4-1-2005

A Training Sample Sequence Planning Method for Pattern Recognition Problems

Chen-Wen Yen
National Sun Yat-Sen University

Chieh-Neng Young
National Sun Yat-Sen University

Mark L. Nagurka
Marquette University, mark.nagurka@marquette.edu

Accepted version. *Automatica*, Vol. 41, No. 4 (April 2005): 575-581. DOI. © 2004 Elsevier Ltd. Used with permission.

Automatica, Vol. 41, No. 4 (April 2005): 575-581. DOI. This article is © Elsevier and permission has been granted for this version to appear in e-Publications@Marquette. Elsevier does not grant permission for this article to be further copied/distributed or hosted elsewhere without the express permission from Elsevier.

A Training Sample Sequence Planning Method for Pattern Recognition Problems

Chen-Wen Yen
Department of Mechanical Engineering, National Sun-Yat Sen University, Kaohsiung, Taiwan 80424, China

Chieh-Neng Young
Department of Mechanical Engineering, National Sun-Yat Sen University, Kaohsiung, Taiwan 80424, China

Mark L. Nagurka
Department of Mechanical and Industrial Engineering, Marquette University, P.O. Box 1881, Milwaukee, WI

Abstract

In solving pattern recognition problems, many classification methods, such as the nearest-neighbor (NN) rule, need to determine prototypes from a training set. To improve the performance of these classifiers in finding an efficient set of prototypes, this paper introduces a training sample sequence planning method. In particular, by estimating the relative nearness of the training samples to the decision boundary, the approach proposed here incrementally increases the number of prototypes until the desired classification accuracy has been reached.

This approach has been tested with a NN classification method and a neural network training approach. Studies based on both artificial and real data demonstrate that higher classification accuracy can be achieved with fewer prototypes.

Keywords

Classification, Decision boundary Nearest-neighbor rule, Neural networks, Training set editing

1. Introduction

The goal of a classification problem is to find the decision boundary so that objects or events of different classes can be classified accurately. To achieve this goal, a popular approach is to find a subset of the training samples called prototypes to become part of the classifier parameters. A well-known example is the K-nearest-neighbor (K-NN) rule, which classifies a sample based on the class memberships of its K-nearest prototypes. A radial basis function (RBF) neural network can also be classified into this category since the centers (i.e., the prototypes) of its hidden neurons are typically selected from the training set.

The performance of these classifiers depends strongly on the prototypes. Intuitively, the closer a training sample is to the decision boundary, the more information it should be able to provide. It has been shown that classification results can indeed be improved by focusing the training algorithm on the boundary region data. For example, by labeling the desired output of true and false class samples as 1 and 0, respectively, a neural network inversion search technique can be used to find the boundary vicinity data (by finding appropriate inputs for the neural network to generate an output value of 0.5) (Davis & Hwang, 1991). However, since there are usually an infinite number of solutions for such a neural network inversion problem, it is difficult to determine whether the boundary vicinity data have been generated for every portion of the decision boundary. In addition, solving such a neural network inversion problem is often a computationally intensive process.

A computationally simpler approach for finding boundary vicinity data is based on the mutual neighborhood value (MNV) (Chidananda Gowda & Krishna, 1978). To illustrate the idea of the MNV, a two-class problem is considered. Denoting the union of the class i data as C_i, let $\boldsymbol{X}_A$ represent the input portion of a training sample in C_1 and $\boldsymbol{X}_B$ its NN among the training sample input vectors in C_2. Then, if $\boldsymbol{X}_A$ is the Kth nearest neighbor to $\boldsymbol{X}_B$ among the training sample input vectors in C_1, the MNV associated with $\boldsymbol{X}_A$ is defined as

(1) $$\mathrm{MNV}(\boldsymbol{X}_A) = 1 + K.$$

This concept can be generalized to multi-class problems by defining the following minimum mutual neighborhood value (MMNV):

(2) $$\mathrm{MMNV}(\boldsymbol{X}) = \min_{j,j\neq i} \mathrm{MNV}_j(\boldsymbol{X}) \boldsymbol{X} \in C_i,$$

where MNVj(X) represents the MNV value for an $\boldsymbol{X}$ from C_i with respect to C_j. It has been shown that using the MMNV to determine the processing sequence of the training samples can improve the performance of a NN classification method (Davis & Hwang, 1991) and a neural classifier (Sin & deFigueiredo, 1993).

A potential problem of the MMNV method is that it is relatively ineffective in handling problems with overlapping classes. This difficulty can be illustrated by considering a two-class problem whose decision boundary is a simply closed curve. If $\boldsymbol{X}_A$ is a sample in C_1 and $\boldsymbol{X}_B$ is its nearest C_2 neighbor, then $\boldsymbol{X}_A$ and $\boldsymbol{X}_B$ should be at different sides of the decision boundary when there is no class overlapping. Since $\boldsymbol{X}_B$ is closer to $\boldsymbol{X}_A$ than all the other C_2 samples, $\boldsymbol{X}_B$ should therefore be relatively close to the decision boundary and can thus be used to estimate the relative nearness of $\boldsymbol{X}_A$ to the decision boundary. However, for problems with class overlapping, it is possible that $\boldsymbol{X}_A$ and $\boldsymbol{X}_B$ are located on the same side of the decision boundary. As a result, $\boldsymbol{X}_B$ may not be at the vicinity of the decision boundary and is thus not a good reference point for estimating the relative nearness of $\boldsymbol{X}_A$ to the decision boundary. Since $\boldsymbol{X}_B$ can also be the nearest C_2 neighbor for many other C_1 samples, this problem can significantly degrade the performance of a MMNV-based classifier. A remedy of this weakness of the MMNV method is provided in this paper.

The paper is organized as follows. Two previously proposed MMNV-based classification methods are discussed in the next section. The proposed approach is presented in Section 3, and experimental results that demonstrate the effectiveness of the approach with real-world data are provided in Section 4. Conclusions are drawn in Section 5. For the sake of simplicity, the scope of the paper is limited to classification problems with only two classes, namely, C_1 and C_2. However, the approach can readily be generalized to multi-class problems.

2. Previous work

Two MMNV-based classification methods are presented to demonstrate how an appropriately arranged training sample sequence (TSS) can help a classifier find an efficient set of prototypes. The two classification methods are implemented and compared with the proposed approach in Section 4.

2.1. The condensed NN rule

To build a compact NN classifier, the condensed nearest neighbor (CNN) rule uses a subset of the training samples as prototypes by increasing the number of prototypes one at a time (Hart, 1968). In particular, the design process of the CNN rule can be described conceptually by the following procedure:

1. Divide the training set into two parts called STORE and GRABBAG. Initially, STORE contains only the first sample of the training set whereas GRABBAG includes all the remaining training samples.
2. With the points in STORE as the prototypes, use the NN rule to classify the next sample in GRABBAG. This sample stays in GRABBAG if it can be classified correctly. Otherwise, it is transferred to STORE and thus becomes a new prototype.
3. Repeat step 2 until all the samples in GRABBAG have been tested.
4. Start a new pass through GRABBAG by repeating steps 2 and 3. Repeat this procedure until no sample can be transferred from GRABBAG to STORE in one pass.

In this procedure, when a GRABBAG sample is classified incorrectly, the sample is transferred to STORE and becomes a new prototype. Thus, the earlier a sample is sent to STORE for classification, the more likely it will become a prototype. However, some of these prototypes may become redundant when their decision boundary forming function is replaced by other prototypes, which enter into STORE later but are closer to the decision boundary. Therefore, the order of the training samples in GRABBAG can influence the final classification result significantly. Hereafter, the order of the training sample will be referred to as the TSS. In view of the importance of TSS, the MMNV method has been used to arrange the TSS for the CNN rule (Chidananda Gowda & Krishna, 1979). This MMNV-based CNN rule will be tested in Section 4.

2.2. The OI network

In terms of architecture, the optimal interpolative (OI) network is essentially a one-hidden-layer feedforward network. In contrast to conventional multilayered feedforward networks which are typically trained by iterative gradient search methods, an OI network is trained by a noniterative least-squares (LS) algorithm called RLS-OI. Conceptually, the procedure of this RLS-OI is illustrated below. For simplicity, this work assumes that the activation function used by the hidden layer is the Gaussian function.

1. Divide the training samples into active and inactive sets. Initially, the active training set is empty and the inactive training set contains all training samples.

2. Use the MMNV to set up the TSS. Essentially, this determines the order of the training samples in the inactive training set.
3. Transfer the first misclassified inactive set training sample to the active training set. Initially, all training samples are considered to be classified incorrectly.
4. Try to add a new hidden unit into the neural network using the input portion of the newly added active training set sample as the center of the new hidden neuron.
5. Compute the weights between the hidden and output layers by trying to satisfy the constraints associated with the active training set in the LS sense.
6. Compute the classification accuracy of the neural network for the entire training set.
7. If the result of 6 is satisfactory, admit this newly appended hidden neuron, Otherwise, remove the newly added neuron from the neural network.
8. Continue the training process from step 3 until termination.

This training procedure can be terminated in several ways. Here, the training process is stopped when all training samples have been transferred to the active training set. In addition, during the training process, the validation error is computed and stored after admitting every new hidden unit. At the end of the training process, the network structure that yields the smallest validation error is recovered and employed for future classification tasks. For a detailed mathematical treatment of the OI network training process, readers are referred to a reference (Sin & deFigueiredo, 1993).

One of the distinct features of this RLS-OI training algorithm is that in computing the connection weights, only the active training set samples are employed. In responding to unsatisfactory classification results, the RLS-OI training algorithm sends misclassified samples incrementally to the active training set. In general, the earlier a sample is sent to the active training set, the more likely it will become a prototype, which in this case is the center of a hidden unit. Therefore, by sending samples closer to the decision boundary earlier into the active training set, the OI network has a higher likelihood of finding an efficient set of prototypes. Again, this demonstrates the importance of a successful TSS planning method.

3. Methodology

As indicated in Section 1, one drawback of the MMNV method is that it is relatively ineffective in dealing with problems with overlapping. To resolve this problem, the proposed approach applies two filtering techniques to the training set before computing MMNVs. With the application of the 3-NN rule, the first phase of the proposed approach uses an edited NN rule for training sample screening (Ferri, Albert, & Vidal, 1999). In particular, samples whose actual class membership is different from the class membership determined by the 3-NN rule are removed from the training set (Wilson, 1972). Henceforth, this technique will be referred to as the 3-NN editing rule.

To investigate the effect of this 3-NN editing rule, consider the following Bayes decision rule which assigns a training sample to class C_b if

(3) $$p(\boldsymbol{X}|C_b)P(C_b) \geqslant p(\boldsymbol{X}|C_l)P(C_l) \mathrm{for all} l \neq b,$$

where $p(\boldsymbol{X}|C_b)$ and $P(C_b)$ are the class-conditional and a priori probability density functions (PDFs) of class C_b. In this work, the class membership b determined by Eq. (3) is called the Bayes class membership.

As shown by the Bayes decision rule, there can be two reasons for a classification error. One reason is that the classifier cannot provide sufficiently accurate PDFs. A second reason is that the actual class membership of the sample is different from its Bayes class membership determined by Eq. (3). This occurs when

(4) $p(\boldsymbol{X}|C_b)P(C_b) > p(\boldsymbol{X}|C_a)P(C_a)$

with a denoting the actual class membership of sample $\boldsymbol{X}$. Such samples cannot be correctly classified by the Bayes decision rule. For convenience, these samples will be referred to as inconsistent membership samples in the remaining part of this paper.

Since the PDFs are typically unknown, it is usually difficult, if not impossible, to identify inconsistent membership samples in the training set. To partially resolve this problem, the first phase of the proposed approach uses the 3-NN rule to estimate the actual membership of the training samples. Fig. 1 graphically illustrates the influence of the 3-NN editing rule, depicting the PDFs of a two-class classification problem before and after the application of the rule (Ferri et al., 1999). As shown in Fig. 1, the 3-NN editing rule has a tendency to convert an overlapping problem into a problem with little or no overlapping. However, perfect conversion of the PDFs is difficult to achieve in practice, as described below.

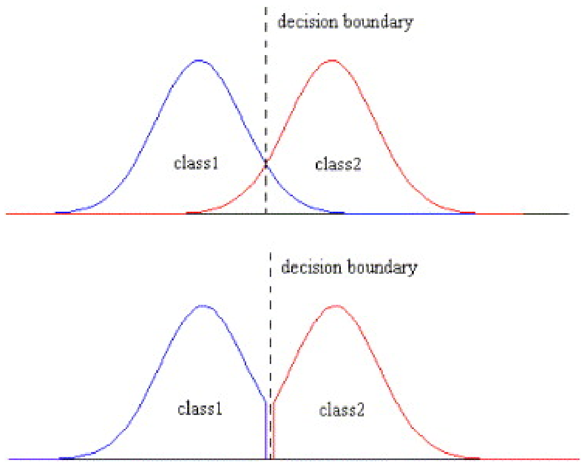


Fig. 1. Effect of the 3-NN editing rule for the PDF (top: before; bottom: after).

The 3-NN rule estimates which class has the larger PDF value by finding the three closest neighbors for the tested sample. With the K-NN rule, the PDF $p(\boldsymbol{X})$ of a random variable $\boldsymbol{X}$ can be estimated from N observations of $\boldsymbol{X}$ by using

(5) $\hat{p}_K(\boldsymbol{X}) = \frac{K-1}{N}\frac{1}{V(K,N,\boldsymbol{X})},$

where $V(K,N,\boldsymbol{X})$ is the smallest hyper-volume enclosing all points at least as near to $\boldsymbol{X}$ as the Kth NN of $\boldsymbol{X}$ (Lofsgaarden & Quesenbery, 1965). It has been shown (Fukunaga & Hostetler, 1973) that, in the neighborhood of a sample point $\boldsymbol{X}$, the expected value of the mean-square error of this estimation can be approximated by

(6) $E\{(\widehat{p}_K(\boldsymbol{X}) - p(\boldsymbol{X}))^2\} \approx \frac{p^2(\boldsymbol{X})}{K} + c^2(\boldsymbol{X})p^{-4/N}(\boldsymbol{X})\left(\frac{K}{N}\right)^{4/N},$

where

(7) $c(\boldsymbol{X}) = \frac{1}{2(N+2)\pi}\Gamma^{2/N}\left(\frac{N+2}{2}\right)\mathrm{tr}\left\{\frac{\partial^2 p(\boldsymbol{X})}{\partial \boldsymbol{X}^2}\right\},$

where $\Gamma()$ denotes the gamma function. Based on this result, for asymptotic unbiasedness and consistency of the estimator, K should go to infinity and K/N should go to zero. With a finite number of samples, a PDF estimation error is often inevitable for the K-NN rule. Thus, the 3-NN editing rule may not be able to remove all inconsistent membership samples from the training set. This can significantly degrade the performance of a MMNV-based classifier since a single inconsistent membership sample can become the NN of many of its opposite class samples.

A possible approach to improve the accuracy of the K-NN-based PDF estimation is to use the following average K-NN rule:

(8) $$\bar{p}_K = \frac{1}{K}\sum_{i=1}^{K} \widehat{p}_i(\boldsymbol{X}),$$

where $\widehat{p}_i(\boldsymbol{X})$ denotes the i-NN estimates of $p(\boldsymbol{X})$ (Parthasarathy & Chattarji, 1990). Statistically, this method is expected to provide a more reliable PDF estimate than one involving any individual K-NN rule since it has been shown that

(9) $$E\{(\bar{p}_K(\boldsymbol{X}) - p(\boldsymbol{X}))^2\} = \frac{1}{K^2}\sum_{i=1}^{K} E\{(\widehat{p}_i(\boldsymbol{X}) - p(\boldsymbol{X}))^2\}.$$

Based on this improvement, the weighted average K-NN rule (Parthasarathy & Chattarji, 1990) is adopted here to estimate the PDF

(10) $$\bar{p}_K = \sum_{i=1}^{K} W_i \cdot \widehat{p}_i(\boldsymbol{X}) / \sum_{i=1}^{K} W_i,$$

where the weighting coefficients are determined by the following rule:

(11) $$W_i = \sin(\pi i / 2k_0) \text{for} 1 \leqslant i \leqslant 2k_0.$$

Otherwise, W_i is set to zero. Note that with n denoting the dimensionality of the sample space, k_0 is specified as

(12) $$k_0 = N^{4/(n+4)}.$$

With this average K-NN rule for PDF estimation, after the application of the 3-NN editing rule, the second phase of the proposed approach applies the following procedure to every remaining training sample:

1. For $k = 1, \dots, K$, use the estimation rule of Eq. (10) to compute $\bar{p}_k^1$ and $\bar{p}_k^2$, which represent the PDF estimates for classes 1 and 2, respectively.
2. Define d_k as

   (13) $$d_k = P(C_1)\bar{p}_k^1 - P(C_2)\bar{p}_k^2 \text{for} k = 1, \dots, K,$$

   where $P(C_1)$ and $P(C_2)$ are the a priori PDFs of C_1 and C_2, respectively. If not all d_k values for the given sample are of the same sign, the proposed approach removes this sample from the training set before the computation of the MMNV.

The idea behind this procedure can be explained as follows. As shown by the Bayes rule of Eq. (3), the sign of d_k can be used as a criterion for class membership determination. Compared with the conventional NN rule, requiring all d_k values of a sample to have the same sign is a much more conservative criterion for sample classification. By using such a conservative measure for ensuring samples to have identical actual and Bayes class memberships, the second phase of the proposed approach can remove more inconsistent membership samples than the 3-NN edited rule. For convenience, the MMNV computed after filtering the training set with the proposed approach is called EMMNV (edited MMNV) henceforth.

A trade-off of using this average K-NN rule-based filtering technique is that it increases the risk of incorrectly removing samples whose actual membership and Bayes membership are the same. The training set may thus have fewer samples than necessary. This can degrade the accuracy of the classification result especially for problems with few samples. Therefore, in applying the proposed approach, special attention should be given to the number of training samples after using the proposed techniques to filter the training set.

An important practical issue for a classification method is its computational complexity. In the development stage of the proposed approach, the K-NN-based training set filtering technique is used which adds a computational demand. In particular, since locating the NNs for every training sample involves computing the distances between all training samples, this part of the computational cost increases quadratically with the number of training samples. However, with a smaller training set, the proposed approach can find the prototypes more efficiently, offsetting the additional computational cost required by the training set filtering process. Thus, for the development stage, the overall computational requirement depends on the actual content of the original training set and the classification method under development.

For a classifier, the development stage is basically a one-time process and is typically performed off-line. As such, the focus of the computational burden is on the implementation stage in which the classifier is often required to perform on-line classifications repeatedly. For this application, the number of computations is proportional to the number of prototypes of the classifier. With the proposed TSS planning method, the number of prototypes can be reduced significantly, as is demonstrated by the experimental results presented in the following section. Consequently, the real-time application of the classifier can be implemented efficiently.

4. Experimental results

The goal of this section is to test the proposed approach and compare it with MMNV-based TSS planning methods. Based on the nature of the training data, this section is divided into two parts. In the first part, a set of artificial data is used to represent problems with different degrees of overlapping. The second part tests the proposed approach in solving real-world problems.

4.1. Artificial data

One of the motivations of this work is to investigate the difficulty of the MMNV-based TSS method to solve problems with overlapping. Therefore, the objective here is to compare the performance of the classification methods in addressing problems with different degrees of overlapping. In particular, a series of two-class, two-dimensional, Gaussian-distributed classification problems are considered. The means of the PDFs of C_1 and C_2 are [0, 0] and $[m_x, 0]$, respectively, with mx being a parameter used to adjust the degree of overlapping. Both classes have a unit covariance matrix. With an increment of $\Delta m_x = 0.2$, problems with $m_x = 0.4, 0.6, \ldots, 6.0$ are solved by the MMNV- and EMMNV-based OI network training methods.

For each m_x value, 500 training, 500 validation and 10,000 test samples were randomly generated for each class of the training samples. For the sake of reliability, each of the tested problems was solved ten times using ten different training sets. The differences between the optimal classification error (obtained by the analytically derived Bayes rule) and the test sample classification error obtained by the OI networks are plotted in Fig. 2 as functions of m_x. Also as functions of mx, the averages of the number of prototypes required by the OI networks are plotted in Fig. 3.

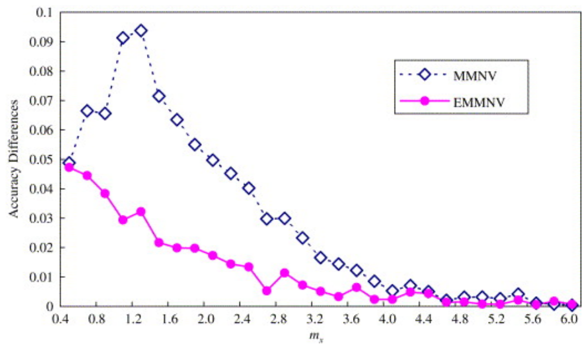


Fig. 2. Accuracy differences for the two tested methods of example.

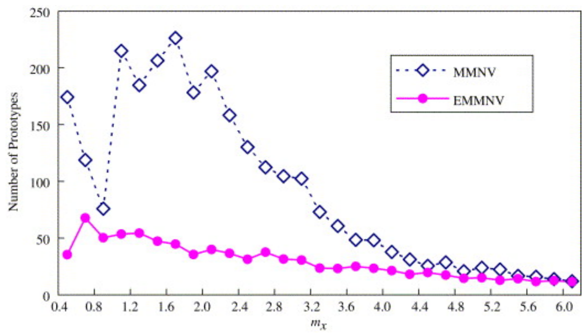


Fig. 3. Number of prototypes for the two tested methods of example.

The results of Fig. 2 show two general trends. The first trend is that the inaccuracy of the two methods increases with the degree of overlapping. Hence, as the value of m_x becomes smaller, the differences between the optimal classification error and the error obtained by the OI networks become larger. This indicates that higher degrees of overlapping can increase the complexity of the classification problem. The second trend is that the proposed approach improves the classification results for problems with more overlapping. In contrast, as the value of m_x becomes larger, the classification accuracy improvement achieved by the EMMNV becomes less significant. This is expected since the proposed approach is designed specifically to overcome the weakness of the MMNV in dealing with problems with overlapping.

Fig. 3 indicates an additional advantage of the proposed approach in that it requires fewer prototypes in most cases. As expected, this reduction in the number of prototypes becomes less significant as the degree of overlapping becomes smaller. Fig. 3 verifies the effectiveness of the proposed approach in remedying the weakness of the MMNV in dealing with classification problems with overlapping.

4.2. Real-world data

Three real-world problems are considered. They have all been studied in the Statlog project (Hichie, Spiegelhalter, & Taylor, 1994) and solved by different classification methods, including a backpropagation (BP) method, a NN classifier, a learning vector quantization (LVQ) classifier, a support vector machine (SVM) method as well as a conventional RBF network. In addition, a 10-fold cross-validation technique was employed to evaluate the accuracy of the tested classification methods. Specifically, by dividing samples into 10 subsets, classifiers were designed on 9 subsets and tested on the subset left out. By using every subset once as the test subset, this procedure was repeated 10 times and the averages of the classification errors for the test subsets and the prototype number were reported for comparison. The problems selected here are:

1) *Wisconsin breast cancer data*: This database was obtained from the UCI repository of Machine Learning Databases and Domain Theories. It includes 699 samples, each of which has nine features of a breast tumor. The output indicates whether the tumor is benign or malignant. After testing 34 classification

methods, the best 10-fold cross-validation classification accuracy obtained by the Statlog project was 97.2% (www.phys.uni.torun.pl/kmk/projects/datasets.html).

2) *Australian credit card data*: This data set was used by the Statlog project to assess applications for credit cards based on 14 attributes. This problem, with 690 samples in total, was solved using 27 classification methods. The best 10-fold cross-validation classification accuracy obtained by the Statlog project was 86.9% (www.phys.uni.torun.pl/kmk/projects/datasets-stat.html).
3) *Diabetic data*: Based on eight features, the objective of this problem obtained from the UCI repository was to determine whether a person is diabetic. This problem, with 768 examples, was tested with 25 classification methods. The best 10-fold cross-validation classification accuracy obtained by the Statlog project was 77.7% (www.phys.uni.torun.pl/kmk/projects/datasets.html).

By employing the same 10-fold cross-validation technique as the Statlog project, this part of the experiment integrates the MMNV- and EMMNV-based TSS planning methods with the CNN rule and the OI network. The results are summarized in Tables 1 and 2 for the CNN rule and the OI network, respectively. The results indicate that the EMMNV-based method outperforms the MMNV-based methods in both classification accuracy and prototype requirements. The only exception is the breast cancer OI network result where the MMNV provides slightly higher classification accuracy than the EMMNV (0.971 versus 0.965). Judging from the high classification accuracy, this problem has relatively small overlapping and the MMNV should therefore be reasonably accurate in characterizing the relative nearness of the samples to the decision boundary. Even in this case, the proposed approach reduces the average prototype number from 12.9 to 3.6.

Table 1. Summary of results for real-world data classified by the CNN rule

Problem	**Classification accuracy**		**Number of prototypes**	
	MMNV	**EMMNV**	**MMNV**	**EMMNV**
Breast cancer	0.937	0.954	58	14
Credit card	0.746	0.846	192.1	17.1
Diabetes	0.649	0.736	310.2	63.8

Table 2. Summary of results for real-world data classified by the OI network

Problem	**Classification accuracy**		**Number of prototypes**	
	MMNV	**EMMNV**	**MMNV**	**EMMNV**
Breast cancer	0.971	0.965	12.9	3.6
Credit card	0.841	0.875	54.9	13.6
Diabetes	0.760	0.783	100.8	26

Since the focus of this work is on TSS planning, no effort has been made to improve the training method itself. However, as shown in Table 2, with the EMMNV-based TSS, the OI network provides higher classification accuracy than the best Statlog project results for the credit card assessment (0.875 versus 0.869) and the diabetes determination (0.783 versus 0.777). For the breast cancer data, due to the small overlapping nature of the problem, the classification accuracy of the proposed approach is slightly inferior to the best Statlog project result (0.965 versus 0.972). However, with the MMNV, the 0.971 classification accuracy obtained by the OI network is very close to the best Statlog project result.

5. Conclusions

The classification power of many classifiers comes partially from their prototypes, which are often chosen as a subset of the training samples. Compared with a randomly determined training sample sequence, it has been shown that planning the training sample sequence in accordance with the relative nearness of the training samples to the decision boundary can assist a classifier in finding a better set of prototypes and thus enabling a higher classification accuracy to be achieved. However, the conventional training sequence planning method has difficulty in dealing with problems with overlapping. With the introduction of two training set filtering techniques, this paper develops a new criterion to characterize the relative nearness of the training samples to the decision boundary and proposes a new training sample sequence planning method. Experimental results demonstrate that the proposed approach can achieve higher classification accuracy with fewer prototypes than conventional methods. It is hoped that this approach can be integrated with more prototype-based classifiers. A future research direction is to develop new training methods specifically for the proposed training sample sequence planning method. Another area of future research is to investigate the influence of training set size on the proposed approach.

Acknowledgements

This research was supported in part by National Science Council of Republic of China under Grant no. NSC 89-2212-E110-020.

References

Chidananda Gowda and Krishna, 1978 K. Chidananda Gowda, G. Krishna **Agglomerative clustering using the concept of mutual nearest neighborhood** Pattern Recognition, 10 (1978), pp. 105-112

Chidananda Gowda and Krishna, 1979 K. Chidananda Gowda, G. Krishna **The condensed nearest neighbor rule using the concept of mutual nearest neighborhood** IEEE Transactions on Information Theory, IT-25 (1979), pp. 488-490

Davis & Hwang, 1991 Davis, T., & Hwang, J.-N. (1991). Attentional focus training by boundary region data selection. *Proceedings of the international joint conference on neural networks*, Vol. 1, Baltimore, MD (pp. 676–681).

Ferri et al., 1999 F.J. Ferri, J.V. Albert, E. Vidal **Considerations about sample-size sensitivity of a family of edited nearest-neighbor rules** IEEE Transactions on Systems, Man and Cybernetics—Part B: Cybernetics, 29 (1999), pp. 667-672

Fukunaga and Hostetler, 1973 K. Fukunaga, L.D. Hostetler **Optimization of k-nearest-neighbor density estimates** IEEE Transactions on Information Theory, 19 (1973), pp. 320-326

Hart, 1968 P.E. Hart **The condensed nearest neighbor rule** IEEE Transactions on Information Theory, IT-14 (1968), pp. 515-516

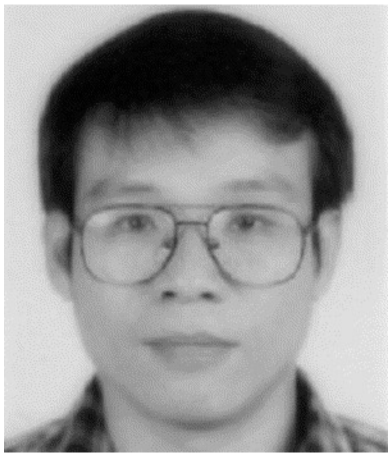
Hichie et al., 1994 D. Hichie, D.J. Spiegelhalter, C.C. Taylor **Machine learning, neural and statistical classification** Ellis Horwood, New York (1994)

Lofsgaarden and Quesenbery, 1965 D.O. Lofsgaarden, C.P. Quesenbery **A nonparametric estimate of a multivariate density function** Annals of Mathematical Statistics, 36 (1965), pp. 1049-1051

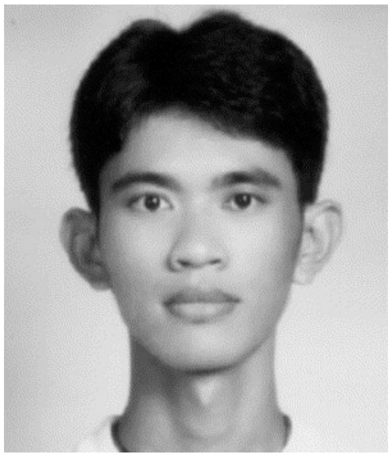
Parthasarathy and Chattarji, 1990 G. Parthasarathy, B.N. Chattarji **A class of new KNN methods for low sample problems** IEEE Transactions on Systems, Man and Cybernetics, 20 (1990), pp. 715-718

Sin and deFigueiredo, 1993 S.K. Sin, R.J.P. deFigueiredo **Efficient learning procedures for optimal interpolative nets** Neural Networks, 6 (1993), pp. 99-113

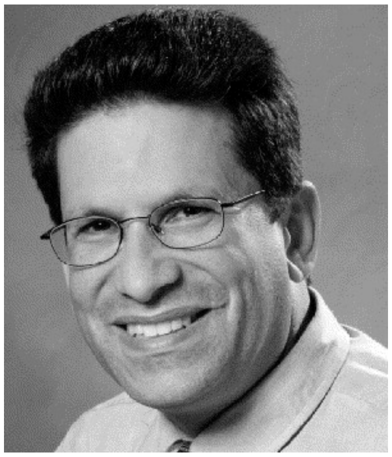
Wilson, 1972 D.L. Wilson **Asymptotic properties of nearest neighbor rules using edited data** IEEE Transactions on Systems, Man and Cybernetics, SMC-2 (1972), pp. 408-421



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.



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. in mechanical engineering from MIT in 1983. He joined the Department of Mechanical and Industrial Engineering at Marquette University (Milwaukee, WI, USA) in 1996 after teaching at Carnegie Mellon University (Pittsburgh, PA, USA) and working as a senior research engineer at the Carnegie Mellon Research Institute. His research interests include automation, mechatronics, control design, and biomechanics.

☆This paper was not presented at any IFAC meeting. This paper was recommended for publication in revised form by Associate Editor K. Uchida under the direction of Editor I. Petersen.