On the Transferability of Deep Neural Networks for Recommender System

Duc NGUYEN†, Hao NIU†, Kei YONEKAWA†, Mori KUROKAWA†, and Chihiro ONO†

† KDDI Research Inc.,
Iidabashi, Chyoda-ku, Tokyo, 102–8460, Japan
E-mail: †{du-nguyen,ha-niu,ke-yonekawa,mo-kurokawa,ono}@kddi-research.jp

Abstract Recommender system is an essential component in many practical applications and services. Recently, significant progress has been made to improve performance of recommender system utilizing deep learning. However, single-domain recommender system suffers from the long-standing data sparsity problem. Transfer learning is a potential approach to deal with the data sparsity problem in recommender system. In this paper, we investigate the transferability of deep neural networks for recommender system. It is found that the neural network responsible for learning the user-item interaction function can be transferred to the target domain, resulting in significant improvement in recommendation performance.

Key words Recommender System, top-N item recommendation, deep learning, transfer learning

1 Introduction

With the explosive growth of information available on the Internet, it is challenging for users to find their desired products/services. Thus, recommender systems (RSs) play a central role to enhance user experience, especially in online news services, E-commerce websites, and online advertising [1]. The main task of RSs is to provide suggestions for items (e.g., news, books, movies, event tickets, etc.) to individual users. RSs enable the so-called *personalized experience*, which is the key to the successes of many Internet companies like Amazon [2], Netflix [3].

Starting with the Netflix Prize [4], significant progress has been made in recommender system research [5]. The past few years have also witnessed the great success of deep learning in many application domains, especially in computer vision and natural language processing [6]. In this trend, in the past few years, deep learning has been studied extensively for recommender system such as in [7] [8] [9] [10] [11] [12] [13]. Although these deep learning-based methods are effective in improving the performance of recommender system, they are mostly based on information (e.g., ratings, reviews) in a single domain. As a result, these methods inevitably suffer from the data sparsity problem because each item is usually rated or reviewed by a few users [1].

Transfer learning is a machine learning technique capable of transferring knowledge from one domain to another related domain [14]. Transfer learning is thus a potential method to deal with the data sparsity problem in recommender system. Generally, there are three transfer approaches, namely *instance-based*, *feature-based*, and *network-based*. Most previous works on transfer learning for recommender system are either instance-based [15] [16] [17] [18] or feature-based [19], [20]. As deep networks are able to learn high-level abstractions (features) from data, it is important to understand the ability to transfer knowledge of deep neural networks or the *transferability* of deep networks for recommender system. Some previous studies such as [21], [22] found that first layers of convolutional neural network are transferable. However, these studies are specific to image classification task. Thus, the findings from those studies might not valid in the context of recommender system, which have many distinguish characteristics.

In this paper, we investigate the transferbility of deep networks for recommender system. For that purpose, we consider *top-N item recommendation task*. A recommender system built on Multi-layer Perception neural network is used as the base network. The base network consists of an embedding layer and a interaction function layer. Then, we examine various options to transfer the knowledge of the base network to a target domain. Experiment results demonstrate that transferring the interaction function layer can significant improve recommendation performance on the target domain. Interestingly, it is found that performance gain is highest when all the hidden layers of the interaction function layer are transferred. We also found that the effectiveness of transfer seems independent with the ratio of share users between domains.

The remaining of the paper is organized as follows. Section 2 surveys related works. The base network and transfer options are described in Section 3. The evaluation is given in Section 4. Finally, the paper is concluded in Section 5.

2 Related Work

In recent years, deep learning-based methods have been studied extensively for recommender system. These methods mainly focus on replacing one or more components in conventional methods by deep networks. For instance, in [7], instead of using the dot product as in traditional matrix factorization [23], the interaction function is learned by a Multilayer Peceptron (MLP) network. In [8], [24], Autoencoder is utilized to learn the user/item embeddings. In [9], Gate Recurrent Unit is used to exploit the order of words in sentences, which is shown to outperform a simple average of word embeddings. Other deep network architectures such as Generative Adversarial Network(GAN) [10] and Attention Model [11] have also been used in recommender system. A comprehensive survey of deep learning-based methods can be found in [5]. In this paper, we adopt the Multilayer Peceptron (MLP) network as the base network due to its simplicity. Other deep networks will be studied in our future work.

In the literature, transfer learning has been used to tackle the data sparsity problem in recommender system. Most transfer learning methods in previous studies are either *instance-based* or *feature-based*. In [15], training samples of a source domain are directly used to train the recommendation model at the target domain. In [16], users/items in a source domain are clustered to construct a codebook, which is then transferred to a target domain. In [19], [20], the user/item latent factors learned on a source domain are transferred to the target domain by means of a mapping function. Different from these previous studies, our work focuses on investigating the transferability of deep networks. Unlike instance-based and feature-based approaches, transfer-based approach does not require that the data of the source domains must be available at the target domain. Thus, it is more applicable considering the strengthen of privacy policy recently.

Some recent studies leverage multi-task learning to enable dual knowledge transfer between domains such as [12], [13]. However, unlike transfer learning, multi-task learning requires that training data of all involved domains must be available at individual domains. Due to privacy and security issues, the above requirement may not always be satisfied in practice.

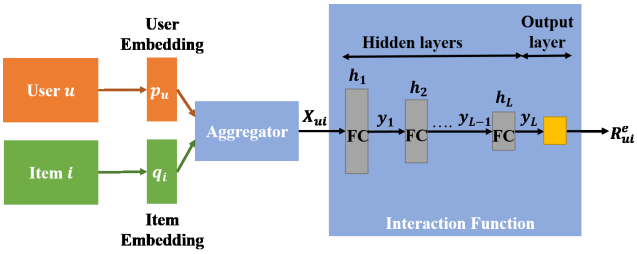


Figure 1: Base network for top-N item recommendation task.

3 Evaluation Framework

3 1 Problem Formulation

In this part, the top-N item recommendation problem in recommender system is formulated and its importance aspects are presented.

Suppose that we need to recommend N items to individual users of a particular system. Let $\mathcal{U}$ and $\mathcal{I}$ respectively denotes the sets of users and items. We define the variables $\{R_{ui}\}$ to represent user-item interactions as follows.

$$R_{ui} = \begin{cases} 1 & \text{if user } u \text{ has interacted with item } i \\ 0 & \text{otherwise} \end{cases}$$

Here, an interaction can be a purchase or rating of the item, a click on the item's advertisement, or a visit to the item's website. The set of items that a user u has interacted with in the past is denoted by $\mathcal{I}_u$, i.e., $\mathcal{I}_u = \{i|R_{ui} = 1\}$. The top-N item recommendation problem can be formulated as follows.

For a user $u \in \mathcal{U}$, determine N items $\{i_1, i_2, .., i_N\} \in \mathcal{I} \backslash \mathcal{I}_u$ that have the highest likelihoods that the user u will interact with.

To solve the above problem, one popular approach is to predict the value R_{ui} for every item $i \in \mathcal{I} \setminus \mathcal{I}_u$. Then, N items with the highest values of R_{ui} are recommended to the user. In this paper, we assume that only implicit feedback is available. Thus, user-item interactions are represented by binary values.

3 2 Network-based Transfer Learning for Recommender System

In this part, a deep learning-based base network for predicting user-item interactions is first presented. Then, we present several typical options for transferring the knowledge of a base network trained on a source domain to a target domain.

3 3 Base Network

In this paper, we follow the NeuMF framework proposed in [7] to build the base network as follows. Each user/item is characterized by a latent vector or embedding. The user-item interactions are modeled by an interaction function. Simi-

lar to [7], the interaction function is a Multi-layer Perception network, which is learned during training.

Figure 1 shows the architecture of the base network used in this paper. As aforementioned, each user $u \in \mathcal{U}$ is characterized by an embedding vector $p_u \in \mathbb{R}^d$, where d is the embedding size. Similarly, each item $i \in \mathcal{I}$ is mapped to an item embedding vector $q_i \in \mathbb{R}^d$. In this paper, we assume that the user and item embeddings have the same size. Yet, it is important to note that the embedding size of the user might be different from that of the item. Given an interaction between user u and item i, the corresponding user and item embeddings are aggregated by the *aggregator*, forming X_{ui}, which is the input of the *interaction function*. In this paper, the aggregator simply concatenates the user and item embedding vectors as follows.

$$X_{ui} = [p_u, q_i] \tag{1}$$

The interaction function consists of L fully connected hidden layers and an output layer. Let s_l denote the size of hidden layer l. The output of the lth $(1 \leqq l \leqq L)$ hidden layer $y_l^h \in \mathbb{R}^{s_l}$ is given by,

$$y_l^h = f_l^h(y_{l-1}^h * W_l^h + b_l^h) \qquad (1 \leqq l \leqq L), \tag{2}$$

$$y_0^h = X_{ui}, \tag{3}$$

where f_l^h, $W_l^h \in \mathbb{R}^{s_{l-1} \times s_l}$, and $b_l^h \in \mathbb{R}^{s_l}$ respectively denotes the activation function, weight, and bias of hidden layer l. At the output layer, the predicted user-item interaction R_{ui}^e is given by,

$$R_{ui}^e = f^o(y_L * W^o + b^o), \tag{4}$$

where f^o, $W^o \in \mathbb{R}^{s_L}$, and $b^o \in \mathbb{R}$ are respectively the activation function, weight, and bias of the output layer. The base network parameter set θ includes the user and item embeddings, the hidden layers' weights and biases, and the output layer's weight and bias.

$$\theta = \{\{p_u\}_{u \in \mathcal{U}}, \{q_i\}_{i \in \mathcal{I}}, \{W_l^h, b_l^h\}_{1 \leqq l \leqq L}, W^o, b^o\} \tag{5}$$

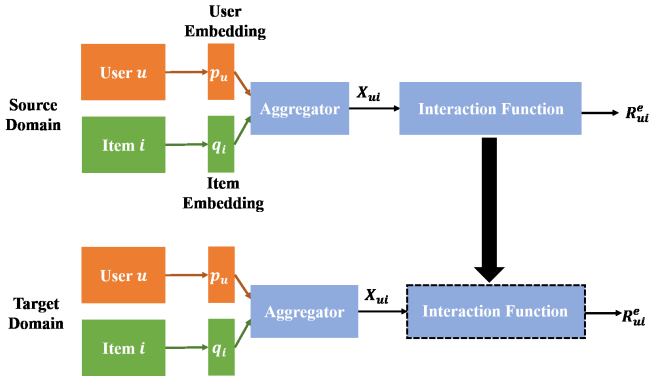
The parameter set θ is learned so as to minimize a loss L, which is a function of the predicted interaction and the actual ones.

$$\min_{\theta} \frac{1}{|\mathcal{U}| \times |\mathcal{I}|} \sum_{(u,i)} L(r_{ui}, r_{ui}^e) \tag{6}$$

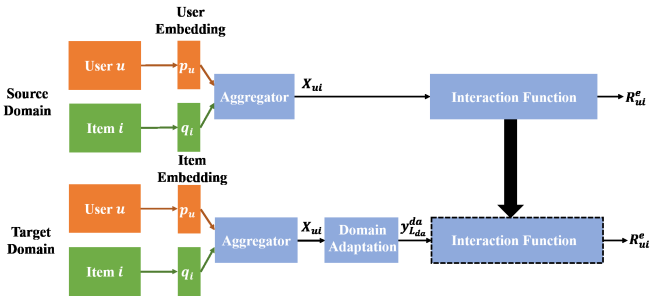
In this paper, since the interaction values are binary, we adopt the binary cross-entropy loss function.

3.4 Network-based Transfer Mechanism

Given a trained base network in a source domain, transferable components are the user/item embeddings, and the interaction function. The interaction function consists of multiple layers, which can be partly or entirely transferred to the target domain. Transferring of the user/item embeddings requires prior knowledge of shared user/item. In this paper, we investigate two transfer options using the interaction function as described in the next part. The transfer of user/item embeddings are reserved for our future work.



(a) *Transfer Option #1*



(b) *Transfer Option #2*

Figure 2: Two transfer options for transfering knowledge of the base network.

3.4.1 Transfer Option #1

In the first transfer option, denoted *TransOpt-1*, the trained interaction function on the source domain is transferred to the target domain. Since the interaction function consists of multiple hidden layers, it is possible to freeze a subset of the hidden layers, whereas retraining the others. We will discuss this in more details in the next section.

3.4.2 Transfer Option #2

Similar to *TransOpt-1*, the interaction function is transferred to the target domain in the second transfer option, denoted *TransOpt-2*. In addition, the output of the aggregator goes through a new component called *domain adaptation*, which consists of fully connected hidden layers. The goal of the domain adaptation is to deal with the discrepancy between the source and target domains. Specifically, the domain adaptation consists of L_{da} fully connected hidden layers. The output of the lth $(1 \leqq l \leqq L_{da})$ hidden layer of the domain adaptation is given by,

$$y_l^{da} = f_l^{da}(y_{l-1}^{da} * W_{l-1}^{da} + b_{l-1}^{da}) \qquad (1 \leqq l \leqq L_{da}), \tag{7}$$

Table 1: Statistics of the Movies and Books datasets

Statistics	Movies	Books
#Users	14581	14581
#Items	10686	19818
#Ratings	1634705	2037320
Sparsity(%)	99.87	99.93

Table 2: Transfer settings of hidden layers of the interaction function network.

Setting	**Layers to freeze**
All	The entire network
h[1,2,3]	The first, second, third hidden layers
h[1,2]	The first and second hidden layers
h[1]	The first hidden layer

$$y_0^{da} = X_{ui}. \tag{8}$$

Accordingly, the first hidden layer of the interaction function is equal to the output of the domain adaptation. As a result, equation (3) now becomes

$$y_0^h = y_{L_{da}}^{da}. \tag{9}$$

It should be noted that other deep networks such as Autoencoder can be used for the domain adaptation. Therefore, determining the optimal network for the domain adaptation is an important issue and will be considered in our future work.

4 Evaluation

4 1 Experiment Setup

4 1. 1 Datasets

In our evaluation, two real-world datasets provided at [25] are used. The first dataset, Books, contains user ratings of books sell on Amazon.com website. The second dataset, Movies, includes user ratings of movies sold on the same website. The original datasets are preprocessed as follows. For both datasets, we first remove users and items with less than 5 interactions. Then, we retain only the shared users between the two datasets. Statistics of the two datasets after processing are shown in Table 1. In our evaluation, we consider two transfer scenarios. In the first scenario, Movies is taken as source domain and Books is taken as target domain. In the second scenaio, Books is chosen as the source domain, wheares Movies is the target domain.

4 1. 2 Model Parameters

The user and item embedding sizes are both set to 32. The interaction function consists of 4 layers with the sizes of 64, 32, 16, and 8. It should be noted that the size of the first hidden layer of the interaction function network is equal to the sum of the user and item embedding sizes. All the hidden layers of the domain adaptation of *TransOpt-2* have the same size of 64. We compare the two transfer options with a baseline in which the base network are trained from scratch using data in the the target domain. For both the transfer options and baseline, Adam optimizer is used. The learning rate is set to 0.001. The batch size is 256. The number of epoch is 50. For each method/option, we run the experiment five times and report the average values.

4 1. 3 Evaluation Protocol

To evaluate the proposed method, we follow the leave-one-out evaluation protocol [7]. Specifically, for a user, a test item is randomly chosen among the items that the user have interacted with. In addition, 99 negative items, which have not been interacted by the user, are randomly selected. The predicted scores for the test and negative items are calculated. Then, the test item is rank against the negative ones based on the predicted scores. Two performance metrics of hit ratio (HR) and normalized discounted cumulative gain (NDCG) are computed as follows. Let h_u denote the hit position (rank) of the test item of user u. HR is defined as:

$$\text{HR} = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \max(0, 1 - \lfloor h_u/(N+1) \rfloor) \tag{10}$$

The NDCG metric takes into account the hit position of each test item, and are defined as follows.

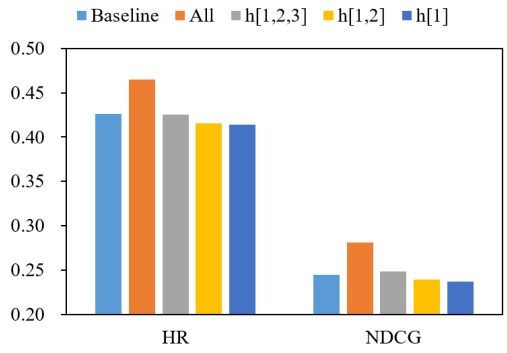
$$\text{NDCG} = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{\log 2}{\log(1 + h_u)} \tag{11}$$

For both metrics, a higher value indicates better performance with the maximum value of 1.

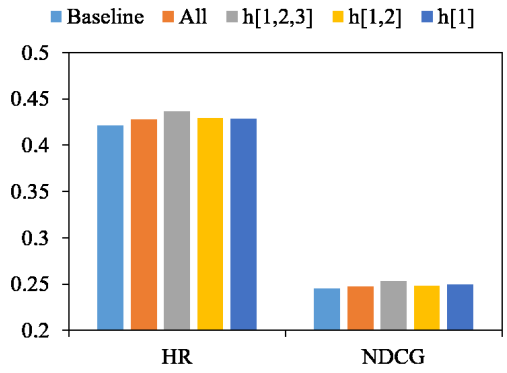
4 2 Evaluation Results

As aforementioned, it is possible to freeze some hidden layers of the interaction function, whereas the other hidden layers are re-trained. Thus, we first investigate how the selection of layers to freeze affect the transfer performance. For that purpose, we consider multiple settings for transferring the interaction function as described in Table 2.

Figure 3 and Figure 4 show the performances of two transfer options under different transfer settings when $N = 10$. Surprisingly, for the *TransOpt-1* option, using the trained rate prediction network without re-training (i.e., *All*) yields the highest performance and significantly outperforms the baseline method (i.e., Fig. 3a and Fig. 4a). In the *All* setting, when transfering from Movie domain to Book domain, the *TransOpt-1* options improves HR and NDCG by 3.9% and 3.7%, respectively. The improvement of the *TransOpt-1* option slightly reduces when transfering from Book domain to Movie domain. In addition, it can be noted that partly retraining the network does not lead to performance improvement compared to the baseline. In the cases of h[1,2] and

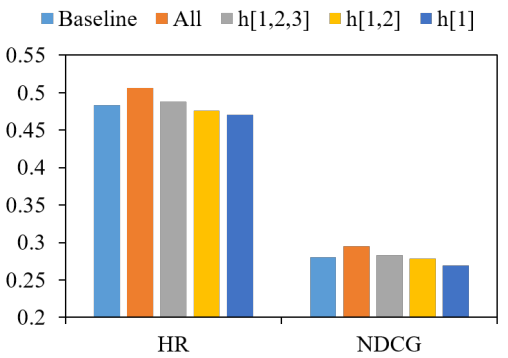


(a) *TransOpt-1*

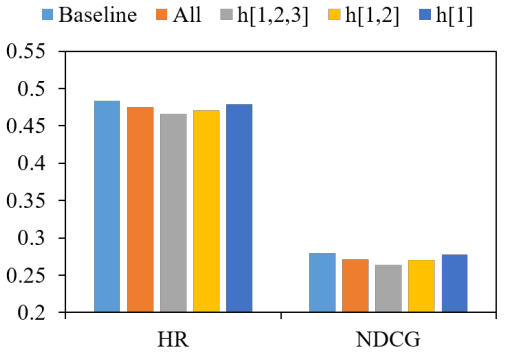


(b) *TransOpt-2*

Figure 3: Performance of the two transfer options under different transfer settings (Movies → Books, $N = 10$).



(a) *TransOpt-1*



(b) *TransOpt-2*

Figure 4: Performance of the two transfer options under different transfer settings (Books → Movies, $N = 10$).

h[1], the *TransOpt-1* option is even worsen than the baseline. Thus, in the following experiments, we adopt the *All* transfer setting when transferring the interaction function for the *TransOpt-1* option.

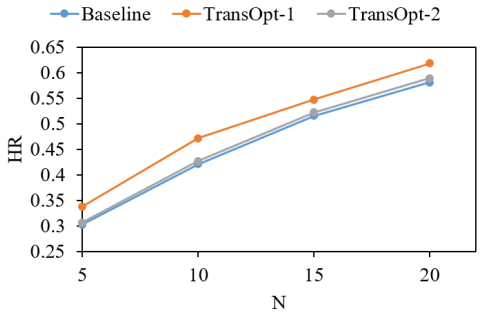
Unlike the *TransOpt-1* option, the best transfer setting for the *TransOpt-2* option is different between two transfer scenarios. When transfering from Movie domain to Book domain, the best transfer setting is h[1,2,3], meaning that all hidden layers are transferred while the output layer is retrained, as can be seen in Fig. 3b. Meanwhile, the transfer setting of h[1] yields the highest recommendation performance when transfering from Book domain to Movie domain (i.e., Fig. 3b). In addition, it can be noted that in the Movie → Book scenario, all the transfer settings lead to better performance than the baseline in terms of both HR and NDCG. In constrast, all the transfer settings of the *TransOpt-2* are worsen than the Baseline in both terms of HR and NDCG. In the following experiments, h[1,2,3] transfer setting is adopted for the *TransOpt-2* option.

Fig. 5 and Fig. 6 show the performance of the two transfer options and the baseline under different values of number of items to recommend N. It can be seen that, the higher the value of N is, the better the performance becomes. In addition, the *TransOpt-1* option always achieves the highest values of HR and NDCG at all considered values of N. Meanwhile, the *TransOpt-2* option is only slightly better than the baseline when transfering from Movie domain to Book domain. In Book → Movie scenario, the performance of the *TransOpt-2* option is lower than that of the *TransOpt-1* option. This result implies that the simpler transfer option *TransOpt-1* is more effective in improving the recommendation performance on the target domain.

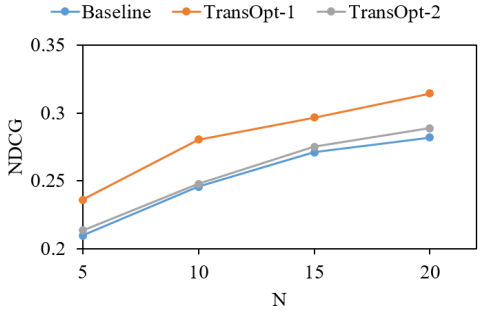
Table 3: Statistics of modified target datasets (Books) with different share user ratios.

Share user ratio (%)	#Users	#Items	#Ratings	Sparsity (%)
100	14581	37449	422045	99.922
69.5	14794	30979	349346	99.923
54.5	14046	26309	288182	99.922
42.4	14524	24699	266971	99.925
17.4	15717	21785	228210	99.933
5.8	16567	19506	211445	99.934
2.8	16781	19009	211343	99.934

In the second part of the experiment, we investigate the impact of the ratio of the share users between the two domains. For that purpose, we replace some users in the target dataset with target-only users. The statistics of the modi-

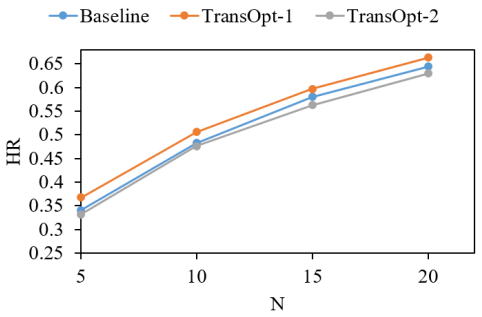


(a) *Hit Ratio*

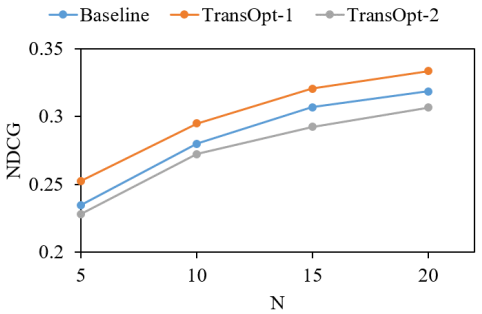


(b) *NDCG*

Figure 5: Impacts of number of recommended items on the performance of the baseline and two transfer options. (Movie → Book)



(a) *Hit Ratio*



(b) *NDCG*

Figure 6: Impacts of number of recommended items on the performance of the baseline and two transfer options. (Book → Movie)

fied target datasets at different share user ratios are shown in Table 3. It is important to noted that due to the requirement that each user/item must has at least 5 interactions, the number of users of the modified datasets are variable.

Fig. 7 shows the performance of the baseline and two transfer options under different share user ratio. Here, the value of N is set to 10. It can be seen that the *TransOpt-1* option consistently improves recommendation performance compared to the baseline. In particular, the *TransOpt-1* enhances the HR by 3.6∼7.0% and the NDCG by 3.2∼5.2%. On the other hand, the improvement of the *TransOpt-2* option compared to the baseline reduces as the share user ratio reduces. Especially, when the share user ratio drops below 40%, the performance of the *TransOpt-2* option is almost the same as that of the baseline in terms of both HR and NDCG. It can also be noted that the performance of the baseline and two transfer options change as the share user ratio decreases. It should be noted that the test sets of different modified datasets are different.

5 Conclusion

In this paper, we investigate the transferability of deep networks for top-N item recommendation task in recommender system. Specifically, we adopt Multi-layer Peceptron (MLP) as the base network, and consider two transfer options using the interaction function. Experiment results with two real-world datasets demonstrate that parts of deep neural network are transferable, leading to enhanced performance on the target domain. Among the two considered transfer options, the simpler one, which transfers the whole interaction function network without re-training, performs better. In future work, we will extend our investigation to other deep network architectures and different transfer options. Also, the evaluation will be extended to include other datasets.

Acknowledgment

This research was partially supported by JST CREST-Grant Number J181401085, Japan.

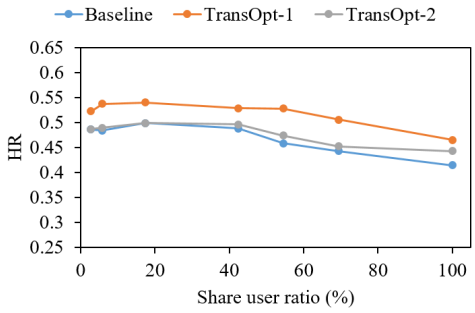
References

[1] F. Ricci, L. Rokach, and B. Shapira, "Introduction to recommender systems handbook," in *Recommender systems handbook*. Springer, 2011, pp. 1–35.

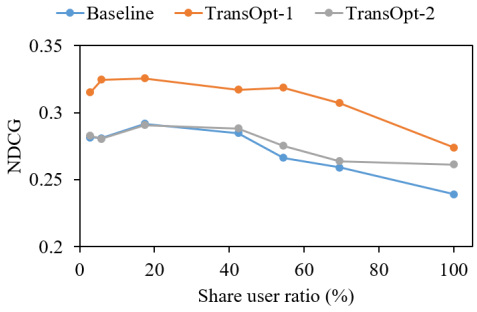
[2] B. Smith and G. Linden, "Two decades of recommender systems at amazon.com," *IEEE Internet Computing*, vol. 21, no. 3, pp. 12–18, May 2017.

[3] C. A. Gomez-Uribe and N. Hunt, "The netflix recommender system: Algorithms, business value, and innovation," *ACM Trans. Manage. Inf. Syst.*, vol. 6, no. 4, pp. 13:1–13:19, Dec. 2015.

[4] J. Bennett, S. Lanning *et al.*, "The netflix prize," in *Pro-*



(a) *Hit Ratio*



(b) *NDCG*

Figure 7: Impacts of share user ratios on the performance of the baseline and two transfer options (Movie → Book, $N = 10$).

ceedings of KDD cup and workshop, vol. 2007, New York, NY, USA., 2007, p. 35.

[5] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Comput. Surv.*, vol. 52, no. 1, pp. 5:1–5:38, Feb. 2019.

[6] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[7] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th International Conference on World Wide Web*, Perth, Australia, 2017, pp. 173–182.

[8] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "Autorec: Autoencoders meet collaborative filtering," in *Proceedings of the 24th International Conference on World Wide Web*, Florence, Italy, May 2015, pp. 111–112.

[9] T. Bansal, D. Belanger, and A. McCallum, "Ask the gru: Multi-task learning for deep text recommendations," in *Proceedings of the 10th ACM Conference on Recommender Systems*, Boston, Massachusetts, USA, Sep. 2016, pp. 107–114.

[10] D.-K. Chae, J.-S. Kang, S.-W. Kim, and J.-T. Lee, "Cfgan: A generic collaborative filtering framework based on generative adversarial networks," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, Torino, Italy, Oct. 2018, pp. 137–146.

[11] S. Seo, J. Huang, H. Yang, and Y. Liu, "Interpretable convolutional neural networks with dual local and global attention for review rating prediction," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, Como, Italy, Aug. 2017, pp. 297–305.

[12] G. Hu, Y. Zhang, and Q. Yang, "Conet: Collaborative cross networks for cross-domain recommendation," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, Torino, Italy, Oct. 2018, pp. 667–676.

[13] F. Zhu, C. Chen, Y. Wang, G. Liu, and X. Zheng, "Dtcdr: A framework for dual-target cross-domain recommendation," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, Dec. 2019.

[14] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, Oct 2010.

[15] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, "Boosting for transfer learning," in *Proceedings of the 24th International Conference on Machine Learning*, Corvalis, Oregon, USA, Jun. 2007, pp. 193–200.

[16] B. Li, Q. Yang, and X. Xue, "Can movies and books collaborate? cross-domain collaborative filtering for sparsity reduction," in *Twenty-First International Joint Conference on Artificial Intelligence*, CA, USA, Jul. 2009, pp. 2052–2057.

[17] W. Pan, E. W. Xiang, N. N. Liu, and Q. Yang, "Transfer learning in collaborative filtering for sparsity reduction," in *Twenty-fourth AAAI conference on artificial intelligence*, Georgia, USA, Jul. 2010, pp. 230–235.

[18] S. Gao, H. Luo, D. Chen, S. Li, P. Gallinari, and J. Guo, "Cross-domain recommendation via cluster-level latent factor model," in *Joint European conference on machine learning and knowledge discovery in databases*, Prague, Czech, Sep. 2013, pp. 161–176.

[19] T. Man, H. Shen, X. Jin, and X. Cheng, "Cross-domain recommendation: An embedding and mapping approach," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, Melbourne, Australia, Aug. 2017, pp. 2464–2470.

[20] F. Zhu, Y. Wang, C. Chen, G. Liu, M. Orgun, and J. Wu, "A deep framework for cross-domain and cross-system recommendations," in *IJCAI International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, Jul. 2018, pp. 3711–3717.

[21] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in neural information processing systems*, 2014, pp. 3320–3328.

[22] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, OH, USA, June 2014, pp. 1717–1724.

[23] R. Bell, Y. Koren, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 08, pp. 30–37, Aug. 2009.

[24] Z. Zhu, J. Wang, and J. Caverlee, "Improving top-k recommendation via jointcollaborative autoencoders," in *The World Wide Web Conference*, ser. WWW '19, May 2019.

[25] J. Ni, J. Li, and J. McAuley, "Justifying recommendations using distantly-labeled reviews and fine-grained aspects," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 188–197. [Online]. Available: https://www.aclweb.org/anthology/D19-1018