Classification Based on the Optimal K-Associated Network

Alneu A. Lopes, João R. Bertini Jr., Robson Motta, and Liang Zhao

University of São Paulo, Institute of Mathematics and Computer Science,
Av. Trabalhador São Carlense 400, São Carlos, Brazil
{alneu,bertini,rmotta,zhao}@icmc.usp.br

Abstract. In this paper, we propose a new graph-based classifier which uses a special network, referred to as optimal *K-associated network*, for modeling data. The K-associated network is capable of representing (dis)similarity relationships among data samples and data classes. Here, we describe the main properties of the K-associated network as well as the classification algorithm based on it. Experimental evaluation indicates that the model based on an optimal K-associated network captures topological structure of the training data leading to good results on the classification task particularly for noisy data.

Keywords: Complex Network, Data Mining, Data Classification, Network formation.

1 Introduction

Complex networks are today a unifying topic in complex systems. Such theory not only had provided a deeper understanding of complex systems, such as biological and social networks, but also gave rise a new approach for modeling such structures. Recently, triggered by some discoveries [1], [2], the theory of complex networks had spread to many branch of sciences. Results obtained so far ranged from microbiology to economy or epidemic analysis to traffic planning (see [3],[4],[5],[6] for instance). Although complex networks theory have been used in a large number of areas there are still plenty of tasks that could be potentially helped by such theory, such as Data Mining tasks. In this scenario complex networks can improve the representation of data from the traditional attribute-value paradigm to a richer relational representation.

Data mining tasks, in general, can be divided in two categories, predictive and descriptive mining [7] and two respective major branches are data classification and clustering. In data classification, each data instance has an associated label that characterizes it in a group named class. The objective is to predict the classification of a new pattern based on a dataset with patterns already classified, used as training set. Some examples of classification problem are medical diagnosis, credit assignment, market prediction [8]. In clustering tasks the data does not have an associated label and the objective in this case is to describe the patterns

J. Zhou (Ed.): Complex 2009, Part I, LNICST 4, pp. 1167–1177, 2009.

formed by some groups of the dataset. Some common application is clustering are gene categorization, Web documents classification, market research [9].

Throughout recent years, graph-based clustering algorithms have received great attention and have been widely studied (see [10],[11],[12]). This interest is mostly justified due to some advantages the network approach provides. Among the most important ones, graph representation (i) can capture topological underline structure of similarity relation among data leading to interesting approach for dealing with clustering problem or community detection; (ii) promotes hierarchical representation of cluster and; (iii) enables detection of clusters with arbitrary shapes. In graph-based algorithms, each node of the graph represents a data point and the edges the similarity between them.

There exists various ways of connecting the nodes in a network (see [10] for instance), but usually the basic idea is that the probability of connection between two vertices are proportional to the similarity between them. Stated in this way, it is clear that close groups of instances tend to be heavier linked together than the rest of the data. Hence, it is straightforward the usage of community detection algorithms [13] to reveal similarity in data which in fact has been widely used clustering problems [14],[15].

This wide usage of complex network in clustering problems does not reflect its use in classification tasks. Motivated by the richness of graph representation as well as the new methods provided by complex networks theory, this paper presents a classification method based on complex networks. Taking into consideration that graph-based classification has not been much explored in the literature, this work is an effort toward this direction. In what follows will be developed a network-based approach for dealing with classification tasks. The obtained results indicate a potentially new approach for dealing with classification problems especially in the presence of noise.

The remainder of the paper is organized as follows: Section 2 presents the three parts for generating a network that will be used by the classifier: 1) the K-associated network - how the network is built from data; 2) how to extract the purity measure for a given component and 3) how to obtain the network that maximizes the component's purity, called optimal network. In Section 3 the classifier that uses the network described in Section 2 is derived. Section 4 presents some results of using our classifier in ten knowledge domain, as well as a comparison with two other well know algorithms. Finally, Section 5 concludes the paper.

2 The Graph-Based Model

In this section we present a new network based on similarity relations among data and on its classes, referred to as K-associated Network. The main properties of such network are presented, as well as the algorithm to build this network from training data and the usage of this model in the classification task. First of all given a vector-based dataset it is necessary a method for converting it to a network. Once the network is obtained, the purity measure for each component of the network is computed. This measure is detailed next.

Note that the K-associated network depends on the parameter K, and as will become clear ahead, each value of K will generate a network with different number of components with different purity. In a classification context however, it is desirable components with minimum of noise, i.e. with high purity. The idea is to overcome the parameter restriction for creating a single network by creating various networks with different K, keeping, along this process the components with highest purity, obtaining an optimal network.

2.1 The K-Associated Network

A network built from data that supposes to represent the training set must inherit the main characteristic of the data. In this network, each pattern in the training set is mapped to a node in the network and the linkages must be done in order to preserve some desirable similarity relations.

In a K-associated network, among the K nearest neighbors of a given vertex, the connections will be established between the given vertex and those within the same class. As this process is done for all vertices in the network, in many cases a vertex v_i is already connected to vertex v_j (supposing v_i and v_j belongs to the same class) when vertex v_j selects vertex v_i to connect, it is possible that v_i also selects v_j, resulting in two undirected connections between vertices v_i and v_j. This is justified by the fact that if v_j is in the K-neighborhood of v_i it would establish a connection but this connection does not mean that vi belongs to the K-neighborhood of v_j since this not necessarily happens. So when v_j is found in the K-neighborhood of v_i a connection is established no matter they were already connected. This particular way of wiring the network is fundamental for the properties defined ahead. At the end of this process, for any K, there will be at least as many components as the number of classes. Notice also that the number of components decreases monotonically to the number of classes as K increases.

In a formal way, the resulting K-associated network $A = (V, E)$ consists of a set of labeled vertices V and a set of edges E between them, where an edge e_{ij} connects vertex v_i with vertex v_j if $class(v_i) = class(v_j)$ and v_j belongs to the K nearest neighbors of v_i. Where $class(v_i)$ stands for the class associated with vertex v_i.

Let $knn(v_i)$ be the set of the K nearest neighbors of vertex v_i by a given similarity measure. The neighborhood N_{K_i} for a vertex v_i is defined as its immediately connected K-neighbors as follows: $N_{K_i} = \{e_{ij} \in E | v_j \in class(v_i)$ and $v_j \in knn(v_i)\}$. The degree g_i of a vertex v_i is defined as the number of vertices, $|N_{K_i}|$, in its neighborhood N_{K_i}.

Following the above definitions, we can notice:

1. If $class(v_i) = class(v_j)$ and $v_j \in knn(v_i)$ e $v_i \in knn(v_j)$ both vertices (e_{ij} and e_{ji}) are considered in the network, and in the calculation of g_i.
2. One can easily check that (see Fig. 2) the maximum degree of v_i is $2K$, that occurs when there are only vertices with the same label in the K-neighbourhood of v_i in the component. This maximum degree can be achieved only when the component of v_i has more than K vertices.

3. In a component, only vertices which belong to a same label can be linked. Thus each component is associated to only one label.

Figure 1 illustrates a bi-dimensional representation of a toy dataset with 10 examples from black label and 5 from white label, and its correspondent 1,3, and 5-associated networks.

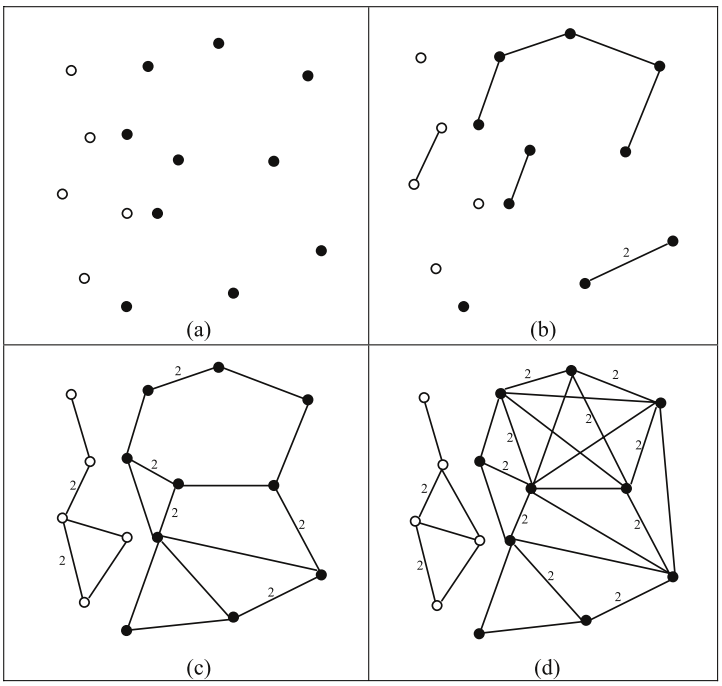

Fig. 1. (a) 2D representation of the dataset. (b) (c) and (d) are 1, 3 and 5-associated correspondent networks, respectively. Notice that edges between vertices can represent more than one connection.

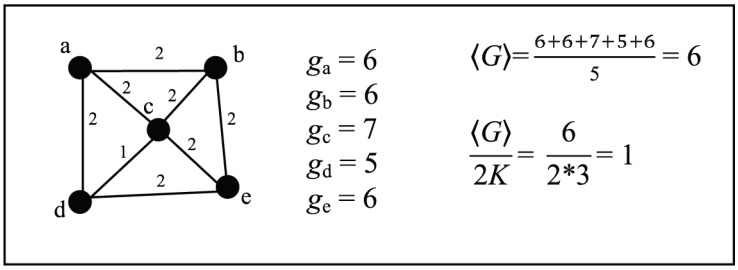

Fig. 2. An example of a "pure" component with 5 vertices and $K = 3$

2.2 The Purity Measure

The method of generation of the K-associated network improves the representation of the training set and enables some interesting calculation such as the measure of purity. Such measure uses network topologies to quantify how intertwined are the nodes of different class in the network.

Let g_i be the degree of vertex v_i, N the number of patterns in the training set, K the number of neighbors used in the construction of the network. Consider the ratio $g_i/2K$, this relation corresponds to the fraction of links between the vertex v_i and vertices in its own component. This ratio varies between 0 and 1, inclusively. Hence, the total of links between N_c vertices in a component C is given by eq. (1).

$$|E_c| = \frac{1}{2}\sum_{i=1}^{N_c} g_i = \frac{N_c}{2}\sum_{i=1}^{N_c}\frac{g_i}{N_c} = \frac{N_c}{2}\langle G_c\rangle \tag{1}$$

Now, if we consider that for each vertex v_i there are $(2K-g_i)$ edges "rejected" (they would link v_i to vertices of another component), the total number of such rejected edges is given by,

$$\begin{aligned} |-E_c| &= -\frac{1}{2}\sum_{i=1}^{N_c}(2K-g_i) = K\left(\sum_{i=1}^{N_c}1 - \sum_{i=1}^{N_c}\frac{g_i}{2K}\right) \\ &= K\left(N_c - \frac{N_c}{2K}\langle G_c\rangle\right) = N_c\left(K - \frac{1}{2}\langle G_c\rangle\right). \end{aligned} \tag{2}$$

Such total can be seen as the number of vertices that would link vertices with different components (classes), in a network in which any vertex was linked to its K neighbors.

Thus, the probability of edges between vertices in the same component C (intra-component links) is given by,

$$P_i = \frac{\frac{N_c\langle G_c\rangle}{2}}{\frac{N_c\langle G_c\rangle}{2} + \frac{N_c(2K-\langle G_c\rangle)}{2}} = \frac{\langle G_c\rangle}{2K}. \tag{3}$$

In the equation (3) $P_i = 1$ when there are only vertices with the same label in the K-neighborhood of every vi in the component, see Fig. 2. Thus, $\langle G_c\rangle/2K$ ratio can be seen as a measure of "purity" of the region of the component C.

We can also compute the probability of links between vertices from distinct components (which do not exist in a K-associated network, but such probability can be understood as a measure of "impurity" in the region of the component).

$$P_i = \frac{N_c\,(2K - \langle G_c\rangle)}{N_c\langle G_c\rangle + N_c\,(2K - \langle G_c\rangle)} = \frac{2K - \langle G_c\rangle}{2K} \tag{4}$$

We have demonstrated that the ratio $\langle G_c\rangle/2K$ expresses the probability of intra-component connection. Thus it expresses the purity in the region of a component. This assumption can be also empirically evaluated. In Fig. 3 is represented the ratio (averaged in 10 runs) for five artificial datasets with 250 vertices.

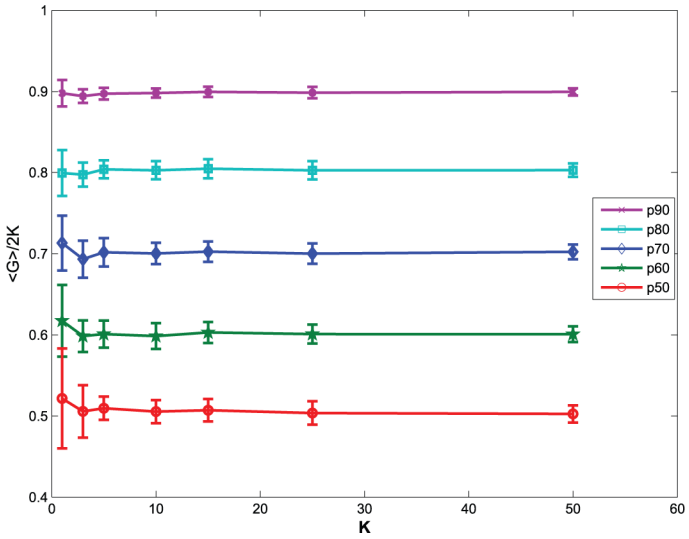

Fig. 3. The average $\langle G \rangle / 2K$ for 5 K-associated networks from datasets with 90, 80, 70, 60, and 50% of purity in the region of the component

The datasets, referred to as p90, p80, p70, p60, p50, respectively, were built using a normal distribution with 90.0, 80.0, 70.0, 60.0, and 50.0% of "purity". This experiment shows that $\langle G \rangle / 2K$ is a good approximation of the purity of the component.

2.3 Optimal Network

In the process described so far, each K yields a network; clearly some networks will have better components than others, according to the notion of purity. In general, rarely a network obtained with a unique value of K will have all the best components among all others components in all K possible networks. Bearing this in mind, a suggestive idea is to obtain a network with the best organization of data into components independently of a unique K. For doing this, the idea is to vary K keeping the best components found. This process will result in a network called optimum network, with components formed by distinct values of K.

The idea of the optimum network based only in the purity of components has some drawbacks. As purity does not consider the length of the component it tends to favor the small ones, hence it is not a good gauge for constructing a network that must optimally represent the data. The quantity to be maximized though is not purity itself but a slightly variant that considers the size of the components. An intuitive way for overcoming this problem is multiply purity by the number of vertices for a given component, as stated in eq. (5). However the measure W incurs another problem. Now, very large components with low purity may have advantage over smaller ones with high purity. To solve this problem a new constraint that relates size and purity in an indirect way is added to the equation.

$$W_j = \frac{\sum_{i=1}^{N_j} g_i}{2K_j} \qquad and \qquad \langle G_c \rangle > K \tag{5}$$

Where W_j is the new measure for component C_j, g_i is the degree of vertex v_i, N_j is the number of patterns in component C_j, K_j is the number of neighbors used in the construction of component j.

Notice that the purity measure will still be used, after the optimum network has been obtained, in the classification process. The optimum network is the final structure obtained through this process. This network can be viewed as the result of a supervised learning process and will be used in the classification process as exposed in the next section. The algorithms 1 and 2 detail the optimum K-associated model generation from data.

Algorithm 1. Optimum K-associated Model

Input:

$V = \{v_i, ..., v_n\}$	set of vertices - examples
D	matrix of distance between vertices
$L = \{label(v_i), ..., label(v_n)\}$	set of labels
K_{max}	number of iterations

Output

$LC_{best} = \{(C_i(V_i; E_i); P_i)\};$ best components' network

Algorithm

$K = 1;$
$LC_{best} = Kac(V, D, L, K);$
For$(K = 2; K \leq K_{max}; K = K + 1)$
  $LC_K = Kac(V, D, L, K);$
    For each component C_{K_i} in LC_K
      Determines correspondent components $j\{C_{(K-1)j}\}$ in LC_{K-1}
      If $(W(C_{K_i}) \geq W(C_{(K-1)j})$ for any j and $\langle G_i \rangle > K$
        $LC_{best} = LC_{best} - \{C_{(K-1)j}\} \cup \{C_{K_i}\};$

Return (LC_{best})

In this algorithm, the optimum network is firstly set as the 1-associated network ($K = 1$). As K increases, different components from (K-1)-associated network can be merged into just one component C in the current K-associated network. If this new component has a better measure W_c, and $\langle G_c \rangle > K$ it replaces the corresponding components of the previous (K-1)-associated networks (LC_best). The experiments have shown that in few iterations (at most 5) the number of components converges to the number of classes. The following algorithm details the construction of a K-associated network from data.

Algorithm 2. K-associated Network from data (Kac)

Input:

$V = \{v_i, ..., v_n\}$	set of vertices - examples
D	matrix of distance between vertices
$L = \{label(v_i), ..., label(v_n)\}$	set of labels
$K = \{1, ..., n\}$	number of nearest neighbors to be used

Output

$LC = \{(C_i(V_i; E_i); P_i)\}$	set of components and purity

Algorithm

$C = \emptyset$;
For each vertex v_i in V
  $N_{k_i} = \{v_j | v_j = knn(i)$ and $label(v_j) == label(v_i)\}$;
  $E = E \cup \{e_{ij} | v_i \in N_{K_i}$ and $v_j \in N_{K_i}\}$;
  For each connected sub-graph C_i from $A(V; E)$
    Compute the purity of the component (P_{C_i})

Return ($LC = \{(C_i(V_i; E_i); P_i)\}$)

This algorithm builds the K-associated network for desired K and return a list of its components with respective purity.

3 Non-parametric *K*-Associated Classifier

The objective is to derive a non-parametric classifier that uses the optimal K-associated network as model from training data to accurately classify new patterns. As stated before this structure stores the best components of data found through a large range of K. The component purity can be seen as a priori of the data represented in the component. Since each component contains vertices (instances) from only one class, we can compute the probability of a new instance to belong to given class by computing the probability of this instance to belong to the components of the same class. Before presenting details on this classifier some notation must be introduced.

Typically a training pattern x_i is represented by $x_i = (x_{i1}, x_{i2}, ..., x_{ip}, \omega_i)$, which x_i represents the i-th training pattern with ω_i its associated class, in a M-class problem $\Omega = \{\omega_1, \omega_2, ..., \omega_M\}$. In the same way, a new pattern is defined as $y_j = (y_{j1}, y_{j2}, ..., y_{jp})$, excepted that now the class ω_j associated with the new pattern y_j must be estimated. Consider also the set of components of the optimum network $C = \{C_1, ..., C_R\}$, where R is the number of components and $R \geq M$.

According to Bayes theory [16] the posteriori probability of a new instance y_i to belong to the component C_j given the neighbors N_{K_i} of y_i that belongs to the component C_j is,

$$P\left(y \in C_j | N_{K_i}\right) = \frac{P\left(N_{K_i} | C_j\right) P\left(C_j\right)}{P\left(N_{K_i}\right)}. \tag{6}$$

It is important to bear in mind that each component C_i came from a particular K-associated network. Hence, the neighborhood N_{K_i} must considers this particular K.

As purity scores individually how pure is each component, the normalized purity acts as priori probability,

$$P\left(C_j\right) = \frac{g_j}{\sum_{i=1}^{M} g_i}. \tag{7}$$

Probability of having N_{K_i} connections, among the K_j possible, to component C_j, is

$$P\left(N_{K_i} | C_j\right) = \frac{\#\{e_{N_k} \in C_j\}}{K_j}. \tag{8}$$

Probability of N_{K_i} connections is given by eq. (9).

$$P\left(N_{K_i}\right) = \sum_{i=1}^{M} P\left(N_{K_i} | C_i\right) P\left(C_i\right) \tag{9}$$

As in many cases there are more components than classes, according to Bayes optimal classifier, it is necessary to sum the posteriori probability that correspond to a common class. So the posteriori probability of the new instance to belong to a given class is given by eq. (10).

$$P\left(y | \omega_i\right) = \sum_{C_j = \omega_j} P\left(y \in C_j | N_j\right) \tag{10}$$

Finally the greatest values between the found *posteriori* probabilities reflect the most probable class to assign for the new instance, according to eq. (11).

$$\varphi\left(y\right) = arg\ max\{P\left(y|\omega_1\right), ..., P\left(y|\omega_M\right)\} \tag{11}$$

where $\varphi(y)$ stands for the class attributed for instance y.

4 Experiments and Results

This section presents and discusses the results of using the proposed algorithm and the two well known multiclass classification algorithms, the K-nearest neighbor, for tree frequently used values of K (1, 3, and 5) and the decision tree C4.5 algorithm. The tests were carried out learning from nine multiclass knowledge

domain data taken from the UCI-Repository [17]. Each of the algorithms was implemented in Java and the results were obtained through 10-fold stratified cross-validation process. Table 1 presents the test averaged under 10 runs followed by its standard deviation. From these ten datasets, we produce new datasets with noise changing the classes in 5 and 10% of the training data. The results carried out on these noise data are also showed in the Table 1.

Table 1. Comparison results through nine knowledge domains

Domain	**Proposed Algorithm**	**C4.5**	**K-NN K=1**	**K-NN K=3**	**K-NN K=5**
Yeast	98.2±0.79	**98.9±7.8**	98.8±0.8	98.5±0.9	98.3±1.0
Yeast (5%)	85.2±2.7	81.3±3.2	80.5±1.7	86.9±2.6	**88.6±3.1**
Yeast (10%)	**78.5±2.1**	66.5±1.5	68.6±4.4	74.3±3.6	76.9±2.9
Tae	**63.6±15.1**	61.6±11.9	63.3±11.7	41.9±11.7	43.6±11.5
Tae (5%)	50.3±14.5	51.7±18.5	**55.9±12.8**	34.6±13.1	39.3±13.5
Tae (10%)	47.1±14.2	**48.9±14.4**	47.0±11.8	36.1±12.1	31.2±10.5
Zoo	**97.1±4.7**	93.1±9.5	96.2±7.2	93.4±9.6	88.0±11.9
Zoo (5%)	**86.1±8.5**	82.2±13.2	78.6±12.6	82.7±12.4	79.2±10.4
Zoo (10%)	**72.2±16.8**	63.2±15.9	63.0±15.1	71.3±12.9	71.5±14.5
Image	74.3±8.1	**79.5±7.5**	74.9±8.9	74.4±8.7	70.4±8.1
Image (5%)	60.9±12.0	**66.2±9.4**	61.3±12.1	64.1±11.4	64.6±9.3
Image (10%)	**56.3±11.3**	54.7±10.8	54.2±10.9	54.9±10.3	54.2±10.8
Wine	88.8±6.9	**90.9±6.8**	83.9±7.4	80.3± 8.5	83.1±8.6
Wine (5%)	**76.5±8.9**	74.3±10.7	72.6±10.1	69.9±10.2	75.1±10.1
Wine (10%)	64.6±10.7	64.1±10.9	59.3±10.3	61.7±12.3	**65.1±9.7**
Iris	**98.0±3.2**	94.6±6.1	97.8±3.6	98.1±3.7	97.8±3.5
Iris (5%)	**88.6±8.9**	82.0±6.3	83.9±9.1	85.7±7.6	87.6±8.7
Iris (10%)	**81.3±8.2**	70.6±8.4	71.9±10.5	78.3±10.2	80.8± 11.0
Glass	66.8±9.3	64.9±5.7	**73.3±8.3**	70.1±13.9	68.7±9.1
Glass (5%)	**64.0±8.6**	58.4±9.6	57.6±9.8	63.2±11.9	59.1±10.2
Glass (10%)	**57.7±9.2**	49.0±11.4	56.4±9.9	54.2±10.2	54.8±9.4
E.coli	**97.6±2.6**	96.5±4.3	97.6±2.8	97.0±3.2	96.8±2.9
E.coli (5%)	**85.4±6.6**	80.4±6.4	79.8±6.3	83.6±5.7	84.2±6.7
E.coli (10%)	**74.4±6.1**	71.2±5.8	62.1±7.6	72.2±7.5	74.1±7.7
Balance	94.2±3.4	89.9 4.2	**96.9±1.9**	94.7±2.9	95.6±2.4
Balance (5%)	80.3±3.9	78.8±3.9	82.3±4.1	82.8±4.2	**84.8±4.1**
Balance (10%)	72.9±5.6	61.9±4.3	66.3±5.4	73.6±5.5	**75.9±5.0**

The proposed classifier had better performance on 15 of 27 datasets; the C4.5 had better performance on 4 of 27; the KNN for (K=1, 3, and 5) had better performance on 4 of 27, 0 of 27, and 4 of 27, respectively. As one could expect, the KNN with K=5 has better performance than KNN with K=1 for noise data, but poorer results in the other cases.

5 Conclusions

This paper presented an effort considering complex networks for dealing with classification problems. Previously unseen in complex network literature, this paper proved that it is possible to use complex networks not only to clustering problems but also to classification tasks. The results, although considering little datasets and few comparison, indicates a favorable scenario for our approach; particulary for noisy data. Future work expects some more comparisons to better validate our method.

References

1. Watts, D.J., Strogatz, S.H.: Collective Dynamics of 'Small-World' Networks. Nature 393, 440–442 (1998)
2. Albert, R., Jeong, H., Barabási, A.-L.: Diameter of the World Wide Web. Nature 401, 130–131 (1999)
3. Newman, M.E.J.: The Structure and Function of Complex Networks. SIAM Review 45(2), 167–256 (2003)
4. Albert, R., Barabási, A.-L.: Statistical Mechanics of Complex Networks. Review of Modern Physics 74, 47–97 (2002)
5. Bornholdt, S., Schuster, H.G.: Handbook of Graphs and Networks: From the Genome to the Internet. Wiley-vch, Weinheim (2003)
6. Dorogovtsev, S.N., Mendes, J.F.F.: Evolution of Networks: From Biological Nets to the Internet and WWW. Oxford University Press, Oxford (2003)
7. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann, San Francisco (2006)
8. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. John Wiley & Sons, Inc., Chichester (2001)
9. Berkhin, P.: Survey of Clustering Data Mining Techniques. Technical report, Accrue Software (2002)
10. Schaeffer, S.E.: Graph Clustering. Computer Science Review 1, 27–34 (2007)
11. Karypis, G., Han, E.-H., Kumar, V.: Chameleon: Hierarchical Clustering using Dynamic Modeling. IEEE Computer 32(8), 68–75 (1999)
12. Guha, S., Rastogi, R., Shim, K.: CURE: An Efficient Clustering Algorithm for Large Databases. In: Proc. of 1998 ACM-SIGMOD Int. Conf. on Management of Data, pp. 73–84 (1998)
13. Newman, M.E.J., Girvan, M.: Finding and Evaluating Community Structure in Networks. Physical Review E 69, 026113(1-15) (2004)
14. Danon, L., Duch, J., Arenas, A., Dáz-Guilera, A.: Comparing Community Structure Identification. Journal of Statistical Mechanics: Theory and Experiment, P09008(1-10) (2005)
15. Hopcroft, J., Khan, O., Kulis, B., Selman, B.: Tracking Evolving Communities in Large Networks. Publications of the National Academy of Sciences USA 101(1), 5249–5253 (2004)
16. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference and Prediction. Springer, Heidelberg (2001)
17. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA (2007), `http://www.ics.uci.edu/~mlearn/MLRepository.html`