Signal Processing 124 (2016) 93–102

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Supervised sampling for networked data

Meng Fang [a,b,*], Jie Yin [b], Xingquan Zhu [c]

[a] University of Technology, Sydney, NSW, Australia
[b] CSIRO, Sydney, NSW, Australia
[c] Florida Atlantic University, FL, USA

ARTICLE INFO

Article history:
Received 7 May 2015
Received in revised form
29 September 2015
Accepted 29 September 2015
Available online 17 October 2015

Keywords:
Supervised sampling
Random walks
Information network

ABSTRACT

Traditional graph sampling methods reduce the size of a large network via uniform sampling of nodes from the original network. The sampled network can be used to estimate the topological properties of the original network. However, in some application domains (e.g., disease surveillance), the goal of sampling is also to help identify a specified category of nodes (e.g., affected individuals) in a large network. This work therefore aims to, given a large information network, sample a subgraph under a specific goal of acquiring as many nodes with a particular category as possible. We refer to this problem as *supervised sampling*, where we sample a large network for a specific category of nodes. To this end, we model a network as a Markov chain and derive supervised random walks to learn stationary distributions of the sampled network. The learned stationary distribution can help identify the best node to be sampled in the next iteration. The iterative sampling process ensures that with new sampled nodes being acquired, supervised sampling can be strengthened in turn. Experiments on synthetic as well as real-world networks show that our supervised sampling algorithm outperforms existing methods in obtaining target nodes in the sampled networks.

1. Introduction

Many research works have been carried out on networked data to study the problem of node classification at various levels, including the Web, citation networks, and online social networks. The large size of these networks and other restrictions, such as privacy, make learning from the entire network become extremely computational expensive or even impossible. For example, discovering a specific community in the DBLP citation network would require searching all the HTML pages and downloading terabyte-level data, which is most likely impractical. Therefore, research studies have attempted to address the problem of acquiring a smaller, but representative, subset of samples from a large graph [1,2] and then proceed with subsequent network mining tasks.

Currently, most graph sampling algorithms have been mainly focused on generating a uniform sample of nodes and edges at random from the original graph. Assuming that the node and edge information is readily observable, they usually operate on an entirely, static graph. These methods are characterized by the order in which the nodes are visited (or traversed), for example, Bread-First Search (DFS), Depth-First Search (DFS), forest fire, and snowball sampling. They typically start at a seed node, and recursively visit (one, some or all) its neighbors. These methods are varied and distinct with each other because of different ordering strategies of visiting the nodes. Although some research works have shown that these methods are biased

* Corresponding author.
E-mail addresses: Meng.Fang@student.uts.edu.au, Meng.Fang@csiro.au (M. Fang), Jie.Yin@csiro.au (J. Yin), xzhu3@fau.edu (X. Zhu).

http://dx.doi.org/10.1016/j.sigpro.2015.09.040
0165-1684/

towards high-degree nodes [3], they are still found to be very popular and widely used for sampling nodes in real-world large networks.

In reality, however, real-world networks may not be immediately accessible until each node and its connections are progressively crawled. For example, in a citation network, papers need to be read or preprocessed so as to find their citations, as well as categories, general terms, keywords, and authors. Thus, collecting a paper's detailed information or identifying a paper's research topic incurs a cost. It would be desirable to minimize the cost by collecting a small portion of the network instead of the entire network. Similar issues may also occur in large online social networks such as Facebook or Twitter, where one may be interested in identifying a specific group of users with certain professions or hobbies. In addition, real-world networks often have imbalanced node distributions where the majority of nodes belong to one class and very few nodes belong to the minority class. As a result, uniform sampling may fail to include the nodes belonging to the minority class because these nodes often have low degree and few connections. For example, in disease surveillance, there may exist very few affected individuals in a large population network. Due to the fact that the nodes' attribute information is not considered, as well as their bias towards high-degree nodes, traditional sampling methods are not effective for sampling nodes of minority category in large networks.

Motivated by the above observations, in this paper, we propose a new strategy for obtaining a biased sample of nodes by carrying out network sampling under supervision. We refer to this class of problems as *supervised sampling*, where we aim to identify nodes belonging to a specific category (i.e., positive instances) that may comprise only a small portion of the overall network. We provide practical implementations of supervised sampling, where given a large graph and a specific category, the goal is to iteratively sample a subgraph from the original graph under the requirements related to the category. To tackle this problem, we model a graph as a Markov chain, where nodes are considered as interior states and edges are chains between states. We design a supervised random walk to compute the stationary distributions of nodes, which indicate the probability of nodes being positive, by using nodes' attribute information. Unlike uniform sampling, we iteratively choose the best nodes to be sampled in the next iteration based on their probabilities of being positive. At each iteration, the sampling process is guided by a supervised random walk that is more likely to visit positive nodes in the neighborhood. Once a node is visited, the sampled network is expanded to include the node itself, its neighboring nodes, as well as new edges between them. After a node is sampled, the genuine label of the node is also revealed. All such information can be used to update the stationary distribution of the sampled network, which will strengthen supervised sampling at the next iteration.

The main contribution of this work is twofold: first, we introduce a new supervised sampling problem on large networks; second, we present a novel unified framework to perform supervised sampling for a given task through formulating a supervised random walk as an optimization problem. Experiments on synthetic and real-world networks show that our proposed algorithm achieves a higher recall of positive nodes while sampling large networks than baseline methods, especially for networks having imbalanced class distributions.

2. Related work

In recent years, there have been many research efforts on studying information networks, such as node classification [4], link prediction [5,6], active learning [7], transfer learning [8,9], personalized recommendation [10,11], and so on. These studies are different from traditional instance-based learning problems because both instance content and network structure information are available for learning. Sen et al. [4] introduced a classification framework for networked data as collective classification. Collective classification is a combined classification of a set of interlinked objects using correlations between node labels and node content (i.e., attributes), and information of each node's neighborhood. Even when the instances are not explicitly linked to form a network, the use of the correlations between instances is also beneficial for improving the classification performance (e.g., [12]). Link prediction is also a fundamental problem in the network settings [5], which aims to predict the presence of links between network nodes. Backstrom and Leskovec [5] proposed to combine network structure information with rich node and edge attributes. Ye et al. [6] adopted Non-negative Matrix Tri-Factorization (NMTF) to learn latent topological features from network structure, and use them to enhance nodes' features. Bilgic et al. [7] proposed an active learning algorithm for node classification based on collective classification.

Sampling techniques have also been extensively studied on very large scale information networks. Traditional graph sampling techniques can be roughly classified into two categories: *graph traversals* and *random walks* [3]. For graph traversals, nodes are sampled without replacement; once a node is visited, it is never revisited again. Depending on the order in which nodes are visited, these methods include Breadth-First Search (BFS), Depth-First Search (DFS), forest fire, and snowball sampling [13–15]. Among others, BFS has been popularly used for sampling social networks, which has been studied extensively [14–18]. However, existing research has shown that BFS is biased towards high-degree nodes in real-world networks [19,20]. When using graph traversals for sampling, the sampling process terminates after a fraction of graph nodes are collected.

Random walks fall into the other category of sampling techniques, which usually start at any specific node and initiate a random walk by proceeding to the next node selected at random from the neighbors of the current node. It is found that random walks are biased towards high degree nodes in the graph [21]. Some methods have been proposed to correct the bias of random walks. For example, Gjoka et al. [3] proposed a Metropolis-Hastings algorithm to collect an unbiased sample of *Facebook* users.

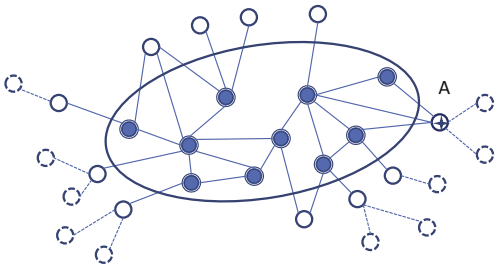

Fig. 1. A partially observed subgraph $G_t = (\mathcal{V}_t, \mathcal{E}_t)$. Intra-acquired nodes are denoted by double solid circles and nodes directly connected to Intra-acquired nodes are Border-acquired nodes, from which star node A is selected to be sampled next because it has maximum probability (according to our formulation) belonging to positive nodes.

Likewise, Hübler et al. [22] presented a Metropolis algorithm for sampling a representative subgraph, requiring that sampled graph preserves crucial graph properties of the original graph.

Graph sampling techniques provide an efficient, yet inexpensive solution for social network analysis. Leskovec and Faloutsos [23] examined different sampling methods on social networks and found that best performing methods are random walks and forest fires. Papagelis et al. [24] introduced sampling-based algorithms that given a user in a social network efficiently obtain a near-uniform random sample of nodes in its neighborbood. Maiya and Berger-Wolf [25] described an online sampling technique to sample large social networks in order to discover the most influential individuals within the network.

The distinction between the aim of existing graph sampling methods and our objective is fundamental: The early works are seeking to obtain a smaller subgraph capture the key properties of the original graph. In contrast, we aim to supervise the sampling process to explore the network by visiting more important nodes belonging to a desired class.

Our work is also related to active sampling [1], in which both instances' labels and edges are acquired through an iterative process to update the classifier for discovering the nodes with a specific label. This work assumes that a node has no other known attributes apart from its label. In contrast, we formulate a supervised learning task by combining the network structure with rich node and edge attributes and use it to guide a random walk on the graph for discovering the nodes having a particular label while sampling the network.

3. Problem definition

Let $G = (\mathcal{V}, \mathcal{E})$ be an undirected graph where $\mathcal{V}$ denotes a set of nodes (or instances) and $\mathcal{E}$ denotes a set of edges between nodes. Each node $v_i \in \mathcal{V}$ is described by a feature vector x_i and a class label $y_i \in \mathcal{Y}$, where $\mathcal{Y}$ denotes a set of class labels. Each edge $(v_i, v_j) \in \mathcal{E}$ has a corresponding feature vector $r(v_i, v_j)$ which describes relationships between nodes v_i and v_j. In this work, the specific task under our consideration is a binary classification problem, in which each node v_i belongs to a positive class ($y_i = +1$) or a negative class ($y_i = -1$), and positive nodes comprise a small portion of the overall network.

Given a very small set of labeled nodes, also called seed nodes, $\mathcal{V}^l \in G$ with $\mathcal{V}^l = \{(v_i, y_i)\}_{i=1}^{K}$, our supervised sampling problem *aims* to: sample a representative, connected subgraph G' from the original large graph G, under a specific task, and identify as many positive nodes as possible through biasing the sampling process. The generated subgraph G' consists of the nodes, their attributes and labels, as well as the edges connecting the nodes. In our problem, In our problem, we assume that a full graph is too large for its global network structure to be known as a whole. Therefore, only a partial network G_t can be observed at time t. In this setting, the sampling process is, given a partially observed subgraph $G_t = (\mathcal{V}_t, \mathcal{E}_t)$, to decide which node v is the best to be sampled next. After a node is sampled, the subgraph is expanded to include a new node v, its neighboring nodes $\mathcal{N}(v)$, and new edges between them. The genuine label y of node v is also revealed.

4. Our proposed approach

The aim of supervised sampling is to obtain a maximum number of the nodes belonging to a desired category while sampling the network. However, traditional graph sampling techniques can not be directly applied to achieve this objective, because they all assume that the nodes are equally important for sampling. Therefore, we propose a novel algorithm to solve our supervised sampling problem.

Fig. 1 gives an example to illustrate key concepts behind our proposed algorithm. Given a partially observed subgraph G_t, which is a sampled network up to time t, we define two types of nodes: Intra-acquired nodes $\mathcal{I}_{intra}$ and Border-acquired nodes $\mathcal{I}_{border}$. Intra-acquired nodes are the nodes that have been sampled up to time t, and Border-acquired are those directly connected to Intra-acquired nodes. Our proposed algorithm is to determine which node from Border-acquired nodes $\mathcal{I}_{border}$ should be sampled next and perform sampling iteratively. For example, in Fig. 1, star node A is selected from Border-acquired nodes to be sampled next, because it has the maximum probability of being positive.

To enable our proposed algorithm to sample more positive nodes, one important issue is how to calculate the probability of nodes being positive. To this end, we model a graph as a Markov chain, where nodes are considered as different interior states and edges are chains between states. In particular, we consider two virtual absorbing states: one virtual positive node, and one virtual negative node. We assume that positive nodes are all connected to the virtual positive node, and negative nodes are all connected to the virtual negative node. Let p denote the probability of a node being positive, which is calculated as the probability for a node to transfer to the positive absorbing state. To capture such transition probabilities, we consider a random walk on the Markov chain, in which a walk is stopped when it reaches an absorbing state. Whereas traditional random walks assume that transition probabilities of all edges to be the same, we learn how to assign each edge a transition probability so that the random walk is more likely to visit positive nodes than other negative nodes in a network.

Below, we first formulate supervised random walks as an optimization problem and derive its solution. Based on this, we then discuss selection criteria used for sampling. Finally, we present our proposed algorithm for sampling networks.

4.1. Supervised random walks

Given an observed subgraph G_t, we propose a supervised random walk that naturally combines information from network topology structure with node and edge features. One way to bias the random walk is to assign each edge a random walk transition probability (i.e., strength). Therefore, we aim to learn a strength function $f_w(v,u)$ for each edge (u,v), based on features of nodes u and v, as well as the features of the edge (u,v). Intuitively, a random walk is more likely to traverse an edge with high strength, and thus the connected node via the path of the strong edge would be more likely visited by the random walk.

Now the task is to learn parameters w of function $f_w(v,u)$ that assigns each edge a transition probability. To achieve this, we formulate an optimization problem

$$\min_w F(w) = \sum_{i\in L+, j\in L-} h(p_j - p_i) + \sum_{y_i y_j = 1} \| p_i - p_j \|^2 + \lambda \| w \|^2 \tag{1}$$

where $L+$ and $L-$ is a set of labeled nodes with the positive label, and the negative label, respectively. The stationary distribution Dis_G of the random walk assigns each node a probability score p, which depends on $f_w(v,u)$ that is parameterized by w. Parameter λ controls the trade-off between the model complexity, i.e., norm of parameter vector w, and two constraints. $h(\cdot)$ is a loss function that assigns a non-negative penalty according to the difference of the scores $p_j - p_i$. If $p_j - p_i < 0$, then $h(\cdot) = 0$. If $p_j - p_i > 0$, then $h(\cdot) > 0$. Therefore, the first term of the objective function indicates that we want the probability scores of nodes in $L+$ to be greater than the scores of nodes in $L-$. The second term indicates that nodes having the same class label should have close probability scores. In the following, we discuss solutions to solve this optimization problem.

As discussed before, each edge (u,v) in a graph has a corresponding feature vector r_{uv} that describes nodes u and v (e.g., words in paper titles) and the interaction attributes (e.g., the time that the edge occurs, or how many words in their titles are shared). Thus, for edge (u,v) we define the strength function as $R_{u,v} = f_w(r_{u,v})$. Function f_w parameterized by w takes the edge feature vector $r_{u,v}$ as input and computes the corresponding edge strength $R_{u,v}$ that models the random walk transition probability. We then build the random walk stochastic transition matrix Tr:

$$Tr_{u,v} = \begin{cases} \dfrac{R_{u,v}}{\sum_v R_{u,v}} & \text{if } u, v \in \mathcal{E}, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Here, since two virtual absorbing states are only connected with labeled nodes having the same label, we can define the edge strength for virtual absorbing states $\hat{R}_{s,v}$. Let f_w be a linear function, $\hat{R}_{s,v}$ can be computed as

$$\hat{R}_{s,v} = \sum_{i\in\mathcal{N}(v)} R_{v,i}, \tag{3}$$

where $\hat{R}_{s,v}$ has the same linear form of f_w. Intuitively, $\hat{R}$ can be considered as the sum of the information flow originating from virtual absorbing states to node v's neighbors via node v on the Markov chain.

The vector P is the stationary distribution Dis_G of the random walk, and it is the solution to the following eigenvector equation

$$P^T = P^T Tr. \tag{4}$$

The above equation establishes relationships between the node probability scores $p_v \in P$ and the parameter w of function $f_w(r_{u,v})$ via the random walk transition matrix Tr.

Now we can minimize Eq. (1) with respect to parameter vector w. The optimization problem can be solved by deriving the gradient of $F(w)$ with respect to w, and then using a gradient based method to find w that minimizes $F(w)$. First, we have derivative of $F(w)$ with respect to w as

$$\begin{aligned} \frac{\partial F(w)}{\partial w} &= \sum_{i\in L+, j\in L-} \frac{\partial h(p_j - p_i)}{\partial w} + \sum_{y_i y_j = 1} \frac{\partial (p_i - p_j)}{\partial w} + 2\lambda \| w \|, \\ &= \sum_{i\in L+, j\in L-} \frac{\partial h(p_j - p_i)}{\partial (p_j - p_i)} \left(\frac{\partial p_j}{\partial w} - \frac{\partial p_i}{\partial w} \right) \\ &\quad + 2 \sum_{y_i y_j = 1} \left(\frac{\partial p_i}{\partial w} - \frac{\partial p_j}{\partial w} \right) + 2\lambda w. \end{aligned} \tag{5}$$

We can easily compute $\frac{\partial h(p_j - p_i)}{\partial (p_j - p_i)}$ for any differentiable loss function $h(.)$, for example squared loss. However, it is difficult to compute $\frac{\partial p_v}{\partial w}$ because we do not have the exact function form of $p(w)$. Therefore, we compute the derivative of p with respect to the vector w based on Eq. (4). Since Tr is a symmetric matrix, we have

$$p_v = \sum_i p_i Tr_{i,v}. \tag{6}$$

Therefore, the derivative of p_v is given as:

$$\frac{\partial p_v}{\partial w} = \sum_i Tr_{i,v} \frac{\partial p_v}{\partial w} + p_v \frac{\partial Tr_{i,v}}{\partial w}. \quad (7)$$

We can calculate this equation by iteratively computing p_v and $\frac{\partial p_v}{\partial w}$. Firstly, we compute p_v.

- Initialization: for $v \in V$, let $p_v^{(0)} = \frac{1}{|V|}$.
- Iteration: step n:

$$p_v^{(n)} = \sum_i p_i^{(n-1)} Tr_{i,v}. \quad (8)$$

Secondly, we compute $\frac{\partial p_v}{\partial w}$. For each $w_c \in w, c = 1, \ldots, |w|$, let $\frac{\partial p_v}{\partial w_c}^{(0)} = 0$ then for $v \in V$, we have

$$\frac{\partial p_v}{\partial w_c}^{(n)} = \sum_i Tr_{i,v} \frac{\partial p_v}{\partial w_c}^{(n-1)} + p_v^{(n-1)} \frac{\partial Tr_{i,v}}{\partial w_c} \quad (9)$$

To solve Eq. (1), we need to further calculate $\frac{\partial Tr_{i,v}}{\partial w}$ as

$$\frac{\partial Tr_{i,v}}{\partial w} = \frac{\frac{\partial f_w(r_{v,u})}{\partial w} \left(\sum_u f_w(r_{v,u})\right) - f_w(r_{v,u}) \left(\sum_u \frac{\partial f_w(r_{v,u})}{\partial w}\right)}{\left(\sum_u f_w(r_{v,u})\right)^2} \quad (10)$$

where $f_w(r_{v,u})$ is the edge strength function. We define f_w to be differentiable, so $\frac{\partial f_w(r_{v,u})}{\partial w}$ can be easily computed.

We now have an iterative way to calculate the derivation $\frac{\partial F(w)}{\partial w}$. After that, we can iteratively update parameters by using a gradient descent based method to solve the optimization problem and obtain optimal solutions for p and w.

4.2. Supervised sampling algorithm

One important objective of supervised sampling is to identify as many positive nodes as possible from the original network. We thus choose to sample the node which is most likely to be positive, and then acquire its neighbors, including neighboring nodes and edges. Based on supervised random walks, we can construct a Markov chain with probabilities, in which Dis_G is the optimal stationary distribution of the network. Each node v_i in the network G_t is assigned with a probability score p_i, indicating the probability of node v_i being positive. The probability score is used to guide the sampling process to more likely visit a positive node. Intuitively, if a node has a larger value of p_i, it is more likely to be a positive node because it is closer to the virtual positive node. Therefore, we choose a node v^* from Border-acquired nodes to be sampled next, such that it has the largest value of p_v,

$$v^* = \arg \max_{v \in \mathcal{I}_{border}} p_v. \quad (11)$$

The detailed procedure of our supervised sampling algorithm is given in Algorithm 1. This algorithm starts from some seed nodes (i.e., a few connected, labeled nodes), and iteratively samples other nodes in the network. At each step t, we construct a Markov chain based on the sampled network G_t obtained so far, and compute the stationary distribution Dis_G (lines 2–3). After that, the sampling process determines the best node v to be sampled using Eq. (11) (line 4). After node v is sampled, we obtain the information of its neighborhood, and the sampled network G_t is expanded to include the node v together with its genuine label, its neighboring nodes, and new edges between them. All such information is used to update the stationary distribution of the sampled network, which will guide the sampling at the next step.

Algorithm 1. Supervised sampling for networked data.

Input: (1) A network $G_t = (\mathcal{V}_t, \mathcal{E}_t)$ with seed nodes $\mathcal{V}_t^l$;
(2) The maximum number of sampled nodes: Budget.
Output The sampled network G_t.
1: **while** t ≤ Budget **do**
2: Construct a Markov chain for G_t;
3: Compute the stationary distribution Dis_G using our optimization problem Eq. (1);
4: Select a node v to be sampled using Eq. (11);
5: Update G_t with node v and edges between v and $\mathcal{N}(v)$;
6: $t = t+1$.
7: **end while**

5. Experiments

In this section, the effectiveness of the proposed algorithm is evaluated on both synthetic and real-world networks.

5.1. Experimental settings

5.1.1. Benchmark data

To study the algorithm performance with respect to different network features, we generate scale-free graphs with 400 nodes and 4000–6000 edges to simulate networks, including labeling information and features for the network nodes. Because real-world networks usually have community structures, we use random graph to create network components, each containing a number of nodes, and then connect these components by randomly creating edges between different components [26]. To generate a class label for each node, we simply assign all nodes within one component as one class (we focus on binary classification problems so each node is labeled as either $+1$ or -1). Details about synthetic networks are described in Section 5.2.

Besides synthetic networks, we also validate our proposed algorithm on PubMed citation network.[1] Detailed information is introduced in Section 5.3.

5.1.2. Baseline methods

To study the empirical performance of our proposed algorithm, referred to as Supervised Sampling for NETwork data (*SSNET*), we use four baseline methods for comparison:

[1] http://www.cs.umd.edu/projects/linqs/projects/lbc/Pubmed-Diabetes.tgz

- *USNET*: This is a variant of our proposed SSNET algorithm by removing the weight optimization module. In other words, USNET does not consider node features and there is no strength function for each edge. At each step, USNET computes the stationary distribution of a standard random walk. It chooses to sample a node with the maximum probability score.
- *Degree*: This method uses node degree as the measure to guide the sampling process. At each iteration, it samples the node with the maximum node degree and continuously outreaches to the neighbors of the selected node.
- *BFS*: This is the original BFS strategy for sampling the nodes from a network.
- *Random*: This method carries out network sampling in a completely random manner. At each iteration, it randomly selects a node to be sampled.

5.1.3. Performance metrics

Recall: Because the goal of supervised sampling is to obtain a sampled network that includes a maximum number of positive nodes and their structures. To this end, we use recall to compare different methods with respect to different sizes of sampled networks. In our experiments, although we know genuine class labels of each node, we carry out sampling without using class labels of the nodes (i.e., we pretend that we do not know node class labels during the sampling process). Once a node is sampled, we assign its genuine label to the node. By doing so, we can compute recall as the number of sampled positive nodes divided by the number of genuine positive nodes.

Network centrality: To evaluate the quality of the sampled network, in comparison with the original network, we focus on network structure, and compare the sampled network and the original network with respect to two popular measures: betweenness centrality and closeness centrality.

Betweenness measures the degree of brokerage [27,28] for nodes in a network. It measures how much information is propagated through each node. It is defined as

$$C_B(v) = \sum_{s,t \in V} \frac{\sigma(s,t|v)}{\sigma(s,t)}, \tag{12}$$

where $\sigma(s,t)$ is the number of shortest paths between nodes s and t in the graph, and $\sigma(s,t|v)$ is the number of (s,t)-paths that go through node v.

Closeness is another popular measure of centrality [29]. It measures how close a node is to all other nodes in the network as defined by the shortest path from the source node to the destination node

$$C_C(v) = \frac{1}{\sum_{t \in V} d(v,t)}, \tag{13}$$

where $d(v,t)$ is the (weighted, directed) distance from node v to node t in the graph.

5.1.4. General parameter settings

Setup: For supervised sampling, we need to set up several initial nodes to start the sampling process. In our experiments, we randomly choose three connected nodes as the initial network, which contains both positive and negative nodes, *e.g.* we start with two positive nodes and one negative node, and they are connected with each other. After that, the algorithm samples other nodes from the network in an iterative manner.

Edge strength function: According to Eq. (3), we employ a linear function $f_w(\cdot)$ to calculate the edge strength. Let r denote the feature vector of the edge connecting nodes u and v, $f_w(\cdot)$ is defined as

$$f_w(r_{u,v}) = w^T r. \tag{14}$$

Loss function: To define the penalty for the optimization function in Eq. (1), we use a common squared loss with margin b as:

$$h(x) = \max\{x+b, 0\}^2. \tag{15}$$

Parameter λ: λ is used as a regularization term for avoiding overfitting. However in our experiment, we find that vector w is relatively small and λ has little impact in our problem. Empirically, we set $\lambda = 1$ because it can give good performance.

5.2. Results on synthetic data

5.2.1. Synthetic networks

In our experiments, we build two synthetic networks, and nodes in the networks have two labels: positive and negative. The two networks have different network

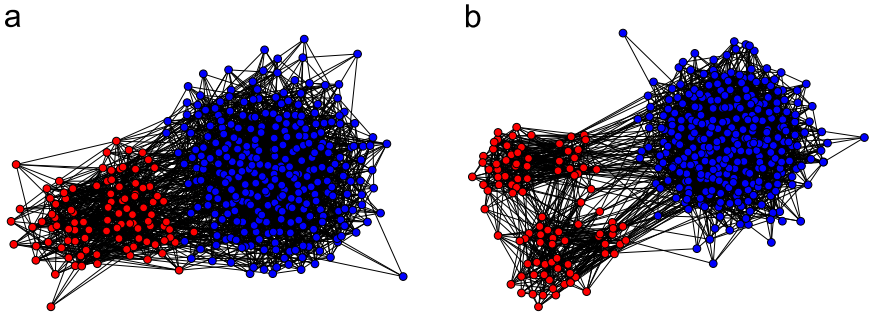

Fig. 2. A snapshot of two synthetic networks with different levels of biased node distributions. (a) P100-N300. (b) P50∗2-N300.

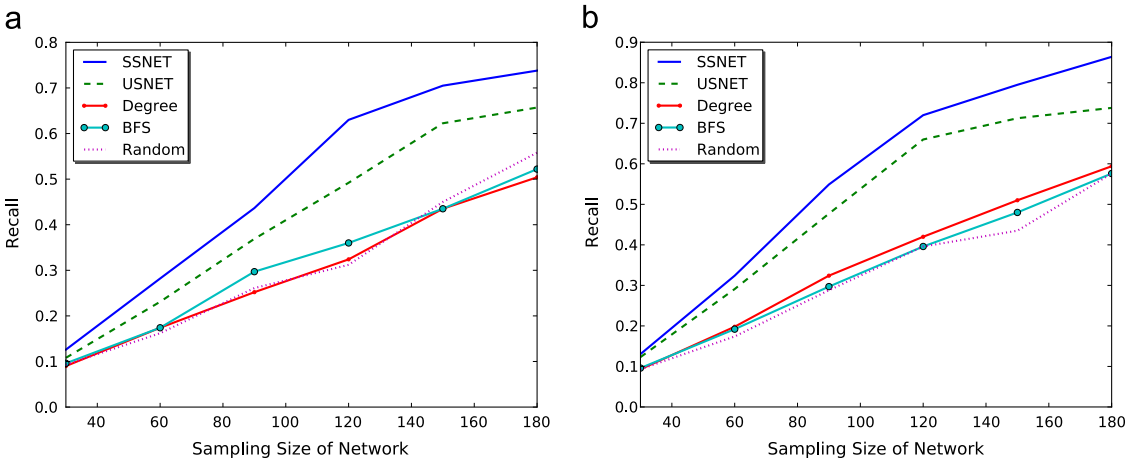

Fig. 3. Recall of positive nodes with respect to different sampling sizes. (a) P100-N300. (b) P50∗2-N300.

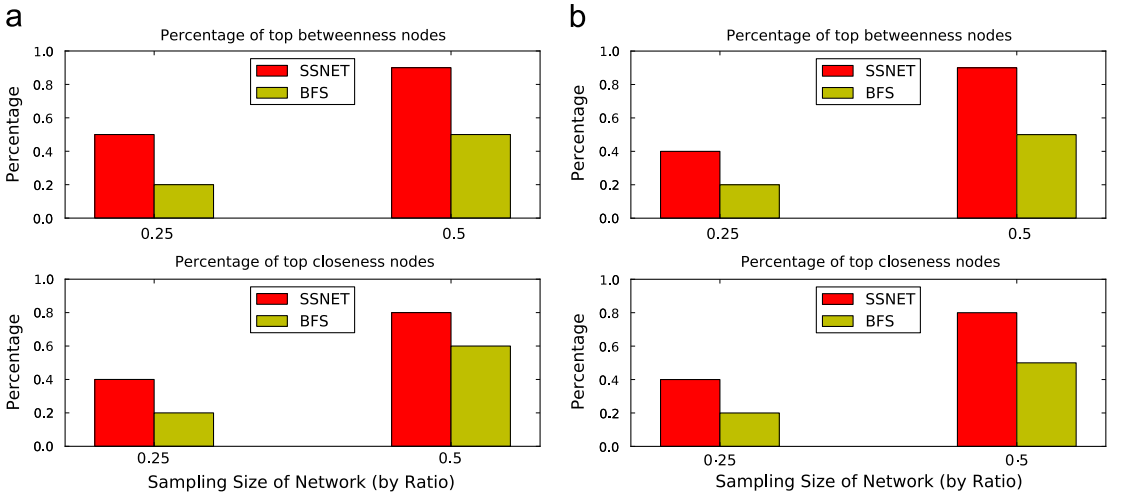

Fig. 4. Percentage (y-axis) of nodes with the Top-10 maximum betweenness scores (upper panel) and Top-10 maximum closeness scores (lower panel) in the original network discovered in the sampled network. The x-axis defines the ratio of the size of the sampled network compared to the original network. SSNET shows much better performance in preserving nodes with larger betweenness and closeness scores. (a) P100-N300. (b) P50∗2-N300.

features with different levels of biased node distributions. Snapshots of these two networks are shown in Fig. 2.

P100-N300: The P100-N300 network contains two components, which have 100 and 300 nodes, respectively. The component with 100 nodes belongs to the positive class, and the second component belongs to the negative class. Each node in the network has six random edges on average. After that, we randomly create 480 edges between two components. This network is used to simulate real-world situation with moderately biased node distributions.

P50∗2-N300: The P50∗2-N300 network contains three components, where the largest one contains 300 nodes, belonging to the negative class, and the other two components both contain 50 nodes, belonging to the positive class. Meanwhile, each node has six randomly connected edges within its component. After that, we create 480 edges to randomly connect the three components. This network is used to simulate real-world situation with severely biased node distributions.

For each node in the networks, we create two node features: (1) the first feature is a random variable which follows a zero mean (variance $\sigma=1$) Gaussian distribution. It acts as a noisy feature without any specific meaning; and (2) the second feature is a also a random variable with Gaussian distribution but subject to different means. Specifically, if a node belongs to positive class, it would follow a Gaussian distribution with $\mathcal{N}(0,1)$. If it belongs to negative class, it would follow a Gaussian distribution with $\mathcal{N}(1,1)$. In addition, given an edge (v,u) with two nodes u and v, we define the edge feature as

$$r^i_{v,u} = |x^i_v - x^i_u|, \quad i = 1,2. \tag{16}$$

5.2.2. Results

The results in Fig. 3 show that biased sampling can help acquire more positive nodes. SSNET and USNET are both biased towards sampling positive nodes using the probability scores of nodes, leading to higher recall values than other baselines with respect to the same sampling size.

SSNET outperforms USNET. It makes sense because the nodes belonging to the same class label in synthetic networks are correlated in the feature space. SSNET leverages the correlations, whereas USNET discards the edge strength that captures the node correlations. The recall achieved by Degree, BFS and Random are all very low, and significantly worse than SSNET. Degree, BFS and Random select the next node to be sampled solely based on the structure of the network. They all tend to sample nodes with high degree. In practice, positive nodes do not necessarily have a high degree, which explains why these methods fail in achieving good performance.

To evaluate the quality of sampled networks in preserving the major structure of the original network, we find the nodes with Top-k ($k=10$) betweenness and closeness scores in the original network and calculate the percentage of those nodes collected in the sampled network. We compare results of two different sampling sizes of networks in Fig. 4, which represents 50% and 25% size of the original network. The results show that SSNET achieves much better performance than BFS in preserving major network structures (*i.e.* including nodes with Top-k betweenness and closeness scores in the sampled network), especially when positive nodes are significantly infrequent in the networks. This indicates that SSNET is helpful not only in acquiring positive nodes, but also in preserving important network structures for positive nodes.

5.3. Results on real-world data

For real-world networks, we use an existing PubMed citation network, which includes 19,717 (i.e. nodes) scientific publications from the PubMed database pertaining to diabetes, and classifies each of them into one of three classes: "Diabetes Mellitus, Experimental" (7739), "Diabetes Mellitus Type 1" (7875), and "Diabetes Mellitus Type 2" (4103) (The number in the bracket denotes the number of papers in each class). The citation network consists of 44,338 links. We use the PubMed network to construct three network sampling problems for its three classes, respectively:

- *Problem* 1: we define "Diabetes Mellitus, Experimental" as positive and others as negative, and sample a network for "Diabetes Mellitus, Experimental".
- *Problem* 2: we define "Diabetes Mellitus Type 1" as positive and others as negative, and sample a network for "Diabetes Mellitus Type 1".

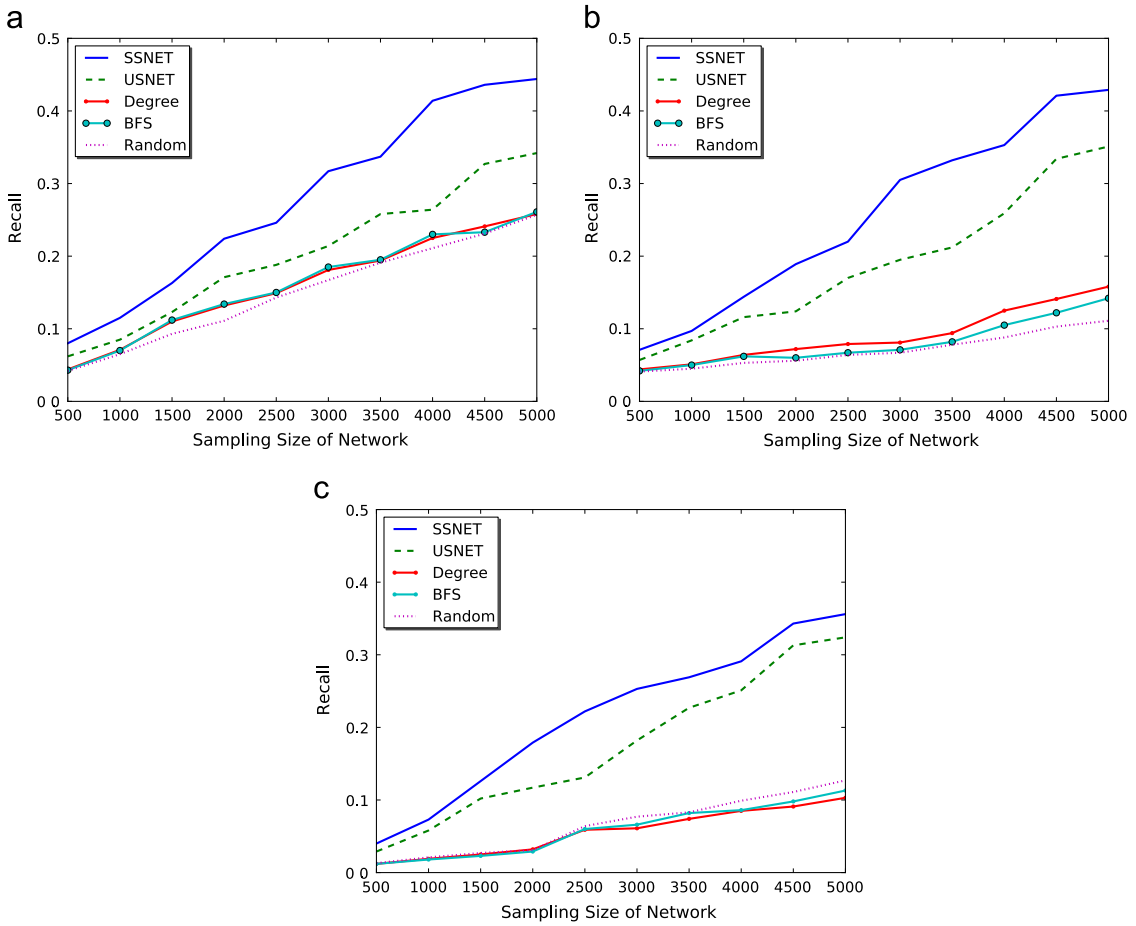

Fig. 5. Recall of positive nodes with respect to different sizes of sampled networks. (a) Diabetes Mellitus, Experimental. (b) Diabetes Mellitus Type 1. (c) Diabetes Mellitus Type 2.

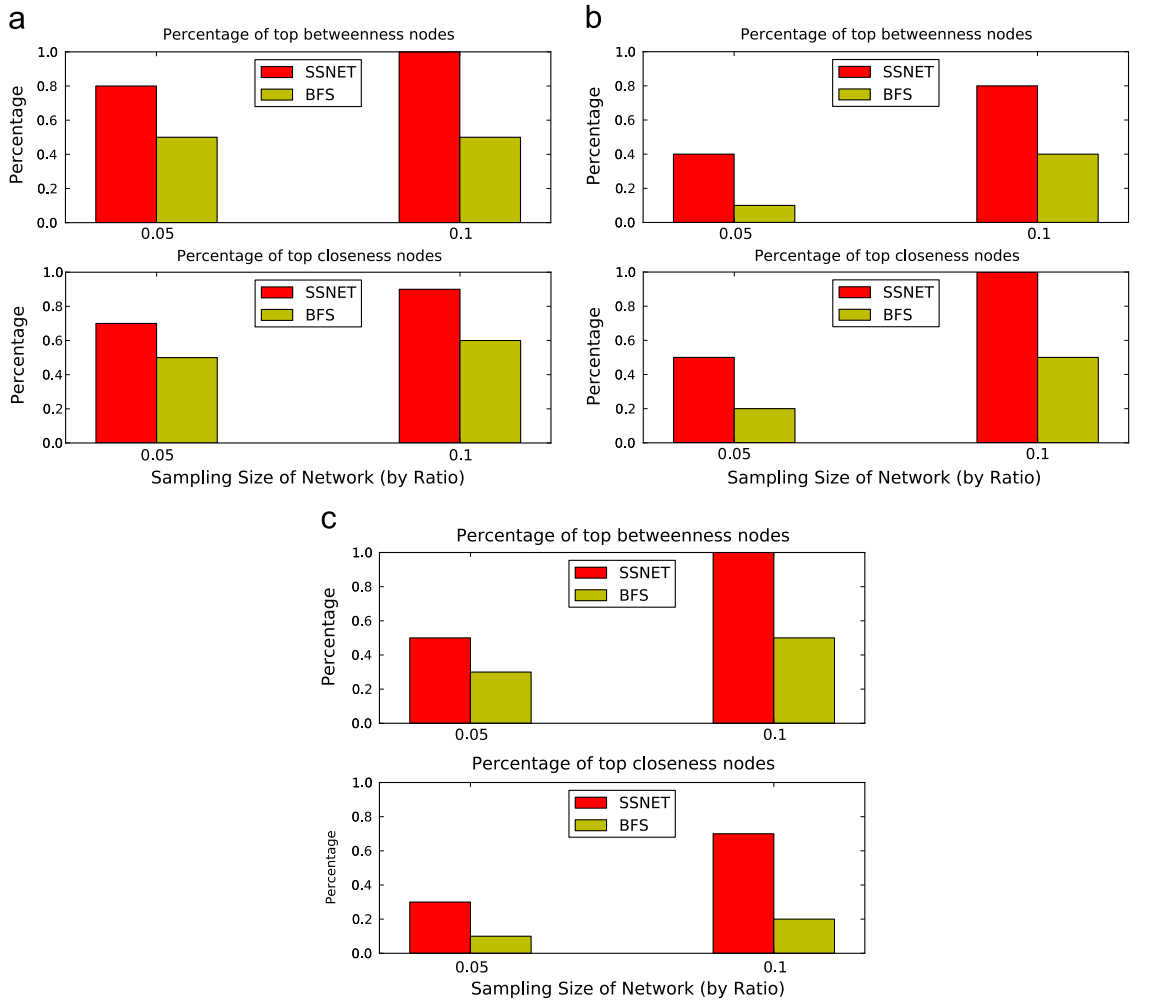

Fig. 6. Percentage (y-axis) of nodes with the Top-10 maximum betweenness scores (upper panel) and Top-10 maximum closeness scores (lower panel) in the original network discovered in the sampled network. The x-axis defines the ratio of the size of the sampled network compared to the original network. SSNET shows much better performance in preserving nodes with large betweenness and closeness scores. (a) Diabetes Mellitus, Experimental. (b) Diabetes Mellitus Type 1. (c) Diabetes Mellitus Type 2.

- *Problem* 3: we define "Diabetes Mellitus Type 2" as positive and others as negative, and sample a network for "Diabetes Mellitus Type 2".

In our experiments, we use node features to construct edge features. For each edge between two nodes, each representing a paper, the first edge feature is defined as the number of shared words between two papers

$$r^1_{u,v} = k, k = \left|\{w | w \in W_u \bigcap W_v\}\right|, \tag{17}$$

where W denotes the words of a paper. The second edge feature is defined as the cosine similarity between two papers

$$r^2_{u,v} = \cos(w_u, w_v), \tag{18}$$

where w is the bag-of-word vector to represent each paper using the occurrence of the words in the paper [30]. The edge strength function and loss function are the same as the ones used for synthetic networks.

Fig. 5 reports the recall of positive nodes with respect to different sampling sizes of networks. It shows that SSNET and USNET outperform the three baselines, Degree, BFS and Random, which do not use supervised sampling strategy for identifying positive nodes. In addition, SSNET works better than USNET. This is because the papers in the same class often share common keywords, which is captured by the edge strength function defined in SSNET. In comparison, USNET discards edge strength and therefore ignores the degree of correlations between papers during the sampling process.

The results in Figs. 5 also show SSNET has a larger slope of improvement at the beginning of the sampling process. After 4500 iterations, the recall values become relatively stable. This demonstrates that SSNET has good performance when the supervised sampling process starts. It can thus potentially find useful positive nodes with very little cost. The decreasing slope of performance improvement, at the latter stage of the sampling process, is mainly

because the number of undiscovered positive nodes decreases so it becomes more difficult to find them.

Fig. 6 reports the quality of the sampled networks in preserving major structure of the original PubMed network. The x-axis in the figure denotes the ratio of the size of the sampled network compared to the original network. The y-axis shows that out of the Top-k ($k=10$) nodes with the largest betweenness centrality and closeness centrality scores in the original network, how many of them (the percentage) actually appear in the sampled networks. The results clearly show that SSNET outperforms other methods in preserving important network structures.

6. Conclusion

In this paper, we introduced the problem of supervised sampling, which samples a large network to generate a small subset of nodes to represent the original network. Unlike most graph sampling algorithms which focused on generating a uniform random sample of the original graph, supervised sampling aims to sample a network under a specific goal of acquiring a maximum number of positive nodes. We proposed to model a network as a Markov chain, and derived a supervised random walk to learn a stationary distribution of the sampled network. We showed that the learned stationary distribution can help guide the sampling process to visit the nodes that are more likely to be positive. Experiments on both synthetic and real-world networks demonstrated that our supervised sampling can indeed identify more positive nodes than other methods, particularly for networks with imbalanced class distributions.

References

[1] J. Pfeiffer III, J. Neville, P. Bennett, Active sampling of networks, in: Proceedings of the ICML Workshop on Mining and Learning with Graphs, Edinburgh, Scotland, 2012.

[2] M. Fang, J. Yin, X. Zhu, C. Zhang, Active class discovery and learning for networked data, in: Proceedings of the 2013 SIAM International Conference on Data Mining, SIAM, Austin, TX, USA, 2013, pp. 315–323.

[3] M. Gjoka, M. Kurant, C. Butts, A. Markopoulou, Walking in facebook: a case study of unbiased sampling of osns, in: Proceedings of the 2010 IEEE Conference on Computer Communications, IEEE, San Diego, CA, USA, 2010, pp. 1–9.

[4] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, T. Eliassi-Rad, Collective classification in network data, AI Mag. 29 (3) (2008) 93.

[5] L. Backstrom, J. Leskovec, Supervised random walks: predicting and recommending links in social networks, in: Proceedings of the 4th International Conference on Web Search and Data Mining, ACM, Hong Kong, 2011, pp. 635–644.

[6] J. Ye, H. Cheng, Z. Zhu, M. Chen, Predicting positive and negative links in signed social networks by transfer learning, in: Proceedings of the 22nd International Conference on World Wide Web, ACM, Rio de Janeiro, Brazil, 2013, pp. 1477–1488.

[7] M. Bilgic, L. Mihalkova, L. Getoor, Active learning for networked data, in: Proceedings of the 27th International Conference on Machine Learning, Haifa, Israel, 2010, pp. 79–86.

[8] M. Fang, J. Yin, X. Zhu, Transfer learning across networks for collective classification, in: Proceedings of the 13th IEEE International Conference on Data Mining, IEEE, Dallas, TX, USA, 2013, pp. 161–170.

[9] M. Fang, J. Yin, X. Zhu, C. Zhang, TrGraph: Cross-network transfer learning via common signature subgraphs, IEEE Trans. Knowl. Data Eng. 27 (9) (2015) 2536–2549.

[10] J. He, W.W. Chu, A social network-based recommender system (SNRS), Data Min. Social Netw. Data, Springer, 2010, 47–74.

[11] M. Fang, D. Tao, Networked bandits with disjoint linear payoffs, in: Proceedings of SIGKDD, ACM, 2014, pp. 1106–1115.

[12] J. Yu, Y. Rui, D. Tao, Click prediction for web image reranking using multimodal sparse coding, IEEE Trans. Image Proc. 23 (5) (2014) 2019–2032.

[13] S. Wasserman, K. Faust, Social Network Analysis, Cambridge University Press, 1995.

[14] Y.-Y. Ahn, S. Han, H. Kwak, S. Moon, H. Jeong, Analysis of topological characteristics of huge online social networking services, in: Proceedings of the 16th International Conference on World Wide Web, ACM, Banff, Alberta, Canada, 2007, pp. 835–844.

[15] A. Mislove, M. Marcon, K. Gummadi, P. Druschel, B. Bhattacharjee, Measurement and analysis of online social networks, in: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement, ACM, San Diego, CA, USA, 2007, pp. 29–42.

[16] A. Mislove, H. Koppula, K. Gummadi, P. Druschel, B. Bhattacharjee, Growth of the flickr social network, in: Proceedings of the 1st Workshop on Online Social Networks, ACM, Seattle, WA, USA, 2008, pp. 25–30.

[17] B. Viswanath, A. Mislove, M. Cha, K. Gummadi, On the evolution of user interaction in facebook, in: Proceedings of the 2nd ACM SIGCOMM Workshop on Online Social Networks, ACM, Barcelona, Spain, 2009, pp. 37–42.

[18] C. Wilson, B. Boe, A. Sala, K. Puttaswamy, B. Zhao, User interactions in social networks and their implications, in: Proceedings of the 4th ACM European Conference on Computer Systems, ACM, Nuremberg, Germany, 2009, pp. 205–218.

[19] L. Becchetti, C. Castillo, D. Donato, A. Fazzone, A comparison of sampling techniques for web graph characterization, in: Proceedings of the 2006 Workshop on Link Analysis: Dynamics and Static of Large Networks, Philadelphia, PA, 2006.

[20] S. Ye, J. Lang, F. Wu, Crawling online social graphs, in: Proceedings of the 12th International Asia-Pacific Conference on Web Conference, IEEE, Busan, South Korea, 2010, pp. 236–242.

[21] L. Lovász, Random walks on graphs: a survey, combinatorics, Paul erdos is eighty 2 (1) (1993) 1–46.

[22] C. Hübler, H.-P. Kriegel, K. Borgwardt, Z. Ghahramani, Metropolis algorithms for representative subgraph sampling, in: Proceedings of the 8th IEEE International Conference on Data Mining, IEEE, Pisa, Italy, 2008, pp. 283–292.

[23] J. Leskovec, C. Faloutsos, Sampling from large graphs, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, Philadelphia, PA, USA, 2006, pp. 631–636.

[24] M. Papagelis, G. Das, N. Koudas, Sampling online social networks, IEEE Trans. Knowl. Data Eng. 25 (2013) 662–676.

[25] A. Maiya, T. Berger-Wolf, Online sampling of high centrality individuals in social networks, in: Proceedings of the 14th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Hyderabad, India, 2010, pp. 91–98.

[26] P. Erdős, A. Rényi, On random graphs I, Publ. Math.-Debr. 6 (1959) 290–297.

[27] S.P. Borgatti, M.G. Everett, A graph-theoretic perspective on centrality, Soc. Netw. 28 (4) (2006) 466–484.

[28] G. Sabidussi, The centrality index of a graph, Psychometrika 31 (4) (1966) 581–603.

[29] M.A. Beauchamp, An improved index of centrality, Behav. Sci. 10 (2) (1965) 161–163.

[30] G. Namata, P. Sen, M. Bilgic, L. Getoor, M. Sahami, A. Srivastava, Collective classification for text classification, in: Text Mining: Classification, Clustering, and Applications, Taylor and Francis Group, 2009.