IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received March 9, 2017, accepted March 23, 2017, date of publication July 4, 2017, date of current version July 24, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2723433

Behavior-Based Collective Classification in Sparsely Labeled Networks

JUNYI XU[1], LE LI[1], XIN LU[1,2,3], SHENGZE HU[1], BIN GE[1], WEIDONG XIAO[1], AND LI YAO[1]

[1]College of Information System and Management, National University of Defense Technology, Changsha 410073, China
[2]Department of Public Health Sciences, Karolinska Institute, 17177 Stockholm, Sweden
[3]Key Laboratory of Surveillance and Early-Warning on Infectious Disease, Chinese Centre for Disease Control and Prevention, Division of Infectious Disease, Beijing 102206, China

Corresponding author: Le Li (lile10@126.com)

This work was supported by the National Natural Science Foundation of China under Grant 71371184 and Grant 71331008.

ABSTRACT Classification in sparsely labeled networks is challenging to traditional neighborhood-based methods due to the lack of labeled neighbors. In this paper, we propose a novel behavior-based collective classification (BCC) method to improve the classification performance in sparsely labeled networks. In BCC, nodes' behavior features are extracted and used to build latent relationships between labeled nodes and unknown ones. Since mining the latent links does not rely on the direct connection of nodes, decrease of labeled neighbors will have minor effect on classification results. In addition, the BCC method can also be applied to the analysis of networks with heterophily as the homophily assumption is no longer required. Experiments on various public data sets reveal that the proposed method can obtain competing performance in comparison with the other state-of-the-art methods either when the network is labeled sparsely or when homophily is low in the network.

INDEX TERMS Behavior feature, sparsely labeled networks, collective classification, within-network classification.

I. INTRODUCTION

Given a partially labeled network, in which labels of some nodes are known, within-network classification aims to predict labels of the rest nodes. Due to the increasingly wide applications in counterterrorism analysis [1], [2], fraud detection [3], [4] and product recommendations [5], [6] etc., within-network classification has received a lot of attention in recent years.

Conventional classification methods assume the data is independent and identically distributed (i.i.d.). Nevertheless, in network data, the nodes are interconnected with each other, making the label of nodes are correlated with not only its own attributes, but also the label of neighbors [7]–[11]. For example, wvRN [7], [8] predicts the label of unknown nodes via a weighted average of the estimated class membership of the node's neighbors. In a range of real networks, wvRN has shown to obtain a surprisingly good performance [7]. However, wvRN relies heavily on the homophily assumption, i.e., nodes belonging to the same class tend to be linked with each other [12], and thereby are limited in the analysis of networks where nodes are not clustered by the studied property. Probabilistic relational models [9]–[11] can overcome this limitation. In probabilistic relational models, by constructing the dependence between connected nodes, the probability of an unknown node's label is conditioned not only on the labels of its neighbor nodes, but also on all observed data (i.e., network structure and all labeled nodes).

While the rapid development of information technology has greatly improved our ability to collect data in recent years, traditional methods of network classification are facing new challenges: in the era of big data, substantial proportion of nodes are typically unlabeled in many settings. For such sparsely labeled networks, the neighbors of an unknown node are mostly unlabeled as well [13]; consequently, many neighborhood-based methods cannot achieve satisfied performance for such kind of networks. For this reason, a lot of efforts have been made recently in order to develop new techniques for sparse labeling problem, such as semi-supervised learning [14], [15], active learning [16]–[19] and latent link mining [13], [20], [21].

All the above methods can handle the sparse labeling problem to some extent, however, the interacting behavior of nodes, which is important to the formation of network structure, is not considered. In addition, as pointed in [21], when the number of nodes in one class is much larger than the other class, unknown nodes are more likely to be classified as the same category as the majority.

To overcome these limitations, we propose a novel behavior based collective classification (BCC) method for network data in this study. In the new method, firstly, we extract the behavior feature of nodes in the network; then, instead of including all labeled nodes in the classification process, we screen valuable nodes which are most relevant for the classification; finally, since latent links can be estimated between unknown nodes and valuable nodes by analyzing their behavior feature, collective classification is performed based on the latent links to infer the class of unknown nodes. Experiment reveals that the method performs competitively on several public real-world datasets and can overcome the challenge of classification in sparsely labeled networks and networks with lower homophily.

The rest of the paper is organized as follows: We review related existing work in Section II, and propose the behavior based collective classification method in Section III. Designing and realization of BCC are presented in Section IV, with a focus on the behavior feature extraction, similarity analysis and collective classification. Details of experimental setup are introduced in Section V, and extensive experimental results are demonstrated and discussed in Section VI, followed by conclusion in Section VII.

II. RELATED WORK

A. SEMI-SUPERVISED LEARNING

Making use of both labeled and unlabeled data, semi-supervised learning is an effective method for classification in sparsely labeled networks [22], [23]. One type of this method is to design a classification function which is sufficiently smooth with respect to the intrinsic structure collectively revealed by labeled and unlabeled points [24]. Zhou *et al.* [24] propose a simple iteration algorithm, which considered global and local consistency by introducing a regularization parameter. By modeling the network with constraint on label consistency, Zhu *et al.* [25] propose a Gaussian random field (GRF) method by introducing a harmonic function, of which the value is the average of neighboring points. Another type of semi-supervised learning methods is the graph-cut method [26]–[28], which assumes that more closely connected nodes tend to belong to the same category. The core idea is to find a cut set with the minimum weight by using different criteria. However, the high cost of computing often lead to poor performance of the algorithm when applied in large networks. Some other algorithms use random walk on the network to obtain a simple and effective solution by propagating labels from labeled nodes to unknown nodes. Based on passaging time during random walks with bounded lengths, Callut *et al.* [29] and Newman [30] introduce a novel technique, called D-walks, to handle semi-supervised classification problems in large graphs. Zhou and Schlkopf [31] define calculus on graphs by using spectral graph theory, and propose a regularization framework for classification problems on graphs. However, many semi-supervised learning methods rely heavily on the assumption that the network exhibits homophily, i.e., nodes belonging to the same class tend to be linked with each other [12]. Meanwhile, the implementation of semi-supervised learning algorithm often requires a large amount of matrix computation, and thus is infeasible for processing large datasets [25]. Many methods have been developed to overcome these limitations. For example, Tong *et al.* propose a fast random walk with restart algorithm [32] to improve the performance on large-scale dataset. Lin *et al.* propose a highly scalable method, called Multi-Rank-Walk (MRW), which requires only linear computation time in accordance to the number of edges in the network [12]. Mantrach *et al.* [33] design two iterative algorithms which can be applied in networks with millions of nodes to avoid the computation of the pairwise similarities between nodes. Gallagher *et al.* [13] design an even-step random walk with restart (Even-step RWR) algorithm, which mitigates the dependence on network homophily effectively.

B. ACTIVE LEARNING

In active learning [34], the number of known labels required for accurate learning is reduced by intelligently selecting to-be-labeled nodes to achieve improved classification performance in sparsely labeled networks. Lewis and Catlett [35] propose a method based on uncertainty reduction, which selects the data with lowest certainty for querying. However, the method will fail when there are a certain number of outliers. The outliers have high uncertainty in the network, but getting their labels doesn't help to inference the rest data. To handle this limitation, Roy and McCallum [36] design a method to determine the impact on the expected error of each potential labeling request by using Monte Carlo approach. In the active learning process, the feature of linked data in the network can also be taken into account. Bilgic and Getoor [18] propose several ways of adapting existing active learning methods to network data. Macskassy [37] designs a novel hybrid approach by using community detection and social network analytic centrality measures to identify the candidates for labeling. When network structure and node attribute information are available, Bilgic *et al.* [19] apply several classic active learning strategies such as disagreement and clustering to select samples for labeling, which has shown significant improvements over baseline methods. Active learning is able to overcome the sparse labeling problem to some extent, but it still requires the participation of experts and lacks an automatic learning process.

C. LATENT LINK MINING

In sparsely labeled networks, the neighbors of unknown nodes are mostly unlabeled as well, so the key idea of latent link mining is to find the relationship between labeled nodes and unknown nodes. When the dataset is non-relational, there are many methods [27], [38] which can transform the data into a weighted network and estimate latent links. For example, Wang and Zhang [39] create links by calculating similarity scores between pair of nodes, and each nodes will link to the K instances that are most similar to them. Recently, several novel techniques, which use only network structure to mine latent links, are proposed. For example, Gallagher *et al.* [13] use an even-step random walk with restart (Even-step RWR) algorithm to estimate latent links between the labeled nodes and unknown nodes. Zhang *et al.* [21] apply several similarity-based link prediction methods in sparsely labeled network classification. By comparing the similarity of nodes, latent links can be mined between labeled nodes and unknown nodes, and link weights are calculated according to different similarity indices. However, the inference of these methods are based on information from all labeled nodes, i.e., without considering the impact of noisy data or unrelated data. Therefore, when the network becomes large, many unrelated nodes will be included in the classification process and may reduce the performance of algorithms.

In a sparsely labeled network, the labels of nodes are indeed fewer, but the attributes of nodes are still sufficient. Here we indicate the differences of label and attribute in a network. Take webpage network for example, a webpage in such network can be treated as a node, and an edge exits if there is a hyperlink from one webpage to another. In this situation, the category of a webpage can be treated as the label, and the content of the webpage are treated as attribute. From this point of view, the sparsely labeled problem can be solved by using the attributes for classification. For instance, we can build a local classifier based on node's attributes, and combine it with relational features to make prediction [7]. In this situation, the result of local classifier can be treated as preprocessing result for the relational classifier, or as a distinct classifier that makes separate prediction. Moreover, in network data, Macskassy [20] tries to compare the similarity of nodes based on their attributes. If the similarity exceeds a certain threshold, a new link will be created. Then it combines explicit links such as hyperlinks between webpages and latent links to predict unknown nodes.

It can be seen that the attributes of nodes can improve the performance in sparsely labeled networks to some extent. However, the attributes information may not be accessed due to the privacy and security in some situations. Therefore, without losing generality, we build the BCC method as a relational model that only utilized network structure and label information. Moreover, if the attributes information is given, it can be used to build a local classifier, and combined with BCC to make prediction easily.

D. SOCIAL DIMENSIONS EXTRACTION

Network often consists of various relations, however most existing approaches treat these relations as the same type and lead to poor performance. In order to handle this problem, Tang and Liu [40] propose a novel classification framework, SocioDim, to learn a classifier based on social dimensions extracted from network structure. Various methods can be used to mine the social dimensions, for example, Tang and Liu [40] choose spectral clustering to extract social dimensions and use Support Vector Machine (SVM) for classification. Modularity maximization [41] can also be used to calculate social dimensions, but the computation complexity is too high to apply in large dataset. EdgeCluster [42] is another effective method which uses an edge-centric k-means clustering to obtain social dimensions. It has been shown to perform comparably to the Modularity maximization method, with the added advantage of scaling to graphs which are too large for spectral decomposition. Wang and Sukthankar [43] use the same method to construct the social feature space and estimate a node's label based on its neighbors' class labels, the similarity between connected nodes, and its class propagation probability.

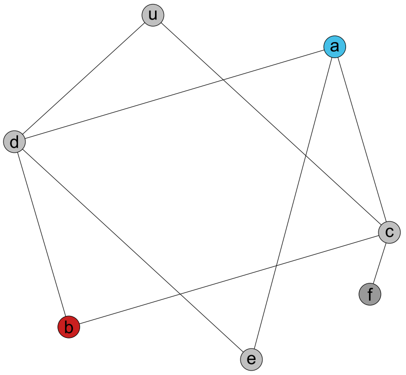


FIGURE 1. Toy Example: a sparsely labeled network. The red and blue colors represent the labels of nodes, and nodes with gray color are unknown nodes.

As we will see in the next section, our method does not make use of social dimension, but tries to extract the behavior feature of each node to find the latent relationships in the network. With these latent relationships, we can avoid modeling various types of edges and handle various relations in the classification process effectively.

III. METHOD

In this section, we will describe the intuition of behavior based classification at first, and show that the behavior feature is more discriminative than traditional similarity measures. Then, the framework of our method is introduced in detail.

A. INTUITION

In sparsely labeled networks, the labels of nodes are much fewer, making it difficult to leverage label dependencies to make accurate prediction. Without considering the label information, it can be found that the network structure can still provide useful information. Therefore, most researches focus on utilizing the network structure to predict unknown nodes. For example, CN method [21] estimates the similarity of nodes by local structure (the number of common neighbors). However, it becomes ineffective when handling the sparsely labeled network classification task in some situations.

Figure 1 shows a sparsely labeled network, in which only node *a* and node *b* are labeled and the task is to predict the

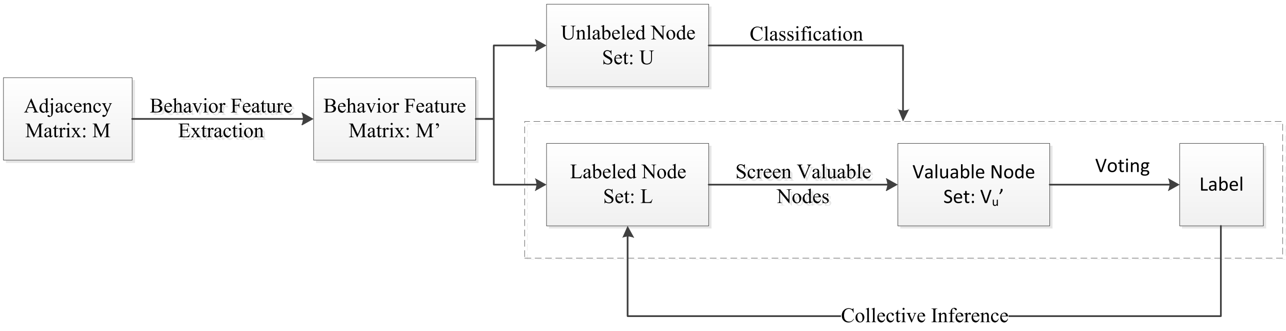


FIGURE 2. Framework of BCC.

label of node u (the true color is ''red''). CN method considers that node u has two common neighbors with node a, so the similarity between node u and node a is 2. Then we can find that the similarity between node u and node b is 2 as well. In this situation, CN method cannot determine which is the most similar node with u, and thus, leading to lower performance.

Traditional methods consider the network structure is fixed, and estimate the similarity of nodes by local or global measures (such as common neighbors, random walk, etc.). However, from the perspective of network evolution, the network is generated by the interaction behavior of nodes. In the generation process, the interaction behavior and label of nodes may be of high correlation, i.e., nodes tend to connect with other nodes based on their labels (such as interest, gender, etc.). For instance, the persons with ''pet'' label tend to follow veterinarian and pet photographer. An intuition is that we can utilize similarity of behavior feature to predict the label of nodes.

Considering the connection behaviors in Fig. 1, it can be found that node u only connects with node c and node d, so does node b. In contrast, node a connects with three nodes c, d and e. As it can be seen, the behavior features of node u and node b are the same, our method tends to treat node b as the most similar node with node u, and thus, achieving more accurate results.

B. BEHAVIOR BASED COLLECTIVE CLASSIFICATION

Since behavior feature can provide a different kind of information that may be useful in sparsely labeled networks, we propose a novel Behavior-based Collective Classification method (BCC) in this paper to handle the sparse labeling problem. The process of BCC in network data consists of four steps: behavior feature extraction, screening valuable nodes, classification by voting and collective inference. The framework is shown in Fig. 2.

As shown in Fig. 2, we assume that nodes may belong to the same class if their behavior features are similar. Therefore, given the adjacency matrix M of a network, we will extract nodes' behavior feature at first to obtain the feature matrix $M^{'}$, of which the i-th row vector is the behavior feature of node i.

Instead of including all labeled nodes, BCC only allows the most relevant nodes for classification to improve the performance on sparsely labeled networks. So in the next, we screen valuable nodes by using correlation analysis and similarity analysis respectively. Given an unknown node u, we first compare the correlation between u and each labeled node, then, nodes with correlation coefficients exceeding a threshold will be added into the valuable node set V_u. After that, we compare the similarity between u and each node in V_u, and add the top-K similar nodes into set $V_u^{'}$, which is then used to classify the unknown node u by voting. It should be noted that, our method is flexible to integrate other techniques in each step, e.g., classification by voting can be replaced by other classifiers, such as SVM, linear regression and so on. Finally, in order to deal with challenges of classification in extremely sparsely labeled network, we perform collective inference, in which the newly labeled nodes will be added to the labeled node set and used for inferring the rest unknown nodes.

IV. IMPLEMENT

BCC method consists of four steps for classification, and in this section, we introduce the implement of each step in detail. Firstly, we will describe how to extract behavior feature, which has shown more discriminative ability in sparsely labeled networks. In order to handle the imbalanced dataset, we only allow the most relevant nodes in the classification process by using correlation and similarity analysis. Then we introduce the strategy of voting for classification. Collective inference procedure is used to handle the extremely sparse labeling problem, which is described afterwards. Finally, the algorithm is given to show the details of our method.

A. BEHAVIOR FEATURE EXTRACTION

Let $w(i, j)$ be the weight of the edge from node i to node j, then the adjacency vector $\overrightarrow{w}_i = \{w(i, 1), w(i, 2), \ldots, w(i, N)\}$ can be used to describe the behavior pattern of node i. However, it should be noted that $\vec{w}_i$ is the observed value in the current time, which may change by time with the evolution of network. Therefore, instead of using $\vec{w}_i$, we need to extract more stable behavior feature to be able to reflect nodes' intrinsic attribute.

In the network generation process, we assume that each node in the network has a certain probability to connect with other nodes. The observed connection behaviors of nodes are driven by the implicit individual-based probabilities, which are stable and may reflect the nodes' essential attributes. Let $p_{(i,j)}$ be the connection probability for node i to connect node j, then $\vec{p}_i = \left\{p_{(i,1)}, p_{(i,2)}, \ldots, p_{(i,N)}\right\}$ is the probability distribution of node i to connect other nodes in the network, which can be treated as the behavior feature of node i.

As the number of nodes is fixed in the network, the probability distribution of connecting behavior can be regarded as a multinomial distribution. In order to integrate prior knowledge and simplify calculations, we choose dirichlet distribution, which is conjugate with the multinomial distribution, as prior distribution. $\vec{p}_i$ is node i's probability distribution of connecting behavior, which has a dirichlet prior distribution $\sim Dir(\vec{p}_i|\vec{\alpha})$ with hyperparameter $\vec{\alpha}$; and $w(i,j)$ is the number of observed connections from nodes i to node j, which follows a multinomial distribution $\sim Mult(w(i,j)|\vec{p}_i)$. As it can be seen, the latent behavior feature can be extracted by maximizing the posterior.

Let S_i be the set of nodes connected with node i and s_m be the m-th connected node in S_i, let $w(i,j)$ be the weight of the edge from node i to node j, and $p_{(i,j)}$ be the probability of connection from node i to node j, then we can calculate node i's probability distribution of connecting behavior $\vec{p}_i$ as follows:

$$\begin{aligned}
p(\vec{p}_i \mid S_i, \vec{\alpha}) &= \frac{p(S_i \mid \vec{p}_i)p(\vec{p}_i \mid \vec{\alpha})}{p(S_i, \vec{\alpha})} && (1)\\
&= \frac{\prod_{m=1}^{M} p(s_m \mid \vec{p}_i)p(\vec{p}_i \mid \vec{\alpha})}{\int \prod_{m=1}^{M} p(s_m \mid \vec{p}_i)p(\vec{p}_i \mid \vec{\alpha})d\vec{p}_i} && (2)\\
&= \frac{\prod\limits_{j=1}^{N} {p_{(i,j)}}^{w(i,j)} \frac{1}{\Delta(\vec{\alpha})} {p_{(i,j)}}^{\alpha_j - 1}}{\int \prod_{m=1}^{M} p(s_m \mid \vec{p}_i)p(\vec{p}_i \mid \vec{\alpha})d\vec{p}_i} && (3)\\
&= \frac{\Delta(\vec{\alpha})}{\Delta(\vec{\alpha}+\vec{w}_i)} \prod_{j=1}^{N} {p_{(i,j)}}^{w(i,j)} \frac{{p_{(i,j)}}^{\alpha_j - 1}}{\Delta(\vec{\alpha})} && (4)\\
&= \frac{1}{\Delta(\vec{\alpha}+\vec{w}_i)} \prod_{j=1}^{N} p_{(i,j)}^{w(i,j)+\alpha_j - 1} && (5)\\
&= Dir(\vec{p}_i \mid \vec{\alpha}+\vec{w}_i), && (6)
\end{aligned}$$

in which $\frac{1}{\Delta\ (\vec{\alpha})} = \frac{\Gamma\ (\sum_{k=1}^{N} \alpha_k)}{\prod_{k=1}^{N} \Gamma\ (\alpha_k)}$, $p_{(i,j)} = \frac{w(i,j)+\alpha_j}{\sum_{j=1}^{N} (w(i,j)+\alpha_j)}$ is the expectation of the Dirichlet distribution, and $\vec{p}_i = \left\{p_{(i,1)}, p_{(i,2)}, \ldots, p_{(i,N)}\right\}$ is the extracted behavior feature of node i.

It is worth noting that the behavior feature is very similar to the term "degree distribution" in network literature, the difference is that, instead of calculating overall connection frequency distribution, in our method the behavior feature is individual-based and the probability vector $\vec{p}$ represents connection behavior feature of each individual.

B. SCREEN VALUABLE NODES FOR CLASSIFICATION

The labeled nodes are much fewer in sparsely labeled network, so traditional methods tend to utilize all the labeled nodes in the classification process. However, involving unrelated nodes in the classification process will only bring noise data and lead to poor performance. Moreover, when classes of labeled nodes are imbalanced, unknown nodes will be more likely to be labeled the same as the majority. To solve this issue, we show how to find the most relevant nodes, from the perspective of correlation and similarity of behavior feature, to reduce the impact of noise data.

1) CORRELATION OF BEHAVIOR FEATURE

Correlation analysis is an important method to measure the relationship between two observed variables. We assume that nodes of the same class should have higher correlation of their behavior feature. Therefore, given an unknown node u, the labeled node set L, and Pearson correlation threshold P, we can screen out the valuable node set V_u by:

$$V_u = \{v | v \in L \wedge corr(v,u) > P\}, \tag{7}$$

where $corr(v,u)$ represents pearson correlation value between node v and u. $corr(v,u)$ can be calculate by $corr\ (v,u) = \frac{1}{N-1} \sum\limits_{i=1}^{N} (\frac{v_i - \bar{v}}{s_v})(\frac{u_i - \bar{u}}{s_u})$, where N is the number of nodes in the network, $\bar{v}$ is the mean value of node v's behavior feature vector, s_v is the standard deviation of node v's behavior feature vector, and analogously for $\bar{u}$ and s_u. As we will see in the experiments, labeled nodes of higher correlation with u will have bigger influence in the classification process.

TABLE 1. Summary of connection behaviors.

Node index	A	B	C	D	E	F	G
A	0	0	0	10	40	60	70
B	0	0	0	10	40	61	70
C	0	0	0	100	400	600	700

2) SIMILARITY OF BEHAVIOR FEATURE

Correlation analysis is able to discover the latent relationship of behavior features, but not enough for finding the most relevant nodes in weighted networks. For example, in Table 1, it can be found that the connection behavior of node A and node B are almost same, except subtle changes when connecting node F. As we know, experimental datasets are crawled from real-world networks. In the crawling process, information may be lost inevitably, which means node A and node B may have the same connection behaviors with node F in real-world network. In this situation, it is obvious that the connection behavior of node B is more similar with A compared to C. However, by using the correlation analysis, C will have a higher correlation value with A ($corr(A,C) = 1$, $corr(A,B) = 0.99$).

In order to improve the ability to handle this problem, we implement a similarity analysis procedure after the correlation analysis. We assume that nodes of the same class should have more similar behavior features. Since nodes' behavior features are expressed as probability distributions, symmetric Kullback-Leibler (KL) divergence [44] can be used to measure the similarity:

$$D_{sKL}(i,j) = \frac{1}{2}[\sum_{n=1}^{N} p_{(i,n)} \ln \frac{p_{(i,n)}}{p_{(j,n)}} + \sum_{n=1}^{N} p_{(j,n)} \ln \frac{p_{(j,n)}}{p_{(i,n)}}]. \quad (8)$$

Where $p_{(i,j)}$ is the probability of connection from node i to node j.

A node with smaller KL divergence will indicate that it has similar behavior feature to the unknown node and thus is more valuable for the classification. Therefore, given the unknown node u, we calculate the similarity of node u with each node in V_u, and add the top-K similar nodes to set $V_u^{'}$.

For the example in Table 1, by using similarity analysis, we can find that the KL divergence between node B and node A is much smaller ($D_{sKL}(A, B) = 0.000043724$, $D_{sKL}(A, C) = 0.0055$), which identifies B as the more relevant node.

C. BEHAVIOR BASED CLASSIFICATION BY MAJORITY-VOTING

After the above screening process, the valuable node set $V_u^{'}$, is then used to classify unknown nodes. We use the majority-voting strategy, which means that the label of an unknown node is determined by the class of nodes which belongs to the majority in $V_u^{'}$:

$$C(u|V_u^{'}) = \arg\max_{C_j} \sum_{x \in V_u^{'}} I(C(x) = C_j), \quad j = 1, \cdots, J, \quad (9)$$

in which $C(u)$ represents the class of node u, J is the total number of classes in the network, and C_j is the j-th class. $I(\cdot)$ is a discriminant function such that when $C(x) = C_j$, $I(\cdot) = 1$ and otherwise $I(\cdot) = 0$.

D. COLLECTIVE INFERENCE

In order to improve the classification performance in sparsely labeled network, collective inference procedure is introduced in our method, in which newly labeled nodes will be used for inferring the rest unknown nodes. Consequently, as the classification process goes on, the labeled node set expands constantly and existing knowledge continues to accumulate to guide subsequent classification process.

However, introducing collective inference process will come with a new problem: unknown nodes that have been labeled will affect subsequent prediction process, so labeling is relevant to the order of how unknown nodes are classified. To mitigate such effect, we propose an iteration strategy. In the i-th iteration, the labeled node set L_i will use the labels at the end of the previous iteration. Then, each initial unknown node will be classified by using behavior based classification method and get a new label. If the node has never been labeled in the previous iteration, it will be added to L_i; otherwise we will update L_i with the new label. The iteration continues until labels of all initial unknown nodes stay unchanged in L_i or the maximum number of iterations is reached.

This process inherits the idea of iterative classification (IC) method [45]. However, instead of using local neighbors, our method relies on latent links created by behavior feature. Since we extract a few valuable nodes to participate in the classification, it does not need to update numerous nodes in each iteration and the process typically converges efficiently in a limited number of iterations.

When the labeled data is very sparse, the performance of traditional collective classification might be largely degraded due to the lack of sufficient neighbors [13]. However, in our method, latent links can be mined between labeled nodes and unknown nodes by using behavior feature, even nodes do not connect directly. It means that in our method, the label of node u is only affected by valuable nodes in $V_u^{'}$, rather than its local neighbors. Therefore, decrease of labeled neighbors will have minor effect on classification performance, making BCC more suitable for handling sparse labeling problem. Moreover, we can see that the proposed method does not rely on the homophily assumption, so it can be applied to network with lower homophily as well.

Finally, the algorithm of BCC is presented in Table 2.

V. EXPERIMENTAL SETUP

A. DATASETS

We evaluate the proposed BCC method by comparing it with other baseline methods for the classification performance on the following real-world datasets.

1, Enron emails [46]. We choose 151 persons as nodes in the network and retain 2235 edges connected between these persons. 102 out of these 151 nodes are assigned a role according to the role list [47], where 37 nodes are labeled as "employee". The network is a small directed weighted graph composed of email communication among users and the experimental task is to identify the "employee" class.

2, WebKB [48]. *WebKB* is a dataset of webpages gathered from different universities, in which nodes are webpages and edges are hyperlinks. Each webpage is classified into one of the five classes: "course, faculty, student, project, staff". There are four networks in this data set: *cornell*, *texas*, *washington* and *wisconsin*, and the task is to identify the "student" webpage.

3, Cora [48]. *Cora* is a citation network formed of 2708 scientific publications and 5429 links. Each publication is classified into one of seven classes. The task is to identify the "Neural_Networks" class.

4, Citeseer [48]. *Citeseer* is a citation network consists of 3312 scientific publications and 4732 links. Each publication is classified into one of six classes. The task is to identify the "DB" class.

B. BASELINE METHODS

The following four representative baseline methods are selected for comparison with the BCC method:

TABLE 2. The algorithm of BCC.

```
input: $(N, P, K, H)$
/* $N$: Sparsely labeled network (the adjacency matrix and labels of some nodes);
$P$: Pearson correlation threshold; $K$: Number of most similar nodes; $H$: Hyperparameter*/
output: $(N')$
/*$N'$: Fully labeled network (the adjacency matrix and labels of all nodes)*/
$L$ = Get_labeled_nodes $(N)$        /* labeled node set: $L$ */
$U$ = Get_unlabeled_nodes $(N)$        /* initial unknown node set: $U$ */
$M$ = Get_adjacency_matrix$(N)$
$M'$ = Expectation_Dirichlet_distribution$(M, H)$        /* behavior feature extraction */
while ( convergence != true and iteration_num $<$ MAX )
    for each $u$ in $U$
        $V_u$ = pearson_correlation $(u, M', L, P)$        /* correlation analysis */
        $V'_u$ = symmetric_KL_divergence $(u, M', V_u, K)$        /* similarity analysis */
        $l(u)$ = Get_label_by_vote $(V'_u)$        /* majority - voting */
         $L = L + \{u,l(u)\}$        /* labeled node is added to the labeled node set */
    end for
end while
$N'$= output_ labeled_network $(N, L)$
```

TABLE 3. Statistics of the four networks in WebKB.

name	nodes	edges
cornell	195	304
texas	187	328
washington	230	446
wisconsin	265	530

a: SIMILARITY-BASED CLASSIFICATION BY USING COMMON NEIGHBORS (SC_CN) [21]

SC_CN is a popular technique to solve the sparse labeling problem. As one of the fundamental methods, Common Neighbors method has been widely used for link prediction [49], which utilized the number of common neighbors as the similarity measure. Intuitively, this similarity measure can be used to find similar nodes to predict the label of unknown nodes. For each pair of nodes, x and y, SC_CN uses the number of common neighbors to calculate a similar score as $s_{x,y}$. Given an unknown node, u, the total number of classes in the network, J, and the j-th class C_j, the probability for u belonging to C_j is:

$$p(C_j|u) = \frac{\sum_{\{v|label(v)=C_j\}} s_{u,v}}{\sum_{\{v|label(v)\neq\emptyset\}} s_{u,v}}, \quad j = 1, 2, \ldots, J. \tag{10}$$

The predicted label of node u is determined by the largest $p(C_j|u)$.

b: WEIGHTED-VOTE RELATIONAL NEIGHBOR (wvRN) [8]

wvRN is a recommended baseline method for comparison as it has shown a surprisingly good performance in many real world datasets [7]. Given an unknown node u, wvRN calculates the probability of each class c for node u as:

$$P(C(u) = c|N_u) = \frac{1}{Z}\sum_{j\in N_u} w_{u,j} \cdot P(C(j) = c|N_j), \tag{11}$$

where $C(u)$ represents the class of node u, N_u is the set of nodes that are linked to node u, $w_{u,j}$ is the weight of the edge between node u and node j, and Z is a normalization factor.

c: SPECTRALCLUSTERING [40]

Spectralclustering is a representative method for handling network with heterophily. In SocioDim framework, spectral clustering is used to extract latent social dimensions based on the network structure. Then, by using social dimensions as new features, it applies SVM to classify unknown nodes.

d: EDGECLUSTERING [42]

Edgeclustering also applies the SocioDim classification framework, but it uses an edge-centric clustering scheme to extract sparse social dimensions. Each edge is treated as one data instance, and the connected nodes are corresponding features. Then, the proposed k-means clustering algorithm can be applied to partition edges into disjoint sets, with each set representing one possible affiliation. Lastly, SVM is applied to classify the unknown nodes.

C. EXPERIMENT DESIGN

1) PERFORMANCE MEASUREMENT

In the experiment, we use accuracy as an evaluation measurement to compare the performance of different methods. The dataset is divided into training set and testing set, where nodes in the training set are labeled and nodes in the testing set are unknown. Then we use the labeled nodes in the training set to predict labels of nodes in the testing set.

For BCC, SC_CN and wvRN, there may be more than one optimal result in the classification process. When this phenomenon happens, we choose the label according to the prior, i.e., the majority class in the training set.

2) TRAINING DATA AND TEST DATA

The ten-folder cross-validation method is used to partition the training data and test data. We first generate a random order of the instances in the data set, and then divide it into ten parts $\{d_1, d_2, \ldots, d_{10}\}$ equally. Given the proportion of labeled nodes: p, for each parts d_i, we use $\left\{d_i, \ldots, d_{1+(i-2+p\times 10) \bmod 10}\right\}$ as training data, and the rest

$(1-p) \times 10$ parts as test data to calculate the accuracy A_i. The average value of $\{A_1, \ldots, A_{10}\}$ is treated as the accuracy of a ten-folder cross-validation.

3) HANDLING UNDIRECTED GRAPH

Based on the assumption of the generative process, BCC can extract behavior feature from directed weighted graph; when the network is undirected, the extension of BCC method is straightforward: given an undirected graph G, we assume that undirected edge $e(i,j)$ can be viewed as two directed edges, respectively $e^{'}(i,j)$ and $e^{'}(j,i)$, with the same weight of $e(i,j)$, so we have $w^{'}(i,j) = w^{'}(j,i) = w(i,j)$.

As we will see in the experiments, for undirected graphs, BCC can still get satisfied classification result.

4) PARAMETERS OF BCC

There are three parameters in the BCC method, which are as follows:

a: PEARSON CORRELATION COEFFICIENT THRESHOLD

P, which is used to screen valuable nodes from the perspective of correlation of the behavior feature. Given unknown node u, the labeled nodes of higher correlation with u $(corr(v,u) > P)$ will be considered to be valuable nodes.

b: NUMBER OF MOST SIMILAR NODES

K, which is used to screen more valuable nodes from the perspective of similarity of the behavior feature. Given unknown node u and V_u, the top-K similar nodes in V_u with will be considered as more valuable nodes. As we have mentioned in the previous sections, the similarity analysis is optional when handing undirected networks. However, to maintain consistency, we include this component in all the following experiments.

c: HYPERPARAMETER

H, which is the parameter of Dirichlet prior distribution. H is used to integrate prior knowledge and avoid over-fitting. For simplicity, we choose symmetric dirichlet with a scalar parameter H. A symmetric dirichlet is a dirichlet distribution where all of the elements of the parameter vector have the same value.

In the next section, we will evaluate the sensitivity of our method to these parameters and discuss how to select appropriate parameters in practice.

VI. RESULTS AND DISCUSSION

In this section, we first give an example to illustrate the characteristics and advantages of BCC method in imbalanced classification, and then compare the classification performance on several public datasets. Finally, we analyze the sensitivity of different parameters.

A. CASE STUDY FOR IMBALANCED CLASSIFICATION

In Fig. 1, we have shown that behavior feature can handle sparse labeling problem with more discriminative ability. In this section, we use a case to indicate the advantage by screening valuable nodes. When the network is imbalanced, i.e., the majority class occupies much more nodes than the other class, many traditional methods may fail. Take the network in Fig. 3 for example, we need to predict the label of node e (the true label is ''red'').

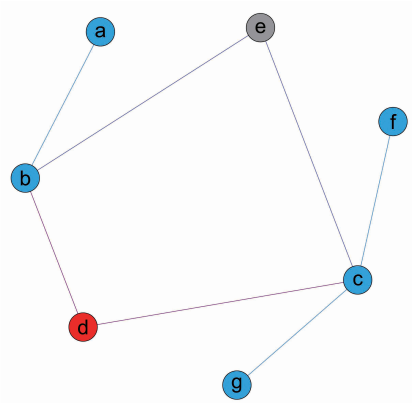


FIGURE 3. **A small network illustrating the fail of baseline methods when handling imbalanced classification. The red and blue colors represent the labels of nodes, and nodes with gray color are unknown nodes.**

According to (10), we can get $p(\text{''red''}) = 2/5$ and $p(\text{''blue''}) = 3/5$, so node e would be labeled as ''blue'' by SC_CN. wvRN relies on the local neighborhood, i.e., the labels of node b and node c, so it also tends to classify node e as ''blue''. For spectralclustering and edgeclustering, since the instances in the training set are imbalanced, they will tend to classify node e as ''blue'' as well.

BCC can handle this problem effectively. In the network, node e only connects with node b and node c, so does node d. Therefore, the behavior features of node e and node d are the same, which means that node d is the most similar node with node e. In this situation, using a higher value of P or a smaller value of K to reduce the impact of noise data, BCC will predict the label of e as ''red'' correctly.

B. CLASSIFICATION RESULTS ON EMPIRICAL DATA

Here we present the experimental analysis of our method on the four data sets introduced in the previous section. As there are three types of data sets: a directed weighted communication network (*Enron*), a network of webpages with hyperlinks (*WEBKB*), and citation networks of academic papers (*Cora* and *Citeseer*), we choose different parameters for BCC accordingly.

For *Enron* data, since the average weight of edges is high (22.6), in order to highlight the importance of observed data, we choose a relatively small hyperparameter value, H = 1. And then, we choose a smaller P and a larger K (P = 0, K = 10) to reduce the impact of outlier data and to allow more nodes to be included in the classification process. For *WEBKB*, *Cora* and *Citeseer*, which are unweighted networks, the weight of edges can be regarded as 1, so we choose a small hyperparameter, H = 0.05, to highlight the importance of observed data. There are fewer edges in these networks, so the behavior feature extracted may not reflect node's essential attribute accurately. In order to handle this problem, nodes in these networks need to be screened by more strict criteria. Thereby, we choose a larger P and a smaller K (P = 0.5, K = 5 for *WEBKB*, and P = 0.2, K = 50 for *Cora* and *Citeseer*).

The spectralclustering and edgeclustering methods require a latent social dimension parameter, d. To implement the algorithm, we follow the study of Tang and Liu on preferred dimensionality [40] and let $d = 500$ for *Cora* and *Citeseer*. For *WEBKB* dataset, the optimal value of d is found after a number of cross-validation tests ($d = 50$ in spectralclustering and $d = 5$ in edgeclustering).

Under the above setting, the BCC method is then compared with baseline methods for classification on the four datasets. Since SC_CN, spectralclustering and edgeclustering are made for undirected unweighted networks, we only compare BCC with wvRN method on *Enron* data set. And for *WEBKB*, *Cora* and *Citeseer* dataset, we transform them to undirected networks by considering the edges to be undirected and retaining the weight. Here we did not convert the *Enron* dataset to undirected unweighted network, since we aim to verify the classification ability of BCC on directed weighted networks.

We gradually increase the proportion of labeled nodes from 10% to 90%. For each setting, we run 10 times ten-folder cross-validation, and the average accuracy is recorded. The performances of different methods are plotted in Fig. 4-6.

We can see that, on *Enron* dataset (Fig. 4), BCC has better performance than wvRN when nodes are labeled sparsely ($< 50\%$), for example, when only 20% of nodes are labeled, the classification accuracy for BCC is 3% higher than wvRN. However, the accuracy for wvRN improves steadily and exceeds BCC when the proportion of labeled nodes increases.

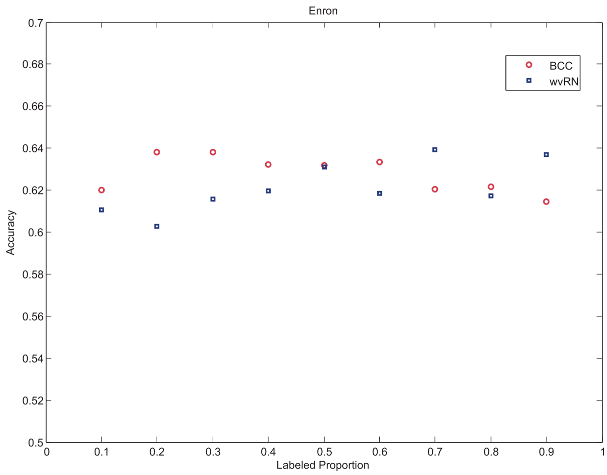


FIGURE 4. **Classification result on Enron dataset.**

Experiments on *Cora* and *Citeseer* (Fig. 5 a-b) yield similar results: wvRN performs better at higher labeled proportions, but with labeled proportion decreases, wvRN cannot avoid the impact of sparse labeling, the classification accuracy decreases significantly. Spectralclustering and edgeclustering encounter the same problem, the accuracy drops rapidly as the labeled proportion decreases. For BCC, on contrary, the accuracy remains satisfactory even when the labeled nodes decrease to less than 20%.

In all four data sets in *WEBKB* (Fig. 6 a-d), BCC produces competitive (yet the highest) accuracy with SC_CN in the *cornell* and *washington* data, and the highest accuracy in the *texas* and *wisconsin* data, while SC_CN in the most worse case, is 4% less accurate than BCC. wvRN performs extremely poorly, the accuracy of BCC is higher than wvRN from a low of 8%, to a high of 40%. The performances of edgeclustering in *washington* is competitive, but beyond that spectralclustering and edgeclustering produce

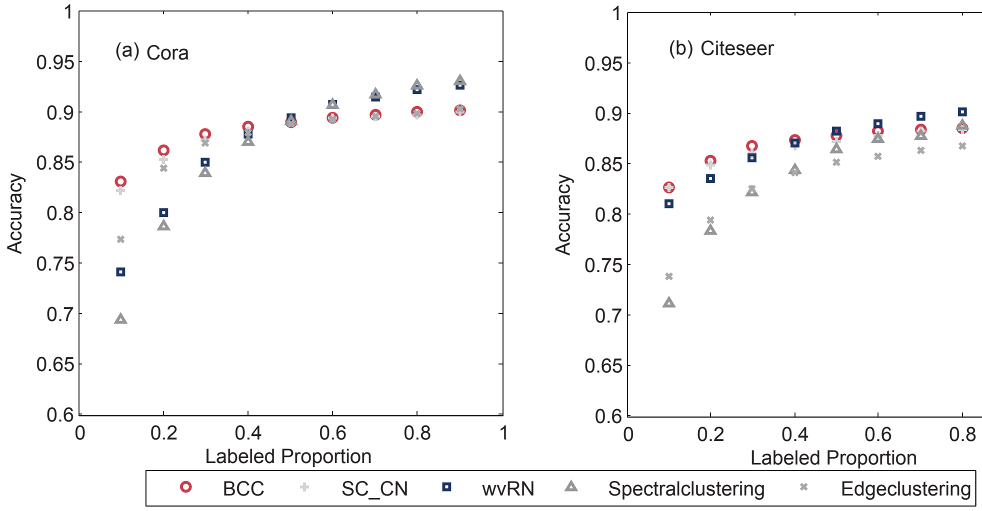


FIGURE 5. **Classification result on Cora and Citeseer dataset.**

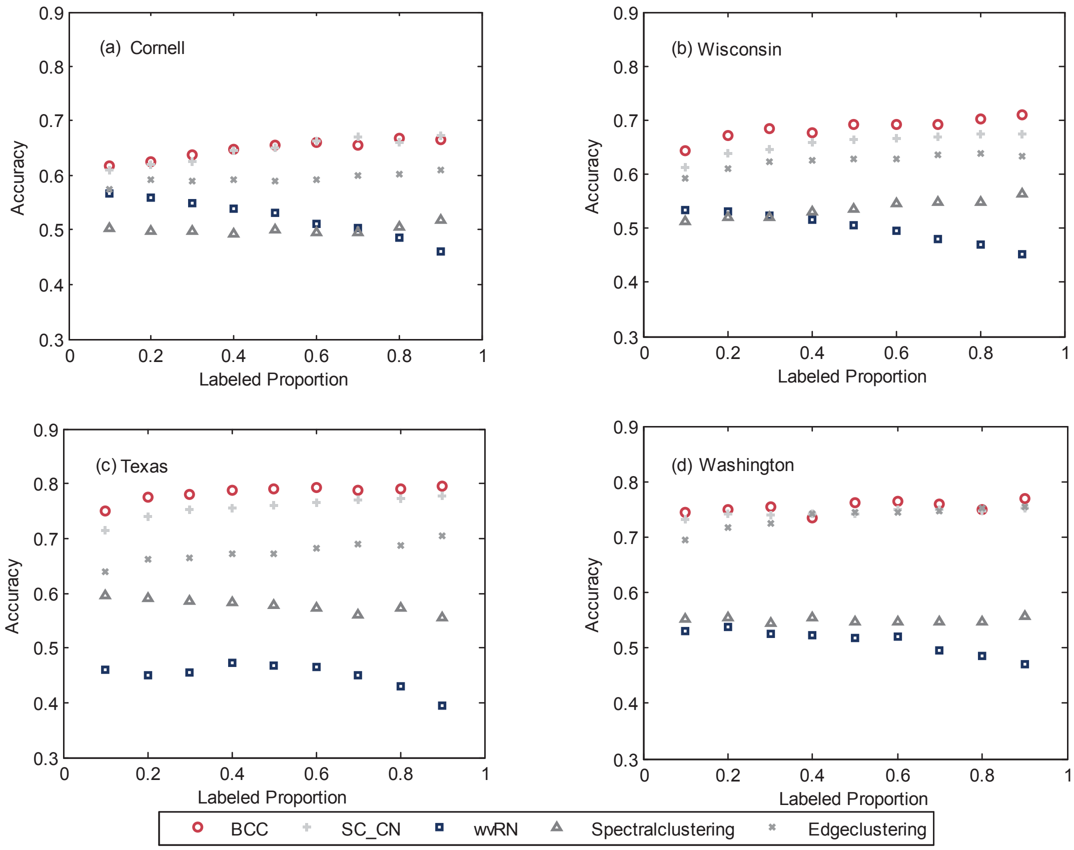


FIGURE 6. Classification result on WebKB dataset.

unsatisfied result. Compared with BCC, edgeclustering is 5% lower and spectralclustering is 10% lower in *cornell*, *texas* and *wisconsin* data.

Two important conclusions can be summarized from the above analysis:

Firstly, BCC method performs better in sparsely labeled networks. As it can be seen, BCC produces competitive performance in all the above real-world datasets, and it achieves the highest accuracy when labeled nodes become sparse (less than 50%). The reason is that wvRN predicts the unknown nodes based on local neighbors, so as the labeled proportion decreases, the performance will be largely degraded due to the lack of sufficient labeled neighbors. Likewise, when the labeled proportion is low, spectralclustering and edgeclustering will also be affected since there will be fewer training instances. SC_CN uses all labeled nodes for classification, so the performance drops when labeled proportion decreases. BCC, on the other hand, does not rely on local neighboring nodes. It tries to mine latent links between all nodes and only requires a few valuable nodes to make prediction, so it is less affected when the labeled neighbors decrease. In addition, BCC method uses collective classification to accumulate experience constantly in the classification process, making the sparse labeling problem gradually ease and yielding better result.

Secondly, BCC can obtain satisfied classification results when homophily is low in the network. As pointed in [8], we can find that the homophily is much lower in *WEBKB* dataset. In all four networks, BCC handles classification task effectively and produces the highest accuracy. In contrast, wvRN relies on the homophily assumption to make prediction, so it performs extremely poorly in *WEBKB* dataset. SC_CN, spectralclustering and edgeclustering can overcome heterophily problem to some extent because they do not rely on direct neighbors. However, the scale of networks in *WEBKB* are relatively small, which means that there will be only a few labeled nodes in the training set, so methods with a learning process (spectralclustering and edgeclustering) will be affected and perform poorly in this situation. SC_CN does not have a learning process, but in small networks, labeled nodes may be imbalanced, so SC_CN, which relies on all the labeled nodes for classification, will be affected and tend to predict unlabeled nodes as majority. In contrast, by using the top-K valuable nodes, BCC handles the problem effectively and obtains satisfactory performance when homophily is low in the network.

C. PARAMETER SENSITIVITY

In this section, we are going to discuss the impact of parameters for BCC. Three parameters, correlation coefficient

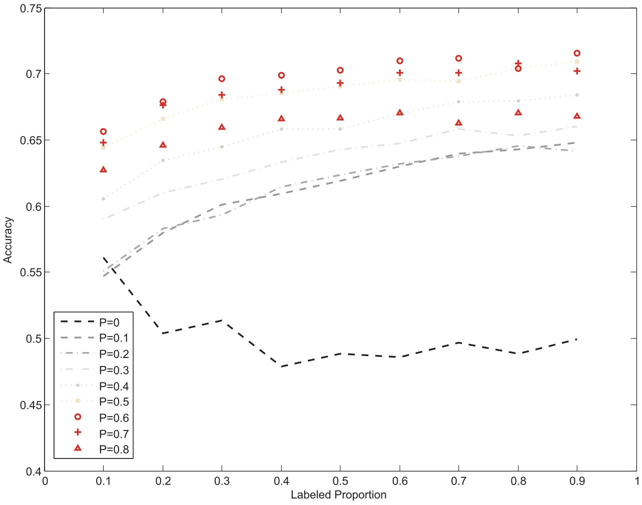


FIGURE 7. Sensitivity to Pearson correlation coefficient threshold value, P.

threshold, number of most similar nodes, and hyperparameter are tested and analyzed. *wisconsin* in *WEBKB* dataset, which is a medium size network from above datasets, is used for the following experiments.

1) PEARSON CORRELATION COEFFICIENT THRESHOLD

In order to evaluate the impact of P on classification results, we choose the *wisconsin* dataset in *WEBKB* and fix $K = 5$, $H = 0.05$. Then we vary the value of P (from 0 to 0.8) in the experiment. Results are shown in Fig. 7.

When the value of P increases from 0 to 0.6, nodes with high correlation are extracted and included in the classification process, and the classification accuracy improves constantly from about 50% to 70%. This suggests that using correlation of behavior feature does improve the classification performance. However, when P continues to increase (higher than 0.6), we find that the classification accuracy starts to decrease. This is because that extreme high correlation filters out a lot of valuable nodes, and there will be only a few nodes involved in the classification. In such situations, results will be strongly affected by outliers and begin to decline.

Therefore, in the implementation of BCC, we need to choose the value of P carefully, such that, on the one hand, to ensure nodes with high correlation to be involved in the classification process; and on the other hand, to make sure that valuable nodes won't be filtered out by excessive high threshold value. In practice, the optimal P depends on the attribute of different data sets and can be determined by cross validation.

2) NUMBER OF MOST SIMILAR NODES

In order to verify the impact of K, we choose the *wisconsin* dataset in *WEBKB* and fix $H = 0.05$. In BCC method, P and K are two parameters used to screen valuable nodes, so we want to find out how they affect each other in the classification process. We select three different values of P ($P = 0.5$, $P = 0.1$, $P = 0$), and vary the value of K in experiments. Results are shown in Fig. 8.

When we set a larger value of P ($P = 0.1$ and $P = 0.5$), the classification accuracy are steady as K varies. The reason is that when P is large, there will be only a few nodes left for K to screen, which will make K have minor impact on the result. In contrast, when P is small ($P = 0$), many unrelated nodes may be involved in the classification process. In this situation, setting different K will affect the accuracy. As shown in Fig. 8-c, K has minor impact on the accuracy when labeled proportion is low, this is because *wisconsin* is a small dataset and there will be only a few labeled nodes left after the correlation analysis. However, when the labeled proportion increases (higher than 0.6), a small K may involve some outliers in the classification process, leading to a lower classification accuracy. If we choose a larger K, the impact of outliers will be relatively small, and the method can achieve higher classification accuracy.

In BCC, correlation analysis plays a role before similarity analysis, therefore, we recommend selecting an appropriate P at first, on this basis, a slightly larger K may achieve better performance. However, K should not be larger than 10% of the total number of nodes; otherwise it will include too much noise data and may affect classification accuracy. It should be noted that similarity analysis is introduced for handling weighted network, so in unweighted network, this step can be omitted or replaced by other measurements, which reveals the flexibility of our method.

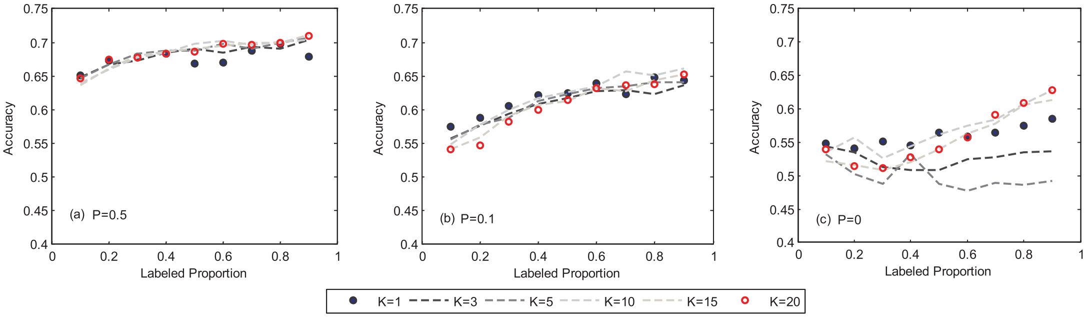


FIGURE 8. Sensitivity to number of most similar nodes, K.

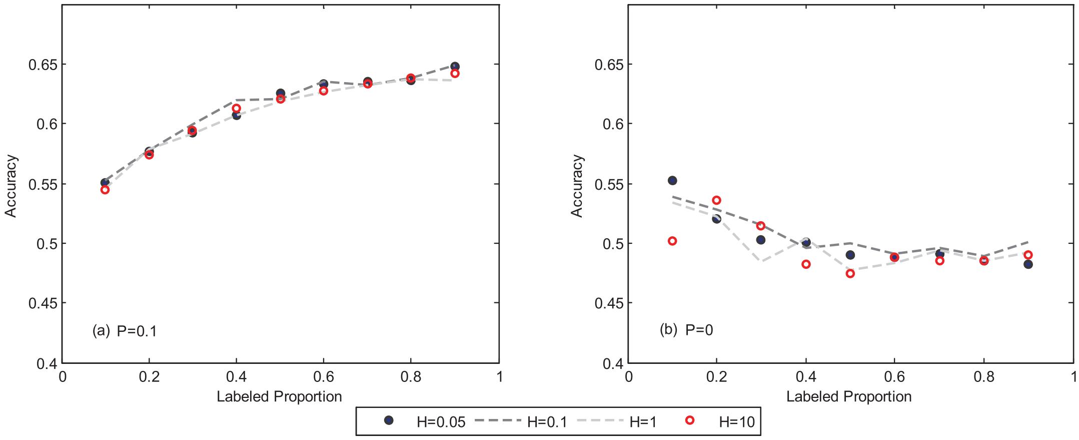


FIGURE 9. Sensitivity to hyperparameter, H.

3) HYPERPARAMETER

Using prior distribution is a basic Bayesian approach to integrate prior knowledge and avoid over-fitting. In order to evaluate the impact of H, we choose the *wisconsin* dataset in *WEBKB* and fix K = 5. In the above discussion, we already see that the setting of P will affect classification results, therefore, in order to verify the impact of H in different cases, we choose two different values of P (P = 0.1, P = 0), and vary the value of H in the experiment. Results are shown in Fig. 9.

We can see that the value of H has minor impact on classification accuracy for different value of P, while smaller H achieves slightly better results. This is because in unweighted networks (*wisconsin*), the weight of each edge can be regarded as 1, a small H will be able to highlight the importance of observed data.

Therefore, in the experiment, we need to choose appropriate value of H according to the observed data. In order to highlight the role of observed data, it will be more appropriate to select 0.01-0.1 in unweighted networks. While in weighted networks, setting H to be 1/100-1/10 of average weight of edges will be more appropriate.

VII. CONCLUSION

In order to improve classification accuracy in sparsely labeled networks, we propose a novel behavior based collective classification method, BCC, in this study. In BCC, the behavior feature of nodes is extracted for classification, which has shown more discriminative ability to traditional methods. Then, instead of using all the labeled nodes, we screen the most-relevant nodes according to the calculation of correlation and similarity, which can overcome the effects of noise and imbalanced dataset. Finally, collective inference is introduced to utilize both labeled nodes and unlabeled nodes, which can relieve the sparse labeling problem effectively.

Extensive experiments on public data set demonstrate that BCC method outperforms several baseline methods, especially when the network is sparsely labeled. Meanwhile, instead of relying on local neighbor nodes, BCC method predicts unknown nodes by using valuable nodes which may not even connected directly, making it a preferable method for classification in networks with heterophily. Note that in *Enron* dataset, only a subset of nodes have labels and we can only compare different methods on these nodes, but unlabeled nodes and their connections to labeled nodes may still provide useful behavior information, which can be utilized in BCC method. From this point of view, BCC shares the similar idea with semi-supervised learning.

The current implementation of BCC has limited computing efficiency for similarity comparison, when the network is large, it may become a bottleneck for the algorithm. Future work may also model the network with different generation process, and other types of behavior feature and strategies in the classification process may be applied. Another challenging extension is the multi-label classification in sparsely labeled networks, where instances can be assigned with multiple labels and the labeled nodes are few in the network. We believe this study highlights the importance of behavior feature in improving performance of network classification and the BCC method could be used in a variety of settings with generalized stability.

REFERENCES

[1] S. A. Macskassy and F. Provost, "A brief survey of machine learning methods for classification in networked data and an application to suspicion scoring," in *Statistical Network Analysis: Models, Issues, and New Directions*. Berlin, Germany: Springer, 2007, pp. 172–175.

[2] S. A. Macskassy and F. Provost, "Suspicion scoring based on guilt-by-association, collective inference, and focused data access," in *Proc. Int. Conf. Intell. Anal.*, 2005.

[3] S. Hill, D. K. Agarwal, R. Bell, and C. Volinsky, "Building an effective representation for dynamic networks," *J. Comput. Graph. Statist.*, vol. 15, no. 3, pp. 584–608, 2006.

[4] J. Neville, Ö. Şimşek, D. Jensen, J. Komoroske, K. Palmer, and H. Goldberg, "Using relational knowledge discovery to prevent securities fraud," in *Proc. 11th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2005, pp. 449–458.

[5] P. Domingos and M. Richardson, "Mining the network value of customers," in *Proc. 7th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2001, pp. 57–66.

[6] Z. Huang, H. Chen, and D. Zeng, "Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 116–142, 2004.

[7] S. A. Macskassy and F. Provost, "Classification in networked data: A toolkit and a univariate case study," *J. Mach. Learn. Res.*, vol. 8, pp. 935–983, May 2007.

[8] S. A. Macskassy and F. Provost, "A simple relational classifier," in *Proc. 2nd Workshop Multi-Relational Data Mining (MRDM)* , 2003, pp. 1–13.

[9] B. Taskar, E. Segal, and D. Koller, "Probabilistic classification and clustering in relational data," in *Proc. Int. Joint Conf. Artif. Intell.*, vol. 17. 2001, pp. 870–878.

[10] B. Taskar, P. Abbeel, and D. Koller, "Discriminative probabilistic models for relational data," in *Proc. 18th Conf. Uncertainty Artif. Intell.*, 2002, pp. 485–492.

[11] J. Neville and D. Jensen, "Collective classification with relational dependency networks," in *Proc. 2nd Int. Workshop Multi-Relational Data Mining*, 2003, pp. 77–91.

[12] F. Lin and W. W. Cohen, "Semi-supervised classification of network data using very few labels," in *Proc. Int. Conf. Adv. Social Netw. Anal. Mining (ASONAM)*, Aug. 2010, pp. 192–199.

[13] B. Gallagher, H. Tong, T. Eliassi-Rad, and C. Faloutsos, "Using ghost edges for classification in sparsely labeled networks," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2008, pp. 256–264.

[14] R. Xiang and J. Neville, "Pseudolikelihood em for within-network relational learning," in *Proc. 8th IEEE Int. Conf. Data Mining (ICDM)*, Dec. 2008, pp. 1103–1108.

[15] J. J. Pfeiffer, J. Neville, and P. N. Bennett, "Overcoming relational learning biases to accurately predict preferences in large scale networks," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 853–863.

[16] L. K. Mcdowell, "Relational active learning for link-based classification," in *Proc. IEEE Int. Conf. Data Sci. Adv. Anal.*, Oct. 2015, pp. 1–10.

[17] A. Kuwadekar and J. Neville, "Relational active learning for joint collective classification models," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Bellevue, WA, USA, Jun./Jul. 2012, pp. 385–392.

[18] M. Bilgic and L. Getoor, "Link-based active learning," in *Proc. NIPS Workshop Anal. Netw. Learn. Graph.*, 2009, pp. 1–7.

[19] M. Bilgic, L. Mihalkova, and L. Getoor, "Active learning for networked data," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, 2010, pp. 79–86.

[20] S. A. Macskassy, "Improving learning in networked data by combining explicit and mined links," in *Proc. Nat. Conf. Artif. Intell.*, vol. 22. Menlo Park, CA, USA, 2007, p. 590.

[21] Q.-M. Zhang, M.-S. Shang, and L. Lü, "Similarity-based classification in partially labeled networks," *Int. J. Modern Phys. C*, vol. 21, no. 6, pp. 813–824, 2010.

[22] X. Zhu, "Semi-supervised learning literature survey," Dept. Comput. Sci., Univ. Wisconsin-Madison, Madison, WI, USA, Tech. Rep. 1530, 2005.

[23] X. Zhu and A. B. Goldberg, "Introduction to semi-supervised learning," *Synth. Lect. Artif. Intell. Mach. Learn.*, vol. 3, no. 1, pp. 1–130, 2009.

[24] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Proc. Adv. Neural Inf. Process. Syst.*, 2004, vol. 16. no. 16, pp. 321–328.

[25] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. 12th ICML*, vol. 3. 2003, pp. 912–919.

[26] J. Kleinberg and E. Tardos, "Approximation algorithms for classification problems with pairwise relationships: Metric labeling and Markov random fields," *J. ACM*, vol. 49, no. 5, pp. 616–639, 2002.

[27] A. Blum, "Learning from labeled and unlabeled data using graph mincuts," in *Proc. 18th Int. Conf. Mach. Learn.*, 2001, p. 19.

[28] A. Blum, J. Lafferty, M. R. Rwebangira, and R. Reddy, "Semi-supervised learning using randomized mincuts," in *Proc. 21st Int. Conf. Mach. Learn.*, 2004, p. 13.

[29] J. Callut, K. Françoisse, M. Saerens, and P. Dupont, *Semi-Supervised Classification from Discriminative Random Walks*. Berlin, Germany: Springer, 2008, pp. 162–177.

[30] M. E. J. Newman, "A measure of betweenness centrality based on random walks," *Soc. Netw.*, vol. 27, no. 1, pp. 39–54, 2005.

[31] D. Zhou and B. Schölkopf, *Learning from Labeled and Unlabeled Data Using Random Walks*. Berlin, Germany: Springer, 2004, pp. 237–244.

[32] H. Tong, C. Faloutsos, and J.-Y. Pan, "Fast random walk with restart and its applications," in *Proc. 6th Int. Conf. Data Mining (ICDM)*, Dec. 2006, pp. 613–622.

[33] A. Mantrach, N. van Zeebroeck, P. Francq, M. Shimbo, H. Bersini, and M. Saerens, "Semi-supervised classification and betweenness computation on large, sparse, directed graphs," *Pattern Recognit.*, vol. 44, no. 6, pp. 1212–1224, 2011.

[34] Y. Fu, X. Zhu, and B. Li, "A survey on instance selection for active learning," *Knowl. Inf. Syst.*, vol. 35, no. 2, pp. 249–283, 2013.

[35] D. D. Lewis and J. Catlett, "Heterogenous uncertainty sampling for supervised learning," in *Proc. ICML*, vol. 94. 1994, pp. 148–156.

[36] N. Roy and A. McCallum, "Toward optimal active learning through Monte Carlo estimation of error reduction," in *Proc. ICML*, Williamstown, MA, USA, 2001, pp. 441–448.

[37] S. A. Macskassy, "Using graph-based metrics with empirical risk minimization to speed up active learning on networked data," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 597–606.

[38] T. Joachims, "Transductive learning via spectral graph partitioning," in *Proc. 20th Int. Conf. Mach. Learn. (ICML)*, 2003, pp. 290–297.

[39] F. Wang and C. Zhang, "Label propagation through linear neighborhoods," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 1, pp. 55–67, Jan. 2008.

[40] L. Tang and H. Liu, "Leveraging social media networks for classification," *Data Mining Knowl. Discovery*, vol. 23, no. 3, pp. 447–478, 2011.

[41] L. Tang and H. Liu, "Relational learning via latent social dimensions," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 817–826.

[42] L. Tang and H. Liu, "Scalable learning of collective behavior based on sparse social dimensions," in *Proc. 18th ACM Conf. Inf. Knowl. Manage.*, 2009, pp. 1107–1116.

[43] X. Wang and G. Sukthankar, "Multi-label relational neighbor classification using social context features," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2013, pp. 464–472.

[44] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 1951.

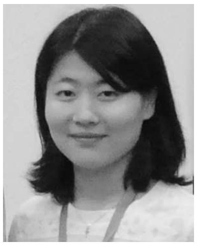
[45] Q. Lu and L. Getoor, "Link-based classification," in *Proc. 20th Int. Conf. Mach. Learn. (ICML)*, 2003, pp. 496–503.

[46] *Enron Emails Dataset*, Aug. 5, 2016. [Online]. Available: https://www.cs.purdue.edu/homes/jpfeiff/enron.html

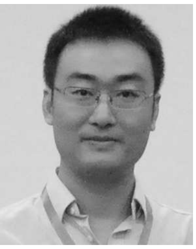
[47] *Role List for Enron Emails Dataset*, Aug. 5, 2016. [Online]. Available: http://www.ahschulz.de/enron-email-data/

[48] *WebKB, Cora, and Citeseer Datasets*, Aug. 5, 2016. [Online]. Available: http://linqs.cs.umd.edu/projects/projects/lbc/index.html

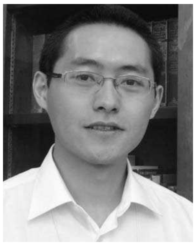
[49] L. Lü and T. Zhou, "Link prediction in complex networks: A survey," *Phys. A, Statist. Mech. Appl.*, vol. 390, no. 6, pp. 1150–1170, 2011.



JUNYI XU is currently pursuing the Ph.D. degree with the National University of Defense Technology, Changsha, China. Her research interests include machine learning, agents and data mining interaction, and argumentation technology on multi-agent system.



LE LI is currently pursuing the Ph.D. degree with the National University of Defense Technology, Changsha, China. His research interests are in the areas of distributed data mining, complex networks, and machine learning algorithms on graph.



XIN LU received the Ph.D. degree from the Department of Public Health Sciences, Karolinska Institutet. He was with the Department of Sociology, Stockholm University, from 2009 to 2012, and the Institute for Future Studies, Stockholm, in 2013. He is currently an Associate Professor with the College of Information System and Management, National University of Defense Technology, Changsha, China. His research is distributed in mobile phone data mining, operational research, and graph algorithms.

SHENGZE HU received the Ph.D. degree in management science and engineering from the National University of Defense Technology, Changsha, China. He is currently an Associate Professor with the College of Information System and Management, National University of Defense Technology. His research interests include big data and text mining.

BIN GE received the Ph.D. degree in management science and engineering from the National University of Defense Technology, Changsha, China. He is currently an Associate Professor with the College of Information System and Management, National University of Defense Technology. His research interest covers text analysis and social computing.

WEIDONG XIAO received the Ph.D. degree in management science and engineering from National University of Defense Technology, Changsha, China. He is currently a Professor with the College of Information System and Management, National University of Defense Technology. His research interests include data mining and social network.

LI YAO received the Ph.D. degree in computer science from the National University of Defense Technology, Changsha, China. She is currently a Professor with the College of Information System and Management, National University of Defense Technology. Her research interests include artificial intelligence and multi-agent system.