Eコマース環境におけるコールドスタート問題のための深層学習に基づくクロスドメイン推薦システムに関する検討

Hanxin WANG†, Daichi AMAGATA†, Takuya MAEKAWA†, Takahiro HARA†, Hao NIU††, Kei YONEKAWA††, and Mori KUROKAWA††

† Graduate School of Information Science and Technology, Osaka University
1-5, Yamadaoka, Suita, Osaka, 565-0871, Japan
†† Integrated Analysis Platform Laboratory, KDDI Research, Inc.
Chiyoda-ku, Tokyo, 102-8460, Japan
E-mail: †{wang.hanxin,amagata.daichi,maekawa,hara}@ist.osaka-u.ac.jp,
††{ha-niu,ke-yonekawa,mo-kurokawa}@kddi-research.jp

Abstract 多くのアプリケーションでは，ユーザの嗜好を予測する推薦システムを用いており，カスタマーエクスペリエンスや売上の向上に繋げている．しかし，推薦システムは，サービスを利用したことがない新規ユーザへの商品の提案が困難という問題（コールドスタート問題）を抱えている．本研究では，この問題を解決するため，オンラインショッピングと広告プラットフォームの両ドメインのブリッジユーザに基づいて，深層学習によりクロスドメイン推薦システムを構築する．評価実験により本システムがコールドスタート問題の解決策の一つになり得ることを示す。

Key words Recommender Systems, Cross-domain Recommendation, Deep Learning, Implicit Feedback, E-commerce, Online advertising

1. Introduction

In the era of information explosion, it is hard for users of online shopping services to explore all available products in limited amount of time. Thus, users prefer to get suggestions of products they might like to buy directly from online vendors. Because of that, recommender systems have become an important part in the e-commerce field, and have also been widely adopted by many other online services like online news [1], video-sharing websites [2] and music streaming platforms [3].

The key part of the personalized recommender systems, which is to model users' preferences on items based on their past interactions such as purchase history, is collaborative filtering (CF) [4]. The most popular collaborative filtering technique is matrix factorization (MF) [5, 6], which projects users and items to a shared latent space by factorizing a user-item interaction matrix. As the immense success on computer vision and natural language processing yielded by deep learning, some recent works have also employed deep neural networks (DNNs) for recommendation, such as neural collaborative filtering [7] or using neural networks to model auxiliary information (e.g., Word2vec [8]). With the powerful ability to learn a high-order nonlinear function, deep neural networks are suitable to learn complex relationship between users and items. However, both traditional MF methods and deep learning based methods suffer from the user cold-start problem, which means that it is difficult to perform accurate recommendation when a new user comes, in particular for recommending items to users who have never used the service before.

It is important to note that situations where a user participates in some different services are getting more and more common. For example, a user buys products in an online shopping site and reads news in other websites. One effective solution for the user cold-start problem is to transfer the knowledge from relevant domains and build a cross-domain recommender system. Because online shopping sites are always combined with Ad platforms to promote the products empirically, it is not difficult to collect a large amount of browsing history in the Ad domain of users who have bought products in the online shopping site. We would like to call users who have both purchase history in online shopping domain and browsing history in Ad domain as *bridge users* in this paper.

Motivated by the above observation, we use a method to represent users by their browsing histories in one Ad platform, and learn user-item relationship in one online shopping service based on bridge users' purchase history to improve the recommendation performance on new users of online shopping service with browsing history in Ad platform. We can even accurately recommend products of online shop-

ping sites to latent users who have never used the service before through ad distribution. We apply different kinds of collaborative filtering models and perform extensive experiments on real-world dataset to show the effectiveness of applying deep learning to cross-domain recommender systems for alleviating the user cold-start problem.

The rest of this paper is organized as follows. We briefly review related work on applying neural networks to recommender systems in Section 2. Then we describe the basic information of the real-world dataset we used and our used method in Section 3. The experimental setup and results are presented in Section 4. Finally, the conclusion and future work are given in Section 5.

2. Related Work

There has been extensive study on applying deep learning to recommender systems. In this section, we review a set of representative studies related to what we use in this paper.

The pioneer study on recommender systems, which used neural networks, proposed a two-layer Restricted Boltzmann Machine (RBM) to model users' explicit ratings on items [9]. Because of the strong performance of neural networks, deep learning has become a popular choice for building a collaborative recommender system [10, 11]. The user-based AutoRec [10] aims at learning a complex hidden structure to reconstruct a user's rating given his or her historical ratings of other items. In order to avoid learning an identity function that always returns the same value as its argument, which performs very poorly on unseen data by general autoencoders, denoising autoencoders (DAEs) [11] have been used to learn from corrupted data, which means artificially adding some noisy to original input data, in order to train more powerful model which can generalize to unseen data. Although these methods have showed the effectiveness of neural networks for CF, they focused on explicit ratings only, which means they could fail to learn users' preference based on positive-only implicit data, such as purchase history and movie viewing activity, because the lack of negative feedback.

For implicit feedback, there are also some recent works that applied deep learning to improve recommendation performance [7, 12]. [12] proposed a collaborative denoising autoencoder (CDAE) for CF with implicit feedback, which adds an additional user node to the input of the model compared with DAEs. [7] presented a state-of-the-art model called neural matrix factorization (NeuMF) which combines a generalized matrix factorization (GMF) model with a multi-layer feed-forward neural network to model user-item interactions. However, CF is generally unable to handle new users and new items, which is known as the cold-start problem.

Handling the cold-start problem in recommender systems is studied mainly for new items (items which have no interactions with any users). This kind of works primarily used DNNs to model auxiliary information, such as acoustic features of musics [3], images of items [13], and textual information of items [14]. Different from most traditional recommender systems within a single domain, in order to alleviate the user cold-start problem, many recent works have focused on combining data from multiple domains, which is called a cross-domain recommender system. One approach is to utilize information of a shared set of users between different domains. For example, [15] proposed a method which improves performance of apps recommendation, news recommendation and movie recommendation simultaneously based on shared users' search queries. The other approach can build a cross-domain recommender system without shared users or items between domains. In [16], the authors developed a generative model to find common clusters between different domains and do recommendation. However, the approach is difficult to scale well because of the computational cost.

Our work combines the advantages of utilizing deep learning as CF technique and applying DNNs to extract users' and items' features. We use information from multiple domains to build a cross-domain recommender system which alleviates the user cold-start problem for a real-world e-commerce site. More specifically, to perform more accurate recommendation for new users without interactions with the service.

3. Method

In this section, we would like to briefly describe the whole process of our method first. We represent users by browsing history in Ad platform without any users' information in the online shopping domain, and represent items by their titles, sub-titles and descriptions through Word2vec. Then we train NeuMF based on bridge users of these two domains and do recommendation to users who have little interaction with the online shopping domain but with enough browsing records in the Ad platform through ads distribution. Figure 1 illustrates the process of our work.

Then we would like to describe some basic information of the real-world dataset we used and existing solutions for handling implicit feedback. Then, we present our architecture to represent users and items in a latent low-dimensional space. Lastly, we give details about the state-of-the-art neural collaborative filtering model NeuMF we used and how we modified the model to fit our problem.

3.1 Learning from implicit feedback

We use a real-world online shopping dataset, an Ad platform dataset and a shared users dataset between these two domains. Online shopping dataset contains users' purchase

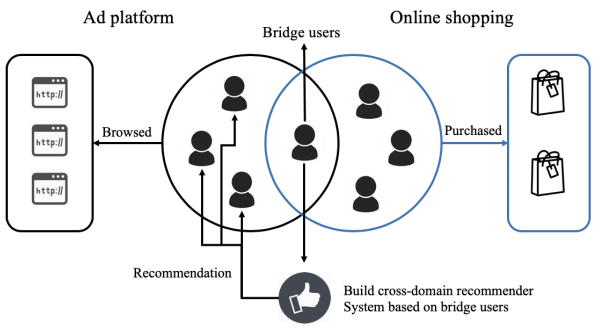

Figure 1 Process of our work

records from *2017-05-11* to *2017-09-30* and products information such as title, textual description, category and price. In Ad platform dataset, there is large amount of users' browsing histories in the same period as the online shopping dataset, when a user accesses a web page where the ads are distributed, an access record will be stored in Ad platform. The shared users dataset is an id mapping table of bridge users between the online shopping domain and the Ad platform domain.

We build the cross-domain recommender system based on users' purchase records, which is a kind of implicit feedback that can be defined as,

$$y_{ui} = \begin{cases} 1, & \text{if user } u \text{ purchased item } i; \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

It is important to note that a value of 1 for y_{ui} does not mean user u actually likes item i, but indicates that an interaction happened between u and i, or u has interest in i. Similarly, $y_{ui} = 0$ does not necessarily mean the user u does not want to purchase this item, maybe just the user did not find the item before, which can be viewed as missing data. According to this fact, implicit feedback contains noisy signals about users' preference and naturally lacks negative feedback.

To handle the problem of recommendation with implicit feedback, many previous works formulated it as predicting the scores of unobserved y_{ui}, which are used for ranking the items [12, 17, 18, 19]. So the problem can be seen as learning $\hat{y_{ui}} = f(u, i|\theta)$ by optimizing an objective function, where θ denotes parameters of predictive model and f denotes an interaction function that represents the relationship between users and items. For the objective function, we follow works [17, 18] to use point-wise loss, which aims at minimizing the point-wise loss between $\hat{y_{ui}}$ and y_{ui}, and sample negative instances from unobserved entries to deal with the absence of negative feedback [7, 17].

3.2 Representation of Users and Items

Because our target is to recommend products in an online shopping domain to new users or people who have never used the service before with abundant browsing history in Ad platform, we decide to represent users only by corresponding browsing records in Ad platform but not information like users profile in online shopping domain, in order to generalize our recommender system to users in other domains. The process for creating representation of users and items are shown in Figure 2.

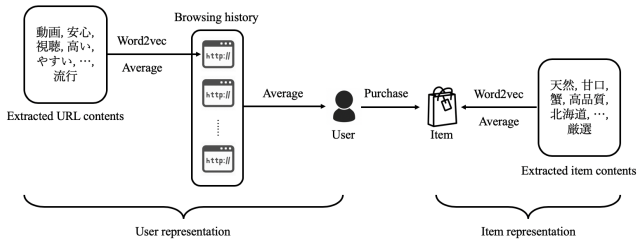

Figure 2 Process for users and items representation

As different websites have different structures and there are so many websites nowadays, it is unrealistic to run crawlers for each website to collect contents, so we only extracted representative contents (title, keywords and description) from web pages in browsing records. In order to save time and reduce computational cost in collecting step, we collected all the contents in the domain name level of URLs (e.g, *https://www.kddi-research.jp/*). However, this method has a problem. It will lose some important information which can distinguish different users more accurately. Although this might further improve the performance, we leave it as a future work. After getting a set of textual contents for each browsing record, we use Mecab(注1) to do the segmentation of Japanese words, and only extract nouns, verbs, and adjectives to filter out meaningless words like stop words. Then we use a pre-trained Word2vec model to turn each word to its corresponding vector, and represent each browsing record as the averaged vector. Similarly, for each user, we average vectors of his/her browsing records as his/her representation.

We adopt a similar process for item representation, i.e., use textual features such as title, subtitle and description of each item and turn them to vectors by Word2vec.

3.3 Generalization of NeuMF

Neural matrix factorization (NeuMF) is a state-of-the-art neural collaborative filtering model that ensembles linear generalized matrix factorization (GMF) and non-linear multi-layer perceptron (MLP). We would like to briefly introduce these two important components of NeuMF first, then present generalized version of NeuMF.

GMF. GMF is a model generalized from MF, which is the

(注1)：http://taku910.github.io/mecab/

most popular model for the recommendation field. In general, MF takes the one-hot encoding of user id and item id as input, learns latent embedding vectors, and finally calculates inner product of user's and item's latent vectors as the predicted score. The interaction function of MF can be defined as

$$y_{ui} = a_{out}(\mathbf{h}^T(\mathbf{p}_u \odot \mathbf{q}_i)), \tag{2}$$

where $\mathbf{p}_u$ denotes user latent vector, $\mathbf{q}_i$ denotes item latent vector, $\odot$ denotes the element-wise product of vectors. Also, a_{out} and $\mathbf{h}$ denote an identity function and a uniform vector with all elements as 1, respectively. If we generalize a_{out} to other activation functions like sigmoid, and set $\mathbf{h}$ as weights that can be learnt from data, MF can be easily generalized to a more powerful and expressive version, which is the GMF.

MLP. MLP is a multi-layer feed-forward neural network, which is used to learn interactions between users and items. It is a more flexible model with a higher level of non-linearity than GMF which can only model user-item interactions linearly from inner-product of two vectors. The MLP model under [7] is defined as

$$\begin{aligned} \mathbf{z}_1 &= \begin{bmatrix} \mathbf{p}_u \\ \mathbf{q}_i \end{bmatrix}, \\ \mathbf{z}_2 &= a_2(\mathbf{W}_2^T \mathbf{z}_1 + \mathbf{b}_2), \\ &\ldots\ldots \\ \mathbf{z}_L &= a_L(\mathbf{W}_L^T \mathbf{z}_{L-1} + \mathbf{b}_L), \\ \hat{y}_{ui} &= \sigma(\mathbf{h}^T \cdot \mathbf{z}_L) \end{aligned} \tag{3}$$

where $\mathbf{p}_u$ denotes a user latent vector, and $\mathbf{q}_i$ denotes an item latent vector as in GMF. Furthermore, $\mathbf{W}_x$, $\mathbf{b}_x$ and $\mathbf{a}_x$ denote a weight matrix, a bias vector and an activation function for the x-th layer of MLP, respectively. For $\mathbf{a}_x$, there are many available choices such as sigmoid, tanh and ReLU. As shown in [20], the sigmoid function is easy to suffer from saturation, which will cause vanishing gradient problem and makes network stop learning. The tanh function can alleviate the problems of sigmoid to some extent [21]. The ReLU has been proven to be non-saturated, well-suited for sparse data, and has higher convergence speed to speed up training. It can also reduce overfitting problem of models [22]. We select ReLU as the activation function in our work just like many other works [7, 23, 24].

NeuMF. Figure 3 shows the structure of generalized NeuMF in our work. NeuMF combines GMF and MLP in their last hidden layers for better modeling user-item interactions by enabling them to mutually reinforce each other. As the authors in [7] allow GMF and MLP to learn separate

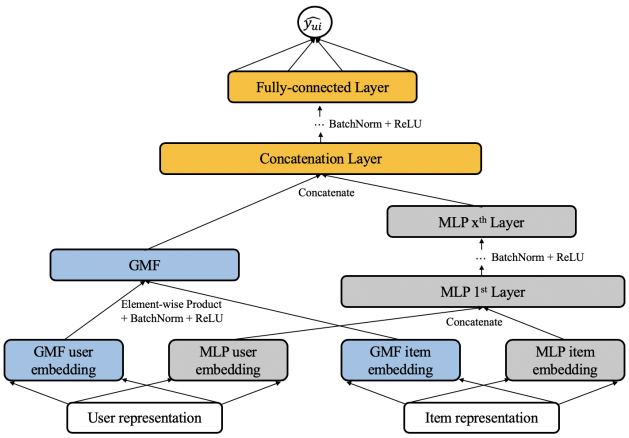

Figure 3 Structure of generalized NeuMF

embeddings, which is more flexible and expressive than sharing the same embedding between these two components for NeuMF. We adopt this setting and generalize NeuMF to fit our problem by using representation of users and items mentioned in Section 3.2 as input, while [7] only uses one-hot encoding of user (item) id as input. The interaction function of NeuMF is defined as follows:

$$\begin{aligned} \mathbf{z}_{GMF} &= \mathbf{p}_u^G \odot \mathbf{q}_i^G, \\ \mathbf{z}_{MLP} &= a_L(\mathbf{W}_L^T(a_{L-1}(\ldots a_2(\mathbf{W}_2^T \begin{bmatrix} \mathbf{p}_u^M \\ \mathbf{q}_i^M \end{bmatrix} + \mathbf{b}_2)\ldots)) + \mathbf{b}_L), \\ \hat{y}_{ui} &= \sigma(\mathbf{h}^T \cdot \begin{bmatrix} \mathbf{z}_{GMF} \\ \mathbf{z}_{MLP} \end{bmatrix}), \end{aligned} \tag{4}$$

where $\mathbf{p}_u^G$ and $\mathbf{p}_u^M$ denote user embedding for GMF and MLP, respectively. Similarly, $\mathbf{q}_i^G$ and $\mathbf{q}_i^M$ denote item embedding for these two parts, and $\hat{y}_{ui}$ is the predicted result of NeuMF. We add a fully-connected layer after concatenation layer to give NeuMF ability to learn more complex user-item interactions. Then we apply batch normalization for each layer in order to overcome the hard to converge problem which might still be caused by ReLU and speed up training [25].

For all models mentioned above, the objective function is defined as

$$L = -\sum_{(u,i)\in\mathcal{Y}\cup\mathcal{Y}^-} y_{ui}\log\hat{y}_{ui} + (1-y_{ui})\log(1-\hat{y}_{ui}), \tag{5}$$

where $\mathcal{Y}$ denotes the set of observed interactions and $\mathcal{Y}^-$ denotes the set of negative samples. This objective function is known as *logloss*.

4. Experiment

In this section, we conduct experiments to evaluate the performance of NeuMF for alleviating the user cold-start

Table 1 Statistics of the training set

Dataset	#interactions	#users	#items	Sparsity(%)
Training	156,287	14,659	18,511	99.942

Table 2 Statistics of the validation set and test set

Dataset	#interactions	#users	#items
Validation	12,410	9,937	1,539
Test	14,443	10,273	2,369

problem. Then we compare NeuMF with several baseline recommendation algorithms to demonstrate the effectiveness of deep cross-domain recommender systems.

We first introduce the setting of real-world dataset we used for training and evaluation. Then we present evaluation criteria, briefly review baseline methods and implementation details about all methods that have been used in the experiment. Finally, we show the recommendation performance of different methods and discuss the results.

4.1 Experimental Settings

Dataset. As mentioned in Section 3.1, we have an ad platform dataset with users' browsing histories and an online shopping dataset with users' purchase records from *2017-05-11* to *2017-09-30*. We divide purchase records into three parts: *2017-05-11* to *2017-09-10* for training, *2017-09-11* to *2017-09-17* for validation, *2017-09-18* to *2017-09-30* for testing, and only use browsing histories from *2017-05-11* to *2017-09-10* for user modeling to avoid using future information to predict the past. As the original data is highly sparse, we retained only users with at least 5 interactions (purchases) as a training set. The statistics are summarized in Table 1. In order to evaluate recommendation performance on new users, we filtered all the purchase records of users in training set out of validation and test sets. The statistics of these two datasets are shown in Table 2.

Evaluation Criteria. To evaluate the performance of item recommendation, we follow the common strategy of top-K recommendation in [7, 23] to rank a target item of each interaction with items that are not interacted by the user. Because the items in our online shopping domain will only exist during a period of time, we found there are on average about 1,500 items for sale each day from *2017-05-11* to *2017-09-10*. Due to this fact, we randomly sample 1499 items having no interaction with the user, and rank the 1500 items, which is different from [7, 23] that only sampled 99 items with no interaction for ranking. The performance of a ranked list can be evaluated by *Hit Ratio* (HR) and *Normalized Discounted Cumulative Gain* (NDCG) [26]. HR intuitively measures whether the target item is presented on the top-K list, while NDCG accounts for hit position by assigning higher scores to hits at top ranks. In our work, we

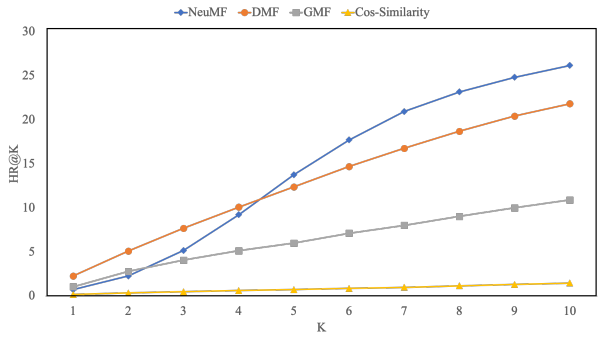

Figure 4 Performance of HR@K

set K as integers from 1 to 10 for evaluation, calculate both metrics for each interaction and show the average score.

Baselines. We compare the NeuMF that we used in our work with different kinds of methods:

- **Cosine Similarity.** This is the standard method to model user-item relationship, which is widely adopted as the final step in recommendation models.

- **GMF [7].** As mentioned in Section 3.3, GMF is a generalized version of traditional MF. It is a shallow model that learns user-item interactions from inner-product of input vectors, which is more expressive than traditional MF.

- **DMF [24].** DMF is a state-of-the-art deep matrix factorization model that projects users and items into a low-dimensional latent space by DNN, and calculates cosine similarity of these two vectors as predicted result. We follow the result in [24] to build a DMF with 2 layers and tune other hyper-parameters to report the best performance.

Implementation Details. All methods are implemented based on PyTorch(注2). For all models except Cosine Similarity, we randomly sampled four negative samples for each positive instance. Parameters to be trained are randomly initialized from Gaussian distribution $\mathcal{N}(0, 0.01^2)$. The optimizer is SGD with batch size of 256, and we tested learning rate of [0.001, 0.005, 0.01]. We also tested the number of neurons in the last hidden layer of [16,32,64,100]. As for structure of neural network models, we used a tower pattern, which halves the layer size for each successive higher layer. For determining the best hyper-parameters for each model, we tuned hyper-parameters based on the validation set.

4.2 Experimental Results

In this section, we report the recommendation performance of the methods and discuss the results. Figure 4 and Table 3 show the performance of HR@K with K varied from 1 to 10. Figure 5 and Table 4 show the performance of NDCG@K in the same situation.

We can see that deep learning based models NeuMF and

(注2)：https://pytorch.org

Table 3 Performance of HR@K(%)

K	1	2	3	4	5	6	7	8	9	10
NeuMF	0.727	2.278	5.172	9.229	**13.771**	**17.725**	**20.951**	**23.167**	**24.815**	**26.165**
DMF	**2.278**	**5.11**	**7.699**	**10.088**	12.394	14.706	16.769	18.694	20.425	21.817
GMF	1.052	2.804	4.078	5.144	6.003	7.125	8.032	9.036	10.005	10.898
Cos-Similarity	0.187	0.346	0.485	0.623	0.727	0.872	0.99	1.156	1.316	1.475

Table 4 Performance of NDCG@K

K	1	2	3	4	5	6	7	8	9	10
NeuMF	0.0073	0.0171	0.0315	0.049	0.0666	0.0807	**0.0914**	**0.0984**	**0.1034**	**0.1073**
DMF	**0.0228**	**0.0406**	**0.0536**	**0.0639**	**0.0728**	**0.081**	0.0879	0.094	0.0992	0.1032
GMF	0.0105	0.0216	0.0279	0.0325	0.0359	0.0399	0.0429	0.046	0.049	0.0515
Cos-Similarity	0.0019	0.0029	0.0036	0.0042	0.0046	0.0051	0.0055	0.006	0.0065	0.0069

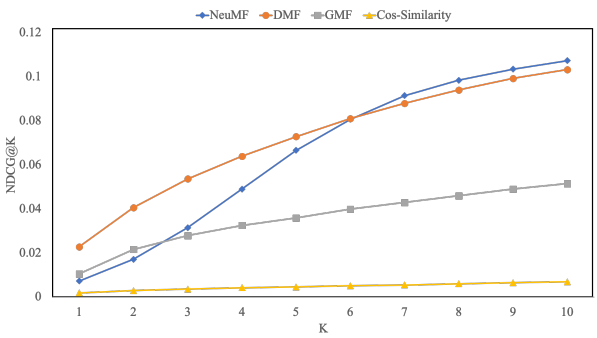

Figure 5 Performance of NDCG@K

DMF consistently give better performance in both metrics than other existing methods for all the values of K, with improvement up to 15.267% in HR@10 compared to the best shallow model GMF. For NDCG@10, the best deep learning based model NeuMF is about 2.08 times as that of GMF. It shows the effectiveness of neural approaches in alleviating user cold-start problem with knowledge introduced from other domain.

We can also find that DMF achieves the best performance than other methods in HR@K for K from 1 to 4. But when K is larger than 4, NeuMF outperforms DMF and achieves 4.348% improvement when $K = 10$. It shows that DMF is better at ranking a part of target items in higher positions than NeuMF does. While NeuMF is better at modeling the overall user-item interactions and can rank more target items into the top-10 ranked list than DMF, which is more practical because it is general to recommend a list of items to users. The results of NDCG@K also show this fact, as DMF still gives better performance than NeuMF in NDCG@5 and NDCG@6, while it has already underperformed NeuMF in HR@5 and HR@6.

5. Conclusion and Future Work

In this work, we transferred information from an Ad platform to an online shopping domain based on bridge users of these two domains. Then we applied different kinds of collaborative filtering models to build cross-domain recommender systems and evaluated their recommendation performance with respect to the new users of online shopping domain. The experimental result shows that deep learning based models far outperform other shallow models for collaborative filtering for alleviating the user cold-start problem. Moreover, NeuMF can model overall user-item interactions better than DMF in HR@K when K is between 5 and 10, which shows the model combines linearity and non-linearity is more powerful than the pure non-linear model in our task.

In future, we will study different kinds of user modeling methods, such as applying TF-IDF [27] to give different weights to different URLs and words, or directly turn URLs to vectors to represent users [28], to further improve the result of NeuMF. Also, we plan to further explore the impact of time to the recommendation performance and how these methods perform respect to new items in online shopping domain. Moreover, we are particularly interested in building cross-domain recommender systems through transfer learning methods like domain adaptation [29].

Acknowledgment

This research is partially supported by JST CREST Grant Number J181401085.

References

[1] Shumpei Okura, Yukihiro Tagami, Shingo Ono, and Akira Tajima. Embedding-based news recommendation for millions of users. In *KDD*, pages 1933–1942, 2017.

[2] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *RecSys*, pages 191–198, 2016.

[3] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. In *NIPS*, pages 2643–2651, 2013.

[4] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation

algorithms. In *WWW*, pages 285–295, 2001.

[5] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

[6] Andriy Mnih and Ruslan R Salakhutdinov. Probabilistic matrix factorization. In *NIPS*, pages 1257–1264, 2008.

[7] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *WWW*, pages 173–182, 2017.

[8] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119, 2013.

[9] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted boltzmann machines for collaborative filtering. In *ICML*, pages 791–798, 2007.

[10] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. Autorec: Autoencoders meet collaborative filtering. In *WWW*, pages 111–112, 2015.

[11] Florian Strub and Jeremie Mary. Collaborative filtering with stacked denoising autoencoders and sparse inputs. In *NIPS workshop on machine learning for eCommerce*, 2015.

[12] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. Collaborative denoising auto-encoders for top-n recommender systems. In *WSDM*, pages 153–162, 2016.

[13] Wenhui Yu, Huidi Zhang, Xiangnan He, Xu Chen, Li Xiong, and Zheng Qin. Aesthetic-based clothing recommendation. In *WWW*, pages 649–658, 2018.

[14] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. Collaborative deep learning for recommender systems. In *KDD*, pages 1235–1244, 2015.

[15] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *WWW*, pages 278–288, 2015.

[16] Bin Li, Qiang Yang, and Xiangyang Xue. Transfer learning for collaborative filtering via a rating-matrix generative model. In *ICML*, pages 617–624, 2009.

[17] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. Fast matrix factorization for online recommendation with implicit feedback. In *SIGIR*, pages 549–558, 2016.

[18] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM*, pages 263–272, 2008.

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

[20] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, pages 249–256, 2010.

[21] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[22] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *AISTATS*, pages 315–323, 2011.

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

[24] Hong-Jian Xue, Xinyu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. Deep matrix factorization models for recommender systems. In *IJCAI*, pages 3203–3209, 2017.

[25] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv:1502.03167*, 2015.

[26] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *TOIS*, 20(4):422–446, 2002.

[27] Gerard Salton and Michael J McGill. Introduction to modern information retrieval. 1986.

[28] Yukihiro Tagami, Hayato Kobayashi, Shingo Ono, and Akira Tajima. Representation learning for users' web browsing sequences. *IEICE Transactions on Information and Systems*, 101(7):1870–1879, 2018.

[29] Heishiro Kanagawa, Hayato Kobayashi, Nobuyuki Shimizu, Yukihiro Tagami, and Taiji Suzuki. Cross-domain recommendation via deep domain adaptation. *arXiv:1803.03018*, 2018.