IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 25, 2017, accepted September 17, 2017, date of publication September 26, 2017,
date of current version October 12, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2756102

Metric Learning Combining With Boosting for User Distance Measure in Multiple Social Networks

YUFEI LIU[1], DECHANG PI[1,2], AND LIN CUI[1]
[1]College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China
[2]Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing 211106, China

Corresponding author: Dechang Pi (dc.pi@nuaa.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant U1433116 and Grant 61702355, in part by the Fundamental Research Funds for the Central Universities under Grant NP2017208, and in part by the Funding of Jiangsu Innovation Program for Graduate Education under Grant KYLX15_0324 and Grant KYLX15_0321.

ABSTRACT How to model user distance from multiple social networks is an important challenge. People often simultaneously appear in multiple social networks that can provide complementary services. Thus, knowledge from different social networks can help overcome the data sparseness problem. However, the knowledge cannot be directly obtained due to that they are from different social networks. To solve this problem, we construct an adaptive model to learn user distance in multiple social networks via combining distance metric learning and boosting technologies. The basic idea of our model is to embed related social networks into a potential feature space, while retaining the topologies of social networks. To get the solution to our model, we formulate it as a convex optimization problem. Moreover, we propose an adaptive user distance measure algorithm whose time complexity is linear with the number of the links. We verify the feasibility and effectiveness of our model on the link prediction problem. Experiments on two real large-scale data sets demonstrate that our method outperforms the compared methods. To the best of our knowledge, the joint learning of metric learning with boosting is first studied in multiple social networks.

INDEX TERMS Multiple social networks, user distance, metric learning, boosting.

I. INTRODUCTION

Social networks are ubiquitous and increasingly popular in everyday life. The applications of social networks can capture a user's information, such as his/her attributes (e.g., age, gender, education level) and behaviors (e.g., posting messages, being a fan of Bill Gates, etc.). However, due to the data sparseness in a single social network, most users only have a few of neighbors and some users (e.g., QQ users) do not provide true information in many cases. Measuring the distance between two users is difficult if they do not share anything directly. It is a crucial observation that a user usually appears in various social networks at the same time. For example, users may exchange information with their friends on Tencent QQ, while pushing feedback in Tencent Weibo and commenting a new film on Douban. Therefore, data from a single sparse network easily leads to over-fitting of the constructed models. However, data resources from different social networks can describe the user distance from different aspects. For a sparse social network, its interconnected networks have related information and can solve the problem of data sparseness [1]–[3]. Due to the inherent differences among multiple networks, a network cannot be simply fused with other networks [4], [5]. First of all, different networks have different attributes, such as the density and the degree distribution. If we directly fuse a dense network with a sparse one, knowledge of the sparse network will be submerged in the dense network. Secondly, the distance between two users may be very different in different social networks. For example, two users are similar in Tencent Weibo as they have similar interests and hobbies, but they may be not friends on Tencent QQ. Besides, most internet users are keen on multiple online social networks. In most cases, a user joins several distinct social networks at the same time and is a member of different social networks, as shown in Fig. 1. He/She has different behaviors in different social networks, and simultaneously shares different interests in each social network. People tend to establish their social relationships based on their interests. Some users share some common

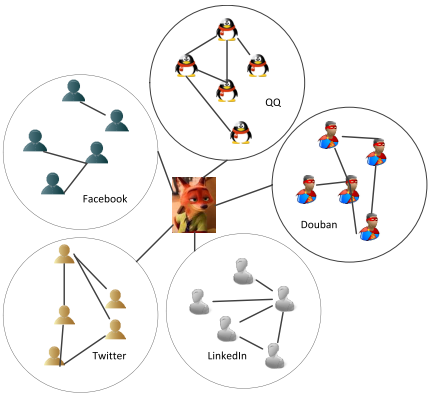


FIGURE 1. A user in five different social networks. In these networks, each subnetwork can reflect users' relationships from one aspect. For example, friendships on Tencent QQ reflect users' familiarity in their daily life, while relationships on Douban reflect their common interests on films. In practice, most interactive nodes are blank and models are bound to encounter the data sparseness problem in a single social network.

interests across social networks. Thus, links (relationships) in different social networks can help to solve the sparseness problem. However, simple fusion of social networks does not accurately reflect the distance between users.

In this paper, our motivation is to simultaneously handle multiple related social networks, which are exploited for mining valuable information. To achieve this goal, we construct an adaptive distance measure model from the multiple social networks by combining the metric learning and the boosting technologies, where the model is further formulated into a convex optimization problem [6]–[8]. It is the basic idea of our model to embed multiple related social networks into a potential feature space according to the link information and the characteristics of the nodes. In the potential feature space, topologies and public structures are kept in order to accurately measure the distance between two users. Besides, the irrelevant data from related networks can be eliminated in boosting in order to avoid network differences and useless information which may bring negative effects [9]–[11].

Concretely, we use the metric learning model to measure the distance between users, where the main purpose of metric learning is to assess the distance or similarity [12]. Metric learning plays an important role in the tasks of classification, clustering and pattern recognition [13]. Therefore, our research adopts metric learning model to measure the distance between two users. We use AdaBoost to eliminate irrelevant attributes to alleviate the negative effects resulted from the differences of social networks, where AdaBoost is a very popular and widely used ensemble learning algorithm [14]–[16]. Therefore, our research employs AdaBoost to further improve the prediction performance. The conducted theoretical analysis demonstrates that our proposed method can converge rapidly and has good generalization ability. Then, we propose an Adaptive User Distance Measure algorithm (short for AUDM) for measuring user distance. Its time complexity is linear with the number of links, which indicates that it can be applied to large-scale social network datasets. We evaluate the proposed AUDM algorithm on the link prediction problem, which is a common task in social network analysis [17]. The main contributions of our work are summarized as follows.

1) We formulate the problem of learning user distance from multiple social networks and design a distance measure model by utilizing the metric learning approach and the boosting framework.
2) AUDM, an adaptive user distance measure algorithm based on the above mentioned model, is proposed. Its time complexity is linear with the number of the links, which means that our algorithm can be applied to large-scale social datasets.
3) We conduct extensive experiments to assess the performance of the proposed method. Experimental results verify that AUDM is feasible and effective, and it is superior to the compared methods on the problem of link prediction.

The rest of this paper is organized as follows. Section II summarizes some related works and briefly reviews the metric learning and the AdaBoost to facilitate the latter description. In Section III, we introduce the formulation of the problem (measuring user distance), and discuss the basic idea of our method, and propose an adaptive user distance measure algorithm. In Section IV, we make analysis of the generalization error and time complexity of our method. Section V illustrates the performances of our method on two real large-scale datasets. The discussion is given in Section VI.

II. RELATED WORK

It is known that metric learning is effective in classification, clustering and pattern recognition. Weinberger *et al.* [12] proposed a graph embedding method, which can preserve the topological structural network seen in dataset and predict users' behaviors. However, this method depends on graph size and does not take into consideration the community structure as well as the relationship features among users. Bronstein *et al.* [18] had used the boosting technology in metric learning, which can be regarded as an effective solution to specific positive semi-definite programs related to metric learning [19]. However, all the above methods are only applicable to the case of a single social network, and they are bound to encounter the data sparseness problem. Therefore, the accurate model of the above methods may not be set up.

A. METRIC LEARNING

Moutafis *et al.* [20] defined the problem of metric learning: given an initial metric distance $d(x_i, x_j)$, such as Euclidean distance to measure the distance between the input sample x_i and x_j, metric learning constructs a new measure distance, which is better than the initial metric distance function $d(x_i, x_j)$. In order to construct a better measure distance, nowadays most metric learning algorithms focus on learning a mapping. In other words, it uses the original distance measure to calculate the distance between the samples after mapping. According to this definition, metric learning

algorithm can be divided into two categories: linear and nonlinear.

It is very important to define an appropriate distance measure function for classification, clustering, etc. For example, if the appropriate distance measurement function is defined for image classification, the KNN classifier can obtain better classification results [21], [22]. The traditional distance measure method often does not obtain ideal effectiveness, because of the lack of choice of metric space transformation, and ignoring the background environment and constraint information.

B. ADABOOST

In the past two decades, some effective ensemble learning algorithms have been proposed [15], [16], where AdaBoost is one of the most popular algorithms and has been widely used to improve the classification performance of individual classifiers in practical applications. To get a strong classifier, AdaBoost first sequentially generates many diverse weak classifiers whose prediction accuracies are slightly better than random guessing, and then combines them using the weighted majority voting rule to obtain the final ensemble classifier. Concretely, when training the mth ($1 \preceq m \preceq M$) individual classifier, AdaBoost first resamples the original training set D_{tr} according to the current weight distribution and obtains a training subset D_{tr_m} on which classifier $G_m(x)$ is generated by applying a classifier learning algorithm, then classifier $G_m(x)$ is used to classify the original training set D_{tr}, and the training samples misclassified by classifier $G_m(x)$ will have a higher possibility to be selected into the training set of the $(m+1)$th classifier. In this way, AdaBoost focuses on learning the hard-to-learn samples. When using AdaBoost to generate an ensemble classifier, we only need to specify the number of iterations and do not need any prior knowledge. AdaBoost can adaptively adjust all the parameters in the running process [23]–[25].

The main steps of AdaBoost algorithm are detailed as follows. First, the weight distribution of the original training set is initialized. Each training sample is given the same weight $\frac{1}{N}$ (N is the number of samples). Second, if a training sample is correctly classified by the classifier generated in the current iteration, then the weight of this sample is decreased and thus it will have a lower possibility to be selected into the training set of the next classifier. On the contrary, if a sample is misclassified, then its weight is increased and thus it will have a higher possibility to be selected into the training set of the next classifier. Fig. 2 shows the general procedure of AdaBoost algorithm.

III. PROPOSED METHODOLOGY

In this section, first, we present the related formal definitions of measuring user distance. Next, according to the metric learning technology, we design a distance measure for our model in multiple social networks. Then, according to the boosting framework, we design a boosting technology

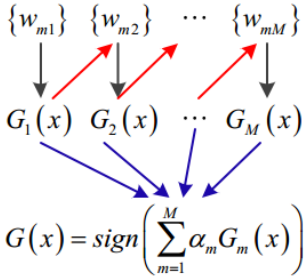


FIGURE 2. The general procedure of AdaBoost. α_m is the weight of the mth classifier and w is the weight of the training sample. Red arrows denote updating the weight of the training dataset, and blue arrows denote the weighted combination of the base classifiers.

TABLE 1. Notations used in this model.

Symbol	Description
$\mathbb{N}$	Social networks set
N_i	The ith network
n	Number of users
$\mathbf{U}$	All user set
U_i	The ith Users set
A_i	Adjacent matrix of the ith network
E_i	User relationship of the ith network
l	Number of social networks
N_t	Target social network
N_r	Related social network

suitable for our model to solve the problem of sparseness in single social networks. Finally, we analyze the convergence of our method and implement our model.

A. RELATED DEFINITIONS

We now formulate the problem of measuring user distance in multiple social networks, and the notations are summarized in Table 1. In this Table, $\mathbb{N} = \{N_i = (U_i, E_i)\}_{i=1}^{l}$ denotes a set of social networks; $U = \cup\{U_i\}_{i=1}^{l} = \{u_j\}_{j=1}^{n}$ is the set of all users, where n is the total number of users. The adjacent matrix of the ith ($1 \leq i \leq l$) social network is defined as A_i. For the sake of simplicity, we assume that there are two social networks in $\mathbb{N}$: N_t and N_r, where N_t is the target social network and N_r is the related network of N_t. In addition to social relationship, a user has other profiles (such as age, gender, education level, etc.) and behavior characteristics (e.g., listening to music, being a fan of Bill Gates, etc.) in social networks. Thus, a feature vector $x_i \in \mathbb{R}^{1\times f}$ denotes a user, where f is the number of features. Each dimension denotes one characteristic or one behavior; $X = \{x_i\}_{i=1}^{n}$ denotes the feature matrixes of U. Thus, user pairs can form graph-based features, such as the number of common links. We define the graph-based features of user pairs as the relational features, which are formulated as $R = \{r_{ij}\}_{u_i,u_j\in U}$, where $r_{ij} \in \mathbb{R}^{1\times z}$ and z is the dimension of the relationship. The task of our user distance measure is to measure the distance between u_j and $u_k (1 \leq k \leq n)$ in N_t by exploiting the knowledge from X, R and N_r.

B. METRIC LEARNING MODEL IN AUDM

We use metric learning to measure the users' distance. Formula (1) shows our measure model.

$$D(u_j, u_k|N_t) = \left(x_j - x_k\right) M \left(x_j - x_k\right)^T + \omega r_{jk}^T \tag{1}$$

where the distance $D(\cdot|N_t)$ is the user distance in the target social network. $M(M \succeq 0, M \in \mathbb{R}^{f\times f})$ is a positive semi-definite matrix of node features. $M = LL^T$ and $L \in \mathbb{R}^{f\times f}$, where M is equivalent to a linear embedding in an input feature space XL. The second term in formula (1) is an auxiliary term. $\omega \in \mathbb{R}^{1\times z}$ is the vector of the relational feature mapping. r_{jk} is the relational feature in R. If the distance between x_j and x_k is closer, then each dimension in ωr_{jk}^T is smaller. It is a distance-preserving projection. M and ω can be constructed from users link information and community structure in the following paragraphs.

Here, metric learning is not the same as the metric learning in clustering or classification. Social network contains relational data but does not have instance or the class labels [26]. The classical independence assumption in the traditional machine learning is violated. Thus, when we construct M and ω, two constraints must be satisfied. One is keeping network topology, and the other is keeping the user community in the feature space. The motivation of using the two constraints is as follows: (1) for a given user, the distance from his/her non-neighbor must be further than the distance from his/her neighbor; (2) in the same community, users are closer and share more common interests. Thus, if the community structure is reserved after embedding, we can more accurately measure the distance between users. Formally, we define the constraints as follow:

$$D(u_i, u_j) > (1 - A_{ij}) \max_k (A_{ik} D(u_i, u_k)), \quad \forall i, j, k \tag{2}$$

We reserve the community structure in the embedded space by maximizing standard module [27]. Let d_i denote the degree of u_i and m denote the number of links. Formally, we have

$$\max_L \frac{1}{4m} \sum_{ij} \left(A_{ij} - \frac{d_i d_j}{2m}\right) (x_i L) \left(x_j L\right)^T = \frac{1}{4m} Tr\left((XL)^T B\,(XL)\right) \tag{3}$$

where B is a modularity matrix $B_{ij} = A_{ij} - \frac{d_i d_j}{2m}$. We integrate the formula (2) and (3) together and reinforce the restrictions on L and ω to control their complexity. Therefore, we can get the following objective function.

$$\begin{aligned} &\max_{M,\omega} \frac{1}{4m} Tr\left((XL)^T B\,(XL)\right) - \beta\left(\|L\|_2^2 + \|\omega\|_2^2\right) \\ &s.t. \quad D\left(u_i, u_j\right) \succ \left(1 - A_{ij}\right)\left(A_{ik} D\left(u_i, u_k\right)\right), \quad \forall i, j, k \end{aligned} \tag{4}$$

The objective function in the formula (4) is non-convex, and thus it is difficult to optimize. Aiming to solve this problem, we deduce another equivalent convex objective function by using the Lagrange multiplier method. Then, by solving Lagrange function, we have

$$\begin{aligned} \min_{M,\omega} &\sum_{ij} A_{ij}\left(D\left(u_i, u_j\right) - \omega r_{ij}\right) \\ &+ \lambda \sum_{i,j,k} \left(Tr\left(C^{(i,j,k)} X M X^T\right) + \omega\left(r_{ij} - r_{ik}\right)\right) \\ &+ \frac{\beta}{2}\left(\|M\|_2^2 + 2\|\omega\|_2^2\right) \end{aligned} \tag{5}$$

where $C^{(i,j,k)}$ represents the sparse matrix with a constraint $\left\{(i, j, k), A_{ij} = 1, A_{ik} = 0\right\}$. $C_{ij}^{(i,j,k)} = -1$, $C_{ji}^{(i,j,k)} = -1$, $C_{kk}^{(i,j,k)} = -1$, $C_{jj}^{(i,j,k)} = +1$, $C_{ik}^{(i,j,k)} = +1$, $C_{ki}^{(i,j,k)} = +1$. Formula (5) is convex with respect to M and ω. It is a hard margin, but we use soft margin in literature [28]. It has previously been suggested that a combined hypothesis is a large margin, but not necessarily the maximum hard margin in AdaBoost algorithm. And after each iteration the value of the function (5) decreases. Thus, it can converge to the global optimal solution. For the reason of efficiency, we derive the projection stochastic gradient descent algorithm (SGDA) [28] to optimize the formula (5) instead of using semi-definite programming [29]. Its sub-gradient is formulas (6) and (7) with respect to M and ω, respectively.

$$\nabla_M = \beta M + \lambda X^T C X + \sum\nolimits_{i,j,A_{ij}=1} Y_{ij} \tag{6}$$

$$\nabla_\omega = 2\beta\omega + \lambda \sum\nolimits_{i,j,k} \left(r_{ij} - r_{ik}\right) - \sum\nolimits_{i,j,A_{ij=1}} r_{ij} \tag{7}$$

where $Y_{ij} = \left(x_i - x_j\right)^T \left(x_i - x_j\right)$, $C = \sum_{i,j,k} C^{(i,j,k)}$. We first calculate the matrix C, and then update M with Eq. (6) in each iteration.

C. WEIGHT-BOOSTING TECHNOLOGY IN AUDM

Let V_r and V_t denote the constraint matrixes of the related network and the target network, respectively. Adaptive thought is that these restrictions are removed from the related network, because the restrictions are not the same as the knowledge from the target network, and they cannot make contributions to the adaptive model. We first learn ω, and then learn $\widetilde{M}$ based on the optimal value of ω. Therefore, the objective function that we need to optimize can be represented as:

$$\begin{aligned} \min_M &\sum_{i,j} A_{ij} D_M\left(u_i, u_j\right) + \lambda \sum_{i,j,k} Tr\left(C^{(i,j,k)} X \widetilde{M} X^T\right) \\ &+ \beta/2\left\|\widetilde{M}\right\|_2^2 + o, \quad s.t.\ \widetilde{M} \succeq 0 \end{aligned} \tag{8}$$

where o is a constant decided by the optimal ω and $D_{\widetilde{M}}\left(u_i, u_j\right) = \left(x_i - x_j\right) \widetilde{M} \left(x_i - x_j\right)^T$. First we will formulate the formula (8) as a problem of boosting, then put forward a new weight updating rules to absorb related knowledge [30]. $\widetilde{M}$ can be decomposed into $\widetilde{M} = L\Sigma L^T = \sum_k \alpha_k L_k L_k^T$, where Σ is a diagonal matrix and its kth element (α_k) is its kth eigenvalues. As a result, the distance with respect to the measure matrix $\widetilde{M}$ can be considered to be the distance with respect to the measure matrix Σ in the embedded space.

We define a weak learner as shown in Eq. (9), which is boosted to get a single rule by using the same process as the one in the inner loop of AUDM essentially. In each loop, the weak learner generates an individual weak hypothesis, and then the instances are reweighted according to the classification results of the generated hypothesis. Finally, all the generated weak hypotheses are combined into a strong hypothesis with weighted combination.

$$h_k\left(x_i, x_j\right) = \left(x_i - x_j\right) L_k L_k^T \left(x_i - x_j\right)^T \tag{9}$$

Thus, $h\left(x_i, x_j\right) - h\left(x_i, x_k\right)$ is the loss of constraint (i, j, k). We make it into $(-1, 1)$ for ease of use. From the perspective of boosting, our goal is to minimize the following exponential loss function.

$$E = \sum_{i,j,k} \exp\left\{\sum_q \alpha_q \left(h_q\left(x_i, x_j\right) - h_q\left(x_i, x_k\right)\right)\right\} \tag{10}$$

In the qth iteration, we optimize the objective function according to α_q and h_q by fixing the previous q-1 weak learners and their coefficients. Next, the objective function becomes $E = \sum_{i,j,k} w_{i,j,k}^q \exp\left\{\alpha_q \left(h_q\left(x_i, x_j\right) - h_q\left(x_i, x_k\right)\right)\right\}$, where $w_{i,j,k}^q = \exp\left\{\sum_{c=1}^{q-1} \alpha_c \left(h_q\left(x_i, x_j\right) - h_q\left(x_i, x_k\right)\right)\right\}$. Because they are independent of α_q and h_q, they can be regarded as constants. Finally, according to the derivation in [31], E can be minimized with respect to $h_q = \arg\min_{h_c} \sum_{i,j,k} w_{i,j,k}^q \left[h_c\left(x_i, x_j\right) - h_c\left(x_i, x_k\right)\right]$, where $[\pi] = \begin{cases} 1, & h_c\left(x_i, x_j\right) \geq h_c\left(x_i, x_k\right) \\ 0, & otherwise \end{cases}$. We use vector L_q to replace L and set $\Sigma = I$ (identity matrix) according to the projected gradient descent algorithm. The gradient of L_q is as follows:

$$\nabla_{L_q} = 2\beta L_q + \lambda \sum_p X^T C_p X L_q + \sum_p \sum_{i,j} A_{ij}^p \left(x_i - x_j\right)\left(x_i - x_j\right)^T L_q \tag{11}$$

After learning h_q, we need to find the optimal value of α_q, which is implemented by calculating the deviation of E and restructuring the form of deviation. The obtained value of α_q is shown as follows:

$$\alpha_q = \log\left(\frac{1+\varepsilon_q}{1-\varepsilon_q}\right)/2, \quad \varepsilon_q = \sum_{i,j,k} w_{i,j,k}^q \left(h_c\left(x_i, x_k\right) - h_c\left(x_i, x_j\right)\right) / \sum_{i,j,k} w_{i,j,k}^q \tag{12}$$

Finally, after the values of α_q and h_q are obtained, each constraint $(i, j, k) \in V_t$ is updated according to Eq. (13).

$$w_{i,j,k}^{q+1} = w_{i,j,k}^q \exp\left\{\alpha_q \left(h_q\left(x_i, x_j\right) - h_q\left(x_i, x_k\right)\right)\right\} \tag{13}$$

According to the above description, we summarize the AUDM algorithm in a pseudo-code format in Algorithm 1. Step 3 - step 12 are the procedure of projected stochastic gradient descent, and are to optimize the formula (5) whose subgradients are the formulas (6) and (7) with respect to M and ω, respectively. Thus, L_i can be learned in step 13. Step 13 - step 20 are the procedure of boosting, where the L_i and α_i can be learned. Step 15 updates the weight of constraints in V_t. Step 16 - step 18 update the weight of constraints in V_r. Finally, we optimize the formula (8). In algorithm 1, q is the iteration number of boosting.

Algorithm 1 Adaptive User Distance Measure (AUDM)

Require: I, N, f, constraint matrixes: $V = \left\{C^{(i,j,k)}\right\}$, V_t, V_r
Ensure: Metric matrix $\widetilde{M}$, ω

1: Initialize the ω for each constraint as $\frac{1}{|V_r|+|V_t|}$, $M \leftarrow I$
2: **for** $i = 1$ to f **do**
3: **for** $j = 1$ to I **do**
4: Learn ω by using stochastic gradient descent algorithm based on formula (7) and let $C = \{0\}^{n\times n}$
5: **for** o=1 to N **do**
6: Resample matrix $C^{(i,j,k)}$ from V
7: *if* $D(u_i, u_j) - D(u_i, u_k) + 1 \succ 0$ *then*
8: $C = C + C^{(i,j,k)}$
9: **end for**
10: Update M using equation (6) and project M into a positive semi-definite cone
11: *if* convergence *break*
12: **end for**
13: Learn L_i using M with weights w^{i-1}
14: Compute α_i^t using formula (12) in V_t
15: Use formula (13) to update weights for constraints in V_t
16: **for** $(i, j, k) \in V_r$ **do**
17: $w_{i,j,k}^q = w_{i,j,k}^{q-1}\left(h_q\left(x_i, x_j\right) - h_q\left(x_i, x_k\right)\right)$
18: **end for**
19: Normalize w^i to a distribution
20: $\widetilde{M} = \sum_i \alpha_i^t L_i^T L_i$
21: **end for**
22: **return** $\widetilde{M}$, ω

IV. ANALYSIS OF GENERALIZATION ERROR AND COMPLEXITY

In the fth iteration of AUDM, the weight expectation of each constraint in V_r is the Eq. (14).

$$E_{(i,j,k)\in V_r}\left\{w_{r,i,j,k}^f\right\} = \frac{1}{|V_r|+|V_t|}\prod_{t=1}^{f}(1-\varepsilon_t) \geq \frac{(0.5+\gamma)^f}{|V_r|+|V_t|} \tag{14}$$

In the target network V_t, training error of h is limited $\varepsilon_t(h) \leq e^{-2\gamma^2 f}\left(\frac{|V_r|+|V_t|}{V_t} - \frac{(0.5+\gamma)^f |V_r|}{V_t}\right)$, where $|V_t|$ and $|V_r|$ represent the number of constraints in the target network and the related network, respectively. Based on the result of Kar-Ann *et al.* [32] and a higher probability, the generalization error of AUDM is no more than $\varepsilon_t(h) + O\left(\sqrt{f d_v / |V_t|}\right)$

in the target network, where d_v represents its VC-dimension.

The proposed AUDM algorithm needs to execute I iterations before convergence. In each iteration, it samples constraints N times; and for each constraint, the time complexity of calculating the distance between two users is $O(I * f)$. Therefore, the total time complexity is $O(I * N * f * f)$. Actually, N is the number of the links. This means that the proposed algorithm can be applied to large-scale social datasets.

V. EXPERIMENTS

In this section, we compare our method with the Structure Preserving Metric Learning (SPML) [33], the Supervised Random Walks (SRW) [34], and the Relational Topic Model (RTM) [35] on the link prediction problem on two real world datasets. SPML is a metric learning method without considering the community structure for a single social network. SRM is a learning algorithm based on random walks for link prediction and link recommendation by utilizing node and edge attribute data. RTM method is appropriate for data that consists of counts, and can learn a link probability function in addition to latent topic mixtures describing each node.

A. DATASETS

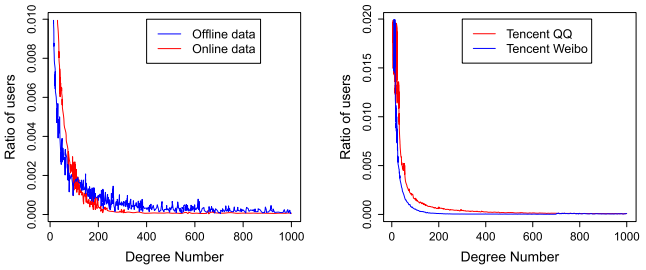
The first dataset used in our experiments is crawled from Douban,[1] which includes two social networks. One shows that who takes notice of another on the Chinese website Douban, and the other shows that who knows well another in our daily life, i.e., the physical offline network. They contain a large amount of user behavior information. We use the API provided by Douban website to crawl the online data in this manner: randomly select 20 advanced users whose registration time are more than five years and still active. Then, we collect more users' information by repeatedly conducting breadth-first search on online data until the number of users reaches 50,000. Each user is regarded as a node in the network and has 150 features. Meanwhile, online relationship of users is recorded. As Douban website does not provide the API that can be used to collect offline data, we utilize the co-occurrence relationship among users to obtain offline data, which represents the contact of users in the real world. For example, we consider users to know each other in case that they participate in the same type of part more than three times based on the prior knowledge. Finally, we collect approximately 5 million edges for online data and 4 million edges for offline data.

The complete second dataset used in our experiments is crawled from Tencent, which also includes two social networks. One is the Tencent QQ[2] representing instant messaging network in China, and the other is the Tencent Weibo[3] representing a microblogging network. First, we randomly select twenty users (the number of their neighbors is more than one hundred). Then, we collect more users' information by repeatedly conducting breadth-first search on both of the networks until the number of users reaches the threshold of one million. Each user is regarded as a node in a social network. Each user has 98 features in the Tencent QQ network, while the number of the features of each user in the Tencent Weibo network is less than that of the Tencent QQ network. The two networks have some common features. After the fusion of the two networks, the number of features of each user becomes 110. Finally, the information we have collected is about one million users, and has approximately 8 million edges for the Tencent QQ network dataset and 32 million edges for the Tencent Weibo dataset. Table 2 summarizes the features for the different social networks.

[1](http://www.douban.com)
[2](http://www.tencent.com/zh-cn/index.shtml)
[3](http://t.qq.com/)

TABLE 2. Summary of datasets features.

Category	Douban		Tencent	
Single network	Online	Offline	QQ	Weibo
Users number	50,000		1 million	
Features number	150		110	
Edges number	5 million	4 million	8 million	32 million



(a) (b)

FIGURE 3. Degree distributions of two datasets. (a) Degree distribution of the Douban dataset. (b) Degree distribution of the Tencent dataset.

The degree distributions of the datasets are plotted in Fig. 3. From the Fig. 3, we observe that each sub-network dataset obeys the Power Log Distribution [36]. The relationships among users in each sub-network dataset are very sparse, which means that both of the networks are sparse and the most relationships among users are limited. Therefore, we should use the knowledge from other social networks to analyze the behaviors of users.

B. EXPERIMENTAL SETTING

In order to train our model, we construct a set of samples including two classes, i.e., the positive class and the negative class. A training sample is marked as a positive sample if there is a corresponding edge between the node pair; otherwise, it is marked as a negative one. Because there are $O(n^2)$ possible samples, both network graphs sparse, and base classifiers unable to run in a reasonable time without resampling, we resample negative samples randomly so that

each dataset contains roughly the same number of negative and positive samples.

We skip the step of the positive semi-definite projection as our experiments are mainly concerned with the link prediction instead of a true measure. In the following experiments, we run each algorithm 50 times. AUDM only takes a few minutes to converge. For each individual network, we divide the obtained data into two parts: the training set and the testing set, and they contain 80% and 20% samples of the original dataset, respectively. We set each parameter by using 10-fold cross-validation in the training phase. In the training phase, 50000 iterations are executed on our platform. The longest training time is about two minutes.

According to the above description, first, we carry out a series of pretreatment for the used datasets. Then, according to the community compatibility matrix, a variable matrix $L^T_{i\to j}L_{j\to i}$ is calculated. This variable matrix indicates whether there is a link between two users (u_i and u_j) or not. Finally, according to the Bernoulli distribution, we resample links between users based on the variable matrix. All the experiments are conducted on a PC with i5 3.2GHz processor and 16GB memory. The above process is described in detail as follows.

Step 1: Feature matrix is obtained for each network.
Step 2: Potential feature vectors of users are obtained.
Step 3: Calculate the existing links of each user on the embedded network feature space.
Step 4: Resample links according to the Bernoulli distribution.
Step 5: Repeatedly run the program for 50 times.
Step 6: Calculate the values of area under receiver operating curve and mean average precision.

C. EXPERIMENTAL RESULTS AND ANALYSIS

In order to assess our algorithm, it is the evaluation task to predict who will establish contact with the others. We randomly divide the training set into 10 subsets for 10-fold cross-validation and then calculate the prediction error.

1) EXPERIMENTAL RESULTS

In order to evaluate the link prediction accuracy of the proposed algorithm, we divide the collected set of links (edges) D into the training set D_{tr} and the testing set D_{te}, which have the following relation:$D = D_{tr} + D_{te}$, $D_{tr} \cap D_{te} = \Phi$. The area under curve (AUC) [37] is a performance evaluation of metric facilitating comparison between different methods. AUC evaluation index is essentially a probability value, which represents the probability that the obtained link score is higher than a randomly selected link score in the testing set. The randomly selected link is selected from the links that do not exist in the network graph. Then, calculate a score for each two links. If the score of the former is greater than that of the latter, then the evaluation value is 1; If the scores are equal, then the evaluation value is 0.5. After that, we calculate the weighted average of the two values. We assume that n trials are evaluated and they are independent and not mutually affected. The results of evaluation include r times for '1' and s times for '0.5'. The AUC value is defined as $AUC = \frac{r+0.5s}{n}$. If the values of links are randomly generated, then the above comparison (greater, less or equal) is the same, and the AUC value is 0.5. Therefore, using AUC to indicate the accuracy of an algorithm needs to compare with 0.5, i.e., comparing the accuracy of an algorithm with that of the random selection method. The closer to 1 the AUC value, the better the prediction performance of the considered method.

The precision index evaluates the proportion of the accurate prediction links in the top links prediction. There are m accurate prediction links in the first l predictions, which are ranked according to the corresponding links scores. Thus, the precision index is defined as $precision = \frac{m}{l}$. We can easily see that the greater the *precision* value is, the higher the prediction accuracy becomes. When comparing two prediction algorithms, if their AUC values are equal but the precision value of algorithm A is greater than that of algorithm B, it can be inferred that algorithm A achieves better prediction performance than algorithm B. Concretely, the mean average precision (MAP) is used to compare different methods [38]. In the following, we give the experimental results measured using AUC and MAP, respectively.

Table 3 shows the performance of each method on each single social network. RTM and SRW only cope with data from one perspective so that their performance is poorer. SPML and our proposed approach are still poorly performing due to the data sparseness problem. Nonetheless, this provides evidence that our proposed approach is the best on each single dataset. Table 4 shows the AUC and MAP performance of the baseline methods and the proposed approach on the two fused networks. We can observe that the RTM and SRW still perform poorly due to that they consider network only from one perspective. SPML considers user features and link information, thus it improves the AUC and MAP than RTM and SRW. However, the SPML does not achieve satisfactory performance due to that it does not consider the community structure and link features. On the contrary, the proposed approach has better performance than the other methods on AUC and MAP. The AUC value obtained using AUDM is higher than that of baseline methods at least 0.0805 and 0.0271 in Douban and Tencent, respectively, while the MAP value obtained using AUDM is higher than that of baseline methods at least 0.0602 and 0.0507 in Douban and Tencent, respectively. This means that the AUDM has much better performance on sparse networks, and we believe that this is due to that our approach is based on the boosting framework of the instance weighting strategy, which benefits from the related social network and can alleviate the negative effects of network differences. Table 3 and Table 4 present that the distance metric learned using our new approach on multiple social networks is more accurate than that on a single social network.

In our experiment, the AUC that is related to network topology is recorded. Fig. 4 displays the AUC performance

TABLE 3. Comparison results of methods on AUC and MAP on single social networks.

Method	AUC				MAP			
	Douban		Tencent		Douban		Tencent	
	Online	Offline	QQ	Weibo	Online	Offline	QQ	Weibo
RTM	0.5990	0.6235	0.5623	0.6256	0.4170	0.5006	0.3310	0.3978
SRW	0.6005	0.6496	0.5589	0.6201	0.4138	0.5133	0.3394	0.4053
SPML	0.5755	0.6220	0.5429	0.6069	0.3906	0.4940	0.3101	0.3953
AUDM	**0.6346**	**0.6658**	**0.6308**	**0.6603**	**0.0.4534**	**0.0.5302**	**0.3553**	**0.4255**

TABLE 4. Comparison results of methods on AUC and MAP on two fused networks.

Method	AUC		MAP	
	Douban	Tencent	Douban	Tencent
RTM	0.6206	0.6017	0.6532	0.6093
SRW	0.6173	0.6085	0.6774	0.6148
SPML	0.6207	0.6106	0.6815	0.6715
AUDM	**0.7012**	**0.6377**	**0.7417**	**0.7222**

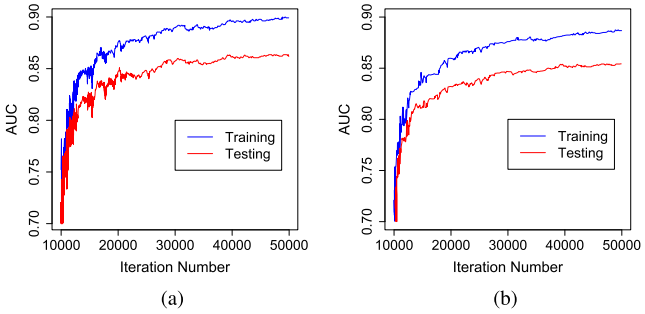


FIGURE 4. AUDM converges quickly after multiple iterations. (a) On Douban dataset. (b) On Tencent dataset.

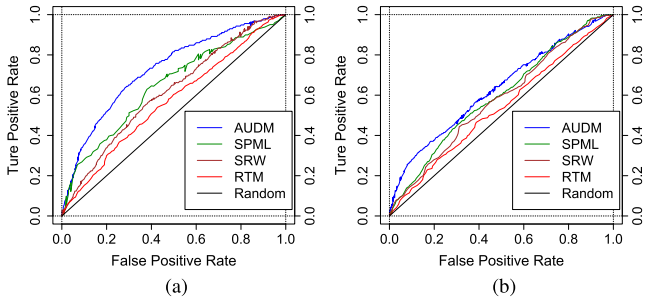


FIGURE 5. Average ROC performance for methods on two datasets. (a) On Douban dataset. (b) On Tencent dataset.

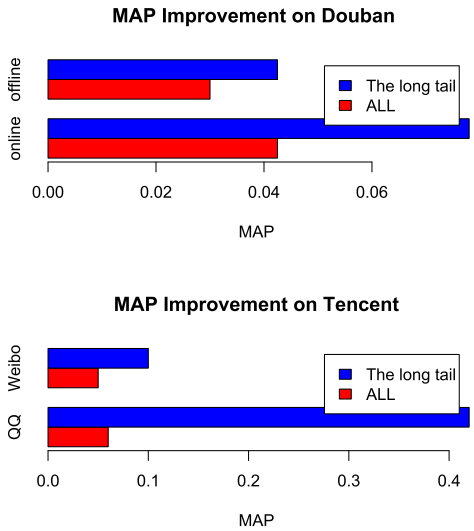


FIGURE 6. Performance of AUDM on Long tail users.

of AUDM on the training set and the testing set, respectively. We can clearly observe that the AUDM converges quickly after multi-iterations. In section III, the convergence of AUDM has been described in detail.

We assign each algorithm to rank edges according to the measure of link likelihood, and compare the accuracy of the rankings using Receiver Operator Characteristic (ROC) curves. Fig. 5 shows the ROC curves, where there is a strong lift for AUDM over competing methods on the two datasets. Why the SRW and RTM are unable to gain better performance over SPML? One possible explanation is that the wide range of degrees for nodes in the two datasets makes it difficult to find a threshold that divides the edges.

Our method can obtain enough knowledge to correctly predict their links for the users having enough neighbors. However, can the knowledge from the related network help solve the problem of network sparseness? We verify and report experimental results on the long tail users (the number of neighbors is less than ten) [39], who are restricted by the sparse problem. We use the boosting framework to make the user's mean average precision improved, as shown in Fig. 6. The reason is that our method can develop the knowledge from the other interconnected network to enhance link prediction for the long tail of users. This provides evidence for the proposed method to help solve the sparseness problem in a single network.

2) PARAMETER ANALYSIS

We study the influence of the parameters (i.e., the features number and λ) on the performance of the methods. As the number of features indicates the complexity of the user relationship network, we study the relationship between the number of the features and the performance of the methods. The obtained results are shown in Figure 7. We vary the number of features from 1 to 150 for the Douban dataset as shown in Fig.7(a) and from 1 to 110 for the Tencent dataset as shown in Fig.7(b). We notice that methods slightly improve their performance with the increase of the features number. In this

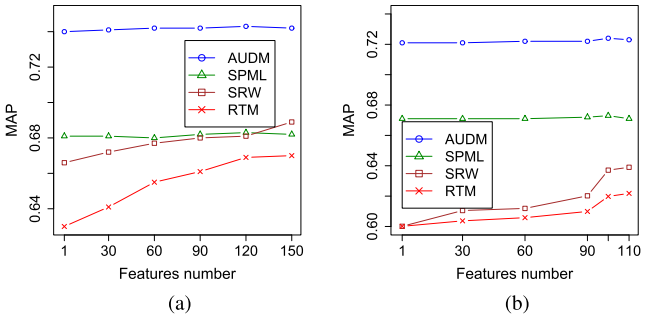


FIGURE 7. Performance of MAP with different feature numbers. (a) On Douban dataset. (b) On Tencent dataset.

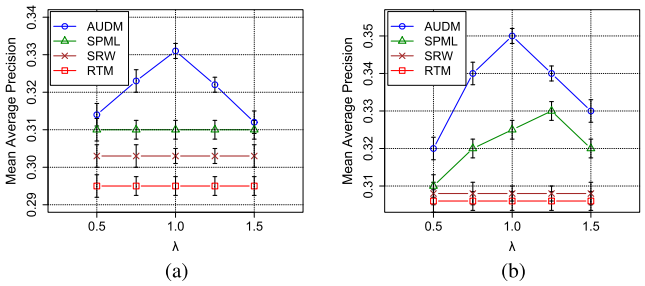


FIGURE 8. Performance of MAP with various λ (confidence coefficient of 0.95). (a) On Douban dataset. (b) On Tencent dataset.

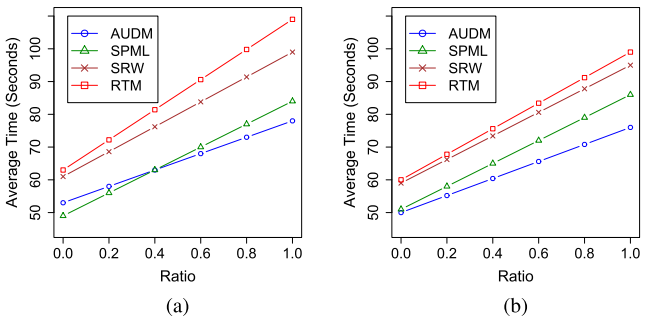


FIGURE 9. Time consumed with different ratios. (a) On Douban dataset. (b) On Tencent dataset.

sense, the methods are not very sensitive to the number of features on the two datasets.

λ represents the importance of trade-off between the graph topology and the community structure in the embedded social network. In Fig. 8, we can observe that λ varies from 0.5 to 1.5, and the step length is 0.25. The performance of our algorithm first increases and then gradually decreases with the increase of λ. Although the effectiveness of the graph topology is over-emphatic when λ is small, the impact of the community structure is ignored when λ increases. In particular, Fig. 8 shows that the performance of our method is better than that of SPML, SRW, and RTM. It means that keeping and modeling community structure is important in the embedded space because it essentially measures the potential similarities between two users.

3) TIME ANALYSIS

In section IV, we analyze the time complexity of our method and get a conclusion that the time complexity of our method is linear with the number of links. We verify the conclusion empirically as shown in Fig. 9. It illustrates the relationship between the computation time and the different ratios of the two datasets for the methods. The computation time increases linearly with respect to the scale of links. In our experimental setting, each iteration takes about 90 seconds. Not only Table 3 and Table 4 show that our method has better performance than SPML, SRW and RTM, but also Fig. 9 shows that our method is faster than the competing methods.

Finally, the comparative results show that our method outperforms the other methods in all the settings of both datasets. This is because our method combines metric learning with boosting for modeling user distance from the relevant social networks. Our method inherits the advantage of AdaBoost and can effectively tackle the over-fitting problem in the modeling of sparse social networks, while other methods in a single network are bound to encounter the data sparseness problem.

VI. CONCLUSION

In this paper, we exploited the boosting technology to construct a metric learning model that adaptively measured the distance between two users in relevant social networks. Then, we proposed the AUDM to address the challenging problem of modeling user distance in multiple social networks. In particular, our model is able to resist the over-fitting problem. Finally, extensive numerical experimental results on two real large-scale datasets demonstrate that our method outperforms the compared methods on AUC and MAP. In the future work, we will use our model on other issues (e.g., community detection and social influence analysis) in multiple social networks. In addition, we will develop a mechanism to resolve the conflict that some users have links and some users have no links in a social network but the relations among these users are swapped in another social network.

REFERENCES

[1] Z. Yang, Y. Xiang, K. Xie, and Y. Lai, "Adaptive method for nonsmooth nonnegative matrix factorization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 4, pp. 948–960, Apr. 2017.

[2] B. Wang *et al.*, "Similarity network fusion for aggregating data types on a genomic scale," *Nature Methods*, vol. 11, no. 3, pp. 333–337, Jan. 2014.

[3] J. Ma *et al.*, "Detecting rumors from microblogs with recurrent neural networks," in *Proc. IJCAI*, 2016, pp. 3818–3824.

[4] D. Tian, J. Zhou, and Z. Sheng, "An adaptive fusion strategy for distributed information estimation over cooperative multi-agent networks," *IEEE Trans. Inf. Theory*, vol. 63, no. 5, pp. 3076–3091, May 2017.

[5] E. E. Papalexakis, C. Faloutsos, and N. D. Sidiropoulos, "Tensors for data mining and data fusion: Models, applications, and scalable algorithms," *ACM Trans. Intell. Syst. Technol.*, vol. 8, no. 2, pp. 16:1–16:44, Oct. 2016. [Online]. Available: http://doi.acm.org/10.1145/2915921

[6] J. Shen, Y. Zhao, S. Yan, and X. Li, "Exposure fusion using boosting Laplacian pyramid," *IEEE Trans. Cybern.*, vol. 44, no. 9, pp. 1579–1590, Sep. 2014.

[7] G. Braun, C. Guzmán, and S. Pokutta, "Lower bounds on the oracle complexity of nonsmooth convex optimization via information theory," *IEEE Trans. Inf. Theory*, vol. 63, no. 7, pp. 4709–4724, Jul. 2017.

[8] P.-Y. Chen and I. W. Selesnick, "Group-sparse signal denoising: Non-convex regularization, convex optimization," *IEEE Trans. Signal Process.*, vol. 62, no. 13, pp. 3464–3478, Jul. 2014.

[9] Y. Yao and G. Doretto, "Boosting for transfer learning with multiple sources," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 1855–1862.

[10] C. Xiao and W. A. Chaovalitwongse, "Optimization models for feature selection of decomposed nearest neighbor," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 46, no. 2, pp. 177–184, Feb. 2016.

[11] J. A. Wall, L. J. McDaid, L. P. Maguire, and T. M. McGinnity, "Spiking neural network model of sound localization using the interaural intensity difference," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 4, pp. 574–586, Apr. 2012.

[12] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *J. Mach. Learn. Res.*, vol. 10, pp. 207–244, Feb. 2009.

[13] Y. Wang and H.-X. Li, "Burg matrix divergence-based hierarchical distance metric learning for binary classification," *IEEE Access*, vol. 5, pp. 3423–3430, 2017.

[14] S. Paisitkriangkrai, C. Shen, and A. van den Hengel, "Pedestrian detection with spatially pooled features and structured ensemble learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 6, pp. 1243–1257, Jun. 2016.

[15] Z.-H. Zhou, J. Wu, and W. Tang, "Ensembling neural networks: Many could be better than all," *Artif. Intell.*, vol. 137, no. 1, pp. 239–263, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S000437020200190X

[16] P. L. Bartlett and M. Traskin, "AdaBoost is consistent," *J. Mach. Learn. Res.*, vol. 8, no. 1, pp. 2347–2368, 2007.

[17] A. Clauset, C. Moore, and M. E. J. Newman, "Hierarchical structure and the prediction of missing links in networks," *Nature*, vol. 453, no. 7191, pp. 98–101, 2008.

[18] M. M. Bronstein, A. M. Bronstein, F. Michel, and N. Paragios, "Data fusion through cross-modality metric learning using similarity-sensitive hashing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 3594–3601.

[19] J. Bosveld, A. Mahmood, D. Q. Huynh, and L. Noakes, "Constrained metric learning by permutation inducing isometries," *IEEE Trans. Image Process.*, vol. 25, no. 1, pp. 92–103, Jan. 2016.

[20] P. Moutafis, M. Leng, and I. A. Kakadiaris, "An overview and empirical comparison of distance metric learning methods," *IEEE Trans. Cybern.*, vol. 47, no. 3, pp. 612–625, Mar. 2017.

[21] D. M. Johnson, C. Xiong, and J. J. Corso, "Semi-supervised nonlinear distance metric learning via forests of max-margin cluster hierarchies," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 4, pp. 1035–1046, Apr. 2016.

[22] J. Hou, H. Gao, Q. Xia, and N. Qi, "Feature combination and the kNN framework in object classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 6, pp. 1368–1378, Jun. 2016.

[23] K. Kim, H. Lin, J. Y. Choi, and K. Choi, "A design framework for hierarchical ensemble of multiple feature extractors and multiple classifiers," *Pattern Recognit.*, vol. 52, pp. 1–16, Apr. 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320315004264

[24] J. F. Díez-Pastor, J. J. Rodríguez, C. I. García-Osorio, and L. I. Kuncheva, "Diversity techniques improve the performance of the best imbalance learning ensembles," *Inf. Sci.*, vol. 325, pp. 98–117, Dec. 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020025515005186

[25] P.-B. Zhang and Z.-X. Yang, "A novel adaboost framework with robust threshold and structural optimization," *IEEE Trans. Cybern.*, to be published. [Online]. Available: https://doi.org/10.1109/TCYB.2016.2623900

[26] A. Halevy, N. Noy, S. Sarawagi, S. E. Whang, and X. Yu, "Discovering structure in the universe of attribute names," in *Proc. 25th Int. Conf. World Wide Web (WWW)*, 2016, pp. 939–949. [Online]. Available: https://doi.org/10.1145/2872427.2882975

[27] U. Brandes *et al.*, "On modularity clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 2, pp. 172–188, Feb. 2008.

[28] G. Rätsch, T. Onoda, and K.-R. Müller, "Soft margins for adaboost," *Mach. Learn.*, vol. 42, no. 3, pp. 287–320, 2001. [Online]. Available: http://dx.doi.org/10.1023/A:1007618119488

[29] A. Mobasher, M. Taherzadeh, R. Sotirov, and A. K. Khandani, "A near-maximum-likelihood decoding algorithm for MIMO systems based on semi-definite programming," *IEEE Trans. Inf. Theory*, vol. 53, no. 11, pp. 3869–3886, Nov. 2007.

[30] Z. Wang, M.-J. Lai, Z. Lu, W. Fan, H. Davulcu, and J. Ye, "Rank-one matrix pursuit for matrix completion," in *Proc. ICML*, 2014, pp. 91–99.

[31] T. Trzcinski, M. Christoudias, and V. Lepetit, "Learning image descriptors with boosting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 597–610, Mar. 2015.

[32] K.-A. Toh, Q.-L. Tran, and D. Srinivasan, "Benchmarking a reduced multivariate polynomial pattern classifier," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 6, pp. 740–755, Jun. 2004.

[33] B. Shaw, B. Huang, and T. Jebara, "Learning a distance metric from a network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 1899–1907.

[34] L. Backstrom and J. Leskovec, "Supervised random walks: Predicting and recommending links in social networks," in *Proc. 4th ACM Int. Conf. Web Search Data Mining (WSDM)*, New York, NY, USA, 2011, pp. 635–644. [Online]. Available: http://doi.acm.org/10.1145/1935826.1935914

[35] J. Chang and D. M. Blei, "Hierarchical relational models for document networks," *Ann. Appl. Stat.*, vol. 4, no. 1, pp. 124–150, 2010.

[36] L. Muñoz-González, M. Lázaro-Gredilla, and A. R. Figueiras-Vidal, "Laplace approximation for divisive Gaussian processes for nonstationary regression," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 3, pp. 618–624, Mar. 2016.

[37] W. Xu, S. Liu, X. Sun, S. Liu, and Y. Zhang, "A fast algorithm for unbiased estimation of variance of AUC based on dynamic programming," *IEEE Access*, vol. 4, pp. 9553–9560, 2016.

[38] H. Liu, X. Kong, X. Bai, W. Wang, T. M. Bekele, and F. Xia, "Context-based collaborative filtering for citation recommendation," *IEEE Access*, vol. 3, pp. 1695–1703, Oct. 2015.

[39] F. Cai, S. Liang, and M. de Rijke, "Prefix-adaptive and time-sensitive personalized query auto completion," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 9, pp. 2452–2466, Sep. 2016.

YUFEI LIU received the M.S. degree in computer science from the College of Computer Science and Information Engineering, Northwest University for Nationalities, China, in 2009. He is currently pursuing the Ph.D. degree with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China. His research interests include data mining, machine learning, and social network analysis.

DECHANG PI received the Ph.D. degree from the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics (NUAA), China. He is currently a Professor with the College of Computer Science and Technology, NUAA, where he is also the Vice Dean. His research interests include data mining and social network analysis.

LIN CUI received the master's degree from the School of Computer Science and Information Engineering, Hefei University of Technology, China, in 2008. She is currently pursuing the Ph.D. degree with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China. She is also an Associate Professor with the Intelligent Information Processing Laboratory, Suzhou University, China. Her research interests include data mining, POI recommendation, and social network analysis.