2015-07

Link prediction in graph construction for supervised and semi-supervised learning

International Joint Conference on Neural Network, 2015, Killarney.
http://www.producao.usp.br/handle/BDPI/49490

Link prediction in graph construction for supervised and semi-supervised learning

Lilian Berton, Jorge Valverde-Rebaza and Alneu de Andrade Lopes
Department of Computer Science
ICMC, University of São Paulo
C.P. 668, CEP 13560-970, São Carlos, SP, Brazil
{lberton,jvalverr,alneu}@icmc.usp.br

***Abstract*—Many real-world domains are relational in nature since they consist of a set of objects related to each other in complex ways. However, there are also flat data sets and if we want to apply graph-based algorithms, it is necessary to construct a graph from this data. This paper aims to: i) increase the exploration of graph-based algorithms and ii) proposes new techniques for graph construction from flat data. Our proposal focuses on constructing graphs using link prediction measures for predicting the existence of links between entities from an initial graph. Starting from a basic graph structure such as a minimum spanning tree, we apply a link prediction measure to add new edges in the graph. The link prediction measures considered here are based on structural similarity of the graph that improves the graph connectivity. We evaluate our proposal for graph construction in supervised and semi-supervised classification and we confirm the graphs achieve better accuracy.**

***Keywords*—Graph construction, Graph-based classification, Link prediction**

I. Introduction

Many social, biological and information systems can be naturally described as networks, where vertices represent entities (individuals or organizations) and links/edges denote relations or interactions between vertices [1]. Networks or graphs are a powerful representation that has been employed in different tasks of machine learning (ML) and data mining (DM) [2]. An important scientific issue regarding network analysis that has attracted increasing attention in recent years is the link prediction. This problem aims to estimate the likelihood of the future existence of a link between two disconnected vertices [3]–[5].

Detection of hidden social relationships, such as friendship suggestion mechanism used by some online social networks, constitutes one of the main application of link prediction. In such case, hidden relationships may consist in existing social ties that have not been established yet in a social network or in social ties missed during the social network evolution [3], [4], [6], [7]. Link prediction has many applications outside the domain of social networks, such as to infer the importance of documents in scientific publications [8] and hypertext [9], [10], classify documents [11], [12], discover genetic or protein-protein interactions [13], etc.

Some data does not have a relational nature, i.e. is flat data. Many ML and DM techniques were proposed to resolve different problems represented as tabular attribute-value data. Recent studies have shown that converting attribute-value data to relational data can help to improve classification accuracy [14], [15]. Furthermore, to apply graph-based methods to attribute-value data is necessary to convert the data into a network. Consequently, the issue of graph construction has received increasing attention in the last years [16]–[19]. Despite many methods for graph construction have been proposed, it is still an open problem.

Considering the fact that link prediction is a mechanism for analyzing the growth and quick changes over time in underlying structures of the networks, it is feasible to think that link prediction can be used as a framework to evolve an initial neighborhood graphs constructed from tabular data. However, few studies have considered this hypothesis. In [20], several methods for generating content graphs that outperformed the existing k-nearest neighbors (kNN) approach using a set of network metrics through a link prediction mechanism is presented. In [21], a graph construction scheme that consider the optimal neighborhood graph as a subgraph of a kNN is presented. The traditional kNN scheme is modified including a mechanism of links selection based on their existence probability and the links connecting different classes are pruned.

In this paper, we propose a novel method for graph construction based on link prediction. First, an initial graph structure over the attribute-value data set is constructed using some traditional graph construction technique. After, some link prediction method is computed with the objective of estimate new links in the graph. Our hypothesis considers that if a network is very sparse, for example when a minimum spanning tree is applied, it misses structural information for the inference algorithms. Alternatively, if a network is very dense, for example when kNN considering $k > 10$ is applied, the excess of edges become noise in the graph. Taking this into consideration, the proposed approach considers a basic graph structure with additional predicted edges, generating a balanced graph structure. We show that our proposal improves the quality of graphs leading to better classification accuracy in supervised and semi-supervised domains.

The remainder of the paper is organized as follows. Section II describes background information on graph construction, classification and the link prediction measures used for our proposal; Section III describes the proposed method, which uses a baseline graph construction method combined with a link prediction measure; Section IV evaluates the performance of the proposed graphs in supervised and semi-supervised scenarios along with some graph analysis. Finally, Section V presents the conclusions and future work.

978-1-4799-1959-8/15/$31.00

II. BACKGROUND

The graph construction methods, the classification algorithms and the link prediction measures considered in this paper are presented in this section.

A. Graph construction

Many data sets are available in tabular flat format (Table I). In this case the line is the i-th example ($i = 1, 2, \ldots, n$) and the column is the j-th attribute ($j = 1, 2, \ldots, m$). The last column Y is the class, $y_i \in \{c_1, c_2, \ldots, c_i\}$. When we have data sets in tabular format is necessary to convert the data into a network to be able to apply a graph-based algorithm. Each instance of the Table I is mapped to a vertex in the graph and similarity or distance between two vertices is represented by an edge.

Table I. TABLE-BASED REPRESENTATION

	X_1	X_2	$\ldots$	X_m	Y
Z_1	x_{11}	x_{12}	$\ldots$	x_{1m}	y_1
Z_2	x_{21}	x_{22}	$\ldots$	x_{2m}	y_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
Z_n	x_{n1}	x_{n2}	$\ldots$	x_{nm}	y_n

Therefore, we have a weighted graph $G(V, E, W)$ built on n data points Z. V is a set of vertices with each v_i representing a data point $x_i \in Z$, $E \subseteq V \times V$ is a set of edges or links connecting adjacent vertices, and $W \in R^{n \times n}$ is a weighted adjacency matrix which measures the similarity among edges, in this way $w(v_i, v_j)$ is the weight of the link between vertices v_i and v_j. The data set Z is contained in a manifold and the graph G reflects the local connectivity and distances of points on the manifold.

For the graph construction we consider the k-nearest neighbors (kNN) [22]. In this approach, each vertex considers its k neighbors by a similarity function and instantiates k undirected edges between itself and these neighbors. There are also the Mutual kNN graphs (MkNN) in which there is a connection between two vertices only if the rule of the neighborhood is fulfilled by both vertices. Another way to estimate a basic graph is applying Maximum/Minimum Spanning Tree (MaxST/MinST). MaxST is a spanning tree of a weighted graph having maximum weight and MinST is the spanning tree with lowest total weight [23].

B. Semi-supervised classification

In semi-supervised learning (SSL) an iterative process is performed. At every step, each vertex is assigned to the class to which the maximum number of its neighbors belong to. Considering V as a set of vertices defined in R^m, F is a function that associates a feature vector to each $v \in V$ and c_i is a set of vertices that belongs to the i^{th} class.

The iterative process performed by label propagation updated the classification score F minimizing a cost function Q. Let $\mathcal{F}$ denote the set of $n \times c$ matrices, the classifying function is given by Eq. 1.

$$F^* = \arg\min_{F \in \mathcal{F}} Q(F) \tag{1}$$

The usual graph based SSL techniques include the Mincut [24], the Gaussian Fields and Harmonic Functions [25], the Local and Global Consistency [26], the Manifold Regularization or Laplacian Support Vector Machine [27], etc. Here we apply the popular Local and Global Consistency (LGC) method [17], [19].

C. Supervised classification

In supervised learning, classification algorithms analyze the training data and produce an inferred function that can be used for classifying new examples. Due to the graph data conveys relational information, which imply dependency relations between instances, is necessary use strictly relational classifiers. Relational classifiers require to a graph G with known labels for some of the vertices to predict the labels of the remaining vertices. We considered three relational classifiers: relational neighbor (*prn*), network-only Bayes (*no-Bayes*) and network-only link-based (*no-lb*).

The *prn* classifier estimates class membership probabilities by assuming that the label of a node depends only on its immediate neighbors and that linked nodes tend to belong to the same class [28]. The *no-Bayes* classifier employs multinomial naïve Bayesian classification based on the classes of the neighborhood of each vertex [14]. Furthermore, these both relational classifiers use the relaxation label as a collective inference method.

The *no-lb* classifier creates a feature vector for a vertex by aggregating the labels of its neighborhood and then use logistic regression to build a discriminative model based on those feature vectors [29]. From *no-lb* classifier three aggregation methods have been considered: binary-link (*no-lb-binary*), mode-link (*no-lb-mode*) and count-link (*no-lb-count*). All the *no-lb* aggregations use the iterative classification as a collective inference method.

D. Link prediction

Link prediction (LP) addresses the problem of predicting the existence of missing relations or new ones [3], [5]. Many methods for link prediction are based on similarity between vertices since similar vertices likely share the same relations (links). When the similarity between vertices is based solely on network structure, it is called structural similarity [5]. Structural similarity measures can be classified in different ways, such as the based on local or global information.

Frequently used local measures are: Common Neighbors, Jaccard coefficient, Adamic Adar, Resource Allocation and Preferential Attachment measures. The most used global measures are: Katz, Rooted PageRank and SimRank [4]. Furthermore, sometimes the presence of weighted links is fundamental to improve the link prediction accuracy. Consequently, weighted measures were proposed as variants from some local measures [30], [31]. In this work we consider just one measure from each type of structural similarity measures. From local measures we consider Common Neighbors and its corresponding weighted variant, referred to as Weighted Common Neighbors. From global measures we consider Katz.

For each pair of disconnected vertices v_i and v_j and using any link prediction measure, is assigned a score s_{v_i,v_j}. Then,

from all the disconnected pairs of vertices is produced a ranked list in decreasing order of scores. Considering that, let $\Gamma(v_i)$ denote the set of neighbors of v_i, the formal definitions of link prediction measures considered in this paper are described as follows.

Common Neighbors (c): Two vertices, v_i and v_j, are more likely to be connected if they have more common neighbors. Thus, this measure refers to the size of the set of all common neighbors of both v_i and v_j according to Eq. 2 [32].

$$s^c_{v_i,v_j} = |\Gamma(v_i) \cap \Gamma(v_j)| \tag{2}$$

Weighted Common Neighbors (w): The weighted variant of Common Neighbors measure is defined by Eq. 3 [31].

$$s^w_{v_i,v_j} = \sum_{v_k \in \Gamma(v_i) \cap \Gamma(v_j)} w(v_i, v_k) + w(v_k, v_j) \tag{3}$$

Katz (k): This measure is based on the ensemble of all paths, which directly sums over the collection of paths and is exponentially damped by length to give the shorter paths more weights according to Eq. 4 [33].

$$s^k_{v_i,v_j} = \sum_{l=1}^{\infty} \beta^l \cdot |paths^{\langle l \rangle}_{v_i,v_j}| = \beta A_{v_i,v_j} + \beta^2 (A^2)_{v_i,v_j} + \dots \tag{4}$$

where $paths^{\langle l \rangle}_{v_i,v_j}$ is the set of all paths with length l connecting v_i and v_j, A is the adjacency matrix, and β is a free parameter (i.e., the damping factor) controlling the path weights. A very small β yields a measurement close to common neighbors measure, because the long paths contribute very little. To optimize the computational cost is possible calculates a matrix version as Eq. 5.

$$S^k_{v_i,v_j} = (I - \beta A)^{-1} - I \tag{5}$$

Note that, β must be lower than the reciprocal of the largest eigenvalue of matrix A to ensure the convergence of Eq. 4. In the experiments β and l are set as 0.05 and 5, respectively.

III. Graph construction based on link prediction

As our goal is to predict new links we must consider links that already exist. Hence, an initial graph structure is necessary that should be as sparse as possible. Sparse graphs minimize the application of the methods of link prediction and maintain the sparsity even with the addition of new edges.

A. LP graph construction

Minimum/maximum spanning tree (MinST/MaxST) or k-nearest neighbors (kNN), with small k values, can be used to obtain an initial graph structure. From this initial graph, a set of potential links among disconnected vertices is predicted. Here, we used three structural similarity-based algorithms considering different categories: a local measure (Common Neighbors), a weighted measure (Weighted Common Neighbors) and a global measure (Katz). Furthermore, for our purpose, is possible to apply any link prediction measure.

To predict new links, each pair of disconnected vertices v_i and v_j is assigned a score $s_{v_i v_j}$, which is directly defined as the similarity between v_i and v_j. All non-observed links are ranked according to their scores, and the links connecting more similar nodes are supposed to be of higher existence likelihoods. A percentage of these ranked links, for example the top 10%, can be considered.

An example of the graph construction schema is shown in Figure 1. From a flat data set an initial graph structure (MinST graph) are constructed. Then, a link prediction measure (Katz) considering a percentage of predicted edges (top 30%) is applied generating a final graph that contains the initial MinST graph and the predicted edges. The final graph can be used for any learning task, such as classification.

Algorithm 1 shows the steps to construct graphs following our proposal, referred as LP graphs. The input are a flat data set X and the percentage of links to be considered, $percentageL$. The output is a graph structure G. Lines 1 to 4 initialize the structures. Lines 8 to 14 create an initial graph structure. Here is possible apply any graph construction technique such as kNN, MinST, and others. Finally, lines 15 to 22 apply a link prediction measure on the initial graph to compute the possible new edges. Here is possible apply any link prediction measure such as, Common Neighbors, Weighted Common Neighbors, Katz, and others. A percentage of these new edges are considered in the final graph.

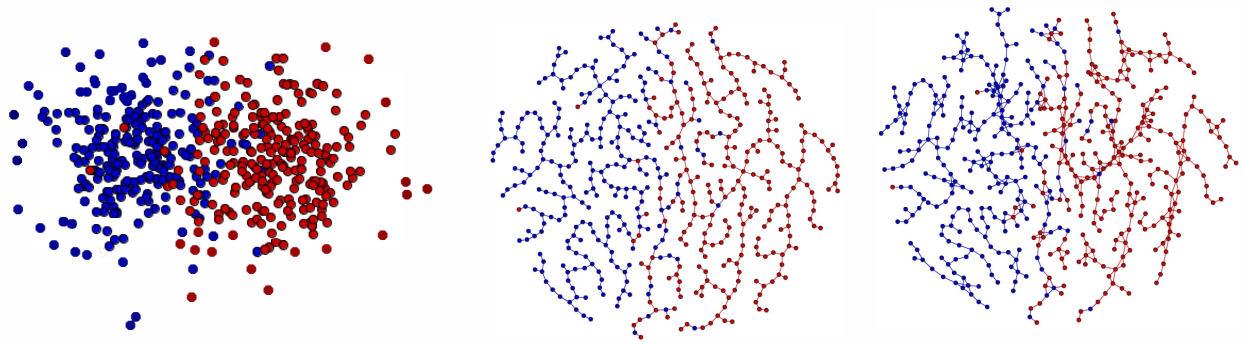
Algorithm 1: LP graph construction

```
   Input: X, percentageL
   Output: G
1  begin
2      V ← create a set of vertices from X;
3      E, W ← ∅;
4      G ← (V, E, W);
5      SearchGraph(G);
6      SearchLP(G, percentageL);
7  end
8  Procedure SearchGraph(G)
9      begin
10         for vertex v_i ∈ G do
11             N ← getNeighbors(v_i);
12             E ← getEdges(N);
13         end
14     end
15 Procedure SearchLP(G, percentageL)
16     begin
17         for vertex v_i ∈ G do
18             LP ← getPredictedLinks(G);
19         end
20         E* ← getTopLinks(percentageL, LP);
21         E ← E* ∪ E;
22     end
```

B. Complexity analysis

In this paper we apply kNN, MkNN, MinST and MaxST methods to compute the initial graph structure. In the following, we show the time complexity for kNN and MinST methods, since MkNN and MaxST have a similar time.

To compute the kNN, a static Kd-tree must be built from n samples taking $O(n \log n)$ if a linear median-finding algorithm



(a) (b) (c)

Figure 1. LP construction steps: (a) Gaussian data set. (b) MinST graph. (c) MinST+LP(Katz) considering the top 30% of predicted links.

is used [23]. Finding one nearest neighbor in a balanced Kd-tree with randomly distributed points takes $O(\log n)$. Searching k nearest neighbor take $O(k \log n)$. Coming up with about $O(nk \log n)$ to search over n instances.

To compute the MinST, a fully connect graph must be built initially. As this is a basic operation, the time is constant (for instance, create a matrix $A^{n \times n}$ full of 1). Then, Kruskal's or Prim's algorithm can be applied to find a minimum spanning tree. Both of these algorithms explore the given graph, starting from an arbitrary vertex v_i, by looping through the neighbors of the vertices adding each unexplored neighbor to a data structure to be explored later. Then, a minimum spanning tree can be formed by connecting each vertex that minimizes the sum of the weights of their edges. These algorithms take $O(|E| \log |E|)$ [23], where $|E|$ is the number of edges in the graph. As the number of edges in a complete undirected graph is $\frac{n(n-1)}{2}$ the final time complexity is $O(n^2 \log n)$.

Subsequently, a link prediction measure must be calculated. In this case, a different measure can result in different time complexity. Considering the Common Neighbors measure, for each jth neighbor of an element v_i, it is necessary to check which neighbors they have in common. If we have a kNN list, for each vertice it is necessary to access its k neighbors k times resulting in $O(nk^2)$. The same occurs with the Weighted Common Neighbors measure. The Katz measure is usually cubic if we consider the matricial Equation 5.

The final complexity of our proposal results in the sum of the complexity of the initial graph construction and the link prediction measure chosen, for instace, it will be $O(nk \log n)+ O(nk^2)$ if kNN graph and Common Neighbors are used.

IV. Experimental results

This section presents the classification results in semi-supervised and supervised domains, as well as an analysis of generated graphs.

A. Semi-supervised results

1) Data sets and experimental setup: The semi-supervised experiments were carried out on five data sets (Table II) frequently used in SSL literature and available in [34].

Table II. Data sets descriptions for SSL classification

Data set	# Instances	# Attributes	# Classes
g241c	1500	241	2
g241n	1500	241	2
Digit_1	1500	241	2
USPS	1500	241	2
COIL_2	1500	241	2

Principal Component Analysis (PCA) was applied to the data sets reducing the dimensions to 50 since in high-dimensional data the distance to the nearest neighbor approaches the distance to the farthest neighbor [35], which degenerates the quality of the graph. Then, the experiments were running using 10 and 100 labeled vertices randomly selected from all the points to test the effectiveness of the graph in extreme cases. For the graph construction we apply MinST, MaxST, kNN and MkNN with $1 \leq k \leq 20$, and the LP graphs (our proposal) considering the same methods combined with a LP measure: MinST+LP, MaxST+LP, kNN+LP and MkNN+LP with $1 \leq k \leq 5$. To generate the weighted graph W the binary weighting approach was applied. The algorithm used for the label inference task was the LGC and the average accuracy of 30 runs was used as evaluation.

2) Classification results: The semi-supervised classification results are shown in Table III. In the first column are the data sets considered, in the second column are the methods for graph construction and in the third and fourth columns, respectively, are the classification results using 10 and 100 labeled vertices, besides a parameter (in brackets). For kNN and MkNN this parameter is the number of neighbors k. For our proposal the parameter is the value of k $(1, \ldots, 5)$, the method of LP used (c, w or k) and the percentage of top links selected $(10, \ldots, 100)$. The highest accuracy for each labeled configuration in each data set are in bold. The LP graphs improve the accuracy specially when few labeled points are considered, in this case less than 1%.

From Table III, the Nemenyi post-hoc test [36] was executed to verify the possibility of detecting differences among the graph construction methods. The results are shown in Figure 2. On the top of the diagrams is the critical difference (CD) and in the axis are plotted the average ranks of the evaluated techniques, where the lowest (best) ranks are in the left side. When the methods analyzed have no significant difference, they are connected by a black line in the diagram.

Table III. SEMI-SUPERVISED CLASSIFICATION RESULTS

BD	Method	LGC(10)	LGC(100)
g241c	kNN	0.544 ± 0.066 (1)	0.581 ± 0.026 (1)
	MkNN	0.515 ± 0.028 (4)	0.607 ± 0.082 (2)
	MinST	0.499 ± 0.011	0.50 ± 0.035
	MaxST	0.499 ± 0.009	0.501 ± 0.028
	kNN+LP	**0.581 ± 0.075 (1, c-10)**	0.597 ± 0.022 (1, c-40)
	MkNN+LP	0.574 ± 0.156 (2, k-50)	**0.622 ± 0.084 (2, c-40)**
	MinST+LP	0.535 ± 0.039 (c-40)	0.588 ± 0.021 (c-40)
	MaxST+LP	0.51 ± 0.016 (c-60)	0.536 ± 0.017 (c-70)
g241n	kNN	0.52 ± 0.03 (4)	0.573 ± 0.065 (4)
	MkNN	0.515 ± 0.024 (11)	0.571 ± 0.052 (12)
	MinST	0.499 ± 0.014	0.502 ± 0.03
	MaxST	0.499 ± 0.019	0.50 ± 0.043
	kNN+LP	**0.524 ± 0.03 (4, c-10)**	0.57 ± 0.023 (1, c-20)
	MkNN+LP	0.508 ± 0.131 (5, c-10)	0.538 ± 0.014 (5, c-30)
	MinST+LP	0.51 ± 0.161 (k-70)	**0.574 ± 0.02 (c-10)**
	MaxST+LP	0.502 ± 0.061 (c-60)	0.504 ± 0.01 (c-90)
Digit_1	kNN	0.894 ± 0.048 (3)	0.971 ± 0.097 (4)
	MkNN	0.89 ± 0.097 (7)	**0.972 ± 0.074 (10)**
	MinST	0.50 ± 0.01	0.501 ± 0.011
	MaxST	0.503 ± 0.099	0.499 ± 0.01
	kNN+LP	0.899 ± 0.06 (3, k-10)	0.966 ± 0.013 (5, w-30)
	MkNN+LP	0.905 ± 0.046 (5, c-40)	0.948 ± 0.01 (5, w-60)
	MinST+LP	**0.917 ± 0.042 (w-40)**	0.94 ± 0.018 (w-50)
	MaxST+LP	0.592 ± 0.077 (c-90)	0.716 ± 0.031 (w-80)
USPS	kNN	0.838 ± 0.03 (3)	0.892 ± 0.027 (2)
	MkNN	0.841 ± 0.063 (12)	0.913 ± 0.027 (9)
	MinST	0.709 ± 0.021	0.74 ± 0.018
	MaxST	0.709 ± 0.021	0.65 ± 0.026
	kNN+LP	0.843 (3, k-10) ± 0.034	0.89 ± 0.032 (2, c-20)
	MkNN+LP	0.799 ± 0.049 (5, w-80)	0.90 ± 0.014 (5, w-60)
	MinST+LP	**0.845 ± 0.063 (k-20)**	**0.933 ± 0.014 (w-50)**
	MaxST+LP	0.792 ± 0.015 (c-80)	0.798 ± 0.057 (c-70)
COIL_2	kNN	0.653 ± 0.045 (3)	**0.971 ± 0.017 (3)**
	MkNN	0.656 ± 0.039 (7)	0.965 ± 0.015 (7)
	MinST	0.499 ± 0.01	0.499 ± 0.031
	MaxST	0.499 ± 0.01	0.498 ± 0.029
	kNN+LP	**0.682 ± 0.062 (5, k-60)**	0.956 ± 0.019 (3, w-80)
	MkNN+LP	0.65 ± 0.038 (5, k-90)	0.953 ± 0.019 (5, c-30)
	MinST+LP	0.647 ± 0.042 (w-80)	0.898 ± 0.028 (w-60)
	MaxST+LP	0.526 ± 0.039 (c-90)	0.567 ± 0.038 (c-100)

According to the Nemenyi statistics, the critical value for comparing the average-ranking of two different algorithms at 95 percentile is 3.32. Note that all methods improved the accuracy when combined with LP measures except MkNN.

There exist statistical difference among all methods against MaxST and MinST, except MaxST+LP. This proves our hypothesis that very sparse graphs are not good for classification. Therefore, when MinST and MaxST are combined with LP measures, they had their accuracy increased. About kNN graphs, the best accuracy already was achieved with small k values, nevertheless kNN+LP increases the accuracy a little. If we had considered high values of k, kNN method would not achieve good results. For MkNN+LP we test values for k smaller than 5 and MkNN is better with high values, as shown by the parameter k in Table III. This way MkNN+LP could not increase the accuracy for MkNN. In future work is recommended to consider higher k values for MkNN+LP.

B. Supervised results

1) Data sets and experimental setup: The supervised experiments were carried out on seven data sets (Table IV) where artificial data sets (Gaussians 3 and 5) and benchmark data sets (Wine, Ecoli, Customers, Cancer and Blood) from UCI Machine Learning Repository [37] were considered.

For Ecoli and Cancer, the first attribute related to a name or id number were removed. For Cancer the instances with missing values were also removed. For the graph construction we apply MinST, MaxST, kNN and MkNN with $1 \leq k \leq 20$, and the LP graphs (our proposal) considering the same methods combined with a LP measure: MinST+LP, MaxST+LP, kNN+LP and MkNN+LP with $1 \leq k \leq 3$. To generate the weighted graph W the opposite of Euclidean Distance (ED) was applied (lower ED higher similarity among the points). The algorithms used for the classification were: nobayes, nolb-lr-binary, nolb-lr-count, nolb-lr-mode, prn. The accuracy of 10-fold cross validation was used as evaluation.

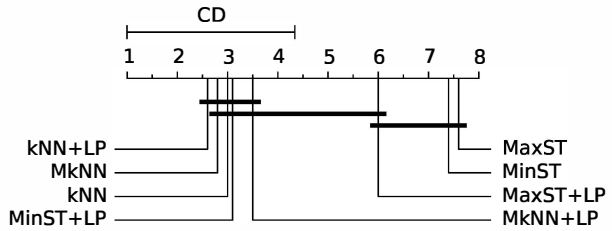


Figure 2. Nemenyi post-hoc test for semi-supervised classification.

Table IV. DATA SETS DESCRIPTIONS FOR SUPERVISED CLASSIFICATION

Data set	# Instances	# Attributes	# Classes
Wine	178	13	3
Ecoli	336	8	8
Customers	440	8	2
Cancer	699	10	2
Blood	748	5	2
Gaussians3	500	2	2
Gaussians5	500	2	2

2) Classification results: The supervised classification results are shown in Table V. In the first column are the data sets considered, in the second column are the methods for graph construction and in the following columns are the classification results together with a parameter (in brackets). For kNN and MkNN this parameter is the number of neighbors k. For the proposed LP variants the parameter is k (1, 2 or 3), the method of LP used (c, w or k) and the percentage of top links selected $(10, \ldots, 100)$. The highest accuracy for each classifier in each data set are in bold. In most cases the LP graph proposed achieve better accuracy.

The Nemenyi post-hoc test was also executed to verify the possibility of detecting differences among the graph construction methods from results of Table V. The best results achieved for each graph construction method in each data set were selected. Figure 3 shows the comparison of all graph construction techniques evaluated. According to the Nemenyi statistics, the critical value for comparing the average-ranking of two different algorithms at 95 percentile is 3.97. The results have a similar behavior with semi-supervised results: i) MinST+LP and MaxST+LP increase the accuracy of MinST and MaxST; ii) MkNN+LP could not increase the accuracy for MkNN, as we consider small values of k for MkNN+LP; iii) kNN+LP increase a little the accuracy of kNN. This confirm that LP graphs have strategic links that improves the connectivity and consequently the classification.

C. Analysis of generated graphs

To know how sensitive graph generation is to input parameters the distribution of them were plotted. In this case,

Table V. SUPERVISED CLASSIFICATION RESULTS

DB	Method	*no-Bayes*	*nolb-lr-binary*	*nolb-lr-count*	*nolb-lr-mode*	*prn*
Wine	kNN	0.353 + 0.071 (11)	0.663 + 0.101 (1)	0.719 + 0.144 (18)	0.731 + 0.130 (2)	0.647 + 0.108 (1)
	MkNN	0.310 + 0.108 (15)	0.618 + 0.103 (2)	0.713 + 0.068 (18)	0.686 + 0.140 (9)	0.579 + 0.133 (2)
	MinST	0.237 + 0.088	0.545 + 0.116	0.715 + 0.101	0.724 + 0.114	0.623 + 0.123
	MaxST	**0.620 + 0.119**	0.613 + 0.085	0.600 + 0.155	0.398 + 0.160	0.307 + 0.050
	kNN+LP	0.410 + 0.062 (1, c-50)	0.675 + 0.081 (1, c-10)	**0.749 + 0.131 (1, c-50)**	**0.760 + 0.010 (2, c-40)**	**0.714 + 0.101 (1, k-100)**
	MkNN+LP	0.427 + 0.120 (1, w-100)	0.664 + 0.139 (1, k -50)	0.670 + 0.103 (2, k-50)	0.698 + 0.144 (2, c-60)	0.646 + 134 (2, c-50)
	MinST+LP	0.348 + 0.154 (w-80)	0.647 + 0.115 (c-10)	0.742 + 0.078 (k-90)	0.749 + 0.148 (k-70)	0.670 + 1.159 (c-30)
	MaxST+LP	0.433 + 0.091 (c-10)	**0.761 + 0.203 (c-10)**	0.695 + 0.115 (w-70)	0.668 + 0.120 (w-80)	0.350 + 0.112 (w-50)
Ecoli	kNN	0.125 ± 0.060 (1)	0.684 ± 0.080 (1)	**0.837 ± 0.070 (17)**	**0.848 ± 0.063 (7)**	0.651 ± 0.105 (1)
	MkNN	0.289 ± 0.055 (1)	0.667 ± 0.089 (4)	0.822 ± 0.064 (12)	0.839 ± 0.036 (11)	0.524 ± 0.080 (4)
	MinST	0.098 ± 0.032	0.670 ± 0.081	0.726 ± 0.064	0.732 ± 0.064	0.565 ± 0.103
	MaxST	0.155 ± 0.090	0.545 ± 0.128	0.538 ± 0.038	0.560 ± 0.123	0.392 ± 0.056
	kNN+LP	0.143 ± 0.044 (1, k-60)	0.753 ± 0.079 (1, w-40)	0.801 ± 0.061 (2, w-40)	0.825 ± 0.044 (3, k-80)	**0.667 ± 0.087 (1, c-20)**
	MkNN+LP	**0.331 ± 0.090 (1, k-30)**	0.634 ± 0.066 (2, w-90)	0.714 ± 0.072 (3, k-10)	0.712 ± 0.090 (3, k-60)	0.530 ± 0.067 (3, k-40)
	MinST+LP	0.143 ± 0.070 (c-60)	**0.690 ± 0.112 (w-50)**	0.806 ± 0.055 (w-100)	0.810 ± 0.066 (w-70)	0.589 ± 0.093 (c-80)
	MaxST+LP	0.152 ± 0.128 (c-70)	0.607 ± 0.185 (k-40)	0.661 ± 0.077 (w-20)	0.664 ± 0.072 (w-100)	0.416 ± 0.052 (w-60)
Customers	kNN	0.543 ± 0.068 (13)	0.782 ± 0.042 (1)	0.870 ± 0.043 (3)	0.866 ± 0.035 (3)	0.789 ± 0.072 (1)
	MkNN	0.614 ± 0.070 (1)	0.807 ± 0.056 (2)	0.884 ± 0.046 (17)	0.884 ± 0.038 (17)	0.748 ± 0.045 (4)
	MinST	0.470 ± 0.089	0.745 ± 0.037	0.859 ± 0.035	0.875 ± 0.036	0.780 ± 0.048
	MaxST	0.509 ± 0.086	0.675 ± 0.042	0.677 ± 0.099	0.677 ± 0.063	0.411 ± 0.121
	kNN+LP	0.555 ± 0.056 (1, c-20)	**0.811 ± 0.057 (1, w-50)**	0.873 ± 0.022 (2, c-50)	0.875 ± 0.053 (3, c-30)	0.807 ± 0.048 (1, c-70)
	MkNN+LP	**0.639 ± 0.048 (1, c-90)**	**0.811 ± 0.056 (2, c-20)**	0.852 ± 0.050 (3, c-40)	0.843 ± 0.045 (3, c-70)	0.791 ± 0.067 (2, k-30)
	MinST+LP	0.541 ± 0.063 (c-80)	0.768 ± 0.084 (k-10)	**0.889 ± 0.020 (w-90)**	**0.886 ± 0.034 (w-90)**	**0.810 ± 0.069 (k-90)**
	MaxST+LP	0.580 ± 0.089 (k-10)	0.709 ± 0.095 (c-30)	0.730 ± 0.072 (w-60)	0.677 ± 0.090 (w-80)	0.523 ± 0.064 (w-30)
Cancer	kNN	0.526 ± 0.070 (17)	0.856 ± 0.067 (1)	0.972 ± 0.019 (5)	**0.974 ± 0.020 (8)**	0.884 ± 0.043 (1)
	MkNN	0.654 ± 0.064 (2)	0.858 ± 0.060 (4)	0.962 ± 0.027 (12)	0.950 ± 0.036 (20)	0.755 ± 0.041 (5)
	MinST	0.475 ± 0.045	0.877 ± 0.058	0.953 ± 0.021	0.957 ± 0.030	0.880 ± 0.031
	MaxST	0.562 ± 0.094	0.789 ± 0.160	0.837 ± 0.058	0.831 ± 0.056	0.396 ± 0.088
	kNN+LP	0.556 ± 0.047 (1, w-70)	**0.893 ± 0.069 (1, w-70)**	**0.974 ± 0.018 (3, w-70)**	0.972 ± 0.028 (3, w-90)	**0.896 ± 0.039 (1, k-50)**
	MkNN+LP	**0.666 ± 0.049 (2, w-50)**	0.856 ± 0.047 (3, k-40)	0.908 ± 0.040 (3, c-90)	0.843 ± 0.056 (3, k-10)	0.746 ± 0.051 (3, c-20)
	MinST+LP	0.591 ± 0.070 (w-20)	0.839 ± 0.115 (k-10)	0.969 ± 0.019 (w-100)	0.971 ± 0.026 (w-100)	0.894 ± 0.052 (k-50)
	MaxST+LP	0.613 ± 0.102 (c-10)	0.792 ± 0.033 (k-60)	0.930 ± 0.032 (w-80)	0.943 ± 0.021 (w-40)	0.531 ± 0.081 (c-100)
Blood	kNN	0.533 ± 0.056 (17)	0.765 ± 0.064 (4)	**0.777 ± 0.049 (4)**	**0.774 ± 0.061 (16)**	0.660 ± 0.051 (2)
	MkNN	0.656 ± 0.052 (1)	0.762 ± 0.059 (4)	0.773 ± 0.056 (5)	0.762 ± 0.058 (1)	0.614 ± 0.044 (4)
	MinST	0.509 ± 0.071	0.762 ± 0.043	0.763 ± 0.031	0.762 ± 0.047	0.603 ± 0.070
	MaxST	0.535 ± 0.093	0.714 ± 0.174	0.769 ± 0.046	0.767 ± 0.051	0.368 ± 0.103
	kNN+LP	0.548 ± 0.072 (1, c-50)	0.769 ± 0.053 (3, k-60)	**0.777 ± 0.038 (3, c-90)**	0.762 ± 0.081 (3, w-10)	**0.666 ± 0.045 (2, c-60)**
	MkNN+LP	**0.680 ± 0.063 (1, w-60)**	0.762 ± 0.051 (3, k-80)	0.775 ± 0.037 (3, c-60)	0.762 ± 0.061 (1, w-30)	0.641 ± 0.073 (2, w-40)
	MinST+LP	0.524 ± 0.063 (k-10)	0.766 ± 0.045 (w-100)	0.770 ± 0.065 (c-40)	0.762 ± 0.068 (k-90)	0.636 ± 0.063 (w-60)
	MaxST+LP	0.549 ± 0.114 (k-40)	**0.774 ± 0.040 (c-50)**	0.769 ± 0.043 (k-80)	0.769 ± 0.045 (k-70)	0.535 ± 0.048 (w-90)
Gaussians3	kNN	0.546 ± 0.061 (9)	0.834 ± 0.057 (1)	**0.916 ± 0.023 (18)**	**0.914 ± 0.032 (6)**	0.828 ± 0.057 (1)
	MkNN	0.548 ± 0.066 (17)	0.774 ± 0.088 (3)	**0.916 ± 0.044 (19)**	**0.914 ± 0.028 (15)**	0.788 ± 0.058 (3)
	MinST	0.488 ± 0.078	0.816 ± 0.066	0.894 ± 0.034	0.880 ± 0.028	0.824 ± 0.029
	MaxST	0.522 ± 0.092	0.818 ± 0.153	0.902 ± 0.030	0.902 ± 0.036	0.440 ± 0.113
	kNN+LP	0.552 ± 0.085 (1, c-100)	0.848 ± 0.069 (1, c-50)	**0.916 ± 0.036 (3, k-50)**	0.912 ± 0.030 (3, w-70)	**0.844 ± 0.042 (1, k-80)**
	MkNN+LP	0.542 ± 0.089 (2, k-30)	0.804 ± 0.064 (3, w-20)	0.870 ± 0.030 (3, w-100)	0.890 ± 0.040 (3, w-40)	0.810 ± 0.066 (3, c-50)
	MinST+LP	**0.554 ± 0.092 (k-80)**	0.840 ± 0.041 (c-30)	0.904 ± 0.045 (k-60)	0.908 ± 0.039 (c-70)	0.832 ± 0.030 (c-20)
	MaxST+LP	0.538 ± 0.111 (c-30)	**0.884 ± 0.077 (c-10)**	0.902 ± 0.042 (c-20)	0.902 ± 0.404 (c-40)	0.538 ± 0.086 (w-70)
Gaussians5	kNN	0.518 ± 0.090 (11)	0.916 ± 0.048 (1)	**0.988 ± 0.010 (7)**	**0.988 ± 0.017 (7)**	0.934 ± 0.035 (1)
	MkNN	0.560 ± 0.075 (12)	0.866 ± 0.084 (3)	0.982 ± 0.026 (19)	0.986 ± 0.016 (19)	0.852 ± 0.046 (3)
	MinST	0.520 ± 0.076	0.878 ± 0.090	0.972 ± 0.019	0.978 ± 0.024	0.868 ± 0.067
	MaxST	0.534 ± 0.140	**0.938 ± 0.106**	0.986 ± 0.021	0.986 ± 0.016	0.358 ± 0.090
	kNN+LP	0.558 ± 0.080 (1, w-100)	0.922 ± 0.048 (1, c-40)	**0.988 ± 0.014 (3, c-80)**	**0.988 ± 0.014 (3, c-10)**	**0.944 ± 0.030 (1, c-20)**
	MkNN+LP	0.532 ± 0.072 (2, k-70)	0.894 ± 0.055 (3, c-20)	0.966 ± 0.031 (3, k-20)	0.966 ± 0.021 (3, k-10)	0.902 ± 0.037 (3, c-60)
	MinST+LP	0.548 ± 0.084 (c-70)	0.898 ± 0.067 (c-20)	0.984 ± 0.023 (w-70)	0.984 ± 0.013 (c-20)	0.914 ± 0.039 (c-40)
	MaxST+LP	**0.570 ± 0.124 (k-100)**	0.926 ± 0.172 (c-10)	0.986 ± 0.017 (c-10)	0.986 ± 0.013 (c-10)	0.548 ± 0.082 (w-40)

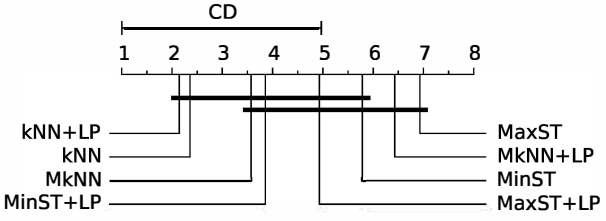


Figure 3. Nemenyi post-hoc test for supervised classification.

they are k (for kNN and MkNN graphs) and top percentage of links (for LP graph variants). Figure 4 shows the number of graphs whose best classification results were achieved by $k = 1, \ldots, 10$ (in the 10th position we consider $k > 10$), and the number of LP graphs variants whose best classification results were achieved by top percentage of links = $10, \ldots, 100$. We observe most graphs use k values equal 1 for kNN and k values bigger than 10 for MkNN. The percentage of links is equal distributed.

Figures 5 and 6 show the average degree and the clustering coefficient for the graphs generated from Gaussians3 data set.

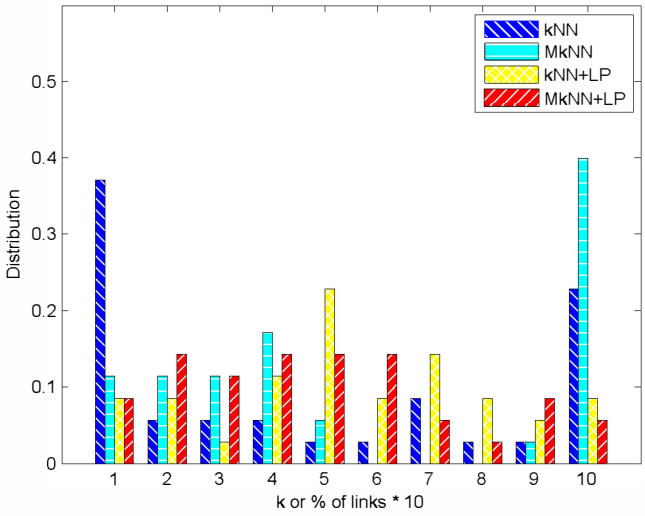


Figure 4. Distribution of parameters k and top percentage of links used for the graph construction methods in the supervised classification.

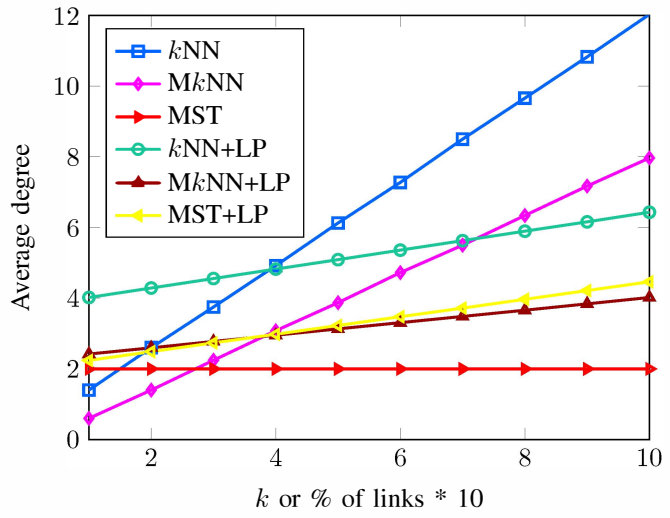


Figure 5. Average degree for kNN, MkNN, MST and LP versions: kNN+LP, MkNN+LP, MST+LP applied to Gaussians3 data set. LP versions use $k = 3$ and the common neighbors measure.

In both of them, MinST and MaxST are referred as MST.

As the value of k increases, the average degree for kNN increases almost linearly and the network becomes dense. For this reason kNN with small values of k are better for classification. MkNN for small values of k are very sparse and the network can be unconnected. For this reason MkNN achieves better results with high k values. MST does not depend on the k and remains with a constant low degree. The average degree remains almost constant for the LP graphs variants (kNN+LP, MkNN+LP, MST+LP). It means our proposal generateS sparse graphs independent of the parameter percentage of links, which is better for classification since a lot of edges are like noise in the graph.

To understand the network topology we plot the clustering coefficient (CC) [1] for all the graphs (Figure 6). This measure indicates the quantity of triangles in the network, it means that if A is friend of B and C, there is a high probability of B and

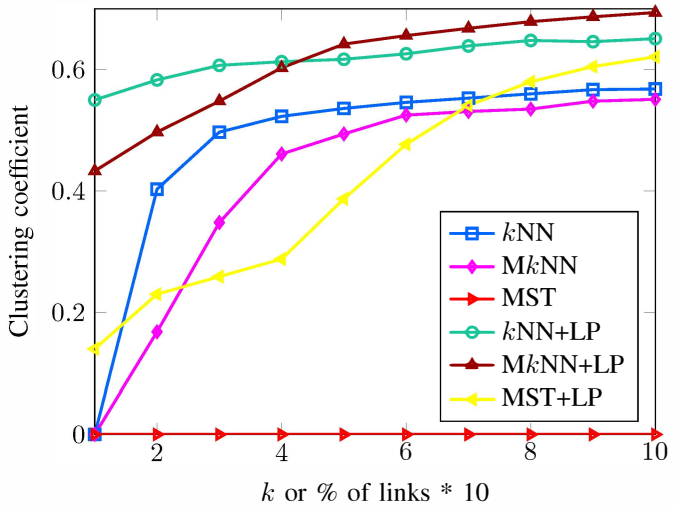


Figure 6. Clustering coefficient for kNN, MkNN, MST and LP versions: kNN+LP, MkNN+LP, MST+LP, applied to Gaussians3 data set. LP versions use $k = 3$ and the common neighbors measure.

C be friends too. MST has the CC = 0 that means no triangle exists in this graph. When MST is combined to LP measure, its CC increases because triangles are formed. kNN graphs have bigger degree than LP variants, however its CC is smaller. The ideal graph should have a balance among degree and CC: small CC/degree indicates it misses structural information for the inference algorithms alternatively the excess edges in network become noise in the graph. It explain the effectiveness of LP graphs, they increase the CC of a graph without increase a lot the average degree.

V. Conclusion

Link prediction has been used in many fields of science, as online social networks where links can be recommended as promising friendships. In this paper link prediction was used for graph construction. From an initial graph structure edges are predict generating a new balanced graph. If a graph is very sparse or very dense it decreases the classification performance. When we consider a basic graph structure and predict new strategic links, it increase the quality of the graph leading to better classification accuracy.

The proposed graphs were evaluated in supervised and semi-supervised classification providing improvements in accuracy. Moreover, the graphs are sparse and represent well the neighborhood of a point. The complexity analysis shows the method has $(nk \log n)$ time complexity using kNN and Common Neighbors link prediction measure.

In future work, other baseline methods could be tested as well other measures for link prediction. Our approach also could be applied in other domains of machine learning using graph-based methods.

Acknowledgments

This research was partially supported by São Paulo Research Foundation (FAPESP) grants: 2013/12191-5, 2011/21880-3 and 2011/22749-8.

REFERENCES

[1] M. E. J. Newman, *Networks: an introduction*. Oxford University Press, 2010.

[2] J. Valverde-Rebaza and A. Lopes, "Link prediction in complex networks based on cluster information," in *SBIA '12*. Springer-Verlag, 2012, pp. 92–101.

[3] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *JASIST*, vol. 58, no. 7, pp. 1019–1031, 2007.

[4] J. Valverde-Rebaza and A. Lopes, "Exploiting behaviors of communities of Twitter users for link prediction," *SNAM*, vol. 3, no. 4, pp. 1063–1074, 2013.

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

[6] J. Valverde-Rebaza and A. Lopes, "Link prediction in online social networks using group information," in *ICCSA 2014*, vol. 8584, 2014, pp. 31–45.

[7] M. Fire, L. Tenenboim, O. Lesser, R. Puzis, L. Rokach, and Y. Elovici, "Link Prediction in Social Networks Using Computationally Efficient Topological Features," in *SOCIALCOM'11*, 2011, pp. 73 –80.

[8] J. Tague-Sutcliffe, "An introduction to informetrics," *Information Processing and Management*, vol. 28, no. 1, pp. 1–3, 1992.

[9] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," in *Proceedings of the WWW*, Brisbane, Australia, 1998, pp. 161–172.

[10] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *J. ACM*, vol. 46, no. 5, pp. 604–632, Sep. 1999.

[11] L. Getoor, E. Segal, B. Taskar, and D. Koller, "Probabilistic models of text and link structure for hypertext classification," in *IJCAI Workshop on Text Learning: Beyond Supervision*, 2001.

[12] R. Ghani, S. Slattery, and Y. Yang, "Hypertext categorization using hyperlink patterns and meta data," in *Proceedings of 18th ICML*. Morgan Kaufmann Publishers, 2001, pp. 178–185.

[13] M. Kotera, Y. Yamanishi, Y. Moriya, M. Kanehisa, and S. Goto, "Genies: gene network inference engine based on supervised analysis," *Nucleic Acids Research*, vol. 40, pp. 162–167, 2012.

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

[15] J. Valverde-Rebaza, A. Soriano, L. Berton, M. C. F. de Oliveira, and A. Lopes, "Music genre classification using traditional and relational approaches," in *Proceedings of BRACIS 2014*. IEEE, 2014, pp. 259–264.

[16] L. Berton and A. Lopes, "Graph construction based on labeled instances for semi-supervised learning," in *Proceedings of 22nd ICPR*, 2014, pp. 2477–2482.

[17] T. Jebara, J. Wang, and S.-F. Chang, "Graph construction and b-matching for semi-supervised learning," in *Proceedings of the 26th ICML '09*. ACM, 2009, pp. 441–448.

[18] M. Maier and U. Luxburg, "Influence of graph construction on graph-based clustering measures," *NIPS*, vol. 22, pp. 1025–1032, 2009.

[19] K. Ozaki, M. Shimbo, M. Komachi, and Y. Matsumoto, "Using the mutual k-nearest neighbor graphs for semi-supervised classification of natural language data," in *Proceedings of CoNLL'11*, 2011, pp. 154–162.

[20] K. Greenfield and W. Campbell, "Link prediction methods for generating speaker content graphs," in *Processings of ICASSP 2013*, 2013, pp. 7721–7725.

[21] M. H. Rohban and H. R. Rabiee, "Supervised neighborhood graph construction for semi-supervised classification," *Pattern Recognition*, vol. 45, no. 4, pp. 1363–1372, 2012.

[22] X. Zhu, "Semi-supervised learning literature survey," Computer Sciences, University of Wisconsin-Madison, Tech. Rep. 1530, 2005.

[23] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. The MIT Press, 2009, ch. 10.

[24] A. Blum and S. Chawla, "Learning from labeled and unlabeled data using graph mincuts," in *Proceedings of ICML'01*. Morgan Kaufmann Publishers Inc., 2001, pp. 19–26.

[25] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *Proceedings of ICML'03*, 2003, pp. 912–919.

[26] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Scholkopf, "Learning with local and global consistency." in *NIPS*. MIT Press, 2003.

[27] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *JMLR*, vol. 7, pp. 2399–2434, Dec. 2006.

[28] S. A. Macskassy and F. J. Provost, "A simple relational classifier," in *2nd Workshop on Multi-Relational Data Mining*, 2003.

[29] Q. Lu and L. Getoor, "Link-based classification," in *Proceedings of ICML'03*, 2003, pp. 496–503.

[30] T. Murata and S. Moriyasu, "Link prediction of social networks based on weighted proximity measures," in *International Conference on Web Intelligence*, Nov 2007, pp. 85–88.

[31] L. Lü and T. Zhou, "Link prediction in weighted networks: The role of weak ties," *EPL (Europhysics Letters)*, vol. 89, no. 1, p. 18001, 2010.

[32] F. Lorrain and H. White, "Structural equivalence of individuals in social networks," *Journal of Mathematical Sociology*, vol. 1, pp. 49–80, 1971.

[33] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, no. 1, pp. 39–43, 1953.

[34] O. Chapelle, B. Schlkopf, and A. Zien, *Semi-Supervised Learning*, 1st ed. The MIT Press, 2010.

[35] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is "nearest neighbor" meaningful?" in *International Conference on Database Theory*, 1999, pp. 217–235.

[36] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *JMLR*, vol. 7, pp. 1–30, 2006.

[37] K. Bache and M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml