Learning Large Margin First Order Decision Lists for Multi-class Classification

Huma Lodhi[1], Stephen Muggleton[1], and Mike J E Sternberg[2]

[1] Department of Computing,Imperial College London,SW7 2AZ
hml@doc.ic.ac.uk, shm@doc.ic.ac.uk
[2] Centre for Bioinformatics, Imperial College London, SW7 2AZ
m.sternberg@imperial.ac.uk

Abstract. Inductive Logic Programming (ILP) systems have been successfully applied to solve binary classification problems. It remains an open question how an accurate solution to a multi-class problem can be obtained by using a logic based learning method. In this paper we present a novel logic based approach to solve challenging multi-class classification problems. Our technique is based on the use of large margin methods in conjunction with the kernels constructed from first order rules induced by an ILP system. The proposed approach learns a multi-class classifier by using a divide and conquer reduction strategy that splits multi-classes into binary groups and solves each individual problem recursively hence generating an underlying decision list structure. We also study the well known one-vs-all scheme in conjunction with logic-based kernel learning. In order to construct a highly informative logical and relational space we introduce a low dimensional embedding method. The technique is amenable to skewed/non-skewed class distribution where multi-class problems such as protein fold recognition are generally characterized by highly uneven class distribution. We performed a series of experiments to evaluate the proposed rule selection and multi-class schemes. The methods were applied to solve challenging problems in computation biology and bioinformatics, namely multi-class protein fold recognition and mutagenicity detection. Experimental comparisons of the performance of large margin first order decision list based multi-class scheme with the standard multi-class ILP algorithm and multi-class Support Vector Machine yielded statistically significant results. The results also demonstrated a favorable comparison between the performances of decision list based scheme and one-vs-all strategy.

1 Introduction

The underlying aim of a multi-class approach is to learn a highly accurate function that categorizes examples into predefined classes. Effective multi-class techniques are crucial to solving the problems ranging from multiple object recognition to multi-class protein fold recognition.

The two areas of machine learning, namely Inductive Logic Programming (ILP) and Kernel based methods (KMs) are well known for their distinguishing

features: ILP techniques are characterized by their use of background knowledge and expressive language formalism whereas strong mathematical foundations and high generalization ability are remarkable characteristics of KMs. Recently some logic based techniques (such as Support Vector Inductive Logic Programming (SVILP) [1], kFOIL [2] and RUMBLE [3]) have been designed which use kernels to solving binary classification problems and performing real-valued predictions. In this paper we study multi-class classification in the combined ILP and kernel learning scenario by extending SVILP. We also propose an effective method to constructing highly informative relational and logical low dimensional feature space. The method is designed in a way so as a classifier trained in the feature space is amenable to highly imbalance category distribution. A skewed class distribution is a common phenomenon in multi-class classification tasks.

SVILP solves binary classification problems in a multi-stage learning process. In the first stage, a set of all the first order horn clauses (rules), constructed during the search of the hypothesis space, is obtained from an ILP system. In the next stages similarity between the examples is computed by the use of novel kernel function that captures semantic and structural commonalities between examples. The computed relational and logic based kernel is used in conjunction with a large margin learning algorithm to induce a binary classifier. In this way, SVILP performs classification task by training a large margin first order classifier.

SVILP [1] uses all the clauses with positive compression: an information theoretic measure. The number of positively compressed rules can vary from zero to thousands for the particular task. Furthermore rules with negative compression can contain crucial information to solving the problem at hand. This scenario can cause a decrease in generalization performance of the learning machine. In order to handle such issues we extend SVILP by introducing a novel rule selection method where the selected rules can have very high information content, generalization ability and can handle class imbalance problem.

In order to solve multi-class problems we propose a simple but accurate approach. The method is designed by reducing the multi-class classification task to binary problems. However our approach is distinguished from the existing reduction techniques as it learns the hidden structure and characteristics of the data and hence improves the performance of the classifier. The proposed method is based on divide-and-conquer strategy and it discriminates different classes by using an underlying structure based on decision lists. The multi-class problem is reduced by recursively breaking it down into binary problems where each binary task is solved by invoking an SVILP machine. At each node of the decision list the algorithm induces a classifier and updates the training set by removing the examples of the class chosen at the previous node. A label is assigned to a new example by traversing the list. We also study the well known one-vs-all scheme in conjunction with SVILP.

During recent years, a number of multi-class classification method have been proposed [4–8]. The focus of the methods has been on the construction of different effective multi-class schemes whereas less attention has been paid to manip-

ulating the hidden structures and characteristics of the data by using expressive representations. In ILP, which is well known for its use of expressive language formalism, the standard method to solve multi-class problems is based upon inducing a set of disjunctive rules for each class and a new example is predicted if it satisfies the conditions of the rules. In the case that multiple classes are assigned to an example, that is common in ILP, the method is biased towards majority class. Within ILP algorithms, the use of decision lists [9] was explored by Mooney and Califf [10] for binary concept learning. The method extended FOIL [11] by incorporating intensional background knowledge and it is characterized by it's ability to induce logic programs without explicitly taking negative examples as input. The logic program generated by the technique comprised ordered list of clauses (rules). The method was successfully applied to the complex problem of learning past tense of English verbs.

In order to evaluate the performance of proposed methods, we conducted a series of experiments. We applied the techniques to solving multi-class protein fold recognition problem and binary class mutagenicity detection and identification task. The results show that the techniques yield substantial and significant improvements in performance.

2 Multi-class Inductive Logic Programming (MC_ILP)

ILP systems have been successfully applied to binary classification tasks in computational biology, bioinformatics, and chemoinformatics. There are few ILP systems that can perform multi-class classification tasks [12]. The standard multi-class logic based method, described below, is biased towards the majority class. The method is based on learning theories H_j(first order horn clauses) for each class j. The obtained theories for r classes are merged into a multi-theory H. For each class the number of correctly classified training examples are recorded. A class is assigned to a new example if the example satisfies the conditions of the rules. In the case that an example is predicted to have multiple classes, then the class with the maximum number of predicted training examples is assigned to the example. If an example fails to satisfy the conditions of all rules in H, a default class (majority class) is assigned to it. The method is termed as multi-class ILP (MC_ILP).

3 Support Vector Inductive Logic Programming

Support Vector Inductive Logic Programming [1] is a new machine learning technique that is at the intersection of Inductive Logic Programming and Support Vector Machines [13]. SVILP extends ILP with SVMs where the similarity between the examples is measured by computing an inner product on the subset of rules induced by an ILP system. It can be viewed as a multi-stage learning algorithm. The four stages that comprise SVILP learning are described as follows.

In the first stage a set of rules $\mathcal{H}$ is obtained from an ILP system that takes relationally encoded examples (positive, negative) and background knowledge as input. The set, $\mathcal{H}$, comprises all the rules constructed during the search of the hypothesis space. This stage maps the examples into a logic based relational space. A first order rule, $h \in \mathcal{H}$, can be viewed as a boolean function of the form, $h : D \rightarrow \{0, 1\}$.

In the next stage a subset $H \in \mathcal{H}$ is selected by using an information theoretic measure, namely compression, described below. The stage maps the examples into another lower dimensional space containing the information relevant to the task at hand. The compression value of a rule is computed by the expression, $C = \frac{PT*(ps-(ng+cl))}{ps}$, where ps is the number of positive examples correctly deducible from the rule, ng is the number of negative examples that satisfy the conditions of the rules, cl is the length of the rule and PT is the total number of positive examples.

In the third stage a kernel function is defined on the selected set of rules where rules can be weighted/unweighted. The kernel is based on the idea of comparing two examples by means of structural and relational features they contain; the more features in common the more similar they are. The function is given by the inner product between the mapped examples where the mapping ϕ is implied by the set of rules H. The mapping ϕ for an example d is given by, [3] $\phi : d \rightarrow \left(\sqrt{\pi(h_1(d))}, \sqrt{\pi(h_2(d))}, \ldots, \sqrt{\pi(h_t(d))}\right)'$, where $h_1, \ldots, h_t$ are rules and π is the weight assigned to each rule h_i. The kernel for examples d_i and d_j is given by, $k(d_i, d_j) = \langle \phi(d_i), \phi(d_j) \rangle = \sum_{l=1}^{t} \sqrt{\pi(h_l(d_i))}\sqrt{\pi(h_l(d_j))}$. The kernel specified by an inner product between two mapped examples is a sum over all the common hypothesized rules. Given that ϕ maps the data into feature space spanned by ILP rules, we can construct Gaussian RBF kernels, $k_{RBF}(d_i, d_j) = \exp\left(\frac{-\|(\phi(d_i)-\phi(d_j)\|^2}{2\sigma^2}\right)$, where $\|(\phi(d_i) - \phi(d_j)\| = \sqrt{k(d_i, d_i) - 2k(d_i, d_j) + k(d_j, d_j)}$.

In the final stage learning is performed by using an SVM in conjunction with the kernel. SVILP is flexible to construct any kernel in the space spanned by the rules. However, in the present work we used RBF kernels, k_{RBF}, and linear kernels, k, in conjunction with an SVILP machine.

We now consider an example that shows how SVILP kernel measures similarity between two protein domains, 'd2hbg_' and 'd1alla__' which belong to α structural class and 'Globin-like' fold (SCOP classification scheme). Figures 1, 3, 2 and 4 show the two domains and their relationally encoded features. Here predicates 'len', 'nb_alpha', and 'nb_beta' denote the length of the polypeptide chain, number of α-helices and β strands respectively. The other predicates represent the relationship between the secondary structure elements and their properties (hydrophobicity, the hydrophobic moment, the length of proline and etc.). Figure 5 shows a set of induced rules together with their English conversion. A rule classifies an example positive (1) if it fulfils the conditions of the rule while an example that fails to satisfy the conditions is classified negative (0). The set of equally weighted rules maps the two examples as fol-

[3] $'$ specifies column vector

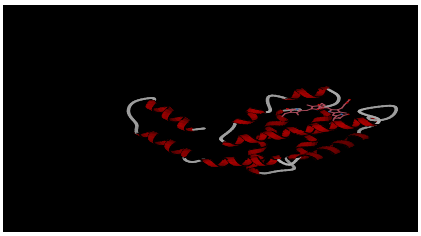

Fig. 1. Protein domain 'd1all__'

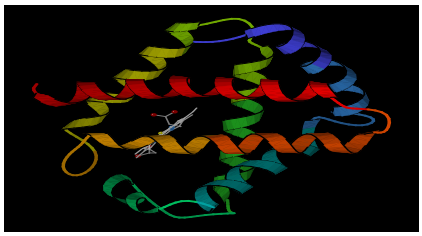

Fig. 2. Protein domain 'd2hbg'

dom_t(d1alla_).
len(d1alla_, 161). nb_alpha(d1alla_,7).
nb_beta(d1alla_,0). has_pro(d1alla_h1).
sec_struc(d1alla_, d1alla_h3).
unit_t(d1alla_h3).
sst(d1alla_h3,4,4,a,104,9,h,0.443,3.003,
116.199, [v,t,p,i,e,e,i,g,v]).
unit_hmom(d1alla_h2, hi).· · ·

Fig. 3. Relationally encoded features of protein domain. 'd1alla_'.

dom_t(d2hbg__).
len(d2hbg__, 147). nb_alpha(d2hbg__,6).
nb_beta(d2hbg__,0).
has_pro(d2hbg__h5).
sec_struc(d2hbg__, d2hbg__h2).
unit_t(d2hbg__h2).
sst(d2hbg__h2,3,3,blank,40,7,h,0.540,
1.812, 213.564, [q,m,a,a,v,f,g]). · · ·

Fig. 4. Relational encoded features of protein domain 'd2hbg__'.

lows: $\phi(d1alla\_) = \phi(d1) = \begin{pmatrix} 1 \times 1 & 1 \times 1 & 1 \times 1 \end{pmatrix}'$ and $\phi(d2hbg\_\_) = \phi(d2) = \begin{pmatrix} 1 \times 1 & 0 \times 1 & 1 \times 1 \end{pmatrix}'$. Given that the rules are equally weighted, each entry of the vector is multiplied by 1. The kernel values between the examples are as follows: $k(d1, d2) = k(d2, d1) = 2$, $k(d1, d1) = 3$ and $k(d2, d2) = 2$. In the proceeding sections we present rule selection and multi-class classification schemes for SVILP.

4 Extending Support Vector Inductive Logic Programming

4.1 Low Dimensional Embedding

As described earlier, an SVILP machine obtains a set $\mathcal{H}$ of all the rules, constructed during the search of the hypothesis space, from an ILP algorithm. The number of rules can be very large and the compression value of a rule can be positive or negative. The wide ranging set $\mathcal{H}$ includes rules that are highly relevant to build a classifier with high generalization ability and rules that are highly irrelevant (noise) and can decrease the generalization performance of the classifier. The irrelevant rules establishe a need to present an effective method to selecting relevant rules and hence embedding the data into an informative lower dimensional logical space. In [1] data was embedded into a lower dimensional space H, where $H \ll \mathcal{H}$, by selecting all the rules with positive compression values. However negatively compressed rules can contain highly relevant information such

fold(Globinlike,A)	←
	adjacent(A,B,C,1,h,h), adjacent(A,C,D,2,h,h), coil(B,C,4).
	/*A domain is classified 1 (belongs to Fold 'Globinlike') if helices B(at position 1) and C are adjacent, C (at position 2) and D are adjacent and length of loop connecting B and C is 4.*/
fold(Globinlike,A)	←
	adjacent(A,B,C,1,h,h), has_pro(C).
	/*A domain is classified 1 if helices B(at position 1) and C are adjacent and C has proline.*/
fold('Globinlike',A)	←
	adjacent(A,B,C,1,h,h), coil(B,C,4), nb_α_interval(4=<(A=<8)).
	/*A domain is classified 1 if helices B (at position 1) and C are adjacent, number of α helices are in range [4,8] and length of loop connecting B and C is 4*/.

Fig. 5. Rules followed by English conversion for Protein domains in Globin-like fold.

as structural and relational features that can be crucial to solving the complex problem at hand. In this section we present a novel method to embed data into a lower dimensional space with extra information.

The proposed method is based on the construction of feature space by exploiting the information content and discriminatory power of the rules. The constructed space is characterized by its amenability to multi-class (/ binary) classification. We now derive an expression to measure the influence of the rules. We use P to denote the number of positive example, and N represent number of negative examples. Similarly, the number of positive examples that fail to satisfy the conditions of a rule are represented by P^-, where N^+ shows the number of negative examples that incorrectly fulfils the conditions of the rule. The expression is given by

$$HD = W_P * P^- + W_N * N^+ \tag{1}$$

where W_P and W_N are the weights assigned to P^-, and N^+ respectively.

The smaller value of HD illustrates the goodness of fit for a rule. The expression can be viewed as weighted sum of hamming distances between two boolean vectors. Let $\hat{\mathbf{c_P}}$ and $\hat{\mathbf{c_N}}$ denote vectors of positive (1) and negative (0) examples respectively. We use $\hat{\mathbf{f_P}}$ to represent vector of the predictions on positive examples by a rule. Similarly, $\hat{\mathbf{f_N}}$ denotes vector of the predictions on negative examples by the rule. The distance between $\hat{\mathbf{c_P}}$ and $\hat{\mathbf{f_P}}$ can be computed by counting the number of entries which differ in both the vectors. Formally, $HD_P(\hat{\mathbf{c_P}}, \hat{\mathbf{f_P}}) = \sum_{i=1}^{P} |c_{P_i} - f_{P_i}|$. $\Big($For labels $\{+1, -1\}$ the distance can be computed by $\sum_{i=1}^{P} \frac{|c_{P_i} - f_{P_i}|}{2}\Big)$. Similarly, $HD_N(\hat{\mathbf{c_N}}, \hat{\mathbf{f_N}}) = \sum_{i=1}^{N} |c_{N_i} - f_{N_i}|$. The weighted sum of the distances is given by $HD = W_P * HD_P(\hat{\mathbf{c_P}}, \hat{\mathbf{f_P}}) + W_N * HD_N(\hat{\mathbf{c_N}}, \hat{\mathbf{f_N}})$. That is like computing the expression HD given in 1.

We now describe how we utilize the expression 1 to obtain a lower dimensional logical and relational feature space with extra information. A set of rules, $\mathcal{H}$, is obtained by an ILP system. In order to measure the score (influence) of rules a

validation set is used. For each rule the values of P^- and N^+ are counted and the goodness of fit is measured by expression, $HD = W_P * P^- + W_N * N^+$. The calculated scores are recorded in a list. Once a list is created, the next step involves sorting it in ascending order. The first t rules with lowest HD values are selected.

The idea behind the use of weights in the expression 1 is to give equal importance to all the classes in a dataset that is characterized by uneven class distribution. We now describe a heuristic method to assign weights. We assume a scenario where a set of examples belong to two classes (positive, negative) and the examples belonging to the negative class make the majority class. In this scenario W_P is set to $\frac{N}{P}$ and W_N is set to 1. We used this approach to compute W_P and W_N for the experiments reported in section 5

4.2 Multi-class Classification

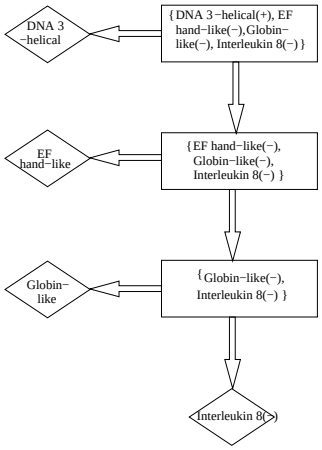

Fig. 6. A decision list, learned by the large margin first order rule learner, for multi-class classification.

We now propose a novel logic based method to solving multi-class classification problems. We apply inductive learning in which an algorithm is provided with a set of examples, D, of the form $D = \{(d_1, c_1), (d_2, c2), \ldots, (d_n, c_n)\}$ where d_i are training examples and $c_i \in \{1, 2, \ldots, r\}$ are classes (labels). The goal of the classification algorithm is to generate a function $f : d \rightarrow \{1, 2, \ldots, r\}$ that assigns a new example d to the class with low error probability.

In order to solve multi-class problems we apply powerful but simple divide and conquer strategy. The complex multi-class classification task is divided into binary problems and each problem is solved recursively. The method constructs a decision list as shown in figure 6. Here each non-leaf node has two children. Classes are represented by non leaf nodes where edges are labeled by the binary classifier's output. We term the technique as decision list based SVILP (DL_SVILP). The method is shown as Algorithm 1. The technique reduces multi-class classification problem to $r-1$ binary problems, where r is the total number of classes. The algorithm can be viewed as comprising $r-1$ iterations. In each

Algorithm 1 Support Vector Inductive Logic Programming (DL_SVILP) for multi-class classification

Input: A set of training examples $\{(d_1, c_1), (d_2, c_2), \ldots, (d_n, c_n)\}$, where $d_i \in D$ and $c_i \in \{1, 2, \ldots, r\}$ and a vector *index* that represents learned structure of the list.
for $j = 1$ to $r - 1$ **do**
/* Select a class p from r classes */
$p = index[j]$
/* Formulate the binary class problem by assigning label '1' to examples of class p and '-1' to examples of remaining classes */
$D_i = \{(d_1, c_1), (d_2, c_2), \ldots, (d_n, c_n)\}$, where $d_i \in D$ and $c_i \in \{1, -1\}$
/* Induce a binary classification function f_i by applying SVILP to set D_i */
$f_i : D_i \rightarrow \{1, -1\}$
/* Reduce the size of set D_i by removing the examples belonging to class p */
$D_{i+1} = D_i \setminus D_p$
end for
return f_i for $i = 1, \ldots, r - 1$

iteration a class is selected as the positive class and the remaining classes are reduced to the negative class. The binary problem is solved by using a large margin first order rule learner. The training set is updated by removing the examples of the chosen class. In this way the root node contains all the classes whereas the node at depth $r - 1$ contains two classes. The size of the training set used at depth $r - 1$ is (much) smaller than the size of the training set for the root node. DL_SVILP assigns a class j to a new example d as follows:

1. Begin at the root node
2. Apply the classifier associated with the node to example d
3. Travel down the edge labeled by the classifier's output
4. If the edge is labeled positive output the class associated with the leaf. If the edge is labeled negative repeat steps 2 and 3 until the last positive edge is reached. Output the label given by the node.

We now describe how the underlying structure of the list is constructed. The method is dynamic and adaptive to the learning process. At each node the selection of the positive class is made in way so as the classifier can have high generalization ability. The method is presented as Algorithm 2. For each class j a binary class problem is formulated by assigning label '1' to examples of chosen class and '-1' to examples of remaining classes. The classifier, induced from the dataset, is evaluated on a validation set. The performance of the classifier is measured by the expression 1 and the values are recorded in a list. In short, r one-vs-all classifiers are trained and a list of scores that represent the performance of the classifiers, is obtained. Finally the list is sorted and this ranked list defines the underlying structure.

4.3 One-vs-all

One-vs-all is a well known multi-class classification strategy. The recent research [7] showed that the solution obtained by the scheme is accurate. We now describe

Algorithm 2 Learning underlying structure for DL_SVILP

Input: Training set, $d_1, d_2, \ldots, d_n$, validation set, $d'_1, d'_2, \ldots, d'_s$, r classes and a large margin first order rule learner (such as SVILP)

for $j = 1$ to r **do**

/* Formulate the binary class problem by assigning label '1' to examples of class j and '-1' to examples of remaining classes */

/* Induce a binary classification function by applying SVILP to training data, $d_1, d_2, \ldots, d_n$ */

/* Apply the learned function to validation set, $d'_1, d'_2, \ldots, d'_s$ */

/* Measure performance of classifier by using expression 1 */

$S[j]' = W_P * P^- + W_N * N^+$

where P = total number of positive example, N = total number of negative examples, P^- = number of misclassified positive examples, N^- = number of misclassified negative examples, $W_P = \frac{N}{P}$ and $W_N = 1$

$index[j]' = j$

end for

/* Sort list S' in ascending order and reorder list $index'$ accordingly */

$S = sort(S')$

$index = reorder(index')$

return $index$ and S

how we design one-vs-all based SVILP multi-class classifier that we term one-vs-all Support Vector Inductive Logic Programming (OVA_SVILP). We construct OVA_SVILP by learning r binary classifiers by using SVILP. A new example is classified by applying all the classifiers to it. The example is assigned a label by the classifier that outputs the largest value(margin).

5 Experiments and Results

We conducted a series of experiments to evaluate the performance of the proposed methods for selecting informative rules and solving multi-class classification problems. We applied the methods to complex tasks, such as mutagenicity detection and protein fold recognition.

For multi-class classification problems we used accuracy and positive predictive value (precision rate) as evaluation measures. Let P_j denote the number of examples belonging to class j, $P = \sum_{j=1}^{j=k} P_j$ represent total number of examples belonging to k classes, and TP_j denote the number of correctly classified examples belonging to class j. The accuracy for each class j is given by $\frac{TP_j}{P_j}$ whereas the overall accuracy is defined by the expression $\frac{\sum_{j=1}^{j=k} TP_j}{P}$. We used two-sample t-test to assess the significance of our results. The performance of the methods was also analyzed in relation to their average positive predictive values (PPVs) that is given by $\frac{TP_j}{TP_j + FP_j}$ for each class j. In the expression FP_j denotes the numbers of examples that are incorrectly classified in class j.

In order to construct underlying binary SVILP classifiers we used CProgol5 (PROGOL) [14] and SVM^{light} [15]. We refer SVILP to SVILP$_C$ for compression

based rule selection whereas SVILP is termed as SVILP$_{HD}$ for the proposed rule selection method. For multi-class classification OVA_SVILP$_C$, DL_SVILP$_C$, OVA_SVILP$_{HD}$, DL_SVILP$_{HD}$ represent compression based and HD (hamming distance) based schemes respectively.

Mutagen Classification: In drug design and development, toxicity classification including mutagen detection and identification is a key task. Mutagenic compounds produce mutations in DNA. In order to validate the use of SVILP as a binary classifier, we applied the algorithm to the mutagen classification problem. For comparison with related techniques, we conducted experiments on a benchmark machine learning dataset, namely mutagenesis [16] that has been widely used for the evaluation of new techniques [17]. We used regression friendly subset comprising 188 molecules and atom and bond background information so that we could compare the performance of SVILP with closely related methods kFOIL and RUMBLE. 10-fold cross validation was used as experimental methodology. At each cross-validation iteration, a classifier was trained on 8 folds, 1 fold was used as the validation set while the remaining 1 fold comprised the test set. We tuned the free parameters clause length and noise of PROGOL, the regularization parameter $\mathcal{C}$ of SVMs and width parameter γ of RBF kernels by using the validation set. The set of values for clause length is {2,4}, noise is {5,10,20}, $\mathcal{C}$ is {1, 10, 100} and γ is {0.001, 0.01,0.1, 1}. Optimal number of rules were selected from the set $\{25, 100, 200, 400\}$. Table 1 shows the results of kFOIL, nFOIL, c-ARMR+SVM, PROGOL, SVILP$_C$ and SVILP$_{HD}$. The reported results of kFOIL, nFOIL, c-ARMR+SVM and RUMBLE appeared in [2] and [3]. The results show that SVILP compares favorably with related approaches. The results also validate the efficacy of the proposed rule selection methodology.

Table 1. Cross-validated accuracy for mutagenesis.

kFOIL	nFOIL	c-ARMR+SVM	RUMBLE	PROGOL	SVILP$_C$	SVILP$_{HD}$
81.3	75.4	73.9	84.0	78.7	85.6	**87.2**

Protein Fold Classification: The recognition of proteins having similar structure is a challenging and complex task in computational biology and bioinformatics. It has key importance in studying protein structure and function and can provide answers to biological problems. In fold recognition, labels are assigned to proteins from a set of predefined annotations (labels, folds). In this way protein fold recognition can be viewed as the multi-class classification task where the problem is characterized by highly skewed class distribution. The aim of a protein fold classification system is to assign proteins to one of many folds with high accuracy. Machine learning methods have been applied to investigate the problem. The studies reported in [18, 4] applied SVMs [13] to solving multi-class protein fold classification problem. Chen and Kurgan [19] and Shen and Chou [20] studied ensemble methods to assign 27 folds, from SCOP, to proteins. [21].

Dataset1: We solved protein fold classification problem by applying the proposed multi-class methods to the dataset presented in [22]. In order to com-

Table 2. 5-fold cross-validated over all accuracy (OA) ± standard deviation for protein fold dataset for MC_ILP, OVA_SVILP$_C$, OVA_SVILP$_{HD}$, DL_SVILP$_C$, DL_SVILP$_{HD}$ and MC_SVM. We also report cross-validated accuracy ± standard deviation for 20 folds. The higher values (shown in bold) demonstrate the advantage of the methods.

Fold	MC_ILP	OVA_SVILP$_C$	OVA_SVILP$_{HD}$	DL_SVILP$_C$	DL_SVILP$_{HD}$	MC_SVM
α						
1	43.3 ± 9.0	76.7 ± 7.7	76.7 ± 7.7	73.3 ± 8.1	66.7 ± 8.6	43.3 ± 9.0
2	28.6 ± 12.1	28.6 ± 12.1	**64.3 ± 12.8**	21.4 ± 11.0	57.1 ± 13.2	14.3 ± 9.4
3	46.2 ± 13.8	**92.3 ± 7.4**	69.2 ± 12.8	61.5 ± 13.5	53.9 ± 13.8	53.8 ± 13.8
4	10.0 ± 9.5	10.0 ± 9.5	30.0 ± 14.5	**40.0 ± 15.5**	30.0 ± 14.5	0.0 ± 0.0
5	40.0 ± 15.5	30.0 ± 14.5	**50.0 ± 15.8**	40.0 ± 15.5	40.0 ± 15.5	20.0 ± 12.6
OA	36.4 ± 5.5	55.8 ± 5.7	**63.6 ± 5.5**	53.3 ± 5.7	54.6 ± 5.7	31.2 ± 5.3
β						
6	73.3 ± 6.6	88.9 ± 4.7	75.6 ± 6.4	**91.1 ± 4.2**	88.9 ± 4.7	71.1 ± 6.8
7	57.1 ± 10.8	90.5 ± 6.4	95.2 ± 4.7	95.2 ± 4.7	90.5 ± 6.4	66.7 ± 10.3
8	0.0 ± 0.0	10.0 ± 6.7	15.0 ± 8.0	15.0 ± 8.0	**35.0 ± 10.7**	15.0 ± 8.0
9	43.8 ± 12.4	68.8 ± 11.6	75.0 ± 10.8	75.0 ± 10.8	75.0 ± 10.8	68.8 ± 11.6
10	64.3 ± 12.8	85.7 ± 9.4	**92.9 ± 6.9**	71.4 ± 12.1	71.4 ± 12.1	64.3 ± 12.8
OA	52.6 ± 4.6	72.4 ± 4.2	70.7 ± 4.2	74.1 ± 4.1	**75.9 ± 4.0**	59.5 ± 4.6
α/β						
11	85.5 ± 4.8	85.5 ± 4.8	**87.3 ± 4.5**	67.3 ± 6.3	76.4 ± 5.7	58.2 ± 6.7
12	52.4 ± 10.9	81.0 ± 8.6	61.9 ± 10.6	76.2 ± 9.3	**90.5 ± 6.4**	28.6 ± 9.9
13	28.6 ± 12.1	35.7 ± 12.8	50.0 ± 13.4	50.0 ± 13.4	50.0 ± 13.4	7.1 ± 6.9
14	7.7 ± 7.4	7.7 ± 7.4	15.4 ± 10.0	30.8 ± 12.8	**38.5 ± 13.5**	0.0 ± 0.0
15	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	8.3 ± 8.0	8.3 ± 8.0	**16.7 ± 10.8**
OA	54.8 ± 4.6	60.9 ± 4.6	60.9 ± 4.6	56.5 ± 4.6	**64.4 ± 4.5**	35.7 ± 4.5
α $+\beta$						
16	53.8 ± 9.8	69.2 ± 9.1	**73.1 ± 8.7**	69.2 ± 9.1	69.2 ± 9.1	23.1 ± 8.3
17	15.4 ± 10.0	30.8 ± 12.8	38.5 ± 13.5	53.9 ± 13.8	53.9 ± 13.8	30.8 ± 12.8
18	7.7 ± 7.4	53.8 ± 13.8	**61.5 ± 13.5**	46.2 ± 13.8	46.2 ± 13.8	30.8 ± 12.8
19	0.0 ± 0.0	8.3 ± 8.0	8.3 ± 8.0	8.3 ± 8.0	25.0 ± 12.5	25.0 ± 12.5
20	77.8 ± 13.9	77.8 ± 13.9	77.8 ± 13.9	66.7 ± 15.7	66.7 ± 15.7	22.2 ± 13.9
OA	32.9 ± 5.8	50.7 ± 5.7	54.8 ± 5.7	52.1 ± 5.8	54.8 ± 5.6	26.0 ± 5.6
OA	46.2 ± 2.6	61.4 ± 2.5	63.3 ± 2.5	60.4 ± 2.5	**64.0 ± 2.5**	40.2 ± 2.5

pare the performance of SVILP based multi-class classification schemes with non-SVILP based methods we used multi-class SVM (MC_SVM) and MC_ILP. MC_SVM was trained by using SVM^{light} [15] where the method was presented in [23]. For MC_SVM, we represented protein domains by using non-relational features namely, total number of residues, α-helices and β-strands. Previous research demonstrated the effectiveness of these features for protein fold classification task. For MC_ILP and SVILP based techniques we used relational fold discriminatory features described in [22]. These features are polypeptide

Fold	#Exm	Fold	#Exm
α		α/β	
1	30	11	55
2	14	12	21
3	13	13	14
4	10	14	13
5	10	15	12
β		$\alpha+\beta$	
6	45	16	26
7	21	17	13
8	20	18	13
9	16	19	12
10	14	20	9

Fig. 7. Class distribution for 20 protein folds of dataset1

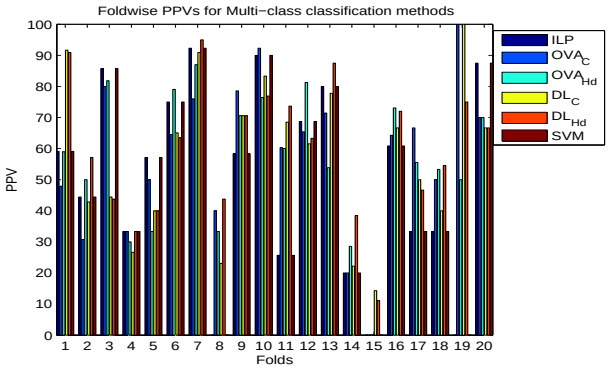

Fig. 8. Fold-wise positive predictive values (PPVs) for MC_ILP (MC), OVA_SVILP$_C$ (OVA$_C$), OVA_SVILP$_{HD}$ (OVA$_{HD}$), DL_SVILP$_C$ (DL$_C$), DL_SVILP$_{HD}$ (DL$_{HD}$) and MC_SVM (SVM).

chain length, number of α-helices and β-strands, adjacent secondary structure elements, properties of the secondary structure such as the hydrophobicity, the hydrophobic moment, the length of proline (number of proline residues) and the length of the loop.

The dataset comprises 381 protein domains. They belong to 20 folds of SCOP that have been categorized into 4 structural classes, namely α, β, α/β and $\alpha+\beta$. The indices 1 to 20 shown in Table 2 represent SCOP folds DNA 3-helical, EF hand-like, Globin-like, 4-Helical cytokines, Lambda repressor, Ig beta-sandwich, Tryp ser proteases, OB-fold, SH3-like barrel, Lipocalins, α/β (TIM)-barrel, Rossmann-fold, P-loop, Periplasmic II, α/β-Hydrolases, Ferredoxin-like, Zincin-like, SH2-like, β-Grasp, and Interleukin respectively. The dataset is characterized by uneven class distribution as shown in figure 7.

We randomly divided the dataset into 5 equal-sized folds and followed the experimental methodology as follows. At each cross validation round 3-folds were used for training the classifiers where the remaining two folds were used as validation set and test set. The free parameter of SVM_MC ($\mathcal{C}$, width of the Gaussian kernel), SVILP_OVA$_C$ ($\mathcal{C}$, width of the Gaussian kernel), SVILP_OVA$_{HD}$ (number of rules, $\mathcal{C}$, width of the Gaussian kernel), SVILP_DL$_C$ ($\mathcal{C}$, width of the Gaussian kernel), and SVILP_DL$_{HD}$ (number of rules, $\mathcal{C}$, width of the Gaussian kernel) were tuned by using the validation set. Table 2 lists the cross-validated accuracy for each protein fold for the multi-class classification methods. Overall accuracy over 20 folds is also given. From the results it is clear the DL_SVILP$_{HD}$ outperforms all other methods in the study. We first focus on the performance of SVILP based methods. In order to assess the effect of low dimensional embedding methods (compression based rule selection, HD based rule selection) on the quality of the trained multi-class classifiers, the performance of DL_SVILP$_{HD}$ was compared with DL_SVILP$_C$. DL_SVILP$_{HD}$ improved the performance over DL_SVILP$_C$ and two sample t-test verified the significance of the gain in accuracy (with p $\ll$ 0.1). Comparison of the performances of OVA_SVILP$_{HD}$ with

Fold	MC_ILP	DL_SVILP$_{HD}$
α	57.78 ± 5.21	**62.22 ± 5.11**
β	33.64 ± 4.57	**45.79 ± 4.82**
α/β	56.45 ± 4.45	**62.90 ± 4.33**
$\alpha+\beta$	66.67 ± 5.41	**72.62 ± 5.27**
All	52.84 ± 2.48	**60.25 ± 2.43**

Fig. 9. Accuracy ± standard deviation for protein fold dataset for MC_ILP and DL_SVILP$_{HD}$. The results are averaged over 5 runs of the techniques.

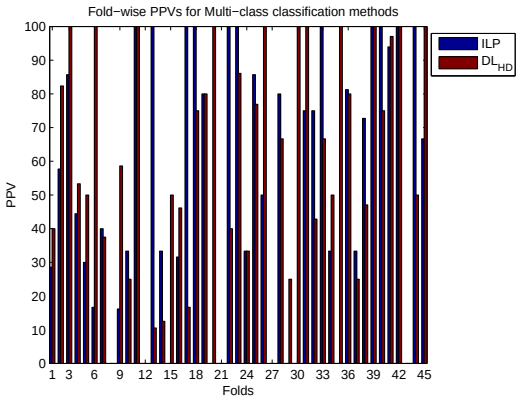

Fig. 10. Fold-wise positive predictive values (PPVs) for MC_ILP (ILP), DL_SVILP$_{HD}$ (DL$_{HD}$).

OVA_SVILP$_C$ demonstrated that OVA_SVILP$_{HD}$ also yielded substantial (but not statistically significant) gain in accuracy. In summary, the results validate the efficacy of HD based rule selection method where the gain in performance is generally substantial and statistically significant.

We now analyze the performance of large margin first order decision list based learner, DL_SVILP$_{HD}$, for multi-class classification. Table 2 shows that the accuracy values of DL_SVILP$_{HD}$ are higher than the other methods. It yielded higher over all accuracy than OVA_SVILP$_C$, OVA_SVILP$_{HD}$ and DL_SVILP$_C$ for folds of β and α/β structural classes. The significance of the results was checked by two sample t-test. The classifiers trained by DL_SVILP$_{HD}$ are statistically significantly better than OVA_SVILP$_C$ (with p $\ll$ 0.1) and DL_SVILP$_C$ (with p=0.11). We also compared the performance of DL_SVILP$_{HD}$ with MC_ILP and MC_SVMs. Table 2 shows the effectiveness of DL_SVILP$_{HD}$ where there is a substantial gain in accuracy values. Again, we used two sample t-test to confirm the statistical significance of the results. The performance of DL_SVILP$_{HD}$ is highly significantly better (with p $\ll$ 0.001) than the performance of MC_ILP and MC_SVM.

The performance of the techniques were also analyzed in terms of average positive predictive values. The values are depicted in figure 8 for 20 folds. The figure demonstrates that SVILP based techniques capture structural and relational similarities between proteins and hence learn accurate classifiers.

Dataset2: We further studied the performance of new logic based multi-class classification strategy, DL_SVILP$_{HD}$, by conducting experiments on the protein folds dataset described in [24]. For this set of experiments we only focused on MC_ILP and DL_SVILP$_{HD}$. In the original study protein fold classification problem was solved by viewing it as a binary problem. The dataset comprises 45 protein folds and 441 protein domains that belong to 4 structural classes. The background knowledge comprised structural information for each protein domain that was derived from known secondary structure and multiple structure alignment information. We performed experiments by using the train/test split

as described in [24]. As there was no validation set, we, therefore, did not tune the parameters of DL_SVILP$_{HD}$ and MC_ILP. Alternatively, we set PROGOL's clause length and noise parameters to 10 and 20 respectively. The regularization parameter $\mathcal{C}$ was set to 1. A linear kernel was used and the number of rules for DL_SVILP$_{HD}$ was set to 100. The performance of DL_SVILP$_{HD}$ was compared to MC_ILP. Table 9 and figure 10 show the results that confirm the usefulness of DL_SVILP$_{HD}$ to solving multi-class classification problems. For the sake of space we only report over all accuracy values for α, β, α/β and $\alpha+\beta$ structural classes. The results show that DL_SVILP$_{HD}$ yielded higher over all accuracy values for all the structural classes. According to the two sample t-test, the performance of DL_SVILP$_{HD}$ is statistically significantly (with $p \ll .01$) better than the performance of MC_ILP. Figure 10 depicts average positive predictive values for 45 protein folds that also confirm the efficacy of DL_SVILP$_{HD}$ to solving protein fold recognition problem.

6 Conclusion

In this paper we proposed a novel logic based multi-class classification method. Furthermore we designed an effective low dimensional embedding technique. The efficacy of the proposed methods was evaluated by applying the techniques to mutagen detection and identification and multi-class protein fold recognition problems. The experimental results demonstrated the efficacy of proposed techniques in selecting highly informative rules and producing accurate solutions to complex (binary) multi-class problems. The method, DL_SVILP, captured structural and relational similarities between examples. The results show that the proposed approach can provide an effective alternative to solving multi-class problems.

Acknowledgements

The authors would like to acknowledge the support of the BBSRC project "Protein Function Prediction using Machine Learning by Enhanced Novel Support Vector Logic-based Approach", Grant Reference BB/E000940/1.

References

1. Muggleton, S., Lodhi, H., Amini, A., Sternberg, M.J.E.: Support Vector Inductive Logic Programming. In: Proceedings of the Eighth International Conference on Discovery Science. Volume 735 of LNAI., Springer Verlag (2005) 163–175
2. Landwehr, N., Passerini, A., Raedt, L., Frasconi, P.: kFOIL: Learning simple relational kernels. In: Proceedings of the National Conference on Artificial Intelligence (AAAI). Volume 21. (2006) 389–394
3. Ruckert, U., Kramer, S.: Margin-base first-order rule learning. Machine Learning **70**(2-3) (2008) 189–206
4. Ding, C.H., Dubchak, I.: Multi-class protein fold recognition using support vector machines and neural networks. Bioinformatics **17** (2001) 349–358

5. Allwein, E.L., Schapire, R.E., Singer, Y.: Reducing multiclass to binary: a unifying approach for marging classifiers. Journal of Machine Learning Research **1**(113–141) (2000)
6. Platt, J.C., Cristianini, N., Shawe-Taylor, J.: Large margin dags for multiclass classification. In: Advances in Neural Information Processing Systems. Volume 12., MIT Press (2000) 457–553
7. Rifkin, R., Klautau, A.: In defense of one-vs-all classification. Journal of Machine Learning Research **5** (2004) 101–141
8. Krebel, U.: Pairwise classification and support vector machines. In: Advances in Kernel Methods: Support Vector Learning. MIT Press (1999) 255–268
9. Riverst, R.: Learning decision list. Machine Learning **2**(3) (1987) 229–246
10. Mooney, R.J., Califf, M.E.: Induction of first-order decision lits: results on learning the past tense of english verbs. Journal of Artificial Intelligence Research **3** (1995) 1–24
11. Quinlan, J.: Learning logical definitions from relations. Machine Learning **5**(3) (1990) 239–266
12. Laer, W.V., de Raedt, L., Dzeroski, S.: On multi-class problems and discretization in Iductive Logic Programming. In: Proceedings of the 10th International Symposium on Foundations of Intelligent Systems. (1997) 277–286
13. Vapnik, V.: The Nature of Statistical Learning Theory. Springer Verlag, New York (1995)
14. Muggleton, S.: Inverse entailment and progol. New Generation Computing **13** (1995) 245–286
15. Joachims, T.: Making large–scale SVM learning practical. In Schölkopf, B., Burges, C.J.C., Smola, A.J., eds.: Advances in Kernel Methods — Support Vector Learning, Cambridge, MA, MIT Press (1999) 169–184
16. Debnath, A.K., de Compadre, R.L.L., Debnath, G., Schusterman, A.J., Hansch, C.: Structure-activity relationship of mutagenic aromatic and heteroaromatics nito compounds. correlation with molecular orbital energies and hydrophobicity. Journal of Medicinal Chemistry **34**(2) (1991) 786–797
17. Lodhi, H., Muggleton, S.: Is mutagenesis still challenging? In: International Conference on Inductive Logic Programming, (ILP - Late-Breaking Papers). (2005) 35–40
18. Shamim, M., Anwaruddin, M., Nagarajaram, H.A.N.J.: Support Vector Machine based classification of protein folds using the structural properties of amino acid residue pairs. Bioinformatocs (2006)
19. Chen, K., Kurgan, L.: PFRES: Protein fold classification by using evolutionary information and predicted secondary structure. Bioinformatics **23** (2007) 2843–2850
20. Shen, H.B., Chou, C.K.: Ensemble classifier for protein fold recognition. Bioinformatics **22** (2006) 1717–1722
21. Murzin, A.G., Brenner, S.E., Hubbard, T., Chothia, C.: SCOP: a structural classification of proteins database for the investigation of sequences and structures. J. Mol. Biol. **247**(536–540) (1995)
22. Turcotte, M., Muggleton, S., Sternberg, J.E.: Automated discovery of structural signatures of protein fold and function. J. Mol. Biol. **306** (2001) 591–605
23. Crammer, K., Singer, Y.: On the algorithmic implementation of multi-class svms. JMLR (2001)
24. Cootes, A.P., Muggleton, S., Sternberg, M.J.: The automatic discovery of structural principles describing protein fold space. Journal of Molecular Biology **330**(4) (2003) 839–850