magazines or products. If an explicit mapping between a user's behaviors in the movie domain and the magazine domain at the same temporal period is constructed, we can promptly update the magazine recommendation based on the user's recent watching behaviors. In this paper, we study the cross-domain recommendation problem, where we aim to exploit a user's behavior data in one domain to generate her item recommendation for the same temporal period in another domain. Figure 1 shows an example of the cross-domain rec-

*These two authors contributed equally to this work.
†Corresponding author: ascymiao@ntu.edu.sg

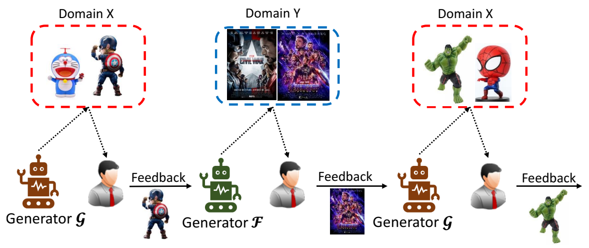


Figure 1: An example of the cross-domain recommendation task studied in this paper. The user clicks the toy "Captain America", which indicates he may have interests in relevant movies. Thus, the recommender agent recommends the movies "Civil War" and "Avengers: Endgame" to the user. After he has watched "Avengers: Endgame", the agent recommends toys – "Hulk" and "Spiderman", who appear in the movie.

ommendation problem studied in this paper.

To solve this recommendation problem, we propose a novel cross-domain recommendation model, *i.e.,* Cycle Generation Networks (CGN), which utilizes two generators to learn a user's personalized mapping between her interaction itemsets in different domains at the same temporal period. For each generator, it uses the set of user's interaction items in one domain as input to generate her item recommendation in the other domain. CGN enables dual-direction mapping across domains. The generators are learned by minimizing the distance between the generated itemset and the relevant itemset in each domain, as well as the cycle consistency loss defined based on the dual-direction mapping process. In addition, we also perform extensive experiments on four real-world datasets to demonstrate the effectiveness of CGN. CGN usually outperforms baseline methods in terms of most evaluation metrics. These experimental results indicate that it is possible to learn a direct mapping between a user's behaviors (*i.e.,* interaction itemsets) across different domains over time.

2 Related Work

One group of cross-domain recommendation systems are based on CF methods [Pan, 2016]. For example, the collective matrix factorization [Singh and Gordon, 2008] transfers knowledge across different domains by jointly learning latent representations from multiple matrices that describe interactions in multiple domains. In [Loni *et al.*, 2014], users' interactions with a specific type of items are treated as a particular domain, and factorization machine is employed to transfer interaction information in auxiliary domain to generate recommendation in a target domain. In [Li and Lin, 2014], a matching procedure is applied to identify the user/item correspondences between different domains, and a transfer learning approach is proposed to improve the recommendation accuracy in the target domain. In [Liu *et al.*, 2015], a hyper-structure transfer method is proposed to capture the non-linear correlation of knowledge in different domains.

Another line of cross-domain recommendation research focuses on developing DL-based recommendation methods. For example, [Man *et al.*, 2017] proposes an embedding and mapping framework to capture the non-linear mapping across different domains by a multi-layer perception. [Zhu *et al.*, 2018] develops a deep learning framework to integrate matrix factorization models and fully connected deep neural networks for cross-domain and cross-system recommendations. [Hu *et al.*, 2018] proposes a deep transfer learning method, called collaborative cross networks (CoNet), which assumes the hidden layers of base networks designed for different domains are connected by cross mapping. The dual knowledge transfer across domains is enabled by the cross connections between base networks. [Fu *et al.*, 2019] proposes a deep fusion model which uses a multi-layer perceptron to transfer users' latent factors in different domains to solve the cold-start problem. Moreover, in [Kanagawa *et al.*, 2019], the recommendation problem is formulated as an extreme multi-class classification problem. The recommendation problem is transferred to be a domain adoption task, and a domain separation network is developed to solve the recommendation problem. In [Hu *et al.*, 2019], the TMH (*i.e.,* transfer meeting hybrid) model employs a memory network to extract useful information from unstructured text, and uses a transfer network to transfer knowledge from the source domain. In [Yuan *et al.*, 2019], a deep domain adaption model is proposed to extract and transfer patterns from rating matrices in different domains, without considering auxiliary information.

3 Cycle Generation Networks

3.1 Problem Formulation

For simplicity, we consider the cross-domain recommendation between two item domains X and Y (*e.g.,* Books and Movies) in this work. We assume that all domains share a common set of users $\mathcal{U}$, and denote the set of items in these domains by $\mathcal{X}$ and $\mathcal{Y}$, respectively. For each user u, we denote the list of her sequential interaction items in domain X by $\mathcal{L}_u^X = \{i_1, i_2, \cdots, i_t, \cdots, i_{|\mathcal{L}_u^X|}\}$, where i_t denote the interaction item at timestamp t. Similarly, we use $\mathcal{L}_u^Y$ to denote her interaction item sequence in domain Y. In addition, we denote the user embedding in the two domains by $\boldsymbol{u}_X \in \mathbb{R}^{1\times d}$ and $\boldsymbol{u}_Y \in \mathbb{R}^{1\times d}$, and use $\boldsymbol{x}_i \in \mathbb{R}^{1\times d}$ and $\boldsymbol{y}_j \in \mathbb{R}^{1\times d}$ to denote the embedding of items $i \in \mathcal{X}$ and $j \in \mathcal{Y}$ respectively, where d is the dimensionality of the latent space. The cross-domain recommendation problem studied in this work can be defined as follows: *for each user, given a set of her latest interaction items in a domain, we generate a set of items she may have interests in the other domain.*

To solve this cross-domain problem, we propose the CGN model to learn a personalized mapping between a user's interaction itemsets in different domains. Figure 2 shows the overall structure of the proposed CGN model. As shown in Figure 2, CGN employs two generator networks $\mathcal{G}$ and $\mathcal{F}$ to enable the dual-direction mappings between different domains over time. The generator $\mathcal{G}$ constructs the mapping from a user's interaction itemset $\mathcal{X}_u$ in domain X to her interaction itemset $\mathcal{Y}_u$ in domain Y, and similarly $\mathcal{F}$ builds the mapping from $\mathcal{Y}_u$ to $\mathcal{X}_u$. Note that the user interacts with $\mathcal{X}_u$ and $\mathcal{Y}_u$ during the same temporal period.

3.2 Loss Function

To enable personalized itemset mapping, we also consider the user's embeddings in different domains as inputs to the

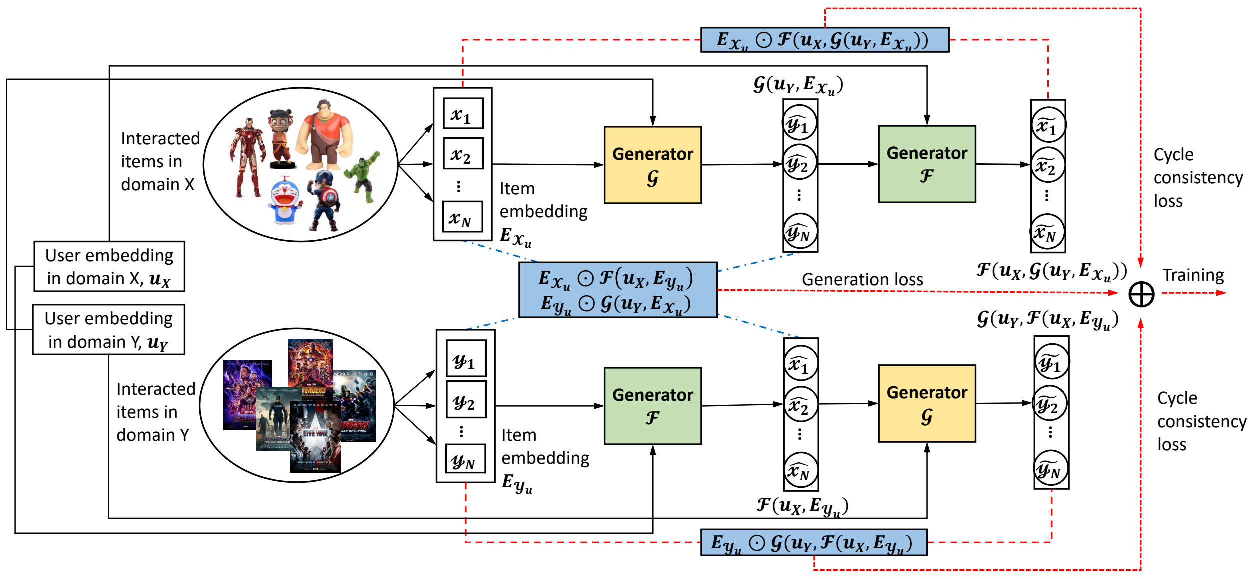

$E_{\mathcal{X}_u} \odot \mathcal{F}(u_X, \mathcal{G}(u_Y, E_{\mathcal{X}_u}))$
$\mathcal{G}(u_Y, E_{\mathcal{X}_u})$
Interacted items in domain X
Item embedding $E_{\mathcal{X}_u}$
Generator $\mathcal{G}$
Generator $\mathcal{F}$
Cycle consistency loss
User embedding in domain X, u_X
User embedding in domain Y, u_Y
$E_{\mathcal{X}_u} \odot \mathcal{F}(u_X, E_{\mathcal{Y}_u})$
$E_{\mathcal{Y}_u} \odot \mathcal{G}(u_Y, E_{\mathcal{X}_u})$
Generation loss
$\mathcal{F}(u_X, \mathcal{G}(u_Y, E_{\mathcal{X}_u}))$
Training
$\mathcal{G}(u_Y, \mathcal{F}(u_X, E_{\mathcal{Y}_u}))$
Interacted items in domain Y
Item embedding $E_{\mathcal{Y}_u}$
Generator $\mathcal{F}$
Generator $\mathcal{G}$
Cycle consistency loss
$\mathcal{F}(u_X, E_{\mathcal{Y}_u})$
$E_{\mathcal{Y}_u} \odot \mathcal{G}(u_Y, \mathcal{F}(u_X, E_{\mathcal{Y}_u}))$

Figure 2: The structure of the proposed cross-domain recommendation model.

generators. For example, while generating item recommendations in domain Y, we feed both the user embedding $\boldsymbol{u}_Y$ and the embeddings $\boldsymbol{E}_{\mathcal{X}_u}$ of her interaction itemset $\mathcal{X}_u$ to the network $\mathcal{G}$. Here, we can treat $\boldsymbol{u}_Y$ as the user u's long-term preferences in domain Y, and $\boldsymbol{E}_{\mathcal{X}_u}$ as the user's short-term preferences in domain X. The objective of $\mathcal{G}$ is to predict u's latest short-term preferences in domain Y, considering both $\boldsymbol{u}_Y$ and $\boldsymbol{E}_{\mathcal{X}_u}$. Therefore, $\mathcal{Y}_u$ can be used as the ground truth to estimate the prediction accuracy of the generator $\mathcal{G}$. As the output of $\mathcal{G}$ is a matrix with size $|\mathcal{Y}_u| \times d$, instead of the concrete items, we cannot directly compare it with $\mathcal{Y}_u$. Therefore, we treat $\mathcal{G}(\boldsymbol{u}_Y, \boldsymbol{E}_{\mathcal{X}_u})$ as the embeddings of the generated itemset and compare it with the embedding $\boldsymbol{E}_{\mathcal{Y}_u}$ of the ground truth itemset $\mathcal{Y}_u$. In this work, we use the maximum mean discrepancy (MMD) [Yujia Li, 2015; Zhang *et al.*, 2017] to define the distance between the output of the generator network and the features of the ground truth itemset. The MMD between a generated item feature distribution $\boldsymbol{M} \in \mathbb{R}^{m\times d}$ and the feature distribution of the interaction items $\widetilde{\boldsymbol{M}} \in \mathbb{R}^{n\times d}$ is defined as:

$$MMD(\boldsymbol{M}, \widetilde{\boldsymbol{M}}) = \frac{1}{m^2}\sum_{a=1}^{m}\sum_{a'=1}^{m}\mathcal{K}(\boldsymbol{M}_a, \boldsymbol{M}_{a'}) + \frac{1}{n^2}\sum_{b=1}^{n}\sum_{b'=1}^{n}\mathcal{K}(\widetilde{\boldsymbol{M}}_b, \widetilde{\boldsymbol{M}}_{b'}) - \frac{2}{mn}\sum_{a=1}^{m}\sum_{b=1}^{n}\mathcal{K}(\boldsymbol{M}_a, \widetilde{\boldsymbol{M}}_b), \tag{1}$$

where $\mathcal{K}(\cdot)$ is the kernel function, $\boldsymbol{M}_a$ and $\widetilde{\boldsymbol{M}}_b$ denote the a^{th} row of $\boldsymbol{M}$ and the b^{th} row of $\widetilde{\boldsymbol{M}}$. Empirically, we use Gaussian kernel with bandwidth τ, *i.e.*, $\mathcal{K}(\boldsymbol{f}, \boldsymbol{f}') = \exp(-||\boldsymbol{f} - \boldsymbol{f}'||^2/2\tau)$, as the kernel function. Then, we can define a ***generation loss*** for learning generators:

$$\ell_{gen}(\boldsymbol{u}_X, \boldsymbol{u}_Y, \mathcal{X}_u, \mathcal{Y}_u) = MMD\big(\mathcal{G}(\boldsymbol{u}_Y, \boldsymbol{E}_{\mathcal{X}_u}), \boldsymbol{E}_{\mathcal{Y}_u}\big) + MMD\big(\mathcal{F}(\boldsymbol{u}_X, \boldsymbol{E}_{\mathcal{Y}_u}), \boldsymbol{E}_{\mathcal{X}_u}\big). \tag{2}$$

Although the generation loss defined in Eq. (2) can help generate relevant but interest-oriented items in target domains, there also exists another important connection between items and networks. More specifically, we argue that the generation procedure should be cycle-consistent [Zhu *et al.*, 2017]. For example, when we use the output of $\mathcal{G}$ as one of the input of network $\mathcal{F}$, the translation cycle needs to bring the output back to one of the initial input of network $\mathcal{G}$, *i.e.*, $\mathcal{G}(\boldsymbol{u}_Y, \boldsymbol{E}_{\mathcal{X}_u}) \rightarrow \mathcal{F}(\boldsymbol{u}_X, \mathcal{G}(\boldsymbol{u}_Y, \boldsymbol{E}_{\mathcal{X}_u})) \approx \boldsymbol{E}_{\mathcal{X}_u}$. Similarly, the translation cycle from $\mathcal{F}$ to $\mathcal{G}$ follows the same strategy: $\mathcal{F}(\boldsymbol{u}_X, \boldsymbol{E}_{\mathcal{Y}_u}) \rightarrow \mathcal{G}(\boldsymbol{u}_Y, \mathcal{F}(\boldsymbol{u}_X, \boldsymbol{E}_{\mathcal{Y}_u})) \approx \boldsymbol{E}_{\mathcal{Y}_u}$. Therefore, we also consider the following ***cycle consistency loss*** for learning the parameters of generators:

$$\ell_{cyc}(\boldsymbol{u}_X, \boldsymbol{u}_Y, \mathcal{X}_u, \mathcal{Y}_u) = \big\|\mathcal{F}(\boldsymbol{u}_X, \mathcal{G}(\boldsymbol{u}_Y, \boldsymbol{E}_{\mathcal{X}_u})) - \boldsymbol{E}_{\mathcal{X}_u}\big\|_F^2 + \big\|\mathcal{G}(\boldsymbol{u}_Y, \mathcal{F}(\boldsymbol{u}_X, \boldsymbol{E}_{\mathcal{Y}_u})) - \boldsymbol{E}_{\mathcal{Y}_u}\big\|_F^2, \tag{3}$$

where $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. Then, we use a linear combination of the loss functions in Eq. (2) and Eq. (3) to define the final loss function as follows:

$$\ell_{gen}(\boldsymbol{u}_X, \boldsymbol{u}_Y, \mathcal{X}_u, \mathcal{Y}_u) + \lambda\ell_{cyc}(\boldsymbol{u}_X, \boldsymbol{u}_Y, \mathcal{X}_u, \mathcal{Y}_u). \tag{4}$$

By incorporating the cycle consistence loss, CGN constrains the sizes of input itemsets to be the same (*i.e.*, $|\mathcal{X}_u| = |\mathcal{Y}_u|$).

3.3 Model Training

To exploit users' historical behavior data for model training, we firstly assume the user's interests are consistent in a short temporal period, and then split the historical interactions of each user into several non-overlapping partitions according to the interaction timestamps. For each user u, her interaction data $\mathcal{L}_u^X$ are divided into K_u parts as $\mathcal{L}_u^X = \{\mathcal{L}_{u,1}^X, \mathcal{L}_{u,2}^X, \ldots, \mathcal{L}_{u,K_u}^X\}$, where $\mathcal{L}_{u,k}^X$ denotes the user's interaction itemset in domain X in the k^{th} temporal period. The same data processing operation has also been applied

Algorithm 1 Training Algorithm of CGN Model

```
Input: The observed interactions in domain $X$ and domain $Y$, embeddings of all users and items in $X$ and $Y$, learning rate $\theta$, batch size $B$, number of recommended items $N$, regularization parameter $\lambda$;
Output: The generation networks $\mathcal{G}$ and $\mathcal{F}$;
 1: Initialize the replay buffer $\mathcal{B} \leftarrow \emptyset$;
 2: Randomly initialize the parameters of generators $\mathcal{G}$, $\mathcal{F}$;
 3: for each $u \in \mathcal{U}$ do
 4:     for $k = 1, 2, \ldots, K_u$ do
 5:         Receive $u$'s itemsets $\mathcal{L}^X_{u,k}$ and $\mathcal{L}^Y_{u,k}$ and construct $\mathcal{X}^k_u$ and $\mathcal{Y}^k_u$ from $\mathcal{L}^X_{u,k}$ and $\mathcal{L}^Y_{u,k}$, respectively;
 6:         Add the training sample $< \boldsymbol{u}_X, \boldsymbol{u}_Y, \boldsymbol{E}_{\mathcal{X}^k_u}, \boldsymbol{E}_{\mathcal{Y}^k_u} >$ to replay buffer $\mathcal{B}$;
 7:     end for
 8: end for
 9: for $episode = 1, 2, \cdots, max\_iter$ do
10:     Sample a mini-batch of $B$ training samples from $\mathcal{B}$ with replay sampling techniques;
11:     Update both $\mathcal{G}$ and $\mathcal{F}$ by minimizing the loss function in Eq. (5);
12: end for
```

to $\mathcal{L}^Y_u = \{\mathcal{L}^Y_{u,1}, \mathcal{L}^Y_{u,2}, \ldots, \mathcal{L}^Y_{u,K_u}\}$. Note that K_u is determined by the data processing strategy. The details of the data processing operation used in this work are introduced in Section 4.1. For each temporal period, we use CGN to construct the itemset mapping between two domains. We denote the itemset inputs of $\mathcal{G}$ and $\mathcal{F}$ in the k^{th} temporal period by $\mathcal{X}^k_u$ and $\mathcal{Y}^k_u$, respectively. According to the loss definition in Eq. (4), we first assume $|\mathcal{X}^k_u| = |\mathcal{Y}^k_u| = N$, where N denotes the number of recommended items. $\mathcal{X}^k_u$ is build based on $\mathcal{L}^X_{u,k}$. If $\mathcal{L}^X_{u,k}$ has more than N items, we randomly sample N different items from $\mathcal{L}^X_{u,k}$ to be $\mathcal{X}^k_u$. Otherwise, we construct $\mathcal{X}^k_u$ by randomly sampling N items from $\mathcal{L}^X_{u,k}$ with replacement. Similarly, we build $\mathcal{Y}^k_u$ from $\mathcal{L}^Y_{u,k}$. Then, we define the following loss function based on all users' historical data:

$$\sum_{u \in \mathcal{U}} \sum_{k=1}^{K_u} \left[\ell_{gen}(\boldsymbol{u}_X, \boldsymbol{u}_Y, \mathcal{X}^k_u, \mathcal{Y}^k_u) + \lambda \ell_{cyc}(\boldsymbol{u}_X, \boldsymbol{u}_Y, \mathcal{X}^k_u, \mathcal{Y}^k_u) \right]. \tag{5}$$

In the training phase, a user's interaction data in different domains appear in pairs, *e.g.,* $(\boldsymbol{u}_X, \boldsymbol{u}_Y, \mathcal{X}^k_u, \mathcal{Y}^k_u)$. To make the training of the networks more stable, a mini-batch of interaction data pairs are randomly sampled to update the model. The details are summarized in Algorithm 1.

3.4 Item Recommendation

Once the parameters of the generators have been learned, we use $\mathcal{G}$ to generate item recommendation in domain Y, and $\mathcal{F}$ to generate item recommendation in domain X. As the output of a generator is a feature matrix instead of a set of items, we develop an item mapping module to map the generator output to real items. Assuming $\boldsymbol{H} \in \mathbb{R}^{N \times d}$ is the output matrix of a generator, we consider each row vector $\boldsymbol{H}_i \in \mathbb{R}^{1 \times d}$ as a pseudo item vector. Then, for each pseudo item vector, we retrieve the most similar item from the candidate item set, according to the Euclidean distance between the pseudo item vector and embeddings of candidate items. Figure 3 shows an example of the item recommendation procedure in domain Y.

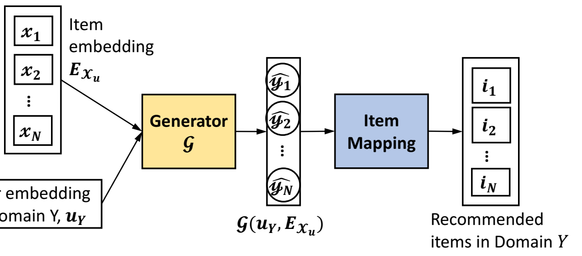


Figure 3: The recommendation process for domain Y

Datasets	#Initial users	#Initial items	#Valid users	#Valid items	#Valid inter.
Home	2,512 K	410 K	206	4,793	6,637
Clothing	3,117 K	1,136 K		6,159	6,525
Books	8,026 K	2,330 K	800	37,533	159,735
Movies	2,089 K	201 K		22,662	114,808

Table 1: The statistics of experimental datasets.

4 Experiments

4.1 Experimental Settings

Datasets: The experiments are performed on the Amazon-review dataset [He and McAuley, 2016]. We choose users' behavior data on four categories for evaluation, *i.e.,* "Home and Kitchen", "Clothing, Shoes and Jewelry", " Books", and "Movies and TV" (respectively denoted by "Home", "Clothing", "Books", and "Movies"). On the Amazon platform, a user can rate a product only after she has purchased the item. Thus, we convert all the observed review ratings to be positive interactions. In the experiments, each category is treated as a domain. We study two cross-domain recommendation tasks by dividing the datasets into two groups: 1) source domain: "Clothing" and target domain: "Home"; 2) source domain: "Movies" and target domain: " Books".

Setup and Metrics: On each dataset, we sort each user's rating history according to the rating timestamps. Then, for each task, we split a user u's rating history into K_u non-overlapping parts in both domains, by the following strategy. We assume the rating time difference between two neighbouring items in the same partition should be smaller than a temporal threshold Δ. For example, for the rating record $\{i_1, i_2, \ldots, i_j, \ldots, i_T\}$, if $t_{j+1} - t_j \le \Delta$, we consider item i_j and i_{j+1} in the same partition, where t_j denotes the rating time for the j^{th} interaction item. To make the number of items in each partition balanced, we empirically set Δ to be 30 days, and denote the number of partitions in both domains by K^X_u and K^Y_u. Then, we match the partitions across both domains, according to the following steps: 1) For the p^{th} partition $\mathcal{L}^X_{u,p}$ in domain X, we retrieve the nearest partition $\mathcal{L}^Y_{u,q'}$ in domain Y by setting $q' = \arg\min_{q \in [1, K^Y_u]} |\tilde{t}_p - \tilde{t}_q|$, where $\tilde{t}_p$ and $\tilde{t}_q$ denotes the

Datasets	Metrics	BPRMF	Caser	CMF	CoNet	CGN$_{\text{w/o UE}}$	CGN$_{\text{w/o Cycle}}$	CGN
Source Domain: Clothing; Target Domain: Home	Hit Ratio@5	0.1582	0.2260*	0.0113*	0.0622*	0.1695	0.1582	**0.2316**
	Hit Ratio@10	0.2316	0.2486*	0.0226*	0.0735*	0.2316	0.2316	**0.2700**
	Hit Ratio@20	**0.3277**	0.2768*	0.0339*	0.1469*	0.2825	0.3107	0.3220
	Precision@5	0.0757	0.0678*	0.0023*	0.0147*	0.1209	**0.1232**	0.1100
	Precision@10	0.0723	0.0514*	0.0023*	0.0090*	0.1113	0.1215	**0.1249**
	Precision@20	0.0715	0.0412*	0.0017*	0.0102*	0.0760	0.0783	**0.0900**
	Recall@5	0.0312*	0.0301*	0.0021*	0.0070*	**0.0655**	0.0651	0.0612
	Recall@10	0.0613*	0.0418*	0.0068*	0.0091*	0.1141	0.1203	**0.1267**
	Recall@20	0.1194	0.0600*	0.0110*	0.0297*	0.1339	0.1428	**0.1606**
Source Domain: Movies; Target Domain: Books	Hit Ratio@5	0.0405	0.0544	0.0038*	0.0139*	0.0126	0.0708	**0.0777**
	Hit Ratio@10	0.0683	0.0822	0.0101*	0.0405*	0.0240	0.0936	**0.1113**
	Hit Ratio@20	0.0910	0.1113	0.0177*	0.0797*	0.0202	0.0961	**0.1290**
	Precision@5	0.0162	0.0134	0.0008*	0.0038*	0.0028	**0.0245**	0.0238
	Precision@10	0.0182	0.0124*	0.0010*	0.0051*	0.0028	0.0214	**0.0230**
	Precision@20	**0.0197**	0.0104	0.0010*	0.0051*	0.0013	0.0130	0.0181
	Recall@5	0.0017	0.0042	0.0003*	0.0009*	0.0005	0.0050	**0.0064**
	Recall@10	0.0045*	0.0081	0.0007*	0.0037	0.0013	0.0092	**0.0140**
	Recall@20	0.0088	0.0120*	0.0018*	0.0079*	0.0025	0.0071	**0.0165**

Table 2: Performances of different recommendation algorithms. The best results are in **bold** faces and the second best results are underlined. "$*$" indicates the improvement of CGN over baseline method is significant with $p < 0.05$ using student *t-test*.

earliest rating time in $\mathcal{L}_{u,p}^X$ and $\mathcal{L}_{u,q}^Y$ respectively. Then, we construct a partition pair $(\mathcal{L}_{u,p}^X, \mathcal{L}_{u,q'}^Y)$; 2) For each partition $\mathcal{L}_{u,q}^Y$ in domain Y, we also retrieve the nearest partition $\mathcal{L}_{u,p'}^X$ in domain X by the same strategy in step 1, and construct a partition pair $(\mathcal{L}_{u,p'}^X, \mathcal{L}_{u,q}^Y)$; 3) We keep the common partition pairs in both steps 1 and 2, and remove other partitions. After these three steps, for each user u, she has the same number of data partitions (*i.e.,* K_u) in both domains. Moreover, we also remove the users that have less than two training partition pairs. After the pre-processing, there are 206 shared users, 4,793 items in the "Home" domain, and 6,159 items in the "Clothing" domain, with 6,637 and 6,525 ratings, respectively. For domains "Books" and "Movies", there are 800 shared users with 37,533 books and 22,662 movies. The numbers of ratings are 159,735 and 114,808, respectively. Table 1 summarizes the statistics of the experimental datasets.

For each user, in each domain, we use her last data partition for testing, and the other partitions for model training. Additionally, the last partition in training data can be used as validation data for choosing hyper-parameters. The performances of the recommendation algorithms are measured by three widely used evaluation metrics: Precision@N, Recall@N, and Hit Ratio@N. In the experiments, we empirically set N to 5, 10, and 20.

Baseline Methods: We compare CGN with the following recommendation methods: (1) **BPRMF** [Rendle *et al.*, 2009]: This is a single-domain recommendation method based on matrix factorization; (2) **Caser** [Tang and Wang, 2018]: This is a deep learning based recommendation method for single-domain recommendation, which employs convolutional neural networks to model the users' personalized sequential behavior patterns; (3) **CMF** [Singh and Gordon, 2008]: This is the collective matrix factorization method for cross-domain recommendation. It assumes a user shares the same embedding in different domains; (4) **CoNet** [Hu *et al.*, 2018]: This is a deep transfer learning approach for cross-domain recommendation. It transfers knowledge across domains by introducing neural layers with shared weight matrices in the deep learning network; (5) **CGN**$_{\text{w/o UE}}$: This is a special case of CGN, without considering the user's embeddings as the generator input; (6) **CGN**$_{\text{w/o Cycle}}$: This special case of CGN does not use cycle consistency loss to learn the model parameters.

Model Implementation: The generators $\mathcal{G}$ and $\mathcal{F}$ of CGN are implemented with the same network structure which consists of five fully connected neural layers. We use ReLU$(\cdot)$ as the activation function in the first four layers, and use Tanh$(\cdot)$ as the activation function in the final layer. The BPRMF method is used to learn the embeddings of users and items in different domains, based on the observed interactions in training data. The Adam algorithm [Kingma and Ba, 2015] is used to learn the model parameters, and the learning rate θ is set to 0.0001. In the experiments, we empirically set the Gaussian kernel width τ to 2, the dimensionality of embedding d to 10, the regularization parameter λ to 0.5, and the training batch size B to 64.

4.2 Summary of Experimental Results

Table 2 summarizes the cross-domain recommendation results on different datasets. We have made the following observations. The proposed CGN method usually achieves better results than baseline methods in terms of most evaluation metrics. For example, CGN significantly outperforms Caser, CMF and CoNet on the target domain "Home". When generating item recommendations in domain "Books", CGN performs better than all baselines, in terms of Hit Ratio and Recall. These results indicate that it is possible to use a user's interaction items in one domain to generate her item recommendation during the same temporal period in another domain. Moreover, the recommendation mechanism of CGN is different from existing methods, by directly learning personalized mapping across different domains over time. Therefore, CGN may be used as a complementary to existing methods. In addition, CGN outperforms CGN$_{\text{w/o UE}}$ and CGN$_{\text{w/o Cycle}}$

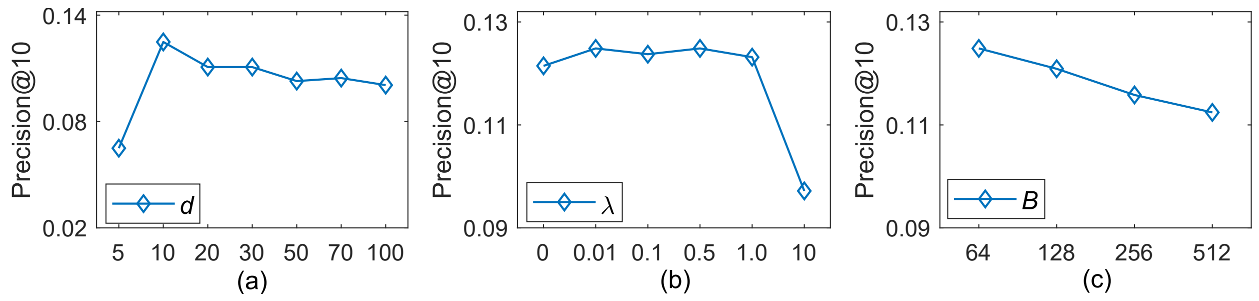
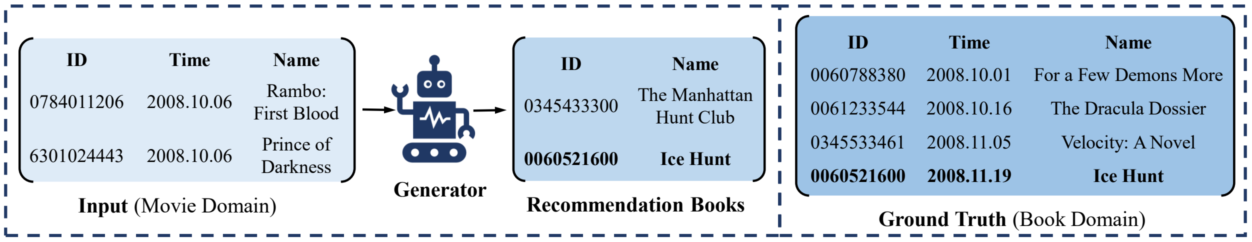


Figure 4: The recommendation accuracy of CGN with respect to different settings of d, λ, and B, measured by Precision@10.



Input (Movie Domain)

ID	Time	Name
0784011206	2008.10.06	Rambo: First Blood
6301024443	2008.10.06	Prince of Darkness

Recommendation Books

ID	Name
0345433300	The Manhattan Hunt Club
0060521600	**Ice Hunt**

Ground Truth (Book Domain)

ID	Time	Name
0060788380	2008.10.01	For a Few Demons More
0061233544	2008.10.16	The Dracula Dossier
0345533461	2008.11.05	Velocity: A Novel
0060521600	**2008.11.19**	**Ice Hunt**

Figure 5: A case study of cross-domain recommendation for user (ID="AIIR8E34EDKCQ").

in most recommendation scenarios. This indicates that both users' personalized preferences and the cycle consistency loss can help improve cross-domain recommendation accuracy.

Moreover, we also study the performances of CGN with respect to (w.r.t.) different settings of three key parameters. Firstly, we vary the dimensionality of the latent space d in $\{5, 10, 20, 30, 50, 70, 100\}$. As shown in Figure 4(a), the recommendation accuracy, in terms of Precision@10, first increases and then gradually decreases with the increase of d. The best performance is achieved when d is set to 10. Moreover, we vary the regularization parameter for cycle consistency loss λ in $\{0, 0.01, 0.1, 0.5, 1.0, 10\}$. From Figure 4(b), we can note that the recommendation accuracy can be improved by incorporating cycle consistency loss in training the generator networks, when λ is set to 0.01, 0.1, 0.5, and 1.0. The recommendation accuracy drops drastically by changing λ from 1.0 to 10. For the batch size B in the model training, we choose B in $\{64, 128, 256, 512\}$. Figure4(c) summarizes the performance trend of CGN w.r.t. different settings of B. We can note that better results can be usually achieved by using a smaller batch size (*e.g.,* $B = 64$).

In addition, we perform a case study to investigate whether the learned itemset mapping can make sense. As shown in Figure 5, we select a user (ID="AIIR8E34EDKCQ") who has watched "Rambo: First Blood" and "Prince of Darkness" in the testing data of Movie domain (*i.e.,* have not been used for model training). After feeding the pre-trained embeddings of the user and these two movies into the generator, we receive two recommended books, *i.e.,* "The Manhattan Hunt Club" and "Ice Hunt". Both "Prince of Darkness" and "Ice Hunt" are about the doomsday theme, and both "Rambo: First Blood" and "The Manhattan Hunt Club" talk about crime. Although the user does not read the book "The Manhattan Hunt Club" in the temporary testing period, she reads a similar book about crime, *i.e.,* "Velocity: A Novel". These results indicate that CGN can learn reasonable mappings between users' behaviors in different domains over time.

5 Conclusion and Future Work

This paper studies the cross-domain recommendation task, where we exploit a user's behavior data in one domain to generate item recommendations in another domain during the same temporal period. We propose a novel recommendation framework named CGN (*i.e.,* Cycle Generation Networks), which aims to learn a personalized mapping between a user's interaction itemsets in different domains. CGN uses two generator networks to enable the dual-direction mapping across domains. Experiments on real datasets demonstrate the effectiveness of the proposed CGN model. As the recommendation mechanism of CGN is different, CGN can be complementary to existing methods. As for future work, we would like to develop different mapping strategies to map the generator outputs to real items. We are also interested in extending CGN to build an interactive recomendation system.

Acknowledgements

This research is supported, in part, by the National Research Foundation, Prime Minister's Office, Singapore under its AI Singapore Programme (AISG Award No: AISG-GC-2019-003) and under its NRF Investigatorship Programme (NRFI Award No. NRF-NRFI05-2019-0002). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of National Research Foundation, Singapore. This research is also supported, in part, by Alibaba Group through Alibaba Innovative Research (AIR) Program and Alibaba-NTU Singapore Joint Research Institute (JRI), Nanyang Technological University, Singapore.

References

[Fu *et al.*, 2019] Wenjing Fu, Zhaohui Peng, Senzhang Wang, Yang Xu, and Jin Li. Deeply fusing reviews and contents for cold start users in cross-domain recommendation systems. In *AAAI*, pages 94–101, 2019.

[He and McAuley, 2016] Ruining He and Julian McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *WWW*, pages 507–517, 2016.

[Hu *et al.*, 2018] Guangneng Hu, Yu Zhang, and Qiang Yang. Conet: Collaborative cross networks for cross-domain recommendation. In *CIKM*, pages 667–676, 2018.

[Hu *et al.*, 2019] Guangneng Hu, Yu Zhang, and Qiang Yang. Transfer meets hybrid: A synthetic approach for cross-domain collaborative filtering with text. In *WWW*, pages 2822–2829, 2019.

[Huang *et al.*, 2019] Ling Huang, Zhi-Lin Zhao, Chang-Dong Wang, Dong Huang, and Hong-Yang Chao. Lscd: Low-rank and sparse cross-domain recommendation. *Neurocomputing*, 366:86–96, 2019.

[Jiang *et al.*, 2016] Meng Jiang, Peng Cui, Nicholas Jing Yuan, Xing Xie, and Shiqiang Yang. Little is much: bridging cross-platform behaviors through overlapped crowds. In *AAAI*, pages 13–19, 2016.

[Kanagawa *et al.*, 2019] Heishiro Kanagawa, Hayato Kobayashi, Nobuyuki Shimizu, Yukihiro Tagami, and Taiji Suzuki. Cross-domain recommendation via deep domain adaptation. In *ECIR*, pages 20–29, 2019.

[Kingma and Ba, 2015] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[Li and Lin, 2014] Chung-Yi Li and Shou-De Lin. Matching users and items across domains to improve the recommendation quality. In *KDD*, pages 801–810, 2014.

[Li and Tuzhilin, 2020] Pan Li and Alexander Tuzhilin. Ddtcdr: Deep dual transfer cross domain recommendation. In *WSDM*, pages 331–339, 2020.

[Lian *et al.*, 2020a] Defu Lian, Qi Liu, and Enhong Chen. Personalized ranking with importance sampling. In *WWW*, pages 1093–1103, 2020.

[Lian *et al.*, 2020b] Defu Lian, Haoyu Wang, Zheng Liu, Jianxun Lian, Enhong Chen, and Xing Xie. Lightrec: A memory and search-efficient recommender system. In *WWW*, pages 695–705, 2020.

[Liu *et al.*, 2015] Yan-Fu Liu, Cheng-Yu Hsu, and Shan-Hung Wu. Non-linear cross-domain collaborative filtering via hyper-structure transfer. In *ICML*, pages 1190–1198, 2015.

[Liu *et al.*, 2017] Yong Liu, Peilin Zhao, Xin Liu, Min Wu, Lixin Duan, and Xiao-Li Li. Learning user dependencies for recommendation. In *IJCAI*, pages 2379–2385, 2017.

[Liu *et al.*, 2018] Yong Liu, Lifan Zhao, Guimei Liu, Xinyan Lu, Peng Gao, Xiao-Li Li, and Zhihui Jin. Dynamic bayesian logistic matrix factorization for recommendation with implicit feedback. In *IJCAI*, pages 3463–3469, 2018.

[Loni *et al.*, 2014] Babak Loni, Yue Shi, Martha Larson, and Alan Hanjalic. Cross-domain collaborative filtering with factorization machines. In *ECIR*, pages 656–661, 2014.

[Man *et al.*, 2017] Tong Man, Huawei Shen, Xiaolong Jin, and Xueqi Cheng. Cross-domain recommendation: An embedding and mapping approach. In *IJCAI*, pages 2464–2470, 2017.

[Pan, 2016] Weike Pan. A survey of transfer learning for collaborative recommendation with auxiliary data. *Neurocomputing*, 177:447–453, 2016.

[Rendle *et al.*, 2009] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461, 2009.

[Singh and Gordon, 2008] Ajit P Singh and Geoffrey J Gordon. Relational learning via collective matrix factorization. In *KDD*, pages 650–658, 2008.

[Tang and Wang, 2018] Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *WSDM*, pages 565–573, 2018.

[Wang *et al.*, 2017] Shoujin Wang, Liang Hu, and Longbing Cao. Perceiving the next choice with comprehensive transaction embeddings for online recommendation. In *ECML-PKDD*, pages 285–302, 2017.

[Wang *et al.*, 2019] Shoujin Wang, Liang Hu, Yan Wang, Quan Z Sheng, Mehmet Orgun, and Longbing Cao. Modeling multi-purpose sessions for next-item recommendations via mixture-channel purpose routing networks. In *IJCAI*, pages 1–7, 2019.

[Wu *et al.*, 2019] Qiong Wu, Yong Liu, Chunyan Miao, Binqiang Zhao, Yin Zhao, and Lu Guan. Pd-gan: adversarial learning for personalized diversity-promoting recommendation. In *IJCAI*, pages 3870–3876, 2019.

[Yuan *et al.*, 2019] Feng Yuan, Lina Yao, and Boualem Benatallah. Darec: Deep domain adaptation for cross-domain recommendation via transferring rating patterns. *IJCAI*, pages 4227–4233, 2019.

[Yujia Li, 2015] Richard Zemel Yujia Li, Kevin Swersky. Generative moment matching networks. In *ICML*, pages 1718–1727, 2015.

[Zhang *et al.*, 2017] Yizhe Zhang, Zhe Gan, Kai Fan, Zhi Chen, Ricardo Henao, Dinghan Shen, and Lawrence Carin. Adversarial feature matching for text generation. In *ICML*, pages 4006–4015, 2017.

[Zhu *et al.*, 2017] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, pages 2223–2232, 2017.

[Zhu *et al.*, 2018] Feng Zhu, Yan Wang, Chaochao Chen, Guanfeng Liu, Mehmet Orgun, and Jia Wu. A deep framework for cross-domain and cross-system recommendations. In *IJCAI*, pages 3711–3717, 2018.