HeroGRAPH: A Heterogeneous Graph Framework for Multi-Target Cross-Domain Recommendation

Qiang Cui
Meituan
Beijing, China
cuiqiang04@meituan.com

Tao Wei
Meituan
Beijing, China
weitao@meituan.com

Yafeng Zhang
Meituan
Beijing, China
zhangyafeng@meituan.com

Qing Zhang
Meituan
Beijing, China
zhangqing31@meituan.com

ABSTRACT

Cross-Domain Recommendation (CDR) is an important task in recommender systems. Information can be transferred from other domains to target domain to boost its performance and relieve the sparsity issue. Most of the previous work is single-target CDR (STCDR), and some researchers recently propose to study dual-target CDR (DTCDR). However, there are several limitations. These works tend to capture pair-wise relations between domains. They will need to learn much more relations if they are extended to multi-target CDR (MTCDR). Besides, previous CDR works prefer relieving the sparsity issue by extra information or overlapping users. This leads to a lot of pre-operations, such as feature-engineering and finding common users. In this work, we propose a heterogeneous graph framework for MTCDR (HeroGRAPH). First, we construct a shared graph by collecting users and items from multiple domains. This can obtain cross-domain information for each domain by modeling the graph only once, without any relation modeling. Second, we relieve the sparsity by aggregating neighbors from multiple domains for a user or an item. Then, we devise a recurrent attention to model heterogeneous neighbors for each node. This recurrent structure can help iteratively refine the process of selecting important neighbors. Experiments on real-world datasets show that HeroGRAPH can effectively transfer information between domains and alleviate the sparsity issue.

CCS CONCEPTS

• **Information systems** → **Collaborative filtering**; **Recommender systems**.

KEYWORDS

heterogeneous, graph, multi-target, cross-domain

Reference Format:
Qiang Cui, Tao Wei, Yafeng Zhang, and Qing Zhang. 2020. HeroGRAPH: A Heterogeneous Graph Framework for Multi-Target Cross-Domain Recommendation. In *3rd Workshop on Online Recommender Systems and User Modeling (ORSUM 2020), in conjunction with the 14th ACM Conference on Recommender Systems, September 25th, 2020, Virtual Event, Brazil.*

ORSUM@ACM RecSys 2020, September 25th, 2020, Virtual Event, Brazil

1 INTRODUCTION

Collaborative Filtering (CF) has become an effective and efficient technique for recommender systems [13]. However, CF methods often face the sparsity issue as real-world datesets usually have a long tail of users and items with few feedbacks. With the development of CF, Cross-Domain Recommendation (CDR) has been proven to be a promising method to alleviate the sparsity. It can transfer rich information from one domain to another to boost performance.

According to different tasks, previous CDR methods can be roughly divided into two categories, i.e., single-target CDR (STCDR) and dual-target CDR (DTCDR). Most CDR methods belong to the former one, which transfers the information from source domain to target domain and not vice versa. These methods can be based on either the feedbacks [8, 19] or the rich side information [2, 14] to relieve the sparsity. The latter DTCDR has recently been studied. Information from source domain and target domain is mutually utilized to improve the performance of both domains. There are usually two approaches to conduct dual-target modeling. The first way is mostly founded on common users [17, 20] as they can clearly restore information from multiple domains. The second way utilizes mapping function [7, 9] performing as a bridge between domains.

Technically, previous works are good at STCDR and DTCDR, but few people study the multi-target CDR (MTCDR). MTCDR is a generalization of DTCDR. Given at least three domains along with the features and feedbacks, the goal of MTCDR is to boost the performance of all domains. It is a more challenging but more general task in real systems. Previous successful DTCDR methods [7, 20] would have some problems if they were extended to MTCDR. First, DTCDR generally models the pairwise relations between domains. If they directly handle n domains, there will be at least C_n^2 relations. Second, most previous works transfer information by users. It is an indirect way to incorporate cross-domain information, because user behaviors at multiple domains are still processed within each domain. Maybe we can collect all behaviors to devise a shared structure such as graph. Such a structure can directly model within-domain and cross-domain behaviors together, because it can acquire feedbacks from all domains as neighbors for a user or an item.

In this work, we propose a **He**t**ero**geneous **GRAPH** framework for MTCDR (**HeroGRAPH**). First, we collect ID information of users and items from multiple domains and build a shared graph.

Nodes include users and items. If a user purchases an item, there will be an edge in this graph. Then we use information within each domain to conduct within-domain modeling, and use the shared graph to handle cross-domain information. Besides, we propose a recurrent attention to aggregate neighbors from multiple domains. Last, we combine within-domain embedding and cross-domain embedding to compute user preference and train the model. The main contributions are listed as follows:

- We propose to introduce a shared structure to model information from multiple domains, such as a graph. This structure can greatly simplify the cross-modeling process.
- We propose to aggregate neighbors from all domains for users and items to relieve the sparsity issue. Besides, we introduce a recurrent attention to iteratively refine the aggregation.
- Experiments on real-world datasets reveal that HeroGRAPH outperforms the state-of-the-art methods and is effective in dealing with the sparsity.

2 RELATED WORK

In this section, we review related works including STCDR, DTCDR and graph neural network.

Compared with single-domain recommendation, STCDR can leverage information from source domain to improve the performance of target domain. [19] proposes a deep adaptation-based model to boost the target domain only with ratings. [8] proposes to use spectral convolutions to acquire high-order connectivity and construct domain-invariant user mapping to transfer knowledge. Other works would use text information like description [14] and attribute information [2] to improve performance. STCDR only aims to have a better target domain performance.

Different from STCDR, DTCDR tries to use the information from target domain to boost the source domain. In another word, the goal of DTCDR is mutual improvement. The concept of DTCDR is first proposed in [20]. This work applies common users to obtain the shared data. It needs different methods to obtain pre-trained features. [7] also belongs to DTCDR. It utilizes an orthogonal mapping to connect two models for two domains. The two models can be mutually connected. This work also generates an extension to muti-target CDR but needs orthogonal matrix between every two domains. These representative DTCDR methods only consider the user for building connections between domains.

Graph neural network has become a rising and shining star nowadays. With the success of GCN [6], graph methods quickly catch the eye of researchers. In recent years, GraphSAGE achieves great success as it can acquire inductive embedding [3]. PinSage can be considered a successful industrial practice [18]. Other graph methods such as GAT [15] also promote development in the field. In recommender systems, graph methods are also outstanding. [16] applies meta-path and attention on heterogeneous graph and achieves the state-of-the-art performance. Encouraged by those works, we can also use graph to address problems in CDR, such as alleviating the sparsity by aggregating neighbors.

3 METHODOLOGY

In this section, we propose a heterogeneous graph framework for multi-target cross-domain recommendation (HeroGRAPH) and its diagram is in Fig. 1. We first formulate the problem. Next, we collect feedbacks for each domain and obtain within-domain embedding for each user and item. Then we gather all feedbacks to build a shared graph and acquire cross-domain embedding. Finally, we compute user preference and apply Bayesian Personalized Ranking (BPR) to train the model.

3.1 Problem Formulation

Let $\mathcal{U}_A$ and $\mathcal{I}_A$ be the sets of users and items respectively. The subscript A represents domain A, and so do domain B, domain C, and so on. Refer to u_A, i_A and (u_A, i_A) as user ID, positive item ID, and a positive feedback pair. In this work, we only use these IDs and feedbacks to build model without any side information. Given at least three domains, our target is to improve the performance of all domains.

3.2 Within-Domain Modeling

In the first stage, we collect feedbacks and obtain within-domain embedding for every user and item in each domain. As the only feature we have is ID, we can easily allocate a vector as the initial embedding for each ID. For domain A, this process can be represented as $E_A(\cdot)$ in Fig. 1, and embeddings for u_A and i_A are E_{u_A} and E_{i_A}, respectively.

3.3 Shared Graph and Cross-Domain Modeling

After obtaining the within-domain embedding, we need to acquire cross-domain embedding.

By using feedbacks from all domains, we construct a heterogeneous graph illustrated in Fig. 2 and all domains will use this graph. In real-world, one platform can provide items in many domains for users. If we split user's feedbacks according to domain label, we will obtain many single-domain datasets and user interest will be split. For example, Amazon dataset [10] has many domains, such as Digital Music, Musical Instruments and Amazon Instant Video. Therefore, it is reasonable to reassemble the data from different domains to acquire better user embedding and item embedding.

The cross-domain modeling can be considered as graph modeling, abbreviated as $G(\cdot)$ in Fig. 1. The corresponding embeddings for u_A and i_A can be represented as G_{u_A} and G_{i_A}, respectively. We consider obtaining G_{u_A} as an example to explain how to fuse information from multiple domains. The basic process is conducted by GraphSAGE [3] with max pooling. Now, suppose we have a user in domain A whoes ID is u_A, and its neighbors $N(u_A) = \{i_A, i_B, ..., i_N\}$ are from multiple domains. Their intermediate graph embeddings can be represented as

$$\begin{aligned} q &= h_{u_A} \\ K &= \{h_j \mid j \in N(u_A)\} = \{h_{i_A}, h_{i_B}, ..., h_{i_N}\} \\ V &= \{Ph_j + p \mid h_j \in K\} \end{aligned} \tag{1}$$

where q, K, V are user vector, neighbor's vector and embeded neighbor representation obtained by a fully connected network, respectively. Then, these vectors are used to compute the cross-domain

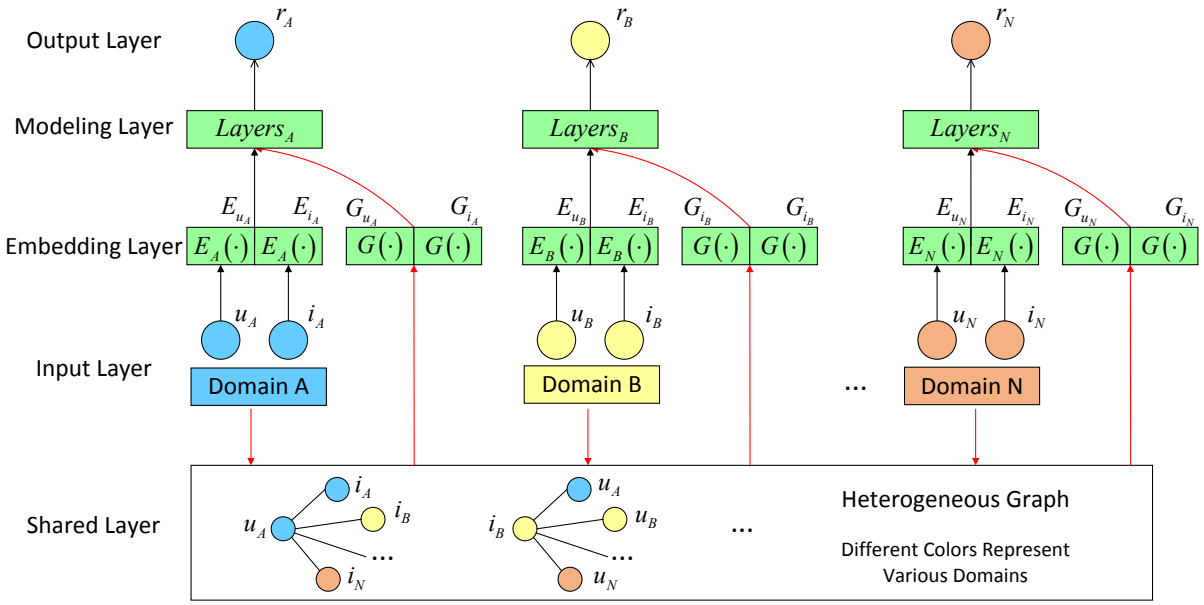


Figure 1: Diagram of the HeroGRAPH model. Black and red arrows between different layers represent the within-domain modeling and cross-domain modeling, respectively. Our model gathers information from multiple domains to construct a heterogeneous graph to transfer knowledge and boost performance of each domain.

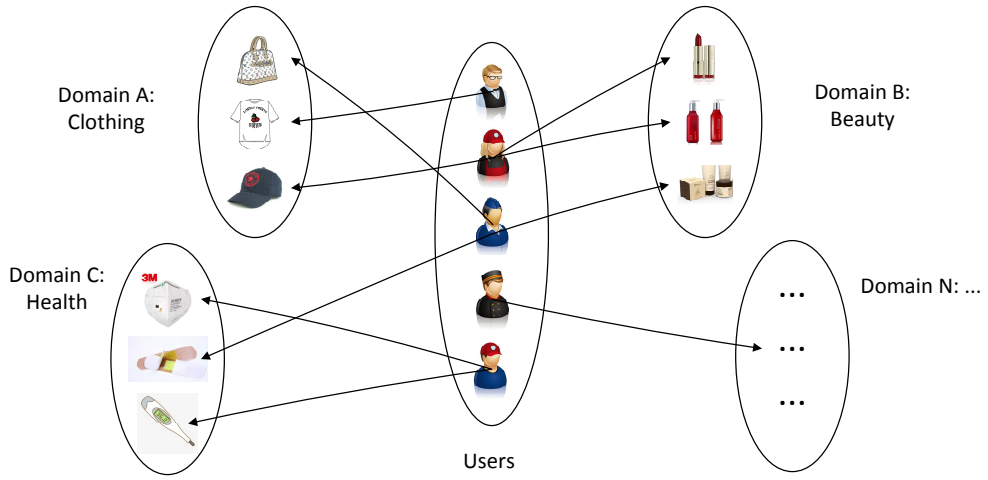


Figure 2: Illustration of the heterogeneous graph. Users may have feedbacks in different domains, and we collect all users and items into one graph as a shared structure. This graph is a bridge among domains. Please note that these domains are limited to one platform, such as Facebook or Amazon.

embedding by

$$\begin{aligned} o_V &= \max(V) \\ G_{u_A} &= \text{ReLU}(W \cdot \text{CONCAT}(q, o_V) + w) \end{aligned} \tag{2}$$

where o_V is the aggregated neighbor representation and the final graph embedding G_{u_A} is a non-linear combination of q and o_V. Please not that although we no longer need to find overlapping users during modeling, graph modeling still depends on overlapping users to incorporate cross-domain informaion.

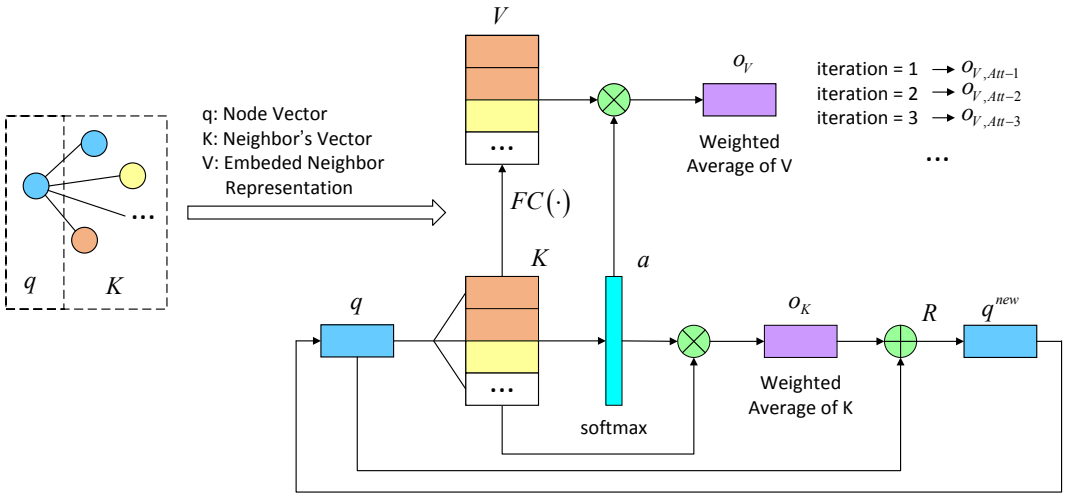


Figure 3: Diagram of the recurrent attention. This attention acts as an aggregator of neighbors of a node. Attention can summarize multiple factors and our work develop a recurrent version to gradually refine this process. The recurrent operation is conducted within node vector q and neighbor's vector K. During each iteration, we compute the output of neighbor's aggregation by attention weight a and embeded neighbor representation V.

3.4 Recurrent Attention for Neighbor Aggregation

In this subsection, we propose a recurrent attention to aggregate neighbors for each node by automatically detecting the importance of each neighbor. The recurrent attention is illustrated in Fig. 3.

Here is the detail of our recurrent attention. First of all, the symbols q, K, V have the same meanings explained in subsection 3.3. Then, the attention weight a is calculated between q and K by Bahdanau Attention [1] and we can obtain a new form of o_V by

$$o_V = V \cdot a \tag{3}$$

As neighbors are from multiple domains, we expect to gradually refine the process of obtaining attention weight a. In order to do this, we aggregate K and update q by

$$\begin{aligned} o_K &= K \cdot a \\ q^{new} &= R \cdot (q + o_K) \end{aligned} \tag{4}$$

where R is a linear mapping and $q + o_K$ is a short-cut connection. Next, q^{new} acts as as new q to recalculate a.

Obviously, we can obtain multiple aggregated neighbor representations as $o_{V,Att-1}, o_{V,Att-2}$ and so on, where subscript $Att-1$ and $Att-2$ means recurrent attention is conducted once and twice respectively. In addition, we add a dropout layer to q and K to avoid overfitting when we first get them.

3.5 Training Framework

In this subsection, we obtain user preference and train the model. Equations are introduced based on domain A.

The positive user preference is calculated based on matrix factorization

$$\hat{x}^t_{ui_A} = E^t_{u_A} \cdot E^t_{i_A} + G^t_{u_A} \cdot G^t_{i_A} \tag{5}$$

where superscript t represents a sample (u_A, i_A) with a certain timestamp. Then we apply the widely-used pair-wise Bayesian Personalized Ranking (BPR) [11] to train the model

$$l^t_{uij_A} = -\ln \sigma \left(\hat{x}^t_{ui_A} - \hat{x}^t_{uj_A} \right) \tag{6}$$

where $\hat{x}^t_{uj_A} = E^t_{u_A} \cdot E^t_{j_A} + G^t_{u_A} \cdot G^t_{j_A}$ is negative preference based on negative feedback pair (u_A, j_A). Finally, the loss function for domain A is

$$\Theta^*_A = \underset{\Theta}{\operatorname{argmin}} \sum_u \sum_{t=1}^{t=|u|} l^t_{uij_A} + \frac{\lambda_\Theta}{2} \|\Theta\|^2 \tag{7}$$

where $|u|$ reprents the number of all samples of user u. The total loss is $\Theta^* = \Theta^*_A + \Theta^*_B + \Theta^*_C + \dots$ and parameters are updated by Adam with default values [5].

4 EXPERIMENTS

In this section, we conduct experiments, analyze the sparsity issue and the proposed recurrent attention.

4.1 Experimental Settings

Datasets. The experiment is conducted on Amazon 5-core dataset [10]. We choose six domains and divide them into two tasks. Each task has three domains. The statistics of each domain are listed in Table 1. Please note that the number of feedbacks is equal to the number of reviews listed on the website [1].

Evaluation Protocols. All datasets are divided into training set, validation set and test set by time. Specifically, the time range of validation set is between 1-Mar.-2014 and 30-Apr.-2014. The ratio of the amount of feedbacks in three sets is approximately 8:1:1. The performance is evaluated on test set by AUC.

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

Table 1: Amazon dataset. We divide six domains into two tasks.

Task	Task 1			Task 2		
Domain	Music	Instrument	Video	Clothing	Beauty	Health
# Users	5,541	1,429	5,130	39,387	22,363	38,609
# Items	3,568	900	1,685	23,033	12,101	18,534
# Feedbacks	64,706	10.261	37,126	278,677	198,502	346,355

Baselines. Our model is compared with several baselines. (1) BPR [11]: We choose the BPR-MF to model implicit feedback. This is a popular and powerful single-domain method. (2) DDTCDR [7]: This is a state-of-the-art DTCDR method and we extend it to handle three domains. (3) GraphSAGE-pool [3]: It is a popular graph method and we select its max pooling variant. Besides, as our proposed network has recurrent attention, we generate three variants: HeroGRAPH-Att-1, HeroGRAPH-Att-2 and HeroGRAPH-Att-3.

In this paper, we aim to explore a novel structure for MTCDR. Therefore, we choose widely-used matrix factorization to compute dot product similarity for all methods rather than a learned similarity such as NeuMF [4], as it still needs to be carefully studied [12].

Parameter Settings. Our method is implemented by Tensorflow 2.2 [2] and hyper-parameters are chosen based on validation set. The embedding size for each ID is 8. The regularization parameter λ_Θ is 0.01. As for the graph modeling, we apply a uniform sampling used in GraphSAGE [3] to choose neighbors and the sample sizes for first-order neighbors and second-order neighbors are 10 and 5 respectively. Correspondingly, output embedding sizes of first-layer aggregation and second-layer aggregation are 64 and 16 respectively. These parameters are used across all tasks in our work.

4.2 Hyperparameter Optimization

Dropout is a powerful technique to prevent overfitting. We introduce dropout layers in subsection 3.4 and take our HeroGRAPH-Att-2 as an example to study different dropout rates. The performance on validation set is illustrated in Fig. 4 and we choose the best dropout rate as 0.2 for all tasks in our work.

The best rates for different domains may vary. On Task1, they are 0.4, 0.2 and 0.2 for Music, Instrument and Video respectively. On Task2, Clothing, Beauty and Health have best rates as 0.1, 0.4 and 0.3 respectively. Although best rates vary among domains, we choose the best from a global perspective rather than selecting domain-specific best rates. This strategy will affect performance but reduce the amount of parameters.

4.3 Performance Comparison

The overall performance is listed in Table 2. From an overall point of view, our HeroGRAPH gains the best performance. It achieves significant improvement on task 1, while the performance on task 2 is not big.

On task 1, HeroGRAPH performs good on all three domains. On task 2, we find that some methods are comparative, especially BPR gains best performance on domain Beauty. There are several reasons for this phenomenon. First, three domains of task 2 have much more data than that of task 1. The larger the amount of data, the easier it is to learn a good expression. In this case, the single-domain method can achieve good results and will not be affected by data from other domains. Therefore, it will be difficult to improve on such domains. Second, we treat multiple domains as a whole and use the global optimal hyperparameters. This will cause our model to be unable to achieve optimal performance on each domain. In this unfavorable situation, our model still obtains good results on domain Clothing and Health, which shows the effectiveness of our model.

4.4 Analysis of Sparsity Issue

In this subsection, we analyze the sparsity issue. First we choose a sparse set from the whole test set. We calculate number of occurrence per item in test set and items with no more than 5 feedbacks makes up a sparse set. We count the total numbers of feedbacks of sparse set and test set and divide them to obtain a proportion. The higher the proportion, the more serious the sparsity issue. Statistics and experimental results are listed in Table 3.

On tasks 1 and 2, our model can achieve great improvement if that domain has a high proportion of sparse items. If there are fewer sparse items, such as in domain Video, Beauty and Health, our HeroGRAPH is not good enough because of the global optimal hyperparameters. This means that by modeling neighbors for users and items, our model can help relieve the sparsity issue.

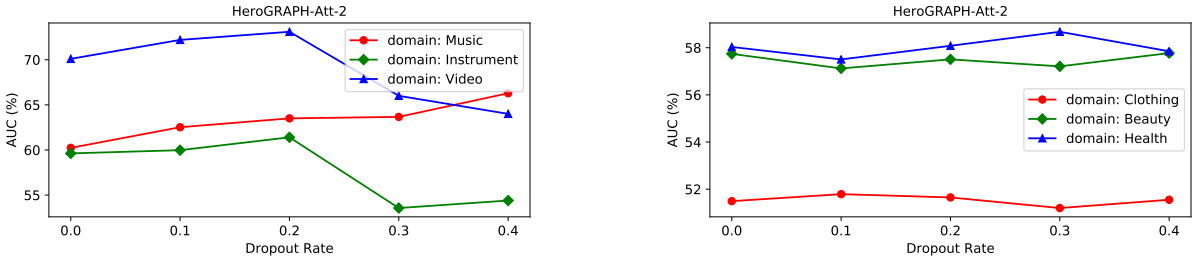
4.5 Analysis of Recurrent Attention

The analysis of recurrent attention is based on Tables 2 and 3 as our variants are listed in the last three lines of each table. Generally speaking, Att-2 is better than Att-1 and Att-3, and it is also better than GraphSAGE. This means that attention may be more useful and recurrent attention can reduce the variance of data. On the other hand, Att-2 performs comparable with Att-1 on task 2. Nearly all values of Att-3 are smaller than that of Att-2. We can conclude that if we have too many iterations, our model may get overfiting. Therefore, we do not perform more iterations.

5 CONCLUSION

In this work, we propose a heterogeneous graph framework for multi-target cross-domain recommendation (HeroGRAPH). This is a challenging but promising task. We fistly propose to use a shared structure to model information from all the domains such as a graph. Then we propose a recurrent attention to gradually refine the process of neighbor aggregation to relieve the sparsity issue. Experiments show the effectiveness of our model.

[2] https://github.com/cuiqiang1990/HeroGRAPH



(a) Task1 (b) Task2

Figure 4: Performance of HeroGRAPH-Att-2 on validation set with different dropout rates.

Table 2: Performance comparison of baselines and our models.

Task		**Task 1**			**Task 2**		
Domain Evaluation on test set		Music	Instrument AUC (%)	Video	Clothing	Beauty AUC (%)	Health
Baselines	BPR [11]	65.36	51.15	67.20	53.84	**59.13**	59.68
	DDTCDR [7]	65.21	53.46	67.40	52.50	57.97	60.23
	GraphSAGE-pool [3]	65.50	59.10	71.30	53.17	58.59	60.04
Our HeroGRAPH	HeroGRAPH-Att-1	66.52	60.14	71.20	52.84	58.40	**60.26**
	HeroGRAPH-Att-2	**67.98**	**62.56**	**73.80**	**54.73**	58.18	60.21
	HeroGRAPH-Att-3	66.38	**62.56**	68.80	51.95	57.23	58.63

Table 3: Performance comparison on sparse set. Items in sparse set appear no more than 5 times in test set.

Task - sparse		**Task 1 - sparse**			**Task 2 - sparse**		
Domain		Music	Instrument	Video	Clothing	Beauty	Health
Number of feedbacks in test set	whole set	687	868	4,988	36,002	23,389	41,622
	sparse set	634	647	1,685	23,266	11,363	18,324
	proportion	92.28%	74.53%	33.78%	64.62%	48.58%	44.02%
Evaluation on sparse set			AUC (%)			AUC (%)	
Baselines	BPR [11]	64.51	51.62	**63.32**	52.64	55.25	52.85
	DDTCDR [7]	67.04	51.93	60.50	52.03	54.69	**53.78**
	GraphSAGE-pool [3]	66.26	54.56	61.40	52.74	**55.51**	53.49
Our HeroGRAPH	HeroGRAPH-Att-1	66.25	56.72	63.20	52.11	54.79	53.57
	HeroGRAPH-Att-2	**68.30**	57.34	60.90	**53.26**	53.88	53.75
	HeroGRAPH-Att-2	67.67	**58.73**	59.30	51.84	53.17	52.18

In the future, we will explore other shared structures such as knowledge graph and try to incorporate user profile or item attribute information. Besides, it is promising to add weights to different preferences within one domain, and to weigh losses between multiple domains. Homoscedastic uncertainty may be a good strategy to investigate weight that can be automatically updated during training.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.
[2] Shlomo Berkovsky, Tsvi Kuflik, and Francesco Ricci. 2007. Cross-domain mediation in collaborative filtering. In *UM*. Springer, 355–359.
[3] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *NeurIPS*. 1024–1034.
[4] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*. 173–182.
[5] Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.
[6] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.

[7] Pan Li and Alexander Tuzhilin. 2020. DDTCDR: Deep Dual Transfer Cross Domain Recommendation. In *WSDM*. 331–339.
[8] Zhiwei Liu, Lei Zheng, Zhang Jiawei, Jiayu Han, and Philip Yu. 2019. JSCN: Joint Spectral Convolutional Network for Cross Domain Recommendation. 850–859. https://doi.org/10.1109/BigData47090.2019.9006266
[9] Tong Man, Huawei Shen, Xiaolong Jin, and Xueqi Cheng. 2017. Cross-Domain Recommendation: An Embedding and Mapping Approach.. In *IJCAI*. 2464–2470.
[10] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *ACM SIGIR*. 43–52.
[11] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*. 452–461.
[12] Steffen Rendle, Walid Krichene, Li Zhang, and John Anderson. 2020. Neural Collaborative Filtering vs. Matrix Factorization Revisited. *arXiv preprint arXiv:2005.09683* (2020).
[13] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW*. 285–295.
[14] Shulong Tan, Jiajun Bu, Xuzhen Qin, Chun Chen, and Deng Cai. 2014. Cross domain recommendation based on multi-type media fusion. *Neurocomputing* 127 (2014), 124–134.
[15] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. *ICLR* (2018).
[16] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *WWW*. 2022–2032.
[17] Xinghua Wang, Zhaohui Peng, Senzhang Wang, S Yu Philip, Wenjing Fu, and Xiaoguang Hong. 2018. Cross-domain recommendation for cold-start users via neighborhood based feature mapping. In *International Conference on Database Systems for Advanced Applications*. Springer, 158–165.
[18] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *ACM SIGKDD*. 974–983.
[19] Feng Yuan, Lina Yao, and Boualem Benatallah. 2019. DARec: deep domain adaptation for cross-domain recommendation via transferring rating patterns. In *IJCAI*.
[20] Feng Zhu, Chaochao Chen, Yan Wang, Guanfeng Liu, and Xiaolin Zheng. 2019. DTCDR: A Framework for Dual-Target Cross-Domain Recommendation. In *CIKM*. 1533–1542.