Graph Matching – Challenges and Potential Solutions

Horst Bunke, Christophe Irniger, and Michel Neuhaus

Institute of Computer Science and Applied Mathematics,
University of Bern, Neubrückstrasse 10, CH-3012 Bern, Switzerland
{bunke, irniger, mneuhaus}@iam.unibe.ch

Abstract. Structural pattern representations, especially graphs, have advantages over feature vectors. However, they also suffer from a number of disadvantages, for example, their high computational complexity. Moreover, we observe that in the field of statistical pattern recognition a number of powerful concepts emerged recently that have no equivalent counterpart in the domain of structural pattern recognition yet. Examples include multiple classifier systems and kernel methods. In this paper, we survey a number of recent developments that may be suitable to overcome some of the current limitations of graph based representations in pattern recognition.

Keywords: structural pattern recognition, graph matching, graph edit distance, automatic learning of cost functions, graph kernel methods, multiple classifier systems, graph database retrieval.

1 Introduction

The use of graph representations has become quite popular in pattern recognition. For an overview of recent literature, we refer the reader to a number of special issues that appeared on the subject [1,2,3,4] and the survey article [5]. Graph representations allow us to explicitly model relationships between different objects, or parts of the objects under consideration. This is a clear advantage over pattern representations based on feature vectors, which are restricted to using unary feature values. Moreover, in a graph representation one can use a variable number of entities, i.e. nodes and edges, depending on the complexity of the object under considerations. By contrast, in the statistical approach to pattern recognition, the number of features, i.e. the dimensionality of the feature space, needs to be defined a priori and is the same for all objects, no matter how simple or complex a particular object instance is.

On the other hand, there exist a number of problems with the use of structural pattern representation by means of graphs. First, we notice the high computational complexity of many operations on graphs. For example, computing the similarity of two graphs is typically exponential in the number of nodes of the two graphs involved, while the computation of the Euclidean distance of a pair of feature vectors needs only linear time. Secondly, the repository of algorithmic

F. Roli and S. Vitulano (Eds.): ICIAP 2005, LNCS 3617, pp. 1–10, 2005.

procedures in the graph domain is quite limited when compared to the tools available for pattern representations through feature vectors.

From the general point of view, we notice that in the field of statistical pattern recognition, a number of powerful concepts emerged recently, which have no equivalent counterparts in the domain of structural pattern recognition yet. One example is multiple classifier systems [6], which have turned out to be a very powerful concept to improve current pattern recognition technology. Another example is kernel methods [7]. While a large number of kernel methods based on pattern representations in terms of feature vectors have been proposed, relatively little work has been published on kernels on graphs and symbolic data structures [8]. Consequently, researchers in structural pattern recognition are faced with a number of challenges, for example, to overcome the high computational complexity of graph matching, to take advantage of new developments in the fast growing field of multiple classifier systems, to make the power of kernel methods available in the graph domain, and more generally, to enlarge the repository of algorithmic tools for graph matching. In the current paper, we will address some of these challenges and sketch some potential solutions.

The rest of this paper is structured as follows. In the next section we introduce a novel approach for the automatic learning of edit cost functions. Then, in Section 3, kernel methods for structural pattern recognition are investigated, while Section 4 reviews recent work on multiple classifier systems using graph matching. Strategies for fast retrieval of graphs from large database are presented in Section 5, and conclusions are drawn in Section 6.

2 Automatic Learning of Edit Cost Functions

There are many tasks in structural pattern recognition, where one needs to measure the similarity, or distance, of a given pair of graphs. A suitable measure for this purpose is graph edit distance [9,10]. In its most general form, a graph edit operation is either a deletion, insertion, or substitution (i.e. a label change). Edit operations can be applied to nodes as well as to edges. By means of edit operations differences between two graphs are modelled. In order to enhance the modelling capabilities, often a cost is assigned to each edit operation. The costs are real numbers greater than or equal to zero. They are application dependent. Typically, the more likely a certain distortion is to occur the lower is its costs. The edit distance, $d(g_1, g_2)$, of two graphs is equal to the minimum cost taken over all sequences of edit operations that transform graph g_1 into g_2. Clearly, if $g_1 = g_2$ then no edit operation is needed and $d(g_1, g_2) = 0$. On the other hand, the more g_1 and g_2 differ from each other, the more edit operations are needed, and the larger is $d(g_1, g_2)$.

It is easy to see that the costs of the edit operations have a crucial impact on $d(g_1, g_2)$ [11]. Despite this impact, we notice a significant algorithmic deficit in structural pattern recognition because no procedures for the automatic learning of graph matching edit costs from a set of samples are available. While some potential solutions have been proposed for the case of string edit distance [12],

the edit costs in graph matching are still manually set in a heuristic trial and error procedure, exploiting problem specific knowledge. For examples see [13,14]. In this section we briefly outline a novel procedure for the automatic learning of the costs of graph edit operations from a set of sample graphs [15].

As one of the basic assumptions of the learning scheme described in the following, graphs with labels from the n-dimensional real space are considered. That is, each node label is a vector of fixed dimension consisting of n real numbers. Similarly, edge labels consist of a vector of m real numbers. The proposed scheme takes a sample set of graphs as input and tries to minimize the average edit distance between each pair of graphs from the same class by suitably adjusting the costs of the underlying edit operations. This is equivalent to minimizing the average intra-class distance of a given set of graphs. The proposed scheme is based on self-organizing map, SOM [16]. There is one SOM for each type of edit operation, i.e. node deletion, node insertion, node substitution, edge deletion, edge insertion, and edge substitution. For example, the map for node substitution is a n-dimensional grid representing the space of node labels (i.e. n-dimensional real vectors). The cost of a node label substitution is proportional to the Euclidean distance between the two corresponding locations in the grid. The SOM learning procedure starts with a non-deformed n-dimensional grid. It computes the edit distance of a pair of graphs and moves each pair of grid points that correspond to a substitution closer to each other. In this way, the Euclidean distance of a pair of labels that are often substituted one by another, is iteratively minimized, which leads to a smaller overall graph edit distance between the two involved graphs. For more details we refer to [15].

The proposed learning scheme has been successfully applied in a recognition experiment involving synthetically generated characters. In addition the SOM-based learning method has been used for the identification of diatoms. Here graphs derived from real images of diatoms were involved and with the automatically derived costs a higher recognition accuracy than with manually chosen costs was achieved. Another procedure for automatically learning the costs of graph edit operations, based on the EM algorithm, has been proposed in [17].

3 Graph Matching Using Kernel Methods

Kernel methods have seen an increasing amount of interest in the pattern recognition community in recent years [7]. On one hand, the theory of kernel methods is well studied and provides us with a number of convenient theoretical results [18,19]. On the other hand, kernel machines have been successful in outperforming other types of classifiers on standard datasets [20]. Even more relevant to the present paper is that they also allow us to extend the repository of algorithms for structural pattern recognition in a very elegant way: once suitable kernel functions have been designed, a large number of methods for pattern classification, clustering, dimensionality reduction etc. become directly applicable to graphs.

With the rise of kernel machines in pattern recognition, a number of structural kernels for strings and graphs have been developed. A kernel for text categorization, for instance, has been developed that encodes the number of occurrences of words in texts [21]. In another approach, a kernel on string and graph data has been proposed that basically describes structures by means of the substructures they contain [22,23]. Another class of kernel functions defined on graphs are marginalized kernels [8,24], which are derived from random walks on attributed graphs. In [25] the authors propose a graph matching system for unconstrained large graphs based on functional interpolation theory. These methods can be characterized by explicitly addressing the matching problem via a kernel function. In this section we describe an alternative approach, originally proposed in [26], where we leave the structural matching to the error-tolerant edit distance algorithm and use kernel machines to subsequently carry out the classification task.

The kernel function introduced in [26] is defined as follows:

$$K(g_1, g_2) = \frac{1}{2}\left(d(g_1, g_0)^2 + d(g_0, g_2)^2 - d(g_1, g_2)^2\right),$$

where $d(g, g')$ is the edit distance between any two given graphs, g and g', and g_0 is a (reference) graph that needs to be defined by the system designer. We observe that this kernel computes the difference of two distances, the first being the squared distance of g_1 to g_0 plus the squared distance of g_0 to g_2, while the second is the direct squared distance of graphs g_1 and g_2.

In general it is not guaranteed that function $K(g_1, g_2)$ is a valid kernel. However, it has a number of interesting properties that justify its use as a kernel function. First, the distance of two vectors in the dot product space is equal to the edit distance of the corresponding graphs. Secondly, element g_0 plays the role of a null, or zero, element in the dot product space. The length of its corresponding vector is zero and the angle it forms with any other element in the dot product space is undefined. For a detailed analysis and a derivation of these properties see [26]. Given the kernel function defined above, one can build more complex kernels by selecting more than one zero graphs and combining the resulting kernels, for example, by their sum or product.

An exhaustive experimental evaluation of the proposed kernel method on eight different data sets has been reported in [26], including not only graph, but also string representations of patterns. In this experimental evaluation a support vector machine (SVM) using the proposed kernel was compared to a nearest neighbor classifier that works directly on the edit distances computed in the graph (or string) domain. This experimental evaluation has shown that the kernel based SVM very clearly outperforms the nearest neighbor classifier in the edit distance space.

4 Multiple Classifier Systems Using Graph Matching

Multiple classifier systems have become a very active field of research in pattern recognition [6]. The basic idea is to combine the output of several different classi-

fiers in order to build one large, and hopefully more accurate, multiple classifier system. From the general point of view, we can distinguish between two principal approaches to building a multiple classifier system. The first approach consists in developing each of the classifiers to be included in the system individually, i.e. the classifiers of the ensemble are built independently. Secondly, methods such as bagging, boosting, or random subspace are applied to one base classifier in order to generate the ensemble members automatically [27,28,29]. In this case only the base classifier needs to be constructed manually. An important issue in multiple classifier systems is diversity. Only if the individual members of the ensemble are diverse, i.e. independent of each other in the ideal case, one can expect that a multiple classifier system is able to outperform the best of its individual members.

Almost all work in the domain of multiple classifier systems has focussed on statistical, i.e. feature vector based, classifiers. However, because structural classifiers are usually very different from statistical ones, one can expect a substantial gain in diversity by including structural classifiers in an ensemble, i.e. by mixing statistical and structural classifiers in a multiple classifier system. Although this approach seems very appealing, little work of this kind has been reported [30,31]. Even more striking is the observation that almost no attempts are known from the literature to create ensembles of structural classifiers automatically from one given base classifier. In this section we first show how the performance of an image classification and retrieval system can be enhanced by mixing graph matching and feature vector based classifiers with each other [32]. Next a system for text classification based on an automatically generated ensemble of structural classifiers is described [33].

Image classification is an important task in image database retrieval and similar applications. Often binary classifiers are applied that have to decide, given an unknown image, whether or not it belongs to a certain predefined category, for example *people*, *countryside*, *city* etc. A number of classifiers have been proposed for that purpose. In [34], for example, a SVM using the values of the image histogram as features has been described. Such a classifier can be expected to perform well on cases where the salient features of an image category are of global nature and the discrimination of classes does not depend on local image regions. Also in [35] a SVM was used for the purpose of image classification. However, in contrast with [34], the presence or absence of particular types of regions is used as basic SVM features. The salient regions are automatically learned from a training set. This classifier is particularly meaningful when some region types are highly discriminative. For example, skin region types are a strong hint to images with people. However, this classifier certainly suffers if image segmentation is poor. As a third classifier, a graph matching based nearest neighbor classifier was proposed in [14]. Images are represented as region adjacency graphs, where region properties are used as node attributes, and properties such as the distance or the length of the common boundary between two adjacent regions serve as edge attributes. The graph edit distance between an unseen sample and each element of the training set is computed and the sample is assigned to its nearest

neighbor in the training set. In contrast with the other two classifiers, the graph based method takes spatial relationships and distance between pairs of regions into account.

Obviously, the three individual classifiers discussed in the previous paragraph are quite diverse. Hence it seems meaningful to combine then into a multiple classifier system. In order to accomplish such a combination, one needs to transform the individual classifiers' outputs into a common format. In [32] such a transformation is described. The resulting multiple classifier system was able to significantly outperform the best individual classifiers in a number of experiments under various combination rules.

Due to the explosion of material on the internet, the automatic classification of text has become a very important area of research. In a recent paper the use of graphs, rather than the standard vector model, has been proposed as a formal model for document representation [36]. The maximum common subgraph based distance measure introduced in [37] and a simple nearest neighbor classifier have been used in order to assign predefined text categories, such as *business*, *computer*, *sports*, *politics* etc. to previously unseen text documents. In an experimental evaluation it has been shown that this approach is able to outperform classifiers that use the traditional vector model of document representation.

In [33] this approach has been enhanced by automatically generating a multiple classifier system out of the base classifier used in [36]. The basic idea is to randomly select subsets of node labels (i.e. words from the underlying dictionary) an take, in the prototypes used by the nearest neighbor classifier, only those nodes into account that are labelled with an element from the chosen subset. This is in analogy to the random feature subset selection method proposed in [29] for the case of classifiers in a feature space. In an experimental evaluation it was shown that the resulting classifier ensemble, which has been automatically generated, is able to outperform the original graph matching based classifier.

5 Fast Retrieval of Graphs from Large Databases

It is well known that graph matching in general is a computationally expensive procedure. In many applications, particularly in pattern recognition and information retrieval, the computational complexity of graph matching is further increased by the need to match an input graph against an entire database of graphs. A variety of mechanisms have been proposed to reduce this additional factor [38,39,40,41]. However, they are generally unsuitable for processing large databases of graphs.

Recently, graph database retrieval has been addressed using machine learning techniques [42,43,44]. The idea is to preprocess the graph database extracting a feature vector representation of the graphs. Following feature vector extraction the database is then "data mined" using machine learning, specifically decision tree, techniques. At runtime, based on an input sample and a given matching paradigm (for example, graph isomorphism, subgraph isomorphism, or error-tolerant matching), all valid candidates are retrieved from the database

by traversing the decision tree previously induced. This approach is referred to as filtering the graph database. In filtering, the size of a database is reduced by first ruling out as many graphs as possible using a few simple and fast tests. After the filtering phase, an ordinary exact matching algorithm is applied to the remaining database graphs.

The matching paradigms considered in [42,43] include graph as well as subgraph isomorphism (both from the input graph to the database and from the database graphs to the input). For graph isomorphism, a necessary condition for two graphs g_s and g_{db} being isomorphic is that they have identical feature values. Given a sample graph g_s, the decision tree induced during preprocessing can be used to compare the most significant features identified on the database, ruling out a great number of graphs to be tested by a full-fledged isomorphism algorithm. The approach used for graph isomorphism filtering can be extended to subgraph isomorphism filtering in a straightforward way. The basic idea is that the feature values of the subgraph g_s occur at most as many times as they do in the designated supergraph g. For this matching paradigm, there exist two filtering methods. Decision trees induced for graph isomorphism filtering can be used to identify subgraph isomorphism candidates if the traversal algorithm is modified [43]. The other approach is to alter the decision tree structure, in which case the same traversal algorithm can be used for both tree types, graph isomorphism as well as subgraph isomorphism trees [42].

In [44], an extension of decision tree filtering to error-tolerant matching, i.e. retrieval of graphs with an edit distance to the input that is smaller than some given threshold distance, is presented. The approach is based on the concept of imposing a lower limit on the size of a possible maximum common subgraph between database graphs and input sample. According to the feature vector representation of the graphs an estimate on the size of the maximum common subgraph between input sample and graphs in the database is derived. If the estimated size is below the given input threshold it is certain that the distance between the graphs is larger than required and the graphs can be ruled out from the candidate set. In a number of experiments, this algorithm has been found to be very effective for database filtering [44].

6 Conclusions

Most of the algorithms commonly used in pattern recognition and intelligent data analysis are based on object representations in terms of feature vectors. This representation formalism is advantageous in the sense that a rich repository of algorithmic tools is available, including subspace projection techniques, methods for clustering and classification, and others. On the other hand, feature vectors are limited because they can represent only unary properties and usually assume that the same number of attribute values being measured on each object.

Symbolic data structures, including strings, trees, and graphs, are very suitable to overcome these limitations. However using such kind of object representation, we are facing the problem of increased computational complexity and

the lack of suitable mathematical tools for a number of tasks. To overcome these problems is a clear challenge to the community of researchers in pattern recognition.

Recent work has resulted in a number of promising approaches, making the repository of algorithmic tools for graph matching richer, for example, through procedures for the automatic learning of edit cost functions from sample sets of graphs. Work on graph kernels and graph matching based kernel functions is also of crucial importance for this purpose. With the availability of suitable kernels we may expect that a large class of powerful methods for intelligent data analysis, originally developed for feature representations, become instantly available in the graph domain. However, there is still a long way to go until mature kernel based methods for graphs and other structural data representations are at our disposal. Another challenging domain is multiple classifier system. While good progress in the field can be observed as far as statistical pattern recognition is concerned, the use of structural classifiers and their automatic generation in the context of multiple classifier systems is still in its infancy. We argue that also the problem of efficient graph retrieval from large databases is a hard problem, providing still a number of challenges.

Acknowledgment

The authors would like to thank Drs. Bertrand Le Saux, Adam Schenker, Mark Last, and Abe Kandel for fruitful cooperation and contributions to this paper. This research was supperted by the Swiss National Science Foundation NCCR program "Interactive Multimodal Information Management (IM)2" in the Individual Project "Multimedia Information Access and Content Protection" as well as the Project "Matching and Retrieval of Graphs from Large Graph Databases" (Nr. 2100-066700) of the Swiss National Science Foundation. The authors thank the Foundation for the support.

References

1. IEEE Transactions on Pattern Analysis and Machine Intelligence: Special section on graph algorithms and computer vision. **23** (2001) 1040–1151
2. Pattern Recognition Letters: Special issue on graph based representations. **24** (2003) 1033–1122
3. Int. Journal of Pattern Recognition and Art. Intelligence: Special issue on graph matching in pattern recognition and computer vision. **18** (2004) 261–517
4. Special Section on Syntactic and Structural Pattern Recognition: IEEE Transactions on Pattern Analysis and Machine Intelligence. **27** (2005) to appear
5. Conte, D., Foggia, P., Sansone, C., Vento, M.: Thirty years of graph matching in pattern recognition. Int. Journal of Pattern Recognition and Artificial Intelligence **18** (2004) 265–298
6. Roli, F., Kittler, J., Windeatt, T., eds.: Proc. 5th International Workshop on Multiple Classifier Systems MCS. LNCS 3077, Springer (2004)

7. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press (2004)
8. Gärtner, T.: A survey of kernels for structured data. ACM SIGKDD Explorations **5** (2003) 49–58
9. Bunke, H., Allermann, C.: Inexact graph matching for structural pattern recognition. Pattern Recognition Letters **1** (1983) 245–253
10. Sanfeliu, A., Fu, K.: A distance measure between attributed relational graphs for pattern recognition. IEEE Transactions on Systems, Man, and Cybernetics **13** (1983) 353–363
11. Bunke, H.: Error correcting graph matching: On the influence of the underlying cost function. IEEE Transactions on Pattern Analysis and Machine Intelligence **21** (1999) 917–922
12. Ristad, E., Yianilos, P.: Learning string-edit distance. IEEE Transactions on Pattern Analysis and Machine Intelligence **20** (1998) 522–532
13. Ambauen, R., Fischer, S., Bunke, H.: Graph edit distance with node splitting and merging, and its application to diatom identification. In Hancock, E., Vento, M., eds.: Proc. 4th IAPR Int. Workshop on Graph Based Representations in Pattern Recognition. LNCS 2726, Springer (2003) 95–106
14. Le Saux, B., Bunke, H.: Feature selection for graph-based image classifiers. In: Proc. 2nd Iberian Conf. on Pattern Recognition and Image Analysis IbPRIA. LNCS, Springer (2005) to appear
15. Neuhaus, M., Bunke, H.: Self-organizing maps for learning the edit costs in graph matching. IEEE Transactions on Systems, Man, and Cybernetics **35** (2005) to appear
16. Kohonen, T.: Self-organizing Maps. Springer Verlag (1995)
17. Neuhaus, M., Bunke, H.: A probabilistic approach to learning costs for graph edit distance. In Kittler, J., Petrou, M., Nixon, M., eds.: Proc. 17th Int. Conference on Pattern Recognition. Volume 3. (2004) 389–393
18. Vapnik, V.: Statistical Learning Theory. John Wiley (1998)
19. Schölkopf, B., Smola, A.: Learning with Kernels. MIT Press (2002)
20. Byun, H., Lee, S.: A survey on pattern recognition applications of support vector machines. Int. Journal of Pattern Recognition and Artificial Intelligence **17** (2003) 459–486
21. Joachims, T.: Learning to Classify Text Using Support Vector Machines. Kluwer Academic (2002)
22. Watkins, C.: Dynamic alignment kernels. In Smola, A., Bartlett, P., Schölkopf, B., Schuurmans, D., eds.: Advances in Large Margin Classifiers. MIT Press (2000) 39–50
23. Lodhi, H., Saunders, C., Shawe-Tayler, J., Cristianini, N., Watkins, C.: Text classification using string kernels. Machine Learning **2** (2002) 419–444
24. Kashima, H., Tsuda, K., Inokuchi, A.: Marginalized kernels between labeled graphs. In: Proc. 20th Int. Conf. on Machine Learning. (2003) 321–328
25. van Wyk, M.A., Durrani, T.S., van Wyk, B.J.: A RKHS interpolator-based graph matching algorithm. PAMI **24** (2002) 988–995
26. Neuhaus, M., Bunke, H.: Edit distance based kernel functions for structural pattern classification (2005) submitted.
27. Dietterich, T.: Ensemble methods in machine learning. In: Proc. 1st Int. Workshop on Multiple Classifier Systems. LNCS, Springer (2000) 1–15
28. Breiman, L.: Bagging predictors. Machine Learning **24** (1996) 123–140
29. Ho, T.: The random subspace method for constructing decision forests. IEEE Transactions on Pattern Analysis and Machine Intelligence **20** (1998) 832–844

30. Marcialis, G., Roli, F., Serrau, A.: Fusion of statistical and structural fingerprint classifiers. In: Proc. 4th International Conf. Audio- and Video-Based Biometrie Person Authentication AVBPA. (2003) 310–317
31. Serrau, A., Marcialis, G., Bunke, H., Roli, F.: An experimental comparison of fingerprint classification methods using graphs. In Brun, L., Vento, M., eds.: Proc. 5th IAPR Int. Workshop on Graph Based Representations in Pattern Recognition. LNCS 3434, Springer (2005) 281–290
32. Le Saux, B., Bunke, H.: Bayesian multiple classifier system for image content recognition (2005) submitted.
33. Schenker, A., Bunke, H., Last, M., Kandel, A.: Building graph-based classifier ensembles by random node selection. In Roli, F., Kittler, J., Windeatt, T., eds.: Proc. 5th Int. Workshop Multiple Classifier Systems MCS. LNCS 3077, Springer (2004) 214–222
34. Chapelle, O., Haffner, P., Vapnik, V.: Svms for histogram-based image classification. IEEE Transactions on Neural Networks **10** (1999) 1055–1065
35. Le Saux, B., Amato, G.: Image recognition for digital libraries. In: Proc. 6th ACM MultiMedia/Int. Workshop on Multimedia Information Retrieval. (2004) 91–98
36. Schenker, A., Last, M., Bunke, H., Kandel, A.: Classification of web documents using graph matching. Int. Journal of Pattern Recognition and Artificial Intelligence **18** (2004) 475–496
37. Bunke, H., Shearer, K.: A graph distance metric based on the maximal common subgraph. Pattern Recognition Letters **19** (1998) 255–259
38. Messmer, B., Bunke, H.: A new algorithm for error–tolerant subgraph isomorphism detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **20** (1998) 493–505
39. Shapiro, L., Haralick, R.: Organization of relational models for scene analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence **4** (1982) 595–602
40. Lopresti, D., Wilfong, G.: A fast technique for comparing graph representations with applications to performance evaluation. Int. Journal on Document Analysis and Recognition **6** (2004) 219–229
41. Giugno, R., Shasha, D.: Graphgrep: A fast and universal method for querying graphs. In: Proc. 16th Int. Conference on Pattern Recognition. Volume 2. (2002) 467–470
42. Irniger, C., Bunke, H.: Decision tree structures for graph database filtering. In Fred, A., ed.: Structural, Syntactic, and Statistical Pattern Recognition, Proc. Joint IAPR Int. Workshops SSPR and SPR. LNCS 3138, Springer (2004) 66–75
43. Irniger, C., Bunke, H.: Graph database filtering using decision trees. In Kittler, J., Petrou, M., Nixon, M., eds.: Proc. 17th Int. Conference on Pattern Recognition. Volume 3. (2004) 383–388
44. Irniger, C., Bunke, H.: Decision trees for error-tolerant graph database filtering. In Brun, L., Vento, M., eds.: Proc. 5th IAPR Int. Workshop on Graph Based Representations in Pattern Recognition. LNCS 3434, Springer (2005) 301–312