Classification and Feature Extraction of High-Dimensionality Binary Patterns using a GA to Evolve Rule

Min Pei [1,2] Ying Ding [2] William F. Punch, III [3] Erik D. Goodman [2]

[1] **Beijing Union University, Beijing China**

[2] **Case Center for Computer-Aided Engineering and Manufacturing**

[3] **Intelligent Systems Laboratory, Department of Computer Science**

Genetic Algorithms Research and Applications Group (GARAGe)
112 Engineering Building, Michigan State University,
East Lansing, MI 48824
Tel: (517)-353-4973. Fax: (517)-355-7516 e-mail: pei@egr.msu.edu

Abstract

A genetic algorithm system is developed and applied to classification and feature extraction of high-dimensionality binary patterns. We represent classifiers/rules in the genetic algorithm, and evolve optimal classifiers for the data sets examined. The approach is very computationally efficient when compared to other GA approaches to evolving classifiers. The approach was applied successfully to three biological binary pattern data sets. It appears to have potential for application in many other fields. It can also be further developed to solve a fairly broad class of complex pattern recognition problems.

In decision-theoretic or statistical approaches, the classification or description of a set of processes or events is based on a set of selected features extracted from the input patterns. Therefore, feature selection and extraction is a crucial problem to the performance of pattern recognition, and strongly affects classifier design. Defining appropriate features requires interaction with experts in the application area. In practice, there is much noise and redundancy in most high dimensionality complex pattern sets. Therefore, it may be hard, even for experts, to determine a minimum or optimum feature set. The so-called “curse of dimensionality” is a difficult problem for both statistical pattern recognition (SPR) and Artificial Neural Network techniques(ANN). . Researchers have discovered that many learning procedures do not scale -- i.e., these procedures simply fail or produce unsatisfactory results when applied to problem of larger sizes [1]. We have previously addressed this problem using a hybrid approach -- a Genetic Algorithm combined with the K Nearest Neighbor technique (GA/KNN) [2]. By applying this hybrid method to the classification and feature extraction of high dimensionality patterns in various real-world domains, we proved we could increase the percent of correct classifications and find a nearly optimal feature subset for classification. However, the computational cost of this method is very high and requires a parallel or distributed processing environment to be attractive.

The current work utilizes a genetic algorithm that develops classification “rules” for both classification and feature extraction of high-dimensionality patterns. This inductive learning method is simpler to implement than the previous hybrid system, and requires substantially fewer computation cycles to achieve answers of similar quality. The system was developed for application to both classification and feature extraction of high dimensionality patterns. In particular, the focus is on binary pattern recognition -- i.e., recognition in which the decision of class inclusion is binary. As such, we have named our system HDBPCS (High Dimensionality Binary Pattern Classification System) and have applied it to three different sets of biological data, with encouraging results. The accuracy of this method is significantly better than the classical KNN method, and significantly faster than our previous hybrid method. Our work builds on that of Wilson[4], Bonneli[5], Riolo[6], Oliver[7], Sedbrook et. al[8], and Liepins[9], all of whom studied systems for inductive learning of classification.

Classification and Feature Extraction of High-Dimensionality Binary Patterns using a GA to Evolve Rules

Abstract

A genetic algorithm system is developed and applied to classification and feature extraction of high-dimensionality binary patterns. We represent classifiers/rules in the genetic algorithm, and evolve optimal classifiers for the data sets examined. The approach is very computationally efficient when compared to other GA approaches to evolving classifiers. The approach was applied successfully to three biological binary pattern data sets. It appears to have potential for application in many other fields. It can also be further developed to solve a fairly broad class of complex pattern recognition problems.

In decision-theoretic or statistical approaches, the classification or description of a set of processes or events is based on a set of selected features extracted from the input patterns. Therefore, feature selection and extraction is a crucial problem to the performance of pattern recognition, and strongly affects classifier design. Defining appropriate features requires interaction with experts in the application area. In practice, there is much noise and redundancy in most high dimensionality complex pattern sets. Therefore, it may be hard, even for experts, to determine a minimum or optimum feature set. The so-called "curse of dimensionality" is a difficult problem for both statistical pattern recognition (SPR) and Artificial Neural Network techniques(ANN). . Researchers have discovered that many learning procedures do not scale -- i.e., these procedures simply fail or produce unsatisfactory results when applied to problem of larger sizes [1]. We have previously addressed this problem using a hybrid approach -- a Genetic Algorithm combined with the K Nearest Neighbor technique (GA/KNN) [2]. By applying this hybrid method to the classification and feature extraction of high dimensionality patterns in various real-world domains, we proved we could increase the percent of correct classifications and find a nearly optimal feature subset for classification. However, the computational cost of this method is very high and requires a parallel or distributed processing environment to be attractive.

The current work utilizes a genetic algorithm that develops classification "rules" for both classification and feature extraction of high-dimensionality patterns. This inductive learning method is simpler to implement than the previous hybrid system, and requires substantially fewer computation cycles to achieve answers of similar quality. The system was developed for application to both classification and feature extraction of high dimensionality patterns. In particular, the focus is on binary pattern recognition -- i.e., recognition in which the decision of class inclusion is binary. As such, we have named our system HDBPCS (High Dimensionality Binary Pattern Classification System) and have applied it to three different sets of biological data, with encouraging results. The accuracy of this method is significantly better than the classical KNN method, and significantly faster than our previous hybrid method. Our work builds on that of Wilson[4], Bonneli[5], Riolo[6], Oliver[7], Sedbrook et. al[8], and Liepins[9], all of whom studied systems for inductive learning of classification.

The paper is arranged as follows: introduction, characteristics of the biological data patterns studied, conceptual design of HDBPCS, application of the method to classification of biological patterns to find a small optimal subset of features for classifying these patterns, and conclusions.

1. Introduction

Most machine learning systems include one of: logical reduction, decision trees, or neural networks, all of which induce classifications based on preclassified examples. All these systems learn by example -- i.e., they require examples which demonstrate, for a sample input, the expected output. Typically, such systems have a particularly

difficult time determining classifications when the data are either noisy or redundant. To address these problems, a new family of inductive learning techniques based on the concept of Genetic Based Machine Learning(GBML) have come into use. The most common GBML architecture is the so-called Classifier System, which has been widely studied in the past ten years. Elements of the classical Holland-style classifier system were used to develop our HDBPCS -- namely, the Holland classifiers/rules and a genetic algorithm, but in a more *standard genetic algorithm* environment. Note that there is some confusion over the term "classifier system" in the literature. For example, [8] and [9] both use an approach similar to HDBPCS for pattern classification, but call their systems "Classifier Systems". This is true in the sense that what is evolved is a set of classifiers/rules, but they are not classifier systems in the sense described by Holland; for example, their systems have no bidding, no credit apportionment algorithm (such as the bucket-brigade), no message list, etc.

Binary feature classification was selected for study because it is both a common problem and easy to represent in the HDBPCS. As with other inductive learning approaches, this algorithm uses a collection of labeled "training" data to drive learning. What is unique about the approach is the "evolution" of classifiers/rules to classify the training data, and subsequent testing of the accuracy of those classifiers on data not shown to the classifier in training. From this process, there are two results: 1) "good" rules are created for classifying "unknown" data, and 2) domain researchers are shown those features which are important for the classification, based on the features used by the rules.

2. Biological Data for Classification

We have applied our HDBPCS to a number of real-world, complex examples from biological research. Researchers at Michigan State University's Center for Microbial Ecology (CME) have microbial samples from different environments in agriculture or industry for study. Their goal is to try to determine if there exists a set of tests that would allow researchers, and ultimately end users, to distinguish from which environment the samples were taken. One such test suite is the Biolog© test suite. Biolog consists of a plate of 96 wells, with a different substrate in each well. These substrates (various sugars, amino acids and other nutrients) are assimilated by some microbes and not by others. If the microbial sample processes the substrate in the well, that well changes color, which can be recorded photometrically. Thus large numbers of samples can be processed and characterized based on the substrates they can assimilate. Each sample was tested on the 96 features provided by Biolog plus (sometimes) several other features provided by various taxonomic tests.

Using the Biolog test suite to generate data, three test sets were used in showing the effectiveness of the HDBPCS.

Rhizosphere Data Set

Soil samples were taken from three environments found in agriculture:
1) near the roots of a crop (rhizosphere), 2) away from the influence of the roots (non-rhizosphere), or 3) a fallow field (crop residue). There are 300 samples in total, 100 samples per area.

2,4-D Data Set

Selected soil samples collected from a site that is contaminated with 2,4-D (dichlorophenoxyacetic acid), a pesticide. There are 3 classes, based on three genetically similar microbial isolates which show the ability to degrade 2,4D. There were are a total of 232 samples.

Chlorinated Organic Data Set

Selected water samples from the wastewater treatment output of a bleach kraft mill and from the river that supplies the mill. They are river, mill clarifier, mill lagoon, and mill pond -- 4 classes totalling 168 samples.

Two questions are asked of the HDBPCS:

1) Classification -- whether the samples from the different environments could be distinguished (such as the 3 environments for rhizosphere data, 3 classes for 2,4-D data, and 4 locations for the chlorinated organic data).
2) Identification -- which of the available features are most important for the discrimination and which are acting primarily as noise -- that is, non-contributing features.

There are several characteristics of this type of problem:

1) High dimensionality -- the feature space is quite large (in our three examples, they are 3x100x96, 5x232x96, and 4x168x117, respectively), and therefore computationally expensive for traditional approaches.
2) Noise -- the data are very noisy. The microstructure of samples is extremely heterogeneous, particularly with the soil samples.

3. Method

Our HDBPCS is an adaptive production rule system which uses a genetic algorithm to discover new rules. We use a GA to find the best classification rule for the classification of binary patterns based on a certain set of known samples. Here the GA, like most machine learning systems, performs supervised learning. The characteristics of the HDBPCS genetic algorithm are described in the following sections.

3.1 Data Characteristics

Our HDBPCS work focuses on the analysis of dichotomous feature spaces, or binary pattern spaces. Dichotomies are very common in biological tests, medical diagnostics, engineering and economic analyses, and many other fields. In binary pattern spaces, features are recorded as either present or absent, and laboratory test results are noted as either positive or negative, normal or abnormal, etc. (feature $x_i \in \{0,1\}$). This is typical of many pattern recognition problems, and yields a feature vector:

$$\underline{X} = (x_1, x_2, ...x_i, \ ..., x_n). \qquad i = 1, 2, ... n,$$

where $x_i \in \{0,1\}$, and n is the number of features. A known data set consists of preclassified examples, which are *labeled* feature vectors.

3.2 Chromosome Structure

A classifier is a production rule. The <condition> part of the rule in HDBPCS is a string of *class vectors*, one class vector per class in the data set being examined. Each class vector consists of n elements, where n is the number of features being used for the classification task. The <message> part of the classifier is a number which indicates into which class the classifier places an example. Therefore the rule form is as follows:

<classifier> : : = <condition> : <message>

An example of a specific rule is:

$$(x_{11}, x_{12}, \ ... x_{1n}), (x_{21}, x_{22}, \ ... x_{2n}), ... (x_{k1}, x_{k2}, \ ... x_{kn}) : class_i ,$$

where $class_i$ is a label for one of the k classes.

The <condition> is a simple pattern matching device. The alphabet of the class vector consists of 0, 1, and a don't-care character, i.e., $\mathbf{x_{ji} \in \{0, 1, \#\}}$

3.3 Training Data

The training data consists of a *training vector* of size n (again, n indicating the number of features being used) and a known classification label.

3.4 Matching method

Each rule in the population is matched against the training data set. In this procedure, every class vector in each rule is compared with the training vector. Thus, for a training set with three classes, the training vector would be compared with three class vectors in each rule. The number of matching features in each class vector of the rule is counted, and that rule's class vector with the highest number of class vector/training vector matches determines the classifier message. Since the class of each training sample is already known, this classification can then be judged correct or not. Based on the accuracy of classification, the matching classifier can be directly rewarded or punished.

Running the HDBPCS yields a small set of "best" classification rules which classify the training set more accurately than other rule sets. By examining those "good" rules, one can determine those features which are most useful for classification. To keep the number of features used to a minimum, one must provide a mechanism by which to foster not only accuracy of classification, but also minimum cardinality of the feature set used. One way to accomplish this is to add another term to the fitness function, a so-called penalty term. Another approach is to choose different proportions of wild cards # in the classifiers of the initial population.

3.5 Fitness function

Each rule's fitness is based on the classification of the known samples. The form of the fitness function is as follows:

$$\textbf{Fitness = CorrectPats/TotPats + } \alpha * \textbf{ n\_don'tcare/TotPats.}$$

where TotPats is the number of total patterns (training samples) to be examined and CorrectPats is the number of patterns correctly classified by the rule, and n_don'tcare is the number of invalid features (see below) for classification. The constant α is used to tune the two terms of the fitness function, and its value is determined on a problem-specific basis.

3.6 Genetic Operators

Crossover was standard one-point crossover. Mutation was standard bit-modification. Operations were performed only on the condition part of the classifiers/rules. Elitism was used to keep the best solution in the population. G=1.0; that is, the entire population (except for the best solution) is replaced each generation.

3.7 Determining Invalid Features

To keep the complexity of the rule generation to a minimum, an optimization step is applied to remove redundant features. For each rule, it is determined whether each class vector has the same value (1 or 0) or a don'tcare (#) at the same position. If they do, the n_don'tcare variable is incremented, as this feature is useless for classification.

$$\begin{array}{l} (\mathbf{x_{11}, x_{12}, \ \ldots x_{1n}}) \\ (\mathbf{x_{21}, x_{22}, \ \ldots x_{2n}}) \\ \quad \mathbf{\ldots} \\ (\mathbf{x_{k1}, x_{k2}, \ \ldots x_{kn}}) \end{array}$$

Figure 1: Check Matrix

4. Results

We applied the HDBPCS to the three biology data sets described in section 2. The parameters used were those that gave the best results over a number of runs. The initial populations were created randomly, with the value at each vector position generated using probability 0.3 for a 1 or 0, and 0.4 for a #. The population size was 200, the selection rate was 0.6, the crossover rate was 0.6, and the mutation rate was 0.005.

In several runs, the HDBPCS system discovered a reasonable classification rule for each data set. The HDBPCS outperformed a standard KNN method in every case, and its performance approached the hybrid GA/KNN method in most cases. It is most important to note, however, that the computational resources required for the hybrid method running under similar circumstances would be approximately 14 days of computation (if run on a single processor), whereas the HDBPCS required only 3 hours on the same processor. The accuracy results are shown in Table 1 below.

Table 1: Classification Results using HDBPCS System

		KNN Correctness rate	GA/KNN Correctness rate	HDBPCS Correctness rate
Rhizosphere data	Training	71.00%	82.00%	80.00%
	Test	68.00%	79.90%	70.00%
2,4-D data	Training	93.36%	99.17%	98.61%
	Test	91.70%	98.00%	96.00%
Chlorinated data	Training	46.43%	76.79%	79.86%
	Test	63.24%	70.47%	66.67%

By examining the best rules in each data set, we can infer which features are most important for the classification task for these data. Those features (out of the 96 possibile in Biolog) are listed in Table 2.

Table 2: The Number of Valid Features and their Indices

	number of valid features	Index
Rhizosphere data	23	1 5 9 13 18 20 21 24 26 29 40 43 45 47 58 61 72 81 84 89 91 93 95
2,4-D data	22	9 17 18 20 26 27 28 47 48 51 52 54 62 64 71 75 81 85 89 91 92 93

Table 2: The Number of Valid Features and their Indices

	number of valid features	Index
Chlorinated data	60	0 1 4 6 7 11 12 13 14 15 17 19 20 25 29 31 32 33 35 37 38 42 44 46 47 49 50 52 56 57 59 61 64 67 68 69 70 72 73 74 80 82 85 86 88 90 91 93 95 96 97 98 99 102 104 105 106 109 110 116

The information shown in Table 2 is particularly important to the biological researchers, as it provides information that can be used to derive new hypotheses. Furthermore, by reducing those features important for discrimination, HDBPCS gives the biological (or any domain) researcher information on possible foci for future research efforts. In particular, for our biology examples, it was shown that HDBPCS can indeed distinguish between different bacterial communities on the basis of which substrates are being used -- i.e., 2,4-D degraders, based on a standardized test that measures substrate uptake. This may help to distinguish what substrates are available in each of the communities, and may have ramifications in determining which bacteria are suitable for bioremediation of particular environments.

Finally, Table 3 shows that the errors of classification in each class are not uniform across the data sets examined. This information may also provide some hints for research in the domain being studied.

.

Not all high dimensionality data sets are readily classified, or can be classified with high accuracy. The reason is some classes in the data set have overlapping features or lack of the significant information for discrimination among the classes. Even if a particular H.D. data set is classifiable, in order to obtain a good set of classification rules, we need to pay attention to the quality and quantity of training samples given. The sample (training) data set needs to be representative, and it must also be large enough to allow for effective training. Otherwise, it will permit a ‘false’ set of rules to be induced, based on a few examples which do not span the space of possible members of each class. The knowledge we obtain from such inductive learning is incomplete.

Table 3: Classification Results using HDBPCS System

		1 class Correctness rate	2 class Correctness rate	3 class Correctness rate	4 class Correctness rate	Total Correctness rate
Rhizosphere data	Training 90%	77.77%	86.66%	75.55%%		80.00%
	Test 10%	60.00%	90.00%	60.00%		70.00%
2,4-D data	Training 90%	98.90%	97.77%	1.00%		98.61%
	Test 10%	90.90%	1.00%	1.00%		96.00%
Chlorinated data	Training 90%	88.57%	61.53%	88.33%	65.85%	79.86%
	Test 10%	1.00%	50.00%	85.71%	20.00%	66.67%

5. Conclusion

This paper presents a genetic algorithm system, HDBPCS, for classification and feature extraction for high-dimensionality patterns. HDBPCS has been applied to a number of real-world, complex biological data sets, and it has been proven both computationally efficient in comparison to similar methods, and reasonably accurate. This approach certainly has potential application in many other fields. The approach is being explored further, particularly through examination of new representations and new matching methods.

For more information on this and related research at Michigan State University's Genetic Algorithms Research and Applications Group (GARAGe), visit the web server, http://isl.msu.edu/GA.

6. Acknowledgments

This work was supported in part by Michigan State University's Center for Microbial Ecology and the Beijing Natural Science Foundation of China.

7. Appendix

The best rule from the rhizosphere run:

Iteration = 400, Population Size = 200, Classifier length = 288 correct rate = 0.800000
0777778. 0.866667 0.755556 0.800000
Number of effective features = 23
The index of features:
1 5 9 13 18 20 21 24 26 29 40 43 45 47 58 61 72 81 84 89 91 93 95
001#000##000#01###0##0##1#0010#0###01#1#1####0#0###001##1111#1#1##0#0###0###0#00#0010#1##0###1#1
01##01##01#000###11101#00110#1####1#10#####0111#00##0#0#1#0#00#1#1#0#1#11#1#01#0#1###11000#100##
0##101#0#10001##011#10####10##0#0#1##0100#01##11##100##111010001110#0#0#0110##00110#1#1001#0#0#0

The best rule from the 2,4-D run:

Iteration = 400, Population Size = 200, Classifier length = 288 correct rate = 0.989011
0.989011 0.977778 1.000000 0.98611
Number of effective features = 22
The index of features:
9 17 18 20 26 27 28 47 48 51 52 54 62 64 71 75 81 85 89 91 92 93
0#0##1#0#0000#1#000#0##01#000#0##0#11#0#0#0#1#1111111#10##0#001#00011######110#1#0#011##11#010##
010##1##01#001##0110#####0111#0010#1#0#1##0#10#01#10001#00####001#0#1##0110#1#11#11##0101010#111
01#0110001##0111#1##1##01#1111##1##########0##110#1#1000##010#0#0#0##0011#00##1#0##01110##11001#

The best rule from the chlorinated organic run:

Iteration rate=1500, Population=200, Classifier length=468. correct rate = 0.798658
0.798658 0.885714 0.615385 0.883333 0.658537
Number of effective features = 60
The index of features:
0 1 4 6 7 11 12 13 14 15 17 19 20 25 29 31 32 33 35 37 38 42 44 46 47 49 50 52 56 57 59 61 64 67 68 69 70 72 73 74 80 82 85 86 88 90 91 93 95 96 97 98 99102 104 105 106 109 110 116
0100#000#0###11#1##010##1####0#100#11000#100110100001##101##01#111##010001######0#0001#01#1##0#1##1#0#1010#0010#001#0

01##0#10##110##1#1#110#010##1110#0#1##1#0##0#1#1#1000#1#0#100##10#101#0##1#1##00##10##001#1#11#01#11010###100111####1
##00000###1#1000##000#10#10111###11111####1#1110#01#00#1#01100####11#01#101#####0#1000#00#011##1#00####00100#0##00#01
10#01#111##0###0#00#0#1##0#####11110#10#0##001####0#1#1110###101##10#1100#011#0#10#0#01000101#1101#00#1#000##00100#01

8. References:

[1] A.K. Jain Artificial Neural Networks and Statistical Pattern Recognition. Editor preface, 1991, Elsevier Science Publishers B.V.

[2] W.F. Punch, E.D. Goodman, Min Pei, et al. Further Research on Feature Selection and Classification Using Genetic Algorithms. In Proc. Fifth Inter. Conf. Genetic Algorithms and their Applications (ICGA), 1993 p.557.

[3] D. E. Goldberg. Genetic Algorithms in Search, Optimization and Machine Learning. New York: Addison Wesley, 1989.

[4] S.Wilson. Classifier Systems and the Animat Problem. In Machine Learning Journal 2: 199-228, Kluwer Academic Publishers, Boston, 1987.

[5] Bonelli, Pierre and Parodi, Alexandre (1991). An Efficient Classifier System and its Experimental Comparison With Two Representative Learning Methods on three Medical Domains. In Proc. The Fourth Inter. Conf. Genetic Algorithms and their Application (ICGA)1991 p. 288.

[6] R. L. Riolo Modeling Simple Human Category Learning with a Classifier System. In Proc. The Fourth Inter. Conf. Genetic Algorithms and their Application (ICGA)1991 p. 324.

[7] J. Oliver, Discovering Individual Rules: An application of Genetic Algorithms. In Proc. The fifth Inter. Conf. Genetic Algorithms and their Application (ICGA)1993 p.216.

[8] T. A. Sedbrook, H. Wright, R. Wright, Application of a Genetic Classifier for Patient Triage. In Proc. The Fourth Inter. Conf. Genetic Algorithms and their Application (ICGA)1991 p. 334.

[9] G.E. Liepins, L.A. Wang, Classifier System Learning of Boolean Concepts In Proc. The Fourth Inter. Conf. Genetic Algorithms and their Application (ICGA)1991 p. 318.