Available online at www.sciencedirect.com
SCIENCE DIRECT®
Pattern Recognition Letters 25 (2004) 725–731
Pattern Recognition Letters
www.elsevier.com/locate/patrec

A vector quantization method for nearest neighbor classifier design

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

[a] Department of Mechanical Engineering, National Sun-Yat Sen University, Kaohsiung 80024, Taiwan
[b] Department of Mechanical and Industrial Engineering, Marquette University, Milwaukee, WI 53201-1881, USA

Received 5 September 2002; received in revised form 11 December 2003

Abstract

This paper proposes a nearest neighbor classifier design method based on vector quantization (VQ). By investigating the error distribution pattern of the training set, the VQ technique is applied to generate prototypes incrementally until the desired classification result is reached. Experimental results demonstrate the effectiveness of the method.

Keywords: Vector quantization; Classification; Nearest neighbor classifier; Supervised learning

1. Introduction

As one of the simplest methods for nonparametric classification, the nearest neighbor (NN) classification rule has been applied extensively in many problems. Typically, a large number of prototypes are desirable for the conventional NN classifiers to acquire statistical information. The resulting computational demands often hinder the on-line application of NN classifiers. As a consequence, many methods have been proposed to reduce the size of the prototypes without significantly compromising the classification accuracy. Many of these prototype-editing approaches (e.g., Hart, 1968; Gates, 1972; Chidananda Gowda and Krishna, 1979; Devijver and Kittler, 1980) adopt an iterative process to move training samples in and out of the prototype set and retain a subset of the training samples as the final prototype set. A drawback of this technique is that the classification error must be recomputed every time a sample is moved in or out of the prototype set. Consequently, these methods may be computationally demanding in solving problems with a large number of training samples.

Xie et al. (1993) present a different approach for NN classifier design. Rather than inspect the necessity of each training sample, they examine the training samples as a group of data. Following this idea, Linde et al. (1980) and Gray (1984) construct an optimal vector quantizer for each class of the training data. The resulting codewords (the points

* Corresponding author. Tel.: +886-7-525-2000; fax: +886-7-525-4299.
E-mail address: vincen@mail.nsysu.edu.tw (C.-W. Yen).

0167-8655/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2004.01.012

that are used to represent the training samples) are then chosen as the prototypes of the NN classifier. Experimental results demonstrate that their approach outperforms several previously proposed NN classifier design methods. However, since the codewords are computed independently for each class of the training data, this method does not consider the interaction among different classes of samples. Another drawback is that when expansion of the prototype set is required for a better classification result, the number of prototypes must be increased by a factor of two. As a result, it is often difficult to use this method to design an NN classifier that has an optimal number of prototypes.

This paper is based on the vector quantization (VQ) technique, and introduces an NN classifier design method that expands the prototype set on a one-at-a-time basis. During this progressive prototype generation process, attention is focused on the interaction between the codewords associated with different classes, with new codewords placed at the region that has the largest classification error. In the following section, the basic idea of the VQ technique is illustrated.

2. Preliminaries

2.1. The NN classifier

The classification problem considered consists of a finite set of training samples $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_I\}$ with every sample $\boldsymbol{x}_i = \{x_{i1}, x_{i2}, \ldots, x_{ir}\}$ being an r-dimensional vector where r is the number of feature variables and I is the number of the training samples. For every sample $\boldsymbol{x}_i$, its real class $C_i \in \boldsymbol{C}$ is known, where $\boldsymbol{C} = \{C_1, C_2, \ldots, C_M\}$ represents a set of M classes. Based on the information provided by $\boldsymbol{X}$, class C_M is represented by a set of N_M prototypes. To determine the class of a tested sample, the NN classifier finds the closest prototype neighbor. The class of this nearest prototype is then chosen as the class of the tested sample. Therefore, the key to the NN classifier design process is finding appropriate prototypes by using information contained in training set $\boldsymbol{X}$.

2.2. VQ and the VQ-NN classifier

In similarity to scalar quantization where a large set of numbers is mapped to a smaller one, the VQ method quantizes groups of numbers together rather than addressing them one at a time. In data compression problems, groups of numbers are input vectors and quantization levels are codeword (or reproduction) vectors. In particular, given a set $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_I\}$ of input vectors, with $M < I$, a set $\boldsymbol{W} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_M\}$ is chosen in VQ to quantize $\boldsymbol{X}$. To transmit any input vector $\boldsymbol{x}_i$ over communication channels, the sender determines the nearest codeword vector $\boldsymbol{w}_j$ according to some distance measure $d(\boldsymbol{x}_i, \boldsymbol{w}_j)$ and transmits the index j of this vector. The distance measure used in this paper is the Euclidean distance:

$$d(\boldsymbol{a}, \boldsymbol{b}) = \sum_{i=1}^{n} (a_i - b_i)^2 \tag{1}$$

for $\boldsymbol{a}, \boldsymbol{b} \in R^n$.

Based on this technique, Xie et al. (1993) proposed the VQ-NN classifier. By treating training samples as input vectors, training samples in each class are quantized independently by the Linde-Buzo-Gray (LBG) method (Linde et al., 1980). The resulting codewords are then chosen as the prototypes for the VQ-NN classifier. Drawbacks of this design method are: (i) it does not consider interaction among classes, and (ii) it is inflexible in determining the number of prototypes. These two issues are addressed in the following section.

3. Proposed method

3.1. Problems to be solved

In developing an incremental prototype set building method, three problems need to be solved. The first problem is how to generate an initial prototype set. The second problem is how to increase the number of prototypes one-at-a-time for better classification results. The last problem is when to stop the prototype set building process. To resolve the first problem, the proposed approach, based on the concept of VQ, generates a

single prototype independently for each class. Specifically, the initial prototype is chosen as the centroid of the training samples associated with each class.

To resolve the second problem, the classification error associated with an NN classifier is investigated. Consider the following Bayes decision rule which assigns training sample $\boldsymbol{x}$ to class C_b if

$$p(\boldsymbol{x}|C_b)P(C_b) \geqslant p(\boldsymbol{x}|C_l)P(C_l) \quad \text{for all } l \neq b \tag{2}$$

where $p(\boldsymbol{x}|C_b)$ and $P(C_b)$ are the conditional and a priori probabilities of class C_b, respectively. In this paper, the class membership b determined by Eq. (2) is called the Bayes class membership. For the training set, an edited NN rule can have the same classification result as the Bayes classifier if the class-conditional probabilities of its prototype set satisfy

$$q(\boldsymbol{x}|C_b)P(C_b) \geqslant q(\boldsymbol{x}|C_l)P(C_l) \quad \text{for all } l \neq b \tag{3}$$

for every training sample. Note that this does not require $q(\boldsymbol{x}|C_i) = p(\boldsymbol{x}|C_i)$ for all classes C_i.

Based on the Bayes decision rule, there can be only two reasons for an NN classifier to misclassify a sample. The first reason is that the actual class membership of the sample is not the same as the Bayes class membership determined by Eq. (2). As a result, such a sample cannot be classified correctly by the Bayes decision rule. This occurs when

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

with a denoting the actual class membership of sample $\boldsymbol{x}$. Forcing such training samples to be classified correctly violates the Bayes decision rule and can actually degrade the generalization accuracy of the classifier. The second reason for the misclassification is that the classifier, in its current form, cannot satisfy Eq. (3). Consequently,

$$q(\boldsymbol{x}|C_r)P(C_r) > q(\boldsymbol{x}|C_b)P(C_b) \quad \text{for some } r \neq b. \tag{5}$$

For convenience, these two types of errors are referred to as the Bayes and classifier errors, respectively, in the remaining part of the paper.

3.2. Procedure for incremental prototype set building

One of the fundamental requirements for an acceptable classifier is reduction of the classifier error to the extent possible. By assuming that a number of prototypes have already been found for the classification problem, the proposed approach uses the following procedure to resolve the problem of incremental prototype set building.

(1) With $\boldsymbol{w}_{i,j}$ denoting the jth prototype of the ith class, perform clustering independently for each class of the training set by assigning every training sample to the nearest prototype of the same class.
(2) Compute the number of ith class training samples whose closest ith class prototype is $\boldsymbol{w}_{i,j}$. Denote this number as $R_{i,j}$.
(3) Use the NN decision rule to classify the training set by associating every sample to its nearest prototype regardless of the class distinction.
(4) Compute the number of ith class training samples whose closest prototype is $\boldsymbol{w}_{i,j}$. Denote this number as $Q_{i,j}$.
(5) Denote the difference between $R_{i,j}$ and $Q_{i,j}$ as $E_{i,j}$, i.e.,

$$E_{i,j} = R_{i,j} - Q_{i,j} \tag{6}$$

(6) Denote the prototype that has the largest value of $E_{i,j}$ as $\boldsymbol{w}^*$ and find the training samples that were assigned to $\boldsymbol{w}^*$ in the clustering process. These training samples are then divided into two clusters by using the LBG method.
(7) Increase the number of prototypes by one by replacing $\boldsymbol{w}^*$ with the two newly generated cluster centers.

The rationale behind this procedure is explained as follows. To correctly classify a training sample, this sample and its nearest prototype should belong to the same class. In contrast, a nonzero $E_{i,j}$ indicates that some of the ith class training samples that were assigned to $\boldsymbol{w}_{i,j}$ in the clustering process of step 1 were "taken away" in the classification process of step 3 by a prototype that has a different class membership. Apparently, for each

of these incorrectly classified samples, there exists at least one r that satisfies the following inequality

$$q(\boldsymbol{x}|C_r)P(C_r) > q(\boldsymbol{x}|C_a)P(C_a). \quad (7)$$

To correctly classify these samples, a possible solution is to make the corresponding $q(\boldsymbol{x}|C_a)$ larger. As shown by Lofsgaarden and Quesenbery (1965), with N observations of a random variable $\boldsymbol{X}$, the estimate of the probability density function $p(\boldsymbol{X})$ is inversely proportional to $V(K,N,\boldsymbol{X})$, which is the smallest hypervolume that encloses all the points at least as near to $\boldsymbol{X}$ as the Kth nearest neighbor of $\boldsymbol{X}$. Hence, step 6 of the procedure splits the cluster that has the largest $E_{i,j}$ (and thus has the largest number of incorrectly classified training samples among all clusters) into two subclusters. This step can reduce the distances between these incorrectly classified samples and their cluster centers and thus reduces the corresponding $V(K,N,\boldsymbol{X})$. As a result, the corresponding $q(\boldsymbol{x}|C_a)$ is made larger and the new cluster centers can recover these samples, which were incorrectly "taken away" by a prototype that has a different class membership.

After increasing $q(\boldsymbol{x}|C_a)$ for the training samples of the cluster that has the largest $E_{i,j}$, the class membership property of some of the samples may change from Eq. (7) to

$$q(\boldsymbol{x}|C_a)P(C_a) > q(\boldsymbol{x}|C_l)P(C_l) \quad \text{for all } l \neq a. \quad (8)$$

The classification accuracy for the training set can thus be improved.

A potential problem of this cluster splitting technique may occur in dealing with any sample that has different actual and Bayes class memberships. For these samples, based on Eq. (2),

$$p(\boldsymbol{x}|C_b)P(C_b) > p(\boldsymbol{x}|C_a)P(C_a) \quad (9)$$

and therefore for such a sample, the classifier should have

$$q(\boldsymbol{x}|C_b)P(C_b) > q(\boldsymbol{x}|C_a)P(C_a). \quad (10)$$

However, this contradicts Eq. (8) for samples that have different actual and Bayes class memberships. To avoid such a problem, one might suggest that the classifier satisfy Eq. (3) instead of Eq. (8), but this is not possible since the Bayes class membership for the training set is typically unknown.

The problem with Eq. (8) occurs because the proposed approach will not only reduce the classifier error but also the Bayes error. As a result, the generalization accuracy of the classifier can thus be degraded. Such a phenomenon is very similar to the overfitting problem encountered with neural networks. To resolve this difficulty, the early stopping method, which is often used in neural network training, is employed (Haykin, 1999). To apply the early stopping method, samples are first divided into training and validation sets. The classifier is then designed by minimizing the classification error associated with the training set. To prevent overfitting, the classification error associated with the validation set is also monitored. The training process is terminated when the classification error of the validation set fails to improve for a prespecified number of iterations.

3.3. Approach with early stopping technique

With the early stopping technique, the proposed approach can be summarized as follows:

(1) Divide samples with known actual class membership into training and validation sets.
(2) Let $\boldsymbol{w}_{i,1}$ be the centroid of the training samples of the ith class. This generates an initial prototype for each class. As such, each class generates a single cluster in the beginning of the training process.
(3) Compute $E_{i,j}$, i.e., the number of incorrectly classified samples associated with the jth cluster of class i.
(4) Find the largest $E_{i,j}$ and the corresponding cluster center $\boldsymbol{w}^*$.
(5) Apply the LBG method to divide the samples that were assigned to $\boldsymbol{w}^*$ in the clustering process into two sub-clusters.
(6) Increase the number of prototypes by one by replacing $\boldsymbol{w}^*$ with the two cluster centers generated in the previous step.
(7) With the updated prototype set, compute the classification error for the validation set.
(8) Terminate the prototype set building process if the classification error for the validation set fails to improve for a prespecified number of

iterations. Otherwise, this process continues from (3).

Finally, the early stopping method provides an answer for the third problem in that it provides a mechanism to automatically terminate the incremental prototype set building process.

4. Experimental results

Experimental results given by Xie et al. (1993) have shown that the VQ-NN classifier outperforms several traditional NN classifier design methods, including the CNN (condensed nearest neighbor; Hart, 1968), the RNN (reduced nearest neighbor; Gates, 1972) and the ENN (edited nearest neighbor; Devijver and Kittler, 1980) in terms of both the prototype reduction rate and the classification accuracy. Therefore, one of the goals here is to compare the proposed approach with the VQ-NN in solving several benchmark problems. Due to the adaptive nature of its prototype generation, the proposed method will be referred to as the *AVQ-NN method* hereafter.

With the unedited training set, the uncondensed nearest neighbor (UNN) rule and the well-known CNN method were also implemented. In testing these methods, a 10-fold cross-validation process was used. In the results below, the average number of prototypes and the average validation accuracy are reported, where the validation accuracy is defined as the percentage of the correctly classified samples in the validation set.

4.1. Smaller data sets

This subsection considers test data sets with a relatively small number of samples:

(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 was benign or malignant. After testing 34 classification methods, the best 10-fold cross-validation classification accuracy obtained by the Statlog project (Hichie et al., 1994) is 97.2% (www.phys.uni.torun.pl/kmk/projects/datasets.html).

(2) Australian credit card data: The goal of this data set, used by the Statlog project, was to assess applications for credit cards based on 14 attributes and involves 690 samples in total. After solving this problem with 27 classification methods, the best 10-fold cross-validation classification accuracy obtained by the Statlog project is 86.9% (www.phys.uni.torun.pl/kmk/projects/datasets-stat.html).

(3) Diabetic data: Based on eight features, the objective of this problem is to determine whether a person is diabetic. This problem includes 768 examples. The data set was also obtained from the UCI repository. After testing 25 classification methods, the best 10-fold cross-validation classification accuracy obtained by the Statlog project is 77.7% (www.phys.uni.torun.pl/kmk/projects/datasets. html).

In developing the VQ-NN classifier for each of these three problems, the number of prototypes was increased by a factor of 2 gradually from 2 to 128. The number of prototypes yielding the smallest validation error was then selected. In addition, in solving these three problems the early stopping technique terminated the AVQ-NN method when the classification error for a validation set failed to improve for 10 iterations.

The classification results for the three problems are summarized in Tables 1–3, respectively, and demonstrate the accuracy of the proposed AVQ-NN approach. The VQ-NN method has the second lowest classification error, followed by the CNN and finally the UNN methods. For all three

Table 1
Summary of the classification result for the Wisconsin breast cancer problem (best Statlog project validation accuracy: 97.2%)

Method	Accuracy (%)	Number of prototypes
UNN	95.7	629
CNN	96.3	27.3
VQ-NN	96.3	2
AVQ-NN	97.7	4.7

Table 2
Summary of the classification result for the Australian credit card problem (best Statlog project validation accuracy: 86.9%)

Method	Accuracy (%)	Number of prototypes
UNN	79.4	621
CNN	83.4	72.5
VQ-NN	86.4	4
AVQ-NN	88.8	3.7

Table 3
Summary of the classification result for the diabetes problem (best Statlog project validation accuracy: 77.7%)

Method	Accuracy (%)	Number of prototypes
UNN	69.3	691
CNN	72.7	114.9
VQ-NN	73.8	4
AVQ-NN	78.0	6.4

problems, the AVQ-NN method yields lower classification errors than the best results obtained by the Statlog project, which comprehensively tested many different classification methods, including a back-propagation (BP) method, a learning vector quantization (LVQ) classifier, a support vector machine (SVM) method as well as a radial basis function (RBF) neural classifier.

The results shown in the tables also demonstrate the potential of the VQ technique in finding an efficient set of prototypes to represent training samples. Specifically, the prototype numbers associated with the two VQ-based methods are significantly smaller than those required by the CNN and UNN methods. The VQ-NN method has the least number of prototypes in two of the three tested problems. However, under the limitation of at most 128 prototypes, the VQ-NN method cannot find a better classification result by using more prototypes for these problems. For the diabetes problem, with 8, 16, and 32 prototypes, the classification accuracy of the VQ-NN method is 72.0%, 70.0% and 68.5%, respectively. In contrast to the inflexible prototype expansion procedure of the VQ-NN, the AVQ-NN method increases the prototypes one-at-a-time, and is therefore able to find better classification results with a few more prototypes.

4.2. Larger data sets

In this subsection, two examples with a relatively large number of training samples are considered.

(4) Phoneme data: This data set is adapted from the ELENA project. (The databases used by this project and a technical report describing them are available via anonymous ftp at ftp.dice.ucl.ac.be in the directory pub/neural-nets/ELENA/databases.) The aim of this problem is to distinguish between nasal (3818 samples) and oral (1586 samples) vowels. Each sample is represented as a five-dimensional vector.
(5) Kr-vs-Kp data: This chess endgame database was also obtained from the UCI repository. This problem consists of two classes, which contain 1669 and 1527 samples, respectively. The dimension of the feature vector is 36.

Given the larger size of the training sets, the maximum allowable number of prototypes for the VQ-NN method was set at 1024. As shown in Table 4, unlike the previous three examples, the two VQ based methods fail to provide better classification accuracy than the UNN and CNN methods in dealing with the Phoneme problem. The question then is why the proposed approach fails to outperform the UNN and the CNN methods in this problem.

The goal of a VQ method is to represent a large number of samples with a relatively small number of prototypes. This goal is easy to achieve if the samples are concentrated in a few clusters. However, if the samples are distributed in a large number of disjointed regions far apart from one another, then the number of prototypes required to achieve a sufficiently small level of distortion

Table 4
Summary of the classification result for the Phoneme problem

Method	Accuracy (%)	Number of prototypes
UNN	90.5	4864
CNN	86.1	545
VQ-NN	81.0	128
AVQ-NN	85.4	199

Table 5
Summary of the classification result for the Kr-vs-Kp problem

Method	Accuracy (%)	Number of prototypes
UNN	90.3	2876
CNN	87.2	282
VQ-NN	94.0	256
AVQ-NN	94.6	157

may become very large. The two VQ-based methods fail to achieve better classification accuracy than the UNN method in this problem since the vowels of the training set were taken from 1809 isolated syllables. As a result, in the five-dimensional feature, it is very likely that these 5404 samples of the Phoneme data set are distributed in a large number of disjointed clusters. Nevertheless, compared with the VQ-NN method, the proposed approach still provides better classification accuracy and its prototype requirement is also much less than those of the UNN and CNN methods.

Finally, the results for the Kr-vs-Kp problem, whose training set is of comparable size as that of the Phoneme problem, are summarized in Table 5. In similarity to the results shown in the first three tables, the two VQ-based methods again provide better classification accuracy with smaller number of prototypes. In addition, the AVQ-NN method outperforms the VQ-NN method since it reduces the number of prototypes by more than 38% and provides slightly better classification accuracy.

5. Conclusions

Based on the concept of vector quantization, this paper proposes a prototype set building technique for the design of a NN classifier. In addition to the application of the VQ technique, a distinct feature of the proposed approach is that the approach expands the prototype set on a one-at-a-time basis with an error-guided procedure.

The effectiveness of the method was tested using five data sets. The results shows that the proposed approach can achieve high classification accuracy with a relatively small number of prototypes. To fully investigate the potential of the method, more comprehensive experiments can be performed. In addition, the approach can be compared with other nonparametric classifiers such as neural networks. Another possible future direction is to investigate the sensitivity of the proposed method to the training set size.

Acknowledgements

This research was supported in part by National Science Council of Republic of China under grant number NSC 89-2212-E110-020.

References

Chidananda Gowda, K., Krishna, G., 1979. The condensed nearest neighbor rule using the concept of mutual nearest neighborhood. IEEE Trans. Inform. Theory 25, 488–490.

Devijver, P.A., Kittler, J., 1980. On the edited nearest neighbor rule. Proc. 5th Internat. Conf. Pattern Recognition, 72–80.

Gates, G.W., 1972. The reduced nearest neighbor rule. IEEE Trans. Inform. Theory 8, 431–433.

Gray, R.M., 1984. Vector quantization. IEEE ASSP Mag. 1, 4–29.

Hart, P.E., 1968. The condensed nearest neighbor rule. IEEE Trans. Inform. Theory 4, 515–516.

Haykin, S., 1999. Neural Networks: A Comprehensive Foundation. Prentice Hall, New Jersey. pp. 215–217.

Hichie, D., Spiegelhalter, D.J., Taylor, C.C., 1994. Machine Learning, Neural and Statistical Classification. Ellis Horwood, New York.

Linde, Y., Buzo, A., Gray, R.M., 1980. An algorithm for vector quantizer design. IEEE Trans. Commun. 28, 84–95.

Lofsgaarden, D.O., Quesenbery, C.P., 1965. A nonparametric estimate of a multivariate density function. Ann. Math. Stat. 36, 1049–1051.

Xie, Q., Laszlo, C.A., Ward, R.K., 1993. Vector quantization technique for nonparametric classifier design. IEEE Trans. Pattern Anal. Machine Intell. 15, 1326–1330.