

An Experimental Study of Graph Classification Using Prototype Selection

Andreas Fischer, Kaspar Riesen and Horst Bunke
Institute of Computer Science and Applied Mathematics
University of Bern, Neubrückstrasse 10, CH-3012 Bern, Switzerland
{afischer, riesen, bunke}@iam.unibe.ch

Abstract

In structural pattern recognition, a major drawback of graph based representation is the lack of algorithmic tools. To overcome this lack, we embed graphs in vector spaces by means of prototype selection and graph edit distance, thus making them available to all algorithms of statistical pattern recognition that operate on feature vectors. In previous work a similar procedure was applied. However, the only classifier used within this framework was Support Vector Machine (SVM). In the present paper, we significantly extend the scope of the previous work and present an experimental study where, in addition to SVM, a number of other well established classifiers from statistical pattern recognition are used for graph classification. On a total of five different graph data sets of diverse nature it is demonstrated that the proposed graph embedding in conjunction with standard classifiers from statistical pattern recognition has great potential to outperform classification methods applied in the original graph domain.

1. Introduction

The first step in any system in pattern recognition consists in the representation of objects by adequate data structures. In statistical pattern recognition, the data structure is given by n -dimensional vectors $\mathbf{x} \in \mathbb{R}^n$, where each of the n dimensions represents the value of a specific feature. A huge amount of algorithms for classification of objects given in terms of feature vectors have been developed. For a survey see [3].

Yet, the use of feature vectors implicates two limitations. First, as vectors describe a predefined set of features, all vectors of a set have to preserve the same length regardless of the size or complexity of the corresponding objects. Furthermore, there is no direct possibility to describe binary relationships among differ-

ent parts of an object. In structural pattern recognition, both constraints can be overcome by graphs [2]. Hence, graph based representation found widespread application in various fields, e.g. in [6, 15, 2].

The major drawback of graphs, however, is that they offer little mathematical structure, i.e. most of the basic mathematical operations, such as sum and product, required to build a classifier or some other pattern recognition algorithm are not available or not defined in a standardized way for graphs. A promising direction to overcome the lack of algorithmic tools for graph classification is graph kernels [5]. Yet graph kernels are still limited to algorithms that need only patterns in terms of scalar products as their input. A much more versatile approach is based on graph embedding in real vector spaces. Basically, an embedding of graphs into vector spaces establishes full access to the rich repository of algorithmic tools for pattern analysis. Well-known examples of such embeddings are spectral decompositions [7, 20, 17]. However, most spectral methods are only applicable to unlabeled graphs or labeled graphs with constrained label alphabets [9].

In the present paper we use graph embedding based on prototype selection and graph edit distance [14]. As a major advantage compared to other approaches, this embedding procedure can be applied to any kind of graphs (directed or undirected, labeled or unlabeled). If we allow labels on the nodes or edges, these labels can be symbolic, numerical or whole attribute vectors. In contrast to the contribution in [14] where only support vector machines are used for classification, we describe a much broader study in the present paper using seven different classifiers, all originally developed for feature vectors.

2. Graph Embedding

The idea of prototype selection emerged from the argument that the notion of dissimilarity is more fundamental than that of a feature [11]. Considering a class

as a collection of similar objects, the dissimilarity between an object and a set of class representatives describes the membership of the object to the class more adequately. The objects representing a class are called *prototypes* and have to be selected carefully with a view to preserving the essential information and avoiding redundancy. This approach has been applied successfully to embed feature vectors [11] and strings [18] into dissimilarity spaces, and recently to embed graphs [14]. Let $\{p_1, \dots, p_n\}$ be a set of prototype graphs. We use the following graph embedding $\chi(g) \in \mathbb{R}^n$ of the graph g

$$\chi(g) = (d(g, p_1), \dots, d(g, p_n))$$

Here, $d(g, p_i)$ denotes a dissimilarity measure between g and the prototype graph p_i . Applied to graphs, the prototype selection approach not only maps graphs into dissimilarity spaces, but also embeds them into a vector space, thus bridging the gap between structural and statistical pattern recognition. That is, by means of our embedding approach the limitation of graphs regarding the lack of algorithmic tools is overcome, because all methods originally developed for vectorial patterns become available to graphs after embedding.

Graph embeddings are closely related to graph kernels [5]. The explicit embedding of graphs into a Euclidean space makes standard kernels applicable, such as the radial basis function (RBF) kernel κ

$$\kappa(g_1, g_2) = \exp(-\gamma \cdot \|\chi(g_1) - \chi(g_2)\|^2), \gamma > 0$$

2.1. Prototype Selection

The goal of prototype selection is to choose a subset of graphs from the training set that represent the different classes most precisely with respect to their graph structure. Although a random selection might work well in some cases [11], we use three other approaches that construct the set of prototypes in a more controllable manner. The *spanning prototype selector* iteratively adds prototypes to the set that are most dissimilar to the already chosen ones, the *k-centers prototype selector* operates in a similar way as *k-means* clustering, and the *targetsphere prototype selector* distributes the prototypes uniformly between the center to the border of the graph data set. For more details we refer to [14].

2.2. Edit Distance

To calculate the dissimilarity $d(g_1, g_2)$ between two graphs, we use the graph edit distance, a very flexible method for error-tolerant graph matching that was originally developed for strings [19] and later transferred to graphs [1]. The idea is to transform one graph into

another by applying distortions and summing up the costs of these distortion operations. Possible distortions, called *edit operations*, are *insertion*, *deletion* and *substitution* of nodes and edges. The sequence of edit operations needed to transform one graph into another is called *edit path*. The edit distance is then defined as the minimal sum of costs over all possible edit paths.

Usually, the edit distance is calculated with the *A** algorithm which performs a best-first tree search, possibly using a lower bound heuristic for the estimated future costs [1]. The *A** algorithm always finds the optimal solution but needs exponential time. To calculate the edit distance for large graphs, we use an approximation to obtain a suboptimal edit distance in polynomial time [13]. Thus, given n predefined prototype graphs, the embedding of an arbitrary graph can be established with n distance computations in polynomial time.

3. Classification Methods

In our classification experiment, we use the following well-known classifiers:

- *Normal Bayes* and *Linear Bayes* classifier [3]
- *Logistic Regression* [8]
- *Support Vector Machine* (SVM) [3]
- *K-Nearest Neighbor* (KNN) [3]
- *Binary Decision Tree* [12]
- *Radial Basis Neural Network* (RBNN) [10]

Note that the Linear Bayes, Logistic Regression, and SVM classifier belong to the group of linear classifiers, whereas all others are non-linear classifiers. For details on the implementation of the classifiers we refer to [4].

4. Experimental Results

A major advantage of the prototype selection approach lies in its applicability to all types of graphs. In our classification experiment we consider five graph data sets that differ in graph size, edge type (directed or undirected), and label domain. The classification task also varies among the data sets in terms of number of graphs, number of classes, and class balance.

The *Fingerprints* data set is extracted from grayscale fingerprint images by representing important regions, such as bifurcation points, by nodes. The *Letters* data set is constructed from letter line drawings by representing lines with edges and line endings with nodes. The *Molecules* and the *Mutagenicity* data set represent atoms with nodes and chemical bonding with edges. The *Websites* data set is extracted from news websites by representing frequent words with nodes that are

Table 1. Experimental Results

Classifier	Fingerprints	Letters	Molecules	Mutagenicity	Websites
KNN (graph)	76.4	77.0	96.5	72.6	79.4
SVM (kernel)	61.8	73.8	90.9	59.3	84.4
Linear Bayes	85.6 ○○	81.1 ○○	96.8 ○	71.8 ○	81.2 ○●
Regression	82.6 ○○	78.5 ○	96.2 ○	70.4 ●○	82.9 ○
SVM	84.2 ○○	81.0 ○○	97.0 ○	74.7 ○○	81.7 ○●
Normal Bayes	80.8 ○○	79.5 ○○	96.9 ○	73.0 ○	76.7 ●●
KNN	82.8 ○○	73.9 ●	95.4 ●○	71.2 ○	74.4 ●●
Decision Tree	77.6 ○	60.9 ●●	94.5 ●○	69.1 ●○	67.6 ●●
RBNN	84.2 ○○	80.2 ○○	95.3 ●○	72.1 ○	82.7 ○●

Accuracy of the reference systems and the classifiers based on prototype selection on the graph test sets.

○/○○ Statistically significantly better than one/both reference systems ($\alpha=0.05$)

●/●● Statistically significantly worse than one/both reference systems ($\alpha=0.05$)

linked with a directed edge if one word precedes another. In Table 2 the main characteristics of the data sets are given. For more details we refer to [14].

Table 2. Data Set Statistics

Data Set	Classes	Training	Validation	Test
Fingerprints	4	500	300	500
Letters	15	750	750	1500
Molecules	2	250	250	1500
Mutagenicity	2	1500	500	2337
Websites	20	780	780	780

4.1. Setup

The graph data sets are first divided into three disjoint data sets for training, validation and testing. The validation set is used to optimize various free parameters, such as the number of prototypes and classifier parameters, independently from the test set. Then, the graphs are classified in three steps. First, we apply the prototype selectors from Section 2.1 to transform all graphs into feature vectors. Secondly, we apply an RBF PCA [16] to improve the classifiability of the feature vectors and to make the RBF information available to all classifiers in a uniform manner. Finally, the classifiers from Section 3 are applied to the feature vectors.

We compare our prototype selection results with two reference systems. First, a KNN classifier in the graph domain in conjunction with graph edit distance is employed. This is a standard classification approach in the graph domain. Secondly, an SVM with kernel values directly gained from graph edit distance $d(g_1, g_2)$ is used:

$$K(g_1, g_2) = -d(g_1, g_2)^2$$

This reference system takes into account recent developments in the field of kernel methods that can be applied on graphs [9]. Note that there are hardly any other reference systems available in the graph domain that are able to deal with all data sets used in this study.

4.2. Results

The results in Table 1 show that significant improvement in classification accuracy can be achieved compared to the reference systems when using statistical classifiers relying on vector space embedded graphs. In three out of five cases (Fingerprints, Letters, Mutagenicity) both reference systems are significantly outperformed, and in all cases one reference system is significantly outperformed by at least one classifier.

Comparing the different classifiers, we observe that the linear classifiers SVM and Linear Bayes are most successful. Among the non-linear classifiers, applying the Binary Decision Tree classifier was not successful; the best results are achieved by the RBNN classifier that proves high stability on all graph data sets.

The feature vector dimension of the embedded graphs is given by the number of prototype training graphs and is then reduced using RBF PCA. On average over all data sets and classifiers, the optimum dimension was about a third of the number of training graphs, ranging from 12% (Mutagenicity) to 47% (Molecules). For details on the influence of the non-linear RBF PCA mapping on the classification accuracy we refer to [4].

5. Conclusion

Choosing dissimilarities to a set of prototypes as features was originally proposed in [11]. This approach turns out to be especially useful in the domain of graphs.

On the one hand, the graph edit distance provides us with a powerful dissimilarity measure and on the other hand, the prototype selection and embedding procedure makes methods from statistical pattern recognition applicable to graphs by representing graphs as feature vectors. The basic idea of our approach is to regard the dissimilarities to a number of predefined prototypes as a vectorial description of an input graph. After embedding the graphs of a given population into a vector space, many algorithms become available that are not applicable on graphs in the original domain.

In this paper, we have investigated the applicability of prototype selection on graphs by conducting a broad experimental study with various graph data sets and classifiers. The results show that our approach outperforms standard reference systems from the graph domain, such as KNN classification and kernel SVM classification based on graph edit distance. As a main contribution of this paper, we show the stability of the prototype selection method applied to graphs; the reference systems are outperformed significantly by numerous standard classifiers from statistical pattern recognition, and not only by an SVM classifier as shown earlier in [14]. After optimizing the set of prototypes, the linear classifiers SVM and Linear Bayes perform best, followed by the Radial Basis Neural Network classifier. Future improvements are to be expected by combining different classifiers and by finding new criteria for selecting a representative set of prototypes.

6. Acknowledgments

This work has been supported by the Swiss National Science Foundation (Project 200021-113198/1). Furthermore, we would like to thank R. Duin and E. Pekalska for valuable discussions and hints regarding our embedding methods.

References

- [1] H. Bunke and G. Allermann. Inexact graph matching for structural pattern recognition. *Pattern Recognition Letters*, 1:245–253, 1983.
- [2] D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(3):265–298, 2004.
- [3] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley Interscience, 2nd edition, 2000.
- [4] A. Fischer. Classification of dissimilarity space embedded graphs. Master’s thesis, University of Bern, Switzerland, 2008.
- [5] T. Gärtner. A survey of kernels for structured data. *SIGKDD Explorations Newsletter*, 5(1):49–58, 2003.
- [6] M. Lozano and F. Escolano. Protein classification by matching and clustering surface graphs. *Pattern Recognition*, 39(4):539–551, 2006.
- [7] B. Luo, R. Wilson, and E. Hancock. Spectral embedding of graphs. *Pattern Recognition*, 36(10):2213–2223, 2003.
- [8] T. Mitchell. *Machine Learning*. McGraw-Hill Higher Education, 1997.
- [9] M. Neuhaus and H. Bunke. *Bridging the Gap Between Graph Edit Distance and Kernel Machines*. World Scientific, 2007.
- [10] J. Park and I. W. Sandberg. Universal approximation using radial-basis-function networks. *Neural Computation*, 3(2):246–257, 1991.
- [11] E. Pekalska and R. Duin. *The Dissimilarity Representations for Pattern Recognition: Foundations and Applications*. World Scientific, 2005.
- [12] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.
- [13] K. Riesen, M. Neuhaus, and H. Bunke. Bipartite graph matching for computing the edit distance of graphs. In F. Escolano and M. Vento, editors, *Proc. 6th Int. Workshop on Graph Based Representations in Pattern Recognition*, LNCS 4538, pages 1–12, 2007.
- [14] K. Riesen, M. Neuhaus, and H. Bunke. Graph embedding in vector spaces by means of prototype selection. In F. Escolano and M. Vento, editors, *Proc. 6th Int. Workshop on Graph Based Representations in Pattern Recognition*, LNCS 4538, pages 383–393, 2007.
- [15] A. Schenker, M. Last, H. Bunke, and A. Kandel. Classification of web documents using graph matching. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(3):475–496, 2004.
- [16] B. Schölkopf, A. Smola, and K. Müller. Kernel principal component analysis. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Method: Support Vector Learning*, pages 327–352. MIT Press, 1999.
- [17] A. Shokoufandeh, D. Macrini, S. Dickinson, K. Siddiqi, and S. Zucker. Indexing hierarchical structures using graph spectra. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1125–1140, 2005.
- [18] B. Spillmann, M. Neuhaus, H. Bunke, E. Pekalska, and R. Duin. Transforming strings to vector spaces using prototype selection. In D. Yeung, J. Kwok, A. Fred, F. Roli, and D. de Ridder, editors, *Proc. 11th Int. Workshop on Structural and Syntactic Pattern Recognition*, LNCS 4109, pages 287–296. Springer, 2006.
- [19] R. Wagner and M. Fischer. The string-to-string correction problem. *Journal of the ACM*, 21(1):168–173, 1974.
- [20] R. Wilson, E. Hancock, and B. Luo. Pattern vectors from algebraic graph theory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1112–1124, 2005.