Classification in Social Networks

Zehra Çataltepe and Abdullah Sönmez

Abstract Production of social network data in different kinds and huge amounts brings with it classification problems which need to be solved. In this chapter, we introduce a framework for classification in a social network. Aggregation of neighbor labels and sampling are two important aspects of classification in a social network. We give details of different aggregation methods and sampling methods. Then, we discuss different graph properties, especially homophily, which may be helpful in determining which type of classification algorithm should be used. We give details of a collective classification algorithm, ICA (Iterative Classification Algorithm), which can be used for semi-supervised learning in general and transductive learning in particular on a social network. We present classification results on three different datasets, using different aggregation and sampling methods and classifiers.

1 Introduction

In most machine learning applications, the observed and unobserved instances are assumed to be drawn independently from the same distribution. Classification problems are solved using instances' features (content) and labels. Connections/dependencies/ relations between instances are not taken into consideration. On the other hand, learning problems with network information, where for each node its features and relations with other nodes are available, become more common in our lives. Examples include social [39], semantic [36], financial [3], communication [10] and gene regulatory [2] networks. Classification of nodes or links in the

Z. Çataltepe (✉) • A. Sönmez
Istanbul Technical University, Computer Engineering Department, Maslak, Istanbul 34469, Turkey
e-mail: cataltepe@itu.edu.tr; sonmezab@itu.edu.tr

Ş. Gündüz-Öğüdücü and A.Ş. Etaner-Uyar (eds.), *Social Networks: Analysis and Case Studies*, Lecture Notes in Social Networks, DOI 10.1007/978-3-7091-1797-2_7,

network, discovery of links or nodes which are not yet observed or identification of essential nodes or links, are some of the research areas in social networks.

In a social network, the class membership of one node may have an influence on the class membership of a related node. Networked data contain not only the features for each node, but also features and sometimes labels of neighbors and sometimes the features and links of the nodes for which labels need to be estimated. Classification in social networks have a number of challenges compared to classification of plain data:

- Design of classifiers which can make the best use of both content and link information available in the social network data is a challenge.
- One other issue that must be considered is the separation of data into training and test sets, because using traditional random sampling methods may not give healthy results [21].
- Even nodes connected through a chain of links may affect each other's class. Classification algorithms may need to take this dependency into account.

In this chapter, we try to address these challenges. Section 4 describes random and snowball sampling which can be used to partition networked data into training and test sets so that different learning methods can be tested against each other. In Sect. 5 we show how neighbor labels may be used through different aggregation mechanisms. In Sect. 6 we discuss the classification methods for networked data and especially the collective classification algorithms, which allow test nodes to be labeled based on the actual labels of training nodes and the estimated labels of test nodes.

In addition to these, we give the notation used in Sect. 2. Details on graph properties, which are important in understanding social networks data and which algorithms to use for learning, are given in Sect. 3. Experimental results on three different datasets are in Sect. 7. The chapter is concluded with Sect. 8.

2 Notation

We assume that there is a networked dataset represented by a graph $G = (V, E)$ with nodes (vertices) V and undirected links (edges) $E \subseteq \{\{u, v\} | u, v \in V\}$.

Each node $u \in V$ has a C dimensional label vector $\mathbf{r}(u) \in \{0, 1\}^C$ which uses 1-of-K representation and shows the class of the node. Some of the vertices are in the training set V_{train} whose labels are known, while the rest are in the test set V_{test} whose labels will be predicted. Note that, $V_{train} \cap V_{test} = \emptyset$ and $V_{train} \cup V_{test} = V$. Note also that if the test set is unknown then $V_{test} = \emptyset$.

Each node $u \in V$ (whether it is in the training or test set) also has a d dimensional node content feature vector $\mathbf{x}(u) \in \{0, 1\}^d$.

In the pattern recognition scenario that we are interested in, given feature vectors of the training nodes and their labels, $\mathbf{x}(u)$ and $\mathbf{r}(u)$, $u \in V_{train}$, we need to train a mapping (classifier) $g(\mathbf{x}(u)) : \{0, 1\}^d \rightarrow \{0, 1\}^C$. Since the classifier $g(\mathbf{x}(u))$

uses only the input features, we will call it the Content Only (CO) classifier $g(\mathbf{x}(u)) = g_{CO}(\mathbf{x}(u))$.

When not only the training node features, but also their links are given, the link information can also be used for classification. Usually link information of neighbors of a specific node are taken into account. The neighborhood function $N(u)$ returns a set of nodes which are neighbors of node u according to the links L:

$$N(u) = \{v : \{u, v\} \in L\}. \tag{1}$$

Most classifiers need fixed dimensional inputs. So, we need to aggregate [29] labels of neighbors of a node into a fixed dimensional vector. Let $\mathbf{r}_N(u)$ denote the *aggregated neighbor labels* for a node u. We will define $\mathbf{r}_N(u)$ based on the aggregation method (Sect. 5) used.

Based on the labels of the neighbors only, a classifier, which we call the Link Only (LO) classifier $g_{LO}(\mathbf{r}_N(u))$ can be trained on the training data. When a test node needs to be classified, if it has neighbors in test set, inputs to the classifier need to be determined iteratively, based on the current label assignment of the neighbors using a collective classification algorithm such as the Iterative Classification Algorithm (ICA) [21, 34].

When both node features and links are known, a classifier that uses both the content features of the node and labels of the neighbors have been used in [34]. We will call this classifier with $d + C$ features, the Content and Link classifier:

$$g_{CL}([\mathbf{x}(u)\ \mathbf{r}_N(u)]). \tag{2}$$

Evaluation of a classifier can be based on its accuracy on the test set:

$$acc(g, V_{test}) = \frac{1}{|V_{test}|} \sum_{v \in V_{test}} 1 - [g(\mathbf{x}(v)) = \mathbf{r}(v)]. \tag{3}$$

Here $[P]$ is the Iverson bracket and returns 1 if condition P is true (i.e. vectors $g(\mathbf{x}(v))$ and $\mathbf{r}(v)$ are equal) and returns 0 if condition P is false (i.e. $g(\mathbf{x}(v))$ and $\mathbf{r}(v)$ differ in at least one position). The test labels and sometimes the identities of the test inputs are not known during training. Since the actual labels are not known, test accuracy can not be evaluated. Instead, validation set(s) extracted from the labeled training set through sampling (Sect. 4) is(are) used to estimate the test accuracy of a classifier.

3 Graph Properties

Social network data consist of a graph with nodes, node features and labels and links between the nodes. Graph properties, such as homophily, degree distribution help us understand the data at hand. Graph properties can also be used as guidelines

in finding out if a sampling (Sect. 4) used for evaluation of algorithms is a good one or not, or which type of aggregation (Sect. 5) of neighbor labels should be used for classification. In this section, we define some of the important graph properties. Please see, for example [27] or [11], for more details on graph properties.

3.1 Homophily

A classification algorithm in a social network aims to use both content and link information. Whether the link information will be useful or not depends on whether linked objects have similar features and or labels. This similarity have been quantified using a number of different criteria.

Neville and Jensen defined two quantitative measures of two common characteristics of relational datasets: *concentrated linkage* and *relational autocorrelation*. Concentrated linkage occurs when many entities are linked to a common entity like the citation links to the key papers. Relational autocorrelation occurs when the features of the entities are similar among entities that share a common neighbor [17]. As pointed out by Neville and Jensen, most of the models (e.g., PRMs, RMNs) do not automatically identify which links are the most relevant to the classification task. In their method, they defined links which are most relevant to the classification task by using concentrated linkage and relational autocorrelation, and explored how to use these relational characteristics to improve feature selection in relational data [42].

Yang et al. identified five hypertext link regularities that might (or not) hold in a particular hypertext corpus [40, 42]. We list three of them here.

- *Encyclopedia regularity:* The class of a document is the same as the class of the majority of the linked documents.
- *Co-referencing regularity:* Documents with the same class tend to link to documents not of that class, but which are topically similar to each other.
- *Partial co-referencing regularity:* Documents with the same class tend to link to documents that are topically similar to each other, but also link to a wide variety of other documents without semantic reason. The presence (or absence) of these regularities may significantly influence the optimal design of a link-based classifier. Most of link analysis methods and link-based classification models are built upon the "encyclopedia" or "co-referencing" regularity. As a result, the models do not automatically identify which links are most relevant to the task [42].

Assortativity index defined by [27] is also a measure of how similar are the labels of connected nodes. Assortativity index is defined using the number of links which connect nodes of same and different classes and it is proportional to the number of links that connect nodes of the same class.

Homophily [21, 31] or label autocorrelation can be defined as the tendency of entities to be related to other similar entities, i.e. linked entities generally have a

tendency to belong to the same class. High homophily usually implies that using link information helps with classification while for low homophily datasets using a content only classifier could do a better job.

We define homophily of a node u as the proportion of neighbor nodes of u which have the same label as u:

$$H(u) = \frac{1}{N(u)} \sum_{v \in N(u)} [\mathbf{r}(u) == \mathbf{r}(v)] \tag{4}$$

In this equation [] is the Iverson bracket and $[p]$ is equal to 1 if p is true, it is 0 otherwise. The graph homophily is then:

$$H(G) = \frac{1}{|V|} \sum_{u \in V} H(u) \tag{5}$$

Note that although homophily is defined as label "sameness", labels could be related to each other, and for example could just be the opposite of each other for a binary classification problem (i.e. heterophily). The classifiers that we use would be able to take advantage of homophily, heterophily or any other correlation between the label of a node and labels its neighbors.

3.2 Degree Distribution

Degree, $k(u)$, of a node u is the total number of its connections to the other nodes in the network. Although the degree of a node seems to be a local quantity, degree distribution of the network often may help to determine some important global characteristics of networks. A scale-free network is type of a network whose degree distribution follows a power law [11]. It was shown that whether an epidemic spreads in a scale-free network or stops can be computed based on the scale free exponent in [16].

3.3 Clustering Coefficient

Clustering coefficient is a measure of degree to which nodes tend to cluster together in a graph. Clustering coefficient property can give information, for example, on how close are two nodes in the graph, and therefore how much their predicted labels would affect each other during prediction of labels. Different types of clustering coefficients can be defined as follows.

3.3.1 Global Clustering Coefficient

The global clustering coefficient, which is a measure of indication of the clustering in the whole network is based on triplets of nodes. A triplet is called an open triplet when three nodes are connected with two links and it is called a closed triplet when all three nodes are tied together. Three closed triplets, one centered on each of the nodes, form triangle. The global clustering coefficient is defined as the ratio of the number of closed triplets to the total number of triplets (open and closed) [11]:

$$CC_G(G) = \frac{Number\ of\ closed\ triplets}{Number\ of\ connected\ triplets} \tag{6}$$

3.3.2 Local Clustering Coefficient

Local clustering coefficient is defined in [11] as the ratio of the actual number of edges between the neighbors of a node u to all the possible numbers of edges is the local clustering coefficient for node u:

$$CC_L(u) = \frac{\sum_{v_1,v_2 \in N(u), v_1 \neq v_2} [\{v_1, v_2\} \in E]}{|N(u)|(|N(u)| - 1)/2} \tag{7}$$

Here [] is the Iverson bracket. The number of all possible undirected links between neighbors $N(u)$ of node u is $|N(u)| * (|N(u)| - 1)/2$.

3.3.3 Average Clustering Coefficient

The network average clustering coefficient is defined by Watts and Strogatz as the average of the local clustering coefficients of all the nodes in the graph[38]:

$$CC_L(G) = \frac{1}{|V|} \sum_{u \in V} CC_L(u) \tag{8}$$

If the average clustering coefficient is zero, then the graph is a tree. If the average clustering coefficient is higher than a random graph with the same degree distribution, then the network may show the small world phenomenon, i.e. any two random nodes can be connected using much smaller number of links than $O(|V|)$. It should be noted that during calculation of the average clustering coefficient the nodes having less than two neighbours, which naturally have a clustering coefficient of zero, are not considered in the average clustering coefficient computation.

3.4 Rich Club Coefficients

Let $V_k \subseteq V$ denote the nodes having degree higher than a given value k and $E_{>k}$ denote the edges among nodes in V_k. The rich-club coefficient is defined as:

$$\phi_k(G) = \frac{|E_{>k}|}{|V_{>k}|(|V_{>k}| - 1)/2} \tag{9}$$

In Eq. (9) $|V_{>k}|(|V_{>k}| - 1)/2$ represents the maximum possible number of undirected links among the nodes in $V_{>k}$. Thus, $\phi(k)$ measures the fraction of edges actually exist between those nodes to the maximum number of edges they may have. The rich club coefficient helps to understand important information about the underlying architecture revealing the topological correlations in a complex network [7].

3.5 Degree-Degree Correlation

Degree-degree correlation is the correlation between the number of neighbors (minus 1) of neighboring nodes. Both homophily and degree-degree correlation can be considered as special case of assortativity [26], the correlation between a certain property (label, degree, etc.) of two neighboring nodes. Average degree of the neighbours' of the nodes having degree k. Kahng et al. [19] found out that among scale-free networks authorship and actor networks are assortative (i.e. nodes with large degree connect to nodes with large degree) while protein-protein interaction and world wide web networks are disassortative (i.e. large degree nodes tend to connect to small degree nodes).

3.6 Graph Radius and Diameter

These two notions are related to each other. We use the definitions from [12].

Let $d_G(u, v)$ be the distance (i.e. the minimum number of edges that connect u and v) between nodes u and v. The eccentricity $\epsilon(u)$ of a node u is defined as the greatest distance between u and any other node in the graph:

$$\epsilon(u) = \max_{v \in V} d_G(u, v) \tag{10}$$

The diameter of a graph is defined as the length of the longest of the shortest paths between any two nodes or equivalently as the maximum eccentricity of any vertex:

$$diam(G) = \max_{u,v \in V} d_G(u, v) \tag{11}$$

The radius of the graph is defined as the minimum eccentricity among all the nodes in the graph:

$$rad(G) = \min_{u \in V} \epsilon(u) = \min_{u \in V} \max_{v \in V} d_G(u, v) \tag{12}$$

3.7 Average Path Length

The Average Path Length of a graph is defined as the average of all shortest paths between nodes:

$$APL(G) = avg_{u \in V} \max_{v \in V} d_G(u, v) \tag{13}$$

[13] has computed the average path length based on whether the graph is a scale free one or not and the scale free exponent.

3.8 Graph Density

The density of a graph is defined as the ratio of the number of edges to the number of possible edges:

$$D(G) = \frac{|E|}{|V|(|V| - 1)/2} \tag{14}$$

3.9 Graph Properties of the Datasets

We used three datasets that have been used in network classification research [21, 23, 34]. We give their graph properties in Table 1.

3.9.1 Cora Dataset

Cora [22] data set consists of information on 2708 Machine Learning papers. Every paper in Cora cites or is cited by at least one other paper in the data set. There are 1,433 unique words that are contained at least 10 times in these papers. There are also seven classes assigned to the papers according to their topics. For each paper, whether or not it contains a specific word, which class it belongs to, which papers it cites and which papers it is cited by are known. Citation connections and paper features (class and included words) are contained in two separate files. Total number of connections between the papers is 5,278. There are 3.898 links per paper.

Table 1 Summary information about the datasets (Cora, Citeseer, WebKB)

DataSet	Cora	CiteSeer	WebKB
Number of features	1,433	3,703	1,703
Number of nodes	2,708	3,312	877
Number of unique links	5,278	4,536	1,388
Number of classes	7	6	5
Average degree	3.90	2.74	3.17
Diameter	19	28	8
Average path length	6.31	9.30	3.14
Density	0.0014	0.0008	0.0036
Global clustering coefficient	0.094	0.130	0.0362
Average clustering coefficient	0.293	0.243	0.294
Alpha (power law)	1.867	1.638	1.810
Homophily	0.83	0.71	0.13

3.9.2 Citeseer Dataset

CiteSeer [14, 33] data set consists of information on 3,312 scientific papers. There are 3,703 unique words that are contained at least 10 times in these papers. There are six classes assigned to the papers according to their topics. Just as in the CoRA dataset, word, class and cites and cited by information are given in two separate files. Total number of connections between the papers is 4,536. There are 2.74 links per paper.

3.9.3 WebKB Dataset

WebKB [9] data set consists of sets of web pages from four computer science departments, with each page manually labeled into five categories: course, project, staff, student, or faculty.

Link structure of WebKB is different from Cora and Citeseer since co-citation links are useful for WebKB data set. The reason for that can be explained based on the observation that a student is more likely to have a hyperlink to her adviser or a group/project page rather than to one of her peers [21].

4 Sampling

The test data in social networks may contain a bunch of nodes that are connected to each other, in which case the training-validation partitioning process needs to take this dependency into account. It is also possible that the test data are randomly distributed among the training nodes. Two different sampling mechanisms, snowball sampling and random sampling, are used to handle these two situations.

4.1 Random Sampling

When random sampling is used, nodes in training, validation, and test sets are selected. It is important to preserve class distribution of the dataset as much as possible during selection of the nodes. One method to achieve this is to partition nodes from every class among themselves randomly proportional to the required train, validation and test set sizes and then combine the nodes from every class in the training set to produce the training set and do the same for validation and test sets. While random sampling is a simple method, even if care is taken to preserve the class ratios, the sampled graph is likely to have very different topological properties than the original graph [1], therefore, the classification algorithms trained/tested on the sampled graph may not perform similarly on the test set or the original dataset.

4.2 Snowball Sampling

When the usual k-fold cross-validation training and test sets are obtained on networked data, especially when the number of links per node is low, k-fold random sampling may generate almost disconnected graphs [34], making learning through the links in the networked data impossible. To overcome the issue of disconnected graphs in k-fold cross validation, snowball sampling is used. In snowball sampling, first of all, different starting nodes are selected. Then new nodes are selected among the nodes which are accessible through the selected nodes. Thus, the selected nodes grow like a snowball and as a result selected nodes are connected. It is important to preserve the class ratios in the selected set of nodes, therefore, at every point during sampling, if a class is underrepresented, it is given higher probability. This sampling procedure continues until there are enough nodes in the selected subset.

In Fig. 1, visualization of train/test partitions for Cora and Citeseer datasets are given for both random sampling and snowball sampling. Red nodes are nodes in the test partition while green ones are in the train partition. Nodes' actual labels are also displayed in the circles representing the nodes. In order to be able to visualize these two sets a subsample of 4 % of the actual data is used.

When random sampling is used, there is no order or dependency between the selection of training, validation and test sets. However, when snowball sampling is used, whether the snowball is selected and taken to be the test set or the training set may give different results. Taking the snowball to be the test set [34], generating k disjoint snowballs and using them for training-validation set formation [23], using temporal sampling and past data for training and generating snowball samples with some portion of provided labels for validation [24] are all different uses of snowball sampling for classification of networked data. While some authors mention that snowball sampling causes bias towards highly connected nodes and may be more suitable to infer about links than to infer about nodes in a social network [35], others suggest that since snowball is not guaranteed to reach individuals with

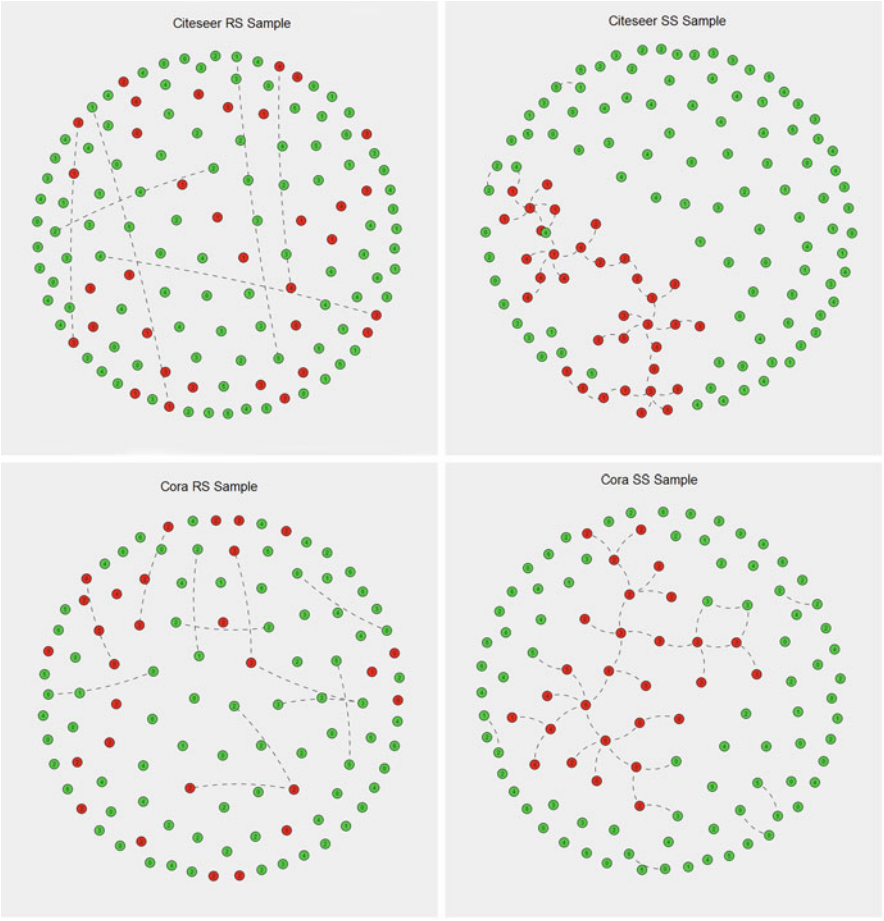

Fig. 1 Random sampling vs snowball sampling

high connectivity and would not reach disconnected individuals, snowball's starting nodes should be chosen carefully [15].

When random sampling is used to generate k-fold cross validation training and validation (and test) sets, there are no overlaps between different test sets. However, when snowball sampling is used to generate the k test sets, the test snowballs created may overlap. Since linked instances in a snowball are correlated, errors made on them may also be correlated. The statistical tests, such as the paired t-test, which is used for model selection, may not give reliable results when test sets are not independent [25]. Forest Fire Sampling (FFS) [20] method may be considered as an alternative to snowball sampling, in this algorithm, as in snowball sampling, a breadth first search technique is used but some of the followed links are burned according to a probability distribution.

4.3 Sampling on Streaming Graphs

When the network has many nodes or the nodes are observed as a stream (as in the case of twitter for example), it is not possible to consider all of the graph when sampling. For streaming or large graphs the sampling algorithm needs to be space and time efficient. In [1] a sampling algorithm for streaming graphs called Partially-Induced Edge Sampling (PIES) is introduced. PIES algorithm always keeps a constant number of nodes in the sample and drops old nodes and adds new observed ones as the network keeps being observed. Note that the same idea can also be used when the graph is too large to fit in the memory, as new nodes are explored, they can be considered as a stream. Ahmed et al. [1] considers sampling algorithms based on network nodes, edge, and topology-based sampling for three different types of networks, static-small, static-large and streaming. Snowball sampling is a topology based sampling method. In [1], the authors' objective is to ensure that the sampled graph is a representative subgraph which matches the topological properties of the original graph.

5 Aggregation

Each node in a graph may have a different degree and therefore different number of neighbors. On the other hand, most classifiers need the input dimensionality for each instance to be the same. Therefore, in order to take advantage of neighbor link or feature information, a mechanism to make them the same dimensional, regardless of the identity of a node is needed. Aggregation methods (also called propositionalization methods or flattening methods) are often used for this purpose. In this section we give common aggregation methods used for aggregation of neighbor labels so that they can be used for classifier training.

The main objective of aggregation in relational modeling is to provide features which improve the generalization performance of the model [29]. However aggregation usually causes loss of information, therefore one needs to be careful about not losing predictive information. Perlich and Provost proposed general guidelines for designing aggregation operators, suggesting that aggregation should be performed keeping the class labels under consideration, aggregated features should cause instances of the same class to be similar to each other and different aggregation operators should be experimented with. Below, we will present performances of different aggregation methods on different datasets. In [29] authors considered both simple aggregators and new more complex aggregators in the context of the relational learning system ACORA (Automated Construction of Relational Attributes). ACORA computes class-conditional distributions of linked object identifiers, and for an instance that needs to be classified, it creates new features by computing distances from these distributions to the values linked to the instance [29]. Lu and Getoor considered various aggregation methods: existence(binary), mode and value counts. The count method performed best in their study [28].

In the following sections, the notation introduced in Sect. 2 is used. Note that, although the neighborhood function $N(u)$ is usually defined to include the immediate neighbors of a node, it could be extended to include neighbors which are at most a number of links away.

5.1 Neighbor Label Aggregation Methods

- Count Method: The count aggregation method [30] determines the frequency of the neighbors having the same class as the node:

$$\mathbf{r}_N^{count}(u) = \sum_{v \in N(u)} \mathbf{r}(v). \tag{15}$$

  The count method does not consider any uncertainty with the labels or links, neither does it consider the edge weights [30].
- Mode Method: This aggregation method considers the mode of the neighbor labels:

$$\mathbf{r}_N^{mode}(u) = mode_{v \in N(u)} \mathbf{r}(v). \tag{16}$$

- Binary Existence Method: This aggregation method only considers whether a certain label exists among the neighbors or not, it does not take into account the number of occurrences, as count or mode aggregation do. For the jth class, the binary existence of neighbor labels' aggregation is computed as:

$$\mathbf{r}_N^{exist}(u, j) = [\mathbf{r}_N^{count}(u, j) > 0] \tag{17}$$

- Weighted Average Method: The weighted average aggregation method [30] sums the weights of the neighbors of the node belonging to each class and then normalizes it with the sum of the weights of all edges to the neighbors. Similar to the count method, it does not consider uncertainty.

$$\mathbf{r}_N^{wavg}(u) = \frac{1}{Z} \sum_{v \in N(u)} w(u, v) \mathbf{r}(v). \tag{18}$$

  Here $w(u, v) \in \mathscr{R}$ is the weight of the link between nodes u and v, Z is a normalization constant:

$$Z = \sum_{v \in N(u)} w(u, v) \tag{19}$$

Table 2 Accuracy comparison of aggregation methods (ICA, SS) (Cora, Citeseer, WebKB)

	Cora	CiteSeer	WebKB
Count method	75.39 ± 1.02	68.86 ± 1.22	84.75 ± 0.67
Mode method	75.42 ± 0.98	69.79 ± 1.00	81.68 ± 0.43
Binary existence method	71.33 ± 1.65	68.58 ± 0.71	80.32 ± 0.86
Proportion method	76.46 ± 1.31	66.81 ± 0.85	76.45 ± 0.93
Pr. weighted average method	64.43 ± 1.15	66.81 ± 0.85	77.82 ± 0.99

Table 3 Accuracy comparison of aggregation methods (ICA,RS) (Cora, Citeseer, WebKB)

	Cora	CiteSeer	WebKB
Count method	87.78 ± 0.48	77.43 ± 0.67	87.01 ± 1.01
Mode method	87.78 ± 0.59	78.13 ± 0.66	85.17 ± 1.93
Binary existence method	85.19 ± 0.61	77.58 ± 0.78	86.78 ± 1.11
Proportion method	86.48 ± 0.63	77.76 ± 0.89	86.32 ± 1.17
Pr. weighted average method	70.15 ± 0.74	70.76 ± 0.93	86.32 ± 1.04

- Probabilistic Weighted Average Method: This aggregation method is the probabilistic version of the Weighted Average method. It is based on the weighted arithmetic mean of class membership probabilities of neighbors of a node. This method was introduced by Macskassy and Provost and was used as a probabilistic Relational Classifier (PRN) classifier [30].

$$\mathbf{r}_N^{pwavg}(u,c) = \frac{1}{Z} \sum_{v \in N(u), c \in C} w(u,v).P(c|v) \tag{20}$$

  where Z is defined as in Eq. (19) and c denotes a certain class.

In Tables 2 and 3, we show the average test accuracies over ten folds, of using iterative classification algorithm (ICA), which is a method for transductive classification of networked data (see the next section). We used logistic regression as the base classifier, during these experiments. As it can be seen from both tables, the count method outperforms the other methods both in terms of its simplicity and the accuracies obtained.

6 Classification in Social Networks

We consider two types of classification in a social network. The content only classification can be used whether the test nodes are known or not. On the other hand, when the test nodes or some unlabeled nodes are known, then semi-supervised classification algorithms can be used.

6.1 Supervised, Content Only Classification

This model consists of a (learned) model, which uses only the local features of the nodes whose class label will be estimated. The local models can also be used to generate priors for the initial state for the relational learning and collective inference components. They also can be used as one source of evidence during collective inference. These models typically are produced by traditional machine learning methods [21].

6.2 Semi-supervised and Transductive Classification in Social Networks

Since social network data usually come in huge sizes, in addition to labeled instances, there are, usually, a huge number of unlabeled instances. In such cases, it could be possible to use the information other than labels that exist in the unlabeled data, which leads to use of semi-supervised learning algorithms. When the test nodes whose class will need to be predicted are known, then we have a transductive learning scenario.

In contrast to the non-relational (local) model, the relational model use the relations in the network as well as the values of attributes of related entities, even possibly long chains of relations. In relational models, a relational classifier determines the class label or estimates the class conditional probabilities. The relational classifier might combine local features and the labels of neighbors using a naive Bayes model or a logistic regression [21].

In semi-supervised learning [43], the unlabeled instances can be used to monitor the variance of the produced classifiers, to maximize the margin and hence to minimize the complexity [18], to place classifier boundaries around the low density regions between clusters in the data [6]. There are also co-training [5] type algorithms which need different classifiers that are obtained through the use of different type of classifiers, different feature subspaces [41] or set of instances. When the classifiers produced are diverse and accurate enough, co-training may improve the final test accuracy [37]. On the other hand, Cozman and Cohen [8] have shown that unlabeled data can degrade the classification performance when there are discrepancies between modeling assumptions used to build the classifier and the actual model that generates the data. Therefore, both for the general semi-supervised and the transductive learning, the use of unlabeled data is not guaranteed to improve performance.

6.3 Collective Classification

Collective classification methods, which are sometimes also called collective inference methods, are iterative procedures, which classify related instances

simultaneously [30,42]. In collective classification, the content and link information for both training and test data are available. First, based on the available training content, link and label information, models are trained. Then, those models are used to label the test data simultaneously and iteratively where each test sample is labeled based on its neighbors.

Collective classification exploits relational autocorrelation. Relational autocorrelation is a very important property of relational data and is used as a measure of how an attribute for an instance is correlated with the same variable from a related instance [30].

However, sometimes, the advantage of exploiting the relationships can become a disadvantage since it is possible to make incorrect predictions about a particular node which propagates in the network and may lead to incorrect predictions about other nodes. Bilgic and Getoor proposed an acquisition method which learns the cases when a given collective classification algorithm makes mistakes, and suggests label acquisitions to correct those mistakes [4].

Iterative classification algorithm (ICA) and Gibbs sampling algorithm (GS), Mean field relaxation labeling (MF), Loopy belief propagation (LBP), are popular approximate inference algorithms used for collective classification [34]. In this book chapter, we explain the ICA algorithm and use it in our experiments. Iterative classification algorithm (ICA) is a popular and simple approximate collective inference algorithm [21, 34]. Despite its simplicity, ICA was shown to perform as well as the other algorithms such as Gibbs Sampling [33]. Please see [21, 34] for details on the other collective classification algorithms.

6.3.1 Iterative Classification Algorithm (ICA)

To determine the label of a node, Iterative Classification Algorithm (ICA) assumes that all of the neighbors' attributes and labels of that node are already known. Then, it calculates the most likely label with a local classifier which uses node content and neighbors' labels. However, most nodes have neighbors which are not in training data and hence are not labeled, therefore the label assignment on one test instance may affect the label assignment on a related test instance. ICA repeats the labeling process iteratively until all of the label assignments are stabilized. Neighbor label information is summarized using an aggregation operator (Sect. 5.1).

Pseudocode for the ICA algorithm (based on [34]) is given in Algorithm 1. In the pseudo code, $\tilde{\mathbf{r}}(u)$ stands for temporary label assignment of instance u in the test set. $g_{CL}([\mathbf{x}(u)\ \mathbf{r}_N(u)])$ is the base classifier which is first trained on training nodes and their neighbors from the training set. The base classifier uses the estimated labels of the neighbors if they are test nodes. O is a random ordering of test nodes.

As shown above, the ICA algorithm starts with a bootstrapping to assign initial temporary labels to all nodes by using only the content features of the nodes. Then, it starts iterating and updating labels according to the both relational and content features [32].

Algorithm 1: $\tilde{\mathbf{r}}(V_{test}) = \text{ICA}(G, V_{train}, V_{test}, g_{CL}())$

for all $u \in V_{test}$ **do**
  Compute $\tilde{\mathbf{r}}_N(u)$ using only neighbors in V_{train}
  Set $\tilde{\mathbf{r}}(u) \leftarrow g_{CL}([\mathbf{x}(u)\ \tilde{\mathbf{r}}_N(u)])$
end for
repeat
  Generate ordering O over nodes in V_{test}
  for all $u \in O$ **do**
    Compute $\tilde{\mathbf{r}}_N(u)$ using current label assignments to nodes in $N(u)$
    Set $\tilde{\mathbf{r}}(u) \leftarrow g_{CL}([\mathbf{x}(u)\ \tilde{\mathbf{r}}_N(u)])$
  end for
until all labels are stabilized or threshold number of iterations

7 Experiments

In this section, we report classification results on three networked datasets. We evaluate content only (CO), ICA using only a link based classifier (LO), ICA using a content and link based classifier (ICA). We show that whether a method works on a dataset or not is highly dependent on the properties, especially homophily of the dataset. In the experiments, we report results using both snowball and random sampling methods to select the test set. We show that sampling may also affect the results.

7.1 Experimental Setup

We use Cora, Citeseer and WebKb datasets whose details are given in Table 1. Both Cora and Citeseer datasets consist of information on scientific papers. As features, the words that occur at least ten times are used. For each paper, whether or not it contains a specific word, which class it belongs to, which papers it cites and which papers it is cited by are known. The WebKB dataset contains web pages, instead of papers as content. The hyperlinks between web pages are used to produce links between nodes.

We experiment with different classification methods, namely, logistic regression (LR), Support Vector Machine (SVM), Naive Bayes (NB), Bayes Net (BN), k-Nearest Neighbor (kNN, k $=$ 3) for the g_{CO}, g_{LO} and g_{CL} classifiers. For all of the methods Weka implementations with default parameters (unless otherwise noted) have been used.

7.2 Performance of Different Classifiers

First of all, we conducted experiments on datasets using Logistic Regression (LR), Support Vector Machine (SVM), Naive Bayes (NB), Bayes Net (BN), k-Nearest

Table 4 Accuracies on Cora dataset using different classifiers (snowball sampling)

	Acc(CO)	Acc(LO)	Acc(ICA)
Logistic regression	0.61 ± 0.01	0.77 ± 0.01	0.72 ± 0.01
SVM	0.70 ± 0.01	0.64 ± 0.02	0.71 ± 0.01
KNN(k = 1)	0.46 ± 0.01	0.66 ± 0.02	0.52 ± 0.01
KNN(k = 3)	0.47 ± 0.01	0.65 ± 0.01	0.54 ± 0.01
Naive Bayes	0.70 ± 0.01	0.60 ± 0.01	0.76 ± 0.01
Bayes Net	0.70 ± 0.01	0.63 ± 0.02	0.78 ± 0.01
Random forest	0.59 ± 0.01	0.61 ± 0.02	0.66 ± 0.01
J48	0.64 ± 0.01	0.52 ± 0.02	0.51 ± 0.03
LibSVM	0.29 ± 0.01	0.61 ± 0.02	0.72 ± 0.01

Table 5 Accuracies on WebKb dataset using different classifiers (snowball sampling)

	A.(CO)	A.(LO)	A.(ICA)
Logistic regression	0.67 ± 0.04	0.56 ± 0.02	0.59 ± 0.05
SVM	0.83 ± 0.02	0.52 ± 0.04	0.83 ± 0.02
KNN1	0.63 ± 0.02	0.46 ± 0.03	0.64 ± 0.02
KNN3	0.56 ± 0.02	0.45 ± 0.02	0.56 ± 0.02
Naive Bayes	0.75 ± 0.02	0.47 ± 0.02	0.75 ± 0.02
Bayes Net	0.78 ± 0.02	0.50 ± 0.02	0.79 ± 0.02
Random forest	0.68 ± 0.02	0.48 ± 0.02	0.70 ± 0.01
J48	0.69 ± 0.02	0.43 ± 0.03	0.67 ± 0.03
LibSVM	0.60 ± 0.02	0.51 ± 0.02	0.64 ± 0.02

Table 6 Accuracies on Citeseer dataset using different classifiers (snowball sampling)

	Acc(CO)	Acc(LO)	Acc(ICA)
Logistic regression	0.55 ± 0.01	0.59 ± 0.02	0.57 ± 0.01
SVM	0.70 ± 0.01	0.56 ± 0.01	0.71 ± 0.01
KNN(k = 1)	0.42 ± 0.01	0.54 ± 0.01	0.43 ± 0.01
KNN(k = 3)	0.38 ± 0.01	0.54 ± 0.02	0.35 ± 0.01
Naive Bayes	0.69 ± 0.01	0.54 ± 0.02	0.73 ± 0.01
Bayes Net	0.70 ± 0.00	0.44 ± 0.03	0.72 ± 0.01
Random forest	0.57 ± 0.01	0.45 ± 0.02	0.57 ± 0.02
J48	0.59 ± 0.00	0.38 ± 0.02	0.44 ± 0.02
LibSVM	0.19 ± 0.00	0.49 ± 0.01	0.60 ± 0.02

Neighbor (kNN) classifiers. We evaluated the accuracies obtained when content only (CO), link only (LO) classifiers and ICA are used with a specific classification method for each dataset.

Tables 4, 5, and 6 show the accuracies obtained when different classifiers are used on Cora CiteSeer and WebKb datasets when Snowball Sampling is used.

Tables 7, 8, and 9 show the accuracies obtained when different classifiers are used on Cora CiteSeer, and WebKb datasets when Random Sampling is used.

Table 7 Accuracies on Citeseer dataset using different classifiers (random sampling)

	Acc(CO)	Acc(LO)	Acc(ICA)
Logistic regression	0.58 ± 0.02	0.68 ± 0.02	0.61 ± 0.02
SVM	0.75 ± 0.01	0.66 ± 0.02	0.75 ± 0.01
KNN(k = 1)	0.42 ± 0.02	0.66 ± 0.02	0.45 ± 0.02
KNN(k = 3)	0.35 ± 0.02	0.65 ± 0.02	0.37 ± 0.02
Naive Bayes	0.71 ± 0.01	0.59 ± 0.03	0.74 ± 0.01
Bayes Net	0.71 ± 0.01	0.65 ± 0.02	0.77 ± 0.01
Random forest	0.60 ± 0.01	0.64 ± 0.02	0.65 ± 0.01
J48	0.63 ± 0.01	0.61 ± 0.02	0.60 ± 0.02
LibSVM	0.16 ± 0.02	0.66 ± 0.02	0.62 ± 0.02

Table 8 Accuracies on Cora dataset evaluated using different classifiers (random sampling)

	Acc(CO)	Acc(LO)	Acc(ICA)
Logistic regression	0.63 ± 0.01	0.85 ± 0.01	0.72 ± 0.01
SVM	0.73 ± 0.01	0.64 ± 0.02	0.76 ± 0.01
KNN(k = 1)	0.48 ± 0.01	0.80 ± 0.01	0.55 ± 0.01
KNN(k = 3)	0.50 ± 0.01	0.81 ± 0.01	0.57 ± 0.01
Naive Bayes	0.73 ± 0.01	0.70 ± 0.02	0.80 ± 0.01
Bayes Net	0.73 ± 0.01	0.78 ± 0.01	0.86 ± 0.01
Random forest	0.64 ± 0.01	0.77 ± 0.01	0.77 ± 0.01
J48	0.65 ± 0.01	0.73 ± 0.01	0.70 ± 0.01
LibSVM	0.33 ± 0.01	0.80 ± 0.01	0.76 ± 0.01

Table 9 Test accuracy results of WebKb dataset evaluated using different classifiers (random sampling)

	A.(CO)	A.(LO)	A.(ICA)
Logistic regression	0.65 ± 0.05	0.52 ± 0.03	0.55 ± 0.05
SVM	0.81 ± 0.03	0.50 ± 0.03	0.81 ± 0.03
KNN1	0.52 ± 0.04	0.48 ± 0.03	0.52 ± 0.04
KNN3	0.51 ± 0.05	0.47 ± 0.03	0.52 ± 0.05
Naive Bayes	0.74 ± 0.03	0.46 ± 0.05	0.74 ± 0.03
Bayes Net	0.77 ± 0.02	0.50 ± 0.03	0.80 ± 0.02
Random forest	0.67 ± 0.03	0.47 ± 0.03	0.69 ± 0.03
J48	0.68 ± 0.03	0.46 ± 0.03	0.67 ± 0.03
LibSVM	0.58 ± 0.05	0.50 ± 0.03	0.62 ± 0.04

For CO classification, while LR, SVM, BN and NB give similar accuracies, kNN usually gives worse accuracies. On the other hand, for LO all classifiers perform similarly.

For the CO classification, SVM, NB and BN classifiers usually performed better than the others. We think that this is due to the high input dimensionality of the content features. On the other hand, for LO classification LR outperformed the other

methods. Since the LO homophily of Cora dataset is higher than the CiteSeer, the LO accuracies are also higher. For the Cora dataset, instead of using thousands of features in CO classifier, simply using the aggregated class neighbors in LO classifier results in a better accuracy. This is expected since both datasets have high homophily and LO (and ICA) benefits from homophily. For both Cora and Citeseer datasets, BN method gives the best results for ICA and ICA performs better than CO methods. However, again due to high link graph homophily, ICA performs just a little better than LO accuracies.

8 Conclusion

In this chapter, we have given details on how content, link and label information on social network data can be used for classification. We defined important properties of social networked data, which may be used to characterize a networked dataset. We defined a list of aggregation operators, which are used to aggregate the labels of neighbors, so that the classifiers trained have the same number of inputs for each node. We evaluated the accuracies of different classifiers, using only the content, only the link or both the content and the link information.

We have shown that graph properties, especially homophily, play an important role on whether network information would help in classification accuracy or not. The Cora dataset which has high homophily benefits a lot from the use of network information. It is also worthwhile noting that when homophily is high, one can only use the link information and may not need the content information at all.

We experimented with a number of label aggregation methods and found out that the count method, which is one of the simplest aggregation methods, is as good as the other methods.

The networked data does not obey the i.i.d. assumptions which are mostly assumed to hold for content only datasets. The test dataset also may come either randomly among distributed in the network, or they may be concentrated on certain parts of the network. We used random and snowball sampling algorithms to separate the a portion of the all available data as test data. We have shown that, with random sampling, the test accuracies of classifiers that use network information are always better. This is due to the fact that with random partition the nodes in the test partition will naturally have more neighbors from train and validation partitions, which have the actual labels, as opposed to the labels estimated by the classifiers.

Depending on whether content only, link only or content and link information is used and depending on the dataset and the type of content information, we have shown that different types of classifiers, such as SVM, Naive Bayes, Logistic Regression classifier, may perform differently. All the datasets we used in our experiments contained text as content information, therefore, classifiers which were able to deal with high dimensional features, such as SVM, Naive Bayes and Bayes Net performed better for content only or content and link classification. On the other hand, when only link information was used, the logistic regression classifier

performed better. We suggest that using different type of classifiers and content only, link only and content and link classification should be experimented with for a networked dataset.

References

1. Ahmed NK, Neville J, Kompella R (2012) Network sampling: from static to streaming graphs. arXiv preprint arXiv:1211.3412
2. Awan A, Bari H, Yan F, Moksong S, Yang S, Chowdhury S, Cui Q, Yu Z, Purisima EO, Wang E (2007) Regulatory network motifs and hotspots of cancer genes in a mammalian cellular signalling network. IET Syst Biol 1(5):292–297
3. Bernstein AA, Clearwater S, Hill S, Perlich C, Provost F (2002) Discovering knowledge from relational data extracted from business news. In: Proceedings of the workshop on multi-relational data mining at KDD-2002, pp 7–22
4. Bilgic M, Getoor L (2008) Effective label acquisition for collective classification. In: Li Y, Liu B, Sarawagi S (eds) Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining, Las Vegas, 24–27 August 2008. ACM, New York, pp 43–51
5. Blum A, Mitchell T (1998) Combining labeled and unlabeled data with co-training. In: COLT, pp 92–100
6. Chapelle O, Zien A (2005) Semi-supervised classification by low density separation. In: AISTAT 2005
7. Colizza V, Flammini A, Serrano MA, Vespignani A (2006) Detecting rich-club ordering in complex networks. Nat Phys 2(2):110–115
8. Cozman FG, Cohen I (2002) Unlabeled data can degrade classification performance of generative classifiers. In: FLAIRS 2002
9. Cravenand M, DiPasquo D, Freitag D, McCallum A, Mitchell T, Nigam K, Slattery S (1998) Learning to extract symbolic knowledge from the world wide web. In: Proceedings of the fifteenth national conference on artificial intelligence (AAAI), Madison, pp 509–516
10. Dasgupta K, Singh R, Viswanathan B, Chakraborty D, Mukherjea S, Nanavati AA, Joshi A (2008) Social ties and their relevance to churn in mobile telecom networks. In: EDBT08
11. Dorogovtsev SN, Mendes JF (2002). Evolution of networks. Adv Phys 51(4):1079–1187
12. Erdos P, Pach J, Pollack R, Tuza Z (1989) Radius, diameter, and minimum degree. J Comb Theory Ser B 47(1):73–39
13. Fronczak A, Fronczak P, Holyst JA (2004) Average path length in uncorrelated random networks with hidden variables. Phys Rev E 70(5):056110, 1–7
14. Giles CL, Bollacker K, Lawrence S (1998) Citeseer: an automatic citation indexing system. In: ACM digital libraries, pp 89–98
15. Hanneman RA, Riddle M (2005) Introduction to social network methods. University of California, Riverside
16. Hein O, Schwind M, Konig W (2006) Scale-free networks: the impact of fat tailed degree distribution on diffusion and communication processes. Wirtschaftsinformatik 48(4):267–275
17. Jensen D, Neville J (2002) Linkage and autocorrelation cause feature selection bias in relational learning. In: Proceedings of the 19th international conference on machine learning. Morgan Kaufmann, San Francisco, pp 259–266
18. Joachims T (2003) Transductive learning via spectral graph partitioning. In: ICML 2003, pp 200–209
19. Kahng GO, Oh E, Kahng B, Kim D (2003) Betweenness centrality correlation in social networks. Phys Rev E 67:01710–01711

20. Leskovec J, Faloutsos C (2006, August) Sampling from large graphs. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, pp 631–636
21. Macskassy SA, Provost F (2007) Classification in networked data: a toolkit and a univariate case study. J Mach Learn Res 8:935–983
22. McCallum A, Nigam K, Rennie J, Seymore K (2000) Automating the construction of internet portals with machine learning. Inf Retr J 3(2):127–163
23. McDowell L, Gupta KM, Aha DW (2007) Cautious inference in collective classification. In: AAAI. AAAI Press, Menlo Park, pp 596–601
24. Neville J, Jensen D (2008) A bias/variance decomposition for models using collective inference. Mach Learn 73(1):87–106
25. Neville J, Gallagher B, Eliassi-Rad T (2009) Evaluating statistical tests for within-network classifiers of relational data. In: ICDM
26. Newman MEJ (2003) Mixing patterns in networks. Phys Rev E 67(2):026126
27. Newman MEJ (2003) The structure and function of complex networks. SIAM Rev 45(2):167256
28. Perlich C, Huang Z (2005) Relational learning for customer relationship management. In: Proceedings of international workshop on customer relationship management: data mining meets marketing
29. Perlich C, Provost FJ (2006) Distribution-based aggregation for relational learning with identifier attributes. Mach Learn 62(1–2):65–105
30. Preisach C, Schmidt-Thieme L (2008) Ensembles of relational classifiers. Knowl Inf Syst 14(3):249–272
31. Provost F, Perlich C, Macskassy S (2003) Relational learning problems and simple models. In: Proceedings of the IJCAI-2003 workshop on learning statistical models from relational data
32. Sen P, Getoor L (2006) Empirical comparison of approximate inference algorithms for networked data. In: ICML workshop on open problems in statistical relational learning (SRL2006)
33. Sen P, Getoor L (2007) Link-based classification. UM Computer Science Department, Technical Report, CS-TR-4858. University of Maryland
34. Sen P, Namata G, Bilgic M, Getoor L, Gallagher B, Eliassi-Rad T (2008) Collective classification in network data. AI Mag 29(3):93–106
35. Snijders TAB (1992) Estimation on the basis of snowball samples: How to weight? BMS: Bull Sociol Methodol (Bulletin de M thodologie Sociologique) 36(1):59–70
36. Tresp V, Bundschus M, Rettinger A, Huang Y (2008) Towards machine learning on the semantic web. In: Uncertainty reasoning for the semantic web I. Lecture notes in AI. Springer, Berlin
37. Wang W, Zhou ZH (2007) Analyzing co-training style algorithms. In: *Machine Learning: ECML 2007* (pp. 454–465). Springer Berlin Heidelberg
38. Watts DJ, Strogatz SH (1998) Collective dynamics of 'small-world' networks. Nature 393: 440–442
39. Xiang R, Neville J, Rogati M (2010) Modeling relationship strength in online social networks. In: Proceedings of the 19th international conference on World wide web. ACM, New York, pp 981–990
40. Yang Y, Slattery S, Ghani R (2002) A study of approaches to hypertext categorization. J Intell Inf Syst 18(2/3):219–241
41. Yaslan Y, Cataltepe Z (2010) Co-training with relevant random subspaces. Neurocomputing 73(10–12):1652–1661
42. Yonghong T, Tiejun H, Wen G (2006) Latent linkage semantic kernels for collective classification of link data. J Intell Inf Syst 26(3):269–301
43. Zhu X (2007) Semi-supervised learning literature survey. Computer Sciences Technical Report, TR 1530. University of Wisconsin Madison