Brief Papers

Dimensionality Reduction Using Genetic Algorithms

Michael L. Raymer, William F. Punch, Erik D. Goodman, Leslie A. Kuhn, and Anil K. Jain

***Abstract*—Pattern recognition generally requires that objects be described in terms of a set of measurable features. The selection and quality of the features representing each pattern have a considerable bearing on the success of subsequent pattern classification. Feature extraction is the process of deriving new features from the original features in order to reduce the cost of feature measurement, increase classifier efficiency, and allow higher classification accuracy. Many current feature extraction techniques involve linear transformations of the original pattern vectors to new vectors of lower dimensionality. While this is useful for data visualization and increasing classification efficiency, it does not necessarily reduce the number of features that must be measured since each new feature may be a linear combination of all of the features in the original pattern vector. Here, we present a new approach to feature extraction in which feature selection, feature extraction, and classifier training are performed simultaneously using a genetic algorithm. The genetic algorithm optimizes a vector of feature weights, which are used to scale the individual features in the original pattern vectors in either a linear or a nonlinear fashion. A masking vector is also employed to perform simultaneous selection of a subset of the features. We employ this technique in combination with the k nearest neighbor classification rule, and compare the results with classical feature selection and extraction techniques, including sequential floating forward feature selection, and linear discriminant analysis. We also present results for the identification of favorable water-binding sites on protein surfaces, an important problem in biochemistry and drug design.**

***Index Terms*—Curse of dimensionality, feature extraction, feature selection, genetic algorithms, pattern classification.**

I. Introduction

A. Feature Selection and Extraction

IN ATTEMPTING to classify real-world objects or concepts using computational methods, the selection of an appropriate representation is of considerable importance. For classical pattern recognition techniques, the patterns are generally represented as a vector of feature values. The selection of features can have a considerable impact on the effectiveness of the resulting classification algorithm [1], [2]. It is not often known in advance which features will provide the best discrimination between classes, and it is usually not feasible to measure and represent all possible features of the objects being classified. As a result, feature selection and extraction methods have become important techniques for automated pattern recognition [1]–[4], exploratory data analysis [5], and data mining [6].

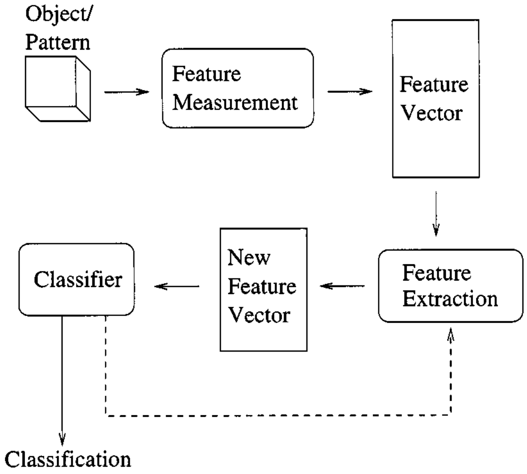

Fig. 1. Model for a pattern recognition system using feature extraction. Feedback from the classifier allows the feature extraction module to iteratively search for a feature vector that provides optimal classification accuracy.

The main purpose of feature selection is to reduce the number of features used in classification while maintaining an acceptable classification accuracy. Less discriminatory features are eliminated, leaving a subset of the original features which retains sufficient information to discriminate well among classes. Feature extraction is a more general method in which the original set of features is transformed to provide a new set of features. Fig. 1 shows the role of feature extraction in a pattern recognition system. The genetic algorithm (GA) feature extractor presented here utilizes feedback from the classifier to the feature extractor; hence, we have the feedback loop in the figure. Most well-known feature extraction methods involve only a single iteration, and do not include such feedback.

Many classical feature extraction methods perform a linear transformation of the original feature vectors. Formally, these linear feature extraction techniques can be stated as follows: given an $n \times d$ pattern matrix $\mathcal{A}$ (n points in a d-dimensional space), derive an $n \times m$ pattern matrix $\mathcal{B}$, $m < d$, where $\mathcal{B} = \mathcal{A}\mathcal{H}$ and $\mathcal{H}$ is a $d \times m$ transformation matrix.

Manuscript received March 8, 1999; revised October 4, 1999. The work of L. A. Kuhn was supported by the National Science Foundation under Grant DBI-9600831.

M. L. Raymer, W. F. Punch, and A. K. Jain are with the Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824 USA (e-mail: raymermi@cse.msu.edu; punch@cse.msu.edu; jain@cse.msu.edu).

E. D. Goodman is with the Case Center for Computer-Aided Engineering and Manufacturing, Michigan State University, East Lansing, MI 48824 USA (e-mail: goodman@egr.msu.edu).

L. A. Kuhn is with the Department of Biochemistry, Michigan State University, East Lansing, MI 48824 USA (e-mail: kuhn@agua.bch.msu.edu).

Publisher Item Identifier S 1089-778X(00)04476-3.

1089–778X/00$10.00 © 2000 IEEE

According to this formalization, the classical methods for linear feature extraction can be specified according to the method of deriving the transformation matrix $\mathcal{H}$. For unsupervised linear feature extraction, the most common technique is principal component analysis. For this method, the columns of $\mathcal{H}$ consist of the eigenvectors of the $d \times d$ covariance matrix of the given patterns. It can be shown that the new features produced by principal component analysis are uncorrelated, and maximize the variance retained from the original feature set [7]. The corresponding supervised technique is linear discriminant analysis. In this case, the columns of $\mathcal{H}$ are the eigenvectors corresponding to the nonzero eigenvalues of the matrix $S_W^{-1}S_B$, where S_W is the within-class scatter matrix and S_B is the between-class scatter matrix for the given set of patterns. Deriving $\mathcal{H}$ in this way maximizes the separation between class means relative to the covariance of the classes [7]. In the general case, the matrix $\mathcal{H}$ is chosen to maximize some criteria, typically related to class separation or classification accuracy for a specific classifier. In this view, feature selection is a special case of linear feature extraction, where the off-diagonal entries of $\mathcal{H}$ are zero, and the diagonal entries are either zero or one.

Feature selection and extraction have many functions in common. Both can be used to project data onto a lower dimensional space for subsequent visualization, clustering, and other exploratory data analysis. By reducing the number of features considered by a classifier, both techniques can improve classification speed and efficiency. Less intuitively, these techniques can improve classification accuracy by reducing estimation errors associated with finite sample size effects [8]. With feature selection, the cost of classification can be reduced by limiting the number of features which must be measured and stored. Some, but not all, feature extraction methods realize this benefit as well. Many well-known techniques, although they reduce the number of features considered by the classifier, do not actually reduce the number of features which must be measured for classification since each extracted feature is usually a linear combination of all of the original input features. An example of such a transform is canonical linear discriminant analysis [9].

The problem of dimensionality reduction, encompassing both feature selection and feature extraction, has been the subject of study in a diverse spectrum of fields. In neural network pattern classification, feature selection can be effected using node-pruning techniques [10]. After training for a number of epochs, nodes are removed from the network in such a manner that the increase in squared error is minimized. When an input node is pruned, the feature associated with that node is no longer considered by the classifier. Similar methods have been employed in the use of fuzzy systems for pattern recognition through the generation and pruning of fuzzy if–then rules [11], [12]. Some traditional pattern classification techniques, while not specifically addressed to the problem of dimensionality reduction, can provide feature selection capability. Tree classifiers [13], for example, typcally partition the training data based on a single feature at each tree node. If a particular feature is not tested at any node of the decision tree, it is effectively eliminated from classification. Additionally, simplification of the final tree can provide further feature selection [14].

B. Pattern Classification Techniques

Feature extraction can be used in conjunction with numerous methods for pattern classification. The well-known statistical methods can be divided into two general classes. The parametric methods, including the commonly used naive Bayesian classifier [7], assume that the form of the class-conditional density function of the features is known in advance. For example, it is commonly assumed that the features follow a multivariate Gaussian distribution. The training data are used to estimate the parameters of these class-conditional densities (e.g., the mean vector μ and the covariance matrix Σ for the Gaussian distribution), and these estimated densities are then used to classify test patterns. The nonparametric methods, including Parzen window density estimation [15] and the k nearest neighbor (knn) classifier [16], make no assumptions about the distributions of feature values for each class. Rather, the form of the density function is either estimated from the training data, as in the Parzen window method, or ignored altogether in the case of the knn method.

In knn classification, each training pattern is represented in a d-dimensional space according to the value of each of its d features. The test pattern is then represented in the same space, and its k nearest neighbors, for some constant k, are selected. Neighbors are usually calculated according to Euclidean distance, although other metrics (e.g., Mahalanobis distance) are sometimes used. The class of each of these k neighbors is then tallied, and the class with the most "votes" is selected as the classification of the test pattern. The knn classification rule has been selected for use in combination with the GA feature extractor for several reasons. The simplicity of the knn classifier makes it easy to implement. Its nonparametric nature allows classification of a broad range of datasets, including those for which feature values do not follow a multivariate Gaussian distribution. Due to the use of the Euclidean distance metric, the knn decision rule is sensitive to scaling of the feature values—a necessary prerequisite for use with the GA feature extraction technique described here. Classifiers based on the class-conditional distributions of feature values, such as the Bayes classfier, are invariant to scaling of the feature values. Finally, the knn classifier is well explored in the literature, and has been demonstrated to have good classification performance on a wide range of real-world data sets. The asymptotic error rate of the knn decision rule can be shown to be bounded by the Bayes error as $k \rightarrow \infty$ [7, pp. 98–105].

C. Genetic Algorithms in Feature Selection and Extraction

Computational studies of Darwinian evolution and natural selection have led to numerous models for solving optimization [17]–[21]. GA's comprise a subset of these evolution-based optimization problems techniques focusing on the application of selection, mutation, and recombination to a population of competing problem solutions [22], [23]. GA's are parallel, iterative optimizers, and have been successfully applied to a broad spectrum of optimization problems, including many pattern recognition and classification tasks.

The problem of dimensionality reduction is well suited to formulation as an optimization problem. Given a set of d-dimensional input patterns, the task of the GA is to find a transformed set of patterns in an m-dimensional space $(m < d)$ that maximizes a set of optimization criteria. Typically, the transformed patterns are evaluated based upon both their dimensionality, and either class separation or the classification accuracy. Fig. 2 shows the structure of a GA-based feature extractor using classification accuracy as an evaluation criterion. The GA maintains a population of competing feature transformation matrices. To evaluate each matrix in this population, the input patterns are multiplied by the matrix, producing a set of transformed patterns which are then sent to a classifier. The available samples are divided into a training set, used to train the classifier, and a testing set, used to evaluate classification accuracy. The accuracy obtained is then returned to the GA as a measure of the quality of the transformation matrix used to obtain the set of transformed patterns. Using this information, the GA searches for a transformation that minimizes the dimensionality of the transformed patterns while maximizing classification accuracy.

A direct approach to using GA's for feature selection was introduced by Siedlecki and Sklansky [24]. In their work, a GA is used to find an optimal binary vector, where each bit is associated with a feature (Fig. 3). If the ith bit of this vector equals 1, then the ith feature is allowed to participate in classification; if the bit is a 0, then the corresponding feature does not participate. Each resulting subset of features is evaluated according to its classification accuracy on a set of testing data using a nearest neighbor classifier.

This technique was later expanded to allow linear feature extraction, by Punch *et al.* [25] and independently by Kelly and Davis [26]. The single bit associated with each feature is expanded to a real-valued coefficient, allowing independent linear scaling of each feature, while maintaining the ability to remove features from consideration by assigning a weight of zero. Given a set of feature vectors of the form $\boldsymbol{X} = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \cdots, \boldsymbol{x_d}\}$, the GA produces a transformed set of vectors of the form $\boldsymbol{X}' = \{\boldsymbol{w_1 x_1}, \boldsymbol{w_2 x_2} \cdots \boldsymbol{w_d x_d}\}$ where w_i is a weight associated with feature i. Each feature value is first normalized, then scaled by the associated weight prior to training, testing, and classification. This linear scaling of features prior to classification allows a classifier to discriminate more finely along feature axes with larger scale factors. A knn classifier is used to evaluate each set of feature weights. The effects of linear feature weighting on the knn classification rule are visualized in Fig. 4. Patterns plotted in feature space are spread out along feature axes with higher weight values, and compressed along features with lower weight values. The value of k for the knn classifier is fixed and determined empirically prior to feature extraction. This GA/knn feature extraction technique has been applied for predicting favorable water-binding sites in proteins, an important problem in biochemistry related to drug binding and design [27].

The GA feature extraction technique has been expanded to include a binary masking vector along with the feature weight vector on the chromosome [6]. If the mask value for a given feature is zero, the feature is not considered for classification. If the mask value is one, the feature is scaled according to the associated weight value, and is included in the classifier. The inclusion of the mask vector allows the GA to more rapidly sample feature subsets while simultaneously optimizing scale factors for features included in the classifier. An additional improvement specific to the knn classifier was the inclusion of the value of

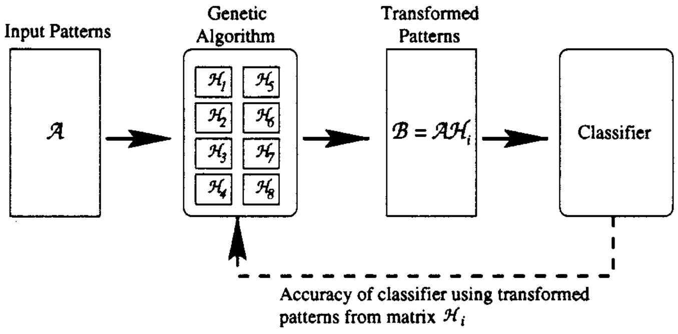

Fig. 2. GA-based feature extractor using an objective function based on classification accuracy. Each transformation matrix from the GA population is used to transform the input patterns, which are then passed to a classifier. The fitness of the matrix is based on the classification accuracy attained on the transformed patterns.

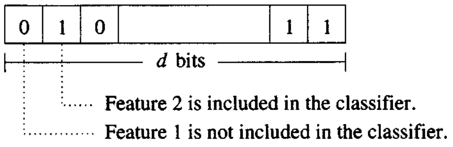
0
1
0
1
1
d bits
Feature 2 is included in the classifier.
Feature 1 is not included in the classifier.

Fig. 3. d-dimensional binary vector, comprising a single member of the GA population for GA-based feature selection.

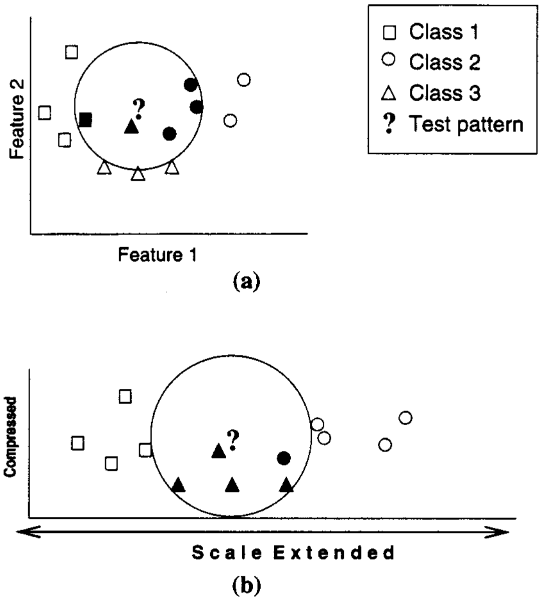

Fig. 4. Effect of scaling feature axes on k $(k = 5)$ nearest neighbor classification. (a) Original data. (b) Scaled data. Extension of the scale of the horizontal axis increases the distance between patterns which differ in feature 1, allowing the knn to discriminate more finely along this dimension. Here, the prediction of the unknown changes from class 2 to class 3 as a result of scaling.

k on the GA chromosome. This modification allows the GA to cooptimize the feature weights and k value for better classification accuracy.

In a recent study of current feature selection techniques, Jain and Zongker [1] evaluated the performance of 15 feature selection algorithms in terms of classification error and run time on a two-class, 20-dimensional, multivariate Gaussian data set. Their findings demonstrated that the sequential floating forward selection algorithm (SFFS) of Pudil *et al.* [28] dominated the other methods for these data, obtaining feature selection results comparable to the optimal branch-and-bound algorithm while requiring less computation time. Further tests of the SFFS technique in combination with a knn classifier were then performed to observe the behavior of the knn classification rule as additional features were provided to the classifier. The results showed that classification accuracy initially improved as additional features were introduced, but eventually reached a maximal value, and began to decline with the introduction of further features. Because of this unimodal behavior, the selection of an optimal number of features is straightforward when using the SFFS method in combination with a knn classifier.

Since the GA/knn algorithm is a hybrid method, combining feature extraction with classifier training and tuning, we present experimental results for the algorithm, both in terms of classification performance and feature selection performance. Classification accuracy is compared with published results for several commonly employed pattern classification algorithms, while feature-selection performance is compared with the SFFS algorithm in combination with a knn classifier.

II. Methods

A. The GA Feature Extractor

There are three major design decisions to consider when implementing a GA to solve a particular problem. A representation for candidate solutions must be chosen and encoded on the GA chromosome, an objective function must be specified to evaluate the quality of each candidate solution, and finally, the GA run parameters must be specified, including which genetic operators to use and their frequencies of operation. For the GA feature extractor, the organization of the chromosome was fairly straightforward. A weight vector consisting of a single real value for each feature was followed by a masking vector consisting of one or more binary mask bits for each feature. When a single mask bit was used, a zero indicated that the feature was to be omitted from consideration during classification, while a one indicated that the feature should be included. At times, it was desirable to use more than one masking bit per feature since each of these bits had a strong impact on the interpretation of the chromosome. The importance of these single bits can cause discontinuities in the objective function, making optimization more difficult for the GA. To mitigate this effect, additional masking bits were used; a feature was then included in classification only if the majority of the masking bits associated with that feature were set to one. Finally, some classifier-specific information was also stored on the chromosome to be optimized along with the feature weights. For example, when a knn classifier was used to evaluate feature extraction results, the value of k was stored on the chromosome and optimized simultaneously with the feature weights. For a Bayes classifier, if the prior probabilities for each class were not known, they could be stored on the chromosome and similarly optimized during feature extraction.

For the experiments presented here, the objective function consisted of the classification performance obtained by a knn classifier, using the value of k provided by the GA. In order to prevent any initial bias due to the natural ranges of values for the original features, input feature values were normalized over the range [1.0, 10.0] as follows:

$$x'_{i,j} = \left(\frac{x_{i,j} - \min\limits_{k=1\cdots n}(x_{k,j})}{\max\limits_{k=1\cdots n}(x_{k,j}) - \min\limits_{k=1\cdots n}(x_{k,j})} * 9 \right) + 1$$

where $x_{i,j}$ is the jth feature of the ith pattern, $x'_{i,j}$ is the corresponding normalized feature, and n is the total number of patterns.

Prior to each GA experiment, the knn classifier was trained with a set of patterns with an equal number of samples of each class. A second set of patterns, disjoint to the training set, was set aside to serve as a tuning set. To evaluate a particular set of weights, the GA scaled the training and tuning set feature values according to the weight set, and classified the tuning set with the weighted knn classifier. The performance of the weight set was determined according to the classification accuracy on the tuning set, the balance in accuracy among classes, the number of correct votes cast during knn classification, and the parsimony oftheweight set (the number of features eliminated from classification by masking). Since the particular GA engine used minimizes the objective function [29], the following error function was used to evaluate each weight set w:

$$\begin{aligned}\text{error}(w) = {} & C_{\text{pred}}\,(\text{no. of incorrect predictions}) \\ & + C_{\text{mask}}\,(\text{no. of unmasked features}) \\ & + C_{\text{vote}}\,(\text{no. of incorrect votes}) \\ & + C_{\text{bal}}\,(\text{difference in accuracy between classes})\end{aligned}$$

where C_{pred}, C_{mask}, C_{vote}, and C_{bal} are the antifitness function coefficients, set as shown in Table I.

The prediction and voting terms both directed the search toward weight sets that achieved better prediction results. The voting term served to smooth the fitness landscape, which tended to have a discrete stepwise character when only the number of incorrect predictions was considered; smoothing the objective function allowed the GA to optimize the weight sets more efficiently. The masking term drove the GA toward more parsimonious solutions—that is, those solutions that classified using the fewest features. Finally, the balance term served to avoid disparities in accuracy among classes. The relative importance of these various objectives was controlled by adjustments to the coefficients associated with each fitness term. The values for the objective function coefficients, as well as the standard GA run parameters (rate of crossover, rate of mutation, population size, etc.), were determined empirically through a set of preliminary experiments. Table I shows the GA run parameters and fitness coefficients chosen after preliminary

TABLE I
GA RUN PARAMETERS; CROSSOVER RATE IS EXPRESSED AS EXPECTED NUMBER OF CROSSOVERS PER INDIVIDUAL PER GENERATION; MUTATION RATE IS EXPRESSED AS EXPECTED NUMBER OF MUTATIONS PER BIT PER GENERATION

Fitness constants		GA parameters	
C_{pred}	20.0	Crossover rate	0.80
C_{mask}	1.0	Mutation rate	0.001
C_{vote}	2.0	Population size	200
C_{bal}	5.0	Generations	200

experimentation was completed. The run length was limited to 200 generations after noting that the population demonstrated convergence by this time in all of the preliminary experiments.

B. Bootstrap Testing

For each GA experiment, the available data were broken into three disjoint sets: training, tuning, and testing. The training and tuning sets were used to train the knn classifier and provide tuning feedback to the GA, as described previously. Once the GA run was completed, the test set was used to perform unbiased testing on the best weight set found by the GA. The holdout testing was done using a variant of the bootstrap test technique [30]–[32]. For each weight set w, 100 bootstrap tests were executed. For each bootstrap test $i \in \{1, 2, \cdots, 100\}$, a random bootstrap set b_i was selected from the holdout set using a uniform random distribution of samples with replacement. The weighted knn classifier was tested on this bootstrap set, and the accuracy for each class c, $\mathrm{acc}(c, w, b_i)$, as well as the total accuracy $\mathrm{acc}_{\mathrm{tot}}(w, b_i)$ were computed as follows:

$$\mathrm{acc}(c, w, b_i) = \frac{|\mathrm{pred}(c, w, b_i) \cap \mathrm{obs}(c, b_i)|}{|\mathrm{obs}(c, b_i)|}$$

$$\mathrm{acc}_{\mathrm{tot}}(w, b_i) = \frac{\sum_{c \in C} |\mathrm{pred}(c, w, b_i) \cap \mathrm{obs}(c, b_i)|}{|b_i|}$$

where $\mathrm{pred}(c, w, b_i)$ is the set of samples in b_i predicted to belong to class c by the weighted knn classifier using the weight set w, $\mathrm{obs}(c, b_i)$ is the set of patterns in b_i observed to belong to class c (that is, the true members of class c), and C is the set of all classes.

Finally, after the 100 bootstrap tests for a given weight set were completed, the performance of the weight set was evaluated according to mean bootstrap accuracy $[\overline{\mathrm{acc}}(w)]$ and variance of bootstrap accuracy $[V_{\mathrm{acc}}(w)]$, computed as follows:

$$\overline{\mathrm{acc}}(w) = \sum_{i=1}^{100} \mathrm{acc}_{\mathrm{tot}}(w, b_i)/100$$

$$V_{\mathrm{acc}}(w) = \frac{100 \sum_{i=1}^{100} \mathrm{acc}_{\mathrm{tot}}(w, b_i)^2 - \left(\sum_{i=1}^{100} \mathrm{acc}_{\mathrm{tot}}(w, b_i) \right)^2}{100 * 99}.$$

III. TRAINING AND TESTING DATA

Several data sets which have been used for classifier benchmarking were employed for testing the classification accuracy of the GA feature extractor in combination with a knn classifier. Classification error for the GA/knn on these data sets was compared to that of several common classification techniques, as reported by Weiss and Kapouleas [33], [34] in a comparative study of various commonly used classification algorithms. Additionally, the effectiveness of the GA feature extractor in producing parsimonious feature sets was evaluated and compared with that of the SFFS feature selection method for each data set.

Further results are given for a particulary difficult data set from biochemistry involoving the classification of water molecules bound to proteins. The effectiveness of the GA/knn for this problem has been explored in some detail [6], [27]. Here, we compare these results with those of the SFFS feature selection method in combination with a knn classifier.

A. Published Data Sets

The GA feature extractor in combination with a knn classifier was tested on two medical data sets from the University of California, Irvine repository of machine learning data sets [35].

The first medical data set consisted of the results of 21 clinical tests for a set of patients tested for thyroid dysfunction [36]. The training set consisted of 3772 cases from the year 1985, while the test set consisted of 3428 cases from 1986. The goal of the classifier was to determine, based upon the test results provided, whether or not a patient should be diagnosed as hypothyroid. The number of samples in the two classes was highly unbalanced in this data. The training set contained 3487 negative (nonhypothyroid) cases and 284 positive cases. Likewise, the testing set consisted of 3177 negative samples and 250 positive.

The second medical data set was from a published study [37] on the assessment of eight laboratory tests to confirm the diagnosis of acute appendicitis. 85 of the 106 patients had confirmed appendicitis.

B. Biochemistry Data

The most extensive application of our GA feature extraction technique has been made in the field of biochemistry. Understanding protein–water interactions is important for *de novo* drug design, database screening for drug leads, ligand docking, and understanding protein structure and function. An important part of this question that has not been fully explored is the identification of regions on the surface of a protein that are favorable for the binding of water molecules, and understanding the features that produce these favorable sites. Application of the GA feature extraction technique for this problem has yielded progress in two areas: a classifier has been developed which can identify likely water-binding sites on the protein surface [27], and a set of features sufficient to make this classification has been identified [38].

The data set for the classification of water molecules consists of eight physical and chemical features describing the protein environments of water molecules from 30 pairs of crystallographic protein structures taken from the Brookhaven Protein Data Bank [39], [40]. The eight features used to characterize each water molecule are described in Table II. For each protein, a structure of the protein bound to another molecule and a structure of the free (unbound) protein were obtained and

TABLE II
PHYSICAL AND CHEMICAL FEATURES USED TO REPRESENT PROTEIN-BOUND WATER MOLECULES

Tag	Name	Description
ADN	Atomic Density	The number of protein atoms within 3.5Å of the water molecule.
AHP	Atomic Hydrophilicity	A measure of the propensity of neighboring atom types to bind water molecules in other protein environments. For more details see [27, 41].
BVAL	B-value	The crystallographic temperature factor of the water molecule. A measure of thermal mobility.
HBDP	H-bonds to protein	The number of hydrogen bonds between the water molecule and neighboring protein atoms.
HBDW	H-bonds to water	The number of hydrogen bonds between the water molecule and other water molecules.
MOB	Mobility	A normalized measure of thermal mobility [42].
ABVAL	Average B-value of protein atom neighbors	The average B-value of all protein atoms within 3.5Å of the water molecule.
NBVAL	Net B-value of protein atom neighbors	The sum of the B-values of all protein atoms within 3.5Å.

superimposed. The water molecules in the free structure were divided into two classes: those that were conserved in the bound structure (conserved water molecules), and those that were found only in the free structure (nonconserved water molecules). The goal of the GA/knn classifier was to find a minimal set of weighted features that provided maximal classification accuracy for distinguishing conserved from nonconserved water molecules. The data set consisted of 5542 water molecules, 3405 of which were conserved and 2137 of which were nonconserved.

IV. RESULTS AND DISCUSSION

A. Tests on Medical Data

For the thyroid data, the sequential floating forward selection method achieved good classification results. The best accuracy obtained by the knn/SFFS algorithm during feature selection was 97.99%, using 6 of the 21 available features. 100 bootstrap tests on this feature set yielded a mean bootstrap accuracy of 98.06%, with a standard deviation of 0.6032%. This accuracy is similar to those obtained by the various methods reported by Weiss and Kapouleas [33], [34] (Table III). The GA feature extractor combined with a knn classifier obtained a similar accuracy, 98.48%, using only 3 of the available 21 features, and a k value of 87. 100 bootstrap tests for this set of feature weights yielded a mean bootstrap accuracy of 98.40%, with a standard deviation of 0.6256%. Features and feature weights obtained by the SFFS algorithm and the GA feature extractor are shown in Table IV.

For the appendicitis data, floating forward selection was again applied with knn classifiers using odd values of k from 1 to 99. The best result was obtained for $k = 7$. The best predictive accuracy during selection was 88.46% using three of the seven available features. Bootstrap testing for 100 trials using the best feature set found by SFFS yielded a mean predictive accuracy of 91.44% with a standard deviation of 3.94%. The results reported by Weiss for this data, as well as the results for the GA feature extractor, are shown in Table V. The GA feature extractor achieved a slightly higher accuracy than SFFS during extraction: 90.38% using two of seven weighted features and $k = 7$. In bootstrap testing, however, the mean bootstrap accuracy over 100 trials proved to be similar to that of SFFS—90.60% with a standard deviation of 4.21% The features selected by each algorithm are shown in Table VI.

TABLE III
RESULTS OF VARIOUS CLASSIFIERS ON HYPOTHYROID DATA, AS REPORTED BY WEISS, IN COMPARISON WITH THOSE OF THE GA FEATURE EXTRACTOR

Method	Accuracy (training)	Accuracy (testing)
GA Feature Extractor	98.5%	98.4%
Linear Discriminant	93.8%	93.8%
Quadratic Discriminant	89.7%	88.4%
Nearest Neighbor	100%	95.3%
Bayes (independent)	97.1%	96.1%
Bayes (2nd order)	97.7%	92.4%
Neural Net (Back prop)	99.5%	98.5%
Predictive Value Max.	99.8%	99.3%
CART Tree	99.8%	99.4%

TABLE IV
FEATURE SELECTION AND WEIGHTS FOR SFFS AND GA FEATURE EXTRACTION ON THYROID DATA

Method	AGE	MALE	OTHY	QTHY	OMED	SICK
GA	0.00	0.00	5.95	0.00	0.00	0.00
SFFS	0	0	1	0	0	0
	PREG	SURG	I131	QPO	QPER	LITH
GA	0.00	78.67	0.00	0.00	0.00	0.00
SFFS	0	1	1	0	0	0
	TUM	GOIT	HPIT	PSY	TSH	T3
GA	0.00	0.00	0.00	0.00	90.04	0.00
SFFS	0	0	0	1	1	0
	TT4	T4U	FTI			
GA	0.00	0.00	0.00			
SFFS	0	1	0			

B. Classification of Protein-Bound Water Molecules

For the identification of favorable water binding sites on protein surfaces, the knn required significantly more evaluation time than for the medical data, due to the large sizes of the water-binding data sets employed. A typical GA experiment for these data required 60–70 h wall-clock time on a Sparc20-612 workstation. As a result, fewer experiments were conducted for water binding site prediction than

TABLE V
RESULTS REPORTED BY WEISS, AND THOSE OF THE GA FEATURE EXTRACTOR, ON APPENDICITIS DATA

Method	Accuracy (training)	Accuracy (testing)
GA Feature Extractor	90.4%	90.6%
Linear Discriminant	88.7%	86.8%
Quadratic Discriminant	79.3%	73.6%
Nearest Neighbor	100%	82.1%
Bayes (independent)	88.7%	83.0%
Bayes (2nd order)	95.3%	81.1%
Neural Net (Back prop)	90.0%	85.8%
Predictive Value Max.	91.5%	89.6%
CART Tree	90.0%	84.9%

TABLE VI
FEATURE SELECTION AND WEIGHTS FOR SFFS AND GA FEATURE EXTRACTION ON APPENDICITIS DATA

GA	43.68	0.00	0.00	21.39	0.00	0.00	0.00
SFFS	0	0	1	1	1	0	0

for the two medical data sets. After the initial experiments to determine run parameters were completed, 21 GA runs were conducted with random initial populations. The weight sets in the final population of each run were then evaluated in terms of prediction accuracy and the dimensionality of the resulting feature set. The mean bootstrap accuracy for the best weight set from each run was evaluated over 100 bootstrap tests. The best weight set found by the GA achieved a mean bootstrap accuracy of 64.20%, with a standard deviation of 1.42% using four of the available eight features. The second-best performing weight set achieved a mean bootstrap accuracy of 63.32% using only two of the eight features [38]. The k value for the knn rule, which was also optimized by the GA, ranged from 17 to 77 for the top five weight sets. The SFFS feature selection algorithm was tested in conjunction with a knn classifier for various values of k ranging from 1 to 65. The best classification accuracy was attained at $k = 65$, and utilized three of the eight features. The mean bootstrap accuracy over 100 trials for this feature subset was 63.21%, with a standard deviation of 2.19%. Table VII compares the feature subsets found by GA/knn and SFFS/knn. The distributions of feature values for the two classes overlap significantly in pairwise two-dimensional marginal feature plots for these data, and linear discriminant analysis yielded a classification accuracy of only 49.7%, approximately equivalent to random class assignment.

C. Discussion

The integrated feature extraction and classification approach described has proved effective on these three disparate data sets. For the thyroid data, the GA–knn was more effective than all but two of the approaches, but required only three of the features to make the classification. The GA–knn obtained a classification accuracy within 1% of the best technique reported. For the appendicitis data, the GA–knn showed the best classification performance among of all the techniques, and required only two of the seven features. Thus, the approach is both accurate, in comparison to other techniques, and parsimonious in the number of features required to achieve that accuracy.

TABLE VII
FEATURE WEIGHTS, k VALUE, AND MEAN BOOTSTRAP ACCURACY (ACC) FOR THE BEST TWO WEIGHT SETS AND FEATURE SUBSETS FOUND USING THE GA FEATURE EXTRACTOR AND SFFS FEATURE SELECTION IN COMBINATION WITH A k NEAREST NEIGHBOR CLASSIFIER

Method	k	Acc	ADN	AHP	BVAL	HBDP	HBDW	MOB	ABVAL	NBVAL
GA/knn	65	64.20	0.00	0.00	0.413	0.135	0.137	0.315	0.00	0.00
GA/knn	29	63.62	0.00	0.00	0.667	0.00	0.00	0.333	0.00	0.00
SFFS/knn	65	63.21	0.00	1.00	1.00	0.00	0.00	1.00	0.00	0.00

The problem of predicting the binding of water molecules to proteins is difficult because of the highly overlapping distributions of the features, as indicated by the linear discriminant results. However, the unbiased accuracy was better than 63% for this difficult problem using the GA–knn approach, even when the number of features considered by the classifier was reduced from seven to as few as two. The GA approach exceeded the SFFS feature selection method slightly, both in terms of classification accuracy and feature subset parsimony.

A key advantage of the GA feature extraction technique is that it combines various benefits of feature selection and extraction into a single method. As with feature extraction, the original features are transformed to produce a new set of features. Although linear coefficients were used in the experiments presented here, the relationship between the input and output features need not be linear. This transformation of features can be used with feedback from the classifier to produce better classification accuracy. Additionally, since each output feature of the GA feature extractor is based only on a single input feature, the relationships between the original features and the transformed features remain explicit, and easy to identify and analyze. Analysis of the features that prove sufficient for classification can lead to a deeper understanding of the data, allowing feature extraction to be used in exploratory data analysis and data mining. This capability was of primary importance in classifying the water conservation data, where the goal was more biochemical—to obtain a better understanding of the features that lead to conserved water binding—than statistical in nature.

While the knn classifier performed well in combination with the GA feature extractor, other classification techniques may alsoprove effective in providing feedback to the GA. If, for example, a Bayesian classification rule were used in place of the knn rule, the GA might be used to simultaneously optimize the feature weights and the *a priori* class distributions. A more general form of feature extraction may be achieved by directly representing the entire transformation matrix $\mathcal{H}$ on the chromosome. This would allow the GA to explore the space of all possible linear transformations of the original data. It remains to be investigated whether the ability of the GA to perform arbitrary transformations would lead to better classification accuracy or would be offset by the increase in the size of the search space to be explored.

Acknowledgment

The authors thank the members of the Genetic Algorithms Research and Applications Group and the members of the Protein Structural Analysis and Design group at Michigan State University for their ideas and critical feedback. They also acknowledge Dr. M. Pei for his contributions to the development of the GA feature extraction technique.

References

[1] A. K. Jain and D. Zongker, "Feature selection: Evaluation, application, and small sample performance," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 153–158, Feb. 1997.

[2] F. J. Ferri, P. Pudil, M. Hatef, and J. Kittler, "Comparative study of techniques for large-scale feature selection," in *Pattern Recognition in Practice IV, Multiple Paradigms, Comparative Studies and Hybrid Syst.*, E. S. Gelsema and L. S. Kanal, Eds. Amsterdam: Elsevier, 1994, pp. 403–413.

[3] C. Lee and D. A. Landgrebe, "Decision boundary feature extraction for neural networks," *IEEE Trans. Neural Networks*, vol. 8, pp. 75–83, Jan. 1997.

[4] A. Biem, S. Katagiri, and B.-H. Juang, "Pattern recognition using discriminative feature extraction," *IEEE Trans. Signal Processing*, vol. 45, pp. 500–504, Feb. 1997.

[5] J. Mao and A. K. Jain, "Artificial neural networks for feature extraction and multivariate data projection," *IEEE Trans. Neural Networks*, vol. 6, pp. 296–317, Mar. 1995.

[6] M. L. Raymer, W. F. Punch, E. D. Goodman, P. C. Sanschagrin, and L. A. Kuhn, "Simultaneous feature scaling and selection using a genetic algorithm," in *Proc. 7th Int. Conf. Genetic Algorithms (ICGA)*, Th. Bäck, Ed. San Francisco, CA: Morgan Kaufmann, 1997, pp. 561–567.

[7] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[8] A. K. Jain and B. Chandrasekaran, "Dimensionality and sample size considerations in pattern recognition in practice," in *Handbook of Statistics*, P. R. Krishnaiah and L. N. Kanal, Eds. Amsterdam: North-Holland, 1982, vol. 2, pp. 835–855.

[9] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. Eugenics*, vol. 7, pp. 179–188, 1936.

[10] J. Mao, K. Mohiuddin, and A. K. Jain, "Parsimonious network design and feature selection through node pruning," in *Proc. Int. Conf. Pattern Recognition*, Jerusalem, Israel, Oct. 1994, pp. 622–624.

[11] H. Ishibuchi, K. Nozaki, N. Yamamoto, and H. Tanaka, "Selecting fuzzy if-then rules for classification problems using genetic algorithms," *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 260–270, Aug. 1995.

[12] K. Nozaki, H. Ishibuchi, and H. Tanaka, "Adaptive fuzzy rule-based classification systems," *IEEE Trans. Fuzzy Syst.*, vol. 4, pp. 238–250, Aug. 1996.

[13] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, pp. 81–106, 1986.

[14] ——, "Simplifying decision trees," *Int. J. Man-Machine Studies*, pp. 221–234, 1987.

[15] E. Parzen, "On the estimation of a probability density function and the mode," *Ann. Math Statist.*, vol. 33, pp. 1065–1076, 1962.

[16] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 21–27, 1967.

[17] A. S. Fraser, "Simulation of genetic systems by automatic digital computers—I: Introduction," *Austrailian J. Biol. Sci.*, vol. 10, pp. 484–491, 1957.

[18] J. L. Crosby, "Computers in the study of evolution," *Sci. Prog. Oxf.*, vol. 55, pp. 279–292, 1967.

[19] H. J. Bremermann, M. Rogson, and S. Salaff, "Global properties of evolution processes," in *Natural Automata and Useful Simulations*, H. H. Pattee, E. A. Edlsack, L. Fein, and A. B. Callahan, Eds. Washington, DC: Spartan, 1966, pp. 3–41.

[20] J. Reed, R. Toombs, and N. A. Barricelli, "Simulation of biological evolution and machine learning: I. Selection of self-reproducing numeric patterns by data processing machines, effects of hereditary control, mutation type and crossing," *J. Theoret. Biol.*, vol. 17, pp. 319–342, 1967.

[21] D. B. Fogel, Ed., *Evolutionary Computation: The Fossil Record*. New York: IEEE Press, 1998.

[22] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: Univ. Michigan Press, 1975.

[23] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[24] W. Siedlecki and J. Sklansky, "A note on genetic algorithms for large-scale feature selection," *Pattern Recognit. Lett.*, vol. 10, pp. 335–347, 1989.

[25] W. F. Punch, E. D. Goodman, M. Pei, L. Chia-Shun, P. Hovland, and R. Enbody, "Further research on feature selection and classification using genetic algorithms," in *Proc. Int. Conf. Genetic Algorithms*, 1993, pp. 557–564.

[26] J. D. Kelly and L. Davis, "Hybridizing the genetic algorithm and the k nearest neighbors classification algorithm," in *Proc. 4th Int. Conf. Genetic Algorithms Appl.*, 1991, pp. 377–383.

[27] M. L. Raymer, P. C. Sanschagrin, W. F. Punch, S. Venkataraman, E. D. Goodman, and L. A. Kuhn, "Predicting conserved water-mediated and polar ligand interactions in proteins using a k-nearest-neighbors genetic algorithm," *J. Mol. Biol.*, vol. 265, pp. 445–464, 1997.

[28] P. Pudil, J. Novovicova, and J. Kittler, "Floating search methods in feature selection," *Pattern Recognit. Lett.*, vol. 15, pp. 1119–1125, Nov. 1994.

[29] N. N. Schraudolph and J. J. Grefenstette, "A user's guide to GAucsd," Dept. Comput. Sci., Univ. California, San Diego, Tech. Rep. CS92-249, 1992.

[30] A. K. Jain, R. C. Dubes, and C. C. Chen, "Bootstrap techniques for error estimation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 9, pp. 628–633, Sept. 1987.

[31] B. Efron, "Bootstrap methods: Another look at the jackknife," *Ann. Statist.*, vol. 7, pp. 1–26, 1979.

[32] ——, "The jackknife, the bootstrap, and other resampling plans," *CBMS–NSF Regional Conf. Ser. Appl. Math., SIAM'91*, no. 38, 1982.

[33] S. M. Weiss and I. Kapouleas, "An empirical comparison of pattern recognition, neural nets, and machine learning classification methods," in *Proc. 11th Int. Joint Conf. Artif. Intell.*, N. S. Sridharan, Ed. Detroit, MI: Morgan Kaufmann, 1989, pp. 781–787.

[34] ——, *An Empirical Comparison of Pattern Recognition, Neural Nets, and Machine Learning Classification Methods*: Morgan Kaufmann, 1990.

[35] C. J. Merz and P. M. Murphy. (1998) UCI repository of machine learning databases. [Online]. Available: http://www.ics.uci.edu/~mlearn/ML-Repository.html

[36] J. R. Quinlan, P. J. Compton, K. A. Horn, and L. Lazurus, "Inductive knowledge acquisition: A case study," in *Proc. 2nd Australian Conf. Appl. Expert Syst.*, Sydney, Australia, 1986.

[37] A. Marchand, F. Van Lente, and R. Galen, "The assessment of laboratory tests in the diagnosis of acute appendicitis," *Amer. J. Clin. Pathol.*, vol. 80, no. 3, pp. 369–374, 1983.

[38] M. L. Raymer, D. Holstius, and L. A. Kuhn, "Identifying the determinants of favorable solvation sites," *Protein Eng.*, submitted for publication.

[39] E. E. Abola, F. C. Bernstein, S. H. Bryant, T. F. Koetzle, and J. Weng, *Protein Data Bank*, Bonn/Cambridge/Chester: Data Commission of the International Union of Crystallography, 1987, pp. 107–132.

[40] F. C. Bernstein, T. F. Koetzle, G. J. B. Williams, E. F. Meyer Jr., M. D. Brice, J. R. Rodgers, O. Kennard, T. Shimanouchi, and M. Tasumi, "The protein data bank: A computer-based archival file for macromolecular structures," *J. Mol. Biol.*, vol. 112, pp. 535–542, 1977.

[41] L. A. Kuhn, C. A. Swanson, M. E. Pique, J. A. Tainer, and E. D. Getzoff, "Atomic and residue hydrophilicity in the context of folded protein structures," *Proteins: Str. Funct. Genet.*, vol. 23, pp. 536–547, 1995.

[42] L. Craig, P. C. Sanschagrin, A. Rozek, S. Lackie, L. A. Kuhn, and J. K. Scott, "The role of structure in antibody cross-reactivity between peptides and folded proteins," *J. Mol. Biol.*, vol. 281, no. 1, pp. 183–201, 1998.