Deep Collaborative Filtering with Multi-Aspect Information in Heterogeneous Networks

Chuan Shi, *Member, IEEE*, Xiaotian Han, Li Song, Xiao Wang, Senzhang Wang, Junping Du, and Philip S. Yu, *Fellow, IEEE*

Abstract—Recently, recommender systems play a pivotal role in alleviating the problem of information overload. Latent factor models have been widely used for recommendation. Most existing latent factor models mainly utilize the interaction information between users and items, although some recently extended models utilize some auxiliary information to learn a unified latent factor for users and items. The unified latent factor only represents the characteristics of users and the properties of items from the aspect of purchase history. However, the characteristics of users and the properties of items may stem from different aspects, e.g., the brand-aspect and category-aspect of items. Moreover, the latent factor models usually use the shallow projection, which cannot capture the characteristics of users and items well. Deep neural network has shown tremendous potential to model the non-linearity relationship between users and items. It can be used to replace shallow projection to model the complex correlation between users and items. In this paper, we propose a Neural network based Aspect-level Collaborative Filtering model (NeuACF) to exploit different aspect latent factors. Through modelling the rich object properties and relations in recommender system as a heterogeneous information network, NeuACF first extracts different aspect-level similarity matrices of users and items, respectively, through different meta-paths, and then feeds an elaborately designed deep neural network with these matrices to learn aspect-level latent factors. Finally, the aspect-level latent factors are fused for the top-N recommendation. Moreover, to fuse information from different aspects more effectively, we further propose NeuACF++ to fuse aspect-level latent factors with self-attention mechanism. Extensive experiments on three real world datasets show that NeuACF and NeuACF++ significantly outperform both existing latent factor models and recent neural network models.

Index Terms—Recommender systems, heterogeneous information network, aspect-level latent factor

1 Introduction

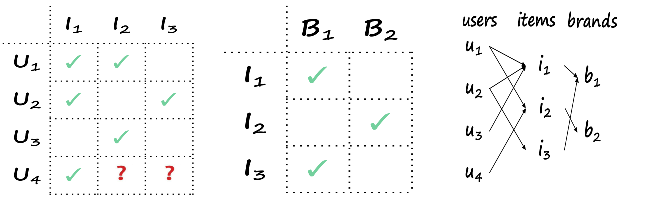
Currently the overloaded online information overwhelms users. In order to tackle the problem, Recommender Systems (RS) are widely employed to guide users in a personalized way of discovering products or services they might be interested from a large number of possible alternatives. Recommender systems are essential for e-commerce companies to provide users a personalized recommendation of products, and thus most e-commerce companies like Amazon and Alibaba are in an urgent need to build more effective recommender systems to improve user experience. Due to its importance in practice, recommender systems have been attracting remarkable attention to both industry and academic research community.

Collaborative Filtering (CF) [1] is one of the most popular methods for recommendation, whose basic assumption is that people who share similar purchase in the past tend to have similar choices in the future. In order to exploit users' similar purchase preference, latent factor models (e.g., matrix factorization) [2], [3] have been proposed, which usually factorize the user-item interaction matrix (e.g., rating matrix) into two low-rank user-specific and item-specific factors, and then use the low-rank factors to make predictions. Since latent factor models may suffer from data sparsity, many extended latent factor models integrate auxiliary information into the matrix factorization framework, such as social recommendation [4] and heterogeneous network based recommendation [5]. Recently, with the surge of deep learning, deep neural networks are also employed to deeply capture the latent features of users and items for recommendation. NeuMF [6] replaces the inner product operations in matrix factorization with a multi-layer feed-forward neural network to capture the non-linear relationship between users and items. DMF [7] uses the rating matrix directly as the input and maps user and items into a common low-dimensional space via a deep neural network.

Although these latent factor models achieve good performance, they usually only capture the information of users' purchase history. Existing models usually focus on extracting latent factors of users and items through their

- *C. Shi, X. Han, L. Song, X. Wang, and J. Du are with the School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China. E-mail: {shichuan, xiaowang}@bupt.edu.cn, {hanxiaotian.h, song200626}@gmail.com, , junpingdu@126.com.*
- *S. Wang is with the Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu 210016, China. E-mail: szwang@nuaa.edu.cn.*
- *P. S. Yu is with the University of Illinois at Chicago, Institute for Data Science, Tsinghua University, Beijing 100084, China. E-mail: psyu@cs.uic.edu.*

Manuscript received 15 Nov. 2018; revised 21 July 2019; accepted 1 Sept. 2019. Date of publication 0 . 0000; date of current version 0 . 0000.
(Corresponding author: Xiao Wang.)
Recommended for acceptance by M. Wang.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2019.2941938



(a) User-Item purchase relation (b) Item-Brand relation (c) heterogeneous network

Fig. 1. A toy example of aspect-level interactions between users and items.

interaction information from ratings, which only reflects user preferences and item characteristics from one aspect, i.e., purchase history. However, the latent factors of users and items usually stem from different aspects in real applications. Particularly, in social media with rich information, user preferences and item characteristics may reflect in many aspects besides rating interactions, e.g., item features, and other interactions between users. These aspect-level features can more comprehensively reflect user preferences and item characteristics. Thus it is very valuable for the latent factor models to exploit latent features of users and items from different aspects. Fig. 1 shows a toy example of our idea. A green check mark indicates that the user purchased the corresponding item in the past. A question mark means that the interaction information is unknown. If we only exploit the interaction matrix (illustrating purchase history) in Fig. 1a, we may infer that user U_4 will purchase item I_2 and I_3. However, when considering the item brand information shown in Fig. 1b, we may find item I_3 is a better recommendation to U_4 because items I_1 and I_3 belong to the same brand B_1.

Although it is promising to comprehensively utilize multiple aspect-level latent features of users and items, it still faces the following two challenges. (1) How to extract different aspect-level features: A systematic method is needed to effectively organize the different types of objects and interactions in recommender systems, and extract different aspect-level features. The extracted aspect-level features should reflect different aspects of users preferences and embody rich semantics. (2) How to learn latent factors from different aspects. Even if we can extract different aspect-level features, it is still not easy to learn their latent factors. Matrix factorization may not be a good option as it only learns the shallow factors. Deep neural network (DNN), which is able to learn the highly nonlinear representations of users and items, is a promising method. However, the current DNN structure lacks of feature fusing mechanism, which cannot be directly applied to our problem. (3) How to fuse latent factors from different aspects effectively. Since the different aspect-level factors only represent aspect-level characteristics of user/item, we need to fuse them effectively. Although deep neural network is a promising method, we still need to design a proper neural network structure and a feature fusing mechanism for our problem settings.

In this paper, to address the challenges above, we propose a novel Neural network based Aspect-level Collaborative Filtering model (NeuACF). NeuACF can effectively model and fuse different aspect-level latent factors which represent the user preferences and item characteristics from different aspects. Particularly, the objects and interactions of different types in recommender systems are first organized as a Heterogeneous Information Network (HIN) [8]. Meta-paths [9], relation sequences connecting objects, are then employed to extract aspect-level features of users and items. As an example shown in Fig. 1c, we can extract the latent factors of users from the aspect of purchase history with the *User-Item-User* path, which is usually analyzed by existing latent factor models. Meanwhile, we can also extract the latent factors from the aspect of brand preference with the *User-Item-Brand-Item-User* path. Furthermore, we design a delicate deep neural network to learn different aspect-level latent factors for users and items and utilize an attention mechanism to effectively fuse them for the top-N recommendation. Note that, different from those hybrid recommendation models [10] that focus on the rating information with the auxiliary information, NeuACF treats different aspect-level latent factors extracted from meta-paths equally, and automatically determines the importance of these aspects. NeuACF is also different from those HIN based methods [11] in its deep model and fusing mechanism. Concretely, a delicately designed attention network is used to fuse aspect-level latent factors. Comparing to the above attention method, we further propose NeuACF++ to fuse aspect information with self-attention mechanism which considers different aspect-level latent factors and learns the attention values simultaneously. Extensive experiments illustrate the effectiveness of NeuACF and NeuACF++, as well as the traits of aspect-level latent factors.

Our main contributions of this paper are summarized as follows.

- To leverage the different aspect-level information of HIN, we design a meta-path based method to capture the aspect-level latent factors of users and items from the similarity matrix obtained from the HIN.
- We propose the NeuACF with deep neural network to learn different aspect-level latent factors and integrate these latent factors with attention mechanism for top-N recommendation, since aspect-level information reflects the characteristics of users and the properties of items more precisely. Moreover, the self-attention mechanism is employed to fuse aspect-level latent factors in our proposed method NeuACF++.
- We preform extensive experiments and provide tremendous analysis to illustrate the effectiveness of NeuACF and NeuACF++.

The rest of this paper is organized as follows. Section 2 reviews the related work. Section 3 summarizes the related work. Section 4 introduces the NeuACF model and NeuACF++ model in details. Section 5 presents and analyzes the experimental results. And Section 6 concludes this paper.

2 Related Work

In this section, we provide a background to our work, and review the relevant works.

2.1 Collaborative Filtering

Traditional recommendation works mainly adopt collaborative filtering (CF) methods to utilize historical interactions for recommendation [3], [12], [13], [14]. As the most popular approach among various CF techniques, matrix factorization (MF) has shown its effectiveness and efficiency in many applications [2], [15]. MF factorizes the user-item interaction matrix into two low-dimension user-specific and item-specific matrices, and then utilizes the factorized matrices for predictions [16]. In recent years, many variants of MF, such as SVD [3], weighted regularized matrix factorization [1], and probabilistic matrix factorization [17] have been proposed. SVD reconstructs the rating matrix only through the observed user-item interactions. Weighted regularized matrix factorization (WR-MF) extends MF by using regularization to prevent over-fitting and to increase the impact of positive feedback. Probabilistic matrix factorization (PMF) models the user preference matrix as a product of two lower-rank user and item matrices. The user and item feature vectors are computed by a probabilistic linear model with Gaussian observation distribution. Bayesian personalized ranking (BPR) [18] is a generic optimization criterion and learning algorithm for implicit CF and has been widespreadly adopted in many related domains [19], [20], [21], [22].

2.2 Neural Networks for Recommendation

Recently, neural network has shown its potential in non-linear transformations and been successfully applied in many data mining tasks [23], [24]. The neural network has been proven to be capable of approximating any continuous function [25]. The pioneer work proposes a two-layers Restricted Boltzmann Machines (RBMs) to model user-item interactions [26]. In addition, autoencoders have been applied to learn user and item vectors for recommendation systems [27], [28], [29]. To overcome the limitation of autoencoders and increase the generalization ability, denoising autoencoders (DAE) have been applied to learn user and item vectors from intentionally corrupted inputs [27], [29]. Cheng et al. [30] combine the benefits of memorization and generalization for recommender systems by jointly training wide linear models and deep neural networks. Compared to Wide & Deep model, Guo et al. [31] propose the DeepFM model that integrates the architectures of factorization machine (FM) and deep neural networks (DNN). This architecture models low-order feature interactions and high-order feature interactions simultaneously. He et al. [6] present a neural network architecture to model latent features of users and items and devise a general neural collaborative filtering (NCF) framework based on neural networks. In addition, NCF leverages a multi-layer perceptron to learn the user-item interaction function instead of the traditional inner product. He et al. [32] propose the neural factorization machine (NFM) model for recommendation. This model combines the linearity of FM in modeling second-order feature interactions and the non-linearity of neural network to model higher-order feature interactions. Xue et al. [7] propose a deep matrix factorization model (DMF) with a neural network that maps the users and items into a common low-dimensional space with non-linear projections. The training matrix includes both explicit ratings and non-preference implicit feedback. The recently proposed convolutional NCF [33] utilizes outer product above the embedding layer results and 2D convolution layers for learning joint representation of user-item pairs.

2.3 Exploiting Heterogeneous Information for Recommendation

To overcome the sparsity of the ratings, additional data are integrated into recommendation systems, such as social matrix factorization with social relations [4] and topicMF with item contents or reviews text [34]. Recently, graph data[35] shows its strong potential for many data mining tasks. There are also many works exploring the graph data for recommendation [36], [37] or web search [38]. As one of the most important methods to model the graph data, heterogeneous information network [8] can naturally characterize the different relations between different types and objects. Then several path based similarity measures are proposed to evaluate the similarity of objects in heterogeneous information network [9], [39], [40]. After that, many HIN based recommendation methods have been proposed to integrate auxiliary information. Feng et al. [41] propose a method to learn the weights of different types of nodes and edges, which can alleviate the cold start problem by utilizing heterogeneous information contained in social tagging system. Furthermore, meta-path is applied to recommender systems to integrate different semantic information [42]. In order to take advantage of the heterogeneity of relationship in information networks, Yu et al. [43] propose to diffuse user preferences along different meta-paths in information networks. Luo et al. [44] demonstrate that multiple types of relations in heterogeneous social network can mitigate the data sparsity and cold start problems. Shi et al. [36] design a novel SemRec method to integrate all kinds of information contained in recommender system using weighted HIN and meta-paths. Zhang et al. [37] propose a joint representation learning (JRL) framework for top-N recommendation by integrating different latent representations.

Most existing latent factor models mainly utilize the rating information between users and items, but ignore the aspect information of users and items. In this paper, we extract different aspect similarity matrices through different meta-paths which characterize the specific aspect information. Then, we delicately design a deep neural network to learn the latent factors of users and items. After that, we utilize attention mechanism to fuse those aspect-level latent factors for top-N recommendation.

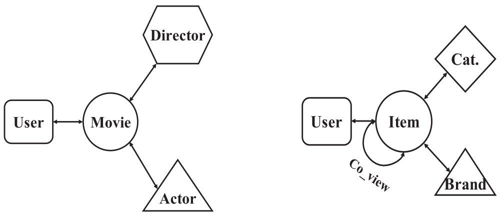
3 Preliminaries

3.1 Latent Factor Model

The latent factor model has been widely studied in recommender systems. Its basic idea is to map users and items to latent factors and use these factors for recommendation. The representative works are Matrix Factorization (MF) [2], PMF [17] and SVD++ [3]. Taking MF for example, the objective function of MF in Equation (1) aims to minimize the following regularized squared loss on the observed ratings:

$$\arg\min_{\boldsymbol{u},\boldsymbol{v}} \sum_i \sum_j (R_{i,j} - \boldsymbol{u}_i^T \boldsymbol{v}_j)^2 + \lambda \left(\sum_i ||\boldsymbol{u}_i||_2^2 + \sum_j ||\boldsymbol{v}_j||_2^2 \right), \tag{1}$$

where $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$ denote the latent factors of user U_i and item I_j, $R_{i,j}$ denote the user U_i rating score to item I_j and the λ



(a) MovieLens (b) Amazon

Fig. 2. Network schema of HINs for the experimental datasets.

controls the strength of regularization, which is usually a L-2 norm aiming to prevent overfitting.

Based on this basic MF framework, many extended latent factor models have been proposed through adding some auxiliary information, such as social recommendation [4] and heterogeneous network based recommendation [36]. The limitation of existing latent factor models is that the latent factors are mainly extracted from one aspect, i.e., the rating matrix. However, some other more fine-grained aspect-level user-item interaction information is largely ignored, although such information is also useful.

3.2 Heterogeneous Information Network

The recently emerging HIN [8] is a good way to model complex relations among different types and objects in recommender systems. Particularly, HIN is a special kind of information network, which either contains multiple types of objects or multiple types of links. The network schema of a HIN specifies the type constraints on the sets of objects and relations among the objects. Two examples used in our experiments are shown in Fig. 2. In addition, meta-path [9], a relation sequence connecting objects, can effectively extract features of objects and embody rich semantics. In Fig. 2b, the meta-path *User-Item-User* (*UIU*) extracts the features of users in the purchase history aspect, which means users having the same purchase records. While the *User-Item-Brand-Item-User* (*UIBIU*) extracts the features of users in the brand aspect, which means users purchase the items with the same brand. In the following section, we use the abbreviation to represent the meta-paths. HIN has been widely used in many data mining tasks [8]. HIN based recommendations also have been proposed to utilize rich heterogeneous information in recommender systems, while they usually focus on rating prediction with the "shallow" model [5], [11].

4 The Proposed Model

4.1 Model Framework

The basic idea of NeuACF is to extract different aspect-level latent features for users and items, and then learn and fuse these latent factors with deep neural network. The model contains three major steps. First, we construct an HIN based on the rich user-item interaction information in recommender systems, and compute the aspect-level similarity matrices under different meta-paths of HIN which reflect different aspect-level features of users and items. Next, a deep neural network is designed to learn the aspect-level latent factors separately by taking these similarity matrices as inputs. Finally, the aspect-level latent factors are combined with an attention component to obtain the overall latent factors for users and items. Moreover, we also employ self-attention mechanism to fuse aspect-level latent factors more effectively. Next we will elaborate the three steps in the following subsections.

TABLE 1
Meta-Paths used in Experiments and the Corresponding Aspects

Datasets	Aspect	Meta-Paths	
		User	Movie/Item
MovieLens	History	UMU	MUM
	Director	$UMDMU$	MDM
	Actor	$UMAMU$	MAM
Amazon	History	UIU	IUI
	Brand	$UIBIU$	IBI
	Category	$UICIU$	ICI
	Co_view	$UIVIU$	IVI

4.2 Aspect-Level Similarity Matrix Extraction

We employ HIN to organize objects and relations in recommender systems, due to its power of information fusion and semantics representation [36]. Furthermore, we utilize meta-path to extract different-aspect features of users and items. Taking Fig. 2b as an example, we can use *UIU* and *IUI* paths to extract features of users and items on the aspect of purchase history, which is extensively exploited by existing latent factor models. In addition, we can also extract features from other aspects. For example, the features of the brand aspect can be extracted from *UIBIU* and *IBI* paths . Table 1 shows more aspect examples in our experimental datasets.

Given a specific meta-path, there are several alternatives to extract the aspect-level features: commuting matrix or similarity matrix. In this paper, we employ the similarity matrix based on the following reasons. (1) Similarity measure can alleviate noisy information; (2) Similar values within the [0,1] range are more suitable for learning latent factors; (3) Many path based similarity measures are available. We employ the popular PathSim [9] to calculate aspect-level similarity matrices under different meta-paths in experiments. For example, we compute the similarity matrices of user-user and item-item based on the meta-path *UIBIU* and *IBI* for the brand-aspect features.

The computation of similarity matrix based on meta path is of great importance in our propose model, so how to compute similarity matrix quickly is an important problem in our method. In real-word application, the complexity of similarity matrix computation is not high because the similarity matrix is usually very sparse for most meta paths. Based on this fact, there are several acceleration computation methods proposed by previous works [9], [40] for similarity matrix computation, for example, PathSim-pruning [9], dynamic programming strategy and Monte Carlo (MC) strategy [40]. Moreover there also many new methods for similarity matrix computation, for example, BLPMC [45], PRSim [46]. In addition, the similarity matrix can be computed offline in advance in our model. The similarity matrix is computed with training data, so we can prepare the similarity matrix before the training processing.

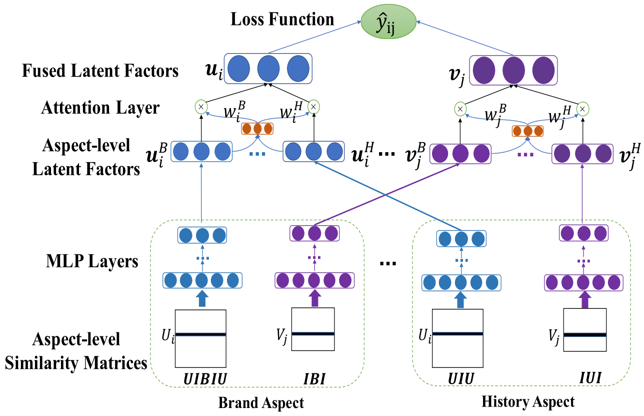


Fig. 3. Deep neural network in the NeuACF model.

4.3 Learning Aspect-Level Latent Factors

With the computed user-user and item-item similarity matrices of different aspects, we next learn their latent factors. Different from previous HIN based recommendation models, we design a deep neural network to learn their corresponding aspect-level latent factors separately, and the model architecture is shown in Fig. 3. Concretely, for each user in each aspect, we extract the user's similarity vector from the aspect-specific similarity matrix. Then we take the similarity matrix as the input of the Multi-Layer Perceptron (MLP) and MLP learns the aspect-level latent factor as the output. The item latent factors of each aspect can be learned in a similar way. Taking the similarity matrix $S^B \in \mathbb{R}^{N\times N}$ of users under the meta-path *UIBIU* as an example, User U_i is represented as an N-dimensional vector S^B_{i*}, which means the similarities between U_i and all the other users. Here N means the total number of users in the dataset. The MLP projects the initial similarity vector S^B_{i*} of user U_i to a low-dimensional aspect-level latent factor. In each layer of MLP, the input vector is mapped into another vector in a new space. Formally, given the initial input vector S^B_{i*}, and the lth hidden layer H_l , the final aspect-level latent factor $\boldsymbol{u}^B_i$ can be learned through the following multi-layer mapping functions,

$$
\begin{aligned}
H_0 &= S^B_{i*},\\
H_1 &= f(\boldsymbol{W}_1^T * H_0 + \boldsymbol{b}_1),\\
&\dots\\
H_l &= f(\boldsymbol{W}_l^T * H_{l-1} + \boldsymbol{b}_l),\\
&\dots\\
\boldsymbol{u}^B_i &= f(\boldsymbol{W}_n^T * H_{n-1} + \boldsymbol{b}_n),
\end{aligned}
\tag{2}
$$

where $\boldsymbol{W}_i$ and $\boldsymbol{b}_i$ are the weight matrix and bias for the ith layer, respectively, and we use the *ReLU*, i.e., $f(x) = max(0, x)$ as the activation function in the hidden layers.

From the learning framework in Fig. 3, one can see that for each aspect-level similarity matrix of both users and items there is a corresponding MLP learning component described above to learn the aspect-level latent factors. As illustrated in Table 1, for each aspect-level meta-path we can get a corresponding user-user similarity matrix and an item-item similarity matrix. Taking the datasets Amazon as example, we can learn the brand latent factors of users as $\boldsymbol{u}^B_i$ and the the brand latent factors of items as v^B_j from the meta-path *UIBIU-IBI*. Similarly, we can get $\boldsymbol{u}^I_i$ and v^I_j from the meta-path *UIU-IUI*, $\boldsymbol{u}^C_i$ and v^C_j from the meta-path *UICIU-ICI*, as well as $\boldsymbol{u}^V_i$ and v^V_j from the meta-path *UIVIU-IVI*. Since there are variety meta-paths connecting users and items, we can learning different aspect-level latent factors.

4.4 Attention Based Aspect-Level Latent Factors Fusion

After the aspect-level latent factors are learned separately for users and items, next we need to integrate them together to obtain aggregated latent factors. A straightforward way is to concatenate all the aspect-level latent factors to form a higher-dimensional vector. Another intuitive way is to average all the latent factors. The issue is that both methods do not distinguish their different importance because not all the aspects contribute to the recommendation equally (we will show that in the experiment part). Therefore, we choose the attention mechanism to fuse these aspect-level latent factors. Attention mechanism has shown the effectiveness in various machine learning tasks such as image captioning and machine translation [47], [48], [49]. The advantage of attention mechanism is that it can learn to assign attentive values (normalized by sum to 1) for all the aspect-level latent factors: higher (lower) values indicate that the corresponding features are more informative (less informative) for recommendation. Specifically, given the user's brand-aspect latent factor $\boldsymbol{u}^B_i$, we use a two-layers network to compute the attention score $\boldsymbol{s}^B_i$ by the following

$$
\boldsymbol{s}^B_i = \boldsymbol{W}_2^T f(\boldsymbol{W}_1^T * \boldsymbol{u}^B_i + \boldsymbol{b}_1) + \boldsymbol{b}_2, \tag{3}
$$

where $\boldsymbol{W}_*$ is the weight matrices and $\boldsymbol{b}_*$ is the biases.

The final attention values for the aspect-level latent factors are obtained by normalizing the above attentive scores with the Softmax function given in Equation (4), which can be interpreted as the contributions of different aspects B to the aggregated latent factor of user U_i,

$$
\boldsymbol{w}^B_i = \frac{exp(\boldsymbol{s}^B_i)}{\sum_{A=1}^{L} exp(\boldsymbol{s}^A_i)}, \tag{4}
$$

where L is the total number of all the aspects.

After obtaining all the attention weights $\boldsymbol{w}^B_i$ of all the aspect-level latent factors for user U_i, the aggregated latent factor $\boldsymbol{u}_i$ can be calculated by

$$
\boldsymbol{u}_i = \sum_{B=1}^{L} \boldsymbol{w}^B_i \cdot \boldsymbol{u}^B_i. \tag{5}
$$

We implement this attention method as NeuACF in our experiments.

4.5 Self-Attention Based Aspect-Level Latent Factors Fusion

Recently, self-attention mechanism has received considerable research interests. For example, Vaswani et al. [50] and Devlin et al. [51] utilize self-attention to learn the relationship between two sequences. Learning dependencies and relationships between aspect-level latent factors is the most important part in our model, and self-attention has ability to model the relationships between the different aspect-level latent factors.

Different from standard attention mechanism, self-attention mainly focuses on the co-learning attentions of two sequences. The vanilla attention mechanism mainly considers computing the attention values based on the user or item representations of one aspect, while self-attention mechanism is able to learn the attention values from different aspects simultaneously. For example, the Brand-level latent factor of users have strong relationship to the Brand-level latent factor of items, and the self-attention mechanism can learn this relationship and promote the performance of recommendation. So the learned values are able to capture more information on the multi-aspects. In details, we first compute the affinity scores between all aspect-level latent factors. For a user U_i, the affinity score of two different aspect-level latent factors $\boldsymbol{u}_i^B$ and $\boldsymbol{u}_i^C$ can be calculated by their inner product:

$$M_i^{B,C} = (\boldsymbol{u}_i^B)^T * \boldsymbol{u}_i^C. \tag{6}$$

The matrix $\boldsymbol{M}_i = [M_i^{B,C}] \in \mathbb{R}^{L \times L}$ is also called the self-attention matrix, where L is the total number of aspects. In fact, there is an affinity matrix $\boldsymbol{M}_i$ for each user. Basically, the matrix $\boldsymbol{M_i}$ characterizes the similarity of aspect-level latent factors for the specific user U_i, which reflects the correlation between two aspects when recommending for this user. When the aspect B is equal to aspect C, $M_i^{B,C}$ will get a high value due to the inner product operator, so we add a zero mask to avoid a high matching score between identical vectors.

The aspect-level latent factors learned from self-attention mechanism are not independent. Users will make a trade-off between those aspects. The affinity matrix measures the importance of different aspect-level latent factors, so we compute the representation of aspect B for the specific user i based on the self-attention matrix as:

$$\boldsymbol{g}_i^B = \sum_{C=1}^{L} \frac{exp(\boldsymbol{M}_i^{B,C})}{\sum_{A=1}^{L} exp(\boldsymbol{M}_i^{B,A})} \boldsymbol{u}_i^C. \tag{7}$$

Then for all the aspects, we can obtain the final representation of users or items as:

$$\boldsymbol{u}_i = \sum_{B=1}^{L} \boldsymbol{g}_i^B. \tag{8}$$

The self-attention mechanism can learn self-attentive representations from different aspect-level information effectively. In order to distinguish with the above attention method NeuACF, we implement the self-attention mechanism as NeuACF++ in our experiments.

4.6 Objective Function

We model the top-N recommendation as a classification problem which predicts the probability of interaction between users and items in the future. In order to ensure that the output value is a probability, we need to constrain the output $\hat{y}_{ij}$ in the range of [0,1], where we use a Logistic function as the activation function for the output layer. The probability of the interaction between the user U_i and item I_j is calculated according to

$$\hat{y}_{ij} = sigmod(\boldsymbol{u}_i * \boldsymbol{v}_j) = \frac{1}{1 + e^{-\boldsymbol{u}_i * \boldsymbol{v}_j}}, \tag{9}$$

where $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$ are the aggregated latent factors of user U_i and item I_j respectively.

Over all the training set, according to the above settings, the likelihood function is:

$$p(\mathcal{Y}, \mathcal{Y}^- | \Theta) = \prod_{i,j \in \mathcal{Y}} \hat{y}_{ij} \prod_{i,k \in \mathcal{Y}^-} (1 - \hat{y}_{ik}), \tag{10}$$

where $\mathcal{Y}$ and $\mathcal{Y}^-$ are the positive and negative instances sets, respectively. The negative instance set $\mathcal{Y}^-$ is sampled from unobserved data for training. Θ is the parameters set.

Since the ground truth y_{ij} is in the set $\{0, 1\}$, Equation (10) can be rewritten as:

$$p(\mathcal{Y}, \mathcal{Y}^- | \Theta) = \prod_{i,j \in \mathcal{Y} \cup \mathcal{Y}^-} (\hat{y}_{ij})^{y_{ij}} * (1 - \hat{y}_{ij})^{(1 - y_{ij})}. \tag{11}$$

Then we take the negative logarithm of the likelihood function to get the point-wise loss function in

$$Loss = - \sum_{i,j \in \mathcal{Y} \cup \mathcal{Y}^-} \big(y_{ij} log \hat{y}_{ij} + (1 - y_{ij}) log(1 - \hat{y}_{ij})\big), \tag{12}$$

where y_{ij} is the ground truth of the instance and $\hat{y}_{ij}$ is predicted score. This is the overall objective function of our model, and we can optimize it by stochastic gradient descent or its variants [52].

4.7 Discussion

Here, we give the analysis of our proposed models NeuACF and NeuACF++.

- NeuACF and NeuACF++ are general frameworks for recommendation. We can learn aspect-level latent factors from aspect-level features computed via different methods. For example, the similarity matrix S^B can also be computed with HeteSim [40] or PCRW [39].
- As a deep neural network model, DMF [53] can be considered as one special case of our model. DMF does not take the heterogeneous information into consideration, so if we only consider the user-item purchase history aspect, our model is equivalent to the DMF model. We argue that the aspect information learned from meta-paths has potential to increase the performance of recommendation.
- We present the time complexity analysis of our proposed models NeuACF and NeuACF++ here. Generally, the time complexity is affected by the epochs of iterator T, the size of training sample S, the number of aspects L and the size of hidden numbers H. When we utilize three-layer MLP to learn user and item latent factors in our models, the time complexity of forward and backward process is bounded by matrix multiplication. Let h_{n_1} be the number of input neurons and h_{n_2} be the number of output neurons, the time complexity of forward process can be

TABLE 2
The Statistics of the Datasets

Dataset	#users	#items	#ratings	#density
ML100K	943	1682	100,000	6.304%
ML1M	6040	3706	1,000,209	4.468%
Amazon	3532	3105	57,104	0.521%

calculated as $O(h_{n_1} * H + H * h_{n_2})$. The attention layer is a two-layer neural network with the number of input size equal to h_{n_2} and the number of output size is 1. The time consumption is negligible comparing to the embedding layers. Therefore, the overall time complexity for training process is $O(STL(h_{n_1} * H + H * h_{n_2}))$. For the prediction process, supposing the number of negative sampling for one user is N_s, the time complexity of prediction is $O(N_s L(h_{n_1} * H + H * h_{n_2}))$.

5 Experiments

5.1 Experimental Settings

5.1.1 Datasets

We evaluate the proposed model over the publicly available MovieLens dataset [54] and Amazon dataset [55], [56]. We use the origin Movielens dataset for our experiment. For Amazon dataset, we remove the users who buy less than 10 items. The network schema is shown in Fig. 2, and the statistics of the datasets are summarized in Table 2.

- MovieLens-100K(ML100k)/MovieLens-1M(ML1M) [1]: MovieLens datasets have been widely used for movie recommendation. We use the versions ML100K and ML1M. For each movie, we crawl the directors, actors of the movie from IMDb.
- Amazon[2]: This dataset contains users' rating data in Amazon. In our experiment, we select the items of Electronics categories for evaluation.

5.1.2 Evaluation Metric

We adopt the leave-one-out method [6], [7] for evaluation. The latest rated item of each user is held out for testing, and the remaining data for training. Following previous works [6], [7], we randomly select 99 items that are not rated by the users as negative samples and rank the 100 sampled items for the users. For a fair comparison with the baseline methods, we use the same negative sample set for each (*user, item*) pair in the test set for all the methods. We evaluate the model performance through the Hit Ratio (HR) and the Normalized Discounted Cumulative Gain (NDCG) defined in

$$HR = \frac{\#hits}{\#users}, NDCG = \frac{1}{\#users} \sum_{i=1}^{\#users} \frac{1}{log_2(p_i + 1)}, \tag{13}$$

where $\#hits$ is the number of users whose test item appears in the recommended list and p_i is the position of the test item in the list for the ith hit. In our experiments, we truncate the ranked list at $K \in [5, 10, 15, 20]$ for both metrics.

1. https://grouplens.org/datasets/movielens/
2. http://jmcauley.ucsd.edu/data/amazon/

5.1.3 Baselines

Besides two basic methods (i.e., ItemPop and ItemKNNN [57]), the baselines include two MF methods (MF [2] and eALS [13]), one pairwise ranking method (BPR [18]), and two neural network based methods (DMF [7] and NeuMF [6]). In addition, we use $\mathrm{SVD_{hin}}$ to leverage the heterogeneous information for recomendation, and we also adopt two recent HIN based methods (FMG
[11] and HeteRs [58]) as baselines.

- ItemPop. Items are simply ranked by their popularity judged by the number of interactions. This is a widely-used non-personalized method to benchmark the recommendation performance.
- ItemKNN [57]. It is a standard item-based collaborative filtering method.
- MF [2]. Matrix factorization is a representative latent factor model.
- eALS [13]. It is a state-of-the-art MF method for recommendation with the square loss.
- BPR [18]. The Bayesian Personalized Ranking approach optimizes the MF model with a pairwise ranking loss, which is tailored to learn from implicit feedback.
- DMF [7]. DMF uses the interaction matrix as the input and maps users and items into a common low-dimensional space using a deep neural network.
- NeuMF [6]. It combines the linearity of MF and non-linearity of DNNs for modelling user–item latent structures. In our experiments, we use the NeuMF with pre-trained, We used hyper-parameters followed the instructions in the paper.
- $\mathrm{SVD_{hin}}$. SVDFeature [59] is designed to efficiently solve the feature-based matrix factorization. $\mathrm{SVD_{hin}}$ uses SVDFeature to leverage the heterogeneous information for recommendation. Specifically, we extract the heterogeneous information (e.g., attributes of movies/items and profiles of users) as the input of SVDFeature.
- HeteRS [58]. HeteRS is a graph-based model which can solve general recommendation problem on heterogeneous networks. It models the rich information with a heterogeneous graph and considers the recommendation problem as a query-dependent node proximity problem.
- FMG [11]. It proposes "MF+FM" framework for the HIN-based rating prediction. We modify its optimization object as point-wise ranking loss for the top-N recommendation.

5.1.4 Implementation

We implement the proposed NeuACF and NeuACF++ based on Tensorflow [60]. We use the same hyper-parameters for all the datasets. For the neural network, we use a three-layer MLP with each hidden layer having 600 hidden units. The dimension of latent factors is 64. We randomly initialize the model parameters with a xavier initializer [61],

TABLE 3
HR@K and NDCG@K Comparisons of Different Methods

Datasets	Metrics	ItemPop	ItemKNN	MF	eALS	BPR	DMF	NeuMF	SVD_{hin}	HeteRS	FMG	NeuACF	NeuACF++
ML100K	HR@5	0.2831	0.4072	0.4634	0.4698	0.4984	0.3483	0.4942	0.4655	0.3747	0.4602	0.5097	**0.5111**
	NDCG@5	0.1892	0.2667	0.3021	0.3201	0.3315	0.2287	0.3357	0.3012	0.2831	0.3014	0.3505	**0.3519**
	HR@10	0.3998	0.5891	0.6437	0.6638	0.6914	0.4994	0.6766	0.6554	0.5337	0.6373	0.6846	**0.6915**
	NDCG@10	0.2264	0.3283	0.3605	0.3819	0.3933	0.2769	0.3945	0.3988	0.3338	0.3588	0.4068	**0.4092**
	HR@15	0.5366	0.7094	0.7338	0.7529	0.7741	0.5873	0.7635	0.7432	0.6524	0.7338	0.7813	**0.7832**
	NDCG@15	0.2624	0.3576	0.3843	0.4056	0.4149	0.3002	0.4175	0.4043	0.3652	0.3844	0.4318	**0.4324**
	HR@20	0.6225	0.7656	0.8144	0.8155	0.8388	0.6519	0.8324	0.8043	0.7224	0.8006	**0.8464**	0.8441
	NDCG@20	0.2826	0.3708	0.4034	0.4204	0.4302	0.3151	0.4338	0.3944	0.3818	0.4002	**0.4469**	**0.4469**
ML1M	HR@5	0.3088	0.4437	0.5111	0.5353	0.5414	0.4892	0.5485	0.4765	0.3997	0.4732	**0.5630**	0.5584
	NDCG@5	0.2033	0.3012	0.3463	0.3670	0.3756	0.3314	0.3865	0.3098	0.2895	0.3183	**0.3944**	0.3923
	HR@10	0.4553	0.6171	0.6896	0.7055	0.7161	0.6652	0.7177	0.6456	0.5758	0.6528	0.7202	**0.7222**
	NDCG@10	0.2505	0.3572	0.4040	0.4220	0.4321	0.3877	0.4415	0.3665	0.3461	0.3767	0.4453	**0.4454**
	HR@15	0.5568	0.7118	0.7783	0.7914	0.7988	0.7649	0.7982	0.7689	0.6846	0.7536	0.8018	**0.8030**
	NDCG@15	0.2773	0.3822	0.4275	0.4448	0.4541	0.4143	0.4628	0.4003	0.3749	0.4034	**0.4667**	0.4658
	HR@20	0.6409	0.7773	0.8425	0.8409	0.8545	0.8305	0.8586	0.8234	0.7682	0.8169	0.8540	**0.8601**
	NDCG@20	0.2971	0.3977	0.4427	0.4565	0.4673	0.4296	0.4771	0.4456	0.3947	0.4184	0.4789	**0.4790**
Amazon	HR@5	0.2412	0.1897	0.3027	0.3063	0.3296	0.2693	0.3117	0.3055	0.2766	0.3216	0.3268	**0.3429**
	NDCG@5	0.1642	0.1279	0.2068	0.2049	0.2254	0.1848	0.2141	0.1922	0.1800	0.2168	0.2232	**0.2308**
	HR@10	0.3576	0.3126	0.4278	0.4287	0.4657	0.3715	0.4309	0.4123	0.4207	0.4539	0.4686	**0.4933**
	NDCG@10	0.2016	0.1672	0.2471	0.2441	0.2693	0.2179	0.2524	0.2346	0.2267	0.2595	0.2683	**0.2792**
	HR@15	0.4408	0.3901	0.5054	0.5065	0.5467	0.4328	0.5258	0.5056	0.5136	0.5430	0.5591	**0.5948**
	NDCG@15	0.2236	0.1877	0.2676	0.2647	0.2908	0.2332	0.2774	0.2768	0.2513	0.2831	0.2924	**0.3060**
	HR@20	0.4997	0.4431	0.5680	0.5702	0.6141	0.4850	0.5897	0.5607	0.5852	0.6076	0.6257	**0.6702**
	NDCG@20	0.2375	0.2002	0.2824	0.2797	0.3067	0.2458	0.2925	0.2876	0.2683	0.2983	0.3080	**0.3236**

and use the Adam [52] as the optimizer. We set the batch size to 1024 and set the learning rate to 0.0005. When training our model, 10 negative instances are sampled for each positive instance. Table 1 illustrates the extracted aspects and corresponding meta-paths. Some meta-paths are also used for FMG. The optimal parameters for baselines are set according to literatures. All the experiments are conducted on a machine with two GPUs (NVIDIA GTX-1080 *2) and two CPUs (Intel Xeon E5-2690 * 2).

5.2 Experiment Results

5.2.1 Performance Analysis

Table 3 shows the experiment results of different methods. Our proposed methods are marked as NeuACF which implements the attention method in Section 4.4 and NeuACF++ which implements the self-attention mechanism in Section 4.5, respectively. One can draw the following conclusions.

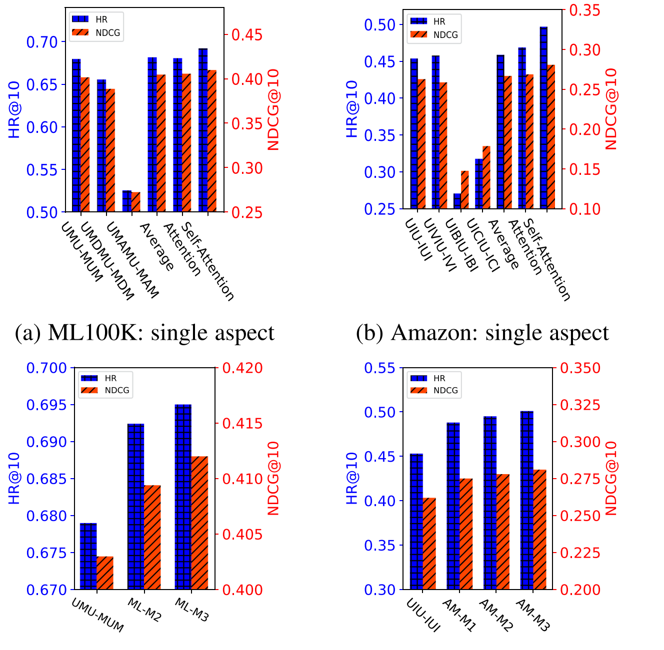
First, one can observe that, NeuACF and NeuACF++ achieve all the best performance over all the datasets and criteria. The improvement of the two models comparing to these baselines is significant. This indicates that the aspect level information is useful for recommendations. Besides, NeuACF++ outperforms the NeuACF method in most circumstances. Particularly, the performance of NeuACF++ is significantly improved in Amazon dataset about ($+2\%$ at HR and $+1\%$ at NDCG). This demonstrates the effectiveness of the self-attention mechanism. Since the affinity matrix evaluates the similarity score of different aspects, we can extract the valuable information from the aspect latent factors.

Second, NeuMF, as one neural network based method, also performs well on most conditions, while both NeuACF and NeuACF++ outperform NeuMF in almost all the cases. The reason is probably that multiple aspects of latent factors learned by NeuACF and NeuACF++ provide more features of users and items. Although FMG also utilizes the same features with NeuACF and NeuACF++, the better performance of NeuACF and NeuACF++ implies that the deep neural network and the attention mechanisms in NeuACF and NeuACF++ may have the better ability to learn latent factors of users and items than the "shadow" model in FMG.

We can also observe that MF based methods outperform the ItemPop and ItemKNN methods. This indicates that the latent factors models can depict the user and item characteristics. Moreover, the performance of NeuMF is better than MF, which indicates that the non-linear projection can capture more information. The performance of BPR is comparable to NeuMF though it does not utilize the non-linear projection. The reason may be that the objective function is prone to tackle those ranking problems.

5.2.2 Impact of Different Aspect-Level Latent Factors

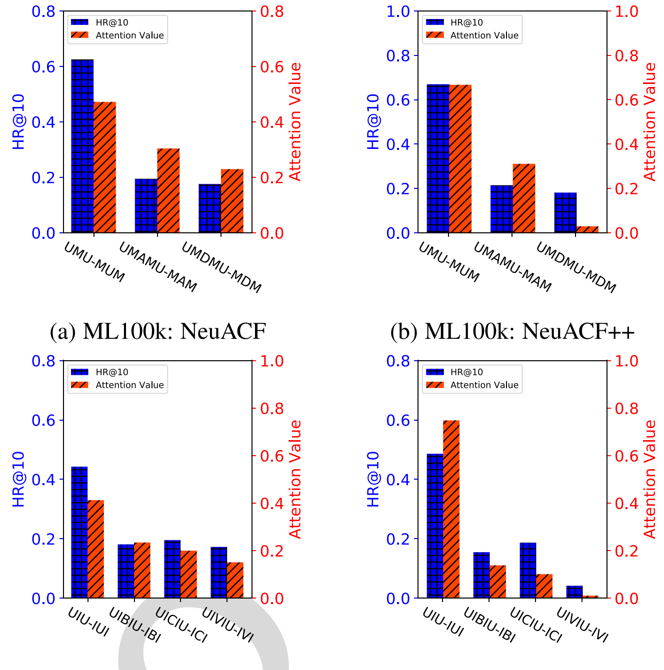
To analyze the impact of different aspect-level latent factors on the algorithm performance, we run NeuACF and NeuACF++ with individual aspect-level latent factor through setting meta-paths. In Fig. 4, for example, *UIBIU-IBI* means that we only learn the brand-aspect latent factor for users and items. In addition, we also run NeuACF with the "Average", "Attention" and "Self-Attention" fusion mechanisms, where "Average" means averaging all the aspect-



(a) ML100K: single aspect (b) Amazon: single aspect

(c) ML-100k: fusing aspects (d) Amazon: fusing aspects

Fig. 4. The impact of different aspect-level latent factors. (a) The performance of single aspect on $MovieLens$. "Attention" means the NeuACF method, and "Self-Attention" means the NeuACF++ method. (b) The performance of single aspect on Amazon dataset. (c) The performance of combination of different meta-paths on ML100k dataset. ML-M2 adds $UMDMU\text{-}MDM$, and ML-M3 adds $UMAMU\text{-}MAM$ to ML-M2. (d) The performance of combination of different meta-paths on Amazon dataset. AM-M1 adds $UIVIU\text{-}IVI$. AM-M2 and AM-M3 add $UIBIU\text{-}IBI$, $UICIU\text{-}ICI$, respectively.



(a) ML100k: NeuACF (b) ML100k: NeuACF++

(c) Amazon: NeuACF (d) Amazon: NeuACF++

Fig. 5. Attention value analysis.

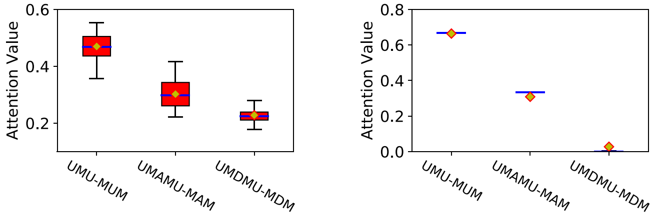
level latent factors, "Attention" means fusing latent factors with the proposed attention mechanism in Section 4.4, and "Self-Attention" means fusing latent factors with the self-attention mechanism mentioned in Section 4.5. From the results shown in Figs. 4a and 4b, one can observe that the purchase-history aspect factors (e.g., $UMU\text{-}MUM$ and $UIU\text{-}IUI$) usually get the best performance in all the individual aspects which indicates that the purchase history of users and items usually contains the most important information. One can also see that "Average", "Attention" and "Self-Attention" always perform better than individual meta-path, demonstrating fusing all the aspect-level latent factors can improve the performance. In addition, the better performance of "Attention" than "Average" also shows the benefit of the attention mechanism in NeuACF. One can also observe that the "Self-Attention" mechanism always perform better than other methods, which indicates that the self-attention mechanism can fuse different aspect information more efficiently.

Further, in order to validate that the additional information from different meta-paths has potential to increase the recommendation performance. We conduct experiments with the increase of meta-paths to fuse more information into our proposed models. The results are shown in Figs. 4c and 4d. It demonstrates that the combination of different meta-paths can increase the performance of recommendation. In particular, ML-M2 means the result of fusing aspect-level latent factors extracted from the meta-paths of $UMU\text{-}MUM$ and $UMAMU\text{-}MAM$. The performance of ML-M2 outperforms the single meta-path $UMU\text{-}MUM$, which is the best result among all the single aspects. ML-M3 means the result of fusing the meta-paths of $UMU\text{-}MUM$, $UMAMU\text{-}MAM$ and $UMDMU\text{-}MDM$. Similarly, the result is better than ML-M2. Moreover, the performance does not improve linearly. Taking the Amazon dataset in Fig. 4d as an example, the meta-path $UIVIU\text{-}IVI$ in AM-M1, comparing to the single meta-path $UIU\text{-}IUI$, provides a large improvement. However, the meta-path $UIBIU\text{-}IBI$ in AM-M2 helps little on the performance. This demonstrates that different aspect-level meta-paths contain unequal information, so it is essential to automatically fuse aspect-level latent factors with attention mechanisms.

5.2.3 Analysis on Attention

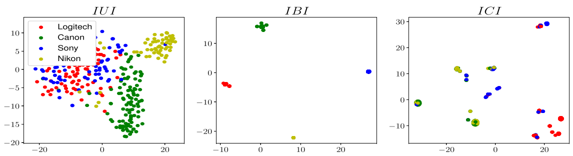
In order to investigate that whether the attention values learned from our proposed models NeuACF and NeuACF++ are meaningful, we explore the correlation between the attention values and the recommendation performance of the corresponding meta-path. Generally, we aim to check whether the recommendation performance with one meta-path will is better when the attention value of this meta-path is larger.

To this end, we conduct experiments to analyze the distribution with attention values and the recommendation performance of single meta-path. Specifically, we can obtain the attention value in each aspect for a user based on NeuACF and NeuACF++, and then we are able to average all the attention values for all the users to obtain the final attention value of the aspect. Also, we can get the recommendation results only based on this aspect. So for one aspect, we are able to check the correlation between its recommendation performance and its attention value. Bascially, the better results usually imply that this aspect is more important to the recommendation task, and therefore, this aspect should have larger attention value. We perform experiments with NeuACF and NeuACF++ models respectively. For example, in ML100k dataset, we can obtain three attention values from three

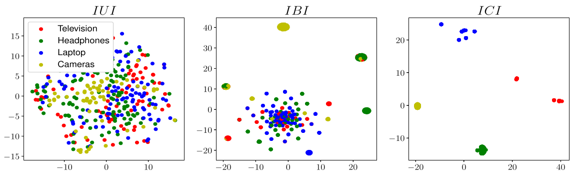


(a) ML100k: NeuACF (b) ML100k: NeuACF++

Fig. 6. The distribution of attention weights of NeuACF and NeuACF++ on the datasets.



(a) t-SNE embedding with brand labels



(b) t-SNE embedding with Category labels

Fig. 7. t-SNE embedding with different labels of the learned latent factors of items for Amazon.

different aspect latent factors *UMU-MUM*, *UMAMU-MAM*, and *UMDMU-MDM* by NeuACF++. We present the result of "Attention Value" and the corresponding single meta-path recommendation results "HR@10" in Fig. 5.

One can observe that the attention values of different aspects vary significantly. If the recommendation performance of one meta-path is higher, the corresponding attention value trends to be larger. Intuitively, this indicates that the aspect information plays a vital role in recommendation, and "Average" is insufficient to fuse different aspect-level latent factors. Another interesting observation is that though the distributions of attention values in different datasets are extremely different, the purchase history (e.g., *UMU-MUM* and *UIU-IUI*) always takes a large proportion. This is consistent with the results in Section 5.2.2, suggesting that purchase history usually contains the most valuable information.

We also present the distribution of attention weights of NeuACF and NeuACF++ on the Movielens dataset in Fig. 6. Fig. 6 indicates that the attention values of different aspects are very different and we can find that attention values of NeuACF++ which adopts self-attention are more stable than NeuACF. The reason of this observation is that the self-attention mechanism is more powerful than vanilla attention network to capture the aspect information and assign more reasonable attention weights to different aspects.

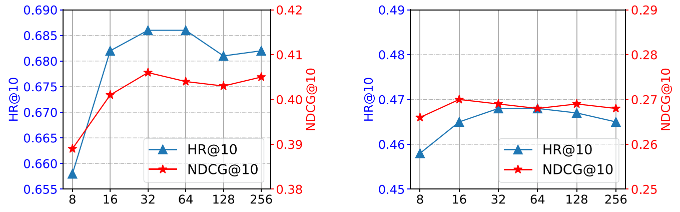
5.2.4 Visualization of Different Aspect-Level Latent Factors

In our model, we aim to learn the aspect-level latent factors from different meta-paths. For example, we expect that the brand-aspect latent factor v_j^B for item I_j can be learned from the meta-path *IBI*, and the category-aspect latent factor v_j^C from the meta-path *ICI*. To intuitively show whether NeuACF performs well on this task, we visualize the learned aspect-level latent factors on the Amazon dataset. We apply t-SNE [62] to embed the high-dimensional aspect-level latent factors into a 2-dimensional space, and then visualize each item as a point in a two-dimensional space.

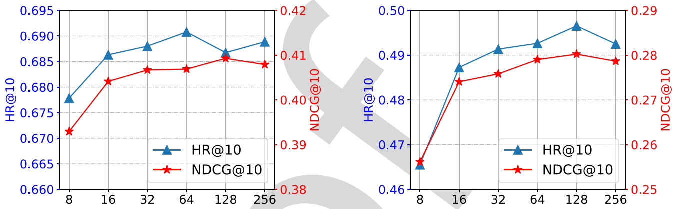
Fig. 7a shows the embedding results for four famous electronics Brand: *Logitech*, *Canon*, *Sony*, and *Nikon*. One can observe that the brand-aspect latent factors can clearly separate the four brands, while the history-aspect and category-aspect latent factors are mixed with each other. It demonstrates the meta-path *IBI* can learn a good brand-aspect latent factors. Similarly, in Fig. 7b, only the category-aspect latent factors learned from the meta-path *ICI* clearly separate the items of different categories including *Television*, *Headphones*, *Laptop* and *Cameras*. The results demonstrate that the aspect-level latent factors of items learned by NeuACF can indeed capture the aspect characteristics of items.

5.2.5 Parameter Study

Effect of the Latent Factor Dimensions. In the latent factor models, the dimension of the latent factors may have a vital impact on the performance of recommendation. Thus we study the effect of the latent factor dimension learned from the last MLP layer in our proposed model NeuACF and NeuACF++. We conduct the experiment on a three-layer model, and set the dimensions of the latent factors increasing from 8 to 256. The results on the ML100k and Amazon datasets are shown in Fig. 8. Figs. 8a and 8b illustrate the performance curve with different numbers of dimensions of NeuACF. One can see that on both datasets the performance first increases with the increase of the dimension, and the best performance is achieved at round 16-32. Then the performance drops if the dimension further increases. Similarly, Figs. 8c and 8d show the results of NeuACF++. We can observe that the best performance of NeuACF++ is achieved at round 64 of ML100K and 128 of Amazon. Generally speaking, a small dimension of latent factors is insufficient to capture the complex relationship of users and items.

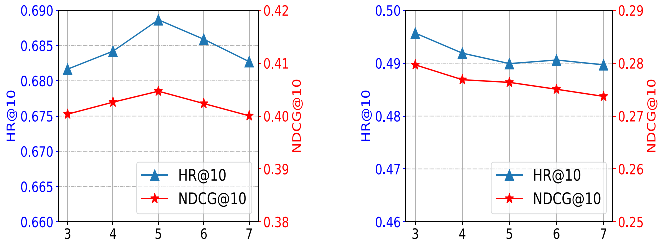


(a) ML100k: NeuACF (b) Amazon: NeuACF



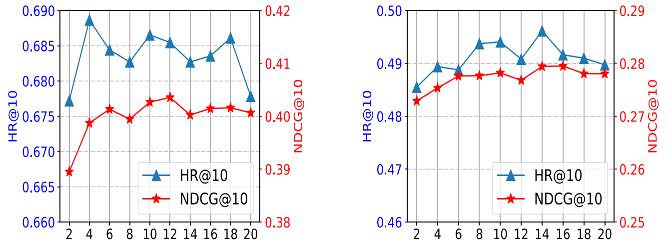
(c) ML100k: NeuACF++ (d) Amazon: NeuACF++

Fig. 8. Performance with different dimensions of latent factors.



(a) ML100k (b) Amazon

Fig. 9. Performance with different numbers of hidden layers.



(a) ML100k (b) Amazon

Fig. 10. Performance with different number of negative samples.

Effect of Network Hidden Layers. As the number of hidden layers can usually affect the performance of deep models, we investigate the effect of the number of network hidden layers on our model NeuACF++. We set the number of hidden layers of NeuACF++ from 3 to 7, and the number of hidden neurons of each layer is set up to 64. The results are illustrated in Fig. 9. As can be seen from Fig. 9a, the performance of ML100k dataset first increases with the increase of hidden layers. The best performance is achieved when hidden layers is 5, and then the performance decreases. The performance of NeuACF++ decreases slightly when hidden layers increase in Amazon dataset. The best performance is achieved when hidden layers is 3. The reason may be that a three-layer neural network model is capable to characterize the aspect latent factors in Amazon dataset. When the number of hidden layers increase, the model may be over-fitting. From both cases, we can find that the best depth of our model is about 3 layers. Moreover, the slightly degradation may also demonstrate that it is hard for the deep model to learn the identity mapping [63].

Effect of Negative Sampling Ratio. As mentioned above, negative sampling is an effective way to train the neural network model instead of using the whole user-item interactions. To illustrate the impact of different negative sampling ratios for NeuACF++ model, we conduct experiments with different negative sampling ratios. The results are shown in Fig. 10. The experiments are preformed with the number of negative sampling from 2 to 20 and the increase step is 2. First, Fig. 10 shows that the number of negative sampling has a significant impact on the model performance. In the ML100k dataset, it demonstrates that less (≤ 4) negative samples per positive instance is insufficient to achieve optimal performance. It also reveals that setting the sampling ratio too huge (≥ 10) may hurt the performance. In Amazon dataset, generally, the performance increases when the number of negative sampling increases. This is probably because of the data sparsity. Table 2 shows that the sparsity of Amazon dataset is about 10 times than ML100k datasets. That means that when the number of negative sampling is 6 in ML100k, there are about 30 percent user-item interactions are utilized for the training process. However, even the number of negative sampling is 20 in Amazon dataset, there are only 10 percent user-item interactions.

6 Conclusion

In this paper, we explore aspect-level information for collaborative filtering. We first propose a novel neural network based aspect-level collaborative filtering model (NeuACF) based on different-aspect features extracted from heterogeneous network with meta-paths. NeuACF is able to learn the aspect-level latent factors and then fuses them with the attention mechanism. Furthermore, in order to better fuse aspect-level information effectively, we propose NeuACF++ which employs the self-attention mechanism to learn the importance of different aspects. Extensive evaluations demonstrate the superior performance of NeuACF and NeuACF++.

In this paper, we mainly focus on fusing the latent factors learned in the last layer of the neural network. In the future, we aim to explore new attention mechanism which is able to consider all the latent factor information in all the network layers, so that we can capture more complete information. Moreover, since retraining the model is time-consuming and expensive for new meta-paths, another future work is to design a effective mechanisms to share the neural network which has been learned by before the aspect-level latent factors.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of China (No. 61532006, 61772082, 61702296, 61602237), the National Key Research and Development Program of China (2017YFB0803304), the Beijing Municipal Natural Science Foundation (4182043), and the 2019 CCF-Tencent Open Research Fund. This work is also supported in part by NSF under grants III-1526499, III-1763325, III-1909323, SaTC-1930941, and CNS-1626432.

References

[1] Y. Hu, Y. Koren, and C. Volinsky, “Collaborative filtering for implicit feedback datasets,” in *Proc. Ind. Conf. Data Mining*, 2008, pp. 263–272.
[2] Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” *Comput.*, vol. 42, no. 8, pp. 30–37, 2009.
[3] Y. Koren, “Factorization meets the neighborhood: A multifaceted collaborative filtering model,” in *Proc. 14th ACM SIGKDD Int. Conf. Special Interest Group Knowl. Discovery Data Mining*, 2008, pp. 426–434.
[4] H. Ma, H. Yang, M. R. Lyu, and I. King, “Sorec: Social recommendation using probabilistic matrix factorization,” in *Proc. 17th Conf. Inf. Knowl. Manage.*, 2008, pp. 931–940.
[5] C. Shi, J. Liu, F. Zhuang, P. S. Yu, and B. Wu, “Integrating heterogeneous information via flexible regularization framework for recommendation,” *Knowl. Inf. Syst.*, vol. 49, no. 3, pp. 835–859, 2016.
[6] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, “Neural collaborative filtering,” in *Proc. Int. World Wide Web Conf.*, 2017, pp. 173–182.
[7] H.-J. Xue, X. Dai, J. Zhang, S. Huang, and J. Chen, “Deep matrix factorization models for recommender systems,” in *Proc. Int. Joint Conf. Artif. Intell.*, 2017, pp. 3203–3209.

[8] C. Shi, Y. Li, J. Zhang, Y. Sun, and P. S. Yu, “A survey of heterogeneous information network analysis,” *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 1, pp. 17–37, Jan. 2017.

[9] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, “Pathsim: Meta path-based top-k similarity search in heterogeneous information networks,” in *Proc. Int. Conf. Very Large Data Bases*, 2011, vol. 4, pp. 992–1003.

[10] H. Wang, N. Wang, and D.-Y. Yeung, “Collaborative deep learning for recommender systems,” in *Proc. 21th ACM SIGKDD Int. Conf. Special Interest Group Knowl. Discovery Data Mining*, 2015, pp. 1235–1244.

[11] H. Zhao, Q. Yao, J. Li, Y. Song, and D. L. Lee, “Meta-graph based recommendation fusion over heterogeneous information networks,” in *Proc. Int. Conf. Special Interest Group Knowl. Discovery Data Mining*, 2017, pp. 635–644.

[12] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, “Collaborative filtering recommender systems,” in *The Adaptive Web*. Berlin, Germany: Springer, 2007, pp. 291–324.

[13] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua, “Fast matrix factorization for online recommendation with implicit feedback,” in *Proc. 39th Int. ACM SIGIR Conf. Special Interest Group Inf. Retrieval*, 2016, pp. 549–558.

[14] Y. Liu, P. Zhao, X. Liu, M. Wu, L. Duan, and X.-L. Li, “Learning user dependencies for recommendation,” in *Proc. 26th Int. Joint Conf. Artif. Intell.*, 2017, pp. 2379–2385.

[15] Y. Shi, X. Zhao, J. Wang, M. Larson, and A. Hanjalic, “Adaptive diversification of recommendation results via latent factor portfolio,” in *Proc. 35th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2012, pp. 175–184.

[16] Y. Koren and R. Bell, “Advances in collaborative filtering,” in *Recommender Systems Handbook*. Berlin, Germany: Springer, 2015, pp. 77–118.

[17] A. Mnih and R. R. Salakhutdinov, “Probabilistic matrix factorization,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2008, pp. 1257–1264.

[18] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, “BPR: Bayesian personalized ranking from implicit feedback,” in *Proc. Conf. Uncertainty Artif. Intell.*, 2009, pp. 452–461.

[19] V. S. Dave, B. Zhang, P.-Y. Chen, and M. A. Hasan, “Neural-brane: Neural bayesian personalized ranking for attributed network embedding,” *Data Sc. Eng.*, vol. 4, no. 2, pp. 119–131, 2019.

[20] R. He and J. McAuley, “VBPR: Visual bayesian personalized ranking from implicit feedback,” in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 144–150.

[21] W. Niu, J. Caverlee, and H. Lu, “Neural personalized ranking for image recommendation,” in *Proc. 11th ACM Int. Conf. Web Search Data Mining*, 2018, pp. 423–431.

[22] B. Zhang, S. Choudhury, M. A. Hasan, X. Ning, K. Agarwal, S. Purohit, and P. G. P. Cabrera, “Trust from the past: Bayesian personalized ranking based link prediction in knowledge graphs,” *CoRR*, vol. abs/1601.03778, 2016. [Online]. Available: http://arxiv.org/abs/1601.03778

[23] P. D. Hoff, A. E. Raftery, and M. S. Handcock, “Latent space approaches to social network analysis,” *J. American Statistical Assoc.*, vol. 97, no. 460, pp. 1090–1098, 2002.

[24] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin, “Graph embedding and extensions: A general framework for dimensionality reduction,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 1, pp. 40–51, Jan. 2007.

[25] K. Hornik, M. Stinchcombe, and H. White, “Multilayer feedforward networks are universal approximators,” *Neural Netw.*, vol. 2, no. 5, pp. 359–366, 1989.

[26] R. Salakhutdinov, A. Mnih, and G. Hinton, “Restricted boltzmann machines for collaborative filtering,” in *Proc. Int. Conf. Mach. Learn.*, 2007, pp. 791–798.

[27] S. Li, J. Kawale, and Y. Fu, “Deep collaborative filtering via marginalized denoising auto-encoder,” in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manag.*, 2015, pp. 811–820.

[28] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, “Autorec: Autoencoders meet collaborative filtering,” in *Proc. Int. World Wide Web Conf.*, 2015, pp. 111–112.

[29] F. Strub and J. Mary, “Collaborative filtering with stacked denoising autoencoders and sparse inputs,” in *Proc. NIPS Workshop Mach. Learn. eCommerce*, Montreal, Canada, Dec. 2015. [Online]. Available: https://hal.inria.fr/hal-01256422/file/Collaborative%20Filtering%20with%20Stacked%20Denoising%20AutoEncoders%20and%20Sparse%20Inputs.pdf

[30] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, et al., “Wide & deep learning for recommender systems,” in *Proc. 1st Workshop Deep Learn. Recommender Syst.*, 2016, pp. 7–10.

[31] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, “DeepFM: A factorization-machine based neural network for CTR prediction,” in *Proc. 26th Int. Joint Conf. Artif. Intell.*, AAAI Press, 2017, pp. 1725–1731.

[32] X. He and T.-S. Chua, “Neural factorization machines for sparse predictive analytics,” in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2017, pp. 355–364.

[33] X. He, X. Du, X. Wang, F. Tian, J. Tang, and T.-S. Chua, “Outer product-based neural collaborative filtering,” in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 2227–2233.

[34] Y. Bao, H. Fang, and J. Zhang, “TopicMF: Simultaneously exploiting ratings and reviews for recommendation,” in *Proc. 28th AAAI Conf. Assoc. Advance Artif. Intell.*, 2014, vol. 14, pp. 2–8.

[35] M. Wang, W. Fu, S. Hao, H. Liu, and X. Wu, “Learning on big graph: Label inference and regularization with anchor hierarchy,” *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 5, pp. 1101–1114, May 2017.

[36] C. Shi, Z. Zhang, P. Luo, P. S. Yu, Y. Yue, and B. Wu, “Semantic path based personalized recommendation on weighted heterogeneous information networks,” in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage.*, 2015, pp. 453–462.

[37] Y. Zhang, Q. Ai, X. Chen, and W. B. Croft, “Joint representation learning for top-n recommendation with heterogeneous information sources,” in *Proc. ACM Conf. Inf. Knowl. Manag.*, 2017, pp. 1449–1458.

[38] M. Wang, H. Li, D. Tao, K. Lu, and X. Wu, “Multimodal graph-based reranking for web image search,” *IEEE Trans. Image Process.*, vol. 21, no. 11, pp. 4649–4661, Nov. 2012.

[39] N. Lao and W. W. Cohen, “Relational retrieval using a combination of path-constrained random walks,” *Mach. Learn.*, vol. 81, no. 1, pp. 53–67, 2010.

[40] C. Shi, X. Kong, Y. Huang, P. S. Yu, and B. Wu, “Hetesim: A general framework for relevance measure in heterogeneous networks,” *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 10, pp. 2479–2492, Oct. 2014.

[41] W. Feng and J. Wang, “Incorporating heterogeneous information for personalized tag recommendation in social tagging systems,” in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2012, pp. 1276–1284.

[42] X. Liu, Y. Yu, C. Guo, and Y. Sun, “Meta-path-based ranking with pseudo relevance feedback on heterogeneous graph for citation recommendation,” in *Proc. 23rd ACM Int. Conf. Inf. Knowl. Manage.*, 2014, pp. 121–130.

[43] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han, “Personalized entity recommendation: A heterogeneous information network approach,” in *Proc. Int. Conf. Web Search Data Mining*, 2014, pp. 283–292.

[44] C. Luo, W. Pang, Z. Wang, and C. Lin, “Hete-CF: Social-based collaborative filtering recommendation using heterogeneous relations,” in *Proc. IEEE Int. Conf. Data Mining*, 2014, pp. 917–922.

[45] Z. Wei, X. He, X. Xiao, S. Wang, Y. Liu, X. Du, and J.-R. Wen, “PRSim: Sublinear time SimRank computation on large power-law graphs,” in *Proc. Int. Conf. Manage. Data*, ACM, 2019, pp. 1042–1059.

[46] Y. Wang, L. Chen, Y. Che, and Q. Luo, “Accelerating pairwise simrank estimation over static and dynamic graphs,” *Int. J. Very Large Data Bases*, vol. 28, no. 1, pp. 99–122, 2019.

[47] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, “Show, attend and tell: Neural image caption generation with visual attention,” in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 2048–2057.

[48] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo, “Image captioning with semantic attention,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 4651–4659.

[49] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” in *3rd Int. Conf. Lear. Representations*, San Diego, CA, USA, May 7-9, 2015. [Online]. Available: http://arxiv.org/abs/1409.0473

[50] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Proc. Advances Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[51] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Human Lang. Technol.*, 2019, vol. 1, pp. 4171–4186.

[52] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *Proc. 3rd Int. Conf. Learn. Representations*, San Diego, CA, USA, May 7-9, 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[53] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Computing Surveys*, vol. 52, no. 1, p. 5, 2019.

[54] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Trans. Interactive Intell. Syst.*, vol. 5, no. 4, 2016, Art. no. 19.

[55] R. He and J. McAuley, "Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering," in *Proc. Int. World Wide Web Conf.*, 2016, pp. 507–517.

[56] J. McAuley, C. Targett, Q. Shi, and A. Van Den Hengel, "Image-based recommendations on styles and substitutes," in *Proc. 38th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2015, pp. 43–52.

[57] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proc. Int. World Wide Web Conf.*, 2001, pp. 285–295.

[58] T.-A. N. Pham, X. Li, G. Cong, and Z. Zhang, "A general recommendation model for heterogeneous networks," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 12, pp. 3140–3153, Dec. 2016.

[59] T. Chen, W. Zhang, Q. Lu, K. Chen, Z. Zheng, and Y. Yu, "Svdfeature: A toolkit for feature-based collaborative filtering," *J. Mach. Learn. Res.*, vol. 13, pp. 3619–3622, 2012.

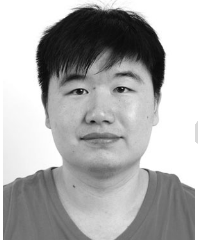
[60] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al., "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," *CoRR*, vol. abs/1603.04467, 2016. [Online]. Available: http://arxiv.org/abs/1603.04467

[61] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.

[62] L. V. D. Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, 2008.

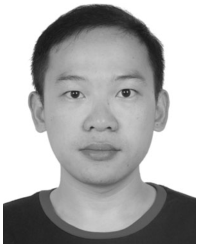
[63] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

Chuan Shi received the BS degree from Jilin University, in 2001, the MS degree from the Wuhan University, in 2004, and the PhD degree from the ICT of Chinese Academic of Sciences, in 2007. He joined the Beijing University of Posts and Telecommunications as a lecturer in 2007, and is a professor and deputy director of the Beijing Key Lab Intelligent Telecommunications Software and Multimedia at present. His research interests include the data mining, machine learning, and evolutionary computing. He has published more than 60 papers in refereed journals and conferences, such as the *IEEE Transactions on Knowledge and Data*, the *Knowledge and Information Systems*, the *Transactions on Intelligent Systems and Technology*, KDD, and IJCAI. He is a member of the IEEE.



Xiaotian Han received the BS degree from Shandong University, the MS degree in computer science from the Beijing University of Posts and Telecommunications. He is currently working toward the PhD degree at Texas A&M University. His research interests are social network and recommender system.

Li Song received the BS degree from North China Electric Power University, and the MS degree from the Beijing University of Posts and Telecommunications, in 2019. He is currently an algorithm engineer at JD. His research interests are recommender system and trajectory data mining.



Xiao Wang received the PhD degree from the School of Computer Science and Technology, Tianjin University, Tianjin, China, in 2016. He is currently an assistant professor with the Beijing University of Posts and Telecommunications, Beijing, China. Prior to that, he was a postdoctoral researcher in the Department of Computer Science and Technology, Tsinghua University, Beijing, China. He got the China Scholarship Council Fellowship in 2014 and visited Washington University in St. Louis, as a joint training student from Nov. 2014 to Nov. 2015. His current research interests include data mining, social network analysis, and machine learning. Until now, he has published more than 30 papers in conferences such as AAAI, IJCAI, etc., and journals such as the *IEEE Transactions on Cybernetics*, the *IEEE Transactions on Knowledge Discovery and Engineering*, etc. Now his research is sponsored by the National Science Foundation of China.

Senzhang Wang received the BSc degree from Southeast University, Nanjing, China, in 2009, and the PhD degree from Beihang University, Beijing, China, in 2016. He is currently an associate professor at the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, and also a "Hong Kong Scholar" postdoc fellow at the Department of Computing, The Hong Kong Polytechnic University. His main research focus is on data mining, social computing, and urban computing. He has published more than 70 referred conference and journal papers.

Junping Du received the PhD degree in computer science from the University of Science and Technology Beijing (USTB), and then held a postdoc fellowship in the Department of Computer Science, Tsinghua University, Beijing, China. She joined the School of Computer Science, Beijing University of Posts and Telecommunications (BUPT), in July 2006, where she is currently a professor of computer science. She was a visiting professor with the Department of Computer Science, Aarhus University, Denmark, from September 1996 until September 1997. Her current research interests include artificial intelligence, data mining, intelligent management system development, and computer applications.

Philip S. Yu received the BS degree in electrical engineering from National Taiwan University, the MS and PhD degrees in electrical engineering from Stanford University, and the MBA degree from New York University. He is a distinguished professor in computer science with the University of Illinois at Chicago and also holds the Wexler chair in information technology. His research interests include big data, including data mining, data stream, database, and privacy. He has published more than 1,000 papers in refereed journals and conferences. He holds or has applied for more than 300 US patents. He is the recipient of the ACM SIGKDD 2016 Innovation Award for his influential research and scientific contributions on mining, fusion, and anonymization of big data, the IEEE Computer Society's 2013 Technical Achievement Award for pioneering and fundamentally innovative contributions to the scalable indexing, querying, searching, mining, and anonymization of big data, and the Research Contributions Award from IEEE ICDM in 2003 for his pioneering contributions to the field of data mining. He also received the ICDM 2013 10-year Highest-Impact Paper Award, and the EDBT Test of Time Award (2014). He was the editor-in-chief of the ACM Transactions on Knowledge Discovery from Data (2011-2017) and the *IEEE Transactions on Knowledge and Data Engineering* (2001-2004). He is a fellow of the IEEE and ACM.