Comparison of Machine Learning Algorithms in Recognation of Regulatory Region of DNA

Günay Karlı, Şenol Doğan,

Faculty of Engineering and Information Technology, International Burch Universiy, Sarajevo, BIH

E –mails: *gkarli@ibu.edu.ba – gkarli@yahoo.com, sdogan@ibu.edu.ba*

Keywords: Data mining, machine learning, supervised learning, classification, rule-based algorithms.

Abstract

Data mining has become an important and active area of research because of theoretical challenges and practical applications associated with the problem of discovering interesting and previously unknown knowledge from very large real world database. These databases contain potential gold mine of valuable information, but it is beyond human ability to analyze massive amount of data and elicit meaningful patterns by using conventional techniques. In this study, DNA sequence was analyzed to locate promoter which is a regulatory region of DNA located upstream of a gene, providing a control point for regulated gene transcription. In this study, some supervised learning algorithms such as artificial neural network (ANN), RULES-3 and newly developed keREM-IREM rule induction algorithms were used to analyse to DNA sequence. In the experiments different option of keREM, RULES-3 and ANN were used, and according to the empirical comparisons, the algorithms appeared to be comparable to well-known algorithms in terms of the accuracy of the extracted rule in classifying unseen data.

1.INTRODUCTION

Data mining is the process of finding hidden patterns from data. It has wide range of applications such as, predicting stock prices, identifying suspected terrorists and scientific discovery like analysis of DNA microarray (Hanuman et al., 2009) Researchers can now routinely investigate the biological molecular state of a cell measuring the simultaneous expression of tens of thousands of genes using DNA microarrays (Shelke and Deshmukh, 2007). Datamining can be used in the classification of proteins by basing on its primary structures (sequences) is presented. It contains four steps which include textmining, feature selection, datamining and classification. The sequences of protein are collected in a file (Mhamdi and Elloumi, 2004).

Artificial neural networks are among the newest signal-processing technologies in the engineer's toolbox. The field is highly interdisciplinary. In engineering, neural networks serve two important functions: as pattern classifiers and as nonlinear adaptive filters.

Well-known ANN algorithms are based on the notion of perceptron (Rosenblatt, 1962). Perceptrons can only classify linearly separable sets of instances. If a straight line or plane can be drawn to separate the input instances into their correct categories, input instances are linearly separable and the perceptron will find the solution. If the instances are not linearly separable learning will never reach a point where all instances are classified properly. Multilayered Perceptrons (Artificial Neural Networks) have been created to try to solve this problem (Rumelhart et al., 1986). Properly determining the size of the hidden layer is a problem, because an underestimate of the number of neurons can lead to poor approximation and generalization capabilities, while excessive nodes can result in overfitting and eventually make the search for the global optimum more difficult. An excellent argument regarding this topic can be found in (Camargo and Yoneyama, 2001). Kon & Plaskota also studied the minimum amount of neurons and the number of instances necessary to program a given task into feedforward neural networks (Kon and Plaskota, 2000). There are several algorithms with which a network can be trained (Neocleous and Schizas, 2002). However, the most well-known and widely used learning algorithm to estimate the values of the weights is the Back Propagation (BP) algorithm. Feed-forward neural networks are usually trained by the original back propagation algorithm or by some variant. Their greatest problem is that they are too slow for most applications. One of the approaches to speed up the training rate is to estimate optimal initial weights (Yam and Chow, 2001). Another method for training multilayered feedforward ANNs is Weight-elimination algorithm that automatically derives the appropriate topology and therefore avoids also the problems with overfitting (Weigend et al., 1991). Genetic algorithms have been used to train the weights of neural networks (Siddique and Tokhi, 2001) and to find the architecture of neural networks (Yen and Lu, 2000). There are also Bayesian methods in existence which attempt to train neural networks. Vivarelli & Williams compare two Bayesian methods for training neural networks (Vivarelli and Williams, 2001).

In recent years, there has been a growing amount of research on inductive learning. In its broadest sense, induction (or inductive inference, supervised learning) is a method of moving from the particular to the general - from specific examples to general rules (Quinlan, 1986).

Induction can be considered the process of generalizing a procedural description from presented or observed examples. The purpose of inductive learning is to perform a synthesis of new knowledge, and this is independent of the form given to the input information.

RIPPER is a well-known rule-based supervised learning algorithm (Cohen, 1995). It forms rules through a process of repeated growing and pruning. Other fundamental learning classifiers based on decision rules include the AQ family (Michalski and Chilausky, 1980) and CN2 (Clark and Niblett, 1989). Bonarini gave an overview of fuzzy rule-based classifiers (Bonarini, 2000). Fuzzy logic tries to improve classification and decision support systems by allowing the use of overlapping class definitions. Furnkranz (2001) investigated the use of round robin binarization (or pairwise classification) as a technique for handling multi-class problems with separate and conquer rule learning algorithms. The PART (Frank and Witten, 1998) algorithm infers rules by repeatedly generating partial decision trees, thus combining the two major paradigms for rule generation − creating rules from decision trees and the separate-and-conquer rule learning technique. RULES family algorithms (Aksoy, 1993) obtain the IF-THEN rules from a given set of examples. REX-1 (Akgöbek et al., 2006) uses the entropy value to give a greater priority to the attributes with higher importance and obtain more general rules.

Segments of genome coding for messenger ribonucleic acids (mRNAs), transfer ribonucleic acids (tRNAs), ribosomal ribonucleic acids (rRNAs) are called genes. Among these mRNAs determine the sequence of amino acids in proteins. The mechanism is simple for the prokaryotic cell where all the genes are converted into the corresponding mRNA (messenger ribonucleic acid) and then into proteins.

Genome analysis (Gene finding) typically refers to the area of computational biology that is concerned with algorithmically identifying stretches of sequence, usually genomic DNA, that are biologically functional. This especially includes protein-coding genes, but may also include other functional elements such as RNA genes and regulatory regions. Gene finding is one of the first and most important steps in understanding the genome of a species.

Computational Gene prediction is relatively simple for the prokaryotes where all the genes are converted into the corresponding mRNA and then into proteins. The process is more complex for eukaryotic cells where the coding DNA sequence is interrupted by random sequences called introns.

Some of the questions which biologists want to answer today are (Jayaram and Bhushan, 2000).:

Given a DNA sequence, what part of it codes for a protein and what part of it is junk DNA.

Classify the junk DNA as intron, untranslated region, transposes, dead genes, regulatory elements etc.

Divide a newly sequenced genome into the genes (coding) and the non-coding regions.

In this study, short sequence of DNA is used as an example set to train the keREM, ANN and RULES-3. The features of the DNA sequence are the nucleotides (a,g,c,t). The learning system is requested to generate a classifier that identifies these sequences whether or not they are in the one of functional DNA regions (coding regions).

2.INTRODUCTION TO GENE

A gene is a segment of nucleic acid that contains the information necessary to produce a functional product, usually a protein. Genes consist of a long strand of DNA (RNA in some viruses) that contains a promoter, which controls the activity of a gene, and a coding sequence, which determines what the gene produces.

The genes are made up of a coding alphabet of 4 nucleotides made up of 4 bases: Adenine(A), Thymine (T), Guanine (G) and Cytosine (C).

The bases Adenine (A) and Guanine (G) are Purines; while Thymine (T) and Cytosine (C) are Pyrimidines.

2.1. Universal Genetic Code

There four bases Adenine (A), Thymine (T), Guanine (G) and Cytosine (C) As there are 20 amino acids if we use 2 codons for an amino acid we will short of the representation as 42 = 16. So we use three codons to represent all the 20 amino acids as 43=64. As there are only 20 amino acids and 64 codon representation most of the codon are degenrative. 'ATG' is the start codon and TAG, TGA, TAA are stop codons usually.

2.2. Genome Organisation

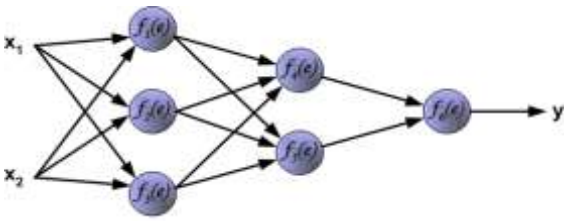
Genome organization refers to the sequential, not the structural organization of the genome. Besides the coding exons, the non-coding DNA in Eukaryotes may fall in the following classes.

Introns: They are DNA sequences inserted between the exons and found in the ORF (Open Reading Frames). They are spliced after the first level of transcription. Most introns are junk inserted within genes. Pseudogenes. 'Dead', non-functional copies of genes present elsewhere in the genome, but no longer of any use.

Retropseudogenes: Like pseudogenes, but have been processed, i.e. lack introns produced by the action of reverse transcriptase (RT) on mRNA, and subsequent incorporation of the cDNA into the genome.

Transposons: Jumping genes, which splice themselves in and out of the genome (in DNA form) randomly, by the action of transposase.

Retrotransposons: Transcribed into an mRNA, which encodes an RT enzyme, which then copies the mRNA back to DNA and incorporates it into the genome.



In fact in humans only 1.5% of the entire genome length corresponds to coding DNA. This 1.5% codes for about 27,000 genes, which in turn code for proteins that are responsible for all the cellular processes

2.3. What are Promoters?

A promoter is a regulatory region of DNA located upstream (towards the 5' region) of a gene, providing a control point for regulated gene transcription.

3 . ARTIFICIAL NEURAL NETWORK (ANN)

Information processing paradigm that is inspired by the way biological nervous systems, such as the rain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurones) working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurones. This is true of ANNs as well (Domingos, 1995).

There are different types of neural networks, which can be distinguished on the basis of their structure and directions of signal flow. Each kind of neural network has its own method of training. Generally, neural networks may be differentiated as follows.

feedforward networks (one-layer networks and multi-layer networks)

recurrent networks

cellular networks

Principles of training multi-layer neural network use backpropagation.

The project describes teaching process of multi-layer neural network employing backpropagation algorithm. To illustrate this process the three layer neural network with two inputs and one output, which is shown in the picture below, is used:

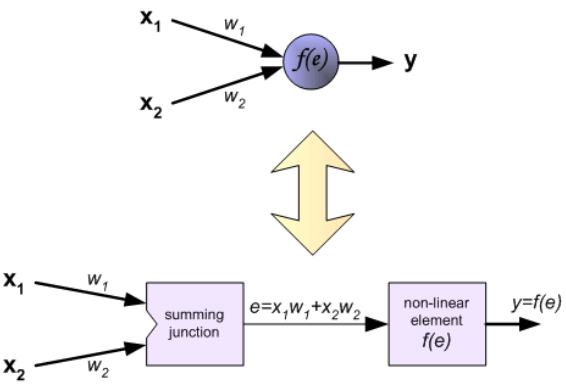


Figure 1: Multi-Layer Neural Network

Each neuron is composed of two units. First unit adds products of weights coefficients and input signals. The second unit realise nonlinear function, called neuron activation function. Signal e is adder output signal, and y = f(e) is output signal of nonlinear element. Signal y is also output signal of neuron.

Figure 2: Teaching Process of Multi-Layer NN

To teach the neural network we need training data set. The training data set consists of input signals (x1 and x2) assigned with corresponding target (desired output) z. The network training is an iterative process. In each iteration weights, coefficients of nodes are modified using new data from training data set. Modification is calculated using algorithm described below: Each teaching step starts with forcing both input signals from training set. After this stage we can determine output signals values for each neuron in each network layer.

4.INDUCTIVE LEARNING

Machine learning algorithms automatically builds a classifier by learning the characteristics of the categories from a set of classified documents, and then uses the classifier to classify documents into predefined categories. (Khan et al., 2010). In recent years, there has been a growing amount of research on inductive learning. In its broadest sense, induction (or inductive inference) is a method of moving from the particular to the general from specific examples to general rules. Michalski explains inductive learning as:

Induction can be considered the process of generalizing a procedural description from presented or observed examples

The purpose of inductive learning is to perform a synthesis of new knowledge, and this is independent of the form given to the input information.

Inductive learning includes learning from examples and learning from observation and discovery. In order to form a knowledge base using inductive learning, the first task is to collect a set of representative examples of expert decisions. Each example belongs to a known class (for example, + or -) and is described in terms of a number of attributes, (for example "hair" or "eyes"). These examples may be specified by an expert as a good tutorial set, or may come from some neutral source such as an archive. The induction process will

attempt to find a method of classifying an example, again expressed as a function of the attributes that explains the training examples and that may also be used to classify previously unseen cases. (Quinlan, 1993).. The outcome of an induction algorithm is either a decision tree or a set of rules. (Quinlan, 1986)..

4.1. RULES-3 Inductive Learning Algorithm

RULES-3 (Aksoy, 1993) is a simple algorithm for extracting a set of classification rules from a collection of examples for objects belonging to one of a number of known classes. An object must be described in terms of a fixed set of attributes, each with its own range of possible values, which could be nominal or numerical. For example, attribute "length" might have nominal values {short, medium, long} or numerical values in the range {-10, 10}.

An attribute-value pair constitutes a condition in a rule. If the number of attributes is Na, a rule may contain between one and Na conditions. Only conjunction of conditions is permitted in a rule and therefore the attributes must be all different if the rule comprises more than one condition.

This algorithm can be summarized as follows:

Step1. Define ranges for the attributes, which have numerical values and assign labels to those ranges.

Step2. Set the minimum number of conditions (Ncmin) for each rule.

Step3. Take an unclassified example.

Step4. Nc=Ncmin-l

Step5. If Nc<Na then Nc=Nc+i

Step6. Take all values or labels contained in the example.

Step7. Form objects which are combinations of Nc values or labels taken from the values or labels obtained in Step6.

Step8. If at least one of the objects belongs to a unique class then form rules with those objects; ELSE go to Step5.

Step9. Select the rule, which classifies the highest number of examples.

Step10. Remove examples classified by the selected rule.

Step11. If there are no more unclassified examples, then STOP; ELSE go to Step3. Here Nc is the number of condition(s) for each rule and Na is the number of attributes for each example.

4.2. keREM Inductive Learning Algorithm

In this section, an algorithm devised for Inductive Learning, newly developed keREM (Inductive Rule Extraction Method) is introduced, which was developed to obtain the IF-THEN rules from a given set of examples. It discards the pitfalls encountered in the some inductive learning algorithms. It uses the value of gain function, to give a greater priority to the attributes with higher importance and obtain rules that are more general.

The algorithm can be summarized as follows:

Step1. In a given training set, probability distribution and class distribution rate of the each attribute-value pairs is computed.

Step2. Power of classification is computed for each attribute in the data set.

Step3. Class-based Gain of the each attribute-value pairs is calculated by using computed probability distributions, class distribution rate and power of classification.

Step4. Any value of which probability distributions one for n=1 can be selected as a rule. The attribute-values are converted into rules. The classified examples are marked.

Step5. Go to step8.

Step6. Beginning from the first unclassified example, combinations with n values are formed by taking the attribute-values whose gain is bigger.

Step7. Each combination is applied to all of the examples in the set of examples. From the values composed of n combinations, those matching with only on class are converted into a rule. The classified examples are marked.

Step8. If all of the examples in the training set are classified then go to step11.

Step9. Perform n=n+1 expression.

Step10. If n<N the go to step6

Step11 if there is more than one rule representing the same examples, the most general one is selected.

Step12. End.

4.3. IREM Inductive Learning Algorithm

Newly developed IREM (Inductive Rule Extraction Method) is introduced In this section, which was developed to obtain the IF-THEN rules from a given set of examples. It uses the class-based entropy value, to give a greater priority to the attributes with higher importance and obtain rules that are more general.

The algorithm can be summarized as follows:

Step1. In a given training set, probability distribution of the each attribute-value pairs is computed.

Step2. The entropy is computed for each attribute and value.

Step3. By using computed probability distributions and entropy, class-based entropy is calculated.

Step4. Any value of which class-based entropy equals zero for n=1 can be selected as a rule. The values are converted into rules. The classified examples are marked.

Step5. Go to step8.

Step6. Beginning from the first unclassified example, combinations with n values are formed by taking the value of the attributes whose class-based entropy is smaller.

Step7. Each combination is applied to all of the examples in the set of examples. From the values composed of n combinations, those matching with only on class are converted into a rule. The classified examples are marked.

Step8. If all of the examples in the training set are classified then go to step11.

Step9. Perform n=n+1 expression.

Step10. If n<N the go to step6

Step11 if there is more than one rule representing the same examples, the most general one is selected.

Step12. End.

5. EXPERIMENTS

Using ANN and RULES-3 systems, experiments were performed for the classification of promoter DNA region. For this purpose promoter data-sets available in the University of California-Irvine's Repository of Machine Leraning Databases were used (Merz and Murphy, 1996).

5.1. Promoter Recognition Experiments with ANN

Using different option of ANN, 30 sets of experiment were performed on the promoter data set of E.coli DNA. 86 (approximately %80 of the data set.) randomly selected instances of the original data set were used as a training data in the experiments (Nayır and Karlı, 2009).

In order to built, train and test an ANN to recognize promoter region of the DNA, the above Matlab code was written. By changing "newff()" function's arguments, some ANN with different number of hidden layer and neurons were designed and tested. Using the code above, an ANN with two hidden layers, the first hidden layer with neuron number of 10 and

the second hidden layer with neuron number of 5 was designed. Activation functions of the input layer and hidden layers were ‘tansig’. And activation function of the output layer was ‘purelin’. Minimum error rate (%95) was gained in this option of the ANN. The following table represents error rate of the ANN with one, two and three hidden layers in our experiments.

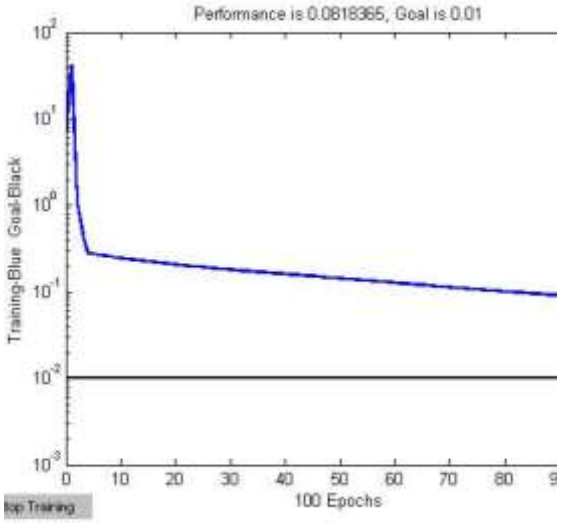


Figure 3: Performance of the NN at 100 epochs.

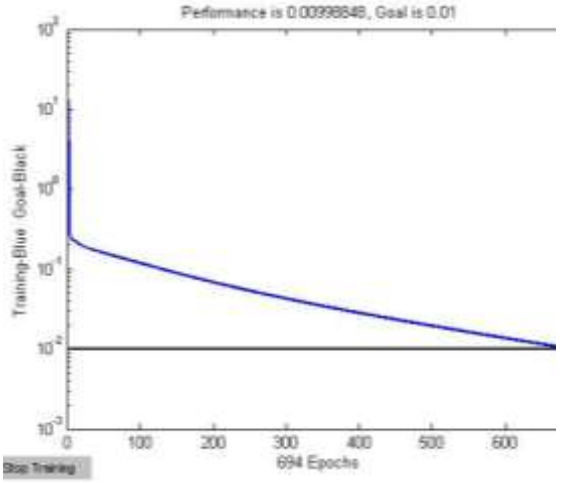


Figure 4: Performance of the NN at 694 epochs.

Table 1: Experiment results of ANN with one hidden layer.

No of set of Exp.	Layer	The number of neuron	Epoch	Min. error rate
1	Input	10	947	35
	1.Hidden	5		
	Output	1		
2	Input	50	244	25
	1.Hidden	25		
	Output	1		
3	Input	100	1404	20
	1.Hidden	50		
	Output	1		

Table 2: Experiment results of ANN with two hidden layer

No of set of Exp.	Layer	The number of neuron	Epoch	Min. error rate
1	Input	25	694	5
	1.Hidden	10		
	2.Hidden	5		
	Output	1		

2	Input	50	851	25
	1.Hidden	25		
	2.Hidden	10		
	Output	1		
3	Input	100	554	10
	1.Hidden	50		
	2.Hidden	25		
	Output	1		

Table 3: Experiment results of ANN with three hidden layer.

No of set of Exp.	Layer	The number of neuron	Epoch	Min. error rate
1	Input	25	785	45
	1.Hidden	10		
	2.Hidden	5		
	3.Hidden	3		
	Output	1		
2	Input	50	978	20
	1.Hidden	25		

	2.Hidden	10		
	3.Hidden	5		
	Output	1		
3	Input	100	889	30
	1.Hidden	50		
	2.Hidden	25		
	3.Hidden	10		
	Output	1		

5.2. Promoter Recognition Experiments with RULES-3

Using different values of number of condition, three sets of experiments were performed on the promoter data set. 40 randomly selected instances of the original data set were used as a training data in the experiments.

When Number of condition was set to 1, rules were produced (Karlı, 2000).

Using the extracted rule, 11 instances could not be recognized. So accuracy on test data was 90,1.

Using different options, three sets of experiments were performed on the promoter data set. As expected, Rules-3 algorithm produced a rule set that classified all training examples correctly. One important conclusion may be driven from table 4.3 is that while number of condition was decreased, extracted rule number decreased. However, accuracy on test data increased. The highest accuracy on test data was gained when number of condition was equal to 1.

Table 4: Results for promoter data set with different values of Nc.

Number of condition	Number of examples	Number of extracted	Accuracy on training	Accuracy on test data(%)
3	40	38	100	61,8
2	40	25	100	76,4
1	40	18	100	90,1

5.3 Promoter Recognition Experiments with keREM

The most important features of keREM algorithm is that it can compute class-based gain of each attribute-value in a given training set. In this context, first, probability distribution, class distribution rate and power of classification of each nucleotide forming DNA sequence were computed in terms of promoter and non-promoter classes. In the next step,

Class-based Gain was computed for each value in the DNA sequence data set by using computed probability distributions, class distribution rate and power of classification. In this way, rules produced by the algorithm were formed by attribute-value whose information value is maximum. The rule set constructed by the method was applied to DNA test set. And the error rate was satisfactory, %97.17

5.4. Promoter Recognition Experiments with IREM

The most important features of IREM algorithm is that it can compute class-based entropy of each attribute-value in a given training set. In this context, first, probability distributions of each nucleotide forming DNA sequence were computed in terms of promoter and non-promoter classes. In the next step, entropy of training set was found. But, the entropy does not contain class information for the value-attribute pairs. Thus, using the entropy of the training set and the probability distributions of the attribute-value, class-based entropy was computed for each value in the DNA sequence data set. In this way, rules produced by the algorithm were formed by attribute-value whose information value is maximum. The rule set constructed by the method was applied to DNA test set. And the error rate was satisfactory, %98.1

6.CONCLUSION

Although from a biochemical view point DNA is a complex molecule, from a computer science view point DNA can be considered a very long string over four alphabets A, C, G T. One of the most important step in analysis of a new DNA sequence is finding out whether or not it contains any genes, and if so, determining exactly where they are. For locating functional region in newly sequenced DNA data keREM, IREM, artificial neural network (ANN) and RULES-3 can be used in such a way that known region in mapped sequences is given as input to the systems. Then, the output classifiers of the keREM, IREM, ANN and RULES-3 are used to locate functional regions of newly sequenced data. In this study, keREM, IREM, ANN and RULES-3 were used to locate promoter region of DNA data.

Some portions of DNA serve as protein coding regions. However, some portions serve as regulatory markers for the processes that convert coding regions into protein. One of these regulatory regions is a promoter that occurs before coding regions to signal where the transcription process begins. To recognize promoter region, some sort of experiments were performed by using different options of keREM, IREM, ANN, different numbers of hidden layers and neurons, and RULES-3,

In the first sort of experiments, one hidden layer with the number of neuron ranging from 5 to 50 was used in ANN. In these experiments, minimum error rate was 15. In the second sort of experiments, two hidden layers were used, the first hidden layer with the number of neuron ranging from10 to 50 and the second hidden layer with the number of neuron ranging from 5 to 25. And minimum error rate of these experiments was 5. This was the best result gained from the experiments. In the last sort of experiments three hidden layers were used, the first hidden layer with the number of neuron ranging from 10 to 50, the second hidden layer with the number of neuron ranging from 5 to 25 and the last hidden layer with the number of neuron ranging from 3 to 10. And minimum error rate was 20.

Using different options of RULES-3, three sets of experiments were performed on the promoter data set. As expected, Rules-3 algorithm produced a rule set that classified all training examples correctly. The highest accuracy on test data was gained when number of condition was equal to 1.

It is observed that, rules formed by IREM were more general than that of keREM, RULES-3 and ANN. Only 2 examples could not be recognized out of 106 example test set. As a result, it was determined that the error rate of the IREM was lower than the error rate of the keREM, RULES-3 and ANN for DNA sequence test set.

Table 5: The errors of some machine learning algorithms on promoter data set.

System	Errors	Comments
REX-1	0/106	Inductive L.A

ILA	0/106	Inductive L.A
IREM	2/106	Class-based entropy
keREM	3/106	Class-based gain
KBANN	4/106	A hybrid ML system
ANN	6/106	ANN with two hidden layer
BP	8/106	Standard backpropagation with one layer
RULES-3	11/106	Nc=1
O'Neill	12/106	Ad hoc tech. from the bio. lit.
Near-Neigh	13/106	A nearest neighbours algorithm
ID3	19/106	Quinlan's decision builder
ANN	21/106	ANN with three hidden l.

As it may be driven from table 5, using the same data set, error rate of accuracy on the test data of the REX and ILA is 0/106, which is the minimum error rate. The second best result belongs to IREM with the error rate 2/106.

REFERENCES

Quinlan, R. (1986). Induction Decision Tree. Machine Learning, 1, 81-106.

Jayaram, P., Bhushan, K. (2000). Bioinformatics For Better Tomorrowç. Indian Institute of Technology, Hauz Khas: New Delhi.

Domingos, P. (1995). Rule Induction and Instance Based Learning. IJCAI-95.

Clark, P., Niblett, T. (1989). The CN2 Induction Algorithm. Machine Learning, 3, 261-283.

Aksoy, M. S. (1993). New Algorithms for Machine Learning, University of Wales: Cardiff.

Merz, C. J., Murphy, P. M. (1996). UCI Repository of Machine Learning Database, Retrieved April 1, 2010, from http://www.ics.uci.edu/~mlearning/MLlearning.

Nayır, A., Karlı, G. (2009). Application of Artificial Neural Network (ANN) to DNA Sequence Analysis, In AICT 2009, The 3rd IEEE International Conference on Application of Information and Communication Technologies.

Karlı, G., (2000). Application of Rule Induction Algorithms to DNA Sequence Analysis. Fatih University: İstanbul.

Akgöbek, Ö., Aydın, Y. S., Aksoy, M. S. (2006). A new algorithm for automatic knowledge acquisition in inductive learning. Knowledge-based Systems, 19, 388-395.

Shelke, RR., Deshmukh, V. M. (2007). Computational analysis of DNA microarray data using data mining. Biosciences Biotechnology Research Asia, 4, 321-324.

Mhamdi, F., Elloumi, M., Rakotomalala, R. (2004). Textmining feature selection and datamining for proteins classification. In ICTTA 2004, International Conference on Information and Communication Technologies: From Theory to Applications, 457-458.

Hanuman, T., Raghava, M., Siva, A., Mrithyunjaya, K , Chandra, V. (2009). Performance Comparative in Classification Algorithms Using Real Datasets. Comput Sci Syst Biol, 2, 97-100.

Khan, A., Baharudin, B., Lee, L. H, Khan, K, A. (2010). Review of Machine Learning Algorithms for Text-Documents Classification, Journal Of Advances In Information Technology.

Rosenblatt, F. (1962). Principles of Neurodynamics. Spartan: New York.

Rumelhart, D. E., Hinton, G. E., Williams, R. J. (1986). Learning Internal Representations by Error Propagation. In: Rumelhart D E, McClelland J L et al. (eds.) Parallel Distributed Processing: Explorations in the Microstructure of Cognition. MIT Press, Cambridge, MA, 1, 318-362.

Camargo, L. S. & Yoneyama, T. (2001). Specification of Training Sets and the Number of Hidden Neurons for Multilayer Perceptrons. Neural Computation 13: 2673–2680.

Kon, M. & Plaskota, L. (2000), Information complexity of neural networks, Neural Networks 13: 365–375.

Neocleous, C. & Schizas, C., (2002), Artificial Neural Network Learning: A Comparative Review, LNAI 2308, pp. 300–313, Springer-Verlag Berlin Heidelberg.

Yam, J. & Chow, W. (2001). Feedforward Networks Training Speed Enhancement by Optimal Initialization of the Synaptic Coefficients. IEEE Transactions on Neural Networks, 12, 430-434.

Weigend, A. S., Rumelhart, D. E., & Huberman, B. A. (1991). Generalization by weight-elimination with application to forecasting. In: R. P. Lippmann, J. Moody, & D. S. Touretzky (eds.), Advances in Neural Information Processing Systems 3, San Mateo, CA: Morgan Kaufmann.

Siddique, M. N. H. and Tokhi, M. O. (2001), Training Neural Networks: Backpropagation vs. Genetic Algorithms, IEEE International Joint Conference on Neural Networks, 4, 2673–2678.

Yen, G. G. and Lu, H. (2000), Hierarchical genetic algorithm based neural network design, IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks, 168–175.

Vivarelli, F. & Williams, C. (2001). Comparing Bayesian neural network algorithms for classifying segmented outdoor images. Neural Networks, 14, 427-437.

W. W. Cohen. (1995). Learning to classify English text with ILP methods. In Luc De Raedt, editor, Advances in inductive logic programming, 124–143. IOS Press, Amsterdam, NL.

Michalski, R. S., Chilausky, R. L. (1980). Learning by being told and learning from examples: an experimental comparison of the two methods of knowledge acquisition in the context of developing and expert system for soybean disease diagnosis. Policy Analysis and Information Systems, 4.

Bonarini, A. (2000), An Introduction to Learning Fuzzy Classifier Systems. Lecture Notes in Computer Science, 1813, 83-92.

Furnkranz, J. (1997). Pruning algorithms for rule learning. Machine Learning, 27, 139-171.

Frank, E. & Witten, I. (1998). Generating Accurate Rule Sets Without Global Optimization. In Shavlik, J., (eds), Machine Learning: Proceedings of the Fifteenth International Conference, Morgan Kaufmann Publishers.

Pham, D.T., Dimov, S.S. (1997). The RULES-4 incremental inductive learning algorithm, in: R.A. Adey, G. Rzevski, R. Teti (Eds.), Applications of Artificial Intelligence in Engineering XII. Computational Mechanics Publications.