# FEATURE SELECTION FOR CLASSIFICATION BY USING A GA-BASED NEURAL NETWORK APPROACH

Te-Sheng Li*
*Department of Industrial Engineering and Management*
*Ming Hsin University of Science and Technology*
*1 Hsin-HsingRoad, Hsin-Fong, Hsinchu 304, R.O.C.*

## ABSTRACT

This paper proposes a method of genetic algorithm (GA) based neural network for feature selection that retains sufficient information for classification purposes. This method combines a genetic algorithm with an artificial neural network classifier, such as back-propagation (BP) neural classifier, radial basis function (RBF) classifier or learning vector quantization (LVQ) classifier. In this article, the genetic algorithm optimizes a feature vector by removing both irrelevant and redundant features and finds optimal ones. First, the procedure of the proposed algorithm is described and then the performance of this method is evaluated using two data sets. The results are compared with the genetic algorithm in combination with the *k*-nearest neighbor (KNN) classification rule. Our results suggest that GA based neural classifiers are robust and effective in finding optimal subsets of features from large data sets.

***Keywords***: genetic algorithm, back-propagation, radial basis function, learning vector quantization, *k*-nearest neighbor

## 1. INTRODUCTION

The classification problem involves multi-dimensional information systems used to determine which item belongs to what class out of a set of possible classes. A number of variables stored in the multi-dimensional data sets are sometimes called features. Unfortunately, numerous of potential features have considerable impact on the efficiency of the classifiers, such as the *k*-nearest neighbor, C4.5 (Quinlan, 1993) and back-propagation classifier. Most of these features are either partially or completely irrelevant or redundant to the classified target. It can not be known in advance which features will provide sufficient information to discriminate among the classes. It is also infeasible to include all possible features in the processes of classifying the patterns and objects. Feature selection is one of the major tasks in classification problems. The main purpose of feature selection is to select a number of features used in the classification and at the same time to maintain acceptable classification accuracy.

Various algorithms have been used for feature selection in the past decades. Narendra and Fukunaga (1977) introduce the branch and bound algorithm to eliminate the cost associated with searching through all of the feature subsets. Later, Foroutan and Sklansky (1987) introduce the concept of approximate monotonicity and use the branch and bound method to select features for piecewise linear classifiers. Siedlecki and Sklansky (1988) integrate the genetic algorithm (GA) (Holland, 1975; Goldberg, 1989) with the *k*-nearest neighbor (KNN) classifier to solve the feature selection problem. The GA plays the role of selector to select a subset of features that can best describe the classification performance evaluated by using the KNN classifier. In this study, we employed the idea from Siedlecki and Sklansky (1989) and used neural network classifier to compare the feature selection classification performance.

The GA is a powerful feature selection tool, especially when the dimensions of the original feature set are large (Siedlecki et al., 1989). Reducing the dimensions of the feature space not only reduces the computational complexity, but also increases estimated performance of the classifiers. Kudo et al. (2000) present three versions of feature selection. These three problem types cause in specific objectives and different types of optimization. The first problem version involves determining a subset that yields the lowest classifier error rate. This version leads to unconstrained combinatorial optimization in

* Corresponding author:Jeff@must.edu.tw

which the error rate is the search criterion. The second version involves seeking the smallest feature subset that has an error rate below a given threshold. This version leads to a constrained combinatorial optimization task, in which the error rate serves as a constraint and the number of features is the search criterion. The third involves finding a compromise objective between version one and version two by minimizing the penalty function. In this study, we will focus the proposed method on the second problem version.

The rest of this paper is organized as follows. Section 2 reviews literature in feature selection. Section 3 proposes a GA-based neural feature selection method and its implementation procedure. Section 4 illustrates two numerical examples, summarized the computation results followed by discussions and comparisons of the currently used algorithms in section 5. Conclusions and the direction for future research are given in Section 6.

## 2. LITERATURE REVIEW

A varied number of algorithms have been proposed for feature selection and some comparative studies have been carried out. Among these algorithms, linear feature selection methods have been developed. The linear methods include projection pursuit (Jimenez et al., 1995), quadratic discriminant analysis (Brunzell et al., 2000), principal component analysis (PCA), and linear discriminant analysis (LDA). These well-known techniques reduce the observed variables into a smaller number of "projections", or "dimensions" that results in decreasing the number of features to be considered by the classifiers. Rather than directly eliminating irrelevant or redundant variables from the original feature space, they merely transform the original variables through linear combination into a new subset of variables. Thus, the linear methods provide a new way of understanding the data, but they are not able to reduce the number of original features.

The well-known search methods in the literature applied to feature selection include floating search methods, feature filter model and wrapper model. Floating search methods (Pudil et al., 1994) establish the best feature set by adding and/or removing a small number of measurements from the current set at a time. The filter model (John et al., 1994) filters the features before applying an induction algorithm. The wrapper model uses the induction algorithm to evaluate the features. The possible search strategies in the feature space include backward elimination and forward selection. The wrapper model performance is determined by the predicted accuracy of the induction algorithm and estimated by using n-fold cross-validation.

As described in the previous section, branch and bound is a common approach in feature selection but it has two major problems. The first is that the constraint criterion must obey the monotonicity property, otherwise the branch and bound procedure cannot access and explore all of the disconnected feasible parts of the feature space. The second problem is that branch and bound procedure conducts an exhaustive search in the feasible region. The size of this region grows at the same rate as the size of the entire search space, i.e., as $2^d$, where $d$ is the original number of features. When using the branch and bound to search in a feature space with more than 30 dimensions, it becomes impractical, causing excessive computational complexity.

Dash et al. (1997) provide a detailed survey and overview of the existing methods for feature selection. They suggest a feature selection process that includes four parts, namely, feature generation, feature evaluation, stopping criteria and testing. In addition to the classic evaluation measures (accuracy, information, distance, and dependence) used for removing irrelevant features, they provide consistency measures (inconsistency rate) to determine a minimum set of relevant features.

The decision tree method shows that feature selection can improve case-based learning (Cardie, et al., 1993). When C4.5 is conducted in the training set, the features that appear in the pruned decision tree are selected. That is, the features appearing in the paths to any leaf node in the pruned tree become the selected subset. Although this algorithm is expected to have an inherent feature selection capability, there is no guarantee that all of the irrelevant and redundant features are totally removed from the decision tree.

Integrating the GA with other classifiers has been used to produce several feature selection algorithms. For example, the KNN-GA feature selection is one of the data reduction techniques introduced by Siedlecki and Sklansky (1989). The KNN classification was selected for use in combination with the GA feature extractor because of its simplicity in implementation. In their study, a GA is used to find an optimal binary vector, where each bit is associated with a variable. If the $i$th bit of this vector equals 1, the $i$th variable is allowed to participate in classification. If the bit is 0, the corresponding variable does not participate. Each remaining subset of variables is evaluated according to its classification accuracy on a set of testing data using the nearest neighbor classifier. With this algorithm, the GA searches for an optimal subset of input vectors that minimizes the dimensionality of the input variables while maximizing the classification accuracy. Raymer et al. (2000) implemented the similar GA-KNN classifier for identification of favorable water-binding sites on protein surface, an important technique in biochemistry and

drug design.

Tsai (2000) develops an ad hoc iterative variable reduction algorithm for a probabilistic neural network (PNN) to identify noise and redundant variables. This iterative approach utilized a weighted PNN with one smoothing factor for each variable in the variable reduction stage. Once a subset of variables was selected, a basic PNN was developed based on the chosen variables for future application.

Meyer-Base and Watzel (1998) propose four layers of radial basis neural networks for selecting relevant features in pattern recognition problems. They defined the relevance $\rho_n$ for each feature $x_n$. If $\rho_n$ falls below the threshold, feature $x_n$ is discarded. Backer et al. (1998) compared four non-linear dimensionality reduction techniques for unsupervised feature extraction. They are multidimensional scaling, Sammon's mapping, self-organizing maps and auto-associative feedforward networks. Evaluating the reduced variable set to perform classification tasks makes a comparison with respect to feature extraction. The experiments involved an artificial data set and color texture data sets.

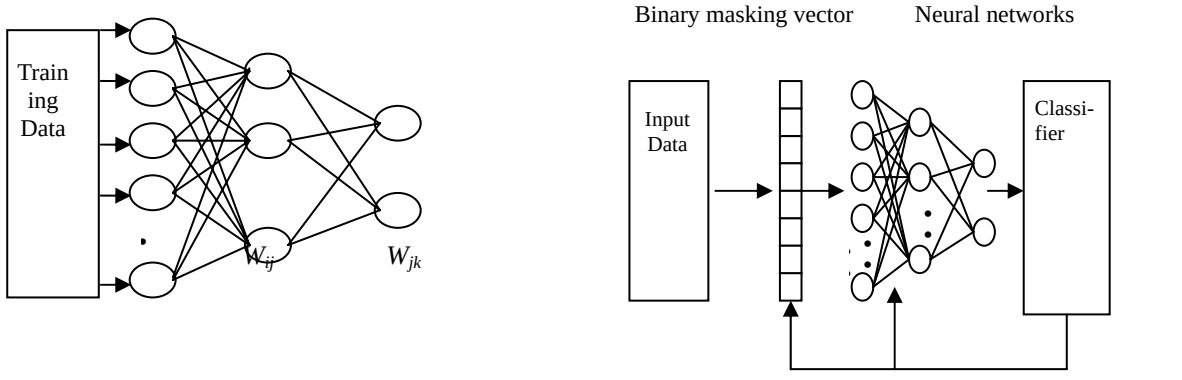
In other methods, such as the neural network classifier, feature selection can be carried out using an input node-pruning algorithm (Mao et al., 1994). After training for a number of epochs, the input nodes are removed from the network. After an input node is pruned, the classifier no longer considers the feature associated with that node. Some researchers employed the fuzzy system to extract the if-then rules from the original feature space for the classification and pattern recognition problem.

## 3. GA-BASED FEATURE SELECTION

The proposed GA-based feature selection approach is similar to the KNN-GA approach developed by Siedlecki and Sklansky (1989) in the literature. In the KNN-GA approach, given a set of feature vectors of the form $X=\{x_1, x_2, \ldots, x_n\}$, the GA produces a transformed set of vectors of the form $X'=\{w_1x_1, w_2x_2, \ldots, w_nx_n\}$ where $w_i$ is a weight associated with feature $i$. A KNN classifier is used to evaluate each set of feature weights. This algorithm introduces a binary masking vector along with feature weight vector on the chromosome. Using the GA optimization technique, this algorithm can efficiently search for the optimal solution, the maximal classification accuracy or minimal classification error rate.

*Phase I*: Training neural networks using all of the features

*Phase II:* Optimizing the GA fitness function by encoding the binary masking vector along with weights between neural layers.




Classifier accuracy is obtained by adjusting the masking vector and weights between the neural network layers

Figure 1. Schema of the proposed GA-based feature selection approach

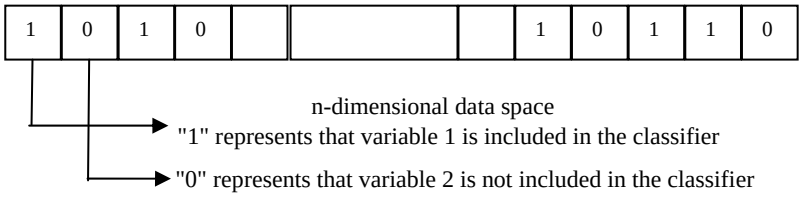



Figure 2. n-dimensional binary mask vector, comprising a set of the GA chromosome for GA-based feature selection method.

The proposed GA-based feature selection method has a procedure similar to that in the KNN-GA algorithm. This procedure can be divided into two main phases (see Fig. 1). The first phase involves training the neural networks via the training data along with the original features. After reaching the acceptable criteria (e.g. the highest classification accuracy or the lowest RMSE), the neural network structure and all of the parameters used in the training are employed in the second phase of the procedure.

The second phase is GA optimization. There are three major design decisions to consider when implementing a GA to solve a particular problem. A representation for candidate solutions must be encoded on the GA chromosome, an objective function must be specified to evaluate the quality of each solution, and finally, the GA run parameters must be specified. From Figure 1, a binary mask vector along with the weights obtained from the training results in phase I, are encoded on the GA chromosome. 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 included in the classifier (see Figure 2). The initial solution is calculated by introducing all of the original variables into the model. The predicted output is then calculated using the input data set and activation functions between layers. After the predicted output is calculated, the next step is to calculate the fitness function and compare the output with desired target. In this work, the problem is regarded as the second problem version, i.e., seeking the smallest or the least costly subset of features for which the classifier's performance is not below a specific threshold. To make the constrained optimization problem suitable for a genetic search, a fitness function is employed:

$$f = (1 - e) \cdot (m - F_s / F_t) \quad (1)$$

In this formula, $F_t$ is the total number of features, $F_s$ is the number of subset features, $e$ is the classification error rate when using the feature subset $F_s$, and $m$ is any number greater than one. The parameter $m$ is used to tune the procedure to compromise between minimizing the number of features in the subset and maximizing the classification rate.

According to the classification accuracy, the GA will change the binary mask vector and weights between the neural network layers and iterate the procedure back and forth until an optimal solution is reached based on the stopping criteria. The GA run parameters are determined empirically prior to the feature selection procedure. When reaching the optimal solution, the bit in the binary mask vector indicates whether or not the given variable is included in the model. In a GA-based feature selection method, the GA procedure will simultaneously maximize the classification accuracy and minimize the number of binary bits in the mask vector. The proposed GA-based feature selection method is summarized as follows:

*Phase I*: Training the neural networks
Step 1: Collect a set of observed data.
Step 2: Divide the data into training and test data sets.
Step 3: Set the training parameters (such as learning rate, momentum, etc).
Step 4: Train the different neural network structures.
Step 5: Choose a trained network with the highest accuracy rate and obtained the weights between the layers.

*Phase II*: GA optimization process
Step 1: Initialize the GA chromosome (assigning 1 to each binary node in mask vector along with the weights obtained from step 5 on phase I)
Step 2: Set the GA operating conditions (e.g. generation size, population size, crossover rate and mutation rate).
Step 3: Use the input data to obtain the initial solution.
Step 4: Repeat steps 5-9 until a stopping condition is reached.
Step 5: Calculate the output value by entering the input data sets and mask vector into the trained network (obtained from step 5, phase I).
Step 6: Transfer the output value to a class label.
Step 7: Calculate the classification accuracy rate by comparing the target with the class label.
Step 8: Select, crossover and mutate the chromosome according to the fitness function (equation 1).
Step 9: Change the binary mask vector and weights between the neural network layers.
Step 10: Obtain an optimal subset of input variables based on the binary mask vector (denoted by "1") and weights between the layers

*Phase III*: Testing process
Step 1: Find the test data.
Step 2: Apply the data to the trained GA-based neural classifier from step 10 in phase II.
Step 3: Obtain the classification results.

Different neural network models can be employed in the proposed GA-based feature selection approach, such as BP, RBF and LVQ. The BP neural networks learn using error back-propagation between the layers along with the delta rule. Sigmoid functions are usually employed as an activation function and minimization is achieved via the gradient descent approach, by which the weights are adjusted in decreasing error. Training in an RBF network, however, involves finding the centers, widths and weights of the connections between hidden neurons and output using the *k*-means clustering algorithm. The *k*-means

algorithm is based on minimization of a performance index that is defined as the sum of the squared distance from each point in a cluster domain to the cluster center. Once learning in the hidden layer is completed, a supervised learning algorithm is applied to train the weights between the hidden and output nodes. The output layer in the RBF network is trained using the Least Mean Squares (LMS) algorithm. Learning in an LVQ network involves finding the reference vectors. "Winner-take-all" learning strategy involves in each learning iteration, the network is only told whether its output is correct or incorrect and only the reference vector of the neuron that wins the competition by being closest to the input vector is activated and allowed to modify its connection weight. The weights of the connections between the input and hidden layers constitute the components of the reference vectors. Their values are modified during learning.

# 4. NUMERICAL ILLUSTRIONS

## 4.1 Example 1: Medical Data

General medical examination data (thirty attributes including sex, age, Neutrophil, Lymphocyte, Monocyte, Basophil, RBC, Hemoglobin,…, etc.) were collected from a hospital in Taipei, Taiwan. These attributes were characterized by using multi-dimensional information to show current health status of patients, but such a large amount of information makes disease diagnosis difficult. Therefore, until now, the medical examination data cannot successfully reveal the symptoms of liver malfunction. Typically, disease diagnosis can be considered as a classification task. Our proposed approaches were employed in this field to clearly classify whether an individual has liver disease.

In this study, a year of medical examination data were collected from 952 individuals, including patients with normally functioning and malfunctioning livers. After data collection, we chose 89 individuals labeled as normal and 79, as liver malfunction based on the medical histories. The Cumulative general examination data of these 168 people were used as the inputs in this study.

For the GA operation presented here, the objective function consisted of the classification performance obtained from three neural classifiers according to the test data set. The objective function can be described by maximizing the classification accuracy or by minimizing the error function, which includes the number of incorrect predictions and number of unmasked variables. By combining a binary mask vector with optimal neural structure weights, the GA was driven to make few but more significant solutions.

The classification results from using the original variables for three neural network classifiers are illustrated in Table 1 The performance of GA-based feature selection approaches is shown in Table 2 The best accuracy obtained by the BP-GA, RBF-GA and LVQ-GA are 85.07%, 82.09% and 92.54% on test data, respectively. The percentage shows that 10 of the 67 test data in BP-GA, 12 of the 67 test data RBF-GA and 5 of the 67 test data in LVQ-GA were incorrect predictions. The classification performances of the GA-based neural approaches were slightly lower than that from the neural classifiers by using all original input variables of test data. Following the feature selection, however, the dimensionalities of the BP-GA, RBF-GA and LVQ-GA models were reduced to 20, 18 and 22. According to Table 4.2, the best accuracy from these three algorithms was produced by LVQ-GA. During variable selection the BP-GA and RBF-GA algorithms used fewer variables than the LVQ-GA algorithm. As described earlier, the greater the number of variables used in the models, the greater the amount of information presented in the models. It is also shown that the advantages and disadvantages of these three algorithms result in different performances. The next section will compare and discuss the characteristics of all algorithms used in feature selection methods.

Table 1. Classification results using the original variables (example 1)

| Neural Networks | | BP | RBF | LVQ |
|---|---|---|---|---|
| Structure | | 30-25-1 | 30-24-1 | 30-16-1 |
| Parameters | | Learning rate=0.15<br>Momentum=0.95 | Learning rate=0.1<br>Momentum=0.4 | Learning rate=0.03 |
| Classification Accuracy | Training | 96.04% | 89.11% | 99.01% |
| | Testing | 88.06% | 88.06% | 92.54% |

Table 2. Results of three GA-based neural selection methods (example 1)

| Neural Networks | | BP-GA | RBF-GA | LVQ-GA |
|---|---|---|---|---|
| Remaining Variables | | 20 | 18 | 22 |
| Parameters | | Crossover rate=0.6<br>Mutation rate=0.05 | Crossover rate=0.6<br>Mutation rate=0.05 | Crossover rate=0.6<br>Mutation rate=0.05 |
| Classification Accuracy | Training | 99.01% | 83.58% | 96.04% |
| | Testing | 85.07% | 82.09% | 92.54% |

## 4.2 Example 2: Glass Identification

The second data set was tested on a glass identification database called repository of machine learning data set from the University of California, at Irvine (German et al., 1987). The data set consisted of 9 attributes for 214 glass instances. All attributes were continuously valued. The goal of the classifier was to determine, based on the glass attributes, whether the glass belonged to the window glass class or non-window glass class. The glass attributes in this case consisted of refractive index, sodium, magnesium, silicon, …, etc. The number of instances in the training set contained 111 window glass samples and 31 non-window glass samples. The test set consisted of 52 window glass samples and 20 non-window glass samples.

As in the previous example, 9 attributes for the 142 training samples were used for supervised neural network training. Nine attribute items for each sample were used as the input. The output was a node indicating whether an individual attribute was window glass or not. In this case, the structure of the neural networks was expressed as 9-X-1, where X represents the number of hidden nodes. *NeuralWorks Professional II* package (Neural Ware, Inc. 1992) was used to perform the computation. After trying different neural structures, learning rate and momentum, the optimal structure for the network was 9-4-1 and its classification rate was 100.0%, i.e., 0 out of the 142 samples were misclassified on BP neural classifier. Correspondingly, the classification rate for test data set was 93.05%, i.e., 5 out of the 72 samples were misclassified on LVQ neural classifier (see Table 3). The highest classification accuracy was provided by RBF, which turned out to be up to 94.44% for the test set.

The performance of GA-based feature selection methods on example two is illustrated in Table 4 The best accuracy obtained by BP-GA, RBF-GA and LVQ-GA were with accuracies of 90.28%, 90.28% and 94.44%, respectively. This means that 7 of the 72 testing data in BP-GA and RBF-GA and 4 of the 72 test data in LVQ-GA were incorrect predictions. The classification performances of GA-based feature selection methods are slightly lower than those neural classifiers using all original input variables of test data. Even the classification results from GA-based methods were not better than those from neural network classifiers. The dimensionalities of the BP-GA, RBF-GA and LVQ-GA models were reduced to 5, 4 and 5. In addition, the best accuracy of these three algorithms, LVQ-GA had the same performance as the best neural classifier, RBF classifier. In general, we expected that, with a reduced set of features, the feature selection method could preserve the same accuracy as that using all of the available features. Or even better, the method may raise the accuracy level due to the elimination of noisy and irrelevant features that may mislead the learning process.

Table 3. Classification results by three neural networks (example 2)

| Neural Networks | | BP | RBF | LVQ |
|---|---|---|---|---|
| Structure | | 9-4-1 | 9-5-1 | 9-6-1 |
| Parameters | | Learning rate=0.15<br>Momentum=0.95 | Learning rate=0.1<br>Momentum=0.4 | Learning rate=0.03 |
| Classification Accuracy | Training | 100.00% | 96.47% | 94.36% |
| | Testing | 93.05% | 94.44% | 93.05% |

Table 4. GA-based classification results (example 2)

| Neural Networks | | BP-GA | RBF-GA | LVQ-GA |
|---|---|---|---|---|
| Remaining Variables | | 5 | 4 | 5 |
| Parameters | | Crossover rate=0.6<br>Mutation rate=0.05 | Crossover rate=0.6<br>Mutation rate=0.05 | Crossover rate=0.6<br>Mutation rate=0.05 |
| Classification Accuracy | Training | 96.48% | 93.66% | 99.30% |
| | Testing | 90.28% | 90.28% | 94.44% |

### 4.3 Comparison with KNN-GA approach

The results listed in Tables 2 and 4 for examples 1 and 2, respectively, will be compared with the benchmark method KNN-GA in Tables 5 and 6. The results shown in the above tables were obtained by running each proposed approach 10 times. KNN-GA was used for selecting the best subset of variables by integrating GA as a selector with KNN classifier. For example 1, the KNN-GA result is the average classification accuracy for training and test data, 92.25% and 84.56%, respectively. This result outperformed the BP-GA and RBF-GA but is lower than the LVQ-GA classifier. For example 2, the result of KNN-GA is that two variables were selected into the model while *k* is equal to 5. The classification accuracy for the training and test data using two input variables was 93.66% and 91.66%, respectively. The proposed GA-based classifiers were better than the KNN-GA algorithm on the test data sets. In these two cases, the LVQ-GA outperformed the BP-GA, RBF-GA and KNN-GA, and thus best described the classification performance.

Table 5. GA-based vs. KNN-GA classifiers descriptive statistics for example 1

| Neural Networks | Net- | BP-GA | RBF-GA | LVQ-GA | KNN-GA |
|---|---|---|---|---|---|
| Average | Training | 98.57% | 85.82% | 94.55% | 92.25% |
| | Testing | 81.59% | 79.50% | 84.82% | 84.56% |
| Standard Deviation | Training | 0.522 | 3.567 | 1.043 | 1.874 |
| | Testing | 4.158 | 3.436 | 5.206 | 3.206 |
| Training | Min | 98.02% | 82.09% | 93.06% | 91.04% |
| | Max | 99.01% | 91.04% | 96.04% | 95.04% |
| Testing | Min | 76.11% | 73.13% | 79.10% | 79.10% |
| | Max | 88.06% | 82.09% | 92.54% | 88.06% |

Table 6. GA-based vs. KNN-GA classifiers descriptive statistics for example 2

| Neural Networks | Net- | BP-GA | RBF-GA | LVQ-GA | KNN-GA |
|---|---|---|---|---|---|
| Average | Training | 98.17% | 93.31% | 95.86% | 93.66% |
| | Testing | 92.77% | 92.36% | 92.71% | 91.66% |
| Standard Deviation | Training | 1.281 | 1.069 | 2.335 | 0.8228 |
| | Testing | 2.487 | 3.903 | 2.542 | 1.2387 |
| Training | Min | 96.48% | 92.25% | 92.25% | 92.95% |
| | Max | 99.30% | 95.07% | 99.30% | 95.77% |
| Testing | Min | 80.55% | 79.16% | 90.28% | 90.28% |
| | Max | 90.28% | 90.28% | 94.44% | 93.05% |

On the other hand, the results listed in Tables 7 and 8 for examples 1 and 2, respectively, show the average number and standard deviation of the remaining variables after running the GA-based reduced models in two case studies. It is notable that the approaches employed in this study are robust and effective for the classification of these two cases because the classification accuracy is close to or higher than eighty percent for the testing data. Moreover, another interesting finding from this study is that eleven variables appear in all four models in Table 7, and two variables appear in all four models in Table 8, respectively. The appearance of these variables may suggest that they have a significant influence on feature selection.

Table 7. Remaining variables in GA-based models (example 1)

| Methods | Remaining Variables | Average # of Variable/std. | Training | Testing |
|---|---|---|---|---|
| BP-GA | 1,2,4,5,6,7,8,13,14,16,17,18,19,20,21,22,25,26,27,30 | 20.4/3.97 | 98.01% | 82.06% |
| RBF-GA | 1,3,4,5,6,7,8,10,11,13,17,18,19,20,21,24,28,30 | 18.2/3.52 | 87.58% | 78.09% |
| LVQ-GA | 1,2,3,4,5,7,9,10,11,13,14,16,17,18,19,20,21,24,26,27, 29,30 | 22.1/3.72 | 95.04% | 86.54% |
| KNN-GA | 1,3,4,5,7,10,11,13,14,17,18,19,20,21,24,26,27,29,30 | 19.5/3.42 | 93.27% | 84.32% |

Notes: (1) Original Var.=X1~X30, (2) 11 variables appear in all four models

Table 8. Remaining variables in GA-based models (example 2)

| Methods | Remaining Variables | Average # of Variable/std. | Training | Testing |
|---|---|---|---|---|
| BP-GA | A,B,C,D,G | 5.9/1.85 | 97.48% | 86.28% |
| RBF-GA | B,D,F,G | 4.7/1.67 | 93.72% | 85.28% |
| LVQ-GA | A,B,C,D,H | 5.5/1.77 | 95.34% | 92.44% |
| KNN-GA | A,B,C,D,H,I | 5.4/1.82 | 93.92% | 91.35% |

Notes: Original Var.=ABCDEFGHI, (2) 2 variables appear in all four models

# 5. DISCUSSIONS

Three GA-based (BP-GA, RBF-GA and LVQ-GA) feature selection methods were trained and tested using two collected data sets. The performance of these three classification approaches is summarized in Tables 1 and 3. It is notable that the approaches employed in this study were effective and efficient for medical data and glass identification because the classification accuracies were just slightly lower than that of neural classifiers that using the original variables for the testing data. In addition, what more interesting is that the classification accuracy of the LVQ-GA feature selection methods had better performance than the other two methods and KNN-GA classifier.

The following discussions are drawn from the above results:

1. Siedlecki and Slkansky (1989) suggest that the GA is a powerful means of reducing the time for finding near-optimal subsets of features from large sets. Kudo and Sklansky (2000) support that the GA is not only useful for large-scale problems but also appropriate for small-scale and medium-scale problems whether the problem belongs to the unconstrained or constrained combinatorial optimization ones. From the above results, the proposed GA-based algorithm is an effective method for eliminating redundant variables and noise data, such as that found in the medical examination data and industrial data used in this study. Although the accuracies of the three GA-based methods were not higher than that from neural classifiers that using all of the original variables, at least, the proposed methods can preserve the classification accuracy.
2. The integrated neural networks and GA feature selection methods were effective in these two examples. In both of them, the LVQ-GA was better than the other data reduction algorithms. The classification accuracy was higher than 92% in these two examples. During the optimization process, the proposed methods utilized the initial weights between layers acquired from the first phase of neural classifier as well as fitness function and its parameter (e.g. a mutation rate). In addition, the performance was highly correlated to the structure of the neural networks. The employed learning algorithms achieved optimal or near-optimal solutions. The LVQ-GA slightly outperformed the other feature selection methods in terms of classification accuracy.
3. A main advantage of the GA-based feature selection methods is that it combines the various benefits of neural networks with the GA approach into a hybrid method. The relationship between the input and output response is a non-linear one rather than linear. The neural networks demonstrated the capability to deal with the non-linear relationship between the input and output data without prior knowledge about the data. The GA can search for the near optimal solution in a reduced time span, and therefore is known for its robustness and effective overall search capabilities. While the KNN classifier performed well in combination with the GA feature extractor, other classification techniques, e.g. BP, RBF and LVQ proposed in this study, were also effective in providing feedback to the GA.

# 6. CONCLUSIONS

This paper proposed a GA-based algorithm that integrated the GA selector and neural network classifiers for feature selection. In this work, the GA searched for near-optimal solutions for the subsets in the feature space. The proposed method can preserve the accuracy using the best subset of features instead of using all of the available features. This algorithm can improve the performance because it can eliminate noisy and irrelevant features that may mislead the learning process. The results demonstrated that the LVQ-GA outperformed the BP-GA and RBF-GA algorithms in both examples because of its learning algorithm and neural network structure. The classification performance also shows that the proposed method is robust and effective in a multi-dimensional data system.

# REFERENCES

1. Backer, S. D., A. Naud and P. Scheunders, “Non-linear dimensionality reduction techniques for unsupervised feature extraction,” *Pattern Recognition Litters*, **19**, 711-720 (1998).

2. Brunzell, H. and J. Eriksson, "Feature reduction for classification of multidimensional data," *Pattern Recognition*, **33**, 1741-1748 (2000).
3. Cardie, C., "Using decision trees to improve case-based learning," *Proceedings of the Tenth International Conference On Machine Learning*, 25-32 (1993).
4. Dash, M. and H. Liu, "Feature selection for classification," *Intelligent Data Analysis*, **1**, 131-156 (1997).
5. Foroutan, I. and J. Sklansky, "Feature selection for automatic classification of non-Gaussian data," *IEEE Transactions on System, Man and Cybernetics*, **17**, 187-198 (1987).
6. German, B., 1987, UCI repository of machine learning database. Available: ftp://ftp.ics.uci.edu/pub/machine-learning-databases/glass/glass.names.
7. Goldberg, D. E., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA (1989).
8. Holland, J. H., *Adaptation in Natural and Artificial System*, University of Michigan Press, Ann Arbor, MI (1975).
9. Jimenez, L., D.A. Landgrebe, "Projection pursuit in high dimensional data reduction: initial conditions, feature selection and the assumption of normality," *IEEE International Conference on Systems, Man and Cybernetics, 'Intelligent Systems for the 21st Century'*, **1**, 401 – 406 (1995).
10. John, G. H., R. Kohavi and K. Pfleger, "Irrelevant features and the subsets selection problem," In: *Proceeding of the Eleventh International Conference on Machine Learning*, 121-129 (1994).
11. Kudo, M. and J. Sklansky, "Comparison of algorithms that select features for pattern classifiers," *Pattern Recognition*, **33**, 25-41 (2000).
12. Mao, J., K. Mohiuddin and A. K. Jain, "Parsimonious network design and feature selection through node pruning," *Proceedings of International Conference of Pattern Recognition*, Jerusalem, Israel, October, 622-624 (1994).
13. Meryer-Base, A. and R. Watzel, "Transformation radial basis neural network for relevant feature selection," *Pattern Recognition Letters*, **19**, 1301-1306 (1998).
14. Narendra, P. M. and K. Fukunaga, "A branch and bound algorithm for feature selection," *IEEE Transactions on Computers*, **26(9)**, 917-922 (1977).
15. Pudil, P., J. Novovicova and J. Kittler, "Floating search methods in feature selection," *Pattern Recognition Letter*, **15**, 1119-1125 (1994).
16. Quinlan, J. R., C4.5: *Programs for Machine Learning*, *Morgan Kaufmann*, San Mateo, CA. (1993).
17. Raymer, M. L., W. F. Punch, E. D. "Goodman, L. A. Kuhn and A. K. Jain, "Dimensionality Reduction Using Genetic Algorithms," *IEEE Transactions on Evolutionary Computation*, **4**, 164-171 (2000).
18. Siedlecki, W. and J. Sklansky, "On automatic feature selection," *International Journal of Pattern Recognition and Artificial Intelligence*, **2**, 197-220 (1988).
19. Siedlecki, W. and J. Sklansky, "A note on genetic algorithms for large-scale feature selection," *Pattern Recognition Letters*, **10**, 335-347 (1989).
20. Tsai, C. Y., "An iterative feature reduction algorithm for probabilistic neural networks," *Omega*, **28**, 513-524 (2000).

## ABOUT THE AUTHORS

**Te-Sheng Li** received the BS degree in industrial management from National Chen-Kung University, Taiwan, ROC and the MS degree in engineering management from University of Missouri-Rolla, USA in 1985 and 1990, respectively. He Holds the PhD degree in industrial engineering and management from the National Chiao-Tung University, Hsinchu, Taiwan, ROC in 2002. Dr. Li is currently an associate professor in the Department of Industrial Engineering at Ming Hsin University of Science and Technology, Hsinchu, Taiwan. His research interests include quality engineering, neural networks, data mining in semiconductor manufacturing.

(Received March 2005; revised May 2005; accepted June 2005)

# 以基因演算法爲基礎之類神經網路進行分類特徵值篩選

李得盛*
明新科技大學工業工程與管理系
304 新竹縣新豐鄉新興路一號

## 摘要

本文提出以基因演算法爲基礎之類神經網路，在分類問題上進行特徵值篩選以維持足夠之資訊。此方法結合基因演算法與類神經分類器，如倒傳遞分類器、放射基底函數分類器以及學習向量量化分類器。本文中基因演算法可移去不相關或多餘之特徵值以得到最佳化特徵向量。首先，本文提出此三方法之執行步驟，然後利用兩筆資料集進行三種方法的績效評估。最後，將比較的結果再與基因演算法結合 *k* 個最近鄰法進行比較。結果顯示，基因演算法結合類神經分類器篩選特徵值法是有效且穩健的。

關鍵詞：基因演算法、倒傳遞網路、放射基底函數、學習向量量化、 *k* 個最近鄰法
(*聯絡人：E-mail:jeff@must.edu.tw)