Pattern classification by stochastic neural network with missing data

Masahiro Tanaka
Okayama University

Yasuaki Kotokawa
Okayama University

Tetsuzo Tanino
Okayama University

Pattern Classification by Stochastic Neural Network with Missing Data

Masahiro Tanaka, Yasuaki Kotokawa and Tetsuzo Tanino
Department of Information Technology, Okayama University
Tsushima-naka 3-1-1, Okayama 700, Japan

ABSTRACT

In this paper, the pattern classification by stochastic neural networks is considered. This model is also termed as Gaussian mixture model. When missing data exist in the training data, it is the usual custom to remove incomplete instants. Here we take another approach, where the missing elements are estimated by using the conditional expectation based on the estimated model by using the EM algorithm. It is shown by using Fisher's Iris data that this approach is superior than removing incomplete data.

1. INTRODUCTION

For pattern classification, various models have been considered in the field of neural networks. Multilayer neural network with sigmoidal activation function is widely used, where the learning is by "back propagation" (Rumelhart et al. [6]).

Gaussian function is also used in neural networks. The so called RBF network is usually adopted for function approximations (e.g. [3]), where Gaussian functions are used as the basis functions.

There is another class of neural networks with Gaussian functions, where the network is used for pattern classification. This is called "stochastic neural network", where the Gaussian functions are used to represent the probability density functions (PDFs) for each pattern class. The simplest way to build stochastic neural networks is to use Parzen window [5].

Gaussian mixture, or also referred to Gaussian sum, has a wide applicability of PDF approximations[7]. The classification is done based on the Bayesian approach, i.e. a vector is judged to belong to the class for which the a posteriori PDF is the largest, i.e.

$$\text{class of } \boldsymbol{x} = \arg \max_{1 \le i \le M} \beta_i g_i(\boldsymbol{x})$$

where $g_i(\boldsymbol{x})$ is the probability density function of $\boldsymbol{x}$ for the class i, and β_i is the prior probability of the class. The parameter estimation scheme was derived by using the EM algorithm [8].

In this paper, we exploit the stochastic representation of the data for the estimation of missing data based on the conditional expectation formula.

In section 2, the model is defined. In section 3, the learning algorithm is shown. In section 4, a numerical example is shown.

2. MODEL

Gaussian mixture model has been often used in statistical estimation problems for more than three decades where non-Gaussian distribution is well approximated and the various estimation schemes developed for the Gaussian model can be applied easily.

Streit and Luginbuhl [8] proposed the model and the learning scheme in this kind of framework.

Let the classes be expressed as $j = 1, \ldots, M$, the problem is to classify n-dimensional real vectors $\boldsymbol{x}$ into M classes.

The Gaussian mixture model for the class $j(= 1, \ldots, M)$ is described by

$$g_j(\boldsymbol{x}|\Theta_j) = \sum_{i=1}^{G_j} \alpha_{ij} p_{ij}(\boldsymbol{x}|\Theta_{ij})$$

$$p_{ij}(\boldsymbol{x}|\Theta_{ij}) = N(\mu_{ij}, \Sigma_{ij}) \qquad (1)$$

$$= (2\pi)^{-n/2} |\Sigma_{ij}|^{-1/2} \exp\left(-\frac{(\boldsymbol{x} - \mu_{ij})^T \Sigma_{ij}^{-1} (\boldsymbol{x} - \mu_{ij})}{2} \right)$$

where

$$\Theta_j = \{\Theta_{1j}, \Theta_{2j}, \ldots, \Theta_{G_j}\}, \quad \Theta_{ij} = \{\mu_{ij}, \Sigma_{ij}\}$$

i denotes the index of the composite Gaussian function and α_{ij} is the weight coefficient.

The learning elements in this model are $\{\alpha_{ij}, \mu_{ij}, \Sigma_{ij}; i = 1, \ldots, G_j, j = 1, \ldots, M\}$.

In the above expression, the following constraints must be satisfied.

$$\alpha_{ij} \ge 0 \quad \forall i, j$$

$$\sum_{i=1}^{G_j} \alpha_{ij} = 1 \quad \forall j$$

0-7803-3280-6/96/$5.00

$\Sigma_{ij} > 0$(positive definite) $\forall i, j$

3. LEARNING

The EM algorithm is even faster than the gradient descent algorithms [4] (the convergence rate is super linear), but the advantage of the usage of EM algorithm is not only that. It has a nice property that the estimated parameters can easily satisfy the constraints that have to be satisfied. These are satisfied automatically, while in the gradient descent, some additional trick is necessary, e.g. penalty for non-feasible values.

3.1 Estimation of Parameters

We derive the learning algorithm based on the EM algorithm. Although the PDF is Gaussian mixture, it is possible to treat that each datum instant is generated from one of the Gaussian sources. This is illustrated in Fig.1.

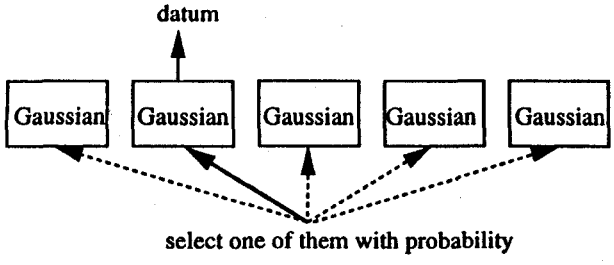

Fig. 1. Generation of Gaussian mixture data

Thus, if we know the source channel of Gaussian signal, the problem is reduced to the Gaussian problem which has been extensively studied so far.

Suppose the channel $i_{k,j}$ has generated $\boldsymbol{x}_{k,j}$. Also,

$$X = \{\{\boldsymbol{x}_{k,j}\}_{k=1}^{T_j}\}_{j=1}^{M}, \quad I = \{\{i_{k,j}\}_{k=1}^{T_j}\}_{j=1}^{M}$$

and further

$$T = \{X, I\}$$

α_l is the a priori probability of the class l. Let $c_{j,l}$ be the cost function where the correct class is l and the judgement class is j. Then the risk $\rho_j(\boldsymbol{x})$ for judging the input $\boldsymbol{x}$ to be the class j is given by

$$\rho_j(\boldsymbol{x}) \sim \sum_{l=1}^{M} c_{j,l}\alpha_l g_l(\boldsymbol{x})$$

Thus the decision to minimize the risk can be written as

$$j = \arg\min_j \rho_j(\boldsymbol{x})$$

3.1.1 *Incomplete Likelihood.* If the channel that generated $\boldsymbol{x}_{kj}$ for the class j is unknown (i.e. the case when I is unknown), the PDF of X given Θ is the usual likelihood, termed "incomplete likelihood" in the EM algorithm, is given by

$$L_u(X;\Theta) = p(X|\Theta) = \prod_{j=1}^{M}\prod_{k=1}^{T_j}\alpha_j g_j(\boldsymbol{x}_{k,j}|\Theta_j)$$
$$= \prod_{j=1}^{M}\prod_{k=1}^{T_j}\alpha_j\left(\sum_{i=1}^{G_j}\alpha_{ij}p_{ij}(\boldsymbol{x}_{kj}|\Theta_{ij})\right)$$

3.1.2 *Complete Likelihood and EM Algorithm.* The EM algorithm proposed by Dempster et al. [1] is a method when there is a many to one mapping from some unobservable intermediate variable to the observation and it is effective when the likelihood function for the intermediate variable is easy to obtain.

From the stochastic formula, we have

$$p(I|X,\Theta) = \frac{p(X,I|\Theta)}{p(X|\Theta)}$$

where $L_u(X;\Theta) = p(X|\Theta)$. Therefore

$$\log L(X;\Theta) = \log p(X,I|\Theta) - \log P(I|X,\Theta)$$

Define

$$Q(\Theta|\Theta') = E\left[\log p(X,I|\Theta)|X,\Theta'\right]$$

From Jensen's inequality

$$E\left[\log P(I|X,\Theta)|X,\Theta'\right] \le E\left[\log P(I|X,\Theta')|X,\Theta'\right]$$

Hence, by selecting Θ such that $Q(\Theta|\Theta') > Q(\Theta'|\Theta')$ holds,

$$E\left[\log L(X;\Theta)|X,\Theta'\right] > E\left[\log L(X;\Theta')|X,\Theta'\right]$$

is guaranteed.

Since

$$p(X,I|\Theta) = P(I|\Theta)p(X|I,\Theta)$$

the complete likelihood is given by

$$p(X,I|\Theta) = \prod_{j=1}^{M}\prod_{k=1}^{T_j}\alpha_j\alpha_{i_{kj},j}p_{i_{kj},j}(\boldsymbol{x}_{kj}|\Theta_{i_{kj},j})$$

The conditional expectation is given by

$$Q(\Theta|\Theta') = E[\log p(X,I|\Theta)|X,\Theta')]$$

Then, since the unspecified stochastic variable is only I, we have

$$Q(\Theta|\Theta') = \sum_I \log p(X,I|\Theta)P(I|X,\Theta')$$

where

$$\sum_I = \sum_{i_{1,1}=1}^{G_1}\cdots\sum_{i_{T_1,1}=1}^{G_1}\sum_{i_{1,2}=1}^{G_2}\cdots\sum_{i_{T_2,2}=1}^{G_2}\cdots$$
$$\cdots\sum_{i_{1,M}=1}^{G_M}\cdots\sum_{i_{T_M,M}=1}^{G_M}$$

By applying the Bayes' rule, we have

$$P(I|X,\Theta) = \frac{p(X|I,\Theta)P(I|\Theta)}{p(X|\Theta)}$$
$$= \prod_{j=1}^{M}\prod_{k=1}^{T_j}\frac{\alpha_j\alpha_{i_{kj},j}p_{i_{kj},j}(\boldsymbol{x}_{kj}|\Theta_{i_{kj},j})}{\alpha_j g_j(\boldsymbol{x}_{kj}|\Theta_j)}$$
$$= \prod_{j=1}^{M}\prod_{k=1}^{T_j}w_{i_{kj},j}(\boldsymbol{x}_{kj})$$

and

$$w_{i_{kj},j}(\boldsymbol{x}_{kj})$$

$$= \frac{\alpha_{i_{kj},j} \exp\left(-[\boldsymbol{x}_{kj} - \boldsymbol{\mu}_{i_{kj},j}]\Sigma_{ij}^{-1}[\boldsymbol{x}_{kj} - \boldsymbol{\mu}_{i_{kj},j}]/2\right)}{\sum_{i=1}^{G_j} \alpha_{ij} \exp\left(-[\boldsymbol{x}_{kj} - \boldsymbol{\mu}_{ij}]\Sigma_{ij}^{-1}[\boldsymbol{x}_{kj} - \boldsymbol{\mu}_{ij}]/2\right)}$$

Note that

$$\sum_I P(I|X,\Theta) = 1$$

and

$$\sum_{I\backslash i_{kj}} P(I|X,\Theta) = w_{i_{kj},j}(\boldsymbol{x}_{kj})$$

Therefore,

$$Q(\Theta|\Theta') = \sum_I \sum_{j=1}^{M} \sum_{k=1}^{T_j} \log\left(\alpha_j \alpha_{i_{kj},j} p_{i_{kj},j}(\boldsymbol{x}_{kj}|\Theta_{i_{kj},j})\right) P(I|X,\Theta')$$

$$= \sum_I \sum_j \sum_k [\log \alpha_j] P(I|X,\Theta')$$

$$+ \sum_I \sum_j \sum_k \left[\log \alpha_{i_{kj},j} p_{i_{kj},j}(\boldsymbol{x}_{kj}|\Theta_{i_{kj},j})\right] P(I|X,\Theta')$$

The first term can be modified as

$$\sum_I \sum_j T_j [\log \beta_j] P(I|X,\Theta') = \sum_j T_j \log \alpha_j$$

and the second one is

$$\sum_I \sum_j \sum_k \log\left(\alpha_{i_{kj},j} p_{i_{kj},j}(\boldsymbol{x}_{kj}|\Theta_{i_{kj},j})\right) P(I|X,\Theta')$$

Separating I into i_{kj} and the other $\sum$'s, we have

$$= \sum_j \sum_k \left[\sum_{i_{kj}=1}^{G_j} \log(\alpha_{ij} p_{ij}(\boldsymbol{x}_{kj}|\Theta_{i_{kj},j}))(\sum_{I\backslash i_{kj}} P(I|X,\Theta'))\right]$$

$$= \sum_j \sum_k \sum_i \log\left(\alpha_{ij} p_{ij}(\boldsymbol{x}_{kj}|\Theta_{ij})\right) w'_{ij}(\boldsymbol{x}_{kj})$$

Hence

$$Q(\Theta|\Theta') = \sum_j T_j \log \beta_j$$

$$+ \sum_j \sum_k \sum_i \log(\alpha_{ij} p_{ij}(\boldsymbol{x}_{kj}|\Theta_{ij})) w'_{ij}(\boldsymbol{x}_{kj})$$

$$= Q_1 + Q_2 + Q_3$$

where

$$Q_1 = \sum_j T_j \log \beta_j$$

$$Q_2 = \sum_{j=1}^{M} \sum_{i=1}^{G_j} \sum_{k=1}^{T_j} w'_{ij}(\boldsymbol{x}_{kj}) \log(\alpha_{ij})$$

$$Q_3 = \sum_j \sum_i \sum_k w'_{ij}(\boldsymbol{x}_{kj}) \log p_{ij}(\boldsymbol{x}_{kj}|\Theta_{ij})$$

The problem is to maximize this with respect to $\Theta = \{\{\beta_j\}, \{\alpha_{ij}, \boldsymbol{\mu}_{ij}\}, \Sigma_{ij}\}\}$.

Based on $Q(\Theta|\Theta')$, we have the following update equations for the parameters.

$$\beta_j = \frac{T_j}{T} \tag{2}$$

$$\alpha_{ij}^{(n+1)} = \frac{\sum_k w_{ij}^{(n)}(\boldsymbol{x}_{kj})}{\sum_i \sum_k w_{ij}(\boldsymbol{x}_{kj})} \tag{3}$$

$$\Sigma^{(n+1)} = \frac{1}{T} \sum_{j=1}^{M} \sum_{i=1}^{G_j} \sum_{k=1}^{T_j} w_{ij}^{(n)} \times$$

$$\times (\boldsymbol{x}_{kj})(\boldsymbol{x}_{kj} - \boldsymbol{\mu}_{ij}^{(n+1)})(\boldsymbol{x}_{kj} - \boldsymbol{\mu}_{ij}^{(n+1)})^T \tag{4}$$

$$\boldsymbol{\mu}_{ij}^{(n+1)} = \frac{\sum_{k=1}^{T_j} w_{ij}^{(n)}(\boldsymbol{x}_{kj}) \boldsymbol{x}_{kj}}{\sum_{k=1}^{T_j} w_{ij}^{(n)}(\boldsymbol{x}_{kj})} \tag{5}$$

$$w_{ij}^{(n)}(\boldsymbol{x}_{kj}) = \frac{\omega_{kij}^{(n)}}{\omega_{kj}^{(n)}} \tag{6}$$

$$\omega_{kij}^{(n)} = \alpha_{ij}^{(n)} \exp[-(\boldsymbol{x}_{kj} - \boldsymbol{\mu}_{ij}^{(n)})^T (\Sigma^{(n)})^{-1} \times$$

$$\times (\boldsymbol{x}_{kj} - \boldsymbol{\mu}_{ij}^{(n)})/2]$$

$$\omega_{kj}^{(n)} = \sum_i \omega_{kij}^{(n)}$$

Here, the technique "pooling" has been adopted, which is to assume that the covariance matrices for all the classes are the same.

3.2 Estimation of Missing Data

It can be found in the commonly used databases (e.g. [3]) that missing data is not a special case but a very common situation. If we delete all the missing data instants, the amount of available data sometimes become very small and the useful information may be lost.

Thus, we restore the missing point based on the estimated model and exploit the partially missing data instants. In the following, how to restore the data is explained.

First the missing elements of the vector $\boldsymbol{x}_{kj}$ are gathered to the top. The transformation matrix is expressed T_{kj}, with which we have

$$T_{kj}\boldsymbol{x}_{kj} = \begin{bmatrix} \boldsymbol{y}_{kj} \\ \cdots \\ \boldsymbol{z}_{kj} \end{bmatrix}$$

where $\boldsymbol{y}_{kj}$ is a vector of missing elements and the remaining are denoted by $\boldsymbol{z}_{kj}$.

Since what we want to know is the estimate of $\boldsymbol{y}_{kj}$, we estimate it by

$$E^{(n)}[\boldsymbol{y}_{kj}|\boldsymbol{z}_{kj}] = \int \boldsymbol{y}_{kj} p^{(n)}(\boldsymbol{y}_{kj}|\boldsymbol{z}_{kj}) d\boldsymbol{y}_{kj} \tag{7}$$

where $E^{(n)}[\cdot]$ means the expectation based on the available model at time n. Since

$$p^{(n)}(\boldsymbol{y}_{kj}|\boldsymbol{z}_{kj}) = \frac{p^{(n)}(\boldsymbol{y}_{kj}, \boldsymbol{z}_{kj})}{p^{(n)}(\boldsymbol{z}_{kj})} = \frac{\sum_i \alpha_{ij}^{(n)} g_i(\boldsymbol{y}_{kj}, \boldsymbol{z}_{kj})}{\sum_i \alpha_{ij}^{(n)} \int g_i(\boldsymbol{y}_{kj}, \boldsymbol{z}_{kj}) d\boldsymbol{y}_{kj}}$$

we have

$$E^{(n)}[y_{kj}|z_{kj}] = \frac{\sum_i \alpha_{ij}^{(n)} \int y_{kj} g_i^{(n)}(y_{kj}, z_{kj}) dy_{kj}}{\sum_i \alpha_{ij}^{(n)} \int g_i^{(n)}(y_{kj}, z_{kj}) dy_{kj}}$$

By some cumbersome matrix manipulations, we have

$$\int g_i^{(n)}(y_{kj}, z_{kj}) dy_{kj} = \frac{1}{(2\pi)^{(n-m)/2}} \frac{1}{|\Lambda_{22}^{(n)}|^{1/2}}$$

$$\times \exp\left\{-\frac{1}{2}\left(z_{kj} - \bar{z}_{kij}^{(n)}\right)^T (\Lambda_{22}^{(n)})^{-1} \left(z_{kj} - \bar{z}_{kij}^{(n)}\right)\right\}$$

and

$$\int y_{kj} g_i^{(n)}(y_{kj}, z_{kj}) dy_{kj} =$$

$$\left(\bar{y}_{kij}^{(n)} + \Lambda_{12}^{(n)}(\Lambda_{22}^{(n)})^{-1}(z_{kj} - \bar{z}_{kij}^{(n)})\right) \frac{1}{(2\pi)^{(n-m)/2}} \frac{1}{|\Lambda_{22}^{(n)}|^{1/2}}$$

$$\times \exp\left\{-\frac{1}{2}\left(z_{kj} - \bar{z}_{kij}^{(n)}\right)^T (\Lambda_{22}^{(n)})^{-1} \left(z_{kj} - \bar{z}_{kij}^{(n)}\right)\right\}$$

where the parameters in the right hand side are the estimated values by the EM algorithm in the current iteration,

$$\begin{pmatrix} \Lambda_{11}^{(n)} & \Lambda_{12}^{(n)} \\ \Lambda_{21}^{(n)} & \Lambda_{22}^{(n)} \end{pmatrix} = T_{kj} \Sigma^{(n)} T_{kj}^T$$

$$\begin{pmatrix} \bar{y}_{kij}^{(n)} \\ \bar{z}_{kij}^{(n)} \end{pmatrix} = T_{kj} \mu_{ij}^{(n)}$$

3.3 Whole Algorithm

Fig. 2 shows the flowchart of the proposed algorithm. The initial model uses random model.

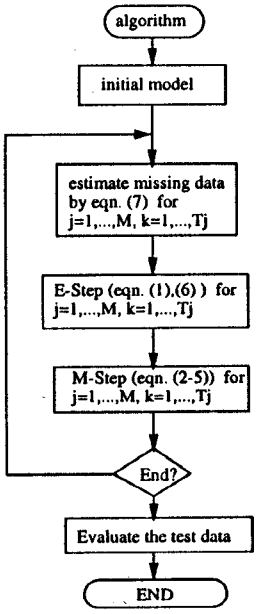

Fig. 2. Flowchart of proposed algorithm

4. NUMERICAL EXAMPLE

The Fisher's iris data [2] is commonly used for testing the classification algorithms. The data of irises are labelled, which are "Iris selosa (1)", "Iris versicolor (2)" and "Iris verginica (3)".

The dataset consists of 4 elements: *Sepal length* (x_1), *Sepal width* (x_2), *Petal length* (x_3) and *Petal width* (x_4).

Table 1 shows some statistics of the data.

TABLE 1 Statistics of original data

	x_1	x_2	x_3	x_4
category	mean value			
1	5.01	3.43	1.46	0.25
2	5.94	2.77	4.26	1.33
3	6.59	2.97	5.55	2.03
	standard deviation			
1	0.35	0.38	0.17	0.10
2	0.51	0.31	0.47	0.20
3	0.63	0.32	0.55	0.27

The contribution of this paper is to simultaneously estimate the unknown system parameters and the missing data. Thus, we compare the results of two cases: one is the approach this paper has proposed (Model A) and the other one is to delete the incomplete data instants and estimate the parameters (Model B).

First 75 (= 25 × 3) instances were selected as the learning vector and same number of instances were for the testing. In each experiment, the performances are compared for the models A and B.

Model A is built with using all the learning data by estimating the missing elements.

Model B is built with data with instances of all the elements existing. By using the model thus obtained, the testing data are evaluated by first interpolating the missing elements and next classifying using the model. This evaluation method is the same for Models A and B. In all the corresponding experiments (e.g.Figs. 3,4), the missing instants were supposed to be the same.

As we can expect that the Model A is generally superior because more information is used, the following results support this hypothesis. In all the following figures, the real lines denote the classification error percentages for the learning vectors, and the dashed lines for the test data.

Figs. 3 and 4 show the misclassification portion for the learning and test data by the model with kernels (1,1,1), respectively for the models A and B, also respectively. Here, about 5% of all the elements were

supposed to be missing.

In the 10% case, 25 instants of data were incomplete, thus, a lot of information has to be lost if partially missing datum is to be removed. Figs. 5 and 6 show the results by using (5,5,5) kernels, and Figs. 7 and 8 show the results for 10% case by (5,5,5) model.

In the 25% case, 50 instants of data were partially missing.Figs. 9 and 10 show the result for this case. The Model B is inappropriate for such very incomplete data. Model B had a numerical problem for computing (6), i.e. the denominator is sometimes 0. This is because, as the learning proceeds with small amount of learning data, the kernel becomes very local (Σ tends to zero matrix) and thus, there is no Gaussian kernel that covers for many instants of test data.

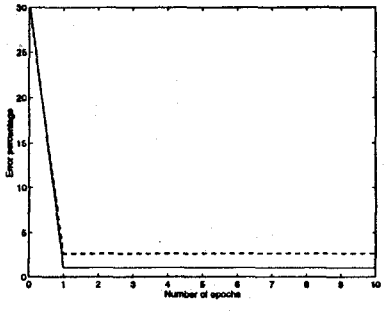

Fig. 3. Model A (1 kernel,5% missing)

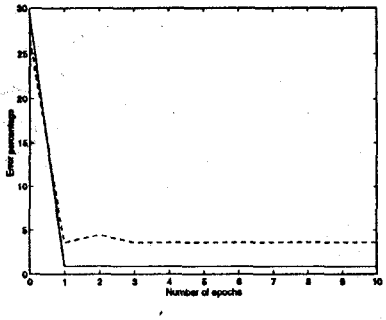

Fig. 4. Model B (1 kernel,5% missing)

It should be noticed that the classification errors are not always monotinically decreasing, rather in many cases, the error rates increase as the learning proceeds. Although it was not shown here, the likelihood function values are always increasing which is the explicit objective function to be maximized.

5. CONCLUSIONS

Since the objective function was the likelihood function, the learning successfully proceeds as the number of epochs increased. But, with respect to the classification result, the ability of the model attains its best property only at the second or third epoch. As far as

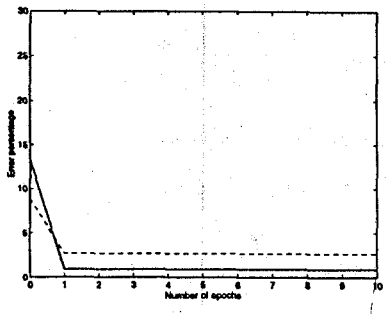

Fig. 5. Model A (5 kernels, 5% missing)

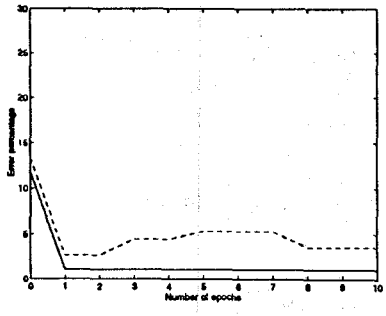

Fig. 6. Model B (5 kernels, 5% missing)

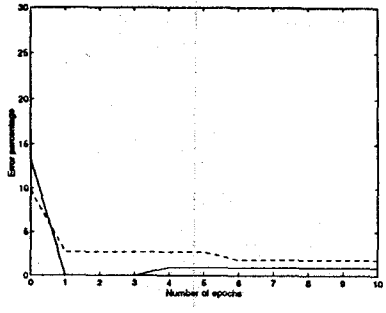

Fig. 7. Model A (5 kernels,10% missing)

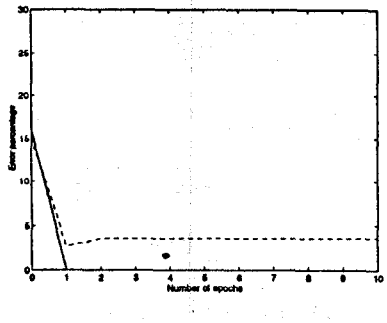

Fig. 8. Model B (5 kernels,10% missing)

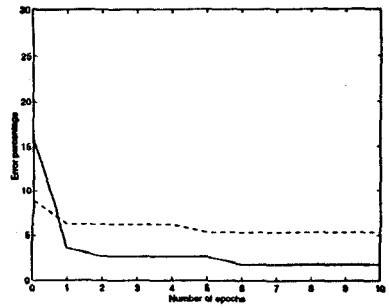

Fig. 9. Model A (3 kernels,25% missing)

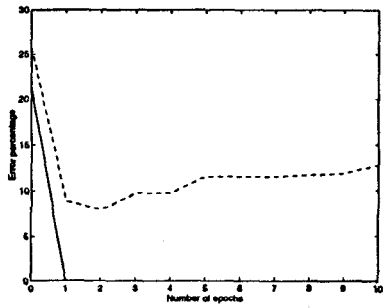

Fig. 10. Model B (3 kernels,25% missing)

we consider the classification problems, this problem has to be extensively studied.

REFERENCES

[1]A. P. Dempster, N. M. Laird and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm", J. Royal Stat. Soc., Vol. B-39, pp. 1-38, 1977.

[2]R. A. Fisher, "The use of multiple measurements in taxonomic problems", Annals of Eugenics, Vol. 7, pp. 179-188, 1936.

[3]J. Moody and C. Darken, "Fast-learning in networks of locally-tuned processing units", Neural Computation, Vol. 1, pp. 281-294, 1989.

[4]P. M. Murphy and D. W. Aha, UCI Repository of machine learning databases [http://www.ics.uci.edu/mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science, 1994.

[5]E. Parzen, "On estimation of a probability density fuction and mode", Annals of Mathematical Statistics, Vol. 33, pp. 1065-1076, 1962.

[6]D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning internal representations by error propagation", in Rumelhart and McCleland eds., Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol.1, MIT Press, 1986.

[7]D. F. Specht, "Probabilistic neural networks", Neural Networks, Vol. 3, pp. 109-118, 1990.

[8]R. L. Streit and T. E. Luginbuhl, "Maximum likelihood training of probabilistic neural networks", IEEE Trans. Neural Networks, Vol. 5, No. 5, pp. 764-783, 1994.

[9]L. Xu and M. I. Jordan, "On convergence properties of the EM algorithm for Gaussian mixtures", Center for Biological and Computational Learning Department of Brain and Cognitive Sciences, paper No. 111, 1995.

TABLE 2 Centers of the kernels

	1	2	3	4	α_{ij}
μ_{11}	4.71	3.10	1.41	0.17	0.29
μ_{21}	5.19	3.69	1.44	0.19	0.30
μ_{31}	5.69	4.15	1.34	0.31	0.09
μ_{41}	5.11	3.54	1.49	0.25	0.13
μ_{51}	4.84	3.32	1.60	0.20	0.19
μ_{12}	6.19	2.20	4.49	1.49	0.04
μ_{22}	5.88	2.71	4.43	1.22	0.28
μ_{32}	6.15	3.14	4.71	1.59	0.15
μ_{42}	5.35	2.58	3.74	1.23	0.26
μ_{52}	6.65	3.00	4.57	1.40	0.26
μ_{13}	7.39	2.76	6.46	2.00	0.20
μ_{23}	7.70	3.80	6.70	2.20	0.04
μ_{33}	6.17	2.69	5.37	1.70	0.12
μ_{43}	5.72	2.78	5.13	2.02	0.25
μ_{53}	6.65	3.08	5.49	2.09	0.39