Genetics and Molecular Biology, 27, 4, 665-672 (2004)
Copyright by the Brazilian Society of Genetics. Printed in Brazil
www.sbg.org.br

Research Article

Evaluation of noise reduction techniques in the splice junction recognition problem

Ana C. Lorena and André C.P.L.F. de Carvalho

Universidade de São Paulo, Instituto de Ciências Matemáticas e de Computação, Laboratório de Computação Bioinspirada, São Carlos, São Paulo, Brazil.

Abstract

The Human Genome Project has generated a large amount of sequence data. A number of works are currently concerned with analyzing these data. One of the analyses carried out is the identification of genes' structures on the sequences obtained. As such, one can search for particular signals associated with gene expression. Splice junctions represent a type of signal present on eukaryote genes. Many studies have applied Machine Learning techniques in the recognition of such regions. However, most of the genetic databases are characterized by the presence of noisy data, which can affect the performance of the learning techniques. This paper evaluates the effectiveness of five data pre-processing algorithms in the elimination of noisy instances from two splice junction recognition datasets. After the pre-processing phase, two learning techniques, Decision Trees and Support Vector Machines, are employed in the recognition process.

Key words: pre-processing, machine learning, splice junction recognition.

Received: August 15, 2003; Accepted: August 20, 2004.

Introduction

During the last years, sequencing projects have accumulated large volumes of data. For a complete understanding of the genetic mechanisms, several analyses need to be carried out on these data. One example is the localization of genes and definition of their structures (Craven and Shavlik, 1994; Semple, 2000).

Performing this kind of analysis in laboratories is usually an arduous and costly task. To overcome this problem, many studies have employed Machine Learning (ML) techniques in the analysis of sequence data (Lapedes *et al.* 1989; Lorena *et al.*, 2002; Rampone, 1998; Towell, 1991; Xu *et al.*, 1994). Machine Learning, a sub area of Artificial Intelligence, provides several techniques which can extract concepts (knowledge) from a given dataset (Mitchell, 1997). These techniques are usually applied in the induction of a hypothesis, also known as a classifier or predictor, through a process called training.

When using genetic datasets, some aspects may influence the performance achieved by ML techniques. Due to the imprecise nature of biological experiments, redundant and noisy samples can be present at a high rate. Noisy patterns can corrupt the generated classifier and should be removed (Lavrac and Gamberger, 2001; Lorena *et al.*, 2002). Redundant and similar samples can also be eliminated without harming the concept induction and may even improve it.

Several algorithms have been proposed to reduce the number of exemplars from a given dataset (Tomek, 1976; Wilson, 1972; Wilson and Martinez, 1997; Wilson and Martinez, 2000). These reduction techniques are based on heuristics defined to stimulate the removal of patterns of low significance present in the dataset. This paper presents an empirical evaluation of five dataset reduction techniques in the pre-processing of two datasets. The task is the identification of splice sites on DNA sequences, a problem that is part of eukaryote gene structure identification.

After the pre-processing phase, Decision Trees (DTs) (Quinlan, 1986) and Support Vector Machines (SVMs) (Cristianini and Shawe-Taylor, 2000) will be trained using the original dataset and also the different sets of pre-processed data for the recognition task. By evaluating the difference in performance among classifiers generated over original (without pre-processing) and pre-processed data, the power of the reduction techniques in maintaining the most important patterns can then be estimated.

This work is organized as follows: next section gives an overview of some Molecular Biology concepts, necessary for the understanding of the splice site recognition

Send correspondece to Ana C. Lorena. Universidade de São Paulo, Instituto de Ciências Matemáticas e de Computação, Laboratório de Computação Bioinspirada, Av. do Trabalhador São-Carlense 400, Centro, Caixa Postal 668, São Carlos, São Paulo, Brazil. E-mail: aclorena@icmc.usp.br.

problem, which is described on Splice Junction Recognition. Materials and Methods presents the materials and methods employed in this work. Results and Discussion presents the experiments conducted and the results achieved. Conclusion concludes this paper.

Molecular Biology

Molecular Biology is a field of research concerned with the study of cells and molecules, the basic blocks of all living beings. In particular, it studies the genomes of organisms, defined as their set of genetic information. This information is coded on genes along DNA (Deoxyribonucleic Acid) molecules.

The DNA molecule is composed of sequences of nucleotides, of which there are four types: Adenine (A), Cytosine (C), Guanine (G) and Thymine (T). A single DNA molecule typically has thousands of genes. From the information coded in the nucleotide composition of a gene, proteins, essential components of living beings, are produced. This is accomplished by a process called gene expression.

The gene expression, illustrated on Figure 1, is composed of two stages: transcription and translation. In transcription, a mRNA (messenger Ribonucleic Acid) is produced from a DNA strand. The mRNA is similar to DNA. It is also composed of nucleotides, except for an Uracil (U) in the place of the Thymine. In the translation stage, the mRNA molecule codes for the final protein.

The understanding of the formation and distribution of the gene sequences is an important source of knowledge. Several areas can benefit from studies of these structures, such as medicine, pharmacology and agriculture. They can provide, for example, insights for the development of new drugs and treatments for diseases like cancer. The next section describes one particular characteristic of eukaryote DNA, the presence of splice sites.

Splice Junction Recognition

There are some differences in the basics steps of gene expression between eukaryote and prokaryote organisms. Eukaryotes (complex beings, like humans) are organisms whose genetic material is delimited in a nucleus. Prokaryote organisms (like bacteria), on the other hand, have their genetic material dispersed in the cell.

The main difference in expression between these classes of organisms is the splicing of some regions of the mRNA before the translation stage in eukaryotes. This splicing step is a result of the fact that eukaryote genes are composed of alternate segments of exons and introns. Exons are regions that code for the final protein. Introns intermediate exons and do not code for proteins[1]. Thus, introns have to be removed from the mRNA molecule, which is accomplished in the splicing step, illustrated in Figure 2.

Splice junctions are the boundary points where splicing occurs. The splice junction recognition problem then involves identifying if a sequence of a fixed size has an intron/exon site (IE), an exon/intron site (EI), or if it does not have a splice site (N). This task is part of eykaryote gene structure identification, since the boundary points between coding and non-coding regions have to be accurately determined in this process.

Many studies report the successful application of Machine Learning (ML) techniques in splice junction recognition. In Towell (1991), an Artificial Neural Network (ANN) (Haykin, 1999) was employed in this task. Prior knowledge of the biological domain has been used to initialize the network. The Statlog Project (Michie, 1994) also reports the use of various ML techniques for primate splice site identification. Another study based on ANNs was proposed by Rampone (1998), in which an ANN refined Boolean formulas inferred from data.

ML techniques have also been successfully applied in the recognition of other types of signals present on genes, such as translation initiation sites (Zien *et al.*, 2000; Pedersen *et al.*, 1997) and promoters (Bajic *et al.*, 2002; Gordon *et al.*, 2003). In Zien *et al.* (2000), for example, SVMs were applied to the recognition of translation initiation sites. Prior knowledge was incorporated into the technique and the results observed were superior to those obtained by an ANN (Pedersen and Nielsen, 1997) and the Markovian method of Salzberg (1997).

The following section describes the approach and techniques employed in this work.

Materials and Methods

This section describes the materials and methods employed in this work. It starts with a description of the

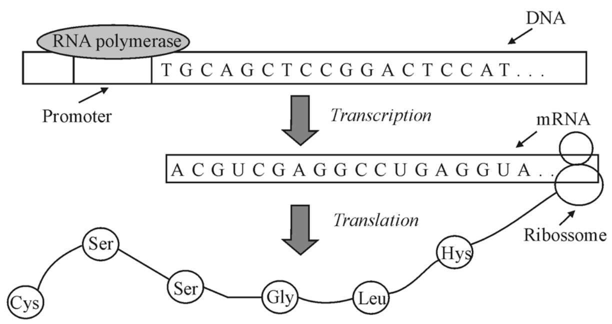
DNA
RNA polymerase
T G C A G C T C C G G A C T C C A T . . .
Promoter
Transcription
mRNA
A C G U C G A G G C C U G A G G U A . .
Translation
Ribossome
Ser
Cys
Ser
Gly
Leu
Hys

Figure 1 - Gene expression process.

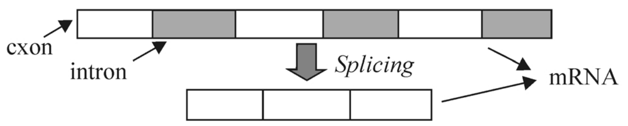

Figure 2 - Splicing in mRNA molecule.

1 In this paper, the occurrence of alternative splicing is not considered.

datasets used in the experiments conducted. Next, the pre-processing stage for the elimination of noisy patterns from these datasets is presented. The learning techniques used in the final recognition problem (of splice junctions) are then described. The power of the pre-processing techniques in maintaining the most important patterns in the datasets will be evaluated according to the test error rates obtained by the learning techniques trained over the pre-processed datasets compared to the test error rates obtained by the same techniques over the original data (without pre-processing).

Datasets

Two datasets consisting of sequences of DNA with and without splice junctions were considered in this work. The first one, from primates, was extracted from the UCI benchmark (Blake, 1998). Although it originally contained 3190 instances, sequences with composition different from the nucleotides A, T, C and G were removed. This procedure left a total of 3175 instances. The other dataset used in this work was extracted from HS3D (Homo Sapiens Splice Sites Dataset), which contains 5947 sequences of human DNA with known splice sites (IE or EI) and 635,666 sequences that do not contain a splice site (false splice sites) (Polastro and Rampone, 2002). From this data, 5000 examples were randomly chosen, of which 25% were of the IE type, 25% of the EI type and 50% of the N type (the same proportion as the dataset from UCI, as can be seen on Table 1), as a final dataset for performing the experiments. Sequences with compositions different from the four DNA nucleotide types were also ignored for this dataset.

Table 1 summarizes both datasets, showing the total number of instances (# Instances), the number of continuous and nominal attributes (# Attributes) and the class distribution (Class%).

Data pre-processing

Biological data is often characterized by the presence of noisy patterns. This kind of data may originate, for example, from errors during biological data collection or errors which occurred in the dataset generation process (Setubal and Meidanis, 1997). Although many of the ML techniques can deal with noise, as is the case of the techniques applied in this study, detecting and filtering noisy instances from the training dataset can help the induction of the target hypothesis (Lavrac and Gamberger, 2001).

Table 1 - Datasets summary description.

Dataset	# Instances	# Attributes (cont., nom.)	Class%
UCI	3175	60 (0,60)	IE 25 EI 25 N 50
HS3D	5000	140 (0,140)	IE 25 EI 25 N 50

This study evaluates the use of five pre-processing techniques in the elimination of less reliable patterns from the datasets considered in the preveious section. The first three techniques are classified in Wilson and Martinez (2000) as noise-filtering approaches. The two remaining are dataset reduction techniques, capable of identifying noise data and also redundant instances.

For a better understanding of the pre-processing techniques used, consider the dataset of Figure 3a. The patterns in this dataset can be divided into five distinct types:

• *Mislabelled cases*: instances incorrectly classified in the database generation process. These cases are noisy and are represented in Figure 3b;

• *Redundant data*: instances that can be represented by others in the dataset. Some examples are presented in Figure 3c. They form "clusters" in the dataset. It should be pointed out that at least one of these patterns should be maintained so that the representativeness of the cluster is conserved;

• *Outliers*: instances too distinct when compared to the other samples present in the dataset. These instances can be noisy or very particular cases and their influence in the hypothesis induction should be minimized. Examples of this case are presented in Figure 3d;

• *Borderlines*: instances close to the decision border. These samples are quite unreliable, since even a small amount of noise could have moved them to the wrong side of the decision border. Examples are presented in Figure 3e;

• *Safe cases*: remaining instances (Figure 3f). These cases should be saved for the learning process.

The objective of the pre-processing phase is to create a training set consisting of only the safe cases.

As previously mentioned, five pre-processing techniques are investigated in this paper. Descriptions of these techniques are presented next:

• *Edited Nearest Neighbour* (ENN) (Wilson, 1972): according to this technique, an instance is removed from the dataset if its class is not equal to the class of the majority

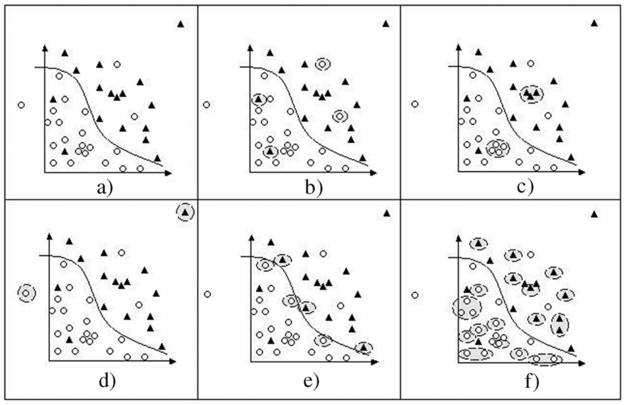

Figure 3 - Dataset example (a); mislabelled instances (b); redundant data (c); outliers (d); borderlines (e); safe data (f).

of its k nearest neighbours. This procedure removes mislabelled data and borderlines;

• *Repeated ENN* (RENN): in this variation, the ENN technique is repeatedly applied until all instances in the dataset have the majority of its neighbours with the same class. This process enlarges the margin of separation between classes and smoothes the decision border used to separate data;

• *All-kNN* (Tomek, 1976): extension of the ENN that can be applied in the detection and removal of mislabelled and borderline data. This technique works in the following way: for i = 1, ..., k, it signalises as "bad" any instance incorrectly classified by its k nearest neighbours. After the loop is complete, it removes the signalized instances;

• *Decremental Reduction Optimization Procedure 2* (DROP2) (Wilson and Martinez, 2000): let S and S' be the original and pre-processed datasets, respectively. Initially, S= S'. A list of nearest neighbours is then built for each instance. A list of associates is also constructed. The associate of a pattern **x** is defined as a sample that has **x** as one of its nearest neighbours. DROP2 eliminates a pattern **x** from S' if the majority of its associates in the original dataset S are correctly classified without this pattern. It lets the nearest enemy of a pattern **x** be the instance of S' of opposite class closest to **x**. The order of checking for removal in DROP2 is given by the samples furthest from their nearest enemies. This technique is capable of removing noisy instances, outliers and redundant data;

• DROP4 (Wilson and Martinez, 2000): this technique is similar to DROP2, but it starts by applying a noise-filtering pass to the dataset. In this filtering pass, an instance is removed only if it is incorrectly classified by its k nearest neighbours (like ENN) and if its removal does not affect the classification of other instances.

In all techniques described, the computation of the distance between the patterns was carried out with a metric employed to measure distance of symbolic attributes called *Value Difference Metric* (VDM) (Stanfill and Waltz, 1986). The VDM considers two values similar if they occur with almost identical relative frequencies for all classes. So the distance d between two values of a certain symbolic attribute V is defined by Equation 1 (Batista *et al.*, 2000), where V_1 and V_2 are two possible values assumed by V, p is the number of classes in the dataset and C_{1i} is the number of samples of class i whose attribute V assumed the value V_1. C_1 is the total number of samples whose attribute V assumed the value V_1 and m is a constant, frequently 1.

$$\mathrm{d}(V_1,V_2)=\sum_{i=1}^{p}\left|\frac{C_{1i}}{C_1}-\frac{C_{2i}}{C_2}\right|^{m} \tag{1}$$

Learning techniques

Several supervised learning algorithms can be used to induce classifiers from a set of samples. Given a training set composed of known samples of sequences with and without splice junctions, a learning algorithm must induce a classifier that should be able to predict the class of any new sample from the same domain where the learning occurred.

The learning methods used in this paper are Support Vector Machines (SVMs) (Cristianini and Shawe-Taylor, 2000) and Decision Trees (DTs) (Quilan, 1986). The predictive error rates of these techniques can be used to obtain an indication of the effectiveness of the pre-processing phase, which will be discussed in more details in Results and Discussion.

SVM is a learning technique based on the Statistical Learning Theory (Vapnik, 1995). Its main goal is to find a hyperplane that separates members and non-members of a class in an abstract space of high dimension, also known as feature space (Cristianini and Shawe-Taylor, 2000). In this space, the classes present in the training set become linearly separable, and the optimal hyperplane is defined as the one that maximizes the separation margin between the classes. The migration of the samples to the feature space is performed with the assistance of functions called Kernels. A Kernel function allows simple access to spaces of high dimensions (in some cases infinite), without the necessity of knowing the form of the mapping function that represents the data in this space, which usually is very complex. The main advantages of the SVMs are their precision and robustness with patterns that have a large number of attributes (high dimensional), like those found in the splice junction recognition problem.

DT is a symbolic learning technique (Quilan, 1986). Its structure is composed of nodes and ramifications. The nodes represent tests applied to data or classes, when the node is a leaf. The ramifications are possible results of the tests. The classification of a new sample is performed following the nodes and ramifications until a leaf is reached. One of the main advantages of DTs is the comprehensiveness of the induced rules. This is an important characteristic of symbolic ML algorithms, since it allows human interpretation of the induced knowledge. As a disadvantage, one can mention the poor robustness of the DT to high dimensional data.

Results and Discussion

To obtain better estimates of the predictive performance of the learning techniques considered in this work, the datasets described in Datasets were first divided into training and test sets following the 10-fold cross-validation methodology (Mitchell, 1997). According to this method, the dataset is divided in ten disjoint subsets of approximately equal size. At each training/test round, nine subsets are used for training and the remaining one is left for testing. This makes a total of ten pairs of training and test subsets. The error of the classifier on the total dataset is then given by the average of the errors observed in each test partition.

The pre-processing procedures described in Data pre-processing were then applied to the training sets obtained. The source code of the pre-processing techniques was obtained from ftp://axon.cs.byu.edu/pub/randy/ml/drop (Wilson and Martinez, 2000). Different values of k, the number of nearest neighbours in the pre-processing techniques, were tested. They were: 1, 3, 5, 7 and 9. Table 2 shows the average number of instances eliminated from each dataset, when a number of k = 3 neighbours, default value of the pre-processing source code, was used.

It should be observed that the techniques DROP2 and DROP4 have lead to very large reductions in the datasets. Although for larger values of k these reduction rates were smaller, they were still high for k = 9, at around 90% for the UCI dataset and 80% in the case of the HS3D dataset.

The pre-processing mean time (in seconds) for k = 3 was also calculated and is presented in Table 3. The experiments were conducted on a dual processor of 2.4 GHz and 1 GB of memory. A longer pre-processing time was observed in the case of the techniques DROP2 and DROP4, which in the end eliminated larger numbers of patterns. Although the pre-processing time presented can be considered relatively high, especially in the case of the HS3D dataset, it should be emphasized that the pre-processing stage is applied only once over the training sets, which can then be used in the learning phase of any supervised ML technique.

For the conversion of the different pre-processed datasets generated during the experiments conducted to the formats demanded by the pre-processing code and of the ML techniques simulators, routines on the Perl language were implemented. The determination of the best classifiers and k pre-processing parameter for each dataset was also made by a routine coded in Perl.

Descriptions and discussions of the results achieved by the SVMs and DTs classifiers over the pre-processed and original datasets are presented next.

Support vector machines

The software used for the SVMs induction was the SVMTorch II (Collobert and Bengio, 2001), a simulator that works for both classification and regression problems. SVMTorch II has been specifically tailored for large-scale problems, like the one investigated in this paper. It must be pointed out that SVMs can only process attributes with numerical values, so the sequences had to be coded accordingly. As in the work of Zien *et al.* (2000), canonical encoding, in which each possible value of a nominal attribute is represented by a vector of binary values, was used. The following coding was then applied: A = (0 0 0 1), C = (0 0 1 0), G = (0 1 0 0) and T = (1 0 0 0). This representation scheme ensures that the distances between different attribute values are equivalent.

SVMs are originally designed for the solution of binary problems, i.e., applications with two classes only. In this study, the extension of SVMs to the multiclass splice junction recognition problem was performed with the one-versus-all technique, implemented in SVMTorch II. In the one-versus-all technique, given *p* classes, *p* binary classifiers are induced, each one responsible for distinguishing one class from the remaining classes (Cristianini and Shawe-Taylor, 2000). The classifier with the highest output value provides the final prediction.

In the experiments conducted, two types of Kernel functions with different values of parameters were tested. The functions used were Polynomials of distinct degrees (1 to 5) and Gaussians with varying standard deviation parameters (5, 10, 25 and 50). Another important parameter for SVMs refers to a constant that is related to the way this technique deals with noise. This constant, usually denoted by C, imposes a tradeoff in the number of patterns allowed to be within the maximized margins. The smaller this value, the smoother is the decision function in relation to noise.

Table 4 presents the best SVM models obtained for each dataset, as well as the best k parameter for the pre-processing techniques. The abbreviations G and P refer to Kernel functions of the Gaussian and Polynomial types,

Table 2 - Datasets average reduction with a pre-processing parameter of k = 3.

Dataset	ENN	RENN	AllKNN	DROP2	DROP4
UCI	14.2%	14.7%	15.9%	90.8%	92.2%
HS3D	23.6%	25.0%	36.2%	81.2%	85.0%

Table 4 - Best SVMs models and corresponding pre-processing technique parameters.

	UCI			HS3D		
	k	Kernel	C	k	Kernel	C
Original	-	G (5)	25	-	G (10)	200
ENN	1	P (4)	25	5	P (2)	25
RENN	3	P (4)	25	1	P (2)	25
AllKNN	3	P(4)	25	1	P (2)	25
DROP2	1	P(2)	25	5	P (2)	25
DROP4	9	G (10)	25	5	P (2)	25

Table 3 - Pre-processing average time (in seconds) for a pre-processing parameter k = 3.

Dataset	ENN	RENN	AllKNN	DROP2	DROP4
UCI	148.1 ± 1.7	156.0 ± 2.1	80.7 ± 0.1	551.6 ± 7.0	705.3 ± 3.5
HS3D	953.2 ± 3.3	1010.0 ± 4.7	457.0 ± 0.3	1850.0 ± 5.0	2271.7 ± 11.0

respectively. The values of their corresponding parameters (standard deviation for the Gaussian Kernel and polynomial degree, in the Polynomial Kernel case), are indicated in parentheses. Table 5 presents the mean error rates (standard deviation) obtained by the indicated SVM models in the test partitions generated by the cross-validation procedure. The lowest and next to lowest error rates for each dataset (UCI and HS3D) are detached in boldface and italics, respectively.

Applying the t student statistical test (Johnson, 2000) to the results of Table 5, it can be verified that the application of the techniques ENN, RENN and AllKNN maintained the error rate verified over the original data for both datasets, with a confidence level of 95%. Besides this, it can be observed from Table 4 that the SVMs models obtained using these pre-processed datasets were also simplified by a reduction in the complexity of the generated models (by Gaussian Kernels, that have infinite VC dimension (Burges, 1998), to Polynomial kernels, that have their complexity controlled by their degree (Burges, 1998; Haykin, 1999)). The DROP2 and DROP4 techniques clearly worsened results were obtained for both datasets. This can be due to the drastic reductions in the datasets with the application of these techniques.

Decision trees

The induction of DTs was carried out using the C4.5 algorithm (Quilan, 1988). Three distinct values of the pruning parameter used by this leaning technique were tested: 0.25, 0.5 and 0.75. Pruning is a procedure employed by DTs for dealing with noisy data. It prunes ramifications of the trained tree that have low expressive power according to some criterion. In this process, whole subtrees are replaced by leaf nodes. The replacement is made if the expected error rate in the subtree is larger than in the single leaf. Higher values of the pruning parameter lead to lower reductions of the induced tree.

An important characteristic of DTs is their average size. This information is closely related to the comprehensiveness of the tree. Smaller trees can be considered more comprehensive. Table 6 shows the DTs mean size of the DT models induced in this work. A large reduction in the size of the DTs trained with the pre-processed data in relation to the DTs obtained using the original data also indicates a simplification of the final model. This reduction is very important for users, since it allows a better understanding of the criteria used by the model to generate its decisions. Table 6 also shows the best pruning parameter in each case and best number of nearest neighbours k of the pre-processing techniques.

Still from Table 6, larger reductions in the mean size of the trees can be verified when the DROP2 and DROP4 techniques are used, which are related to the larger reductions of the original datasets obtained by using these techniques. For the ENN, RENN and AllKNN techniques, large reductions in the DTs mean size are also verified (with more than 50% of reductions).

For a better evaluation of the performance achieved, the average test error rates obtained from each model are presented on Table 7. Like in Table 5, the first and second best results are indicated in boldface and italics, respectively. From this table, it can be seen that the very large reduction of the DTs mean size with the application of techniques DROP2 and DROP4 is related to the drastic reductions of the size of the datasets after they were pre-processed by these techniques. In the case of ENN, RENN and AllKNN techniques, a reduction in the error rates can be observed, although not statistically significant at a confidence level of 95%. However, for ENN, RENN and AllKNN, there were clear simplifications in the induced models, indicating comprehensiveness gains, while maintaining and even reducing the obtained error rates by a factor of approximately 2%. In some situations, the achievement of a higher comprehensibility can be more important than that of a higher accuracy rate.

Table 5 - SVMs test error rates.

Dataset	Original	ENN	RENN	AllKNN	DROP2	DROP4
UCI	**2.9 ± 1.2**	3.0 ± 1.4	*3.0 ± 1.2*	3.1 ± 1.3	11.2 ± 2.6	12.2 ± 1.2
HS3D	**11.3 ± 0.8**	12.5 ± 1.1	*12.4 ± 1.2*	12.7 ± 0.8	21.1 ± 1.4	24.7 ± 2.2

Table 6 - Mean size of DTs, best pruning and pre-processing parameters.

	UCI			HS3D		
	k	DT size	prune	k	DT size	prune
Original	-	417.0 ± 3.4	0.25	-	1387.8 ± 4.8	0.25
ENN	3	226.2 ± 18.2	0.25	9	637.8 ± 22.6	0.25
RENN	9	161.0 ± 8.0	0.25	7	575.0 ± 29.6	0.25
AllKNN	9	173.0 ± 7.5	0.25	5	549.4 ± 60.2	0.25
DROP2	9	153.4 ± 8.1	0.25	1	285.0 ± 37.5	0.25
DROP4	9	127.4 ± 7.4	0.25	7	334.6 ± 59.4	0.25

Table 7 - DT test error rates.

Dataset	Original	ENN	RENN	AllKNN	DROP2	DROP4
UCI	8.1 ± 1.5	*6.5 ± 1.4*	**6.5 ± 1.2**	**6.5 ± 1.2**	26.8 ± 3.5	30.3 ± 5.1
HS3D	23.9 ± 1.7	**21.7 ± 1.7**	*21.9 ± 2.0*	22.9 ± 1.9	31.9 ± 4.1	34.8 ± 3.2

Learning techniques performance comparison

Comparing the results obtained by the learning techniques considered in this work, a better performance of the SVM classifiers regarding classification accuracy can be verified. This can be confirmed statistically, at 95% of confidence level.

The higher test error rates of DTs in comparison to SVMs may be due to their difficulty in dealing with applications with a high number of attributes, as the one investigated in this paper. On the other hand, SVMs are usually very robust for high dimensional data.

If the interest is in comprehensiveness, however, the use of DTs is indicated. Thus, in this case, the use of the pre-processing techniques ENN, RENN and AllKNN can bring high benefits in comprehensiveness gains.

Pre-processing techniques results analysis

From the results presented in previous sections, some conclusions can be drawn concerning the performance of the pre-processing techniques investigated in this paper.

The results suggest that the application of the noise filtering techniques ENN, RENN and AllKNN maintained the most expressive patterns in the datasets UCI and HS3D, which was reflected in the error rates obtained, which for DTs in particular, were even reduced. Simplifications of the induced classifiers were another benefit observed. This fact is in accordance with Occam's razor in ML, that affirms that among several models with the same error, the less complex must be chosen (Lavrac and Gamberger, 2001).

In the case of the pre-processing techniques DROP2 and DROP4, overly drastic reductions were observed on training data. This was reflected in the reduced performance achieved by the learning techniques. So, these techniques did not maintain the most significative patterns in the datasets, and probably eliminated safe samples. As higher values of k had a tendency to decrease the reduction of the training sets, further experiments shall investigate if larger values of k can lead to better results.

Conclusion

This paper investigated the application of data pre-processing techniques in two splice junction recognition datasets. Five techniques were studied. In order to evaluate the power of these techniques in maintaining the most informative patterns, two learning techniques, Support Vector Machines and Decision Trees, were trained over the original and pre-processed datasets.

The results observed indicate that the three first techniques investigated, named ENN, RENN and AllKNN, were more effective in the pre-processing process. This resulted in simplifications of the induced classifiers and, for the DTs, in lower error rates and higher comprehensiveness.

However, more experiments, including other problems and datasets from Molecular Biology, are necessary for a wider evaluation of the benefits in the application of these pre-processing procedures in Bioinformatics. The use of other pre-processing techniques could also lead to similar or even better results than those observed.

Acknowledgements

The authors would like to thank the Brazilian National Research Council (CNPq) and the São Paulo State Research Foundation (Fapesp) for the financial support provided, Wilson and Martinez for the pre-processing source code and Pollastro and Rampone for the HS3D dataset.

References

Bajic VB, Chong A, Seah SH and Brusic V (2002) An intelligent system for vertebrate promoter recognition. IEEE Intelligent Systems 4:64-70.

Batista GEAPA, Carvalho ACPLF and Monard MC (2000) Applying one-sided selection to unbalanced datasets. In: Mexican International Conference on Artificial Intelligence (MICAI) Lecture Notes in Artificial Intelligence, Springer-Verlag, v 1793, pp 315-325.

Blake CL and Merz CJ (1998) UCI repository of machine learning databases. http://www.ics.uci.edu/ ~mlearn/.

Burges CJC (1998) A tutorial on support vector machines for pattern recognition. Knowledge Discovery and Data Mining 2:1-43.

Collobert R and Bengio S (2001) SVMTorch: Support vector machines for large-scale regression problems. Journal of Machine Learning Research 1:143-160.

Craven MW and Shavlik JW (1994) Machine learning approaches to gene recognition. IEEE Expert 9:2-10.

Cristianini N and Shawe-Taylor J (2000) An Introduction to Support Vector Machines. Cambridge University Press, 189 pp.

Gordon L, Chervonenkis AY, Gammerman AJ, Shahmuradov IA and Solovyev VV (2003) Sequence alignment kernel for recognition of promoter regions. Bioinformatics 19:1964-1971.

Haykin S (1999) Neural Networks - A Comprehensive Foundation. Prentice Hall, 842 pp.

Johnson RA (2000) Miller and Freund's Probability and Statistics for Engineers. Prentice Hall, 622 pp.

Lapedes A, Barnes C, Burks C, Farber R and Sirotkin K (1989) Application of neural networks and other machine learning algorithms to DNA sequence analysis. In: Bell G and Marr T (eds) Computers and DNA, SFI in the Sciences of Complexity, v 7, pp 157-182.

Lavrac N and Gamberger D (2001) Saturation filtering for noise and outlier detection. In: Proceedings of the Workshop in Active Learning, Database Sampling, Experimental Design: Views on Instance Selection, European Conference on Machine Learning, pp 1-4.

Lorena AC, Batista GEAPA, Carvalho ACPLF and Monard MC (2002) The influence of noisy patterns in the performance of learning methods in the splice junction recognition problem. In: Proceedings of the VII Brazilian Symposium on Neural Networks, IEEE Computer Society Press, pp 31-36.

Michie D, Spiegelhalter DJ and Taylor CC (1994) Machine Learning, Neural and Statistical Classification. Ellis Horwood, 298 pp.

Mitchell T (1997) Machine Learning. McGraw Hill, 432 pp.

Pedersen AG and Nielsen H (1997) Neural network prediction of translation initiation sites in eukaryotes: Perspectives for EST and genome analysis. In: Proc Int Conf Intell Syst Mol Biol (ISMB'97), pp 226-233.

Pollastro P and Rampone S (2002) HS3D, a dataset of Homo Sapiens splice regions, and its extraction procedure from a major public database. Int J Mod Phys C 13:1105-1117.

Quinlan JR (1986) Induction of decision trees. Machine Learning 1:81-106.

Quinlan JR (1988) C4.5 Programs for Machine Learning. Morgan Kaufmann, CA, 302 pp.

Rampone S (1998) Splice-junction recognition on DNA sequences by BRAIN learning algorithm. Bioinformatics 14:676-684.

Salzberg SL (1997) A method for identifying splice sites and translational start sites in eukaryotic mRNA. Comput Appl Biosci 13:365-376.

Semple C (2000) Gene prediction: The end of the beginning. Genome Biol 1:1-3.

Setubal J and Meidanis J (1997) Introduction to Computational Molecular Biology. PWS Publishing Company, 296 pp.

Stanfill C and Waltz D (1986) Toward memory-based reasoning. Communications of the ACM 29:1213-1228.

Tomek I (1976) An experiment with the edited nearest-neighbor rule. IEEE Trans Syst Man Cybern 6:448-452.

Towell GG (1991) Symbolic knowledge and neural networks: Insertion, refinement, and extraction. PhD thesis, University of Wisconsin, Madison.

Wilson DL (1972) Asymtoptic properties of nearest neighbor rules using edited data. IEEE Trans Syst Man Cybern 2:408-421.

Wilson DR and Martinez TR (1997). Instance pruning techniques. In: Fisher D (ed) Machine Learning: Proceedings of the XIV International Conference, Morgan Kaufmann Publishers, pp 404-411.

Wilson DR and Martinez TR (2000) Reduction techniques for instance-based learning algorithms. Machine Learning 38:257-286.

Vapnik VN (1995) The Nature of Statistical Learning Theory. Springer-Verlag, 214 pp.

Xu Y, Mural RJ, Einstein JR, Shah M and Uberbacher EC (1996) GRAIL: A multi-agent neural network system for gene identification. In: Proceedings of the IEEE 84:1544-1552.

Zien A, Rätsch G, Mika S, Schölkopf B, Lengaeuer T and Müller KR (2000) Engineering support vector machine kernels that recognize translation initiation sites in DNA. Bioinformatics 16:799-807.