Collaborative Neural Social Recommendation

Le Wu, *Member, IEEE*, Peijie Sun, Richang Hong, *Member, IEEE*, Yong Ge, and Meng Wang, *Senior Member, IEEE*

***Abstract*—Collaborative filtering (CF) is one of the most popular techniques for building recommender systems. To overcome the data sparsity in CF, social recommender systems have emerged to boost recommendation performance by utilizing social correlation among users' interests. Recently, inspired by the immense success of deep learning for embedding learning, neural network-based recommender systems have shown promising recommendation performance. Nevertheless, few researchers have attempted to tackle the social recommendation problem with neural models. To this end, in this paper, we design a neural architecture that organically combines the intrinsic relationship between social network structure and user–item interaction behavior for social recommendation. Two key challenges arise in this process: first, how to incorporate the social correlation of users' interests in this neural model, and second, how to design a neural architecture to capture the unique characteristics of user–item interaction behavior for recommendation. To tackle these two challenges, we develop a model named collaborative neural social recommendation (CNSR) with two parts: 1) a social embedding part and 2) a collaborative neural recommendation (CNR) part. In CNSR, the user embedding leverages each user's social embedding learned from an unsupervised deep learning technique with social correlation regularization. The user and item embeddings are then fed into a unique neural network with a newly designed collaboration layer to model both the shallow collaborative and deep complex interaction relationships between users and items. We further propose a joint learning framework to allow the social embedding part and the CNR part to mutually enhance each other. Finally, extensive experimental results on two real-world datasets clearly demonstrate the effectiveness of our proposed model.**

***Index Terms*—Neural recommendation, social correlation, social embedding, social recommendation.**

I. Introduction

RECOMMENDER systems provide personalized item suggestions for each user, which have been widely used in many applications, such as movie recommendation [5], [20], [25], e-commerce recommendation [24], travel recommendation [26], and P2P lending recommendation [12]. Due to the explosive growth of information, collaborative filtering (CF)-based recommender systems have become extremely popular in recent years [1], [58]. CF infers each user's interests to items based on the collaborative behaviors of all users without requiring the creation of explicit user and item profiles. Among all the techniques for CF, latent factor-based models have received significant success from both academia and industry [20], [31], [36]. This kind of method project each user and item into a same low latent space, and then predict each user's unknown preference as a *linear collaborative interaction* between user and item latent vectors, i.e., the inner product of user and item latent vectors [31], [36]. Though widely studied in the past, in the real-world applications, the performance of CF is still unsatisfactory, especially when the user–item interaction data are very sparse [1].

Recently, the rapid expansion of the online social network applications pose new opportunities for CF. Living in the social community, people like to ask friends or social connections for recommendation. In fact, social scientists and some empirical studies have converged that social network users are correlated, i.e., a user's preference is similar to or influenced by her socially connected friends [2], [3]. Thus, the social recommender systems have emerged by incorporating the social network among users [18], [28], [33], [61]. These models were usually based on latent factor-based models, and varied in the form of incorporating the social correlation theory in the modeling process [18], [28]. For example, Ma *et al.* [28] designed a social regularization-based matrix factorization model, where the correlation between social ties is employed as a regularization term in the objective function. They partially solved the data sparsity issue for CF by leveraging social network information. Thus, other researchers extended these social recommendation models with item text information [33], [61]. However, the interaction between users and items in social recommender systems are nonlinear and complex, which could not be well captured by these shallow linear CF models. Thus, the problem of how to better represent users' interests from both social network and rating information to enhance recommendation performance is still under explored.

Luckily, deep learning-based methods have shown great potential for automatically representation (embedding) learning and delivered state-of-the-art performance in computer vision and language understanding domains [23], with broad

Manuscript received May 22, 2018; revised July 30, 2018; accepted September 11, 2018. This work was supported in part by the National Key Research and Development Program under Grant 2017YFB080330, and in part by the National Natural Science Foundation of China under Grant 61602147, Grant 61432019, Grant 61732008, Grant 61725203, and Grant 61722204. This paper was recommended by Associate Editor Q. Wang. *(Corresponding author: Richang Hong.)*

L. Wu, P. Sun, R. Hong, and M. Wang are with the School of Computer and Information, Hefei University of Technology, Hefei 230009, China (e-mail: lewu.ustc@gmail.com; sun.hfut@gmail.com; hongrc.hfut@gmail.com; eric.mengwang@gmail.com).

Y. Ge is with the Department of Management Information Systems, University of Arizona, Tucson, AZ 85721 USA (e-mail: yongge@email.arizona.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMC.2018.2872842

applications such as image classification [22], language translation [40], healthcare [30], and intelligent driving [48], [49]. Thus, researchers from both industry and academia have been in a race to apply deep learning-based methods for recommender systems. Several recent works have employed deep learning-based techniques to model auxiliary data in CF, such as the visual content of items [57] and text descriptions [47]. By leveraging the learned external embedding of a user (item), the predicted preference of the pairwise relationship between a user and an item is still resorted to a shallow CF practice with linear interaction function, i.e., the inner product between user and item latent factors [31]. Instead of modeling user–item interaction as the handcrafted shallow linear interaction function, researchers argued that user–item interaction behavior is complex and could be learned from neural networks [6], [15], [16]. The underlying reason for using deep networks to model user–item interaction is that the classical shallow user–item interaction function could memorize users' interests by comparing similarity of users and items. Modeling the complex interaction relationship between users and items could generalize users' preference prediction that is not captured by the shallow interaction function [6]. To avoid over-generalization of users' interests with deep models, these works showed that there are complementary advantages of modeling user–item interaction behavior with classical shallow CF models and deep neural architecture [6], [15], [16]. Hence, combining the results of these two parts would further improve recommendation performance [6], [14], [16]. Nevertheless, to the best of our knowledge, few have attempted to tackle the social recommendation task with neural models.

To this end, in this paper, we propose to design a neural network-based approach for social recommendation. This is a nontrivial task due to the difficulty of designing a neural architecture that organically combines the intrinsic relationship of social network structure and user–item interaction behavior in a unified recommendation framework. On one hand, social network contains rich correlated interest behaviors among users, and few deep learning models have been proposed to capture this social correlation theory for user interest modeling. Therefore, the social network differs from other kinds of auxiliary data (such as item visual content and text descriptions) in recommender systems that can be pretrained and obtained by mature text and image processing models [34], [41]. How do we capture the unique social interest correlation property in a unified neural recommendation? On the other hand, though there are reinforcement relationships between the shallow linear user–item interaction model and complex deep interaction model for recommendation, previous works tackled it by training a CF model and a deep neural network separately and then combining these two parts [6], [16]. Can the designed deep neural network model combine both the shallow collaborative and complex deep user–item interaction relationship in a unified framework for recommendation?

To tackle the above two challenges, we develop a deep neural model named collaborative neural social recommendation (CNSR), which jointly incorporates the unique characteristic of social network structure and user–item interaction in a unified model for social recommendation. Specifically, we represent each user and each item with a low latent embedding, which resembles the latent factors of users and items in classical CF models [36]. To utilize the social correlation among users in a social network, the user embedding leverages each user's social embedding learned from an unsupervised deep learning technique with social correlation theory-based regularization. Then, the user and item embeddings are fed into a unique deep neural network structure to model the deep complex relationships between users and items. In the meantime, the linear collaborative behavior is also explicitly incorporated in the deep architecture with a designed collaboration layer. Thus, the complex nonlinear and collaborative linear relationship between users and items are modeled in the proposed unified architecture. Given the proposed neural architecture, we design a joint learning framework that allows the social embedding part and the collaborative recommendation part to mutually enhance each other. In summary, the main contributions of this paper are as follows.

1) We propose a unified neural-based CNSR model for social recommender systems. Specifically, CNSR captures the social correlation of users' interests. Furthermore, it has a neural architecture that models both the shallow collaborative and deep complex interaction relationships between users and items for better recommendation performance.
2) We design an effective joint learning algorithm for CNSR. Thus, both the social embedding part and the collaborative neural part could mutually enhance each other. Besides, we also design a loosely learning algorithm for CNSR, which is more time efficient with a little loss in recommendation accuracy.
3) We compare the proposed model with many state-of-the-art baselines. The experimental results clearly show the superiority of our proposed model.

II. Related Work and Preliminaries

We summarize the related work into three categories: 1) the classical CF models for recommendation; 2) the deep learning-based recommendation; and 3) social recommendation.

A. Collaborative Filtering

CF suggests personalized item recommendations that a target user may be interested in based on the crowd users' historical behavior [1]. Among all CF techniques, the latent factor models are the most popular techniques due to their relatively high performance [8], [21], [31], [36]. Specifically, given a sparse user–item interaction matrix $\mathbf{R}$, the latent factor models embed each user and each item in a same latent space. Then, the predicted preference $\hat{r}_{ij}$ is calculated as the *linear collaborative interaction* between user i's embedding $\mathbf{U}_i$ and item j's embedding $\mathbf{V}_j$ [31], [36]

$$\hat{r}_{ij} = \mathbf{U}_i' \times \mathbf{V}_j. \tag{1}$$

In real-world applications, a common scenario is that users only implicitly express their behaviors of action or inaction (e.g., click or not click and buy or not buy). In this paper, we also consider the more practical problem of utilizing users' implicit feedback for item recommendation task. To directly optimize the ranking purpose of users' action or inaction in the rating matrix, BPR is proposed to design a ranking-based optimization function based on the predicted linear interaction rating in (1) [36]. These CF models captured the collaboration behavior of users and items and thus yielded great success in many applications in the past. Much research efforts have been devoted to enhancing these CF models, such as combining it with item topics [45], and extending it to a generic factorization machine without feature engineering efforts [35].

B. Deep Learning-Based Recommendation

Deep learning applies artificial neural networks to automatically learn multiple levels of feature representations of objects [23]. Due to the success of deep learning-based techniques in solving complex tasks, such as computer vision [22], [49] and natural language processing [40], researchers have been in a race to apply deep learning-based techniques for recommender systems [6], [7], [16], [47], [59]. Among all deep learning-based techniques, *autoencoder* provides an unsupervised technique to low-dimensional data representation and is well researched in recommender systems [44], [47]. By assuming the rating matrix $\mathbf{R}$ as a corrupted input of user preferences to items, AutoRec provided a variant of autoencoder that encoded hidden user representations from $\mathbf{R}$ and then decoded users' predicted preferences to unknown items in the output layer [37]. A similar idea of user representation learning from autoencoder has also been applied to the user representation learning in user–user interaction social network [56]. In fact, researchers showed that the denoising autoencoder for recommendation is actually a generalization of several well-known CF techniques [54]. Researchers have also applied various deep learning techniques for automatic feature learning from the auxiliary data in CF, such as visual images of items [50], text descriptions [47], and heterogeneous data sources [11], [57]. By leveraging the learned external embedding of a user (item), the predicted preference of a user to an item is still resorted to a shallow CF practice, i.e., the inner product of the user embedding and the item embedding (1). In contrast, some researchers argued that the user–item interaction behavior is complex and could not be fully captured by the handcrafted collaborative linear interaction between users and items. Thus, some recent studies proposed to adopt the deep neural network architecture to learn the complex interaction relationships between users and items [6], [16], [55]. To better generalize deep learning models for recommendation with various features, DeepFM [14] and NFM [15] are proposed to combine the power of factorization machines (FMs) for recommendation and deep learning for feature leaning in the neural network structure. Most of these studies experimentally validated that considering both the shallow collaborative linear interaction of user–item behavior and the complex deep relationships simultaneously would improve over the cases of considering either alone [6], [14], [16]. Despite the encouraging results of neural-based recommendation, to the best of our knowledge, few have explored the possibility of designing neural-based models for social recommendation.

C. Social Recommendation

Social scientists have long converged that a user's preference is similar to her social connections with the social correlation theories of homophily and social influence [2], [10], [18], [42], [53]. With the increasing popularity of online social networking platforms, the social network information has become an effective data source to alleviate data sparsity problem and enhance recommendation performance [42]. Since the latent factor-based models perform better than the neighborhood-based methods in CF, a popular direction is how to design more sophisticated latent factor-based social recommendation models. For example, Ma *et al.* [27], [28] proposed a latent factor-based framework with social regularization for recommendation. SocialMF is proposed to incorporate the social influence theory into classical latent factor-based models [18]. By treating each user's social connections' preferences of an item as the auxiliary feedback of this user, researchers proposed a trust-based latent factor-based model that leveraged the auxiliary feedback [13]. With the observation that users tend to assign higher rankings to items that their friends prefer, a personalized ranking-based social recommendation is proposed that extends the classical BPR model with the observation [60]. Researchers also argued that both positive and negative links in social networks provide valuable clues for recommendation performance [10]. These social recommendation algorithms have also been extended to incorporate rich context information, such as social circle [33] and item content [61]. Since the performance of these latent factor-based models for social recommendation relied on the initialization of user and item latent factors, researchers proposed to apply autoencoder, an unsupervised deep learning technique in initialization [9]. These models showed improved performance over classical recommendation models. Nevertheless, few have explored the possibility of designing deep learning-based social recommendation models. Recently, neural social collaborative ranking is proposed to solve the problem of bridging a few overlapping users in the two domains of the social network domain and information domain [52]. Researchers also proposed to use deep learning models to model the social influence strength for temporal social recommendation [39]. This paper on social recommendation differs from these works as we focus on the general setting a social platform when the temporal information is not available, and both the social network and user–item behavior are in a same platform.

III. Proposed Model

In a social recommender system, suppose there are a userset U ($|U| = N$) and an itemset V ($|V| = M$). Let the rating matrix $\mathbf{R} \in \mathbb{R}^{N \times M}$ denote the implicit feedback of users to items, with $r_{ij} = 1$ representing that user i is interested in item j.

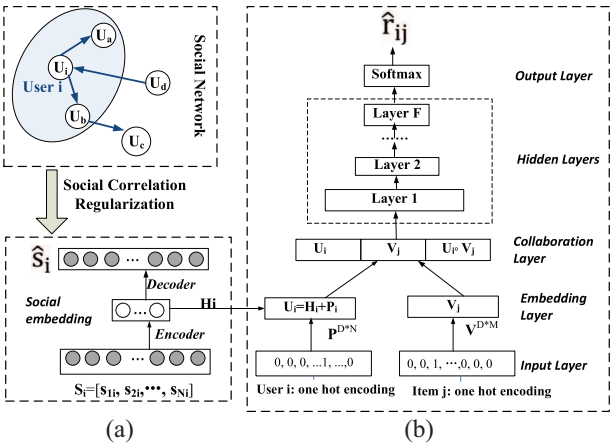


Fig. 1. Overall architecture of CNSR. (a) Social correlation-based embedding. (b) CNR with social embedding.

Otherwise, user i does not rate item j with $r_{ij} = 0$. Besides, users follow other users to form a user–user social matrix $\mathbf{S} \in \mathbb{R}^{N\times N}$, with $s_{ki} = 1$ if user i follows user k; otherwise, it equals 0. Thus, each user i's ego social structure can be represented as the ith column of $\mathbf{S}$, i.e., $\mathbf{S}_i \in \mathbb{R}^N$. Given a userset U, an itemset V, with the corresponding social matrix $\mathbf{S}$ and rating matrix $\mathbf{R}$, the overall goal of social recommendation is to predict each user i's preferences to the unrated items and then select a top-K list of items that have the highest predicted ratings.

A. Overall Structure of CNSR

In this section, we describe the proposed CNSR model for social recommendation. We present the architecture of CNSR in Fig. 1, which is composed of two parts: 1) a social correlation-based embedding part and 2) a collaborative neural recommendation (CNR) part. Taking the social network structure $\mathbf{S}$ as input, the social correlation-based embedding part learns the social interest embedding $\mathbf{H}_i$ of each user i from an unsupervised deep learning technique with social correlation regularization. With the parameters of user offset embedding $\mathbf{P}$ and the item embedding matrix $\mathbf{Q}$, the output of the social embedding part $\mathbf{H}$ is then fed into a CNR part to predict the user–item preference $\hat{r}_{ij}$. Next, we would introduce the social embedding part and the collaborative neural architecture in detail, followed by the joint learning framework.

1) Social Correlation-Based Embedding: The social interest embedding task deals with the problem of embedding a social network $\mathbf{S}$ of users into a low-dimensional interest embedding space, such that the connected users have similar vector representations. In this part, we show how to model the unique social correlation among users' interest in a social network with deep learning-based models. The deep learning-based models show the advantage of capturing nonlinear, complex features compared to other feature extraction techniques [23]. We choose an unsupervised deep learning model, i.e., *autoencoder*, as the base model for the social embedding learning as it is well recognized as a building block for deep learning and has received great success for feature representation learning in various tasks [37], [44], [47]. We put emphasis on how to incorporate the social correlation theory among users in a social network in autoencoder-based approach for better social interest embedding learning. The learned social interest embedding could complement the user interest embedding in the second stage, thus organically alleviating data sparsity problem of CF by mining the unique characteristics of social networks and achieving better performance of social recommendation.

Specifically, an autoencoder is an unsupervised feed-forward neural network for learning a compressed nonlinear representation (encoding) from a set of high-dimensional data [44]. The structure of autoencoder is shown in the left part of Fig. 1, which consists of an encoder part that maps the input vector into a hidden space and a decoder part that reconstructs the input data from the hidden representation. In a social network, given the social matrix $\mathbf{S}$, our goal is to learn a low-dimensional social embedding matrix $\mathbf{H} \in \mathbb{R}^{D\times N}$, where each user i's social embedding is denoted as the ith column of this embedding matrix, i.e., $\mathbf{H}_i$. The encoder and decoder part can be mathematically formulated as follows:

$$\begin{aligned} \text{Encoder:}\ \ \mathbf{H}_i &= f(\mathbf{A} \times \mathbf{S}_i + \mathbf{a}) \\ \text{Decoder:}\ \ \hat{\mathbf{S}}_i &= f'\left(\mathbf{A}' \times \mathbf{H}_i + \mathbf{a}'\right) \end{aligned} \tag{2}$$

where $\mathbf{H}_i \in \mathbb{R}^D$ is the hidden social embedding of user input $\mathbf{S}_i$, and $\hat{\mathbf{S}}_i$ is the reconstructed vector of $\mathbf{S}_i$. $f(x)$ and $f'(x)$ are nonlinear functions. $[\mathbf{A}, \mathbf{A}' \in \mathbb{R}^{N\times D}, \mathbf{a} \in \mathbb{R}^D, \mathbf{a}' \in \mathbb{R}^N]$ are the parameters that need to be learned. Usually, the parameters in an autoencoder are trained to minimize the reconstruction error between the input $\mathbf{S}$ and predicted output $\hat{\mathbf{S}}$.

In online social platforms, two important social theories exist: 1) the homophily theory states that users would like to connect to others who have similar preferences as them in the past [29] and 2) the social influence theory reveals the phenomenon that the actions of a user can induce his/her friends to behave in a similar way [3]. Both theories would lead to the *social correlation* phenomenon among users, i.e., a user's characteristic (e.g., preference and behavior) is correlated to her social affiliations. This phenomenon is well supported by many empirical studies and has been well recognized as a foundation for social recommendation [2], [3], [42]. Thus, instead of simply defining the social embedding loss function as the reconstruction error between inputs and outputs with the autoencoder framework, we formulate a social correlation-based embedding loss function as

$$\begin{aligned} \min \mathcal{L}\left(\mathbf{S}, \hat{\mathbf{S}}\right) = &- \sum_{i=1}^{N}\left\{\sum_{k=1}^{N}\left[s_{ki}\log\left(\hat{s}_{ki}\right) + (1 - s_{ki})\log\left(1 - \hat{s}_{ki}\right)\right]\right\} \\ &+ \beta \sum_{i=1}^{N}\sum_{k=1}^{N} s_{ki}\|\mathbf{H}_i - \mathbf{H}_k\|_F^2. \end{aligned} \tag{3}$$

In the above loss function, similar to many autoencoder-based embedding models, the first term captures the training reconstruction error with log-loss function. In practice, different kinds of loss functions could be applied to the above social embedding construction loss, such as the squared loss [31], the log loss [16], and the ranking-based loss function [36]. Since the social connection matrix contains binary values, we select

the widely used *log loss* function for modeling, which can be seen as the negative log-likelihood of the binary outputs with a Bernoulli distribution. The second term regularizes the social embedding with social correlation theory. Specifically, if user i follows user k (i.e., $s_{ki} = 1$), i either has similar interests with k or is influenced by k, leading to the correlated behavior between this pair of users. We put a social correlation regularization term to minimize the social interest embedding difference between them. This social correlation-based regularization formulation also applies to the undirected friendship network. Under the undirected friendship network, if user i and user k are friends (i.e., $s_{ik} = 1$ and $s_{ki} = 1$), then the social correlation regularization appears in both i's social embedding regularization and k's social embedding regularization. In the above loss function, β is a balance parameter that controls the relative importance of the two terms. The larger the β, the more we rely on the social regularization for the social embedding process. Specifically, if $\beta = 0$, the social embedding part degenerates to a classical autoencoder.

Please note that, in fact, the network embedding learning is a rather hot research topic with numerous modeling techniques [43], [46], [51]. However, these embedding works focused on how to model the generalized structure of graphs. We put emphasis on how to extract useful social interest embeddings of users from a social network by incorporating social correlation theory among users' interests in the social network structure with deep learning techniques. We would illustrate how to leverage the learned social embedding to strength recommendation in our proposed neural architecture in a unified learning framework, and leave the exploration of various social embedding techniques in the future work.

2) Collaborative Neural Recommendation: The CNR part is shown on the right in Fig. 1, which takes each user–item pair (i, j) as input with the auxiliary social embedding vector $\mathbf{H}_i$, and generates the predicted rating $\hat{r}_{ij}$ in the output layer. This part contains five layers: 1) the input layer; 2) the embedding layer; 3) the collaboration layer; 4) the hidden layers; and 5) the output layer. Above the input layer is an embedding layer that projects each user and item into a dense latent space. This embedding layer could be seen as performing the latent modeling of users and items in traditional CF tasks [21], [36]. Then, we merge the user embedding, the item embedding, and collaborative interaction between user–item embeddings into a proposed *collaboration layer* to explicitly model the linear collaborative interaction behavior in recommender systems. The collaboration layer is further fed into a feed-forward neural network with multiple hidden layers to automatically learn the complex relationships between users and items for recommendation. The final output layer is a softmax function that generates the predicted rating $\hat{r}_{ij}$ from the output of the hidden layers. Particularly, there are two key components in this part. On one hand, by expanding the user embedding with the social interest embedding matrix, the recommendation task can partially alleviate the cold start problem by organically combining the social network information. On the other hand, above the embedding layer, we propose a unique collaboration layer that explicitly models the shallow collaborative interaction between users and items. This operation results in an informative representation in the low layers in the proposed neural architecture, facilitating the following deep architectures to learn the nonlinear complex relationships between user and item interactions. Thus, the proposed architecture could capture both the shallow linear and deep complex relationships in a unified model. Next, we would explain these layers in detail.

Input Layer: In a social recommendation task, the input is a sparse user–item interaction matrix $\mathbf{R}$ with the auxiliary user–user interaction matrix $\mathbf{S}$. To keep track of the total N users and M items for recommendation, a simple yet commonly adopted idea is to represent each user and each item with one hot encoding representing their identities [16], [35].

Embedding Layer: The embedding layer learns the low-latent representations of users and items given the input data, which can be seen as performing the latent factor learning in classical CF models [36]. Let $\mathbf{U} \in \mathbb{R}^{D \times N}$ and $\mathbf{V} \in \mathbb{R}^{D \times M}$ denote the user latent matrix and item latent matrix. Given the one hot representations of users and items, the corresponding user and item latent vectors are $\mathbf{U}_i$ and $\mathbf{V}_j$ for user i and item j, respectively. In an online social platform, we assume each user's embedding is composed of two parts: a social embedding part $\mathbf{H}$ that could be captured by the social correlation phenomenon of users' interests in the social network. Besides, each user has her unique interest, which could not be modeled in the social network. The offset embedding matrix $\mathbf{P}$ captures users' unique latent preferences. Then, each user i's embedding U_i could be reformulated as a combination of the parts as

$$\mathbf{U}_i = \mathbf{H}_i + \mathbf{P}_i. \tag{4}$$

In the above equation, specifically, when the social network structure $\mathbf{S}$ is not available, i.e., $\mathbf{H} = 0$, each user i's latent interest embedding $\mathbf{U}_i$ degenerates to $\mathbf{P}_i$. Under this circumstance, the offset embedding matrix $\mathbf{P}$ represents the embedding of each user, which is equal to the base user embedding matrix in latent factor-based recommendation models [16], [36].

Collaboration Layer: We design the collaboration layer to capture the shallow linear user–item interaction in the low layer of the proposed neural architecture. Specifically, given user and item embeddings, the lth input of user–item pair (i, j) is defined as

$$\mathbf{X}_l = \left[\mathbf{U}_i, \mathbf{V}_j, \mathbf{U}_i \circ \mathbf{V}_j\right] \tag{5}$$

where $\circ$ denotes the element-wise product of two vectors. This layer captures the domain knowledge that is related to the user–item interaction score r_{ij}. Specifically, $[\mathbf{U}_i, \mathbf{V}_j]$ is the concatenation vector of the user–item pair, which is widely used for automatically learning user–item complex relationships in the hidden layers in deep learning models [16], [55]. Besides, we also directly model the collaborative user–item interaction behavior as the element-wise product of their corresponding embeddings: $\mathbf{U}_i \circ \mathbf{V}_j$ and concatenate this vector in the collaboration layer. Someone may argue that it is unnecessary to add the collaboration vector $\mathbf{U}_i \circ \mathbf{V}_j$ in this layer, as the future hidden layers can automatically learn the collaborative behavior between users and items. However, we argue

that a key weakness of simply concatenating user and item embeddings is that it carries little information about the shallow collaborative user–item interaction behavior in the low level. Then the neural network needs to rely on the higher layers to learn meaningful interactions between users and items, which is unfortunately difficult to train in practice [23]. To solve this problem, we directly model the collaborative user–item interaction behavior as the element-wise product of their corresponding embeddings: $\mathbf{U}_i \circ \mathbf{V}_j$ and concatenate this vector in the collaboration layer. We would perform experiments to show the soundness of the collaboration layer later.

Hidden Layers: Besides the shallow collaborative interaction, to model the complex relationships of user–item pairs, we feed the output of the collaboration layer into a fully connected feed-forward neural network with multiple hidden layers. These multilayered feedforward networks are useful as they are demonstrated to be able to approximate any measurable function [17]. Specifically, the first hidden layer takes the shallow collaborative interaction layer as input, and the following layers take the output of previous layer as input. Thus, the hidden layers could learn the complex relationship with a greedy layer-by-layer method. The higher layers could learn more abstract concepts from the lower layers. The number of hidden layers controls the complexity of the multilayer neural network.

Given the output $\mathbf{X}_l$ of the collaboration layer, the output of the fth hidden layer, denoted as $h^f(\mathbf{X}_l)$, is a nonlinear feature transformation of the previous $(f-1)$th hidden layer

$$h^f(\mathbf{X}_l) = \sigma\left(\mathbf{W}^f \times h^{(f-1)}(\mathbf{X}_l) + b^{(f)}\right) \tag{6}$$

where $\mathbf{W}^f$ and $\mathbf{b}^f$ are the parameters of the fth hidden layer. $\sigma(x)$ is a nonlinear activation function. In this paper, we adopt the widely used rectifier linear unit (ReLU) activation function as $\sigma(x) = \max(0, x)$ [23].

Softmax Layer With Output: To predict the final rating preference $\hat{r}_{ij}$ that ranges from [0, 1], the last hidden layer is then fed into the output layer with a softmax function for prediction. Specifically, combing all the above variables, the predicted output is defined as

$$\begin{aligned} \hat{r}_{ij} &= f\left(\mathbf{W}^F \times h^F(\mathbf{X}_l) + \mathbf{b}^F\right) \\ h^F(\mathbf{X}_l) &= h^F\Big(\cdots h^2\Big(h^1\big(\big[\mathbf{U}_i, \mathbf{V}_j, \mathbf{U}_i \circ \mathbf{V}_j\big]\big)\Big)\Big). \end{aligned} \tag{7}$$

Note that the softmax function is set as $f(x) = [2/(1 + e^{-x})] - 1$ to ensure the predicted output ranges from [0, 1] as $x \geq 0$ when the activation function is set as ReLU.

B. Joint Model Learning Framework

In this section, we introduce how to design a joint learning framework for parameter learning. Specifically, given the model parameter set $\Theta = [\mathbf{A}, \mathbf{A}', \mathbf{a}, \mathbf{a}', \mathbf{P}, \mathbf{V}, [\mathbf{W}^f, \mathbf{b}^f]_{f=1}^F]]$, the final objective function is composed of two parts:

$$\min_{\Theta} \mathcal{L} = \mathcal{L}\left(\mathbf{R}, \hat{\mathbf{R}}\right) + \alpha \mathcal{L}\left(\mathbf{S}, \hat{\mathbf{S}}\right) \tag{8}$$

where the first part captures the loss between the predicted rating $\hat{\mathbf{R}} = [\hat{r}_{ij}]$ and $\mathbf{R}$, and the second part models the social embedding loss as defined in (3). α is a balance parameter

Algorithm 1 Joint Training Algorithm for CNSR

Input: Rating matrix $\mathbf{R}$ and social matrix $\mathbf{S}$;
Output: Model parameter set Θ;

```
Initialize model parameter set Θ with small random values;
while Not converged do
    for All users i ∈ U do
        Calculate social embedding H_i (Eq. (2));
        Calculate U_i in the embedding layer (Eq. (4));
        for Each user-item (i, j) pair from user i do
            Calculate X_l of the collaboration layer (Eq. (5));
            Claculate h^f(X_l) of the hidden layers (Eq. (6));
            Calculate r̂_ij of the output layer (Eq. (7));
        end for
        Update all parameters in the objective function (Eq. (10)) using back-propagation;
    end for
end while
Return model parameter set Θ;
```

that fuses these two loss functions. Since we focus on implicit feedback of users, we also select the *log loss* function for the rating modeling

$$\min_{\Theta} \mathcal{L}\left(\mathbf{R}, \hat{\mathbf{R}}\right) = -\sum_{i=1}^{N}\sum_{j=1}^{M}\left[r_{ij}\log\left(\hat{r}_{ij}\right) + \left(1 - r_{ij}\right)\log\left(1 - \hat{r}_{ij}\right)\right]. \tag{9}$$

Combining (3) and (9), the final objective function is defined as

$$\begin{aligned} \min_{\Theta} \mathcal{L} = &-\sum_{i=1}^{N}\sum_{j=1}^{M}\left[r_{ij}\log\left(\hat{r}_{ij}\right) + \left(1 - r_{ij}\right)\log\left(1 - \hat{r}_{ij}\right)\right] \\ &+ \alpha\sum_{i=1}^{N}\sum_{k=1}^{N}\left[-s_{ki}\log\left(\hat{s}_{ki}\right) - (1 - s_{ki})\log\left(1 - \hat{s}_{ki}\right)\right] \\ &+ \alpha\sum_{i=1}^{N}\left[\beta\sum_{k=1}^{N}s_{ki}\|H_i - H_k\|_F^2\right]. \end{aligned} \tag{10}$$

For model learning, we propose to train all model parameters Θ in a joint framework. We call this approach as joint training. In other words, joint training optimizes the parameters in the social embedding part (with the parameter set $\Theta_s = [\mathbf{A}, \mathbf{A}', \mathbf{a}, \mathbf{a}']$) and the collaborative neural embedding part ($\Theta_a = [\mathbf{P}, \mathbf{V}, [\mathbf{W}^f, \mathbf{b}^f]_{f=1}^F]]$) simultaneously at the training time. In this way, the two parts could mutually influence each other and improve the overall performance. An alternative solution is train the social correlation-based representation learning first (with the parameter set $\Theta_s = [\mathbf{A}, \mathbf{A}', \mathbf{a}, \mathbf{a}']$), and then fine-tune CNSR from the rating information with the remaining parameters Θ_a with fixed Θ_s. In fact, the second approach works as the pretraining and fine tuning of the deep learning [4]. We call this alternative approach as loosely training. In this loosely approach, the rating prediction process entirely relies on the social embedding part while the social embedding part could not benefit from rating information. We would show the superiority of joint training in the experimental part.

We implement CNSR with TensorFlow[1] to jointly train model parameters by performing stochastic gradient descent with mini-batch Adam [19]. The joint training algorithm of CNSR is shown in Algorithm 1. Specifically, since we only have the observed positive feedback of value 1 with large missing values of 0 in both the rating matrix $\mathbf{R}$ and the social matrix $\mathbf{S}$, we undersample some negative values from the missing data during model training process. Particularly, for each positive value, we randomly select five missing values as observed pseudo negative values at each iteration of the learning process. Since the sampling process is random and each time the negative samples change, each missing value gives very weak negative signal [32].

Dropout Training: While a deep neural network that contains multiple hidden layers is powerful to learn arbitrary relationship between its input and output, it is easy to overfit the training data. Dropout is a technique that significantly reduces overfitting and gives major improvements over other regularization methods [38]. Specifically, the key idea of dropout is to randomly drop a ratio of neurons and their connections from the neural network in the training process. The rationale is that this random dropout procedure prevents complex co-adaptations of neurons in which a neuron is only helpful in the context of several other units. In practice, in each iteration of CNSR, we set a dropout ratio ρ and performs dropout on the collaboration and hidden layers to reduce overfitting.

C. Discussion

The proposed CNSR model has two key characteristics.

1) We propose a social correlation-based interest embedding part that captures the social correlation among users' interest to strengthen the user embedding learning in a unified framework.
2) In the deep neural architecture, we design a unique collaboration layer to explicitly capture the shallow user–item relationship in CF tasks, and feed this layer into a multilayer connected deep network to learn the complex relationship between users and items.

Thus, both the shallow linear interaction behavior and the complex nonlinear interaction behavior between user and item pairs are well captured in a unified framework.

To the best of our knowledge, few research works have applied neural models for social recommendation. If we omit the social embedding part in the proposed CNSR model, i.e., each user i's social embedding $\mathbf{H}_i$ equals to zero, and embedding vector $\mathbf{U}_i$ degenerates to $\mathbf{P}_i$ with no social embedding enhancement. We call this degenerated version of CNSR as CNR. Please note that CNR is also a model proposed by us. In the following, we analyze the detailed properties of CNR.

Generalization of Latent Factor Models: The classical latent factor-based models consider the shallow linear relationship between user and item latent vectors [31], [36]. If we omit the social embedding part ($\mathbf{H}_i = 0$) and the hidden layers ($F = 0$ for the hidden layers) in the CNSR architecture, it is reduced to the generalized latent factor-based models as

$$\begin{aligned}\hat{r}_{ij} &= f\Big(h^0\big([\mathbf{U}_i, \mathbf{V}_j, \mathbf{U}_i \circ \mathbf{V}_j]\big)\Big) \\ &= f\big(\mathbf{W}_U \times \mathbf{U}_i + \mathbf{W}_V \times \mathbf{V}_j + \mathbf{W}_I \times \big(\mathbf{U}_i \circ \mathbf{V}_j\big)\big)\end{aligned}$$

where $\mathbf{W}_U \in \mathbb{R}^D$ and $\mathbf{W}_V \in \mathbb{R}^D$ captures the user and item bias from their corresponding embeddings. The third part $\mathbf{W}_I \in \mathbb{R}^D$ models the shallow interaction between users and items, where the kth dimension of $\mathbf{W}_I$ denotes the importance of this latent dimension. If we set $\mathbf{W}_U = [0, \ldots, 0], \mathbf{W}_V = [0, \ldots, 0], \mathbf{W}_I = [1, \ldots, 1]$, the above prediction function degenerates to the inner product of user and item latent vector ($\hat{r}_{ij} = f(U_i' \times V_j)$, which is the prediction function in classical latent factor-based models [31], [36].

Relations to Related Deep Neural Models: Our proposed simple model CNR has a similar multilayered neural architecture with several deep learning models, such as Wide&Deep [6], NeuMF [16], DeepFM [14], and NFM [15]. Among them, the first three related models are all composed of outputs of two models: 1) a shallow linear model to depict the simple linear relationship and 2) a deep neural network to model complex interaction relationship. The intuition of combining the outputs from a shallow linear model and a deep nonlinear model for the prediction task is that: the shallow linear model could memorize the low level interactions among users and items, and the deep nonlinear model could generalize to the complex nonlinear interactions that are hard to interpret with prior human engineering. Thus, combining the two parts would lead to better performance than either alone.

Specifically, NeuMF is most similar to CNR as both models take user–item interaction records as input. NeuMF is composed of a shallow latent factor model and a classical multilayered neural network with the concatenation of user embedding and item embedding as input, and the outputs of the latent factor model and the neural network are combined together for recommendation [16]. Wide&Deep and DeepFM applied to the scenario where the input includes different kinds of sparse auxiliary features, and the shallow part is modeled as a linear regression in Wide&Deep and an FM in DeepFM. Instead, our simplified model CNR differs from these works by designing a single model with a proposed *collaboration layer* in the low layers of the neural network. This results in a much more informative representation in the lower layers of the neural network, greatly helping the higher layers to learn nonlinear complex relationships. Thus, both the linear collaboration behavior and the nonlinear complex behavior are jointly modeled in a unified framework, avoiding the human labor of training two parts.

NFM is a recently proposed neural model for recommendation that generalizes FM [35] by taking user–item interaction records and the sparse categorical variables as input [15]. The key idea of NFM is a new designed bilinear interaction pooling layer; thus, NMF enhances FM by modeling nonlinear feature interactions. Meanwhile, NFM also preserves the linear computational time complexity of FM. Specifically, given user i and item j without any other features, the bilinear pooling layer in NFM degenerates to the element-wise product of the user embedding $\mathbf{U}_j$ and item embedding $\mathbf{V}_j$ as: $\mathbf{U}_i \circ \mathbf{V}_j$.

[1]https://www.tensorflow.org/

TABLE I
STATISTICS OF THE TWO DATASETS

Dataset	Users	Items	Ratings	Links	Rating Density
Flixster	9874	10978	283503	120306	0.262%
Douban	6739	17902	840828	201014	0.697%

Our designed collaboration layer is more informative under this circumstance as we also concatenate the corresponding user embedding and item embedding in the proposed collaboration layer. Thus, the future hidden layers could learn the complex interactions between $\mathbf{U}_i$ and $\mathbf{V}_j$ that are not captured by NFM. Nevertheless, when applying our proposed model to the general setting with various features, the time complexity increases. In summary, as the input data are different, our model design is different from NFM, as NFM shows its strength that models categorial data with linear time while our model focuses on the input data with user–item interaction records.

IV. EXPERIMENTS

A. Data Description and Experimental Setup

We experiment with two public datasets: *Flixster* [18] and *Douban*.[2] On both datasets, we select users that have at least five rating records and five social links. After that, we filter out items that have been rated less than five times. Since we focus on implicit feedback of users, similar to many recent research works, each entry in the rating matrix is marked as 0 or 1, indicating whether the user has rated the item or not [16], [36]. Table I shows the statistics of the datasets after pruning.

Baselines: We compare our proposed model with the following baselines: *BPR* [36], SocialMF [18], SR [28], AutoRec [37], and NeuMF [16]. Specifically, BPR is a competitive method for CF with binary inputs [36]. SocialMF [18] and SR [28] are two state-of-the-art latent factor models for social recommendation. AutoRec provides an autoencoder-based framework for recommendation [37]. NeuMF provides a neural approach for CF and further linearly combines the neural results with classical CF model to enhance performance [16]. We choose AutoRec and NeuMF as they are two state-of-the-art deep learning models with competitive performance. As some of these baselines are designed for the rating prediction problem with a squared loss function, for fair comparison, we change the squared loss to the log loss function (9) that is more suitable for implicit feedback.

Besides, to better show the effectiveness, we devise two simplified versions of CNSR: a neural social recommendation (NSR) model that does not consider the shallow user–item interaction in the collaboration layer, i.e., $\mathbf{X}_l = [\mathbf{U}_i, \mathbf{V}_j]$ for the lth rating record, and a CNR model that does not incorporate the social embedding for recommendation, i.e., $\mathbf{H}_i$ equals zero for each user. We use two training methods for CNSR as shown in Section III-B, a joint training approach named CNSR_J and a loosely training approach as CNSR_L.

Please note that there are some other neural recommendation models as mentioned before, i.e., Wide&Deep [6] and DeepFM [14]. These two models shared similar parts as the NeuMF baseline, and they need extra profiles and item content for recommendation. Hence, we do not choose them as baselines. We also do not show the results of NFM [15] as it degenerates to a simplified version of our proposed CNR model, and NFM usually performs a little worse than CNR when only user–item interaction data are available.

Evaluation Metrics: In model validation process, we randomly select 80% of the rating records as the training data, 10% as the validation data, and the remaining 10% as the test data. As the user size is huge, it is inefficient to take all users as candidates to generate the top-K recommendations. We tackle this issue as widely used in the ranking-based recommendation tasks: for each test user, we randomly sample 1000 unrated items in the training data. Then we mix the records in the validation set and the test set with the samples together to select the top-K recommendation results. This process is repeated ten times and we report the average results [20]. We select the hit ratio (HR) and normalized discounted cumulative gain (NDCG) metrics to measure the top-K ranking performance. Specifically, HR measures the percentage of ranked items that are liked by users. For each user i, suppose T_i is the itemset that i likes in the test data, and O_i is the predicted top-K ranking list. Then, for each user, the HR_i measure is defined as

$$\text{HR}_i = \frac{\sum_{j=1}^{K} \mathbb{I}_{T_i}(O_{ij})}{|T_i \cup O_i|} \tag{11}$$

where O_{ij} denotes the jth element in the list O_i. $\mathbb{I}_A(x)$ is an indicator function that equals 1 if $x \in A$; otherwise, it equals 0. After that, the overall HR score is the average of each user's HR_i. Instead of treating the hit items as equally important in the HR measure, the NDCG measure gives a higher score to the hit items that are ranked higher in a ranking list. For each user i, we first calculate the discounted cummulative gain (DCG) as

$$\text{DCG}_i = \sum_{j=1}^{K} \frac{\mathbb{I}_{T_i}(O_{ij})}{\log_2(j+1)}. \tag{12}$$

The NDCG value of user i is computed as a normalized value of the DCG:

$$\text{NDCG}_i = \frac{\text{DCG}_i}{\text{IDCG}_i} \tag{13}$$

where IDCG is the ideal discounted cumulative gain as $\text{IDCG}_i = \sum_{l=1}^{|T_i|} (1/[\log_2(l+1)])$.

Parameter Setting: We set the parameters of CNSR with the following values: the dimension of user embeddings and item embeddings is set as $D = 64$, and the number of hidden layers is set as 2 with each layer's size as: $64 \rightarrow 32$. The dropout rate is set as $\rho = 0.2$ for default. For the regularization parameters, the $\alpha = 0.1$ on both datasets, and $\beta = 0.3$ on Flixster and $\beta = 0.2$ on Douban. Due to the nonconvexity of our proposed model, we initialize the P and Q values from the baseline of BPR [36]. For fair comparison, all parameters in the baselines are tuned to ensure the best performance. For example, we perform the same pretraining technique as introduced in NeuMF [16] to ensure the fair comparison.

[2] https://goo.gl/GUPEZp

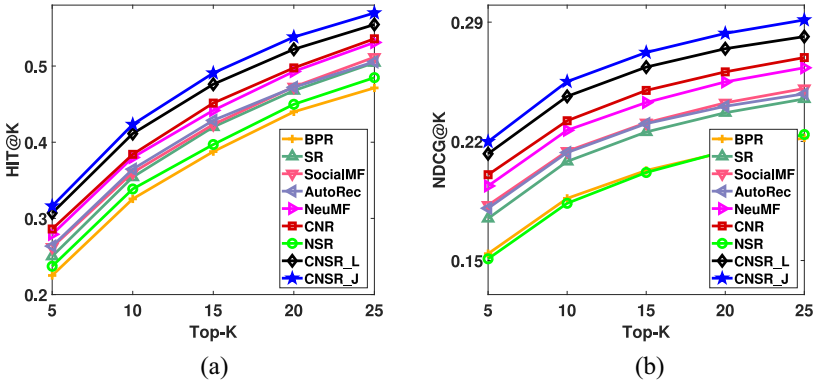


Fig. 2. Overall performance on Flixster. The improvements of CNSR_L and CNSR_J over the baselines pass the *t*-test at a confidence level of 0.01 (better viewed in color). (a) HR@K. (b) NDCG@K.

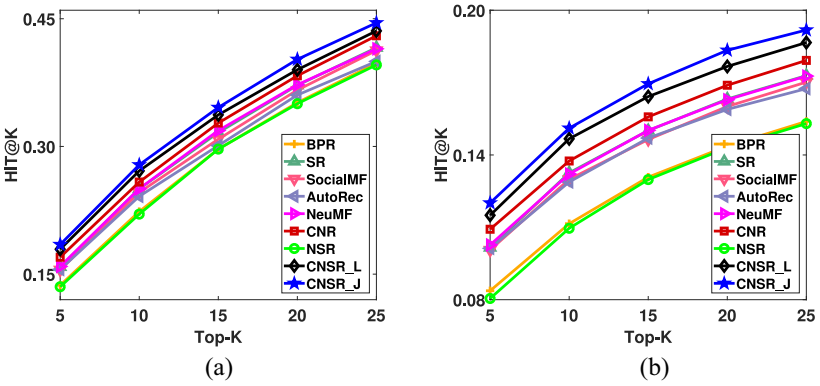


Fig. 3. Overall performance on Douban. The improvements of CNSR_L and CNSR_J over the baselines pass the *t*-test at a confidence level of 0.01 (better viewed in color). (a) HR@K. (b) NDCG@K.

B. Overall Performance

Effectiveness: Figs. 2 and 3 show the performance of different methods on HR@K and NDCG@K on the two datasets. As can be seen from both figures, our proposed method CNSR always performs the best. NeuMF shows the best results among all models that only utilize user–item interaction records. The SocialMF and SR baseline improve over BPR, showing the effectiveness of incorporating social network information. When top-K list equals 5, the improvement of CNSR over NeuMF is more than 15% on NDCG metric and 12% on HR metric on Flixster. As K increases, we observe that the improvement trend of CNSR over baselines decreases and the NDCG metric always has larger improvement over HR. We guess a possible reason is that NDCG considers the ranking position of the hit items, and our proposed CNSR model can better predict the items that are in the top positions in the ranking list.

Next, we compare CNSR with our proposed two simplifications, CNR and NSR. NSR that does not consider the shallow collaboration between users and items performs badly on both datasets. A possible reason is that, as the shallow collaboration behavior between users and items is ignored by NSR, the hidden layers could not learn a reliable user–item interaction from the raw data. This empirical observation also validates the importance of combining the shallow user–item interaction in neural-based recommendation models. This is also the reason why many neural network-based recommendation methods combined a separate classical shallow recommendation model [6], [14], [16]. The performance of CNR is worse than CNSR; however, it competes over all baselines. Based on these two observations, we conclude that the collaboration layer in CNSR is very critical to capture the shallow collaborative user–item interaction behavior for prediction. Also, leveraging the social embeddings could further improve the recommendation performance in our proposed CNSR model. Last but not least, when comparing the two training methods of CNSR, CNSR_J consistently outperforms CNSR_L with about 1% to 2% improvement. This experimental observation also empirically validates that the joint training approach could discover the mutual enhancement between the social embedding part and the CNR part, thus improving the overall performance.

TABLE II
RUNTIME OF ALL MODELS IN EACH ITERATION (SECONDS)

Models	Flixster	Douban
BPR [36]	2.35	3.42
SocialMF [18]	3.17	6.23
SR [28]	3.43	7.22
AutoRec [37]	3.25	21.02
NeuMF [16]	5.36	6.44
CNR	3.23	5.15
NSR	5.17	7.32
CNSR_L	8.34	12.72
CNSR_J	22.57	26.85

TABLE III
NDCG@10 WITH DIFFERENT DEEP ARCHITECTURE

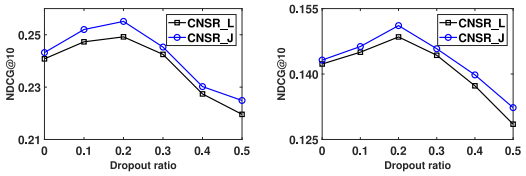
Flixster Dataset								
Model	CNSR_L				CNSR_J			
Capacity	F=0	F=1	F=2	F=3	F=0	F=1	F=2	F=3
D=16	0.2147	0.2178	0.2181	0.2105	0.2191	0.2209	0.223	0.2189
D=32	0.2358	0.2434	0.2451	0.2433	0.2404	0.2475	0.2482	0.2446
D=64	0.2460	0.2484	**0.2492**	0.2442	0.2473	0.2490	**0.2551**	0.2530
Douban Dataset								
Model	CNSR_L				CNSR_J			
Capacity	F=0	F=1	F=2	F=3	F=0	F=1	F=2	F=3
D=16	0.1250	0.1279	0.1289	0.1282	0.1278	0.1293	0.1301	0.1306
D=32	0.1374	0.1422	0.1433	0.1405	0.1389	0.1422	0.1447	0.1430
D=64	0.1412	0.1457	**0.1470**	0.1452	0.1453	0.1493	**0.1501**	0.1478

Efficiency: Since all the proposed algorithms rely on stochastic gradient descent, we show the runtime of the methods in one iteration in Table II. In practice, all the algorithms would converge in less than 100 iterations. Compared to BPR, SocailMF and SR need more runtime by adding the social network information. NeuMF leverages both the shallow linear model and deep networks with separate parts; thus, it costs more time than our simplified model of CNR that organically combines the shallow and deep structure in a unified framework. CNSR needs to calculate both the social embedding of users and the parameters in the collaborative neural architecture; it costs the most time. Specifically, CNSR_L is more time efficient than CNSR_J, as we do not update the social embedding part in a loosely training approach. Instead, CNSR_J jointly optimizes the social embedding part in the loss function, and it usually costs about 2 to 3 times as much as CNSR_L. Last but not least, the runtime of Douban is larger than Flixster on all methods since the Douban dataset has more rating records and social links. Given the above analysis, we empirically conclude that all methods are very time efficient.

C. Parameter Analysis

We now study the performance of CNSR under different parameter settings. For page limit, we only show the results on NDCG metric with top-K value equals 10. The trend is similar on HR metric.

The Impact of Neural Architecture: As the deep hidden layers in CNSR controls the capacity in modeling the complex relationships between users and items, we show the effectiveness of the deep hidden architecture by varying the number of hidden layers as well as the capacity of each hidden layer with different sizes. The dimension of user and item embeddings D is set as the same size of the first hidden layer. Specifically, we vary the size of the first hidden layer and reduce half of the size for each following layer. For example, if the size of the first hidden layer is 64 and there are 3 hidden layers, then the sizes of the layer are $64 \rightarrow 32 \rightarrow 16$. Please note that, when $F = 0$, CNSR degenerates to a generalized matrix factorization model without any hidden layers. Table III shows the model performance with different neural architecture on both datasets with the two training methods of CNSR. As can be seen from this table, as we increase the hidden dimension size and stacking more layers in a reasonable range, the results are better.



(a) (b)

Fig. 4. NDCG@10 with different dropout ratio. (a) Flixster. (b) Douban.

For example, the performance improves quickly as we increase the hidden layer from 0 to 1, denoting the effectiveness of considering the complex deep architecture for modeling. The best results are achieved on both datasets if the first hidden layer size is 64 with 2 hidden layers. The results are encouraging and show the expressiveness of adopting deep neural networks for modeling the complex user–item interaction to enhance recommendation performance.

Impact of Dropout Ratio: Dropout is very important technique in deep neural networks to avoid overfitting. Fig. 4 shows the metric of NDCG@10 on both datasets with different dropout ratio ρ. For example, $\rho = 0.1$ means in each iteration, we randomly discard 10% of the deep neurons in the collaboration and hidden layers. On both datasets, as ρ increases, the performance on both datasets increases at first. For example, the improvement on the NDCG metric is more than 3.5% on Flixster and 4.5% on Douban as we increase the dropout ratio from 0 to 0.2. However, when ρ surpasses 0.2, the performance on both datasets decreases. Thus, on both datasets, we choose the dropout ratio ρ as 0.2.

Impact of Balance Parameter: There are two important balance parameters: α and β in the objective function (10). α controls the balance between the rating loss function and the social embedding loss function, and β determines the relative importance of the social correlation term in the social embedding loss function. The larger the α, the more weight on the social embedding loss. The larger the β, the more weight on the social correlation regularization of the social loss function. For each balance parameter, we adopt the strategy of fixing all the other parameters by varying the current parameter for evaluation.

Fig. 5 shows the performance with the varying parameter α (with $\beta = 0.3$ on Flixster and $\beta = 0.2$ on Douban). Please not

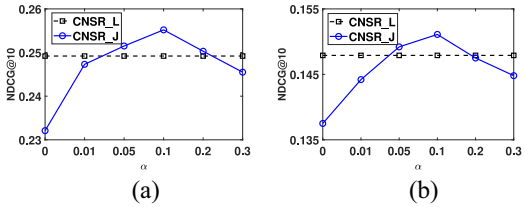


Fig. 5. Impact of parameter α. (a) Flixster. (b) Douban.

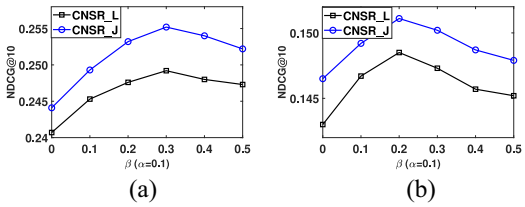


Fig. 6. Impact of parameter β. (a) Flixster. (b) Douban.

that, in the loosely training method (i.e., CNSR_L), we first learn the social embedding and then push the social embedding in the second part; thus, it is not sensitive to α. For CNSR_J, the overall trend is that it improves quickly at first, and it then drops at $\alpha = 0.1$ on both datasets. The reason is that, when $\alpha = 0$, CNSR_J degenerates to the CNR without any social network information. As we increase α from 0, we actually enhance the user embedding part with the social embedding. In contrast, if α is too large, the objective function would bias to the social embedding learning. Given the experimental finding, we set $\alpha = 0.1$ on both datasets.

Fig. 6 shows the performance with the varying parameter β (with $\alpha = 0.1$ on both datasets). As we increase β from 0 to larger values, we regularize the social embedding part with more social correlation constraints. The overall trend is similar to the trend of α, with the performance increasing at first and dropping at a certain value. Based on the results, we set $\beta = 0.3$ on Flixster and $\beta = 0.2$ on Douban.

V. Conclusion

In this paper, we developed a novel neural architecture CNSR that jointly incorporates the social network structure and user–item interaction in a unified model for social recommendation. Specifically, we proposed a social correlation-based interest embedding part that captures the social correlation among users' interest to strengthen the user embedding learning in a unified framework. We designed a newly proposed collaboration layer in the low layers of the proposed deep neural architecture to model the linear collaborative user–item interaction behavior in recommender systems. We also provided a joint learning framework to optimize all the parameters in the proposed CNSR model. Thus, the proposed model captured both the shallow collaborative and the complex deep relationships between users and items for recommendation in a unified framework. In the meantime, the social correlation of users' interests is seamlessly incorporated in this neural framework. Finally, extensive experimental results on two real-world datasets clearly showed the effectiveness of our proposed model.

References

[1] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.

[2] A. Anagnostopoulos, R. Kumar, and M. Mahdian, "Influence and correlation in social networks," in *Proc. KDD*, 2008, pp. 7–15.

[3] S. Aral, L. Muchnik, and A. Sundararajan, "Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks," *Proc. Nat. Acad. Sci. USA*, vol. 106, no. 51, pp. 21544–21549, 2009.

[4] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[5] J. Chen *et al.*, "Attentive collaborative filtering: Multimedia recommendation with item- and component-level attention," in *Proc. SIGIR*, 2017, pp. 335–344.

[6] H.-T. Cheng *et al.*, "Wide & deep learning for recommender systems," in *Proc. 1st Workshop Deep Learn. RecSys*, 2016, pp. 7–10.

[7] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for YouTube recommendations," in *Proc. Recsys*, 2016, pp. 191–198.

[8] L. Cui, J. Wu, D. Pi, P. Zhang, and P. Kennedy, "Dual implicit mining-based latent friend recommendation," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published.

[9] S. Deng, L. Huang, G. Xu, X. Wu, and Z. Wu, "On deep learning for trust-aware recommendations in social networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 5, pp. 1164–1177, May 2017.

[10] M. Eirinaki, M. D. Louta, and I. Varlamis, "A trust-aware system for personalized user recommendations in social networks," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 44, no. 4, pp. 409–421, Apr. 2014.

[11] A. M. Elkahky, Y. Song, and X. He, "A multi-view deep learning approach for cross domain user modeling in recommendation systems," in *Proc. WWW*, 2015, pp. 278–288.

[12] Y. Fu, G. Liu, M. Teng, and C. Aggarwal, "Unsupervised P2P rental recommendations via integer programming," in *Proc. KDD*, 2017, pp. 165–173.

[13] G. Guo, J. Zhang, and N. Yorke-Smith, "TrustSVD: Collaborative filtering with both the explicit and implicit influence of user trust and of item ratings," in *Proc. AAAI*, vol. 15, 2015, pp. 123–125.

[14] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "DeepFM: A factorization-machine based neural network for CTR prediction," in *Proc. IJCAI*, 2017, pp. 1725–1731.

[15] X. He and T.-S. Chua, "Neural factorization machines for sparse predictive analytics," in *Proc. SIGIR*, 2017, pp. 355–364.

[16] X. He *et al.*, "Neural collaborative filtering," in *Proc. WWW*, 2017, pp. 173–182.

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

[18] M. Jamali and M. Ester, "A matrix factorization technique with trust propagation for recommendation in social networks," in *Proc. RecSys*, 2010, pp. 135–142.

[19] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015, pp. 1–12.

[20] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proc. SIGKDD*, 2008, pp. 426–434.

[21] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105.

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

[24] G. Liu, Y. Fu, G. Chen, H. Xiong, and C. Chen, "Modeling buying motives for personalized product bundle recommendation," *ACM Trans. Knowl. Disc. Data*, vol. 11, no. 3, p. 28, 2017.

[25] Q. Liu, E. Chen, H. Xiong, C. H. Ding, and J. Chen, "Enhancing collaborative filtering by user interest expansion via personalized ranking," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 1, pp. 218–233, Feb. 2012.

[26] Q. Liu *et al.*, "A cocktail approach for travel package recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 2, pp. 278–293, Feb. 2014.

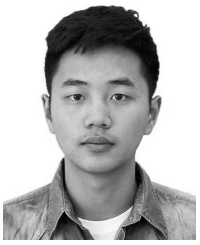
[27] H. Ma, H. Yang, M. R. Lyu, and I. King, "SoRec: Social recommendation using probabilistic matrix factorization," in *Proc. CIKM*, 2008, pp. 931–940.

[28] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King, "Recommender systems with social regularization," in *Proc. WSDM*, 2011, pp. 287–296.
[29] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annu. Rev. Sociol.*, vol. 27, no. 1, pp. 415–444, 2001.
[30] R. Miotto, F. Wang, S. Wang, X. Jiang, and J. T. Dudley, "Deep learning for healthcare: Review, opportunities and challenges," *Briefings Bioinformat.*, pp. 1–11, May 2017.
[31] R. R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *Proc. NIPS*, 2008, pp. 1257–1264.
[32] R. Pan *et al.*, "One-class collaborative filtering," in *Proc. ICDM*, 2008, pp. 502–511.
[33] X. Qian, H. Feng, G. Zhao, and T. Mei, "Personalized recommendation combining user interest and social circle," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 7, pp. 1763–1777, Jul. 2014.
[34] D. Rafailidis and A. Nanopoulos, "Modeling users preference dynamics and side information in recommender systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 46, no. 6, pp. 782–792, Jun. 2016.
[35] S. Rendle, "Factorization machines with libFM," *ACM Trans. Intell. Syst. Technol.*, vol. 3, no. 3, p. 57, 2012.
[36] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *Proc. UAI*, 2009, pp. 452–461.
[37] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "AutoRec: Autoencoders meet collaborative filtering," in *Proc. WWW*, 2015, pp. 111–112.
[38] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.
[39] P. Sun, L. Wu, and M. Wang, "Attentive recurrent social recommendation," in *Proc. SIGIR*, 2018, pp. 185–194.
[40] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. NIPS*, 2014, pp. 3104–3112.
[41] P. Symeonidis, "ClustHOSVD: Item recommendation by combining semantically enhanced tag clustering with tensor hosvd," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 46, no. 9, pp. 1240–1251, Sep. 2016.
[42] J. Tang, X. Hu, and H. Liu, "Social recommendation: A review," *Soc. Netw. Anal. Min.*, vol. 3, no. 4, pp. 1113–1133, 2013.
[43] J. Tang *et al.*, "LINE: Large-scale information network embedding," in *Proc. WWW*, 2015, pp. 1067–1077.
[44] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, Jan. 2010.
[45] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proc. SIGKDD*, 2011, pp. 448–456.
[46] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proc. SIGKDD*, 2016, pp. 1225–1234.
[47] H. Wang, N. Wang, and D.-Y. Yeung, "Collaborative deep learning for recommender systems," in *Proc. SIGKDD*, 2015, pp. 1235–1244.
[48] Q. Wang, J. Gao, and Y. Yuan, "A joint convolutional neural networks and context transfer for street scenes labeling," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 5, pp. 1457–1470, May 2018.
[49] Q. Wang, J. Wan, and Y. Yuan, "Deep metric learning for crowdedness regression," *IEEE Trans. Circuits Syst. Video Technol.*, to be published.
[50] S. Wang *et al.*, "What your images reveal: Exploiting visual contents for point-of-interest recommendation," in *Proc. WWW*, 2017, pp. 391–400.
[51] X. Wang *et al.*, "Community preserving network embedding," in *Proc. AAAI*, 2017, pp. 203–209.
[52] X. Wang, X. He, L. Nie, and T.-S. Chua, "Item silk road: Recommending items from information domains to social users," in *Proc. SIGIR*, 2017, pp. 185–194.
[53] L. Wu *et al.*, "Modeling the evolution of users' preferences and social links in social networking services," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 6, pp. 1240–1253, Jul. 2017.
[54] Y. Wu, C. DuBois, A. X. Zheng, and M. Ester, "Collaborative denoising auto-encoders for top-N recommender systems," in *Proc. WSDM*, 2016, pp. 153–162.
[55] C. Yang, L. Bai, C. Zhang, Q. Yuan, and J. Han, "Bridging collaborative filtering and semi-supervised learning: A neural approach for poi recommendation," in *Proc. KDD*, 2017, pp. 1245–1254.
[56] S. Zhai and Z. Zhang, "Dropout training of matrix factorization and autoencoder for link prediction in sparse graphs," in *Proc. SDM*, 2015, pp. 451–459.
[57] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, "Collaborative knowledge base embedding for recommender systems," in *Proc. SIGKDD*, 2016, pp. 353–362.
[58] H. Zhang, W. Ni, X. Li, and Y. Yang, "Modeling the heterogeneous duration of user interest in time-dependent recommendation: A hidden semi-Markov approach," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 2, pp. 477–494, Feb. 2018.
[59] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *arXiv preprint arXiv:1707.07435*, 2017.
[60] T. Zhao, J. McAuley, and I. King, "Leveraging social connections to improve personalized ranking for collaborative filtering," in *Proc. CIKM*, 2014, pp. 261–270.
[61] Z. Zhao, H. Lu, D. Cai, X. He, and Y. Zhuang, "User preference learning for online social recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 9, pp. 2522–2534, Sep. 2016.

Le Wu (M'18) received the Ph.D. degree in computer science from the University of Science and Technology of China, Hefei, China, in 2016.

She is currently a Faculty Member with the Hefei University of Technology, Hefei, China. She has published several papers in referred journals and conferences, such as IEEE Transactions on Knowledge and Data Engineering, ACM Transactions on Intelligent Systems and Technology, International Conference on Research and Development in Information Retrieval, AAAI Conference on Artificial Intelligence, International Joint Conference on Artificial Intelligence, ACM Knowledge Discovery and Data Mining, SIAM International Conference on Data Mining (SDM), and IEEE International Conference on Data Mining. Her current research interests include data mining, recommender system, and social network analysis.

Dr. Wu was a recipient of the Best of SDM 2015 Award and the Distinguished Dissertation Award from China Association for Artificial Intelligence.



Peijie Sun received the master's degree in signal and information processing from the Hefei University of Technology, Hefei, China, in 2018, where he is currently pursuing the Ph.D. degree in signal and information processing.

He has published a paper in SIGIR 2018. His current research interests include data mining and recommender systems.

Richang Hong (M'12) received the Ph.D. degree in signal and information processing from the University of Science and Technology of China, Hefei, China, in 2008.

He is currently a Professor with the Hefei University of Technology, Hefei. He was a Research Fellow with the School of Computing, National University of Singapore, Singapore, from 2008 to 2010. His current research interests include multimedia question answering, video content analysis, and pattern recognition. He has coauthored over 60 publications in the above areas.

Dr. Hong was a recipient of the Best Paper Award in the ACM Multimedia 2010. He is a member of the Association for Computing Machinery.

Yong Ge received the Ph.D. degree in information technology from the Rutgers, State University of New Jersey, New Brunswick, NJ, USA, in 2013.

He is an Assistant Professor of Management Information Systems with the University of Arizona, Tucson, AZ, USA. He has published prolifically in refereed journals and conference proceedings, such as the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, *ACM Transactions on Information Systems*, *ACM Transactions on Knowledge Discovery From Data*, ACM SIGKDD, SIAM International Conference on Data Mining, IEEE International Conference on Data Mining (ICDM), and ACM RecSys. His current research interests include data mining and business analytics.

Dr. Ge was a recipient of the ICDM-2011 Best Research Paper Award. He was also the Program Committee Member of ACM SIGKDD and IEEE ICDM.

Meng Wang (SM'17) received the B.E. and Ph.D. degrees in signal and information processing from the University of Science and Technology of China, Hefei, China, in 2003 and 2008, respectively.

He is a Professor with the Hefei University of Technology, Hefei. His current research interests include multimedia content analysis, computer vision, and pattern recognition. He has authored over 200 book chapters, journal and conference papers in the above areas.

Dr. Wang was a recipient of the ACM SIGMM Rising Star Award 2014. He is an Associate Editor of the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, and IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS.