Continual Learning based on Task Masking for Multi-Domain Recommendation

Tran-Ngoc-Linh Nguyen[1,2] *, Chi-Dung Vu[1,2] *, Hoang-Ngan Le[1], Anh-Dung Hoang[1], Xuan-Hieu Phan[2], Quang-Thuy Ha[2], Hoang-Quynh Le[2], and Mai-Vu Tran[2]

[1] Data Analytics Center - Viettel Telecom - Viettel Group
[2] University of Engineering and Technology - Vietnam National University
{dungvc2, nganlh, linhntn3, dungha7}@viettel.com.vn
{hieupx, thuyhq, lhquynh, vutm}@vnu.edu.vn

Abstract. In recent years, Multi-Domain Recommendation has become a big challenge of recommendation systems, due to the large number of overlapped users and items between multiple domains of many platforms. Multi-Domain Recommendation focuses on capturing informative domain specific features from all domains to improve the corresponding accuracy. In this paper, we proposed a light model, inspired by technique used in Continual Learning selecting the important parameters of users preference for each domain that highly reduces the bias. Our model could also be applied on top of any existing latent model effectively making them usable in multi-domain recommendation settings. We call our architecture as CL4Rec.

Keywords: recommendation system, cross-domain, fairness, graph neural network, matrix factorization, Auto-encoder

1 Introduction

Personalization is the topic of investment with high returns in recent years. Two typical collaborative filtering (CF) algorithms for the recommendation problem are matrix factorization [1] and two-head DNNs [11], [12]. While recent studies focus on the accuracy of the recommended system in the lab and achieve positive results such as BiVAE [14], VASP [13],...

One of the biggest challenge in recommendation system is lack of information from users and items with very little interaction to learn their preferences. This problem is also called data-sparsity. To solve this problem, Cross-Domain Recommendation (CDR) has been proposed to leverage the relatively richer information from a richer domain to improve the recommendation performance in a sparser domain. One of main types of Cross-Domain Recommendation is Multi-Domain Recommendation, focusing on improving the model performance at users of multiple domains simultaneously.

* These authors contributed equally to this work

In reality, many platforms provide items that belong to many different domains and there are always existing users that overlap between each domain. In addition, there are many same products that are provided by different platforms or belong to different domains. Those are all motivations for Multi-Domain Recommendation in recent years. They are focusing on distilling knowledge to transfer between multiple domains to improve the corresponding recommendation accuracy.

However, most of the research about Multi-Domain Recommendation as does not give attention to improving the recommendation accuracy of each domain, which could lead the model to make bias and unfair recommendation. To address those above limitations, we leverages continual learning methods based on Task Masking to detect and protect the important parameters of each domain for reducing bias. Our model could also be applied on top of any existing latent model effectively . To the best of our knowledge, our work is among first to propose the model that could improve the performance at multiple domains simultaneously in multi-domain recommendation when apply to any existing latent model.

Our main contributions include:

- We propose a Multi-Domain Recommendation model that highly performs at multiple domains.
- We propose a model could be applied on top of any existing latent model. This requires virtually less modification to the original model thus minimizing the risk of implementation for any production environment that's already running latent models and make our proposed model easily applied to product.

2 Related work

2.1 Cross-Domain Recommendation

Cross-Domain Recommendation (CDR) is proposed to transfer knowledge between multiple domains that reduces the data sparsity in the recommendation system. There are three types of CDR system as Single-Target CDR, Multi-Target CDR and Multi-Domain recommendation.

First is (Single) Multi-Target Cross-Domain Recommendation that leverages the information of other domains to improve the performance at one or many domains. In each domain, the recommendation system selects the most suitable items which belong to the domain for suggesting.

Second is Multi-Domain Recommendation that is proposed to improve the corresponding recommendation accuracy when the items we recommend belong to different domains by leveraging the auxiliary information in each item domain.

Our research focuses on the Multi-Domain Recommendation problem. Previous works as Zhang et al. [5] proposed a multi-domain collaborative filtering (MCF) framework for solving the data sparsity problem in multiple domains. After this, the MDR models proposed in Cao et.al [6]; Moreno et.al [7]; Pan and Yang et.al [8]; Zhang et.al [9] employ different techniques, i.e., feature combination, transfer learning, and active learning to transfer the knowledge of similar/common users among multiple domains. Most of previous works have a limitation that does not

give attention to improving the recommendation accuracy of each domain, which could lead the model to make a biased and unfair recommendation between multiple domains. It motivates us to propose a new model that addresses this limitation.

2.2 Continual Learning

Continual Learning (also known as Incremental Learning, Life-long Learning) is a concept to learn a model for a large number of tasks sequentially without forgetting knowledge obtained from the preceding tasks, where the data in the old tasks are not available anymore during training new ones.

Continual Learning has attracted much attention in recent years, due to it being greatly applied to many problems in Computer Vision and Natural Language Processing(NLP).

In NLP, there is research about increasing the performance at multiple domains of Sentiment Analysis equally via Continual Learning. Many of them provide the parameters distilling technique called Task Masking, that are state-of-the-art models. Zixuan Ke et al [10] proposed a model for Multi-Domain Sentiment Analysis problems that provides the Task Masking which represents the importance of each position at weights for each task. The Task Masking prevent important parameters to one task from changing during training the other tasks. Each task may share the important parameters with others via the Task Masking, which increases the transfer knowledge between multiple tasks.

That is also the inspiration of our proposed model along with what we described in the section above. For each domain, our model provides a Task Masking that distils and protects the important parameters to user and item embedding for preventing model from making bias and unfair recommendation.

3 Proposed model

In this section, we will first give an overview about the proposed model, then detail each model component. We propose a model consisting of three components as Domain Masking, Domain Specialization and Behavior Extraction module.

The model learns each domain separately. The parameters will be fully trainable in the first domain and frozen except the parameters of user and item embedding matrices from the second domain to reduce training time.

We define N_U and N_I and N_k as the number of users, items and domains we observe. We define the users set, items set and domain ids set as U, V and K. We define d_U and d_I as the length of each user and item embedding vector.

The architecture of the proposal model is shown in Figure 1. The embedding vector for each user and item are randomly initialized. We defines $X_U \in R^{N_U * d_U}$ and $X_I \in R^{N_I * d_I}$ as one of the embedding matrices of users and items, then $X_{U_i} \in R^{d_U}$ and $X_{I_j} \in R^d$ as the embedding of user i and item j.

We apply Continual Learning methods based on Task Masking to Domain Masking and Domain Specialization modules for reducing bias and unfairness during learning user and item embedding matrices. For each domain, Domain Masking

and Domain Specialization modules using one Task Masking with the same length as embedding vectors that represents the importance of each position at user and item embedding. Adding two Task Masking vectors with size d_u and d_i nearly makes modified model nearly not different in space complexity compared to original model.

The Behavior Extraction module gets knowledge from these two modules described above and learns to extract the behaviors of users and items from their interactions then get them located in fixed length vectors.

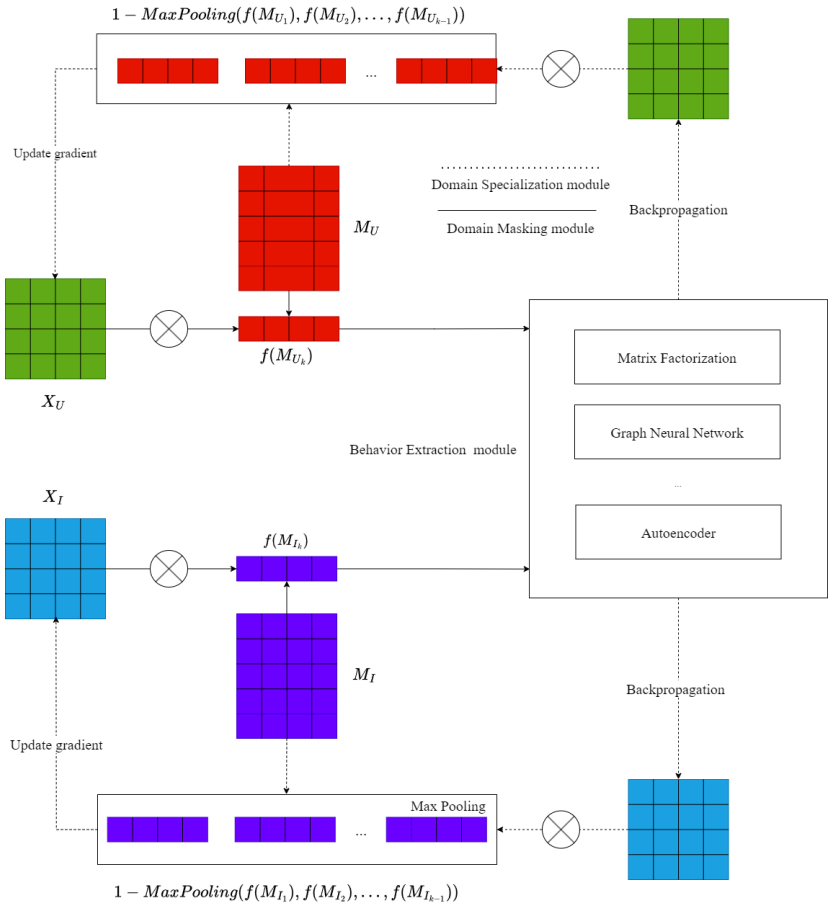

FIG. 1: CL4Rec model architecture

3.1 Domain Masking

Given the domain id k ,the number of users and items of domain k are defined as N_{U_k}, N_{I_k}. The users set and items set of task k are defined as U_k and I_k.

We defines two embedding matrices as $M_U \in R^{N_k * d_u}$ and $M_I \in R^{N_k * d_i}$. For domain k, we uses row k of M_U and M_i as Task Masking called M_{U_k} and M_{I_k} The Domain Masking module transform these two Task Masking via an activation $f(.)$ for better representations, then performs an element-wise between M_{U_k} and X_U, M_{I_k} and X_I for the second preferences of user and item at recent task k

$$X_{U_k} = f(M_{U_k}) \odot X_U$$

$$X_{I_k} = f(M_{I_k}) \odot X_I$$

Our proposed model would use Sigmoid activation function or the function described bellow as $f(.)$ described below, depending on each dataset.

$$f(x) = \begin{bmatrix} 1 \text{ for Sigmoid(x)} > \text{threshold} \\ 0 \text{ for Sigmoid(x)} \leq \text{threshold} \end{bmatrix} \quad (1)$$

Where $threshold$ is a value that we define.

This method reduces overfitting and makes the model focus more on important parameters.

3.2 Domain Specialization

We proposed the Domain Specialization module that protects the important parameters to each domain from easily changing, which may cause a bad effect on recommendation accuracy when learning knowledge from others domains. Since learning the second domain, the Domain Specialization module leverages previous Task Masking vectors to softly mask the position of important values to previous domains at gradient of embedding matrices. We recalculates the gradient of model parameters when training at domain k as:

$$G_{U_k} = G_U \odot (1 - \max(\{f(M_{U_j} \mid j \in N \text{ and } 0 < j < k)\}))$$

$$G_{I_k} = G_U \odot (1 - \max(\{f(M_{I_j} \mid j \in N \text{ and } 0 < j < k)\}))$$

where G_U and G_I denote the gradient of users and items embedding matrix.

With these recalculations, the gradients of most important parameters to the domain model learning in the past are set to nearly zeros, that reduces catastrophic forgetting problem which model forget knowledge at domains learning in the past when learning new domain knowledge. Besides, the less important parameters to model learning in the past are free during learning new domain knowledge, that encourages the knowledge transferring between multiple domains and reduces the sparsity data problem of each domain.

3.3 Inference phase

During Inference or testing phase, the embedding would be calculated as the slide shows to leverage the knowledge from all domains effectively. For each domain k in all domains set K, The embedding vectors of user and item are calculated as:

$$X_{U_k} = X_U \odot \max(\{f(M_{U_j} \mid j \in K)\})$$

$$X_{I_k} = X_I \odot \max(\{f(M_{I_j} \mid j \in K)\})$$

3.4 Behaviors Extraction

As we described in above section, our proposed model is designed to be able to applied on top of any existing latent model. Model is just modifies the embedding vector to extract important knowledge for each domain during inference phase and protects the important parameters for other domains during training phase when model processes the backpropagation, before feeds them to backbone module which could be any existing latent model as Graph Neural Network, Matrix Factorization, Auto-encoder, ... as we called all as Behaviors Extraction module. That is requires only modification to the first embedding vector that is clearly minimizing the risk of implementation for any production environment that's already running latent models.

We consider the backbone module of Graph Neural Network [2] as functions called $GNN(.)$.

Graph Neural Network A graph is represented as $G = (U, V)$, which is defined as $\{(u_i, s_{i_j}, v_j)|u_i \in U, v_j \in V\}$, where U and V separately denote the user and item sets, and a link $s_{i_j} = 1$ indicates that there is an observed interaction between user u_i and item v_j, otherwise $v_{i_j} = 0$. The neighborhood of a node is denoted as $\mathtt{N}(.)$. Given the graph data, the main idea of Graph Neural Network is to iteratively aggregate feature information from neighbors and integrate the aggregated information with the current central node representation during the propagation process

The last embedding vector of users and items at domain k are calculated as:

$$X_{last_{U_k}} = GNN(X_{U_k})$$

$$X_{last_{I_k}} = GNN(X_{I_k})$$

Matrix Factorization The last embedding vector of users and items at domain k are calculated as:

$$X_{last_{U_k}} = X_{U_k}$$

$$X_{last_{I_k}} = X_{I_k}$$

Auto-encoder In the Recommendation System, Auto-encoder model is leveraged to learn to extract features from each user interactions as a vector which each value of it represents user-item rating, then locate them in a latent vector. This model aims to predict unobserved user-item rating or the possible next item user would interact by learning to reconstruct input model or user-item rating from latent vectors described above.

With the Auto-encoder model, our model considers the first MLP layer of Encoder module weight and last MLP layer of Decoder module weight as two item embedding matrices. Final item encoder and decoder embedding are calculated the same as final item embedding in Matrix Factorization.

4 Experiment

We evaluate our proposed method on a real-world popular dataset. We aim to answer the following research questions:

- **RQ1**: How does CL4Rec perform compared to other methods when applying to different types of latent model?
- **RQ2**: How do different components (i.e., Domain Masking, Domain Specialization) affect CL4Rec?

4.1 Experiment setting

Dataset We use Movielens 100K (ML100K) [4], a popular dataset with the demographic of each user, item ratings and user's interaction in the research field to test our proposed architecture performance.

We extract interactions from Action domain as first domain and Fantasy domain as second domain for evaluating Auto-encoder based model and its variant created by combining with our proposed model. The first domain for model learning would be Action domain and the second is Fantasy domain. We extract interactions from Drama domain as first domain and Fantasy domain as second domain for evaluate the based model of Graph Neural Network and Matrix Factorization and their variant created by combining with our proposed model. The first domain for Graph Neural Network and Matrix Factorization model learning would be Drama domain and the second is Comedy domain. The first domain for Auto-encoder model learning would be Action domain and the second is Fantasy domain.

For each user in dataset of one domain , we hold out the last item for testing and use the remaining items for training.

Metrics We define A_i as top k highest ranking items generated by model for user i, B_i as the real items set that user i interacted, N as the number of users.

$$Recall@k = \frac{\sum \frac{|A_i \cap B_i|}{|B_i|}}{N}$$

We calculate model performance at each domain via Recall@k metric and set k as 50. We also calculate the mean performance of two domains as an overall score for fair evaluation.

4.2 Baseline methods

We evaluate our proposed CL4Rec model on ML100K dataset with three representative types of latent model—Graph Bi-Interaction, Matrix Factorization and Auto-encoder—as the base recommenders:

- **Graph Bi-Interaction**: A graph neural network with a new aggregator designed by Xiang Wang et.al [3] which are state-of-the-art in recommendation system models based on graphs.

- **Matrix Factorization**: A popular recommendation system proposed by Y. Koren et.al [1] model based on decomposing the user-item interaction matrix into the product of two lower dimensionality rectangular matrices
- **Auto-encoder**: A model that applies Auto-encoder to recommendation system.

To the best of our knowledge, our work is among first to propose the model that could be applied on top any existing latent model for improving the performance at multiple domains simultaneously in multi-domain recommendation. Then we compare our proposed method with normarlly Training with data of All Domains at Once (TADO) method.

4.3 Performance Comparison(RQ1)

In this experiment, we prove that CL4Rec achieves higher performance when being applied to different types of latent model.

TABLE 1: Overall Performance Comparison

Base recommender	Method	Domain 1	Domain 2	Overall
Matrix Factorization	TADO	35.9	**23.35**	28.615
	CL4Rec	**50.42**	19.31	**34.865**
Graph Bi-Interaction	TADO	47.35	0	23.675
	CL4Rec	**56.31**	**19.2**	**37.755**
Auto-encoder	TADO	**78.34**	53.62	65.98
	CL4Rec	61.95	**100**	**80.98**

Our experiment results at first and second domain show that three popular latent models perform better if they are combined with our proposed model. The based model of Matrix Factorization and Graph Neural Network has a little better performance at first domain but performs so much worse at second domain than their variant created by combining with our proposed model, that lead to worse overall scores. Auto-encoder model combining with our proposed model has worse performance at Action domain, but performs so much better then the based model at Fantasy domain, that leads to higher overall score.

4.4 Ablation study (RQ2)

To verify the impact of the Domain Tasking and Domain Specialization module, we do ablation study by considering two variants of CL4Rec. In particular, we disable the Domain Masking module of CL4Rec. We also disable the Domain Specialization.

Our experiment results show that our proposed model performs better than all of its variants. Our proposed model without Domain Masking module combining with each of three latent model describes above always performs good at first

TABLE 2: Effect of Domain Masking and Domain Specialization

Base recommender	Method	Domain 1	Domain 2	Overall
Matrix Factorization	CL4Rec	50.42	**19.31**	**34.865**
	CL4Rec (w/o Domain Masking)	**59.44**	4.41	31.925
	CL4Rec (w/o Domain Specialization)	43.32	18.24	30.78
Graph Bi-Interaction	CL4Rec	**56.31**	19.2	**37.755**
	CL4Rec (w/o Domain Masking)	55.89	4.89	31.32
	CL4Rec (w/o Domain Specialization)	43.46	**23.06**	33.26
Auto-encoder	CL4Rec	61.95	**100**	**80.98**
	CL4Rec (w/o Domain Masking)	**99.8**	29.4	64.6
	CL4Rec (w/o Domain Specialization)	60.41	99.61	80.01

domain learning, but very bad at second domain learning. It is considered as biased problem, which caused by using all parameters for first domain training make model learns next domain knowledge harder. Our proposed model module combining with each of three latent model describes above always performs better then its variants without Domain Specialization module. There is Graph Bi-Interaction model combining with our model without Domain Specialization has a better performance at second domain learning, but performs so much worse at first domain due to the unprotected of first domain important parameters, that leads to a worse overall score.

5 Conclusion

In this paper, we proposed a Multi-Domain Recommendation model called CL4Rec with efficient knowledge transferring between many domains that reduces bias and improves model performance in all domains simultaneously. Experiment results on the real-world popular dataset show that our model performs better than training data of all domains simultaneously method. Our model achieves higher performance than other methods when being applied to three different types of latent model, that shows our model could truly be applied on top of any existing latent model. This requires virtually less modification to the original model thus minimizing the risk of implementation for any production environment that's already running latent models.

References

1. Y. Koren, R. Bell and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," in Computer, vol. 42, no. 8, pp. 30-37, Aug. 2009, doi: 10.1109/MC.2009.263.
2. Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, Bin Cui,"Graph Neural Networks in Recommender Systems: A Survey", 2 Apr 2022.
3. Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, Tat-Seng Chua,"KGAT: Knowledge Graph Attention Network for Recommendation", 8 Jun 2019.

4. F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4, Article 19 (December 2015), 19 pages. DOI=http://dx.doi.org/10.1145/2827872
5. Yu Zhang, Bin Cao, and Dit-Yan Yeung. Multi-domain collaborative filtering. arXiv preprint arXiv:1203.3535, 2012
6. Bin Cao, Nathan Nan Liu, and Qiang Yang. Transfer learning for collective link prediction in multiple heterogenous domains. In ICML. Citeseer, 2010.
7. Orly Moreno, Bracha Shapira, Lior Rokach, and Guy Shani. Talmud: transfer learning for multiple domains. In CIKM, pages 425–434, 2012.
8. Weike Pan and Qiang Yang. Transfer learning in heterogeneous collaborative filtering domains. Artificial intelligence, 197:39–55, 2013.
9. Zihan Zhang, Xiaoming Jin, Lianghao Li, Guiguang Ding, and Qiang Yang. Multi-domain active learning for recommendation. In AAAI, pages 2358–2364, 2016.
10. Zixuan Ke, Bing Liu, Hu Xu, Lei Shu. CLASSIC: Continual and Contrastive Learning of Aspect Sentiment Classification Tasks.
11. Walid Krichene, Nicolas Mayoraz, Steffen Rendle, Li Zhang, Xinyang Yi, Lichan Hong, Ed Chi, and John Anderson. [n.d.]. Efficient Training on Very Large Corpora via Gramian Estimation. In ICLR 2019
12. Rishabh Mehrotra, Mounia Lalmas, Doug Kenney, Thomas Lim-Meng, and Golli Hashemian. [n.d.]. Jointly Leveraging Intent and Interaction Signals to Predict User Satisfaction with Slate Recommendations. In WWW 2019.
13. Vančura, V., Kordík, P. (2021). Deep Variational Auto-encoder with Shallow Parallel Path for Top-N Recommendation (VASP). In: Farkaš, I., Masulli, P., Otte, S., Wermter, S. (eds) Artificial Neural Networks and Machine Learning – ICANN 2021. ICANN 2021.
14. Truong Quoc-Tuan, Salah Aghiles and Lauw Hady. (2021). Bilateral Variational Autoencoder for Collaborative Filtering. In WSDM 2021