Comparison of Two Different PNN Training Approaches for Satellite Cloud Data Classification

Bin Tian and Mahmood R. Azimi-Sadjadi

***Abstract*—This paper presents a training algorithm for probabilistic neural networks (PNNs) using the minimum classification error (MCE) criterion. A comparison is made between the MCE training scheme and the widely used maximum likelihood (ML) learning on a cloud classification problem using satellite imagery data.**

***Index Terms*—Cloud classification, maximum likelihood, minimum classification error, probabilistic neural network.**

I. Introduction

Probabilistic neural network (PNN) is a kind of supervised neural network that are widely used in the area of pattern recognition, nonlinear mapping, and estimation of probability of class membership and likelihood ratios. The original PNN structure [1], is a direct neural-network implementation of the Parzen nonparametric probability density function (PDF) estimation [2] and Bayes classification rule. Although its training scheme is very simple and fast, one major drawback is that potentially a very large network will be formed since every training pattern needs to be stored. This leads to increased storage and computational time requirements during the testing phase. One natural idea to simplify the PNN is to reduce the number of neurons, i.e., use fewer kernels but place them at optimal places. In [3] Streit *et al.* improved the PNN by using finite Gaussian mixture models and maximum likelihood (ML) training scheme. However, the ML-based training does not necessarily lead to a minimum error performance for the classifier. This may be due to the fact that the Gaussian mixture model may not be an accurate assumption for some of the feature space distribution and the training data set is often inadequate. In [4], Juang *et al.* proposed a new learning scheme based upon the minimum classification error (MCE) criterion. In [5], Gish pointed out that minimization of the number of errors is not the only benefit of the MCE. The MCE criterion is also inherently robust. The robustness stems from its counting misclassifications and ignoring the magnitude of the error, i.e., ignoring how far the misclassified events are from the decision boundary. Owing to its robustness, MCE has been widely used in speech recognition applications [6], [7].

In this study, ML and MCE are used to estimate the parameter sets of the Gaussian mixture model. Their performances are examined on the Geostationary Operational Environmental Satellite (GOES)-8 imagery data for cloud classification. The organization of this paper is as follows. Section II briefly introduces the Gaussian mixture model. In Sections III and IV, ML, and MCE training schemes for PNN are discussed separately. Comparisons of these training algorithms are presented in Section V.

Manuscript received September 9, 1999; revised May 15, 2000. This work was supported by the DoD Center for Geosciences/Atmospheric Research (CG/AR) under Contract DAAL01-98-2-0078.

The authors are with the Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523-1373 USA.

Publisher Item Identifier S 1045-9227(01)00536-7.

II. Gaussian Mixture Model

Consider a d-dimensional input feature vector $\mathbf{x}$ which belongs to one of the K classes, $c_i, i = 1, 2, \ldots, K$. A classifier can be regarded as a mapping, $C: R^d \rightarrow \{c_1, c_2 \ldots, c_K\}$ that classifies the given pattern $\mathbf{x}$ to class $C(\mathbf{x})$. Suppose that the class conditional distribution, $p(\mathbf{x} \mid c_i)$, and the *a priori* class probability $P(c_i)$ are known, then the best classifier is given by the fundamental Bayes decision rule

$$C(\mathbf{x}) = \arg\max_{c_i} P(c_i) p(\mathbf{x} \mid c_i) \quad i = 1, 2, \ldots K. \tag{1}$$

One main concern when implementing the above optimal Bayes classifier is to estimate $p(\mathbf{x} \mid c_i)$ and $P(c_i)$ from the training data set. Generally, $P(c_i)$ is highly dependent on the specific task and should be decided by the physical knowledge of the problem. For the sake of convenience, uniform distribution assumption for $P(c_i)$ is adopted in this study. Also, for any class c_i, we assume that the $p(\mathbf{x} \mid c_i)$ can be represented by a Gaussian mixture model, i.e,

$$p(\mathbf{x} \mid c_i) = \sum_{j=1}^{M_i} \pi_{ji} p_{ji}(\mathbf{x}; \mu_{ji}, \Sigma_{ji}) \tag{2}$$

where M_i is the number of Gaussian components in class c_i and π_{ji}'s are the weights of the components which satisfy the constraint $\sum_{j=1}^{M_i} \pi_{ji} = 1$, $p_{ji}(\mathbf{x}; \mu_{ji}, \Sigma_{ji})$denotes the multivariate Gaussian density function of the jth component in class c_i and μ_{ji} and Σ_{ji} are its mean vector and covariance matrix, respectively. This Gaussian mixture model can be easily mapped to the PNN structure and the resultant PNN will need much fewer neurons. The price paid for this simplification is that the simple noniterative training procedure will no longer be applicable. Instead, the weights of the PNN, i.e., the parameter sets of the mixture model for each class, need to be estimated from the training data set.

III. Maximum Likelihood Training for PNN

Let $\lambda_i = \{\pi_{ji}, \mu_{ji}, \Sigma_{ji}\}_{j=1}^{M_i}$ denote the parameter set used to describe the mixture model of class c_i and $\Lambda = \{\lambda_i\}_{i=1}^{K}$ denote the whole parameter space for the PNN. The goal of training is to estimate the parameter space Λ from the training set. If we assume that the parameters in Λ are unknown fixed quantities, the ML estimation method is a suitable choice.

Now suppose that the training samples drawn independently from the feature space form the set T, which can be further separated into K subsets T_i, $i = 1, \ldots, K$, in which all the samples belong to class c_i. The ML estimation of parameter set Λ is then given by

$$\Lambda^* = \arg\max_{\Lambda} \prod_{i=1}^{K} \prod_{\mathbf{x} \in T_i} p(\mathbf{x} \,|\, c_i; \Lambda). \tag{3}$$

For the computational efficiency, generally we will maximize the equivalent log-likelihood, i.e.,

$$\begin{aligned} \Lambda^* &= \arg\max_{\Lambda} \sum_{i=1}^{K} \sum_{\mathbf{x} \in T_i} \log[p(\mathbf{x} \,|\, c_i; \Lambda)] \\ &= \arg\max_{\Lambda} \sum_{i=1}^{K} \sum_{\mathbf{x} \in T_i} \log[p(\mathbf{x} \,|\, c_i; \lambda_i)]. \end{aligned} \tag{4}$$

The last step in (4) is arrived at based upon the assumption that the conditional probability of class c_i is solely decided by the parameter set of that class, λ_i and not by the parameter set of the other classes. The maximization of the log-likelihood function can be done using a probability gradient descent (PGD) scheme [8]. Let

$$F(T; \Lambda) = \sum_{i=1}^{K} \sum_{\mathbf{x} \in T_i} \log[p(\mathbf{x} \,|\, c_i; \lambda_i)] \tag{5}$$

denote the log likelihood function and take the partial derivative of this function with respect to each parameter in Λ, then we have

$$\frac{\partial F(T; \Lambda)}{\partial a_{ji}} = \sum_{\mathbf{x} \in T_i} \frac{\partial \log p(\mathbf{x} \,|\, c_i; \lambda_i)}{\partial a_{ji}} \qquad i = 1, \ldots, K; \quad j = 1, \ldots, M_i \tag{6}$$

where a_{ji} represents either π_{ji}, μ_{ji}, or Σ_{ji}^{-1}. Based on (5) and (6), we can further get

$$\begin{aligned} \frac{\partial \log[p(\mathbf{x} \,|\, c_i; \lambda_i)]}{\partial \pi_{ji}} &= \frac{p_{ji}(\mathbf{x}; \mu_{ji}, \Sigma_{ji})}{\sum_{m=1}^{M_i} p_{mi}(\mathbf{x}; \mu_{mi}, \Sigma_{mi}) \pi_{mi}} \\ \frac{\partial \log[p(\mathbf{x} \,|\, c_i; \lambda_i)]}{\partial \mu_{ji}} &= \frac{p_{ji}(\mathbf{x}; \mu_{ji}, \Sigma_{ji}) \pi_{ji}}{\sum_{m=1}^{M_i} p_{mi}(\mathbf{x}; \mu_{mi}, \Sigma_{mi}) \pi_{mi}} \\ &\quad \times \Sigma_{ji}^{-1} (\mathbf{x} - \mu_{ji}) \\ \frac{\partial \log[p(\mathbf{x} \,|\, c_i; \lambda_i)]}{\partial \Sigma_{ji}^{-1}} &= \frac{p_{ji}(\mathbf{x}; \mu_{ji}, \Sigma_{ji}) \pi_{ji}}{\sum_{m=1}^{M_i} p_{mi}(\mathbf{x}; \mu_{mi}, \Sigma_{mi}) \pi_{mi}} \\ &\quad \cdot \frac{1}{2} [\Sigma_{ji} - (\mathbf{x} - \mu_{ji})(\mathbf{x} - \mu_{ji})^t]. \end{aligned} \tag{7}$$

Based on the PGD scheme, the log-likelihood function can be maximized using the following training equations:

$$\begin{aligned} \pi_{ji}^{\text{new}} &= \pi_{ji}^{\text{old}} + \alpha_1 \frac{\partial F(T; \Lambda^{\text{old}})}{\partial \pi_{ji}} \\ \mu_{ji}^{\text{new}} &= \mu_{ji}^{\text{old}} + \alpha_2 \frac{\partial F(T; \Lambda^{\text{old}})}{\partial \mu_{ji}} \\ \Sigma_{ji}^{-1\,\text{new}} &= \Sigma_{ji}^{-1\,\text{old}} + \alpha_3 \frac{\partial F(T; \Lambda^{\text{old}})}{\partial \Sigma_{ji}^{-1}} \end{aligned} \tag{8}$$

where α_1, α_2, and α_3 are learning factors with values between zero and one.

There is one important observation from (7) and (8). The updating of parameter set of class c_i is only dependent on the training samples in this class, i.e., the optimization process can be solved separately for each class without considering the effect of the others. This is especially suitable for the cloud classification application since a new cloud type can easily be added to the system without affecting the other classes. Moreover, in the updating process, we have the choice of updating only those classes that are affected by the temporal changes in the cloud features. Another benefit of this property is the reduced training time due to the fact that each class can be trained separately, thus requiring a small number of neurons and training samples.

Due to the nature of the PGD scheme, the PNN training process using (7) and (8) generally needs a lot of iterations before converging to an acceptable result. This incurs expensive cost in the training phase. Fortunately, there is an efficient approach called expectation-maximization (EM) which solves this problem. The reader is referred to [9] for the detail treatment on this topic.

IV. MCE Training for PNN

The main idea behind using ML criterion for the PNN training was to accurately estimate the class conditional probability from the training samples. However, since the number of training samples is limited and the Gaussian mixture assumption may not be correct, the estimated distribution may not be accurate. So the optimal performance of the Bayes classifier may not be reached in practice. If we reexamine the Bayes classification rule in (1), it can be found that the actual value of $p(\mathbf{x} \,|\, c_i)$ is in fact not so important for decision. As long as the conditional probability is larger than the corresponding values for the other classes, the classifier can still make the right decision. Therefore, a natural approach to improve the performance of classifier is to estimate the functions $p(\mathbf{x} \,|\, c_i)$, $i = 1, \ldots, K$, which can most successfully discriminate different classes. This is the basic idea behind the discriminant analysis. In general, the discriminant function can be in any form and may not necessarily relate to the probability, but here we still use the same form of $p(\mathbf{x} \,|\, c_k)$, i.e., Gaussian mixture models.

Consider an input feature vector $\mathbf{x}$ belonging to class c_i. According to the Bayes decision rule in (1), $\mathbf{x}$ will be correctly classified if

$$p(\mathbf{x} \,|\, c_i) P(c_i) > \max_{\substack{k=1,\ldots,K \\ k \neq i}} [p(\mathbf{x} \,|\, c_k) P(c_k)]$$

or

$$p(\mathbf{x}, c_i) > \max_{\substack{k=1,\ldots,K \\ k \neq i}} p(\mathbf{x}, c_k) \tag{9}$$

which also can be rewritten as

$$\frac{p(\mathbf{x}, c_i)}{\max_{\substack{k=1,\ldots,K \\ k \neq i}} p(\mathbf{x}, c_k)} > 1 \quad \text{or}$$

$$\log \left[\frac{p(\mathbf{x}, c_i)}{\max_{\substack{k=1,\ldots,K \\ k \neq i}} p(\mathbf{x}, c_k)} \right] > 0. \tag{10}$$

For the training set X, we can define a cost function

$$E(T;\Lambda) = \sum_{i=1}^{K} \sum_{\mathbf{x}\in T_i} f\left(\log\left(\frac{\max_{\substack{k=1,\ldots,K\\ k\neq i}} p(\mathbf{x}, c_k)}{p(\mathbf{x}, c_i)}\right)\right) \tag{11}$$

where $f(\cdot)$ is a step function. It is very clear that the cost function $E(T;\Lambda)$ is nothing but the count of the number of incorrectly classified samples. Minimizing this cost function thus leads to minimizing the classification error. As a result, this criterion is called MCE criterion [4], [10].

Direct minimization of the cost function in (11) is almost impossible since both the max (maximization) operation and the step function are not differentiable. The maximization operation is used to find the most critical rival class for $\mathbf{x}$. In [4], a function $q(\mathbf{x}, c_i)$ was suggested to approximate the maximization operation

$$q(\mathbf{x}, c_i) = \left[\sum_{\substack{k=1\\ k\neq i}}^{K} p(\mathbf{x}, c_k)^r\right]^{1/r}. \tag{12}$$

For large r, $q(\mathbf{x}, c_i)$ is generally a very good approximation of the $\max_{\substack{k=1,\ldots,K\\ k\neq i}} p(\mathbf{x}, c_k)$ unless several $p(\mathbf{x}, c_k)$s are simultaneously close to or equal to the maximum value. In most of the situations, $r = 3$ or 4 is sufficient [4]. Moreover, we can use the sigmoid function to replace the step function. Sigmoid function can be considered as a smoothed version of step function and is defined as

$$f(y) = \frac{1}{1 + \exp(-sy)} \tag{13}$$

where s is a parameter. Using these approximations, the cost function in (10) becomes

$$E(T;\Lambda) = \sum_{i=1}^{K} \sum_{\mathbf{x}\in T_i} f\left(\log\left(\frac{q(\mathbf{x}, c_i)}{p(\mathbf{x}, c_i)}\right)\right). \tag{14}$$

This cost function in (14) is also called as "smooth count of classification error" since the sigmoid function is used [10].

Now based on the MCE criterion, we want to find the parameter sets of the PNN, Λ, that can minimize the cost function (14). Again we can use the PGD scheme. For the jth Gaussian component in class c_i, we take the derivative of the cost function (14) with respect to the parameters. Using the chain rule and after some manipulations, we can get

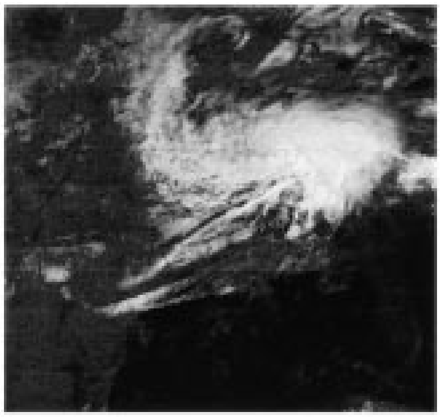
$$\begin{aligned}\frac{\partial E(T;\Lambda)}{\partial a_{ji}} &= \sum_{k=1}^{K} \sum_{\mathbf{x}\in T_k} u(\mathbf{x}; i, k) \frac{\partial \log p(\mathbf{x}, c_i; \lambda_i)}{\partial a_{ji}} \\ &= \sum_{k=1}^{K} \sum_{\mathbf{x}\in T_k} u(\mathbf{x}; i, k) \frac{\partial \log p(\mathbf{x} \mid c_i; \lambda_i)}{\partial a_{ji}} \\ &\qquad i = 1, \ldots, K \quad \text{and} \quad j = 1, \ldots, M_i \end{aligned} \tag{15}$$

where a_{ji} represents either parameter π_{ji}, μ_{ji}, or Σ_{ji}^{-1}. The last step is obtained since the *a priori* class probability, $P(c_i)$, is independent of the parameters a_{ji}s. The results of $(\partial \log p(\mathbf{x} \mid c_i; \lambda_i))/\partial a_{ji}$ for different parameters were given before in (7) in Section II. The function $u(\mathbf{x}; i, k)$ is defined as

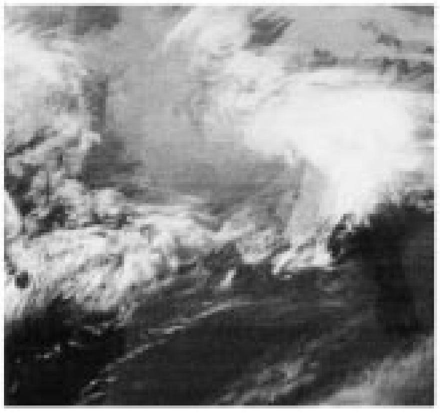
$$u(\mathbf{x}; i, k) = \begin{cases} -f'\left[\log \frac{p(\mathbf{x}, c_i)}{q(\mathbf{x}, c_i)}\right] & i = k \\ -f'\left[\log \frac{p(\mathbf{x}, c_i)}{q(\mathbf{x}, c_i)}\right] \left(\frac{p(\mathbf{x}, c_i)}{q(\mathbf{x}, c_k)}\right)^r & i \neq k \end{cases} \tag{16}$$

Once the derivative is determined, a similar learning rule as in (8) can be used to minimize the cost function in (14).

There are several observations from the training equations in (16). First, the decoupling property **does not** exist for the minimum error criterion. Each training sample contributes to the



(a)



(b)

Fig. 1. GOES 8 satellite images obtained at 15:45 UTC, May 1st, 1995. (a) Visible. (b) IR.

TABLE I
CONFUSION MATRIX FOR THE ML TRAINED PNN. (OVERALL CLASSIFICATION RATE 84.9%)

	Wl	Cl	Ww	Cw	St	Cu	As	Ci	Cs	Sc
Wl	96.1	0.7	0.5	0	0	1	0.2	1.5	0	0
Cl	0.4	91	0	0	0	3.7	0.4	4.5	0	0
Ww	0.4	1.5	95.9	0	0	1.1	0	1.1	0	0
Cw	0	0	13	52.2	0	2.2	0	30.4	0	2.2
St	0	0	0	0	77.6	3.9	12.7	1	0	4.9
Cu	0.2	0	0	0	0.5	86.1	1.7	2.4	0.7	8.3
As	0	0	0	0	2.5	3.2	75.4	4.2	0.4	14.4
Ci	0.5	0.5	0.7	0	0.1	1.2	1.9	86.3	8.2	0.6
Cs	0	0	0	0	0	0.1	1.5	13.2	84.7	0.4
Sc	0	0.2	0	0	2	8.3	14.7	1.4	0.3	73.1

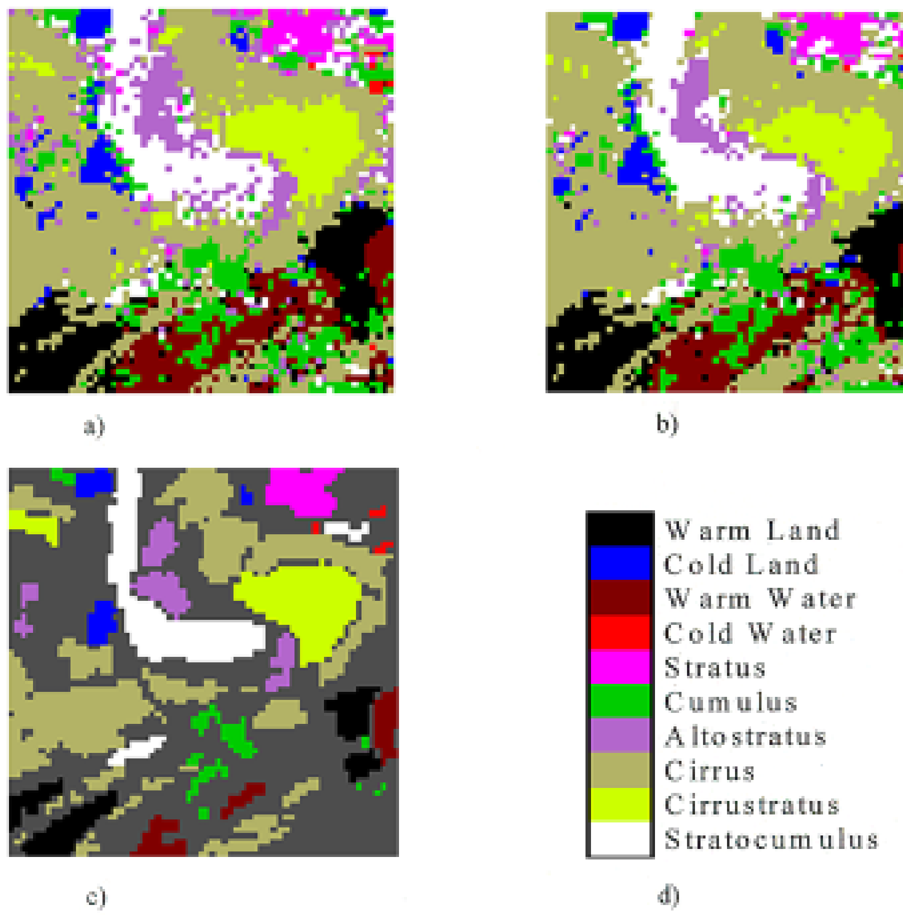


Fig. 2. Comparison of color-coded images. (a) Result of ML trained PNN. (b) Result of MCE trained PNN. (c) Expert labelled Image. (d) Colormap for ten classes.

estimation of parameters of class c_i, no matter whether it belongs to class c_i or not. This is quite different from the situation in the ML training where the parameters of class c_i are only decided by the training samples of that class. Moreover, the contribution of each training sample to the derivative is weighted by $u(\mathbf{x}; i, k)$. The property of this function can be clearly demonstrated in the following cases. Let $\mathbf{x}$ belong to class c_i, and assume $p(\mathbf{x}, c_i) \gg q(\mathbf{x}, c_i)$, i.e, $\mathbf{x}$ can be correctly identified by the current parameter set with confidence, the contribution of this sample to the class c_i will be weighted by $u(\mathbf{x}; i, k) \approx 0$ since $f'(\log(p(\mathbf{x}, c_i)/q(\mathbf{x}, c_i))) \approx 0$ for $(p(\mathbf{x}, c_i)/q(\mathbf{x}, c_i)) \gg 0$. Similarly, if $p(\mathbf{x}, c_i) \ll q(\mathbf{x}, c_i)$, i.e, the input $\mathbf{x}$ is too difficult to be correctly classified, its contribution to the parameter set is also very small. On the other hand, those inputs located on the decision boundary region will lead to comparable $p(\mathbf{x}, c_i)$ and $q(\mathbf{x}, c_i)$ values, thus contribute mostly to the final parameter estimation. Overall, the training results of the MCE criterion are mainly decided by the samples around the decision boundaries formed by the current parameter sets. This is a very distinct characteristic of the MCE training.

Unlike the ML criterion, it is difficult to find an efficient training approach for the MCE criterion to replace the PGD scheme. The PGD solution suffers from several drawbacks. It not only converges very slowly leading to extensive computational cost, but also it is prone to local minimum problem. Based on our experiments, the result of the MCE-PGD training is very sensitive to the initial values of the parameter set. This is partly due to the fact that the MCE training is mainly decided by the distribution of a small subset of the training samples (around the initial boundaries) instead of the whole training set. It is quite common that the subset is not representing well the feature space or its size is too small to lead to any meaningful training result. In order to overcome this initialization problem, in our application we always use the ML trained PNN as the starting point for the MCE-PGD training [4].

V. Results and Discussions

The performance of ML and MCE training algorithm was examined using channel 1 (visible) and channel 4 (IR) of GOES 8 satellite images. One typical image pair obtained at 15:45 universal time code (UTC), May 1st, 1995 is shown in Fig. 1.

After classification, the clouds were separated into ten classes: Warm Land (Wl), Cold Land (Cl), Warm Water (Ww), Cold Water (Cw), Stratus (St), Cumulus (Cu), Altostratus (As), Cirrus (Ci), Cirrostratus (Cs), and Stratocumulus (Sc). Table I presents the classification confusion matrix of the ML training scheme. The numbers located on the diagonal indicate the correct classification rate for each class. The overall classification rate is 84.9%. The color-coded image based on SVD

TABLE II
CONFUSION MATRIX FOR THE MCE TRAINED PNN. (OVERALL CLASSIFICATION RATE 86.9%)

	Wl	Cl	Ww	Cw	St	Cu	As	Ci	Cs	Sc
Wl	97.1	0.7	0.2	0	0	0.7	0	1.2	0	0
Cl	0.4	91.4	0	0	0	3.4	0.4	4.5	0	0
Ww	1.1	1.5	93.7	0	0	1.1	0	2.6	0	0
Cw	0	0	6.5	60.9	0	2.2	0	30.4	0	0
St	0	0	0	0	77.1	3.4	10.7	1	0	7.8
Cu	0.2	0	0	0	0.2	87.1	1.5	2.9	0.2	8
As	0	0	0	0	3.9	2.8	63	5.6	0.4	24.3
Ci	0.5	0.5	0.4	0	0.1	1	0.5	93	3.5	0.6
Cs	0	0	0	0	0	0.1	0.7	23.4	75.1	0.6
Sc	0.2	0	0	0	1.2	7.6	4.7	2.7	0.3	83.3

features [11] and using the ML-based classifier is shown in Fig. 2(a). For the ML training, the EM approach can help to achieve the maximum-likelihood estimation efficiently when the observations can be viewed as incomplete data.

The confusion matrix of the MCE trained PNN is given in Table II. Comparing with the results of the ML-based PNN in Table I, the overall classification rate is improved by 2%, not as dramatic as we expected. This observation indicates that the Gaussian mixture model may in fact be a good representation of the feature space. Among the ten classes, accuracy improvements were observed for six of them, with the exceptions of Warm Water (Ww), Stratus (St), Altostratus (As) and Cirrostratus (Cs). The color-coded classified image is provided in Fig. 2(b). Visual inspection of Fig. 2(b) and (a) reveals that the two images are quite similar except for some minor isolated blocks. Fig. 2(c) and (d) shows the meteorological expert labelled image and the colormap for ten different classes, respectively. Note that in Fig. 2(c) only those areas for which the labeling results of the experts agreed were color coded and used for training and testing of the PNNs.

Overall, this study indicates that the MCE training can provide some improvements in the classification rate when compared with the ML-type training. Clearly, the improvement of MCE training is dependent on the feature space distribution. However, considering that the PGD approach used for the MCE training generally needs much more computational time than that of the EM approach for the ML training, the performance improvements may not be significant enough to justify the additional training cost.

REFERENCES

[1] D. F. Specht, "Probabilistic neural network," *Neural Networks*, vol. 3, pp. 109–118, 1990.

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

[3] R. L. Streit and T. E. Luginbuhl, "Maximum likelihood training of probabilistic neural networks," *IEEE Trans. Neural Networks*, vol. 5, pp. 764–783, 1994.

[4] B. H. Juang and S. Katagiri, "Discriminative learning for minimum error classification," *IEEE Trans. Signal Processing*, vol. 40, pp. 3043–3053, Dec. 1992.

[5] H. Gish, "A minimum classification error, maximum likelihood, neural network," in *Proc. 1992 IEEE Int. Conf. Acoust., Speech, Signal Processing*, vol. 2, pp. 289–292.

[6] R. Chengalvarayan and L. Deng, "Speech trajectory discrimination using minimum classification error learning," in *IEEE Trans. Speech Audio Processing*, vol. 6, pp. 505–515.

[7] D. Rainton and S. Sagayama, "A new minimum error classification training technique for HMM-based speech recognition," presented at the Proc. 3rd Int. Symp. Signal Processing Applicat.. ISSPA-92.

[8] S. Haykin, "Neural networks: A comprehensive foundation," . Englewood Cliffs, NJ: Prentice-Hall, 1994.

[9] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Statist. Soc.*, ser. B, vol. 39, pp. 1–38, 1977.

[10] H. Ney, "On the probabilistic interpretation of neural network classifiers and discriminative training criteria," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, pp. 107–119, 1995.

[11] B. Tian, M. A. Shaikh, M. R. Azimi-Sadjadi, T. H. Vonder-Haar, and D. L. Reinke, "A Study of Clo9ud Classification with Neural Networks Using Spectral and Textural Features," *IEEE Trans. Neural Networks*, vol. 10, pp. 138–151, Jan. 1999.