TR 03-016

Frequent Sub-Structure-Based Approaches for Classifying Chemical Compounds

Mukund Deshpande, Michihiro Kuramochi, and George Karypis

March 10, 2003

Frequent Sub-Structure-Based Approaches for Classifying Chemical Compounds[*]

Mukund Deshpande, Michihiro Kuramochi and George Karypis

University of Minnesota, Department of Computer Science/Army HPC Research Center
Minneapolis, MN 55455
Technical Report #03-016

{deshpand,kuram,karypis}@cs.umn.edu

Abstract

In this paper we study the problem of classifying chemical compound datasets. We present a sub-structure-based classification algorithm that decouples the sub-structure discovery process from the classification model construction and uses frequent subgraph discovery algorithms to find all topological and geometric sub-structures present in the dataset. The advantage of our approach is that during classification model construction, all relevant sub-structures are available allowing the classifier to intelligently select the most discriminating ones. The computational scalability is ensured by the use of highly efficient frequent subgraph discovery algorithms coupled with aggressive feature selection. Our experimental evaluation on eight different classification problems shows that our approach is computationally scalable and outperforms existing schemes by 10% to 35%, on the average.

Keywords: Classification, Chemical Compounds, Graphs, SVM

1 Introduction

Discovering new drugs is an expensive and challenging process. Any new drug should not only produce the desired response to the disease, but should do so with minimal side effects, and be superior to the existing drugs in the market. One of the key steps in the drug design process is to identify the chemical compounds (widely referred to as *"hit"* compounds) that display the desired and reproducible behavior against the disease [27] in a biological experiment. The standard technique to discover such compounds is to evaluate them with a biological experiment, known as an assay. The 1990s saw the widespread adoption of high-throughput screening (HTS), which use highly automated techniques to conduct the biological assays and can be used to screen a large number of compounds. Though in principle, HTS techniques can be used to test each compound against every biological assay, it is never practically feasible for the following reasons. First, the number of chemical compounds that have been synthesized or can be synthesized using combinatorial chemistry techniques is extremely large. Evaluating this large set of compounds using HTS can be prohibitively expensive. Second, not all biological assays can be converted to high throughput format. Third, in most cases it is hard to find all the desirable properties in a single compound, and chemists are interested not just identifying the hits but studying what part of the chemical compound leads to desirable behavior, so that new compounds can be rationally synthesized.

The goal of this paper is to develop computational techniques based on classification that can be used to identify the hit compounds. These computational techniques can be used to replace or supplement the biological assay techniques.

[*]This work was supported by NSF ACI-0133464, CCR-9972519, EIA-9986042, ACI-9982274, ACI-0133464, and by Army High Performance Computing Research Center contract number DAAH04-95-C-0008.

One of the key challenges in developing classification techniques for chemical compounds stems from the fact that their properties are strongly related to their chemical structure. However, traditional machine learning techniques are suited to handle datasets represented by multi-dimensional vectors or sequences, and cannot handle the structural nature of the chemical structures.

In recent years two classes of techniques have been developed for solving the chemical compound classification problem. The first class builds a classification model using a set of physico-chemical properties derived from the compounds structure, called quantitative structure-activity relationships (QSAR) [15, 16, 1], whereas the second class operates directly on the structure of the chemical compound and try to automatically identify a small number of chemical sub-structures that can be used to discriminate between the different classes [3, 43, 18, 25]. A number of comparative studies [40, 20] have shown that techniques based on the automatic discovery of chemical sub-structures are superior to those based on QSAR properties and require limited user intervention and domain knowledge. However, despite their success, a key limitation of these techniques is that they rely on heuristic search methods to discover these sub-structures. Even though such approaches reduce the inherently high computational complexity associated with these schemes, they may lead to sub-optimal classifiers in cases in which the heuristic search failed to uncover sub-structures that are critical for the classification task.

In this paper we present a sub-structure-based classifier that overcomes the limitations associated with existing algorithms. One of the key ideas of our approach is to decouple the sub-structure discovery process from the classification model construction step and use frequent subgraph discovery algorithms to find all chemical sub-structures that occur a sufficiently large number of times. Once the complete set of these sub-structures has been identified, our algorithm then proceeds to build a classification model based on them. The advantage of such an approach is that during classification model construction, all relevant sub-structures are available allowing the classifier to intelligently select the most discriminating ones. To ensure that such an approach is computationally scalable, we use recently developed [23, 25] highly efficient frequent subgraph discovery algorithms coupled with aggressive feature selection to reduce both the amount of time required to build as well as to apply the classification model. In addition, we present a sub-structure discovery algorithm that finds a set of sub-structures whose geometry is conserved, further improving the classification performance of our algorithm.

We experimentally evaluated the performance of our algorithms on eight different problems derived from three publically available datasets and compared their performance against that of traditional QSAR-based classifiers and existing sub-structure classifiers based on SUBDUE [4] and SubdueCl [14]. Our results show that our approach, on the average, outperforms QSAR-based schemes by 35% and SUBDUE-based schemes by 10%.

2 Background

A chemical compound consists of different atoms being held together via bonds adopting a well-defined geometric configuration. Figure 1(a) represents the chemical compound Flucytosine from the DTP AIDS repository [9] it consists of a central aromatic ring and other elements like N, O and F. The representation shown in the figure is a typical graphical representation that most chemists work with.

There are many different ways to represent such chemical compounds. The simplest representation is the molecular formula that lists the various atoms making up the compound; the molecular formula for Flucytosine is `C4H4FN3O`. However this representation is woefully inadequate to capture the structure of the chemical compound. It was recognized early on that it was possible for two chemical compounds to have identical molecular formula but completely different chemical properties [13]. A more sophisticated representation can be achieved using the SMILES [6] representation, it not only represents the atoms but also represents the bonds between different atoms. The SMILES representation for Flucytosine is `Nc1nc(O)ncc1F`. Though SMILES representation is a compact it is not guaranteed to be unique, furthermore the representation is quite restrictive to work with [22].

The activity of a compound largely depends on its chemical structure and the arrangement of different atoms in 3D space. As a result, effective classification algorithms must be able to directly take into account the structural nature of these datasets.

In this paper we represent each compound as undirected graphs, this is the most generic representation for a graph. The vertices of these graphs correspond to the various atoms, and the edges correspond to the bonds between the atoms. Each one of the vertices and edges of has a label associated with it. The labels on the vertices correspond to the type of atoms, and the labels on the edges correspond to the type of bond. Furthermore, 3D coordinated are associated with

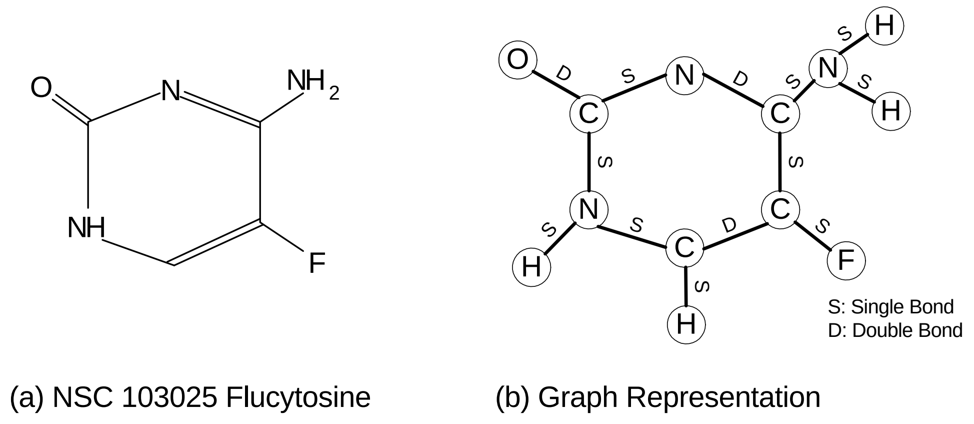

Figure 1: Chemical and Graphical representation of Flucytosine

each vertex indicating the position of the corresponding atom in 3D space[1]. The graph representation of Flucytosine (without the 3D coordinates) is shown in Figure 1(b). A graph in which all the vertices have 3D coordinates associated with them is referred as a ***geometric graph***, whereas a graph without the 3D coordinates is referred as ***topological graph***. Such topological/geometric representations are quite commonly used by many chemical modeling software and are referred as the *connection table* for the chemical compound.

The meaning of the various classes in the input dataset is application dependent. In some applications, the classes will capture the extent to which a particular compound is toxic, whereas in other applications they may capture the extent to which a compound can inhibit (or enhance) a particular factor and/or active site. In most applications each of the compounds is assigned to only one of two classes, that are commonly referred to as the *positive* and *negative* class. The positive class corresponds to compounds that exhibit the property in question, whereas the compounds of the negative class do not. Throughout this paper we will be restricting ourselves to only two classes, though all the techniques described here can be easily extended to multi-class as well as multi-label classification problems.

Another important aspect of modeling chemical compounds is the naming of single and double bonds inside aromatic rings. Typically in an aromatic ring of a chemical compound though the number of single and double bonds is fixed, the exact position of double and single bonds is not fixed, this is because of the phenomenon of resonance [13]. It is worth noting that the exact position of double and single bond in an aromatic ring does not affect the chemical properties of a chemical compound. To capture this uncertainty in the position of single and double bond we represent all the bonds making up the aromatic ring with a new bond type called the *aromatic bond*. Another aspect of the chemical compounds is that the number of hydrogen bonds connected to a particular carbon atom can usually be inferred from the bonds connecting that carbon atom [13], therefore in our representation we do not represent the hydrogen atoms that are connected to the carbon atoms, such hydrogen atoms are referred as non-polar hydrogen atoms.

3 Related Research

In the early 1960s, the pioneering work of Hansch *et al*. [15, 16] demonstrated that the biological activity of a chemical compound is a function of its physico-chemical properties. These physico-chemical properties are usually derived from the compound's structure and are called quantitative structure-activity relationships (QSAR). Examples of such physico-chemical properties include the molecular weight, total energy, dipole moment, solvent accessible area, *etc*. Over the years a number of different QSAR properties have been developed (GAUSSIAN contains over 50) and they are used extensively to model and analyze chemical compounds within the pharmaceutical industry. The amount of time required to compute these QSAR properties varies from property to property. Some of them can be computed very fast (*e.g.*, molecular weight), while others require time-consuming numerical simulations (*e.g.*, dipole-moment, total energy) that can only be performed for small datasets.

In QSAR-based classification methods, each chemical compound is transformed into a vector of numerical values, corresponding to the various QSAR properties. After this transformation any traditional classifier capable of handling numerical features can be used for the classification task. Early research on QSAR-based classification methods

[1]We use the software package Corina [12] to compute the 3D coordinates.

focused primarily on regression-based techniques [15, 11]; however, more sophisticated classifiers have also been used. For example, Andrea and Kalayeh [2] show that neural networks can achieve better accuracies over regression-based techniques, whereas An and Wang [1] report that decision tree classifiers applied on QSAR features outperform those based on neural networks and logistic regression.

The key challenge in using QSAR-based approaches stems from the fact that the classification performance relies, to a large extent, on the a priori identification of the relevant QSAR properties that capture the structure-activity relationships for the particular classification problem. Identifying this relevant set of QSAR properties requires considerable domain expertise and extensive experimentation. To overcome this problem, a different set of techniques have been developed that operate directly on the structure of the chemical compound and try to automatically identify a small number of chemical sub-structures that can be used to discriminate between the different classes.

One of the earlier approaches that follow this paradigm are based on inductive logic programming (ILP) [33]. In this approach the chemical compound is expressed using first order logic. Each atom is represented as a predicate consisting of atomID and the element, and a bond is represented as a predicate consisting of two atomIDs. Using this representation, an ILP system discovers rules (*i.e.*, conjunction of predicates) that are good for discriminating the different classes. Since these rules consist of predicates describing atoms and bonds, they essentially correspond to sub-structures that are present in the chemical compounds. The pioneering work in this field was done by King *et al*. in the early 1990s [21, 20]. They applied an ILP system, Golem [34], to study the behavior of 44 trimethoprin analogues and their observed inhibition of *Escherichia coli* dihydrofolate reductase. They reported an improvement in classification accuracy over the traditional QSAR-based models. Srinivasan *et al*. [40] present a detailed comparison of the features generated by ILP with the traditional QSAR features used for classifying chemical compounds. They show that for some applications features discovered by ILP approaches lead to a significant lift in the performance. Besides improved classification performance, an additional advantage of these structure-based approaches is that the discovered rules (*i.e.*, sub-structures) can be easily understood by experts and could be used to check the correctness of the model and to provide insights in the chemical behavior of the compounds.

Though ILP-based approaches are quite powerful, the high computational complexity of the underlying rule-induction system limits the size of the dataset for which they can be applied. Furthermore, they tend to produce rules consisting of relatively small sub-structures (usually three to four atoms [5, 7]), limiting the size of structural constraints that are being discovered and hence affecting the classification performance. Another drawback of these approaches is that in order to reduce their computational complexity they employ various heuristics to prune the explored search-space [32], potentially missing sub-structures that are important for the classification task. One exception is the WARMR system [5, 7] that is specifically developed for chemical compounds and discovers all possible sub-structures above a certain frequency threshold. However, WARMR's computational complexity is very high and can only be used to discover sub-substructures that occur with relatively high frequency.

One of the fundamental reasons limiting the scalability of ILP-based approaches is the first order logic-based representation that they use. This representation is much more powerful than what is needed to model chemical compounds and discover sub-structures. For this reason a number of researchers have explored the much simpler graph-based representation of the chemical compound's topology and transformed the problem of finding chemical sub-structures to that of finding subgraphs in this graph-based representation [3, 43, 18]. Probably the most well-known approach is the SUBDUE system [17, 4]. SUBDUE finds patterns which can effectively compress the original input data based on the minimum description length (MDL) principle, by substituting those patterns with a single vertex. To narrow the search-space and improve its computational efficiency, SUBDUE uses a heuristic beam search approach, which quite often results in failing to find subgraphs that are frequent. The SUBDUE system was also later extended to classify graphs and was referred as SubdueCL [14]. In SubdueCL instead of using minimum description length as a heuristic a measure similar to confidence of a subgraph is used as a heuristic.

Finally, another heuristics based scheme targeted for chemical compounds is MOLFEA [22]. In this scheme each chemical compound is represented as a SMILES string, and is thought of as sequence of SMILES objects. This representation simplifies the problem to discovering frequently occurring sub-sequences.

4 Classification Based on Frequent Subgraphs

The previous research on classifying chemical compounds (discussed in Section 3) has shown that techniques based on the automatic discovery of chemical sub-structures are superior to those based on QSAR properties and require limited user intervention and domain knowledge. However, despite their success, a key limitation of both the ILP- and the subgraph-based techniques, is that they rely on heuristic search methods to discover the sub-structures to be used

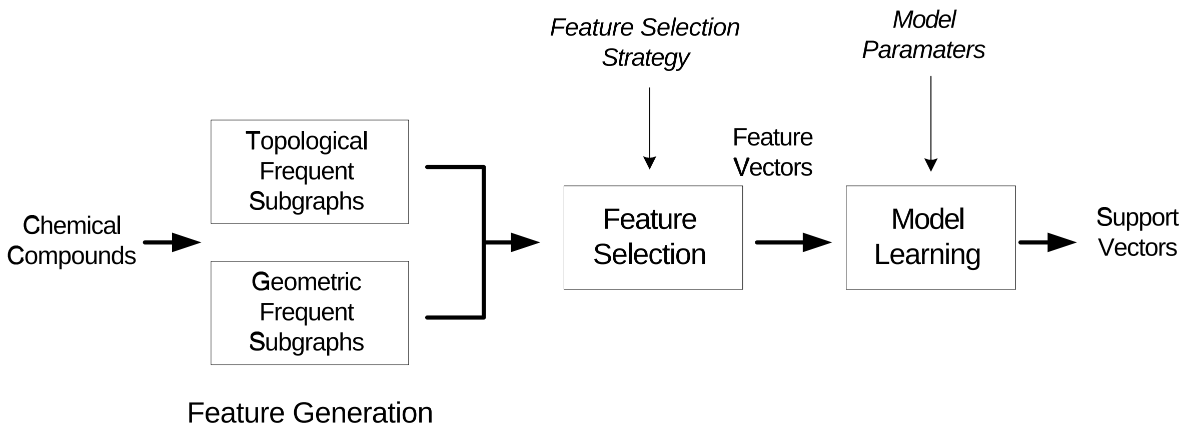

Figure 2: Frequent Subgraph Based Classification Framework

for classification. As discussed in Section 3, even though such approaches reduce the inherently high computational complexity associated with these schemes, they may lead to sub-optimal classifiers in cases in which the heuristic search failed to uncover sub-structures that are critical for the classification task.

To overcome this problem, we developed a classification algorithm for chemical compounds that uses the graph-based representation and limits the number of sub-structures that are pruned a priori. The key idea of our approach is to decouple the sub-structure discovery process from the classification model construction step, and use frequent subgraph discovery algorithms to find all chemical sub-structures that occur a sufficiently large number of times. Once the complete set of such sub-structures has been identified, our algorithm then proceeds to build a classification model based on them. To a large extent, this approach is similar in spirit to the recently developed frequent-itemset-based classification algorithms [29, 28, 8] that have been shown to outperform traditional classifiers that rely on heuristic search methods to discover the classification rules.

The overall outline of our classification methodology is shown in Figure 2. It consists of three distinct steps: (i) feature generation, (ii) feature selection, and (iii) classification model construction. During the feature generation step, the chemical compounds are mined to discover the frequently occurring sub-structures that correspond to either topological or geometric subgraphs. These sub-structures are then used as the features by which the compounds are represented in the subsequent steps. During the second step, a small set of features is selected such that the selected features can correctly discriminate between the difference classes found in the dataset. Finally, in the last step each chemical compound is represented using these set of features and a classification model is learnt. The rest of this section describes these three steps in detail.

4.1 Feature Generation

Our classification algorithm finds sub-structures in a chemical compound database using two different methods. The first method uses the topological graph representation of each compound whereas the second method is based on the corresponding geometric graph representation (discussed in Section 2). In both of these methods, our algorithm uses the topological or geometric connect subgraphs that occur in at least σ% of the compounds to define the sub-structures.

There are two important restrictions on the type of the sub-structures that are discovered by our approach. The first has to do with the fact that we are only interested in sub-structures that are connected and is motivated by the fact that connectivity is a natural property of such patterns (**this needs work**). The second has to do with the fact that we are only interested in frequent sub-structures (as determined by the value of σ) and this ensures that the discovered sub-structures are statistically significant and not spurious. Furthermore, this minimum support constraint also helps in making the problem of frequent subgraph discovery computationally tractable.

4.1.1 Frequent Topological Subgraphs

Developing frequent subgraph discovery algorithms is particularly challenging and computationally intensive as graph and/or subgraph isomorphisms play a key role throughout the computations. Despite that, in recent years, four different algorithms have been developed capable of finding all frequently occurring subgraphs with reasonable computational efficiency. These are the AGM algorithm developed by Inokuchi et al [18], the FSG algorithm developed by members of our group [23], the chemical sub-structure discovery algorithm developed by Borgelt and Berthold [3], and the gSpan algorithm developed by Yan and Han [43]. The enabling factors to the computational efficiency of these schemes have been (i) the development of efficient candidate subgraph generation schemes that reduce the number of times the same candidate subgraph is being generated, (ii) the use of efficient canonical labeling schemes to represent the various subgraphs; and (iii) the use of various techniques developed by the data-mining community to reduce the number of times subgraph isomorphism computations need to be performed.

In our classification algorithm we find the frequently occurring subgraphs using the FSG algorithm. FSG takes as input a database D of graphs and a minimum support σ, and finds all connected subgraphs that occur in at least σ% of the transactions. FSG, initially presented in [23], with subsequent improvements presented in [25], uses a breadth-first approach to discover the lattice of frequent subgraphs. It starts by enumerating small frequent graphs consisting of one and two edges and then proceeds to find larger subgraphs by joining previously discovered smaller frequent subgraphs. The size of these subgraphs is grown by adding one-edge-at-a-time. The lattice of frequent patterns is used to prune the set of candidate patterns, and it only explicitly computes the frequency of the patterns which survived this downward closure pruning. Despite the inherent complexity of the problem, FSG employs a number of sophisticated techniques to achieve high computational performance. It uses a canonical labeling algorithm that fully makes use of edge and vertex labels for fast processing, and various vertex invariants to reduce the complexity of determining the canonical label of a graph. These canonical labels are then used to establish the identity and total order of the frequent and candidate subgraphs, a critical step of redundant candidate elimination and downward closure testing. It uses a sophisticated scheme for candidate generation [25] that minimizes the number of times each candidate subgraph gets generated and also dramatically reduces the generation of subgraphs that fail the downward closure test. Finally, for determining the actual frequency of each subgraph, FSG reduces the number of subgraph isomorphism operations by using TID-lists [10, 38, 45, 44] to keep track of the set of transactions that supported the frequent patterns discovered at the previous level of the lattice. For every candidate, FSG takes the intersection of TID-lists of its parents, and performs the subgraph isomorphism only on the transactions contained in the resulting TID-list. As the experiments presented in Section 5 show, FSG is able to scale to large datasets and low support values.

4.1.2 Frequent Geometric Subgraphs

Topological sub-structures capture the connectivity of atoms in the chemical compound but they ignore the 3D shape (3D arrangement of atoms) of the sub-structures. For certain classification problems the 3D shape of the sub-structure might be essential for determining the chemical activity of a compound. For instance, the geometric configuration of atoms in a sub-structure is crucial for its ability to bind to a particular target [27]. For this reason we developed algorithms that find all frequent sub-structures whose topology as well as geometry is conserved.

There are two important aspects specific to the geometric subgraphs that need to be considered. First, since the coordinates of the vertices depend on a particular reference coordinate axes, we would like the discovered geometric subgraphs to be independent of these coordinate axes, *i.e.*, we are interested in geometric subgraphs whose occurrences are translation, and rotation invariant. This dramatically increases the overall complexity of the geometric subgraph discovery process, because we may need to consider all possible geometric configurations of a single pattern. Second, while determining if a geometric subgraph is contained in a bigger geometric graph we would like to allow some tolerance when we establish a match between coordinates, ensuring that slight deviations in coordinates between two identical topological subgraphs do not lead to a creation two geometric subgraphs. The amount of tolerance (r) should be a user specified parameter. The task of discovering such r-tolerant frequent geometric subgraphs dramatically changes the nature of the problem. In traditional pattern discovery problems such as finding frequent itemsets, sequential patterns, and/or frequent topological graphs there is a clear definition of what a pattern is, given its set of supporting transactions. On the other hand, in the case of r-tolerant geometric subgraphs, there are many different geometric representations of the same pattern (all of which will be r-tolerant isomorphic to each other). The problem becomes not only that of finding a pattern and its support, but also finding the right representative for this pattern. The selection of the right representative can have a serious impact on correctly computing the support of the pattern. For example, given a set of subgraphs that are r-tolerant isomorphic to each other, the one that corresponds to an *outlier* will tend to have a lower support than the one corresponding to the *center*. These two aspects of geometric subgraphs makes the task of discovering the full fledged geometric subgraphs extremely hard [24].

To overcome this problem we developed a simpler, albeit less discriminatory, representation for geometric subgraphs. We use a property of a geometric graph called the ***average inter-atomic distance***, that is defined as the average Euclidean distance between all pairs of atoms in the molecule. Note that the average inter-atomic distance is computed between all pairs of atoms irrespective of whether a bonds connects the atoms or not. The average inter-atomic distance can be thought of as a geometric signature of a topological subgraph. The geometric subgraph consists of two components, a topological subgraph and an interval of average inter-atomic distance associated with it. A geometric graph contains this geometric subgraph if it contains the topological subgraph and the average inter-atomic distance of the embedding (of the topological subgraph) is within the interval associated with the geometric subgraph. Note that this geometric representation is also translation and rotation invariant, and the width of the interval determines the tolerance displayed by the geometric subgraph. We are interested in discovering such geometric subgraphs that occur above $\sigma\%$ of the transactions and the interval of average inter-atomic distance is bound by r.

Since a geometric subgraph contains a topological subgraph, for the geometric subgraph to be frequent the corresponding topological subgraph has to be frequent, as well. This allows us to take advantage of the existing approach to discover topological subgraphs. We modify the frequency counting stage of the FSG algorithm as follows. If a subgraph g is contained in a transaction t then all possible embeddings of g in t are found and the average inter-atomic distance for each of these embeddings is computed. This list of average inter-atomic distances is added to the $g.aiadList$ such that at the end of the frequent subgraph discovery each topological subgraph has a list of average inter-atomic distances associated with it. Each one of the average inter-atomic distances corresponds to one of the embeddings *i.e.*, a geometric configuration of the topological subgraph. The task of discovering geometric subgraphs now reduces to identifying those geometric configurations that are frequent enough, *i.e.*, to identify intervals of average inter-atomic distance such that each interval contains the minimum number geometric configurations (σ) and the width of the interval is less than the tolerance threshold (r). In our experiments we set the value of r to be equal to the half of the minimum distance between any two pair of atoms in the compounds.

This task can be thought of as 1D clustering on the vector of average inter-atomic distances such that each cluster contains items above the minimum support and the spread of each cluster is bounded by the tolerance r. Note that not all items will belong to a valid cluster as some of them will be infrequent.

To find such clusters we perform agglomerative clustering on the vector of average inter-atomic distance values. The distance between any two average inter-atomic distance values is defined as the difference in their numeric values. To ensure that we get the largest possible clusters we use the maximum-link criterion function for deciding which two clusters should be merged. The process of agglomeration is continued till the interval containing all the items in the cluster is below the tolerance threshold (r), if we reach a stage where further agglomeration would increase the spread of the cluster beyond the tolerance threshold, we check the number of items contained in the cluster. If the number of items is above the support threshold, then the interval associated with this cluster is considered as a geometric feature. Note that since we are clustering one-dimensional datasets, the clustering complexity is low.

Note that this algorithm for computing geometric subgraphs is approximate in nature for two reasons. First, the average inter-atomic distance may map two different geometric subgraphs to the same average inter-atomic distance value. Second, the clustering algorithm may not find the complete set of geometric subgraphs that satisfy the r tolerance. Nevertheless as our experiments in Section 5 show the geometric subgraphs discovered by this approach improve the classification accuracy of the algorithm.

4.1.3 Additional Considerations

Even though FSG provides the general functionality required to find all frequently occurring sub-structures in chemical datasets, there are a number of issues that need to be addressed before it can be applied as a black-box tool for feature discovery in the context of classification. One issue deals with the selecting the right value for the σ, the support constraint used for discovering frequent sub-structures. The value of σ controls the number of subgraphs discovered by FSG. Choosing a good value of σ is especially important for the dataset containing classes of significantly different sizes. In such cases, in order to ensure that FSG is able to find features that are meaningful for all the classes, it must use a support that depends on the size of the smaller class.

For this reason we first partition the complete dataset, using the class label of the examples, into specific class specific datasets. We then run FSG on each of these *class datasets*. This partitioning of the dataset ensures that sufficient subgraphs are discovered for those class labels which occur rarely in the dataset. Next, we combine subgraphs discovered from each of the *class dataset*. After this step each subgraph has a vector that contains the frequency with which it occurs in each class.

4.2 Feature Selection

The frequent subgraph discovery algorithms described in Section 4.1 discovers all the sub-structures (topological or geometric) that occur above a certain support constraint (σ) in the dataset. Though the discovery algorithm is computationally efficient, the algorithm can generate a large number of features. A large number of features is detrimental for two reasons. First, it could increase the time required to build the model. But more importantly, a large number of features increases the time required to a classify a chemical compound; because before we can classify a chemical compound, we need to identify whether or not the discovered features set contained in the chemical compound. This involves applying the costly graph isomorphism operation for all the discovered features. This problem is especially critical in the drug discovery process where the classification model is learnt on a small set of chemical compounds and it is then applied on large chemical compound libraries containing millions of compounds.

To overcome this problem we developed a feature selection schemes that select a small set of features, from those discovered by the frequent subgraph discovery algorithms, without compromising the classification accuracy of the model. The feature selection scheme used in this paper is based on the ***sequential covering paradigm*** used to learn rule sets [30]. To apply this algorithm we assume that each discovered sub-structure corresponds to a rule, with the class label of the sub-structure as the *target attribute*, such rules are referred as *class-rules* in [29]. The sequential covering algorithm takes as input a set of examples and the features discovered from these examples, and iteratively applies the feature selection step. In this step the algorithm selects the feature that has the highest estimated accuracy. After selecting this feature all the examples containing this feature are eliminated and the feature is marked as selected. In the next iteration of the algorithm the same step is applied, but on a smaller set of examples. The algorithm continues in an iterative fashion till either all the features are selected or all the examples are eliminated.

In this paper we use a computationally efficient implementation of sequential covering algorithm known as CBA [29], this algorithm proceeds by first sorting the features based on confidence and then applying the sequential covering algorithm on this sorted set of features. One of the advantages of this approach is that it requires minimal number of passes on the dataset, hence is very scalable. To obtain a better control over the number of selected features we use an extension of the sequential covering scheme known as *Classification based on Multiple Rules* (CMAR). In this scheme instead of removing the example after it is covered by the selected feature, the example is removed only if that example is covered by δ selected features. The number of selected rules increases as the value of δ increases, an increase in the number of features usually translates into an improvement in the accuracy as more features are used to classify a particular example. The value of δ is specified by the user and provides a means to the user to control the number of features used for classification.

4.3 Classification Model Construction

Given the frequent subgraphs discovered in the previous step, our algorithm treats each of these subgraphs as a feature and represents the chemical compound as a frequency vector. The ith entry of this vector is equal to the number of times (frequency) that feature occurs in the compound's graph. This mapping into the feature space of frequent subgraphs is performed both for the training and the test dataset. Note that the frequent subgraphs were identified by mining *only* the graphs of the chemical compounds in the training set. However, the mapping of the test set requires that we check each frequent subgraph against the graph of the test compound using subgraph isomorphism. Fortunately, the overall process can be substantially speeded up by taking into account the frequent subgraph lattice that is also generated by FSG. In this case, we traverse the lattice from top to bottom and only visit the child nodes of a subgraph if that subgraph is isomorphic to the chemical compound.

Once the feature vectors for each chemical compound have been built, any one of the existing classification algorithms can potentially be used for classification. However, the characteristics of the transformed dataset and the nature of the classification problem itself tends to limit the applicability of certain classes of classification algorithms. In particular, the transformed dataset will most likely be high dimensional, and second, it will be sparse, in the sense that each compound will have only a few of these features, and each feature will be present in only a few of the compounds. Moreover, in most cases the positive class will be much smaller than the negative class, making it unsuitable for classifiers that primarily focus on optimizing the overall classification accuracy.

In our study we built the classification models using support vector machines (SVM) [41], as they are well-suited for operating in such sparse and high-dimensional datasets. Furthermore, an additional advantage of SVM is that it allows us to directly control the cost associated with the miss-classification of examples from the different classes [31]. This allow us to associate a higher cost for the miss-classification of positive instances; thus, biasing the classifier to learn a model that tries to increase the true-positive rate, at the expense of increasing the false positive rate.

5 Experimental Evaluation

We experimentally evaluated the performance of our classification algorithm and compared it against that achieved by earlier approaches on a variety of chemical compound datasets. The datasets, experimental methodology, and results are described in subsequent sections.

5.1 Datasets

We used three different publicly available datasets to derive a total of eight different classification problems. The first dataset was initially used as a part of the Predictive Toxicology Evaluation Challenge [39] which was organized as a part of PKDD/ECML 2001 Conference[2]. It contains data published by the U.S. National Institute for Environmental Health Sciences, the data consists of bio-assays of different chemical compounds on rodents to study the carcinogenicity (cancer inducing) properties of the compounds [39]. The goal being to estimate the carcinogenicity of different compounds on humans. Each compound is evaluated on four kinds of laboratory animals (*male Mice*, *female Mice*, *male Rats*, *female Rats*), and is assigned four class labels each indicating the toxicity of the compound for that animal. There are four classification problems one corresponding to each of the rodents and will be referred as *P1*, *P2*, *P3*, and *P4*.

The second dataset is obtained from the National Cancer Institute's DTP AIDS Anti-viral Screen program [9, 22] [3]. Each compound in the dataset is evaluated for evidence of anti-HIV activity. The screen utilizes a soluble formazan assay to measure protection of human CEM cells from HIV-1 infection [42]. Compounds able to provide at least 50% protection to the CEM cells were re-tested. Compounds that provided at least 50% protection on retest were listed as *moderately active* (CM, confirmed moderately active). Compounds that reproducibly provided 100% protection were listed as *confirmed active* (CA). Compounds neither active nor moderately active were listed as *confirmed inactive* (CI). We have formulated three classification problems on this dataset, in the first problem we consider only *confirmed active* (CA) and *moderately active* (CM) compounds and then build a classifier to separate these two compounds; this problem is referred as *H1*. For the second problem we combine *moderately active* (CM) and *confirmed active* (CA) compounds to form one set of *active* compounds, we then build a classifier to separate these *active* and *confirmed inactive* compounds; this problem is referred as *H2*. In the last problem we only use *confirmed active* (CA) and *confirmed inactive* compounds and build a classifier to categorize these two compounds; this problem is referred as *H3*.

The third dataset was obtained from the Center of Computational Drug Discovery's anthrax project at the University of Oxford [37]. The goal of this project was to discover small molecules that would bind with the heptameric protective antigen component of the anthrax toxin, and prevent it from spreading its toxic effects. A library of small sized chemical compounds was screened to identify a set of chemical compounds that could bind with the anthrax toxin. The screening was done by computing the binding free energy for each compound using numerical simulations. The screen identified a set of 12,376 compounds that could potentially bind to the anthrax toxin and a set of 22,460 compounds that were unlikely to bind to the chemical compound. The average number of vertices in this dataset is 25 and the average number of edges is also 25. The classification problem for this dataset was given a chemical compounds classify it in to one of these two classes, *i.e*, will the compound bind the anthrax toxin or not. This classification problem is referred as *A1*.

Some important characteristics of these datasets are summarized in Table 1. The right hand side of the table displays the class distribution for different classification problems, for each problem the table displays the percentage of positive class found in the dataset for that classification problem. Note that both the DTP-AIDS and the Anthrax datasets are quite large containing 42,687 and 34,836 compounds, respectively. Moreover, in the case of DTP-AIDS, each compound is also quite large having on an average 46 atoms and 48 bonds.

5.2 Experimental Methodology & Metrics

The classifications results were obtained by performing 5-way cross validation on the dataset, ensuring that the class distribution in each fold is identical to the original dataset. For the SVM classifier we used SVMLight library [19]. All the experiments were conducted on a 1500MHz Athlon MP processors having a 2GB of memory.

[2]http://www.informatik.uni-freiburg.de/˜ml/ptc/.

[3]http://dtp.nci.nih.gov/docs/aids/aids_data.html.

	Toxic.	*Aids*	*Anthrax*	*Class Dist.* (% +ve class)	
N	417	42,687	34,836	**Toxicology**	
$\bar{N}_A$	25	46	25	P1: Male Mice	38.3%
$\bar{N}_B$	26	48	25	P2: Female Mice	40.9%
$\bar{L}_A$	40	82	25	P3: Male Rats	44.2%
$\bar{L}_B$	4	4	4	P4: Female Rats	34.4%
max N_A	106	438	41	**AIDS**	
min N_A	2	2	12	H1: CA/CM	28.1%
max N_B	1	276	44	H2: (CA+CM)/CI	3.5%
min N_B	85	1	12	H3: CA/CI	1.0%
				Anthrax	
				A1: active/inactive	35%

Table 1: The characteristics of the various datasets. N is the number of compounds in the database. $\bar{N}_A$ and $\bar{N}_B$ are the average number of atoms and bonds in each compound. $\bar{L}_A$ and $\bar{L}_B$ are the average number of atom- and bond-types in each dataset. max N_A/min N_A and max N_B/min N_B are the maximum/minimum number of atoms and bonds over all the compounds in each dataset.

Since the size of the positive class is significantly smaller than the negative class, using *accuracy* to judge a classifier would be incorrect. To get a better understanding of the classifier performance for different cost settings we obtain the ROC curve [35] for each classifier. ROC curve plots the false positive rate (X-axis) versus the true positive rate (Y-axis) of a classier; it displays the performance of the classifier without regard to class distribution or error cost. Two classifiers are compared by comparing the area under their respective ROC curves, a larger area under ROC curve indicating better performance. The area under the ROC curve will be referred by the parameter A.

5.3 Results

Varying Minimum Support The key parameter of the proposed frequent sub-structure-based classification algorithm is the choice of the minimum support (σ) used to discover the frequent sub-structures (either topological or geometric). To evaluate the sensitivity of the algorithm on this parameter we performed a set of experiments in which we varied σ from 10% to 20% in 5% increments. The results of these experiments are shown in the left sub-table of Table 2 for both topological and geometric sub-structures.

Dset.	*σ=10.0%*				*σ = 15.0%*				*σ = 20.0%*				Optimized σ					
	Topo.		*Geom.*		*Topo.*		*Geom.*		*Topo.*		*Geom.*		*Topo.*		*Geom.*		Per class	$Time_p$
	A	N_f	A	N_f	A	N_f	A	N_f	A	N_f	A	N_f	A	N_f	A	N_f	σ	(sec)
P1	66.0	1211	65.5	1317	66.0	513	64.1	478	64.4	254	60.2	268	65.5	24510	65.0	23612	3.0, 3.0	211
P2	65.0	967	64.0	1165	65.1	380	63.3	395	64.2	217	63.1	235	67.3	7875	69.9	12673	3.0, 3.0	72
P3	60.5	597	60.7	808	59.4	248	61.3	302	59.9	168	60.9	204	62.6	7504	64.8	10857	3.0, 3.0	66
P4	54.3	275	55.4	394	56.2	173	57.4	240	57.3	84	58.3	104	63.4	25790	63.7	31402	3.0, 3.0	231
H1	81.0	27034	82.1	29554	77.4	13531	79.2	8247	78.4	7479	79.5	7700	81.0	27034	82.1	29554	10.0, 10.0	137
H2	70.1	1797	76.0	3739	63.6	307	62.2	953	59.0	139	58.1	493	76.5	18542	79.1	29024	10.0, 5.0	1016
H3	83.9	27019	89.5	30525	83.6	13557	88.8	11240	84.6	7482	87.7	7494	83.9	27019	89.5	30525	10.0, 10.0	392
A1	78.2	476	79.0	492	78.2	484	77.6	332	77.1	312	76.1	193	81.7	3054	82.6	3186	5.0, 3.0	145

Table 2: Varying minimum support threshold (σ). "A" denotes the area under the ROC curve and "N_f" denotes the number of discovered frequent subgraphs.

From Table 2 we observe that as we increase σ, the classification performance for most datasets tends to degrade. However, in most cases this degradation is gradual and correlates well with the decrease on the number of sub-structures that were discovered by the frequent subgraph discovery algorithms. The only exception is the H2 problem for which the classification performance (as measured by ROC) degrades substantially as we increase the minimum support from 10% to 20%. Specifically, in the case of topological subgraphs, the performance drops from 70.1 down to 59.0, and in the case of geometric subgraphs it drops from 76.0 to 58.1.

These results suggest that lower values of support are in general better as they lead to better classification performance. However, as the support decreases, the number of discovered sub-structures and the amount of time required also increases. Thus, depending on the dataset, some experimentation may be required to select the proper values of support that balances these conflicting requirements (*i.e.*, low support but reasonable number of sub-structures).

In our study we performed such experimentation. For each dataset we kept on decreasing the value of support down to the point after which the number of features that were generated was too large to be efficiently processed by the SVM library. The resulting support values, number of features, and associated classification performance are shown in the right sub-table of Table 2 under the table header "Optimized σ". Note that for each problem two different support values are displayed corresponding to the supports that were used to mine the positive and negative class, respectively.

Also, the last column shows the amount of time required by FSG to find the frequent subgraphs and provides a good indication of the computational complexity at the feature discovery phase of our classification algorithm.

Comparing the ROC values obtained in these experiments with those obtained for $\sigma = 10\%$, we can see that as before, the lower support values tend to improve the results, with measurable improvements for problems in which the number of discovered sub-structures increased substantially. In the rest of our experimental evaluation we will be using the frequent subgraphs that were generated using these values of support.

Varying Misclassification Costs Since for each classification problem instance the number of positive examples is in general much smaller than the number of negative examples, we performed a set of experiments in which the misclassification cost associated with each positive example was increased to match the number of negative examples. That is, if n^+ and n^- is the number of positive and negative examples, respectively, the misclassification cost β was set equal to $(n^-/n^+ - 1)$ (so that $n^- = \beta n^+$). We refer to this value of β as the *"EqCost"* value. The classification performance achieved by our algorithm using either topological or geometric subgraphs for $\beta = 1.0$ and $\beta = EqCost$ is shown in Table 3. Note that the $\beta = 1.0$ results are the same with those presented in the right subtable of Table 2.

Dataset	Topo		Geom	
	$\beta = 1.0$	$\beta = EqCost$	$\beta = 1.0$	$\beta = EqCost$
P1	65.5	65.3	65.0	66.7
P2	67.3	66.8	69.9	69.2
P3	62.6	62.6	64.8	64.6
P4	63.4	65.2	63.7	66.1
H1	81.0	79.2	82.1	81.1
H2	76.5	79.4	79.1	81.9
H3	83.9	90.8	89.5	94.0
A1	81.7	82.1	82.6	83.0

Table 3: The area under the ROC curve obtained by varying the misclassification cost. "$\beta = 1.0$" indicates the experiments in which each positive and negative example had a weight of one, and "$\beta = EqCost$" indicates the experiments in which the misclassification cost of the positive examples was increased to match the number of negative examples.

From the results in this table we can see that, in general, increasing the misclassification cost so that it balances the size of positive and negative class tends to improve the classification accuracy. When $\beta = EqCost$, the classification performance improves for four and five problems for the topological and geometric subgraphs, respectively. Moreover, in the cases in which the performance decreased, that decrease was quite small, whereas the improvements achieved for some problem instances (*e.g.*, P4, H1, and H2) was significant. In the rest of our experiments we will focus only on the results obtained by setting $\beta = EqCost$.

Feature Selection We evaluated the performance of the feature selection scheme based on sequential covering (described in Section 4.2) by performing a set of experiments in which we varied the parameter δ that controls the number of times an example must be covered by a feature, before it is removed from the set of yet to be covered examples. Table 4 displays the results of these experiments. The results under the column labeled "Original" shows the performance of the classifier without any feature selection. These results are identical to those shown in Table 3 for $\beta = EqCost$ and are included here to make comparisons easier.

Two key observations can be made by studying the results in this table. First, as expected, the feature selection scheme is able to substantially reduce the number of features. In some cases the number of features that was selected decreased by almost two orders of magnitude. Also, as δ increases, the number of retained features increases; however, this increase is gradual. Second, the overall classification performance achieved by the feature selection scheme when $\delta \geq 5$ is quite comparable to that achieved with no feature selection. The actual performance depends on the problem instance and whether or not we use topological or geometric subgraphs. In particular, for the first four problems (P1, P2, P3, and P4) derived from the PTC dataset, the performance actually improves with feature selection. Such improvements are possible even in the context of SVM-based classifiers as models learned on lower dimensional spaces will tend to have better generalization ability [8]. Also note that for some datasets the number of features decreases as δ increases. This is because the features that were selected have higher average support.

Topological versus Geometric Subgraphs The various results shown in Tables 2–4 also provide an indication on the relative performance of topological versus geometric subgraphs. In almost all cases, the classifier that is based on geometric subgraphs outperforms that based on topological subgraphs. For some problems, the performance advantage is marginal whereas for other problems, geometric subgraphs lead to measurable improvements in the

Topological Features										
Dataset.	*Original*		$\delta = 1$		$\delta = 5$		$\delta = 10$		$\delta = 15$	
	A	N_f	*A*	N_f	*A*	N_f	*A*	N_f	*A*	N_f
P1	65.3	24510	65.4	143	66.4	85	66.5	598	66.7	811
P2	66.8	7875	69.5	160	69.6	436	68.0	718	67.5	927
P3	62.6	7504	68.0	171	65.2	455	64.2	730	64.5	948
P4	65.2	25790	66.3	156	66.0	379	64.5	580	64.1	775
H1	79.2	27034	78.4	108	79.2	345	79.1	571	79.5	796
H2	79.4	18542	77.1	370	78.0	1197	78.5	1904	78.5	2460
H3	90.8	27019	88.4	111	89.6	377	90.0	638	90.5	869
A1	82.1	3054	80.6	620	81.4	1395	81.5	1798	81.8	2065
Geometric Features										
Dataset.	*Original*		$\delta = 1$		$\delta = 5$		$\delta = 10$		$\delta = 15$	
	A	N_f	*A*	N_f	*A*	N_f	*A*	N_f	*A*	N_f
P1	66.7	23612	68.3	161	68.1	381	67.4	613	68.7	267
P2	69.2	12673	72.2	169	73.9	398	73.1	646	73.0	265
P3	64.6	10857	71.1	175	70.0	456	71.0	241	66.7	951
P4	66.1	31402	68.8	164	69.7	220	67.4	609	66.2	819
H1	81.1	29554	80.8	128	81.6	396	81.9	650	82.1	885
H2	81.9	29024	80.0	525	80.4	1523	80.6	2467	81.2	3249
H3	94.0	30525	91.3	177	92.2	496	93.1	831	93.2	1119
A1	83.0	3186	81.0	631	82.0	1411	82.4	1827	82.7	2106

Table 4: Results obtained using feature selection based on sequential rule covering. "δ" specifies the number of times each example needs to be covered before it is removed, "*A*" denotes the area under the ROC curve and "N_f" denotes the number of features that were used for classification.

Property	Dim.	Property	Dim.
Solvent accessible area	$Å^2$	Moment of Inertia	*none*
Total accessible area	$Å^2$	Total energy	$kcal/mol$
Total accessible volume	$Å^3$	Bend energy	$kcal/mol$
Total Van der Waal's area	$Å^2$	Hbond energy	$kcal/mol$
Total Van der Waal's volume	$Å^3$	Stretch energy	$kcal/mol$
Dipole moment	*Debye*	Nonbond energy	$kcal/mol$
Dipole moment comp. (X, Y, Z)	*Debye*	Estatic energy	$kcal/mol$
Heat of formation	*Debye*	Torsion energy	$kcal/mol$
Multiplicity	$Kcal$	Quantum total charge	eV

Table 5: QSAR Properties.

area under the ROC curve. For example, if we consider the results shown in Table 3 for $\beta = EqCost$, we can see the geometric subgraphs lead to improvements that are at least 3% or higher for P2, P3, and H3, and the average improvement over all eight problems is 2.6%. As discussed in Section 4.1.2, these performance gains is due to the fact that conserved geometric structure is a better indicator of a chemical compounds activity than just its topology.

5.4 Comparison with Other Approaches

We compared the performance of our classification algorithm against the performance achieved by the QSAR-based approach and the approach that uses the SUBDUE system to discover a set of sub-structures.

Comparison with QSAR As discussed in Section 3 there is a wide variety of QSAR properties each of which captures certain aspects of a compounds chemical activity. For our study, we have chosen a set of 18 QSAR properties that are good descriptors of the chemical activity of a compound and most of them have been previously used for classification purposes [1]. A brief description of these properties are shown in Table 5. We used two programs to compute these attributes, the geometric attributes like solvent accessible area, total accessible area/vol, total Van der Waal's accessible area/vol were computed using the programs SASA [26], the remaining attributes were computed using Hyperchem software.

We used two different algorithms to build classification models based on these QSAR properties. The first is the C4.5 decision tree algorithm [36] that has been shown to produce good models for chemical compound classification based on QSAR properties [1], and the second is the SVM algorithm that was used to build the classification models in our frequent sub-structure-based approach. Since the range of values of the different QSAR properties can be significantly different, we first scaled them to be in the range of [0, 1] prior to building the SVM model. We found that this scaling resulted in some improvements in the overall classification results. Note that C4.5 is not affected by such scaling.

Table 6 shows the results obtained by the QSAR-based methods for the different datasets. The values shown

Dataset	*SVM*	*C4.5*		*Freq. Sub. Prec.*	
	A	Precision	Recall	*Topo*	*Geom*
P1	60.2	0.4366	0.1419	0.6972	0.6348
P2	59.3	0.3603	0.0938	0.8913	0.8923
P3	55.0	0.6627	0.1275	0.7420	0.7427
P4	45.4	0.2045	0.0547	0.6750	0.8800
H1	64.5	0.5759	0.1375	0.7347	0.7316
H2	47.3	0.6282	0.4071	0.7960	0.7711
H3	61.7	0.5677	0.2722	0.7827	0.7630
A1	49.4	0.5564	0.3816	0.7676	0.7798

Table 6: Performance of the QSAR-based Classifier.

Dataset	*Subdue.*			*SubdueCL.*		
	A	N_f	$Time_p$	A	N_f	$Time_p$
P1	61.9	1288	303sec	63.5	2103	301sec
P2	64.2	1374	310sec	63.3	2745	339sec
P3	57.4	1291	310sec	59.6	1772	301sec
P4	58.5	1248	310sec	60.8	2678	324sec
H1	74.2	1450	1,608sec	73.8	960	1002sec
H2	58.5	901	232,006sec	65.2	2999	476,426sec
H3	71.3	905	178,343sec	77.5	2151	440,416sec
A1	75.3	983	56,056sec	75.9	1094	31,177sec

Table 7: Performance of the SUBDUE and SubdueCl-based approaches.

for SVM correspond to the area under the ROC curve and can be directly compared with the corresponding values obtained by our approaches (Tables 2–4). Unfortunately, since C4.5 does not produce a ranking of the training set based on its likelihood of being in the positive class, it is quite hard to obtain the ROC curve. For this reason, the values shown for C4.5 correspond to the precision and recall of the positive class for the different datasets. Also, to make the comparisons between C4.5 and our approach easier, we also computed the precision of our classifier at the same value of recall as that achieved by C4.5. These results are shown under the columns labeled "*Freq. Sub. Prec.*" for both topological and geometric features and were obtained from the results shown in Table 3 for $\beta = EqCost$.

Comparing both the SVM-based ROC results and the precision/recall values of C4.5 we can see that our approach substantially outperforms the QSAR-based classifier. In particular, our topological subgraph based algorithm does 35% better compared to SVM-based QSAR and 72% better in terms of the C4.5 precision at the same recall values. Similar results hold for the geometric subgraph based algorithm. These results are consistent with those observed by other researchers [40, 20] that showed that sub-structure based approaches outperform those based on QSAR properties.

Comparison with SUBDUE & SubdueCL Finally, to evaluate the advantage of using the complete set of frequent sub-structures over existing schemes that are based on heuristic sub-structure discovery, we performed a series of experiments in which we used the SUBDUE system to find the sub-structures and then used them for classification. Specifically, we performed two sets of experiments. In the first set, we obtain a set of sub-structures using the standard MDL-based heuristic sub-structure discovery approach of SUBDUE [17]. In the second set, we used the sub-structures discovered by the more recent SubdueCl algorithm [14] that guides the heuristic beam search using a scheme that measures how well a subgraph describes the positive examples in the dataset without describing the negative examples.

Even though there are a number of parameters controlling SUBDUE's heuristic search algorithm, the most critical among them are the width of the beam search, the maximum size of the discovered subgraph, and the total number of subgraphs to be discovered. In our experiments, we spent a considerable amount of time experimenting with these parameters to ensure that SUBDUE was able to find a reasonable number of sub-structures. Specifically, we changed the width of the beam search from 4 to 50 and set the other two parameters to high numeric values. Note that in the case of the SubdueCl, in order to ensure that the subgraphs were discovered that described all the positive examples, the subgraph discovery process was repeated by increasing the value of beam-width at each iteration and removing the positive examples that were covered by subgraphs.

Table 7 shows the performance achieved by SUBDUE and SubdueCl on the eight different classification problems along with the number of subgraphs that it generated and the amount of time that it required to find these subgraphs. These results were obtained by using the subgraphs discovered by either SUBDUE or SubdueCl as features in an SVM-based classification model. Essentially, our SUBDUE and SubdueCl classifiers have the same structure as our frequent subgraph-based classifiers with the only difference being that the features now correspond to the subgraphs discovered by SUBDUE and SubdueCl. Moreover, to make the comparisons as fair as possible we used $\beta = EqCost$

as the misclassification cost. We also performed another set of experiments in which we used the rule-based classifier produced by SubdueCl. The results of this scheme was inferior to those produced by the SVM-based approach and we are not reporting them here.

Comparing SUBDUE against SubdueCl we can see that the later achieves better classification performance, consistent with the observations made by other researchers [14]. Comparing the SUBDUE and SubdueCl-based results with those obtained by our approach (Tables 2–4) we can see that in almost all cases both our topological and geometric frequent subgraph-based algorithms lead to substantially better performance. This is true both in the cases in which we performed no feature selection as well as in the cases in which we used the sequential covering based feature selection scheme. In particular, comparing the SubdueCl results against the results shown in Table 4 without any feature selection we can see that on the average, our topological and geometric subgraph based algorithms do 9.3% and 12.2% better, respectively. Moreover, even after feature selection with $\delta = 15$ that result in a scheme that have comparable number of features as those used by SubdueCl, our algorithms are still better by 9.7% and 13.7%, respectively. Finally, if we compare the amount of time required by either SUBDUE or SubdueCl to that required by the FSG algorithm to find all frequent subgraphs (last column of Table 2 we can see that despite the fact that we are finding the complete set of frequent subgraphs our approach requires substantially less time.

6 Conclusion

In this paper we presented a highly-effective algorithm for classifying chemical compounds based on frequent sub-structure discovery that can scale to large datasets. Our experimental evaluations showed that our algorithm leads to substantially better results than those obtained by existing QSAR- and sub-structure-based methods.

References

[1] A. An and Y. Wang. Comparisons of classification methods for screening potential compounds. In *ICDM*, 2001.

[2] T. A. Andrea and Hooshmand Kalayeh. Applications of neural networks in quantitative structure-activity relationships of dihydrofolate reductase inhibitors. *Journal of Medicinal Chemistry*, 34:2824–2836, 1991.

[3] Michael R. Berthold Christian Borgelt. Mining molecular fragments: Finding relevant substructures of molecules. In *Proc. of the (ICDM)*, 2002.

[4] D. J. Cook and L. B. Holder. Graph-based data mining. *IEEE Intelligent Systems*, 15(2):32–41, 2000.

[5] King Ross D., Ashwin Srinivasan, and L. Dehaspe. Warmr: A data mining tool for chemical data. *Journal of Computer Aided Molecular Design*, 15:173–181, 2001.

[6] Weininger D. Smiles 1. introduction and encoding rules. *J. of Chemical Information and Computer Sciences*, 28, 1988.

[7] L. Dehaspe, H. Toivonen, and R. D. King. Finding frequent substructures in chemical compounds. In *4th International Conference on Knowledge Discovery and Data Mining*, 1998.

[8] Mukund Deshpande and George Karypis. Using conjunction of attribute values for classification. In *Proceedings of the eleventh CIKM*, pages 356–364. ACM Press, 2002.

[9] dtp.nci.nih.gov. Dtp aids antiviral screen dataset.

[10] B. Dunkel and N. Soparkar. Data organizatinon and access for efficient data mining. In *Proc. of the 15th IEEE ICDE*, March 1999.

[11] J. H. Frank, I.E.& Friedman. A statistical view of some chemometrics regression tools. *Technometrics*, 35:109–148, 1993.

[12] J. Gasteiger, C. Rudolph, and J. Sadowski. Automatic generation of 3d-atomic coordinates for organic molecules. *Tetrahedron Comp. Method*, 3:537–547, 1990.

[13] T. A. Geissman. *Principles of Organic Chemistry*. W. H. Freeman and Company, 1968.

[14] J. Gonzalez, L. Holder, and D. Cook. Application of graph based concept learning to the predictive toxicology domain. In *PTC, Workshop at the 5th PKDD*, 2001.

[15] C. Hansch, P. P. Maolney, T. Fujita, and R. M. Muir. Correlation of biological activity of phenoxyacetic acids with hammett substituent constants and partition coefficients. *Nature*, 194:178–180, 1962.

[16] C. Hansch, R. M. Muir, T. Fujita, C. F. Maloney, and Streich M. The correlation of biological activity of plant growth-regulators and chloromycetin derivatives with hammett constants and partition coefficients. *Journal of American Chemical Society*, 85:2817–1824, 1963.

[17] L. B. Holder, D. J. Cook, and S. Djoko. Substructure discovery in the SUBDUE system. In *Proc. of the AAAI Workshop on Knowledge Discovery in Databases*, pages 169–180, 1994.

[18] A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In 4^{th} *PKDD*, 2000.

[19] T. Joachims. *Advances in Kernel Methods: Support Vector Learning*, chapter Making large-Scale SVM Learning Practical. MIT-Press, 1999.

[20] Ross D. King, Stephen H. Muggleton, Ashwin Srinivasan, and Michael J. E. Sternberg. Strucutre-activity relationships derived by machine learning: The use of atoms and their bond connectivities to predict mutagenecity byd inductive logic programming. *Proceedings of National Acadamey of Science*, 93:438–442, January 1996.

[21] Ross D. King, Stepher Muggleton, Richard A. Lewis, and J. E. Sternberg. Drug design by machine learning: The use of inductive logic programming to model the sturcture-activity relationships of trimethoprim analogues binding to dihydrofolate reductase. *Proceedings of National Acadamey of Science*, 89:11322–11326, December 1992.

[22] S. Kramer, L. De Raedt, and C. Helma. Molecular feature mining in hiv data. In *7th International Conference on Knowledge Discovery and Data Mining*, 2001.

[23] M. Kuramochi and G. Karypis. Frequent subgraph discovery. In *ICDM*, 2001.

[24] Michihiro Kuramochi and George Karypis. Discovering geometric frequent subgraph. In *IEEE International Conference on Data Mining*, 2002.

[25] Michihiro Kuramochi and George Karypis. An efficient algorithm for discovering frequent subgraphs. Technical Report TR# 02-26, Dept. of Computer Science and Engineering, University of Minnesota, 2002.

[26] S. M. Le Grand and J. K. M. Merz. Rapid approximation to molecular surface area via the use of booleean logic look-up tables. *J. of Computational Chemistry*, 14:349–352, 1993.

[27] Andrew R. Leach. *Molecular Modeling, Principles and Applications.* Prentice Hall, 2001.

[28] Wenmin Li, Jiawei Han, and Jian Pei. Cmar: Accurate and efficient classification based on multiple class-association rules. In *ICDM*, 2001.

[29] Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In *International Conference on Knowledge Discovery and Data Mining*, 1998.

[30] Tom M. Mitchell. *Machine Learning.* Mc Graw Hill, 1997.

[31] K. Morik, P. Brockhausen, and T. Joachims. Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring. In *International Conference on Machine Learning*, 1999.

[32] S. Muggleton. Inverse entailment and Progol. *New Generation Computing*, 13:245–286, 1995.

[33] Stephen Muggleton and L. De Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19(20):629–679, 1994.

[34] Stephen H. Muggleton and C. Feng. Efficient induction of logic programs. In Stephen Muggleton, editor, *Inductive Logic Programming*, pages 281–298. Academic Press, London, 1992.

[35] F. Provost and T. Fawcett. Robust classification for imprecise environments. *Machine Learning*, 42(3), 2001.

[36] J. Ross Quinlan. *C4.5: Programs for machine learning.* Morgan Kaufmann, San Mateo, CA, 1993.

[37] Graham W. Richards. Virtual screening using grid computing: the screensaver project. *Nature Reviews: Drug Discovery*, 1:551–554, July 2002.

[38] Pradeep Shenoy, Jayant R. Haritsa, S. Sundarshan, Gaurav Bhalotia, Mayank Bawa, and Devavrat Shah. Turbo-charging vertical mining of large databases. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pages 22–33, May 2000.

[39] A. Srinivasan, R. D. King, S. H. Muggleton, and M. Sternberg. The predictive toxicology evaluation challenge. In 15^{th} *IJCAI*, 1997.

[40] Ashwin Sriniviasan and Ross King. Feature construction with inductive logic programming: a study of quantitative predictions of biological activity aided by structural attributes. *Knowledge Discovery and Data Mining Journal*, 3:37–57, 1999.

[41] V. Vapnik. *Statistical Learning Theory*. John Wiley, 1998.

[42] O.S Weislow, R. Kiser, D. L Fine, J. P. Bader, R. H. Shoemaker, and M. R. Boyd. New soluble fomrazan assay for hiv-1 cyopathic effects: appliication to high flux screening of synthetic and natural products for aids antiviral activity. *Journal of National Cancer Institute*, 1989.

[43] Xifeng Yan and Jiawei Han. gSpan: Graph-based substructure pattern mining. In *ICDM*, 2002.

[44] Mohammed J. Zaki and Karam Gouda. Fast vertical mining using diffsets. Technical Report 01-1, Department of Computer Science, Rensselaer Polytechnic Institute, 2001.

[45] Mohammed Javeed Zaki. Scalable algorithms for association mining. *TKDE*, 12(2):372–390, 2000.