IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 22, 2021, accepted December 22, 2021, date of publication January 11, 2022, date of current version January 18, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3141795

Towards Non-Linear Social Recommendation Using Gaussian Process

JIYONG ZHANG, (Member, IEEE), XIN LIU, AND XIAOFEI ZHOU
School of Automation, Hangzhou Dianzi University, Hangzhou 310018, China

Corresponding author: Xin Liu (lucianliuxin@gmail.com)

This work was supported in part by the Zhejiang Province Nature Science Foundation of China under Grant LZ22F020003, and in part by the HDU-CECDATA Joint Research Center of Big Data Technologies under Grant KYH063120009.

ABSTRACT Recent research on recommender systems has proved that by leveraging social network information, the quality of recommendations can be evidently improved. Traditional social recommendation models typically *linearly* combine social network information. For instance, matrix factorization based models linearly combine latent factors of relevant users and items. However, in practice, the multifaceted social relations are so complex that simple linear combination may not be able to reasonably organize such information for accurate social recommendation. On the other hand, existing deep learning based non-linear methods lack systematic modeling of user-item-friend relations. To handle these issues, we propose a novel, non-linear latent factor model for social recommendations leveraging Gaussian process. By introducing a social-aware covariance function, we organize individual users' past feedback, as well as the associated social information (e.g., friends' feedback to the same items) into a covariance matrix, which non-linearly and systematically learns the complex interactions among users, their interacted items and their friends' opinions. A stochastic gradient descent based optimization algorithm is developed to fit the model. Extensive experiments conducted on three real-world datasets demonstrate that the proposed model outperforms the state-of-the-art social recommendation models and Gaussian process based models.

INDEX TERMS Social recommendation, recommender systems, Gaussian process, social networks.

I. INTRODUCTION

By retrieving relevant items for individual users, recommender systems have become an important tool to handle information overload, which has been a serious issue due to the huge volume of data generated every moment on the Web [1]. For instance, e-commerce sites, e.g., eBay, Amazon have deployed recommendation engines to significantly increase revenues; online video sites, e.g., YouTube, Netflix rely on recommender systems to suggest potentially interesting videos from the (increasingly) large videos pool.

As the mainstream recommendation technique, collaborative filtering infers a user's preference based on her past consumption behavior, as well as that of other *similar* users. The recommendations are then made according to the inferred preference on the unseen items. Recently, online social networks like Facebook, Twitter and LinkedIn not only largely change the way people live, but also provide rich social network information, which can be treated as another dimension of input for building accurate recommender systems. The basic idea of social recommendation is to mimic people's behavior in physical world: we are likely to seek suggestions from our friends before making decisions [2], [3]. By integrating social network information into traditional recommender systems, a set of social recommendation models have been proposed to improve the performance of online recommendations [4]–[13].

Most existing social recommendation models typically linearly combine social information. For instance, memory based collaborative filtering (e.g., user-based collaborative filtering) learns the similarity between a user and her friends based on their past feedback behavior and then linearly combines friends' feedback to predict the target user's preference on the target item [14]; while model based collaborative filtering like Matrix Factorization (MF) linearly combine latent factors of relevant users, items and social relations [5], or impose regularization terms to (linearly) constrain the social influence [15]. However, in practice, social relations

The associate editor coordinating the review of this manuscript and approving it for publication was Fabrizio Messina.

are highly complicated where social connections are of different types (e.g., mutual-friends, follower-followee), different friends may have different tastes with the target user, and such taste difference is typically domain-aware (e.g., a friend who often suggests good restaurants may not be good at recommending clothing stores) and may vary over time. Therefore, the simple linear combination does not necessarily reasonably organize social information for recommendation in complex scenarios [16], or it might be effective by heavily relying on other side information, which is not always available [17]. Although efforts have been made to apply deep learning techniques for non-linear social recommendations [18]–[24], these methods basically model user-friend relations, user-item relations separately, and then force to merge the influences from different components via simple operations like embedding concatenation. For instance, in [21] the graph neural network based model consists of three components, i.e., user modeling part, item modeling part and rating prediction part. However, systematically and sophisticatedly modeling user-friend-item altogether simultaneously is relatively less explored in literature.

In this paper, we aim at breaking the limits of the linearity to improve social recommendations by capturing the complex interplay among users, items and friends' opinions systematically. To this end, we choose MF, one of the most popular recommendation techniques as the base model, and apply Gaussian process (GP) [25], [26], which is an advanced tool for modeling non-linear patterns to learn a non-linear latent function distribution to predict a user's preference on an item taking into account social network information. Particularly, for each user, we organize her past feedback into a covariance matrix where each element is the covariance of the corresponding pair of feedback. We propose a social-aware covariance function based on the well-known squared exponential covariance function where latent factors of the target user, items and the user's friends are used as input for covariance calculation. Furthermore, we design a mean function to capture the biases of the target user and the target item, which are important to model users' preference in recommender systems. Once the covariance function and the mean function are ready, we build the GP based model to learn the distribution of the latent function to infer users' preference in the presence of social network information. To fit the model, we develop a stochastic gradient descent (SGD) based optimization algorithm to learn hyperparameters of the covariance function, latent factors of users and items, as well as their bias factors. Note that due to its unique characteristics and complexity, we differentiate the social network information from the generic contextual information. Therefore, modeling other contexts such as time, emotion, location, etc. is out of the scope of this work.

The main contributions of this work are summarized as follows: (1) We endeavor to advance social recommendation research by breaking the limits of traditional linear social information combination using Gaussian process. Although GP has been used in collaborative filtering for traditional rating based preference inference [27]–[32], this work is the first attempt to apply GP for non-linear social recommendations. Moreover, our approach is able to handle both implicit feedback and explicit feedback. (2) We propose a novel social-aware covariance function to capture the complex interplay among the target user, her friends and items. We also introduce the mean function to take into account users' and items' biases for finer preference inference. (3) We design a SGD based optimization procedure to fit the proposed model. Complexity analysis shows that our model is able to near-linearly scale to the number of observations, thus is applicable to large-scale data. (4) We conduct extensive experiments using three real-world datasets to compare the proposed social recommendation model to the state-of-the-art methods.

The rest of the paper is organized as follows: In Section II, we review the related work about social recommendation and Gaussian process based preference inference in recommender systems. Section III introduces the preliminaries including a formal definition of social recommendation problem and a brief introduction of Gaussian process. We elaborate our GP based non-linear social recommendation model in Section IV where we describe the model specification, present the model fitting procedure and discuss the related issues in Section IV-A, IV-B and IV-C respectively. We report experimental results in Section V, followed by the conclusion and future research outline in Section VI.

II. RELATED WORK

A. SOCIAL RECOMMENDATION

Leveraging social network information to improve recommendation quality has become a hot topic. Here we summarized the related work according to methodology, i.e., neighbour-based, latent factor model based and deep learning based.

1) NEIGHBOUR-BASED METHODS

In [33], a neighborhood-based approach is proposed for social recommendations. The authors conducted comprehensive experiments to compare the performance of the social based and nearest-neighbor based recommendations. TrustWalker [14] employs random walk algorithm to seek relevant feedback over trust networks to alleviate cold start issue. Memory based collaborative filtering is then applied to linearly combine feedback for recommendation. In [34], distrust is explicitly modeled to improve trust-based recommendation models. Yu *et al.* [35] observed that users' explicit social relations may not be reliable for social recommendation, so they designed a method to adaptively identify credible implicit links over heterogeneous networks. Specifically, top-K most similar implicit friends with the target user are identified through the maximum expectation algorithm and the meta path based embedding representation learning. The implicit friends are then adaptively used through item ranking model for social recommendation.

Neighbour-based methods typically learn the similarity between users based on their past behavior and then linearly combines friends' feedback to predict the target user's preference. On the other hand, our approach utilizes Gaussian Process to capture complex non-linear social influence.

2) LATENT FACTOR MODEL BASED METHODS

For latent factor model based approaches, social network information is handled in three ways, i.e., co-factorization, ensemble and regularization [2]: (1) Co-Factorization. In [5], the authors proposed a probabilistic matrix factorization based approach to factorize both user-item-rating matrix and social relations matrix to fuse rating and social information. A similar approach is described in [36]. (2) Ensemble. The basic idea of this approach is to predict the target user's preference on an item by linearly combining friends' preferences, which are derived as the inner product of their and the target item's latent factors. The weight of each friend's preference can be predefined or learned from the data. (3) Regularization. Ma *et al.* [15] introduced the social regularization on the basis of matrix factorization to constrain the taste difference between a user and her friends. Two variants are proposed: (a) average-based regularization that minimizes the difference between a user's latent factors and the average of that of her friends; (b) individual-based regularization that focuses on latent factor vector difference between a user and each of her friends. This work also compared the performance of different similarity measures, i.e., Vector Space Similarity and Pearson Correlation Coefficient (PCC). In [17], with rich contextual information, the authors used the similar social regularization to incorporate social information. A context-aware PCC is proposed to measure the similarity between users' preference. Wang *et al.* [37] utilized social information to model user exposures on items in the modular way of social regularization and social boosting and then generated social recommendation by combining collaborative filtering model, which alleviated the assumption that people must share similar preferences with their social friends.

Latent factor model based methods rely on inner product to capture users' preference considering social influence. Although various contextual information can be integrated, linear combination of latent factors affects the effectiveness in complex scenarios. While our approach endeavors to break the limits of linear modeling using Gaussian Process.

3) DEEP LEARNING BASED METHODS

Recent trends on deep learning brought another line of research. Wang *et al.* [38] extended neural collaborative filtering (NCF) [16] model by combining with graph regularization technology, and developed a neural social collaborative ranking model (NSCR) to solve the problem of cross-domain social recommendation. Different from the previous static models that only leverage local social neighbors of users in social recommendations, Wu *et al.* [18] designed a layer-wise recursive social influence diffusion neural network, which can better model the recursive dynamic social diffusion and item embedding. Song *et al.* [19] proposed a model combining the advantages of RNN and graph-attention neural network to solve the problem that users' interests are dynamic and easily influenced by friends in online communities. Fan *et al.* [21] introduced a social recommendation model based on graph neural network. The model comprises of three components: user modeling that can learn user latent factors from the combination of social graph and user-item graph, item modeling that can learn item latent factors by jointly capturing interactions and opinions in the user-item graph, and rating prediction for generating social recommendations. In [39], a dynamic social aware recurrent neural network and a static social attention network were proposed to model users' dynamic preferences and static preferences respectively. Then the dynamic part and static part were combined to generate temporal social recommendations.

Existing deep learning based methods mostly model user-friend relations and user-item interactions separately and then combine different components via simple operations like concatenation. On the other hand, our approach systematically and sophisticatedly models user-friend-item altogether and provides an end-to-end architecture.

B. GAUSSIAN PROCESS FOR RECOMMENDATION

Gaussian process has been applied to learn users' preference in collaborative filtering. Lawrence *et al.* [40] developed a non-linear extension of matrix factorization. The authors first proved that probabilistic matrix factorization is equivalent to probabilistic principal component analysis, and then extended the model in a non-linear fashion by applying Gaussian process to learn a non-linear latent function. Platt *et al.* [28] applied Gaussian process for regression to learn a user's preference over music. An algorithm, Kernel Meta-Training (KMT) was proposed to derive a kernel from a set of meta-training functions which share the same function distribution with the final training function. Based on KMT, a system AutoDJ was designed to automatically generate music playlists based on the seed songs selected by a user. In [29], Gaussian process and collaborative filtering were combined to learn pair-wise preferences expressed by multiple users. Specifically, the task of learning users' preference was treated as a binary classification with Gaussian process where a preference kernel is used. Bonilla *et al.* [31] focused on generalizing the knowledge of known users to infer the preference of unknown users. Gaussian process prior is applied over users' latent utility functions to learn the similarity of users' preference which can be used to aid in the elicitation process for a new user.

Beyond feedback information, a few GP based approaches have been proposed to incorporate other available information for recommendation. In [41], the authors proposed a framework to incorporate side information by coupling multiple probabilistic matrix factorization via Gaussian process priors, where the latent features are replaced by latent feature functions. GPFM [42] tries to improve context-aware recommendations by relaxing the assumption of linear

combination of latent factors of users, items and contexts like in Factorization Machines [43]. The authors assumed that the roles of items and contexts are equivalent and then introduced the utility of every <item, contexts> pair measured by the function of the corresponding latent factor representations. GP is applied to learn such a utility function. The authors also proposed variants to handle both explicit feedback and implicit feedback.

III. PRELIMINARIES

In this section, we first formally define the social recommendation problem and then provide a short introduction of GP before presenting our social recommendation model in Section IV.

A. SOCIAL RECOMMENDATION

Social recommendation emerges as the combination of two concepts: social networks and traditional recommender systems. Broadly speaking, social recommendation covers every aspect of recommendation in social networks including recommending any objects such as friends, events, tags, groups, etc., [44] or utilizing any information available in social networks for such recommendations (e.g., feedback, textual contents) [45]; narrowly speaking, social recommendation refers to recommendations that are made by leveraging users's social relations, e.g., the connected users tend to share the similar tastes [46]. In this paper, we focus on the narrow definition of social recommendation, which is more straightforward to help us better understand social recommendation and its limitations like linear combination of social information.

In a social recommender system, a set of users $\mathcal{U}$ are connected to each other (e.g., become mutual friends like in Facebook or follow users like in Twitter). We denote $\mathcal{S}$ as user-user social relations, where $S_{i,j} = 1$ if user u_i and user u_j are connected, and $S_{i,j} = 0$ otherwise (see Fig. 1(a)). Any user can consume any item from item set $\mathcal{V}$ based on her preference, and the user's feedback to the interactions can be measured by a variety of means. For instance, the feedback could be binary, e.g., bookmark or not in Delicious.com; on the other hand, some applications like Netflix employ five-point likert scale to measure a user's preference on an item. We denote such feedback by $\mathcal{R}$ (see Fig. 1(b) where '?' indicates the user does not interact with the item).

Typically, social recommendation has two inputs: social relations $\mathcal{S}$ and feedback $\mathcal{R}$. The goal of social recommendation models is to learn a function φ_s that can infer a user's preference on an item, so mathematically, social recommendation model can be defined as:

$$r_{u_i,v_j} = \varphi_s(u_i, v_j|\mathcal{S}, \mathcal{R}), \tag{1}$$

where u_i is the target user and v_j is the target item. Recommendations can then be made according to the predicted preference (in descending order).

Most existing social recommendation models design φ_s as the linear combination of social information. For instance, memory based CF infers a user's preference as the weighted sum of feedback from friends (or the ones that are trustworthy from the perspective of the target user) where the weight is determined based on users' past feedback behavior [14]. For model based CF, e.g., latent factor models, φ_s is derived by linearly combining users' and items' latent factors, and social information is incorporated via co-factorization [5] or regularization [15], [17]. As argued before, such linear combination does not necessarily well organize the social information, so the goal of this work is to remove the constraints of linearity and learn a non-linear function φ_s to improve social recommendations using GP, which will be introduced in the next subsection.

	u_1	u_2	u_3	u_4	u_5
u_1	1	0	1	0	1
u_2	0	1	0	0	0
u_3	1	0	1	1	0
u_4	0	0	1	1	0
u_5	1	0	0	0	1

(a) Social relation matrix $\mathcal{S}$.

	v_1	v_2	v_3	v_4	v_5
u_1	?	?	3	?	2
u_2	?	?	5	?	?
u_3	2	4	?	1	2
u_4	?	?	?	3	?
u_5	1	?	3	2	3

(b) Feedback matrix $\mathcal{R}$ (five-point likert scale rating as an example).

FIGURE 1. An example of input for social recommendations.

B. GAUSSIAN PROCESS

Gaussian processes (GP) [25], [26] are a powerful tool for modeling function distributions from data. Instead of directly fitting observation data, GP relies on a Bayesian approach to learn a posterior distribution over functions. In other words, GP can be treated as the extension of the Gaussian distribution to functions. Mathematically, a Gaussian process $p(\varphi)$' defines a distribution over functions, where φ is a function that maps the input space $\mathcal{X}$ to $\Re$, i.e., $\varphi : \mathcal{X} \to \Re$. Let $\varphi = (\varphi(x_1), \varphi(x_2), \ldots, \varphi(x_n))$ denotes an n-dimensional vector of function values at the corresponding n points $x_i \in \mathcal{X}$. Formally, $p(\varphi)$ is a *Gaussian process* if for any finite subset of $\{x_1, x_2, \ldots, x_n\}$, the marginal distribution over that finite subset $p(\varphi)$ follows a multivariate Gaussian distribution [25].

A Gaussian process is fully specified by a mean function:

$$\mu(x) = \mathbb{E}[\varphi(x)], \tag{2}$$

and a covariance function:

$$k(x, x') = \mathbb{E}[(\varphi(x) - \mu(x))(\varphi(x') - \mu(x'))], \tag{3}$$

Once the mean function $\mu(x)$ and covariance function $k(x, x')$ are defined, one can use Gaussian process for regression, i.e., predicting the value of the next observations. Specifically, Gaussian process regression is a Bayesian approach which assumes a Gaussian process prior over functions, i.e., according to Eq. 4

$$p(\varphi|\mathbf{X}, \Theta) = \mathcal{N}(\mu, \Sigma), \tag{4}$$

where μ is the mean function and Σ is the covariance matrix depending on the input $\mathbf{X} = \{x_1, x_2, \ldots, x_n\}$ and some hyperparameters Θ. The (i,j)*th* element of Σ is the covariance of the function values at the point of x_i and x_j, obtained by the covariance function $k(x_i, x_j)$. When making prediction (without the consideration of any noise), the joint distribution of the training output φ and the test output φ' is:

$$\begin{bmatrix} \varphi \\ \varphi' \end{bmatrix} \sim \mathcal{N}(\begin{bmatrix} \mu(\mathbf{X}) \\ \mu(\mathbf{X}') \end{bmatrix}, \begin{bmatrix} K(\mathbf{X}, \mathbf{X}) & K(\mathbf{X}, \mathbf{X}') \\ K(\mathbf{X}', \mathbf{X}) & K(\mathbf{X}', \mathbf{X}') \end{bmatrix}), \tag{5}$$

where $\mathbf{X}'$ represents the n' test points, $\mu(\mathbf{X}')$ is the mean function of the test points and $K(\mathbf{X}, \mathbf{X}')$ denotes the matrix $(n \times n')$ of the covariances of all pairs of training and test points. We will elaborate how the test φ' can be predicted in the context of modeling users' preference taking into account social network information in the next section.

IV. SOCIAL RECOMMENDATIONS WITH GAUSSIAN PROCESS

A. NON-LINEAR SOCIAL RECOMMENDATION MODEL

Having defined social recommendation problem and introduced GP, we present our GP based non-linear model for social recommendations. Our model is built on top of latent factor models so we assign a latent factor vector $\mathbf{u}_i$ to each user u_i and a latent factor vector $\mathbf{v}_j$ to each item v_j. Note that users' and items' latent factor vectors share the same dimensionality d.

For each user u_i, we assume the system records her past interacted items V_i and the associated (binary or numerical) feedback. We denote friend set of this user by $\mathbf{F}_i = \{\mathbf{f}_{i,1}, \mathbf{f}_{i,2}, \ldots\}$. Note that $\mathbf{F}_i$ records the latent factor vector of every friend (i.e., $\mathbf{f}_{i,x}$ represents latent factor vector of u_i's xth friend). Let $R_{i,j} =< \mathbf{u}_i, \mathbf{v}_j, F_i^j >$ represents an interaction between u_i and v_j, where F_i^j is a subset of $\mathbf{F}_i$ ($F_i^j \subseteq \mathbf{F}_i$), and each element of F_i^j is the friend that also interacted with item v_j. In case no friends have interacted with v_j (i.e., $F_i^j = \Phi$), $R_{i,j}$ is simply represented by the latent factors of the corresponding user and item. Accordingly, we denote all interactions of u_i by $\mathbf{R}_i$, and the corresponding feedback set is denoted by $\mathbf{r}_i$. Note that our social recommendation model is generic in that it can handle both implicit feedback and explicit feedback (e.g., five-point likert scale ratings).

The purpose of our model is to learn a latent function φ_s^i for each user u_i that maps the latent factors of the user, her friends and the interacted items, i.e., $\mathbf{R}_i$ to the feedback values $\mathbf{r}_i$:

$$\varphi_s^i : \mathbf{R}_i \rightarrow \mathbf{r}_i, \tag{6}$$

In order to take into account the noise, which is common in practice, we assume an additive independent identically distributed Gaussian noise ϵ, with variance σ^2:

$$\mathbf{r}_i = \varphi_s^i(\mathbf{R}_i) + \epsilon, \tag{7}$$

We then apply Gaussian process regression to infer the distribution of φ_s^i to predict user u_i's preference on items that u_i has not interacted, taking into account the associated social network information, i.e., $\mathbf{r}_i' = \varphi_s^i(\mathbf{R}_i') + \epsilon$. The joint distribution of the observed feedback values and the feedback to be predicted is obtained, according to the theory that the values of the function to be learnt follow multivariate Gaussian distribution:

$$\begin{bmatrix} \varphi_s^i \\ \varphi_s^{i'} \end{bmatrix} \sim \mathcal{N}(\begin{bmatrix} \mu(\mathbf{R}_i) \\ \mu(\mathbf{R}_i') \end{bmatrix}, \begin{bmatrix} K(\mathbf{R}_i, \mathbf{R}_i) + \sigma^2 \mathrm{I} & K(\mathbf{R}_i, \mathbf{R}_i') \\ K(\mathbf{R}_i', \mathbf{R}_i) & K(\mathbf{R}_i', \mathbf{R}_i') \end{bmatrix}), \tag{8}$$

where I is identity matrix, $\mu(.)$ is the mean function and K is the covariance matrix where the (r, c)th element of K represents the covariance evaluated at the rth and cth interaction (i.e., feedback) of $\mathbf{R}_i$, i.e., $k(R_{i,r}, R_{i,c})$. Accordingly, the elements of $K(\mathbf{R}_i, \mathbf{R}_i)$, $K(\mathbf{R}_i, \mathbf{R}_i')$ and $K(\mathbf{R}_i', \mathbf{R}_i')$ represent the covariances evaluated at all feedback pairs from $\mathbf{R}_i$, $\mathbf{R}_i$ and $\mathbf{R}_i'$, and $\mathbf{R}_i'$ respectively.

By conditioning the joint Gaussian prior distribution on the observed interactions, we derive the probability of $\varphi_s^{i'}$ (which also follows Gaussian distribution):

$$\begin{aligned} &\varphi_s^{i'}|\varphi_s^i, \mathbf{R}_i, \mathbf{R}_i' \\ &\quad \sim \mathcal{N}(\mu(\mathbf{R}_i') + K(\mathbf{R}_i', \mathbf{R}_i)(K(\mathbf{R}_i, \mathbf{R}_i) \\ &\qquad +\sigma^2 I)^{-1}(\varphi_s^i - \mu(\mathbf{R}_i)), K(\mathbf{R}_i', \mathbf{R}_i') - K(\mathbf{R}_i', \mathbf{R}_i) \\ &\qquad \cdot(K(\mathbf{R}_i, \mathbf{R}_i) + \sigma^2 I)^{-1} K(\mathbf{R}_i, \mathbf{R}_i')), \end{aligned} \tag{9}$$

Naturally, the best estimate of the latent function $\varphi_s^{i'}$ is the mean of its distribution:

$$\bar{\varphi_s^{i'}} = \mu(\mathbf{R}_i') + K(\mathbf{R}_i', \mathbf{R}_i)(K(\mathbf{R}_i, \mathbf{R}_i) + \sigma^2 I)^{-1}(\varphi_s^i - \mu(\mathbf{R}_i)), \tag{10}$$

An advantage of GP is that along with the prediction, it also provides the associated variance, which can be used to measure the confidence of the prediction:

$$\begin{aligned} var(\varphi_s^{i'}) = K(\mathbf{R}_i', \mathbf{R}_i') - K(\mathbf{R}_i', \mathbf{R}_i)(K(\mathbf{R}_i, \mathbf{R}_i) + \sigma^2 I)^{-1} \\ \cdot K(\mathbf{R}_i, \mathbf{R}_i')), \end{aligned} \tag{11}$$

From Eq. 10 and 11 we can see that Gaussian process regression is fully specified by the covariance function and the mean function. In the next subsections, we present how to design these two functions to realize non-linear social recommendation.

1) COVARIANCE FUNCTION

There are numerous covariance functions which make GP flexible in different application scenarios. To start with, we choose the most well known one called squared exponential covariance function:

$$k(x, x') = h^2 \exp(-\frac{(x - x')^2}{2\lambda^2}), \tag{12}$$

where h and λ are hyperparameters that control the scale (or variance) of the function output and the length-scale of the input x respectively.

Based on squared exponential function, we propose our social-aware covariance function to measure the covariance of two interactions $R_{i,v}$ and $R_{i,v'}$ of u_i by combining the covariance between items (v and v') and the effects of users' friends (i.e., covariance between the target user and every relevant friend):

$$\begin{aligned} k_i(R_{i,v}, R_{i,v'}) &= k(\mathbf{v}_v, \mathbf{v}_{v'}) \prod_{\mathbf{f} \in F_i^v \cap F_i^{v'}} k(\mathbf{u}_i, \mathbf{f}) \\ &= h^2 \exp(-\frac{(\mathbf{v}_v - \mathbf{v}_{v'})^\top (\mathbf{v}_v - \mathbf{v}_{v'})}{2\lambda^2}) \\ &\quad \times \prod_{\mathbf{f} \in F_i^v \cap F_i^{v'}} h^2 \exp(-\frac{(\mathbf{u}_i - \mathbf{f})^\top (\mathbf{u}_i - \mathbf{f})}{2\lambda^2}) \\ &= h^2 \exp(-\frac{Q}{2\lambda^2}), \end{aligned} \tag{13}$$

where $Q = (\mathbf{v}_v - \mathbf{v}_{v'})^\top (\mathbf{v}_v - \mathbf{v}_{v'}) + \sum_{\mathbf{f} \in F_i^v \cap F_i^{v'}} (\mathbf{u}_i - \mathbf{f})^\top (\mathbf{u}_i - \mathbf{f})$. So the new social-aware covariance function is actually in the form of squared exponential kernel but with more sophisticated organization of latent factors of the target user, her friends and target item pairs. Note that not every interacted item has feedback information from u_i's friends. That is, if $F_i^v \cap F_i^{v'} = \Phi$, the covariance function only captures the covariance of the corresponding pair of items ($\mathbf{v}_v$ and $\mathbf{v}_{v'}$). Once the social-aware covariance function is designed, we can calculate u_i's covariance matrix, which non-linearly captures the interplay among users (u_i and her friends) and items. Fig. 2 shows an example of u_i's covariance matrix.

Note that we are not claiming that squared exponential covariance function is the best base function for designing social-aware covariance function. We leave as a future work a more detailed discussion on the tradeoff between the more sophisticated covariance function (e.g., neural network covariance function) and the associated extra computational overheads.

Through covariance function, latent factors of the target user, the items and the corresponding friends are thoroughly interacted and modeled, where the importance of each friend's opinions to the target user's preference in the corresponding scenario is automatically learned and reflected by the latent factors.

$K(\mathbf{R}_i, \mathbf{R}_i) =$

	$R_{i,1}$	$R_{i,2}$	$R_{i,3}$	.....	$R_{i,m}$
$R_{i,1}$	$k_i(R_{i,1}, R_{i,1})$	$k_i(R_{i,1}, R_{i,2})$	$k_i(R_{i,1}, R_{i,3})$	.....	$k_i(R_{i,1}, R_{i,m})$
$R_{i,2}$	$k_i(R_{i,2}, R_{i,1})$	$k_i(R_{i,2}, R_{i,2})$	$k_i(R_{i,2}, R_{i,3})$	.....	$k_i(R_{i,2}, R_{i,m})$
$R_{i,3}$	$k_i(R_{i,3}, R_{i,1})$	$k_i(R_{i,3}, R_{i,2})$	$k_i(R_{i,3}, R_{i,3})$	.....	$k_i(R_{i,3}, R_{i,m})$
	.....	.....	.....	.....	.....
$R_{i,m}$	$k_i(R_{i,m}, R_{i,1})$	$k_i(R_{i,m}, R_{i,2})$	$k_i(R_{i,m}, R_{i,3})$	.....	$k_i(R_{i,m}, R_{i,m})$

FIGURE 2. An example of u_i's covariance matrix $K(\mathbf{R}_i, \mathbf{R}_i)$ constructed by u_i's m interactions using the proposed social-aware covariance function.

2) MEAN FUNCTION

In recommender systems, users' and items' biases, which are independent of an interactions often play an important role in preference inference [47]. For instance, some users prefer giving high ratings to items, while some other users prefer assigning low ratings; For items, there are always some items that get higher ratings or attract more visits than others. In order to incorporate such biases, we assign a latent bias variable b_i to each user u_i, and a latent bias variable b_j to each item v_j. The mean function of an interaction between u_i and v_j is then designed in the form of biases combination:

$$\mu_i(j) = b_i^u + b_j^v, \tag{14}$$

According to Eq. 10, such a mean function, in addition to latent factors based inference, adds bias factors to learn the function φ_s. We will demonstrate the effectiveness of biases in the evaluation section.

3) RECOMMENDATION

Once the social-aware covariance function and the bias based mean function have been designed, we are able to predict the target user u_i's preference on the set of unseen items using Eq. 10. Essentially, the predicted preference is the weighted combination of u_i's preference on the past items where the weight is determined by non-linearly learning the interplays among u_i, the interacted items and the relevant friends (i.e., by covariance matrix). Note that each unseen item v' is firstly converted to latent representation $R_{i,v} =< \mathbf{u}_i, \mathbf{v}', F_i^{v'} >$. Using Eq. 11, we can decide whether to accept the predicted preference or not.

The final top-N recommendation list is produced in the descending order of the predicted preference on the unseen items.

B. MODEL FITTING

The effectiveness of Gaussian process largely depends on the proper choice of covariance function hyperparameters (i.e., h, λ and σ), as well as the introduced latent factors and biases of users and items. In this section, we develop a SGD based optimization algorithm to fit our non-linear social recommendation model.

According to the definition of Gaussian process, the distribution of observed values follows multivariate Gaussian distribution (for user u_i):

$$p(\mathbf{r}_i|\mathbf{R}_i,\Theta_i) = (2\pi)^{-\frac{|\mathbf{R}_i|}{2}}|K(\mathbf{R}_i,\mathbf{R}_i)|^{-\frac{1}{2}} \cdot e^{-\frac{1}{2}(\mathbf{r}_i-\mu_i)^\top K(\mathbf{R}_i,\mathbf{R}_i)^{-1}(\mathbf{r}_i-\mu_i)}, \quad (15)$$

where μ_i represents the values calculated by the mean function (see Eq. 14), and $|.|$ denotes the cardinality of a set. Then the negative log marginal likelihood is obtained:

$$\begin{aligned} L &= -\log p(\mathbf{r}_i|\mathbf{R}_i,\Theta_i) \\ &= \frac{|\mathbf{R}_i|}{2}\log 2\pi + \frac{1}{2}\log|K(\mathbf{R}_i,\mathbf{R}_i)| \\ &\quad + \frac{1}{2}(\mathbf{r}_i-\mu_i)^\top K(\mathbf{R}_i,\mathbf{R}_i)^{-1}(\mathbf{r}_i-\mu_i), \end{aligned} \quad (16)$$

With Eq. 16, model fitting becomes an optimization problem of minimizing the negative log marginal likelihood L. We use SGD to estimate the parameters Θ by seeking the partial derivatives of L with respect to the covariance hyperparameters, latent factors and bias factors. Each parameter $\theta \in \Theta$ is updated iteratively, until the predefined maximum number of iterations has reached or the negative log marginal likelihood converges:

$$\theta \leftarrow \theta - \alpha\frac{\partial L}{\theta}, \quad (17)$$

where α is the learning rate. Next, we provide details of derivatives of L with respect to individual parameters.

We first derive the derivatives of L with respect to users' and items' biases:

$$\frac{\partial L}{\partial\theta} = (\mathbf{r}_i-\mu_i)^\top K(\mathbf{R}_i,\mathbf{R}_i)^{-1}\frac{\partial\mu_i}{\partial\theta}, \quad (18)$$

where $\theta = b_i^u$ or b_j^v, and $\frac{\partial\mu_i}{\partial b_i} = \frac{\partial\mu_i}{\partial b_j} = 1$.

Then the derivatives of L with respect to covariance function hyperparameters (i.e., h, λ and σ) and the latent factors of users and items are derived as follows:

$$\begin{aligned} \frac{\partial L}{\partial\theta} &= \frac{1}{2}tr(K(\mathbf{R}_i,\mathbf{R}_i)^{-1}\frac{\partial K(\mathbf{R}_i,\mathbf{R}_i)}{\partial\theta}) - \frac{1}{2}(\mathbf{r}_i-\mu_i)^\top K \\ &\quad \cdot(\mathbf{R}_i,\mathbf{R}_i)^{-1}\frac{\partial K(\mathbf{R}_i,\mathbf{R}_i)}{\partial\theta}K(\mathbf{R}_i,\mathbf{R}_i)^{-1}(\mathbf{r}_i-\mu_i), \end{aligned} \quad (19)$$

From Eq. 19 we notice that in order to obtain the final derivatives, we also need to calculate the derivative of covariance matrix with respect to each parameter. Eq. 20, Eq. 21 and Eq. 21 show such derivatives with respect to covariance hyperparameters (i.e., $\theta = h$, λ or σ).

$$\frac{\partial K(\mathbf{R}_i,\mathbf{R}_i)}{\partial h} = 2h\exp(-\frac{Q}{2\lambda^2}), \quad (20)$$

$$\frac{\partial K(\mathbf{R}_i,\mathbf{R}_i)}{\partial\lambda} = h^2\exp(-\frac{Q}{2\lambda^2})\frac{Q}{\lambda^3}, \quad (21)$$

and

$$\frac{\partial K(\mathbf{R}_i,\mathbf{R}_i)}{\partial\sigma} = 2\sigma I, \quad (22)$$

where $Q = (\mathbf{v}_v-\mathbf{v}_{v'})^\top(\mathbf{v}_v-\mathbf{v}_{v'}) + \sum_{f\in F_i^v\cap F_i^{v'}}(\mathbf{u}_i-\mathbf{f})^\top(\mathbf{u}_i-\mathbf{f})$.

Next, the derivative of covariance matrix with respect to the target user u_i's latent factors (the dth factor of latent factor vector $\mathbf{u}_i$) is:

$$\frac{\partial K(\mathbf{R}_i,\mathbf{R}_i)}{\partial\mathbf{u}_{i,d}} = h^2\exp(-\frac{Q}{2\lambda^2})\frac{\sum_{\mathbf{f}\in F_i^v\cap F_i^{v'}}(\mathbf{f}_d-\mathbf{u}_{i,d})}{\lambda^2}, \quad (23)$$

Similarly, the derivatives of covariance matrix with respect to latent factors of the first item v, the second item v' and each relevant[1] friend are obtained:

$$\frac{\partial K(\mathbf{R}_i,\mathbf{R}_i)}{\partial\mathbf{v}_d} = h^2\exp(-\frac{Q}{2\lambda^2})\frac{\mathbf{v}'_d-\mathbf{v}_d}{\lambda^2}, \quad (24)$$

$$\frac{\partial K(\mathbf{R}_i,\mathbf{R}_i)}{\partial\mathbf{v}'_d} = h^2\exp(-\frac{Q}{2\lambda^2})\frac{\mathbf{v}_d-\mathbf{v}'_d}{\lambda^2}, \quad (25)$$

$$\frac{\partial K(\mathbf{R}_i,\mathbf{R}_i)}{\partial\mathbf{f}_d} = h^2\exp(-\frac{Q}{2\lambda^2})\frac{\mathbf{u}_{i,d}-\mathbf{f}_d}{\lambda^2}, \quad (26)$$

where $Q = (\mathbf{v}_v-\mathbf{v}_{v'})^\top(\mathbf{v}_v-\mathbf{v}_{v'}) + \sum_{f\in F_i^v\cap F_i^{v'}}(\mathbf{u}_i-\mathbf{f})^\top(\mathbf{u}_i-\mathbf{f})$.

C. DISCUSSION

The computational complexity of our proposed social recommendation model is mainly determined by the inversion of a matrix, for which the standard methods require time $\mathcal{O}(\tilde{n}^3)$ for a $\tilde{n}\times\tilde{n}$ matrix. Note that $\tilde{n}$ is the averaged number of the interacted items per user. This can be improved by applying faster matrix multiplication method such as Coppersmith-Winograd algorithm [48] or Cholesky decomposition. Alternatively, approximation techniques like variational Bayesian inference can be applied to accelerate the learning process. Besides the cost of matrix inversion, the cost of taking derivatives with respect to parameters is determined by the number of parameters to be learned (for a single user): $\mathcal{O}(\tilde{n}^2\times((\tilde{n}+|\tilde{F}|+u)d+u_b+\sum^{\tilde{n}}i_b+n_k))$, where $\tilde{n}^2$ is the cost of matrix multiplication, $|\tilde{F}|$ is the averaged number of friends (that interacted with the same items with the target user) per user, $u=1$ and $u_b=1$ indicate the target user herself and her bias, $\sum^{\tilde{n}}i_b$ represents the items' biases ($i_b=1$), $n_k=3$ is the number of covariance function hyperparameters. When considering all users, the total computational complexity is $\mathcal{O}(|\mathcal{U}|\tilde{n}^3+|\mathcal{U}|\tilde{n}^2\times((\tilde{n}+|\tilde{F}|)d))$. Note that although a user may have thousands of social connections, only a few of them share the commonly interacted items with the target user, so $|\tilde{F}|$ is typically very small. The final cost is $\mathcal{O}(|\mathcal{R}|\tilde{n}^2d)$, where $|\mathcal{R}|=|\mathcal{U}|\times\tilde{n}$ is the number of all observed feedback. $\tilde{n}$ largely depends on the activity of users, but in most applications, $\tilde{n}$ is reasonably small and follow a long-tail distribution. Furthermore, we may apply sampling techniques to only keep the most informative interactions for training, thus further decreasing $\tilde{n}$. Regarding the dimensionality d of latent factor vector, it is typically small (e.g., 5 - 10) due to the high modeling capacity of GP. Given that both $\tilde{n}$ and d are small, and $\tilde{n}$ increases slowly with the increasing $|\mathcal{R}|$ due to the large user set $\mathcal{U}$, the computational complexity of our

[1]Note that we only update latent factors of the friends who have interacted with the same items with the target user.

social recommendation model is *near-linear* in the number of total observed feedback $\mathcal{R}$.

Besides computational complexity, another issue we want to mention is the well-known data sparsity. Like any other social recommendation models that rely on the information of interactions among users, their friends and items, our model also assumes a user has sufficient historical information for model building. We will demonstrate in the evaluation section how the size of training data (i.e., the number of past interactions) influences the performance of our model.

Finally, we want to emphasize that although our GP based model is designed for social recommendations, in case social information is not available, e.g., movie recommendation in MovieLens, our model is still applicable to model the non-linear interactions between users and items. Experiments results (see Section V-B2) show that such non-linear modeling indeed outperforms traditional linear models.

V. EVALUATION

A. EXPERIMENTAL SETTINGS

In this section, we conduct experiments with the aim of answering the following research question: (RQ1) Does Gaussian Process based non-linear approach effectively improves social recommendation quality? (RQ2) Does our proposed social recommendation model outperforms the state-of-the-art methods?

1) DATASETS

The experiments were conducted over three real-world datasets. The first dataset was collected from bookmarking service site Delicious (http://www.delicious.com). The dataset [49] contains 1,867 users who bookmarked 69,226 URLs. There are 104,799 unique (user, URL) pairs. On average, each user bookmarked 56.132 URLs and each URL was bookmarked by 1.514 users. There are totally 15,328 social connections, i.e., (u_i, u_j) pairs, and each user has 8.236 connections on average. The task of a recommendation model is to recommend the URLs that the target user is likely to bookmark.

The second dataset was collected from Last.fm (http://www.lastfm.com). The dataset [49] records the information about 1,892 users' listening history with 17,632 music artists. There are 92,834 unique (user, artist) pairs. On average, each user listened to 49.067 artists, and each artists were listened by 5.265 users. There are 25,434 social connections and each user has 13.443 friend relations on average. The task of a recommendation model is to recommend the artists that the target user is likely to listen to.

The last dataset was from Epinions (http://www.epinions.com), which records 49,290 users who reviewed 139,738 products [50]. There are 664,824 reviews and 487,181 trust statements (i.e., unidirectional social relations). The task of a recommendation model is to recommend products for users to review.

Note that in Delicious data and Last.fm data, the feedback is binary, i.e., bookmarking a URL or not, listening to an artist or not. In Epinions data, the feedback is in five-point likert scale where 1 for worst and 5 for best. For each dataset, all user-item interactions are sorted in chronological order; the first 80% of data is used for model training, the next 10% of data is for model validation and the rest 10% of data is used as test data to demonstrate model performance. With these three datasets, we evaluate the performance of our social recommendation model in the context of both implicit feedback data and explicit feedback data. It is worth noting that the three datasets were collected from different online applications, proving that our proposed models can be applied to different application scenarios.

2) BASELINES

We compare the proposed social recommendation model with several representative baselines:

- PMF [51]. This is one of the most basic latent factor model. Each user and item is assigned a latent factor vector and the preference is predicted as the inner product of the corresponding user' and item' latent factor vectors. Top-N recommendation is produced by sorting the candidate items in descending order of the predicted preference.
- SocialReg [15]. This is a representative social recommendation model based on matrix factorization. A individual-based social regularization term that constrains the difference between the target user's preference and that of her friends individually is added to control friends' opinions. Pearson Coefficient Correlation was used to measure user similarity.
- SocialMF [52]. This social recommendation model incorporates trust propagation mechanism into matrix factorization techniques. The basic idea is to assume that the latent factors of a user are dependent on that of her neighbors in the social network. In this way, latent factors of users are implicitly connected and accordingly the social/trust influence is propagated through the network.
- SoRec [5]. This social recommendation model assumes that users share the same latent factors in the feedback space and social space. Then the user-item feedback matrix and user-user social relation matrix are co-factorized to learn latent factors of users and items.
- MFGP [40]. On the basis of matrix factorization, this model non-linearly combines latent factors using GP. Specifically, users' latent factors are treated as the input, and a GP regression model is built to learn the distribution of a latent function, which can be used to predict the missing feedback. The covariance function used in this model is squared exponential function. This method is chosen to demonstrate if a non-linear model can help to improve the accuracy of recommendation.
- SoGNN [21]. This method uses graph neural networks to coherently models user-user graph and user-item graph

for social recommendation. The model consists of three components, i.e., user modeling part, where user latent factors are learned, item modeling part, where item latent factor is learned and rating prediction part. We use the official implementation[2] on Github to conduct experiments.

- DiffNet [18]. This model uses a layer-wise recursive social influence diffusion neural network to capture how users' latent factors change as the social influence diffuses. Specifically, for each user, her initial embedding combines a user-specific latent factor vector and related features. The user's embedding is updated when social influence diffusion process continues layer by layer. We use the official implementation[3] on Github to conduct experiments.
- DiffNet++ [23]. This is an improved version of DiffNet. Besides modeling the influence diffusion process in the user-user network, this work also models the latent collaborative interests of users in the user-item interest network. That is, the social influence diffusion and interest diffusion are modeled in a unified framework. Each user's embedding is iteratively aggregated by considering this user's previous embedding, social influence from the connected user neighbors and interest influence from the connected item neighbors. The official implementation of DiffNet++ shares the same Github repository with DiffNet.

PMF is a traditional linear model and MFGP is a GP based non-linear model for recommendation without social network information. Other baselines are social recommendation models. Among these social-aware models, SocialReg, ScoialMF and SoRec are traditional matrix factorization based linear models, while SoGNN and DiffNet are two state-of-the-art deep learning based non-linear models.

3) METRICS

Rating prediction and top-N item recommendation are two major tasks of recommender systems. We argue that item recommendation directly reflects the effectiveness of a recommendation model so in our experiments, we focus on the performance of top-N recommendation, i.e., the popular metrics for rating prediction like Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) will not be adopted. Two metrics are used to measure the performance of top-N recommendation: (1) Precision@N, which is the ratio of the successfully predicted test items to the top-N recommendation. This metric can be used to reflect the hit ratio of the top-N recommendation. (2) Normalized Discounted Cumulative Gain (NDCG), which is a popular ranking metric to measure recommendation/ranking quality in terms of the positions of the successfully predicted items in the recommendation list. Note that for all methods, we ran 10 times and report the averaged results. We also calculated standard deviations to ensure that all comparison results are statistically significant.

All experiments were executed on a server with Intel(R) Xeon(R) CPU E5-2630 2.20GHz, 128GB memory and 4 Nvidia 1080Ti GPUs equipped.

[2]https://github.com/wenqifan03/GraphRec-WWW19
[3]https://github.com/PeiJieSun/diffnet

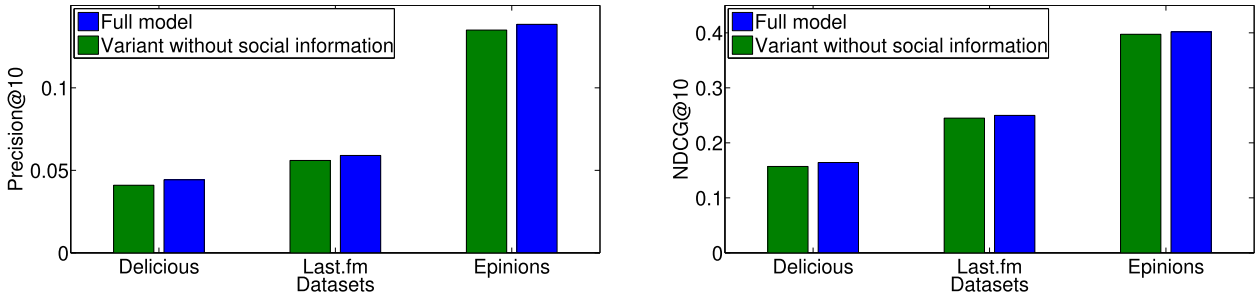
B. EXPERIMENTAL RESULTS

1) DESIGN VALIDATION (RQ1)

We validate the design of the proposed social recommendation model from three aspects: the influence of social network information, the influence of the mean function (biases), and the influence of the minimum number of each user's feedback. We initialized the covariance function hyperparameters λ, h and σ as 1, 0.1 and 0.05. Since SGD is sensitive to the initialized values, for different datasets, we slightly adjusted the three parameters to maximize the performance. According to validation set results, latent factor vector dimensionality was set to 3, 10 and 10 for Delicious data, Last.fm data and Epinions data respectively. Learning rate was initialized as 0.001 with decay rate of 0.95 for each iteration.

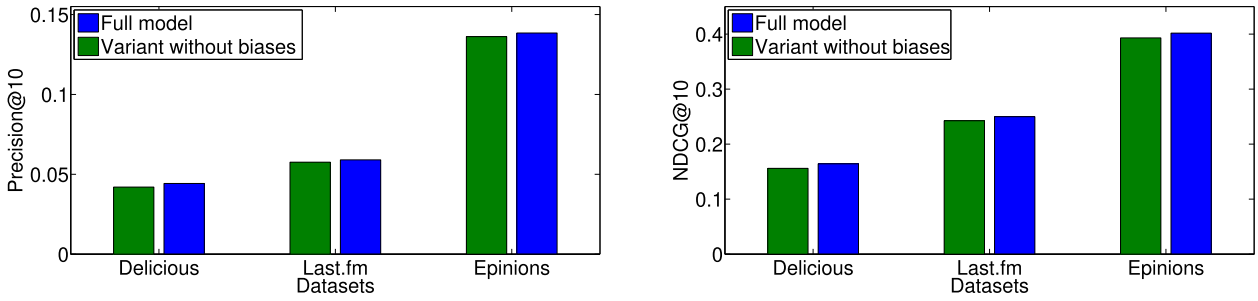
We first demonstrate if social network information is properly handled to improve the quality of recommendations. Fig. 3 shows precision@10 and NDCG@10 of the proposed social recommendation model and a variant where social network information is not considered (i.e., in Eq. 13, $k_i(R_{i,v}, R_{i,v'}) = k(\mathbf{v}_v, \mathbf{v}_{v'})$). For all datasets, the full model outperforms the variant, demonstrating (1) social network information indeed helps to improve recommendation quality, and (2) social network information is well incorporated by our Gaussian process based model. To summarize, social network information improves the performance by 7.97%, 5.44% and 2.56% in terms of precision; 4.58%, 2.00% and 1.51% in terms of NDCG for Delicious data, Last.fm data and Epinions data respectively. The improvements was observed in the datasets that have limited social network information, we believe that in other scenarios with richer social interactions, our model is capable of providing more promising results. To verify this hypothesis in absence of large-scale datasets with more dense social connections, we modify the three datasets by manually creating more social connections. Specifically, for each dataset, for a pair of connected users u_1 and u_2, we extracted u_2's all friends, among which we randomly selected f users and built social connections between u_1 and each of them. Social density factor f is a parameter that can be configured for different experiments. In this way, the social relations become denser. We reran all experiments using the three artificial datasets with different f values, and summarized the corresponding improvements in Tab. 1. Due to denser social information, more evident improvements can be observed compared to the variant without social information.

We then study the influence of mean function, i.e., users' and items' biases. Fig. 4 compares the performance of the proposed social recommendation and a variant where no



(a) Precision. (b) NDCG.

FIGURE 3. **Influence of social network information.**



(a) Precision. (b) NDCG.

FIGURE 4. **Influence of users' and items' biases.**

TABLE 1. **Influence of social density (top-10 recommendation).**

density factor f	Delicious data		Last.fm data		Epinions data	
	Precision	NDCG	Precision	NDCG	Precision	NDCG
1	+8.12%	+4.93%	+6.38%	+2.88%	+3.27%	+2.38%
2	+9.27%	+5.66%	+7.68%	+3.69%	+4.11%	+3.06%
3	+10.04%	+5.95	9.06%	+3.98	+4.75	+4.01%

bias is considered (i.e., the mean function is 0). From the results we observe that although mean function is not the core of Gaussian process (compared to covariance function), by incorporating users' and items' biases that are independent of the interactions between users and items, it still evidently improves the performance. This indicates the importance of biases when inferring users' preference on items. Essentially, biases can be seen as a preference adjustment factor that is learned from past feedback for finer preference inference. Numerically, the mean function improves the performance by 5.40%, 2.51% and 1.66% in terms of precision; 5.30%, 3.03% and 2.21% in terms of NDCG for Delicious data, Last.fm data and Epinions data respectively.

Next we demonstrate the influence of the minimum number of each user's feedback. Theoretically, more training data brings higher accuracy but in most recommender systems, users' feedback information is typically sparse. Tab. 2 summarizes how the precision and the NDCG vary with the varying minimum number of feedback for training (ranging from 5 to 25 with 5 as the increment). It is clear that for all datasets, both precision and NDCG gradually increase with the increasing minimum number of feedback for training (i.e., the users whose training feedback is less than the minimum are discarded from the experiments). On the other hand, higher precision and NDCG (i.e., larger minimum training feedback) means more users with insufficient feedback information are ignored, which damages the coverage of the system. For instance, for Delicious data and Last.fm data, when the threshold is set to 20, only about 85.75% and 31.45% users are able to benefit from the recommendation model. In the following experiments, we set the threshold to 10 to balance the tradeoff between the performance and the coverage of the system for all recommendation models.

Finally, we study the scalability of the proposed models by measuring the training time of the models with (i) different fraction of data used for training (from 40% to 80% with 10% as the increment) and (ii) different latent factor vector dimensionality (from 5 to 20 with 5 as the increment). Using Last.fm dataset,[4] we first ran one experiment with most

[4]Experiments conducted on the other two datasets demonstrated similar scalability trends.

TABLE 2. Influence of the volume of training data (top-10 recommendation).

training vol.	Delicious data		Last.fm data		Epinions data	
	Precision	NDCG	Precision	NDCG	Precision	NDCG
5	0.0401	0.1455	0.0541	0.2047	0.1129	0.3888
10	0.0442	0.1643	0.0590	0.2485	0.1384	0.4017
15	0.0451	0.1695	0.0615	0.2512	0.1425	0.4150
20	0.0462	0.1755	0.0653	0.2655	0.1508	0.4258
25	0.0470	0.1802	0.0660	0.2712	0.1586	0.4375

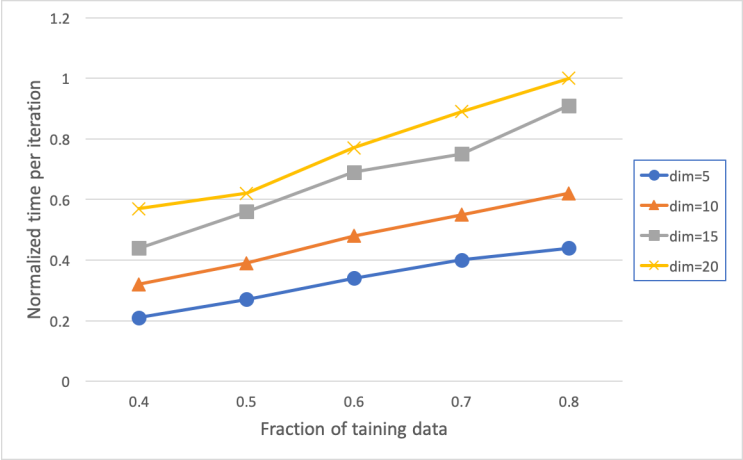


FIGURE 5. Normalized training time comparison.

training data (i.e., 80% of data used for training) and biggest latent factor vector dimensionality, i.e., 20, and recorded that on average, each iteration takes around 75 seconds. We then ran all other experiments with different training data amount and latent factor vector dimensionality settings, and normalized the corresponding training time per iteration by the longest time of 75 seconds, as shown in Fig. 5. We can observe the linear correlation between training time and training data amount for different latent vector dimensionality, so overall, our model can scale linearly with the amount of observation data. Note that all experiments were conducted on the same server with Intel(R) Xeon(R) CPU E5-2630 2.20GHz and 128 GB memory.

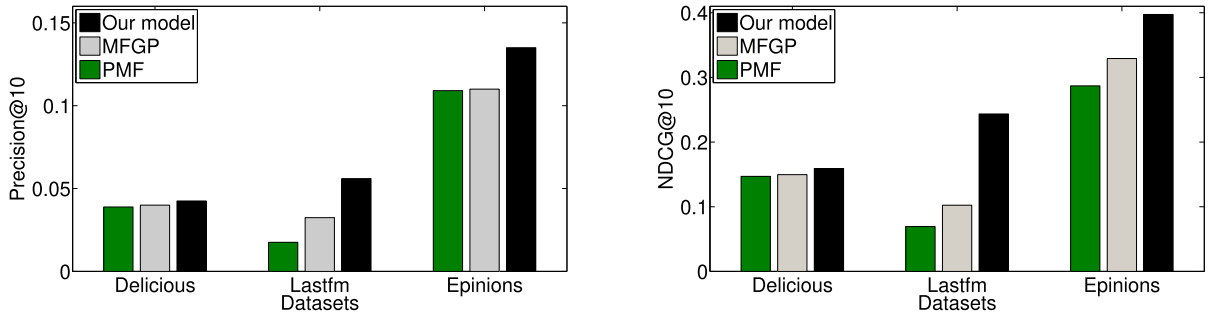
2) COMPARISON STUDIES (RQ2)

We compare the performance of our social recommendation model with that of the state-of-the-art baselines (see Section V-A2). By cross-validation, for PMF, we set latent factor vector dimensionality, learning rate and regularization parameters to (10, 0.02, 1), (10, 0.008, 0.001) and (5, 0.001, 0.01) for Delicious data, Last.fm data and Epinions data respectively. For SocialReg, we set latent factor vector dimensionality, learning rate, regularization parameter and social regularization parameter to (10, 0.05, 1, 0.5), (10, 0.08, 0.0001, 0.00001) and (5, 0.001, 0.1, 2) for Delicious data, Last.fm data and Epinions data respectively. For SocialMF, we set latent factor vector dimensionality, learning rate, regularization parameter and social regularization parameter to (10, 0.01, 0.1, 1), (10, 0.005, 0.1, 1) and (5, 0.05, 0.001, 5) for for Delicious data, Last.fm data and Epinions data respectively. For SoRec, we set latent factor vector dimensionality, learning rate, regularization parameter and social factorization parameter to (10, 0.01, 0.1, 1), (10, 0.005, 0.1, 1) and (5, 0.001, 0.001, 1) for Delicious data, Last.fm data and Epinions data respectively. For MFGP, following the original paper [40], all covariance function hyperparameters were set to 1 except the noise variance σ that was set to 5. The latent factor vector dimensionality and learning rate were set to 10 and 1×10^{-4}. For SoGNN and DiffNet, their hyper-parameters were set according to the corresponding papers, and were then adjusted by cross-validation.

Tab. 3 summarizes the comparison results using the three datasets when top-10 recommendations are provided. Note that we ran each experiment 10 times and provided the mean as final result along with ±standard deviation to indicate statistical significance. For Delicious data, without social information modeling, the linear model PMF performs worst. By taking into account social network information, SocialReg improves PMF a bit, and SocialMF improves SocialReg further, demonstrating the importance of social relations. MFGP outperform both SocialReg and SocialMF, which are linear models, proving that non-linear model can better capture the complex interactions between users and items. For Last.fm data, as expected, MFGP again outperforms PMF, demonstrating that non-linear method is more advanced than its linear counterpart. The behavior of SocialReg is a bit "unexpected": it is better than PMF in terms of precision, but is outperformed by PMF in terms of NDCG

TABLE 3. Performance comparison (top-10 recommendation).

	Delicious data		Last.fm data		Epinions data	
	Precision	NDCG	Precision	NDCG	Precision	NDCG
Our model	$0.0442_{\pm 0.0003}$	$0.1643_{\pm 0.0006}$	$0.0590_{\pm 0.0034}$	$0.2485_{\pm 0.0039}$	$0.1384_{\pm 0.0019}$	$0.4017_{\pm 0.0021}$
PMF	$0.0395_{\pm 0.0003}$	$0.1472_{\pm 0.0006}$	$0.0179_{\pm 0.0031}$	$0.0694_{\pm 0.0038}$	$0.1096_{\pm 0.0024}$	$0.2869_{\pm 0.0032}$
SocialReg	$0.0399_{\pm 0.0006}$	$0.1494_{\pm 0.0010}$	$0.0186_{\pm 0.0062}$	$0.0658_{\pm 0.0079}$	$0.0925_{\pm 0.0083}$	$0.2814_{\pm 0.0102}$
SocialMF	$0.0405_{\pm 0.0007}$	$0.1500_{\pm 0.0013}$	$0.0194_{\pm 0.0051}$	$0.0701_{\pm 0.0063}$	$0.1088_{\pm 0.0042}$	$0.3011_{\pm 0.0049}$
SoRec	$0.0402_{\pm 0.0008}$	$0.1498_{\pm 0.0009}$	$0.0206_{\pm 0.0055}$	$0.0699_{\pm 0.0061}$	$0.1029_{\pm 0.0052}$	$0.2822_{\pm 0.0062}$
MFGP	$0.0408_{\pm 0.0007}$	$0.1502_{\pm 0.0011}$	$0.0325_{\pm 0.0045}$	$0.1024_{\pm 0.0052}$	$0.1103_{\pm 0.0043}$	$0.3292_{\pm 0.0053}$
SoGNN	$0.0420_{\pm 0.0009}$	$0.1542_{\pm 0.0013}$	$0.0428_{\pm 0.0046}$	$0.1836_{\pm 0.0051}$	$0.1229_{\pm 0.0039}$	$0.3612_{\pm 0.0048}$
DiffNet	$0.0413_{\pm 0.0010}$	$0.1525_{\pm 0.0014}$	$0.0375_{\pm 0.0033}$	$0.1629_{\pm 0.0045}$	$0.1202_{\pm 0.0025}$	$0.3465_{\pm 0.0034}$
DiffNet++	$0.0424_{\pm 0.0006}$	$0.1534_{\pm 0.0011}$	$0.0391_{\pm 0.0047}$	$0.1788_{\pm 0.0056}$	$0.1234_{\pm 0.0036}$	$0.3722_{\pm 0.0049}$



(a) Precision. (b) NDCG.

FIGURE 6. Performance comparison when social network information is not available.

(i.e., the successfully predicted items are at the relatively low positions in the recommendation lists). This can be interpreted by the complexity of social information, that is, although SocialReg considers social information, the approach of linear combination may not properly learn the complex social relations. This interpretation is further verified by the results obtained using Epinions data: for both precision and NDCG, SocialReg performs worst. SocialMF and SoRec perform similarly and both consistently outperform SocialReg but the improvements are minor.

SoGNN, DiffNet and DiffNet++ evidently outperform SocialMF, SocialReg and SoRec, indicating the advantage of neural network, which models user-item-friend relations in a non-linear way. SoGNN and DiffNet++ perform similarly, and both of them perform slightly better than DiffNet do, demonstrating that graph neural network is an effective way to model social graph and user-item graph for social recommendation.

In all cases, our social recommendation model outperforms other methods due to the key design that, instead of of linearly combining friends' opinions, the relevant social network information is non-linearly organized into a covariance matrix, which systematically embeds the effects of user-item interactions and the associated social influence through the corresponding latent factor representations. Moreover, biases also contribute to the improvement of recommendation quality. To be specific, our model improves the strongest baseline SoGNN by 18.57% in terms of precision and 17.70% in terms of NDCG (averaged across three datasets).

Finally, we test the effectiveness of our model in the absence of social network information, which is common in many online applications like MovieLens. Fig. 6 shows the comparison results in terms of precision and NDCG. Note that without social information, SocialReg is actually equivalent to PMF so we only present the result of PMF. Similar to previous experimental results, MFGP outperforms PMF, and our model performs best in all three datasets. This again demonstrates the advantage of non-linear modeling of user-item interactions and the importance of biases. In summary, by averaging results from the three datasets, our model improves PMF and MFGP by 84.07% and 33.86% in terms of precision, 99.18% and 54.93% in terms of NDCG. This set of experiments prove that our model can be applied to more generic scenario where social information is not necessarily available.

VI. CONCLUSION

In order to handle the complex social relations in recommender systems, we presented a social recommendation model that non-linearly combines social information using Gaussian process. Our model is built on the basis of a latent factor model where users and items are represented by latent factor vectors. For each user, her past feedback information is organized into a covariance matrix where the covariance of each feedback pair is calculated by the proposed social-aware covariance function to capture the complex interplay among the user, her friends and the corresponding items. We also take into account biases of users and items which are

embedded into the mean function of Gaussian process. We developed a SGD based optimization procedure to fit the model by learning the covariance function hyperparameters, latent factors and biases. Three real-world datasets based experiments demonstrate that our non-linear social recommendation model outperforms the state-of-the-art methods.

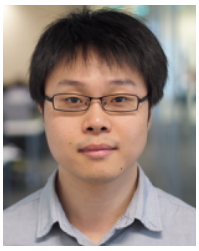
As for the future work, we intend to deeply explore other covariance functions that may better model latent factors of users and items for preference inference. To handle the data sparsity issue that current approach faces, we are interested in exploring collaborative filtering methods that leverage more side information.

REFERENCES

[1] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Comput.*, vol. 42, no. 8, pp. 30–37, Aug. 2009.
[2] J. Tang, X. Hu, and H. Liu, "Social recommendation: A review," *Expert Syst. Appl.*, vol. 3, no. 4, pp. 1113–1133, 2013.
[3] X. Yang, Y. Guo, Y. Liu, and H. Steck, "A survey of collaborative filtering based social recommender systems," *Comput. Commun.*, vol. 41, pp. 1–10, Mar. 2014.
[4] Y. Xiao, Q. Pei, T. Xiao, L. Yao, and H. Liu, "MutualRec: Joint friend and item recommendations with mutualistic attentional graph neural networks," *J. Netw. Comput. Appl.*, vol. 177, Mar. 2021, Art. no. 102954.
[5] H. Ma, H. Yang, M. R. Lyu, and I. King, "SoRec: Social recommendation using probabilistic matrix factorization," in *Proc. 17th ACM Conf. Inf. Knowl. Manage.*, 2008, pp. 931–940.
[6] W. Fan, Q. Li, and M. Cheng, "Deep modeling of social relations for recommendation," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 8075–8076.
[7] X. Liu, "Towards context-aware social recommendation via trust networks," in *Proc. 14th Int. Conf. Web Inf. Syst. Eng. (WISE)*, 2013, pp. 121–134.
[8] W. Lu, S. Ioannidis, S. Bhagat, and L. V. S. Lakshmanan, "Optimal recommendations under attraction, aversion, and social influence," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2014, pp. 811–820.
[9] G. Alexandridis, G. Siolas, and A. Stafylopatis, "Enhancing social collaborative filtering through the application of non-negative matrix factorization and exponential random graph models," *Data Mining Knowl. Discovery*, vol. 31, no. 4, pp. 1031–1059, Jul. 2017.
[10] L. Wu, L. Chen, R. Hong, Y. Fu, X. Xie, and M. Wang, "A hierarchical attention model for social contextual image recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 10, pp. 1854–1867, Oct. 2020.
[11] A. Krishnan, H. Cheruvu, C. Tao, and H. Sundaram, "A modular adversarial approach to social recommendation," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, Nov. 2019, pp. 1753–1762.
[12] C. Liu, C. Zhou, J. Wu, Y. Hu, and L. Guo, "Social recommendation with an essential preference space," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 346–353.
[13] T. Zhao, J. McAuley, and I. King, "Leveraging social connections to improve personalized ranking for collaborative filtering," in *Proc. 23rd ACM Int. Conf. Conf. Inf. Knowl. Manage.*, Nov. 2014, pp. 261–270.
[14] M. Jamali and M. Ester, "*TrustWalker*: A random walk model for combining trust-based and item-based recommendation," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 397–406.
[15] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King, "Recommender systems with social regularization," in *Proc. 4th ACM Int. Conf. Web Search Data Mining (WSDM)*, 2011, pp. 287–296.
[16] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. 26th Int. Conf. World Wide Web*, 2017, pp. 173–182.
[17] X. Liu and K. Aberer, "SoCo: A social network aided context-aware recommender system," in *Proc. 22nd Int. Conf. World Wide Web*, 2013, pp. 781–802.
[18] L. Wu, P. Sun, Y. Fu, R. Hong, X. Wang, and M. Wang, "A neural influence diffusion model for social recommendation," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2019, pp. 235–244.
[19] W. Song, Z. Xiao, Y. Wang, L. Charlin, M. Zhang, and J. Tang, "Session-based social recommendation via dynamic graph attention networks," in *Proc. 12th ACM Int. Conf. Web Search Data Mining*, Jan. 2019, pp. 555–563.
[20] J. Yu, H. Yin, J. Li, M. Gao, Z. Huang, and L. Cui, "Enhance social recommendation with adversarial graph convolutional networks," *IEEE Trans. Knowl. Data Eng.*, early access, Oct. 26, 2020, doi: 10.1109/TKDE.2020.3033673.
[21] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, "Graph neural networks for social recommendation," in *Proc. World Wide Web Conf.*, 2019, pp. 417–426.
[22] Q. Wu, L. Jiang, X. Gao, X. Yang, and G. Chen, "Feature evolution based multi-task learning for collaborative filtering with social trust," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 1–7.
[23] L. Wu, J. Li, P. Sun, R. Hong, Y. Ge, and M. Wang, "DiffNet++: A neural influence and interest diffusion network for social recommendation," *IEEE Trans. Knowl. Data Eng.*, early access, Dec. 31, 2021, doi: 10.1109/TKDE.2020.3048414.
[24] K. Mao, X. Xiao, J. Zhu, B. Lu, R. Tang, and X. He, "Item tagging for information retrieval: A tripartite graph neural network based approach," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2020, pp. 2327–2336.
[25] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT Press, 2005.
[26] G. Gregorčič and G. Lightbody, "Gaussian process approach for modelling of nonlinear systems," *Eng. Appl. Artif. Intell.*, vol. 22, nos. 4–5, pp. 522–533, Jun. 2009.
[27] H. P. Vanchinathan, I. Nikolic, F. De Bona, and A. Krause, "Explore-exploit in top-N recommender systems via Gaussian processes," in *Proc. 8th ACM Conf. Recommender Syst. (RecSys)*, 2014, pp. 225–232.
[28] J. C. Platt, C. J. Burges, S. Swenson, C. Weare, and A. Zheng, "Learning a Gaussian process prior for automatically generating music playlists," in *Proc. 16th NIPS*, 2002, pp. 1425–1432.
[29] N. Houlsby, J. M. Hernandez-Lobato, F. Huszar, and Z. Ghahramani, "Collaborative Gaussian processes for preference learning," in *Proc. 26th NIPS*, 2012, pp. 2096–2104.
[30] Q. Liu, E. Chen, B. Xiang, C. H. Ding, and L. He, "Gaussian process for recommender systems," in *Knowledge Science, Engineering and Management*, vol. 7091. Berlin, Germany: Springer, 2011, pp. 56–67.
[31] E. V. Bonilla, S. Guo, and S. Sanner, "Gaussian process preference elicitation," in *Proc. 24th NIPS*, 2010, pp. 262–270.
[32] X. Liu, "Modeling users' dynamic preference for personalized recommendation," in *Proc. 24th Int. Joint Conf. Artif. Intell. (IJCAI)*, 2015, pp. 1785–1791.
[33] F. Liu and H. J. Lee, "Use of social network information to enhance collaborative filtering performance," *Expert Syst. Appl.*, vol. 37, no. 7, pp. 4772–4778, Jul. 2010.
[34] P. Victor, N. Verbiest, C. Cornelis, and M. D. Cock, "Enhancing the trust-based recommendation process with explicit distrust," *ACM Trans. Web*, vol. 7, no. 2, pp. 1–19, May 2013.
[35] J. Yu, M. Gao, J. Li, H. Yin, and H. Liu, "Adaptive implicit friends identification over heterogeneous network for social recommendation," in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2018, pp. 357–366.
[36] J. Tang, H. Gao, X. Hu, and H. Liu, "Exploiting homophily effect for trust prediction," in *Proc. 6th ACM Int. Conf. Web Search Data Mining (WSDM)*, 2013, pp. 53–62.
[37] M. Wang, X. Zheng, Y. Yang, and K. Zhang, "Collaborative filtering with social exposure: A modular approach to social recommendation," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 2516–2523.
[38] X. Wang, X. He, L. Nie, and T.-S. Chua, "Item silk road: Recommending items from information domains to social users," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2017, pp. 185–194.
[39] P. Sun, L. Wu, and M. Wang, "Attentive recurrent social recommendation," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jun. 2018, pp. 185–194.
[40] N. D. Lawrence and R. Urtasun, "Non-linear matrix factorization with Gaussian processes," in *Proc. 26th Annu. Int. Conf. Mach. Learn. (ICML)*, 2009, pp. 601–608.
[41] R. P. Adams, G. E. Dahl, and I. Murray, "Incorporating side information in probabilistic matrix factorization with Gaussian processes," in *Proc. 26th UAI*, 2009, pp. 1–9.
[42] T. V. Nguyen, A. Karatzoglou, and L. Baltrunas, "Gaussian process factorization machines for context-aware recommendations," in *Proc. 37th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2014, pp. 63–72.

[43] S. Rendle, "Factorization machines," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2010, pp. 995–1000.
[44] Y. Zheng, L. Z. Zhang, Z. X. Ma, X. Xie, and W. Y. Ma, "Recommending friends and locations based on individual location history," *ACM Trans. Web*, vol. 5, no. 1, pp. 1–44, Feb. 2011.
[45] I. Guy and D. Carmel, "Matrix multiplication via arithmetic progressions," in *Proc. 9th Annu. ACM Symp. Theory Comput.*, 2011, pp. 1–6.
[46] I. King, M. R. Lyu, and H. Ma, "Introduction to social recommendation," in *Proc. 19th Int. Conf. World Wide Web (WWW)*, 2010.
[47] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2008, pp. 426–434.
[48] D. Coppersmith and S. Winograd, "Matrix multiplication via arithmetic progressions," in *Proc. 19th Annu. ACM Conf. Theory Comput. (STOC)*, 1987, pp. 1–6.
[49] I. Cantador, P. Brusilovsky, and T. Kuflik, "Second workshop on information heterogeneity and fusion in recommender systems (HetRec2011)," in *Proc. 5th RecSys*, 2011, pp. 1–2.
[50] P. Massa and P. Avesani, "Trust-aware bootstrapping of recommender systems," in *Proc. Workshop Recommender Syst. (ECAI)*, 2006, pp. 29–33.
[51] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *Proc. 21st NIPS*, 2007, pp. 1–8.
[52] M. Jamali and M. Ester, "A matrix factorization technique with trust propagation for recommendation in social networks," in *Proc. 4th ACM Conf. Recommender Syst. (RecSys)*, 2010, pp. 135–142.

JIYONG ZHANG (Member, IEEE) received the B.S. and M.S. degrees in computer science from Tsinghua University, in 1999 and 2001, respectively, and the Ph.D. degree in computer science from the Swiss Federal Institute of Technology at Lausanne (EPFL), in 2008. He is currently a Distinguished Professor with Hangzhou Dianzi University. His research interests include intelligent information processing, machine learning, data sciences, and recommender systems.



XIN LIU received the Ph.D. degree from Nanyang Technological University, Singapore, in 2012. From 2012 to 2014, he was a Postdoctoral Researcher with the LSIR Laboratory, EPFL, Switzerland. He is an Associate Professor with the School of Automation, Hangzhou Dianzi University. His research interests include recommender systems, trust modeling, natural language processing, and computer vision.

XIAOFEI ZHOU received the Ph.D. degree from Shanghai University, Shanghai, China, in 2018. He is currently a Lecturer with the School of Automation, Hangzhou Dianzi University, Hangzhou, China. His research interests include saliency detection, video segmentation, and video quality assessment.