Available online at www.sciencedirect.com
SCIENCE DIRECT®
Neurocomputing 57 (2004) 295–312
NEUROCOMPUTING
www.elsevier.com/locate/neucom

A false acceptance error controlling method for hyperspherical classifiers

Chen-Wen Yen[a], Chieh-Neng Young[a], Mark L. Nagurka[b,*]

[a] *Department of Mechanical Engineering, National Sun-Yat Sen University, Kaohsiung 80424, Taiwan*
[b] *Department of Mechanical and Industrial Engineering, Marquette University, P.O. Box 1881, Milwaukee, WI 53201-1881, USA*

Received 16 August 2001; received in revised form 16 December 2002; accepted 10 October 2003

Abstract

Controlling false acceptance errors is of critical importance in many pattern recognition applications, including signature and speaker verification problems. Toward this goal, this paper presents two post-processing methods to improve the performance of hyperspherical classifiers in rejecting patterns from unknown classes. The first method uses a self-organizational approach to design minimum radius hyperspheres, reducing the redundancy of the class region defined by the hyperspherical classifiers. The second method removes additional redundant class regions from the hyperspheres by using a clustering technique to generate a number of smaller hyperspheres. Simulation and experimental results demonstrate that by removing redundant regions these two post-processing methods can reduce the false acceptance error without significantly increasing the false rejection error.

Keywords: Hyperspherical classifiers; Pattern recognition; Unknown pattern rejection; Self-organization

1. Introduction

Pattern recognition deals with objects or events to be classified. This study assumes the existence of a known finite set of possible events

$$\boldsymbol{c} = [c_1 c_2 \cdots c_K], \tag{1}$$

* Corresponding author. Tel.: +1-414-288-3513; fax: +1-414-288-7790.
E-mail address: mark.nagurka@marquette.edu (M.L. Nagurka).

0925-2312/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2003.10.008

where $\boldsymbol{c}$ is the set of known classes and the elements c_k of $\boldsymbol{c}$ are called classes. A pattern can be defined as a pair of variables

$$\text{Pattern} = [\boldsymbol{x}, c_k], \tag{2}$$

where $\boldsymbol{x}$ is the feature vector that characterizes the property of c_k. The goal of pattern recognition is to establish a mapping from $\boldsymbol{x}$ to $\boldsymbol{c}$ in order to recognize the class c_k when a feature vector $\boldsymbol{x}$ is presented. This mapping can be constructed by a learning-from-example approach where a number of samples of known classes are given. A classifier can then be designed to find the decision boundary for classifying the training samples. Based on the decision boundary, the classifier enables one to infer the class of unknown samples.

Typically, classifiers are designed to be Bayes optimal [5]. However, in many pattern recognition problems, a scheme for flexible classification error adjustment is often added. In such a scheme, one particular class is chosen as the "true class" and the remaining classes are designated as the "false class." Classifying a true class sample into the false class represents a "false rejection error," whereas assigning a false class sample into the true class is a "false acceptance error." While a Bayesian classifier minimizes the sum of false rejection and acceptance errors, this approach may not be optimal for many many real-world applications. For example, in medical diagnostic problems a missed detection is much less desirable than a false detection.

To resolve such a problem, approaches for traditional statistical classifiers and non-parametric classifiers, such as artificial neural networks, have been developed to flexibly balance the false acceptance and rejection errors [11]. A drawback of these methods is that they assume the samples always are members of the set of known classes $\boldsymbol{c}$. As a consequence, it is difficult for these classifiers to reject samples that do not belong to the known class set. For applications whose false acceptance error is of critical importance, this may create serious problems. To control false acceptance errors, this work proposes two post-processing methods that improve the capability of hyperspherical classifiers in rejecting samples of unknown classes. The paper is organized as follows. The basic idea of hyperspherical classifiers is discussed in the following section. The proposed post-processing methods are introduced in Section 3. Section 4 presents simulation and experimental results that demonstrate the effectiveness of the proposed methods. Future work and conclusions are provided in Section 5.

2. Hyperspherical classifiers

The purpose of hyperspherical classifiers is to cover samples of the same class by the same set of hyperspheres. A distinct advantage of hyperspherical classifiers is that they automatically define a rejection criterion for unknown classes. This property can be illustrated using a multilayer perceptron (MLP) and a hyperspherical classifier to solve a two-class classification problem. For the MLP, the decision boundary obtained by the conventional backpropagation algorithm [5] is plotted in Fig. 1. Trained by the method proposed by Yen and Liu [12] and post-processed by an approach introduced

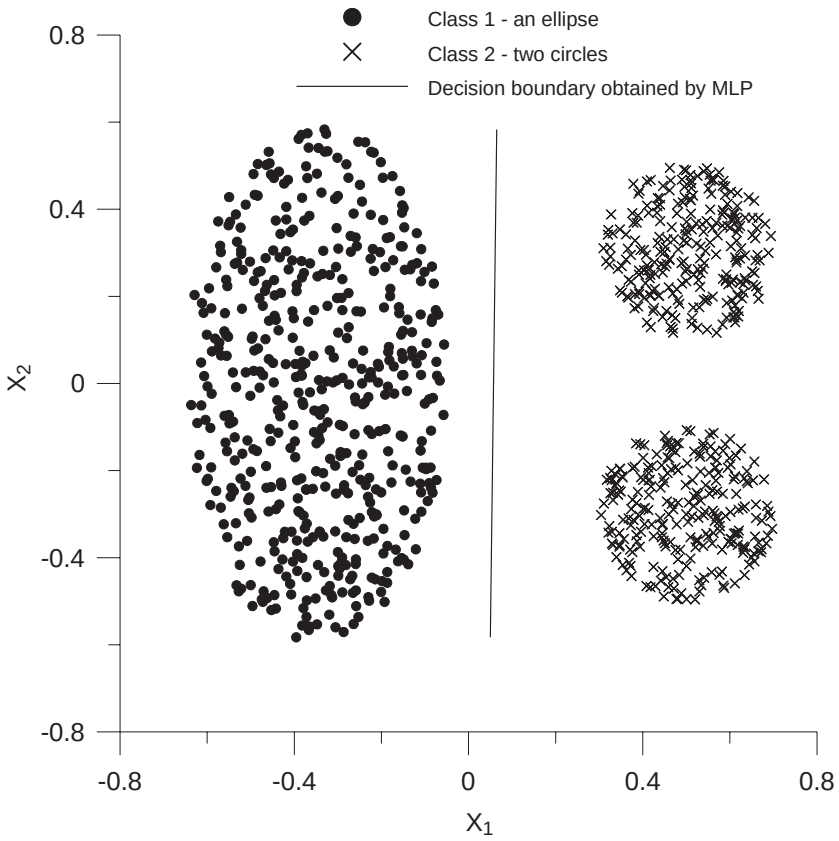

Fig. 1. MLP generated training result for Example 1.

in this paper, the hyperspherical classifier generated the results shown as solid line circles in Fig. 2.

In Fig. 2 each class is enclosed by a circle. This result indicates that the hyperspherical classifier implicitly defines a rejection region, which for this problem encompasses the area outside the two circles. In contrast, no rejection region is defined in Fig. 1. The MLP merely divides the feature space into c_1 and c_2 regions. As a result, the MLP always assigns a sample to one of the two classes even if the sample came from an unknown class.

Several methods have been proposed for the design of hyperspherical classifiers [2,8,9,12]. The common goal of these training methods is to determine the number and parameters (center and radius) of the hyperspheres such that the training set can be classified accurately by the boundary of these hyperspheres. In similarity to conventional classifier training methods, the focus is on finding the decision boundary to classify the given training set. The potential of rejecting samples of unknown classes of the hyperspherical classifiers has not been fully explored.

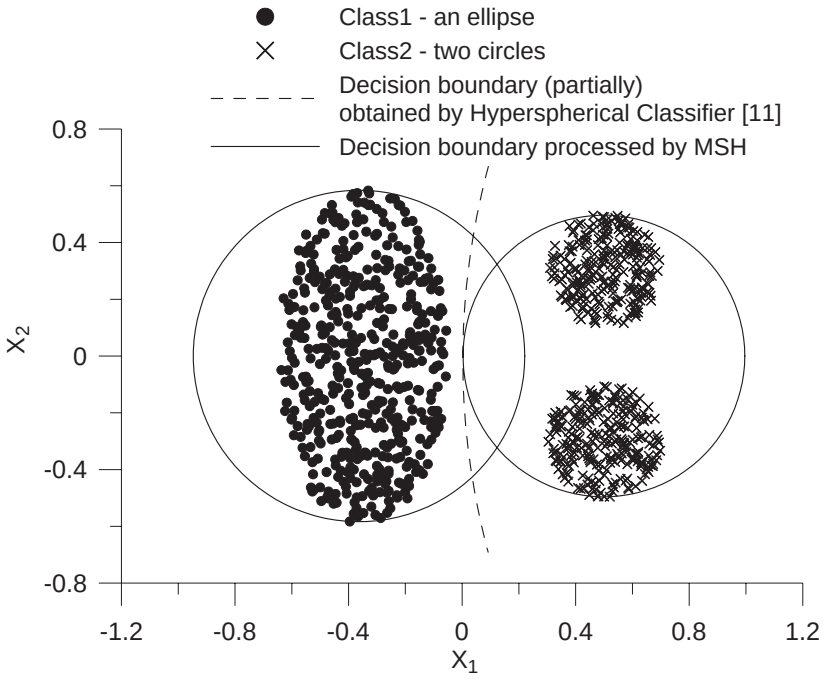

Fig. 2. Hyperspherical classifier generated decision boundaries for Example 1.

This paper proposes two methods to reduce the redundancy of the class region defined by hyperspherical classifiers and introduces a simple technique for balancing the false acceptance and false rejection errors. Since these methods can be used to improve results obtained using any hyperspherical classifier training algorithm, they can be regarded as general post-processing methods for hyperspherical classifiers.

3. Proposed methods

In the previous section Fig. 2 was introduced to illustrate the inadequacy of the hyperspherical classifier training method. As shown by the dashed line circles of Fig. 2, in dealing with c_1 samples, the training method was terminated once a sufficient number of training samples had been classified correctly. Since no consideration is given to the size of the class region, the dashed line circle for c_1 of Fig. 2 is larger than necessary. Hence, a goal of the first proposed method is to design minimum spanning hyperspheres (MSHs) to enclose samples without sacrificing the classification accuracy.

The problem of finding a MSH is very similar to the combinatorial problem for finding a circle of minimum radius to contain a given set of points [1,6]. A mathematical

statement of the problem is

$$\min_{x,y} \max_{j} d((a_j, b_j), (x, y)), \tag{3}$$

where $\{(a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n)\}$ represents a set of n given points that need to be encircled, and $d((a_j, b_j), (x, y))$ is the Euclidean distance from (a_j, b_j) to the center of the minimum spanning circle (x, y).

This paper adopts a generalized version of the self-organization method proposed by Datta [3] to find the MSHs. In particular, at iteration t, the center (x, y) of the minimum spanning circle is updated according to

$$x(t+1) = x(t) + \alpha(a^* - x(t)), \tag{4}$$

$$y(t+1) = y(t) + \alpha(b^* - y(t)), \tag{5}$$

where $(x(0), y(0))$ is the center of gravity of all n given points. Note that (a^*, b^*) represents the farthest point of $(x(t), y(t))$ in $\{(a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n)\}$. The coefficient α is the learning rate selected as [3]

$$\alpha = \alpha_{\max} - t(\alpha_{\max} - \alpha_{\min})/(t_{\max} + 1), \tag{6}$$

where $\alpha_{\max}$ is the maximum learning rate, $\alpha_{\min}$ is the minimum learning rate and $t_{\max}$ is the maximum number of iterations allowed for center updating. With this formula, the learning rate decreases linearly with the iteration number. As indicated by Eqs. (4) and (5), the self-organization method updates the minimum spanning circle by moving its center toward its farthest point. The updating process is repeated until the center converges to a fixed point.

In addition to being simple to implement, the self-organization method can readily be adapted to problems of any dimension. In contrast, it is generally difficult to generalize other minimum spanning circle methods to higher dimension problems. For convenience, the self-organization method will be referred to as the MSH method in this paper. The solid lines of Fig. 2 represent the minimum spanning circles found by the MSH method.

The MSHs are not always inside the class region defined by the original hyperspheres. To avoid unnecessary expansion of the class region, the new class region is defined as the intersection between the class region associated with the old hyperspheres and the region occupied by the MSHs.

Due to the distribution pattern of the samples, the class region found by the MSH method may still be too large even when the smallest possible hyperspheres have been used. For instance, it is impossible to use a circle to construct a nonredundant class region for the c_1 samples of Fig. 2 since the c_1 data are distributed in an ellipse. Similarly, due to the multi-modal distribution pattern of the c_2 samples of Fig. 2, it is also impossible to find a single circle to represent the c_2 region without any class region redundancy. In summary, the incompatibility between the shapes of the actual class region and the hypersphere is one reason behind the class region redundancy of hyperspherical classifiers.

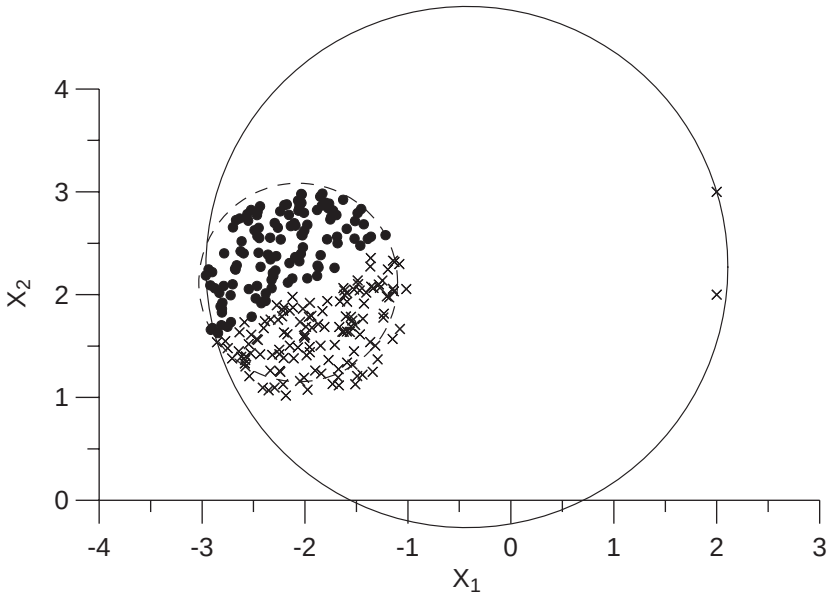

Fig. 3. Clustering results obtained by the conventional ISODATA method.

To resolve this problem, this paper uses an appropriate number of smaller hyperspheres to enclose the data associated with every hypersphere that has a redundant region. To achieve this goal, one can use a conventional clustering method to divide the training samples into a number of groups. A typical criterion of the traditional clustering method is to minimize the within-cluster variation. For example, the following criterion function has often been used

$$E = \sum_{n=1}^{N} \sum_{i=1}^{K_n} (x_i^n - \boldsymbol{m}^n)^{\mathrm{T}} (x_i^n - \boldsymbol{m}^n), \tag{7}$$

where N is the number of clusters, K_n is the number of training samples associated with cluster n, $\boldsymbol{x}_i^n$ is the ith sample belonging to the nth cluster and $\boldsymbol{m}^n$ is the centroid of the nth cluster. An important property of this type of criterion function is that the clustering results depend on the distribution density of the training samples. In particular, cluster centers are apt to be located in the high distribution density region. However, this property may not be desirable for our purpose. To illustrate the nature of the problem, the data points given in Fig. 3 were divided into two clusters by a conventional ISODATA method [4]. All but two samples in Fig. 3 were distributed in a highly concentrated region. Consequently, in order to minimize within-cluster variation, both ISODATA generated cluster centers are located in this high distribution density region. Fig. 3 uses different symbols to denote samples associated with different clusters. Applying the MSH method to find a minimum spanning circle for samples of each cluster yields the two circles of Fig. 3. This result is unsatisfactory since one of the circles contains a significant amount of empty space.

To remedy this problem, this paper uses the following algorithm to find N smaller hyperspheres to contain the same training samples.

1. Denote the feature vector of the training samples as $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots$.
2. Choose $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$ as the initial reference points of the clusters. That is, set $\boldsymbol{r}_1 = \boldsymbol{x}_1, \boldsymbol{r}_2 = \boldsymbol{x}_2, \ldots, \boldsymbol{r}_N = \boldsymbol{x}_N$ where $\boldsymbol{r}_i$ represents the reference point of the ith cluster.
3. Compute the distances between the cluster reference points.
4. With d_{ij} denoting the distance between the ith and jth cluster reference points, find the smallest d_{ij} and the corresponding reference points. Denote the smallest d_{ij} as d^* and the corresponding reference points as $\boldsymbol{r}_i^*$ and $\boldsymbol{r}_j^*$.
5. Denote the nonreference point training samples as $\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots$.
6. Set $k = 1$.
7. Find the smallest distance between the $\boldsymbol{z}_k$ and reference points. Denote this distance as e^*.
8. If $d^* \geqslant e^*$ then continue the algorithm from step 10. Otherwise, continue to the next step.
9. If the distance between $\boldsymbol{r}_i^*$ and its second nearest reference point neighbor is smaller than the distance between $\boldsymbol{r}_j^*$ and the second nearest reference point neighbor of $\boldsymbol{r}_j^*$, replace $\boldsymbol{r}_i^*$ with $\boldsymbol{z}_k$. Otherwise, replace $\boldsymbol{r}_j^*$ with $\boldsymbol{z}_k$.
10. Set $k = k + 1$ and repeat the solution process from step 7 until all the training samples have been used.
11. Repeat the process from step 3 until the reference points no longer change.
12. Associate every training sample to its closest cluster reference point to form N clusters.
13. For the data points associated with each of the clusters, use the MSH method to find a MSH to enclose them.

An important property of the above procedure is that the reference points are determined by trying to maximize the "minimum distance," defined in this study as the shortest distance between all pairs of the reference points. Specifically, in every iteration, the first two reference points associated with the minimum distance are identified first. Next, the procedure seeks to increase the minimum distance by replacing one of these reference points with a training sample selected from the remaining training set to maximize the new minimum distance. This process is repeated until the new minimum distance is not larger than the old minimum distance. Since the reference points have to be a subset of a finite number of training samples, the number of possible reference point combinations is finite. Given this property and the fact that the minimum distance will only get larger, it is impossible for the above procedure to run endlessly. Therefore, this procedure will always converge. Hereafter, this procedure will be referred as the hyperspherical clustering (HC) method.

The circles shown in Fig. 4 are the results obtained by applying the HC method to the data points given in Fig. 3. Note that the two outliers are contained in one circle whereas the other circle encloses all the remaining samples. Compared with the circles of Fig. 3, the HC method has reduced the redundant class region significantly.

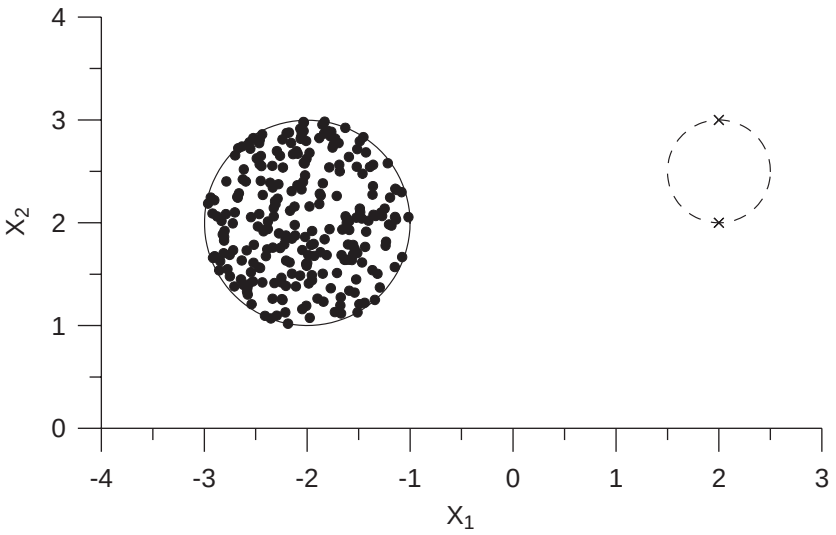

Fig. 4. Clustering results obtained by the HC method.

In addition, by setting a threshold value for the minimum number of data points that a hypersphere must cover, one can easily design an outlier rejection method. This makes the HC method a valuable tool for outlier removal.

With the availability of the MSH and HC methods, the next question is how to increase the number of the hyperspheres so that the false acceptance/rejection errors can be controlled effectively. Toward this goal, this work defines the following index to characterize the "redundancy" of a hypersphere:

$$R = r^n / p, \tag{8}$$

where r is the radius of the hypersphere, n is the dimension of the feature space and p is the number of sample points contained in the hypersphere. For a fixed number of sample points, this redundancy index will be relatively small if the class boundary associated with these points can be approximated closely by a hypersphere. In contrast, if the shape of this class boundary is very different from a hypersphere, then one needs to use a larger hypersphere to enclose these points. This will result in a larger redundancy index. In addition, part of the resulting hypersphere can be considered redundant since it contains no sample points. This redundant part of the hypersphere is to be avoided since any sample points from other classes will be falsely accepted if they appear in such a region. This phenomenon is graphically illustrated in the first example of the following section.

With the redundancy index defined, the following post-processing procedure is proposed.

1. Use an appropriate training method to design a hyperspherical classifier.
2. Use the MSH method to find a MSH for each of the hyperspheres found by the hyperspherical classifier.

3. Compute the redundancy index value for each of the MSH.
4. Use the HC method to split the hypersphere that has the largest redundancy index value into two smaller hyperspheres.
5. Continue the process from step 3, terminating the procedure when one or more of the following criteria have been satisfied.
   1. The false acceptance error is sufficiently small.
   2. The false rejection error has exceeded a prespecified bound.
   3. When the number of hyperspheres is larger than a prespecified limit.

The last criterion can be used to constrain the complexity of the classifier. The first and second criteria are closely related to the trade-off in balancing false acceptance/rejection errors. For example, in designing a verification system where security is the most important concern, the first stop criterion can be used as the design goal for the classifier, whereas the second criterion is used as a safeguard to prevent the false rejection error from exceeding an intolerable limit. However, if convenience for the register users is the major concern for the verification system, then the roles of these two criteria should be switched. The hypersphere splitting process could then be terminated immediately once the false rejection error is larger than the prespecified bound.

When no rigid requirement is imposed on false acceptance/rejection errors, the number of hyperspheres can be chosen to minimize the overall classification error, namely, the sum of false acceptance and false rejection errors. Similarly, the number of hyperspheres could be chosen such that an appropriately weighted sum of the false acceptance and false rejection errors is minimized.

As a final remark, the region occupied by the split hyperspheres is not necessarily a subset of the region covered by the original hypersphere. Therefore, in this study, after every hypersphere splitting operation, the new class region is selected as the intersection between the original hypersphere and the new hyperspheres generated from the hypersphere splitting process.

4. Simulation and experimental results

This section tests the effectiveness of the proposed post-processing methods. Prior to the application of the post-processing methods, the training method of Yen and Liu [12], which was based on an algorithm for linear inequalities [7], was first used to solve the classification problem. In implementing the MSH method, the maximum and minimum learning rates (i.e., $\alpha_{\max}$ and $\alpha_{\min}$ of Eq. (6)) were chosen as 0.04 and 0.0001, respectively. The MSH method was terminated when the distance between hypersphere centers of the successive iterations was less than 10^{-4} or when the number of iterations exceeded 10,000. The false acceptance and false rejection errors are reported as measures of the effectiveness of the post-processing methods, where the error is defined as the ratio of the incorrectly classified testing samples to the total number of testing samples.

Example 1. This example considers a two-dimensional classification problem, illustrated graphically in Fig. 1. In this problem, c_1 samples are inside an ellipse that

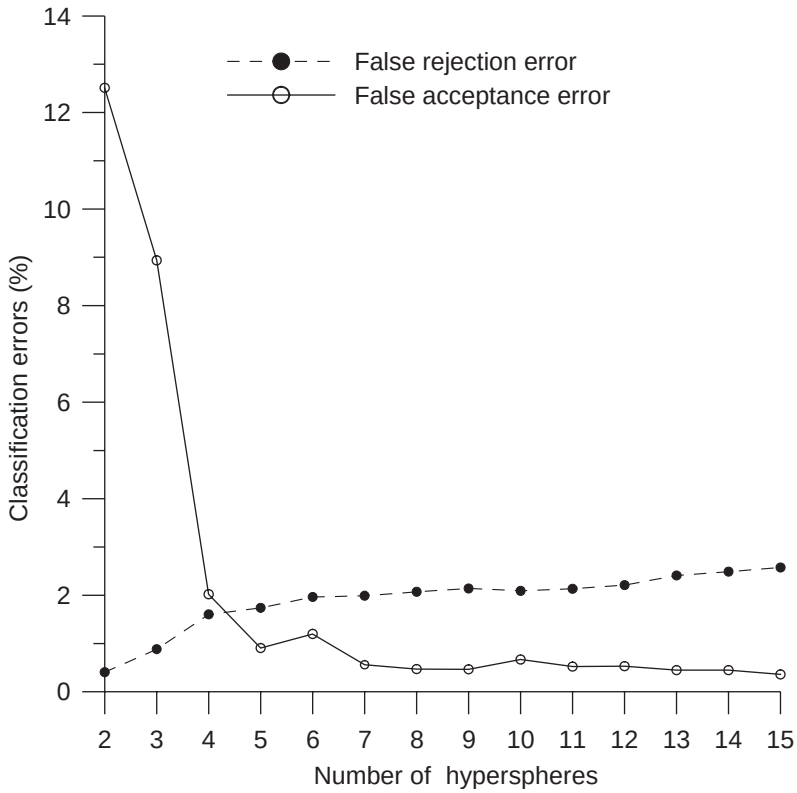

Fig. 5. Classification errors for Example 1.

contains 500 training samples. Two circles, each of which contains 250 training samples, enclose samples of the other class. The circles found by the training method (dashed lines) and the MSH method (solid lines) are depicted in Fig. 2. To test the effectiveness of the proposed post-processing methods, 50,000 rejection area samples and 50,000 c_1 samples were randomly generated to test the classification errors of the c_1 circles. In a similar manner, classification errors of the c_2 circles were also computed.

Before the application of the proposed post-processing methods, the false acceptance and false rejection errors of the hyperspherical classifier were 44.36% and 0.00%, respectively. With the minimum spanning circles, the false acceptance error is reduced dramatically to 12.51%. In contrast, the false rejection error increases only slightly to 0.41%. Next, the HC method was applied to increase the number of hyperspheres one-at-a-time. As functions of the number of hyperspheres, false acceptance and false rejection errors are plotted in Fig. 5. The first five results of the corresponding hypersphere splitting process are plotted in Figs. 6–10, respectively. As shown in these figures, the redundant regions of these circles have a general decreasing tendency as the number of circles goes up.

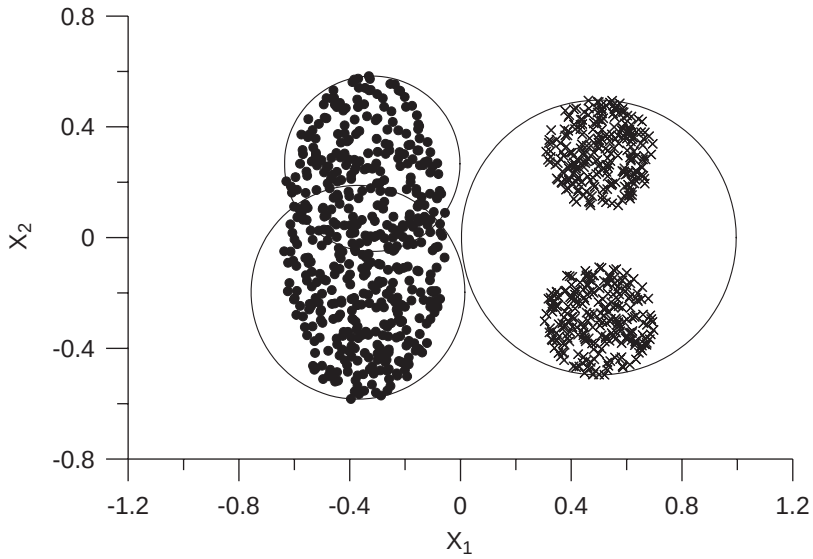

Fig. 6. The first result of the splitting process for Example 1.

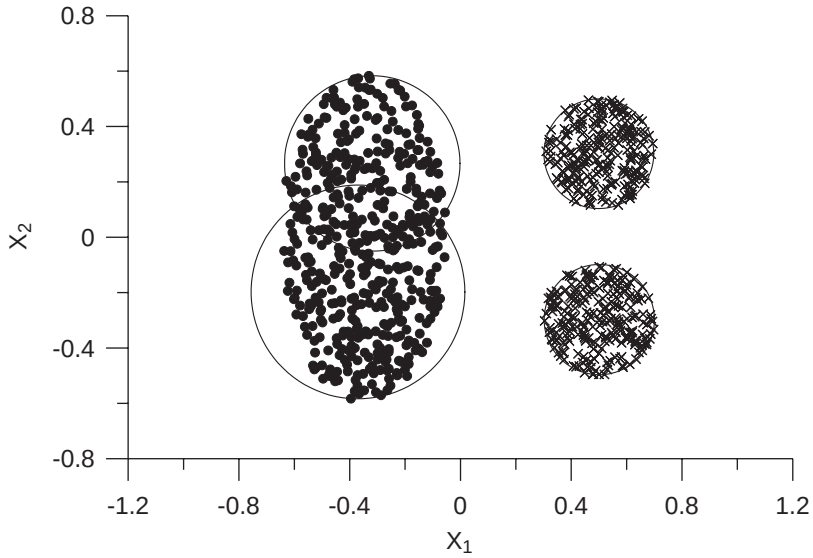

Fig. 7. The second result of the splitting result for Example 1.

As shown in Fig. 5, the false acceptance error is reduced drastically as the number of hyperspheres increases from 2 to 4. This phenomenon can be explained by Figs. 2, 6 and 7, which show that the empty region inside the hyperspheres is

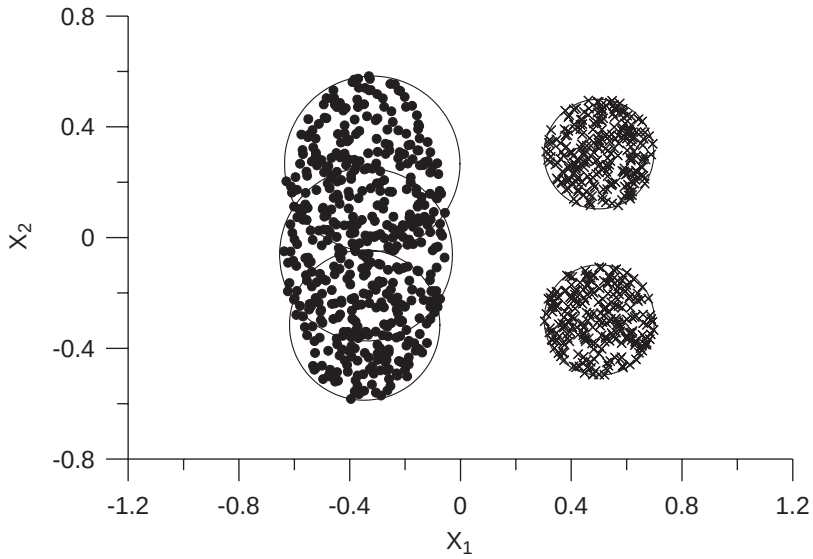

Fig. 8. The third result of the splitting process for Example 1.

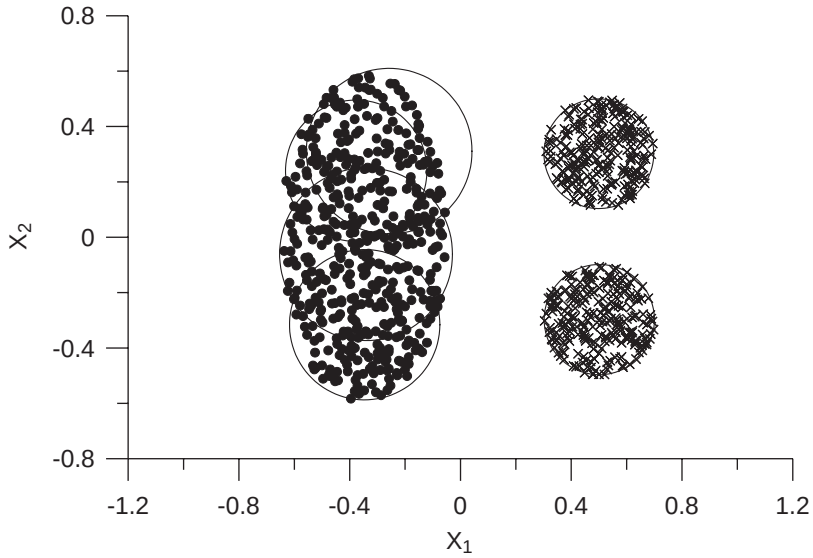

Fig. 9. The fourth result of the splitting process for Example 1.

decreased significantly as the number of hyperspheres increases from two to four. In particular, Fig. 7 uses two circles to enclose c_2 samples and thus closely approximates the optimal decision boundary associated with c_2. As a consequence, with four circles,

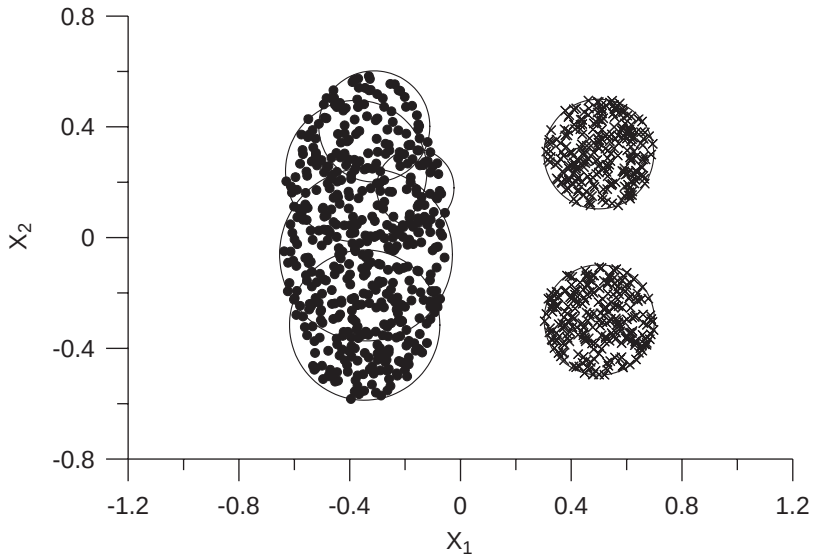

Fig. 10. The fifth result of splitting process for Example 1.

the false acceptance error reduces further to 2.02%, whereas the false rejection error increases mildly to 1.60%. Continuing this hypersphere generation process, it is discovered that the overall classification error is minimized by using eight hyperspheres. The corresponding false acceptance and false rejection errors are 0.47% and 2.07%, respectively.

In order to demonstrate the relative advantage of the proposed approach, this work also uses a MLP to solve the same problem. In training the MLP, the desired outputs for the c_1 and c_2 samples are specified as [1 0] and [0 1], respectively. In standard operation, the MLP assigns a sample to c_1 provided that the first output is larger than the second output of the MLP and so forth. However, in order to control the false acceptance error, this work also requires that the larger output value of the MLP should be no less than a threshold value. Otherwise, the sample is rejected. By varying the threshold value, the MLP possesses the flexibility to control the relative magnitude of false acceptance and false rejection errors. The operating curves obtained by the MLP and hyperspherical classifier are plotted in Fig. 11. The figure shows that the proposed approach provides significantly better overall results than the MLP. Specifically, when both classifiers have the same false acceptance error, the false rejection error obtained by the hyperspherical classifier is much smaller than that of the MLP.

Example 2. This example considers a speaker verification problem with the goal of determining whether a given utterance is produced by a claimed speaker. In creating the database, 10 persons were asked to utter the phrase "kong juy gong cherng" meaning "control engineering" in Chinese. The 8 kHz speech was pre-emphasized with 30 ms Hamming windows shifted every 15 ms. Based on the acoustic parameter selection

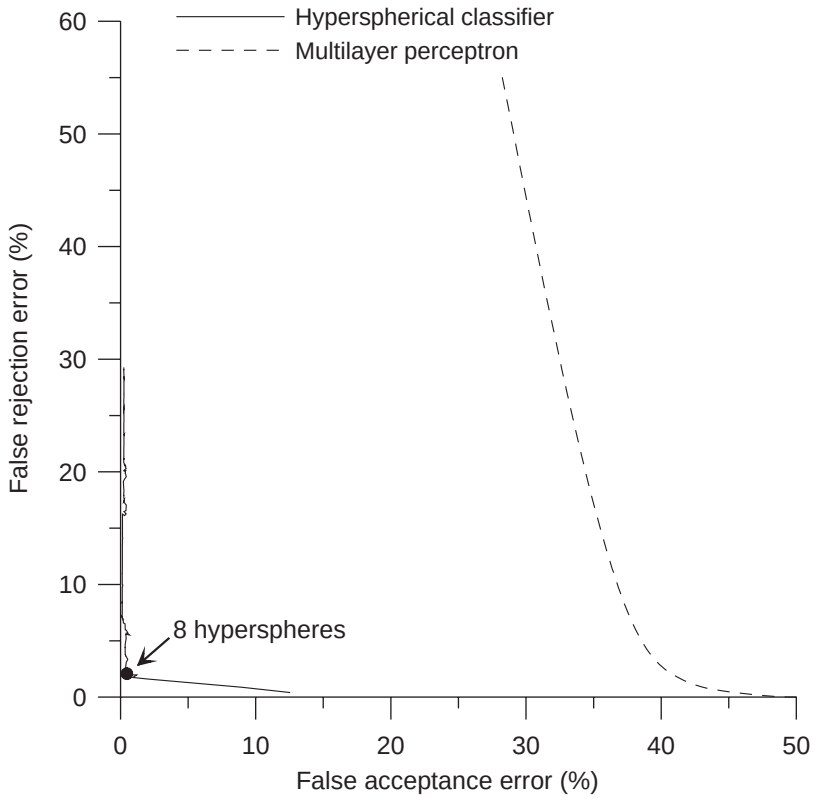

Fig. 11. The operating curves of Example 1.

method proposed by Wolf [10], this example uses 16 feature variables to characterize every set of 400 sets of data taken from each speaker.

The speakers were divided equally into known and unknown class groups. A hyperspherical classifier-based speaker verifier was designed for the five speakers of the known class group. In building these speaker verifiers, 200 sets of samples were taken from each member of the known class group to form the training set. The remaining 1000 known class group samples (200 samples from each person) were used to test the generalization capability of the hyperspherical classifiers. In addition, 2000 sets of unknown class samples (400 samples from each speaker) were used to test the rejection capability of the classifiers.

For the known class group, without any post-processing, the hyperspherical classifier obtains a false acceptance error of 0.075% and a false rejection error of 0.3%, both small enough to be considered very satisfactory. However, the need for false acceptance error suppression comes from the unknown class group whose members are completely unseen during the training phase of the tested classifiers. Without any post-processing,

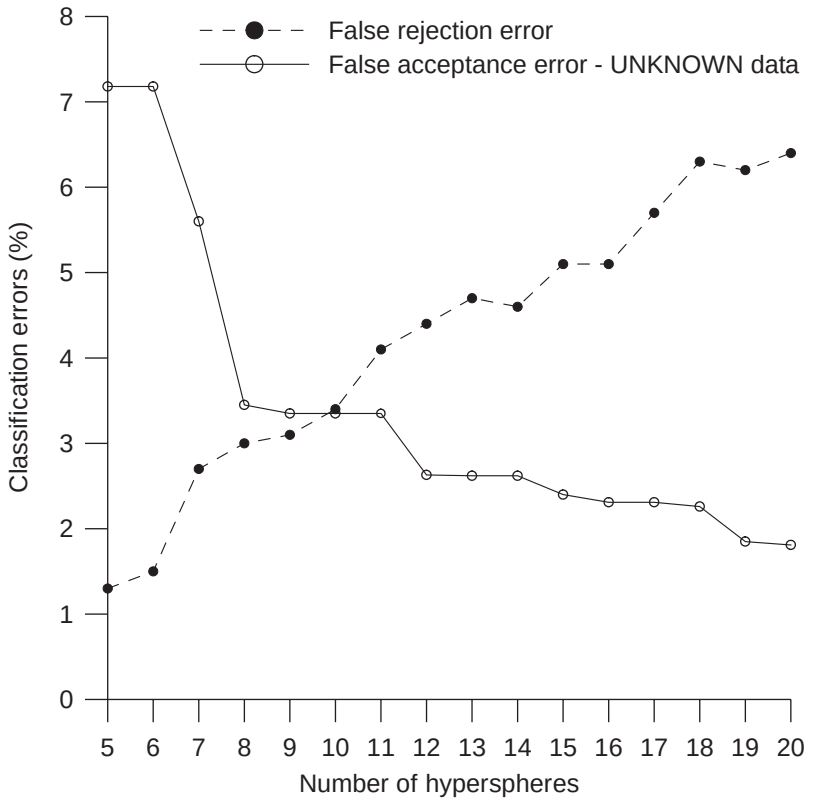

Fig. 12. Classification errors for Example 2.

the hyperspherical classifier incurs a 19.93% false acceptance error in performing verification for the unknown class group speakers. The MSH method reduces this error to 7.18%. At the same time, the false rejection error increases to 1.3%. Since this level of false acceptance error is still not satisfactory for many practical speaker verification applications, the hypersphere splitting method was then applied. The resulting false acceptance and false rejection errors are plotted in Fig. 12 as functions of the number of hyperspheres. As shown in this figure, at the cost of increasing false rejection error, the false acceptance error can now effectively be suppressed. The overall classification error is minimized by using nine hyperspheres, which yields 3.10% false rejection error and 3.35% false acceptance error. However, if the false acceptance error is required to be smaller than 3%, the classifier should have 12 hyperspheres and the false rejection error will increase to 4.4%. Similarly, if the satisfactory false acceptance error is required to be smaller than 2% (1%), the required number of hyperspheres is 19 (29), and the false rejection error would increase to 6.2% (9.0%). These four operating points are also marked in Fig. 13.

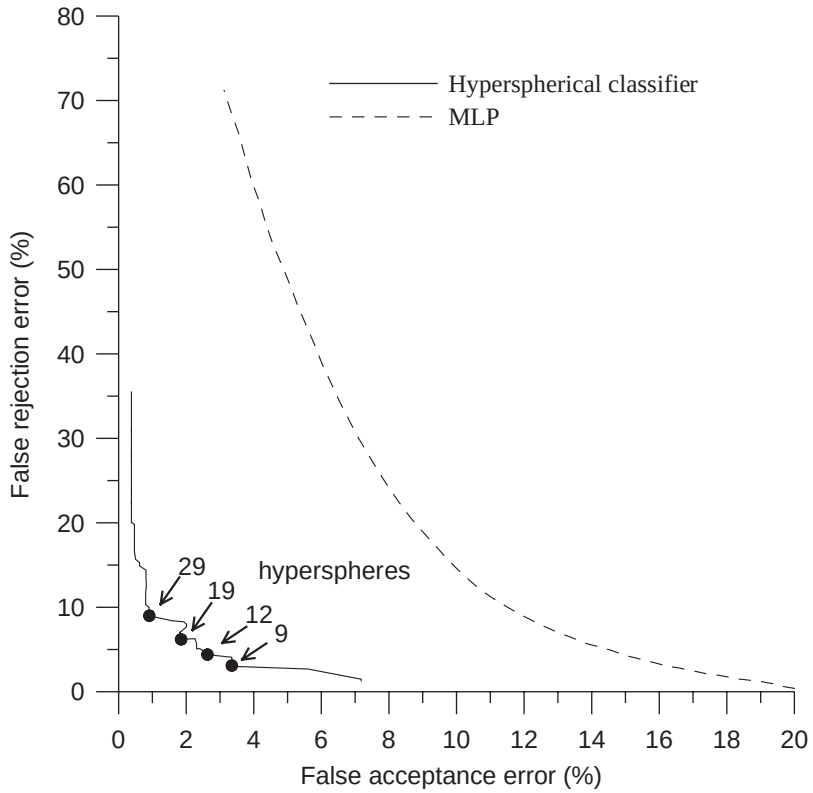

Fig. 13. The operating curves for Example 2.

This problem has also been solved by an MLP. The operating curves obtained by the MLP and hyperspherical classifier are plotted in Fig. 13. This figure demonstrates that the hyperspherical classifier clearly outperforms the MLP.

5. Future work and conclusion

This paper introduces two post-processing methods for hyperspherical classifiers. Based on the self-organization approach, the first method, denoted here as the MSH method, finds minimum radius hyperspheres that contain the data points. By using a clustering method to find a number of uniformly distributed samples as the cluster reference points, the second post-processing method, the HC method, generates several smaller hyperspheres that contain the data points that were originally covered by a single hypersphere. By defining the new class region as the intersection of the class region found by the training method and the interior regions of the hyperspheres found by the post-processing methods, the approach is able to remove redundant class regions

from the hyperspherical classifiers. As demonstrated by the experimental studies, this improves the false acceptance error significantly.

The proposed methods have several distinct features. First, due to their post-processing nature, they can be used in conjunction with any hyperspherical classifier training method. Second, by adjusting the number of hyperspheres generated by the HC method, one can control the relative magnitudes of the false acceptance and false rejection errors of the hyperspherical classifiers. Third, the methods do not require a priori knowledge of the statistical model of the patterns to be classified.

The proposed MSH method uses the minimum hypersphere to avoid the possibility of redundant class regions and thus tries to make the false acceptance error as small as possible. For some applications this may not be an optimal solution. A possible future research direction is to explore this inflexibly of the MSH method. In particular, by systematically increasing the radii of the hyperspheres obtained by the MSH method, one could develop a more flexible tool to balance the false acceptance and false rejection errors of the hyperspherical classifiers. In addition, by analyzing the results obtained by the HC method, another research direction could be to develop a systematic outlier rejection strategy.

Acknowledgements

This research was supported in part by National Science Council of Republic of China under Grant Number NSC 87-2212-E-110-007.

References

[1] L.J. Bass, S.R. Schubert, On finding the disc of minimum radius containing a given set of points, Math. Comput. 21 (1967) 712–714.
[2] B. Bathelor, Practical Approach to Pattern Classification, Plenum Press, New York, 1974.
[3] A. Datta, Computing minimum spanning circle by self-organization, Neurocomputing 13 (1996) 75–83.
[4] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification and Scene Analysis, Wiley, New York, 2000.
[5] S. Haykin, Neural Networks: A Comprehensive Foundation, Macmillan College Publishing Company, New York, 1998.
[6] D.W. Hearn, J. Vijay, Efficient algorithms for the (weighted) minimum circle problem, Oper. Res. 30 (1982) 777–795.
[7] Y.-C. Ho, R.L., Kashyap, An algorithm for linear inequalities and its application, IEEE Trans. Electron. Comput. 14 (1965) 683–688.
[8] D. Relly, L. Dow, C. Erlbaum, A neural model for category learning, Biol. Cybernet. 45 (1982) 35–41.
[9] B.A. Telfer, D.P. Casasent, Minimum-cost associative processor for piecewise-hyperspherical classification, Neural Networks 6 (1993) 1117–1130.
[10] J.J. Wolf, Efficient acoustic parameters for speaker recognition, J. Acoust. Soc. Am. 51 (1971) 2044–2056.
[11] K. Woods, W. Bowyer, Generating ROC curves for artificial neural networks, IEEE Trans. Med. Imag. 16 (1997) 329–337.
[12] C-W.V. Yen, T.-Z. Liu, A competitive learning process for hyperspherical classifiers, Neurocomputing 17 (1997) 99–110.

Mark L. Nagurka received B.S. and M.S. degrees in mechanical engineering and applied mechanics from the University of Pennsylvania in 1978 and 1979, respectively, and a Ph.D. from MIT in 1983. He came to Marquette University (Milwaukee, Wisconsin, USA) in 1996 after teaching mechanical engineering at Carnegie Mellon University (Pittsburgh, Pennsylvania, USA), where he also served as a senior research engineer at the Carnegie Mellon Research Institute. His research interests include mechatronics, vehicle dynamics, control system design, and biomechanics.

Chen-Wen Yen received a B.E. degree in mechanical engineering from Tamkang University in 1982 and M.E. and Ph.D. degrees in mechanical engineering from Carnegie-Mellon University in 1986 and 1989, respectively. Following graduation, he joined the faculty in the Department of Mechanical Engineering at Sun Yat-Sen University. The focus of his research work is the application of neural networks to pattern recognition problems.

Chieh-Neng Young received B.E. and M.E. degrees in mechanical engineering from National Sun-Yat University in 1999 and 2001, respectively. He is now a Ph.D. candidate in the same department. His research interests include learning in artificial neural networks and pattern recognition.