Modified Self-Organizing Mixture Network for Probability Density Estimation and Classification

Lin Chang, Yu Chong-xiu

Abstract—In this paper, a modified algorithm based on the Self-organizing Mixture Network (SOMN) is proposed to learn arbitrarily complex density functions accurately and effectively. The algorithm is derived based on the criterion of minimizing the Kullback-Leibler divergence, maximum likelihood approach and self-organizing principle. It has the advantages of stochastic approximation method such as fewer local optima and faster convergence speed and the prominent properties of the neural networks such as good generalization ability, and overcomes the limitations of the SOMN. These greatly improve its stability, applicability and computation performance. This algorithm also simplifies the competitive and cooperative mechanism used in the self-organizing map (SOM). This lets it has a well-defined objective function and helps to provide a general proof of convergence. Experiments show that this modified algorithm outperforms the Expectation-Maximization (EM) algorithm, the SOMN and the joint entropy maximization algorithm in estimation accuracy. It is far superior to the EM algorithm in terms of learning speed and computational cost. Experimental results show that when used to estimate large datasets, this algorithm is 30-80 times faster than the EM algorithm at least. Owing to its outstanding density estimation performance, this algorithm is very helpful to the construction of optimal classifiers. The effectiveness of the algorithm is demonstrated in several real-world applications.

Keywords-Probability density estimation; Pattern classification; Self-organizing mixture network; Maximum likelihood; Stochastic approximation method.

I. INTRODUCTION

In pattern classification, an accurate estimation of environmental distribution is vital for an optimal classifier. The non-parametric methods, such as kN-nearest-neighbor and Parzen windows, have the advantage of generality. But these methods require a very large number of samples, which leads to severe requirements for computation time and storage. The mixture model provides a more effective method to density estimation, in which the joint distribution is modeled as the weighted combination of simple component densities such as Gaussian, Cauchy, or Laplace. In mixture models, the parameters for each component density are estimated from the data samples solely. And the general maximum likelihood (ML) approach is an empirical method to solve this problem. In statistical theory, ML estimation and likelihood-based inference are of central importance [1]. While Bayesian estimation is to be preferred in principle, ML methods are generally easier to implement and can achieve same accuracy in the limit of large training sets [2]. The expectation-maximization (EM) algorithm is an iterative procedure to maximize the marginal log-likelihood and is powerful in solving mixture models. However, it has a high possibility of being trapped in local optima and is very computationally expensive with large data sets.

The self-organizing map (SOM) is a biologically inspired unsupervised learning algorithm and has been applied to many clustering and pattern recognition problems with considerable success [3]. However, the SOM has two significant deficiencies: it has no well-defined objective function that can be optimized; and its estimation of the probability density lacks accuracy [4]. These limitations have prompted many investigators to devise new ways to optimize the topographic mapping and approximate data distribution. In [5], Bishop and coworkers proposed the generative topographic mapping (GTM) to model and visualize high-dimensional data in a latent variable space of a low dimension. But the GTM uses spherical Gaussians with identical variance and exercises the EM algorithm, and its mapping function cannot be meaningful as an explanation of the possible mappings [6]. In [7], Van Hulle equipped the lattice neurons with local kernel functions to improve the density estimation property of the SOM, yet the complexity of the kernel functions hampers its practical application. Van Hulle also had introduced an algorithm, which has no competitive stage, for kernel-based topographic map formation of heteroscedastic Gaussian mixtures that allows for a unified account of distortion error, log-likelihood, and Kullback-Leibler divergence (KLD) [8]. But the distortion function is very complex and it also trained by the EM algorithm.

To learn arbitrary density functions accurately, Yin and Allinson proposed a self-organizing mixture network (SOMN) which is a combination of Kullback-Leibler information metric, the stochastic approximation method, and the SOM structure [9-11]. The SOMN minimizes the KLD between the true and the estimated densities by means of the Robbins-Monro stochastic approximation method. The KLD is sometimes claimed to be a more suitable measure for density estimation and unsupervised learning [12]. The SOMN is designed to be suited for inhomogeneous mixtures and heteroscedastic components to increase its accuracy of density matching and applicability. And since it is a stochastic gradient based algorithm, it has a high opportunity...
to escape from local minima. In addition, the SOMN also provides an insight into the role of winning rule and neighborhood function used in the SOM. However, it seems that in the SOMN, the derivation of the learning formula for the mixing parameter is incomplete, so that the updating formula will become divergent during learning process and the algorithm cannot fully achieve the desired performance.

To overcome the limitations mentioned above, we analyze the derivation of the SOMN in detail to find out the reasons that cause these deficiencies, and re-derive a stable and reasonable formula for the mixing parameters based on the criterion of minimizing the KL divergence and the maximum likelihood principle. Moreover, we simplify the competitive mechanism used in the SOMN and apply the posterior probability to approximate the mixing role of the neighborhood function. These let the modified algorithm has a simple structure and a well-defined objective function, which is very helpful to prove its convergence. The performance of our algorithm is analyzed and compared with the EM algorithm, the SOM and the nonparametric methods. In section 3, applications to density estimation and pattern classification are presented to demonstrate the accuracy of density matching and the applicability of the algorithm.

II. THE MODIFIED SELF-ORGANIZING MIXTURE NETWORK

A. Mixture Models and Maximum Likelihood Estimation

In mixture models, the sample distribution is often modeled as a mixture of some parametric forms such as Gaussian and Cauchy. Suppose \( x \) is a sample from a \( d \)-dimensional input space \( \Pi \in R^d \), \( K \) is the number of components, then the joint probability density of data sample is given by:

\[
p(x|\theta) = \sum_{i=1}^{K} p_i(x|\theta_i)P_i
\]

where \( P_i \) is the mixing probability or prior probability, \( p_i(x|\theta_i) \) is the \( i \)-th component-conditional density, \( \theta_i \) is the parameter vector for \( p_i(x|\theta_i) \), and \( \theta = (\theta_1, \theta_2, \ldots, \theta_K)^T \).

In this case, the log-likelihood of all samples is given by:

\[
\ell = \sum_{k=1}^{N} \ln p(x_k|\theta) = \sum_{k=1}^{N} \ln \sum_{i=1}^{K} p_i(x_k|\theta_i)P_i
\]

where \( N \) is the number of samples.

In mixture models, the number and the forms of the conditional densities are assumed, so the modeling problem becomes to learn the model parameters from the set of samples. The EM algorithm is usually applied to solve this parameter estimation problem. It is an iterative ML procedure and converges monotonically to a local maximum of the likelihood function. However, it is sensitive to initial states and is likely to suffer from slow convergence.

B. The SOMN Algorithm

The SOMN provides a feasible solution to estimate the mixture models by minimizing the KLD, which is a natural integration of the accumulative log-likelihood [9]. Suppose that the true environmental data density function is \( p(x) \) and the estimated one is \( \hat{p}(x) \), then the KLD is defined as:

\[
I = - \int \ln \frac{\hat{p}(x)}{p(x)} p(x) dx
\]

KLD is also referred to as relative entropy. It measures the divergence between \( p(x) \) and \( \hat{p}(x) \). To seek the optimal estimate of the model parameters, the SOMN takes the partial derivatives of KLD with respect to model parameters \( \theta_i \) and \( \hat{P}_i \) and sets them equal to zero, this leads to the following equations [9]:

\[
\frac{\partial I}{\partial \theta_i} = - \int \frac{\hat{p}_i(x|\theta)}{\hat{p}(x)} \frac{\partial \hat{p}_i(x|\theta)}{\partial \theta_i} p(x) dx = 0
\]

\[
\frac{\partial I}{\partial \hat{P}_i} = - \int \frac{\hat{p}_i(x|\theta)}{\hat{p}(x)} \frac{\partial \hat{p}_i(x|\theta)}{\partial \hat{P}_i} p(x) dx
\]

\[
+ \lambda \frac{\partial}{\partial \hat{P}_i} \left[ \sum_{j=1}^{K} \hat{P}_j - 1 \right] = 0
\]

The method of Lagrange multipliers with a constraint parameter \( \lambda \) is introduced in equation (5), since \( \hat{P}_i \) must obey the completeness constraint, i.e., \( \sum_{i=1}^{K} \hat{P}_i = 1 \).

As \( p(x) \) is not known, equations (4) and (5) are not directly solvable. Thus a recursive stochastic approximation method, e.g. the Robbins-Monro method is used to solve these equations. This results in the following adaptive updating algorithm \[9, 10\]: (In \[9, 10\], Yin and Allinson set the constraint parameter \( \lambda \) equal to one without giving further explanation).

\[
\hat{\theta}_i(n + 1) = \theta_i(n) + \alpha(n) \left[ \hat{P}_i(n) \frac{\partial \hat{p}_i(x|\theta)}{\partial \theta_i(n)} \right]
\]

\[
\hat{P}_i(n + 1) = \hat{P}_i(n) - \alpha(n) [\hat{P}(i|x) - \hat{P}_i(n)]
\]

where

\[
\hat{P}(i|x) = \frac{\hat{p}_i(x|\theta)}{\hat{p}(x)}
\]

is the estimated posterior probability of the \( i \)-th component. \( \alpha(n) \) is the learning rate, which decreases monotonically and must satisfy the following convergence conditions: (i) \( 0 < \alpha(n) < 1 \); (ii) \( \sum \alpha(n) \to \infty \); (iii) \( \sum \alpha(n)^2 < \infty \).

As for Gaussian mixtures, \( p_i(x|\theta_i) \) has the following form:

\[
p_i(x|\theta_i) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} \exp \left[ -\frac{1}{2} (x - m_i)^T \Sigma_i^{-1} (x - m_i) \right]
\]

where \( \theta_i = \{m_i, \Sigma_i\} \) are the mean vector and covariance matrix respectively. Then the corresponding partial differential term in (6) can be calculated straightforward, and thus the learning rules for \( m_i \) and \( \Sigma_i \) is obtained [9]:

\[
\Delta \hat{m}_i = \alpha(n) \hat{P}(i|x) [x - \hat{m}_i(n)]
\]
\[
\Delta \Sigma_i = \alpha(n) \hat{P}(i|x)[(x - \hat{m}_i(n))[x - \hat{m}_i(n)]^T - \Sigma_i(n)]
\] (11)

Yin and Allinson declared that the SOMN can be applied to inhomogeneous mixtures. However, after some algebraic manipulations, formula (7) becomes:

\[
\hat{p}_i(n + 1) = \hat{p}_i(n) \left[ 1 + \alpha(n)[1 - \hat{p}_i(x|\theta_i)/\bar{p}(x|\theta)] \right]
\] (12)

It is obviously that \( \bar{p}(x|\theta) > \hat{p}_i(x|\theta_i) \), so formula (12) is divergent. This reduces the performance of the SOMN evidently.

C. Modification of the SOMN

The preceding analysis shows that formula (7) is divergent, and this should be attributed to the fact that the parameter \( \lambda \) does not be determined rationally. It is easy to find that \( \lambda = 1 \) is not a valid solution of equation (5). Since the updating formula of \( \hat{p}_i \) cannot be derived though the stochastic approximation method, we re-derive it from a new way. We first fix the mixing parameters and minimize the KLD to derive the updating formula for \( \theta_i \), thus obtain the same formula as (6). Then using the obtained conditional densities, we maximize the log-likelihood of all samples to derive a new updating formula for \( \hat{p}_i \). Suppose \( X = \{x_1, x_2, \ldots, x_K\} \) is a set of \( N \) independent samples, then its log-likelihood is defined as (2). To ensure \( \sum_{i=1}^{K} \hat{p}_i = 1 \), the Lagrange multiplier method with a constraint parameter \( \lambda \) is introduced, so we have:

\[
\ell' = \sum_{k=1}^{N} \ln \sum_{i=1}^{K} p_i(x_k|\theta_i)\hat{p}_i + \lambda \left( \sum_{i=1}^{K} \hat{p}_i - 1 \right)
\] (13)

Since the gradient must equal to zero at the value of \( \hat{p}_i \) that maximizes \( \ell' \), the maximum likelihood estimate \( \hat{p}_i \) must satisfy the condition:

\[
\frac{\partial \ell'}{\partial \hat{p}_i} = \sum_{k=1}^{N} \hat{p}_i(x_k|\theta_i) + \lambda = 0, \quad i = 1, 2, \ldots, K
\] (14)

Solving equation (14) for \( \lambda \) we have: \( \lambda = -N \). Then, after some manipulations we obtain the ML estimate for \( p_i \) [2]:

\[
\hat{p}_i = \frac{1}{N} \sum_{k=1}^{N} \hat{p}(i|x_k)
\] (15)

Equation (15) states that the ML estimate of the probability \( p_i \) is the average over the entire data set of the estimate derived from each sample. This provides a meaningful explanation for the roles of the mixing parameters. Since the sample likelihood approximates the KL measure by the time average of sample probabilities, and the sample likelihood approach will equal the KL metric when the number of the sample points tends to infinity and the input is an ergodic process [9, 10], it is convinced that the formula derived from maximizing the sample likelihood is consistent with those from minimizing the KLD, and they can work well together.

In general, the modified algorithm can be summarized as follows:

1. Initialize \( \theta = (\theta_1, \theta_2, \ldots, \theta_K)^T \), set \( \hat{p}_i = 1/K \).
2. Draw a sample \( x_k \) from the training set with a given probability.
3. Calculate the posterior probability of each component using formula (8).
4. Adjust \( \theta_i \) using the corresponding updating formulas such as (10) and (11).
5. Repeat step 2-4 until whole the training set has been searched.
6. Update mixing parameters using formula (15).
7. Repeat step 2-6 until the estimated parameters are stably convergent, namely, no noticeable changes are observed.

This algorithm has the following characteristics:

(1) In the algorithm, the stochastic approximation formulas for \( \theta_i \) is remained to get better convergence properties, but the divergent updating formula of \( \hat{p}_i \) is replaced by a stable one. Since \( \hat{p}_i \) is calculated outside the recursive process of \( \theta_i \), the computational complexity of the algorithm is determined by the inner iteration, so the algorithm still is stochastic gradient descent. In general, this algorithm uses a more reasonable formula to update the mixing parameters, so that each component in the mixture model can be truly used to interpret interesting sub-densities, and thus it is much stable, accurate and effective than the SOMN.

(2) In the SOMN, a winner is chosen according to its posterior probability, and the updating of the weights is limited to a small neighborhood around the winner. On the contrary, the modified algorithm does not apply an explicit neighborhood function. Actually it updates all the parameter vectors according to their posterior probabilities, as shown in formulas (10) and (11). Van Hulle had shown that in many kernel-based topological maps, the posterior may has played the role of the neighborhood function [6]. In [10], the mixing role of the neighborhood function is also regarded as same as a posterior. Based on these researches, it seems that updating all the parameter vectors according to their posterior probabilities can achieve the same functions of the competitive and cooperative mechanism. This provides a probabilistic interpretation for the learning opportunity and the role of the neighborhood function as well. Experiments also show that this learning algorithm can form the topographically ordered maps in some extent.

The benefit of using an implicit neighborhood function is that it can greatly reduce the complexity of the algorithm and lets it has a well-defined objective function. With the simplification of competitive and cooperative mechanism, this algorithm can be regarded as a stochastic approximation method for solving mixture models based on the criterion of minimizing the relative entropy, the maximum-likelihood approach, and the self-organizing principle. And since the sample likelihood is an approximation to the KLD, the objective function of this algorithm can be considered as minimizing the KLD (or, maximizing the sample likelihood) through a stochastic descent method. As Kosko had shown that stochastic gradient methods converge exponentially to
centroids [13], this provides the algorithm with a general proof of convergence, and shows that its computational complexity is $O(NK\ln k)$, where $N$ is the number of samples and $K$ is the number of underlying components.

### III. Experimental Results

#### A. Density estimation of standard normal distribution

We use the same synthetic example in [7] to assess the density estimation performance of the modified algorithm. In this experiment, randomly generated standard Gaussian data are used to train the algorithm. The EM algorithm is also applied to this example. In both algorithms, the number of components $K$ is set to 9 and the times of iteration are set to 100 equally. A typical estimated density of our algorithm (blue continuous line) and the original Gaussian distribution (red dashed line) are given in Fig.1. From Fig.1 we can see that both lines are almost completely coinciding. When calculated at the same positions as shown in [7], the mean squared error (MSE) of the modified SOMN and the EM algorithm are $1.63 \times 10^{-6}$ and $5.58 \times 10^{-6}$ respectively. In [7], the estimation results of the joint entropy maximization algorithm, SOMN, STVQ, SSOM, LDE, and kMER were compared, and the best estimation is achieved by the joint entropy maximization algorithm, which MSE is $9.35 \times 10^{-5}$. It can be seen that the modified SOMN is more accurate than the EM algorithm and the methods discussed in [7].

![Figure 1. One-dimensional, unit-variance Gaussian density (red dashed line) and the estimate (blue continuous line) obtained with the modified algorithm.](image)

#### B. The Wine Dataset Classification

The Wine Data Set [14] is one of the most popular dataset in the UCI Machine Learning Repository. It contains 178 instances, and each instance has 13 attributes. The dataset is divided into 3 categories. In this example, we train the modified algorithm to estimate the distribution functions of each category, and then classify the dataset based on the obtained estimates. For comparison’s sake, the SOM and k-nearest neighbor method are also applied to this problem. All methods use the leave-one-out technique.

In the case of the k-nearest neighbor method, the classification results are 76.4% correct classification when $k=5$, 73.1% correct classification when $k=7$, and 75.9% correct classification when $k=9$. This means that there always has about 1/4 instances cannot be classified correctly. Take the instance 172(3) as an example (the number in the brackets is the category of the instance): its nine nearest neighbors are instances 73(2), 157(3), 91(2), 128(2), 80(2), 108(2), 92(2), 62(2), and 86(2). So, no matter how the rules change, it will always be classified into second category incorrectly.

In the case of the SOM which using a $6 \times 6$ lattice, the classification result is 79.9% correct classification. This result is slightly better than that of the k-nearest neighbor method due to the SOM divides the instances into more subcategories (e.g., 36 here).

These results show that the classifiers rely on a metric or “distance” function between patterns may sometimes lead to very large errors and particularly ill-suite to high-dimensional problems.

When using the modified algorithm to learn the distributions (the number of components $K$ is set to 9), the resulting estimates are shown in Fig.2. We observe that the distribution functions of each category can be separated completely (e.g., the density of category 1 will equal to zero at the values of instances belong to category 2 and 3, and so on), so it can achieve 100% correct classification. In this example, the distribution-based classifier provides a perfect classification.

![Figure 2. Distribution estimates of each category in the Wine Dataset obtained by the modified algorithm.](image)

#### C. Skin modeling and classification

In this example we apply the modified SOMN to model the skin and non-skin pixels, and then use the Bayesian decision rule to classify the modeled distributions. The dataset used here contains 37.55 million skin pixels and 135.58 million nonskin pixels. It is carefully chosen to ensure the diversity in terms of the background scenes, lighting conditions, and skin types [15]. The dataset is divided into two parts: one is used to train the algorithms and the other to evaluate the classification results. The EM algorithm is also applied to this problem. In both algorithms, the number of components $K$ is set to 100 equally. At each iteration step of the modified SOMN, the samples are randomly drawn from training set by a probability of 3-5%.

To evaluate the accuracy of the estimates, we use the training dataset to construct the one-dimensional histograms of the B, G, and R components in the RGB color-space, and then project the estimated distributions into the B, G, and R
planes to assess their estimation accuracy. The estimation accuracy is also measured by MSE in this example. Typical results of the EM and the modified algorithm are listed in Table.1. We observe that the modified SOMN is more accurate than the EM algorithm. Fig.3 gives the comparison of the histograms of B component and its estimated profiles by the modified algorithm. It can be seen that both lines are almost coinciding.

![Figure 3. Comparison of the histograms (gary lines) and the estimated profiles (red lines) of the modified algorithm. (top: skin-color distribution of B component; bottom: nonskin-color distribution of B component)](image)

Table 1. The MSE between the estimated distributions and the histograms

<table>
<thead>
<tr>
<th>Component</th>
<th>B</th>
<th>G</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td>Skin-color set</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Modified SOMN</td>
<td>$4.34 \times 10^{-9}$</td>
<td>$4.38 \times 10^{-9}$</td>
<td>$9.67 \times 10^{-8}$</td>
</tr>
<tr>
<td>EM algorithm</td>
<td>$4.47 \times 10^{-9}$</td>
<td>$6.07 \times 10^{-9}$</td>
<td>$1.10 \times 10^{-7}$</td>
</tr>
<tr>
<td>Nonskin-color set</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Modified SOMN</td>
<td>$6.27 \times 10^{-7}$</td>
<td>$3.40 \times 10^{-7}$</td>
<td>$3.52 \times 10^{-7}$</td>
</tr>
<tr>
<td>EM algorithm</td>
<td>$6.68 \times 10^{-7}$</td>
<td>$3.60 \times 10^{-7}$</td>
<td>$3.80 \times 10^{-7}$</td>
</tr>
</tbody>
</table>

Table 2. The time (seconds) used to train the EM and the modified algorithm

<table>
<thead>
<tr>
<th>Training dataset</th>
<th>Skin-color set (25 million pixels)</th>
<th>Nonskin-color set (90 million pixels)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modified SOMN</td>
<td>605</td>
<td>2106</td>
</tr>
<tr>
<td>EM algorithm</td>
<td>17410</td>
<td>175021</td>
</tr>
</tbody>
</table>

Table 2 gives the time used to train the EM and the modified algorithm. It can be seen that: to train the modified SOMN, it takes 605 seconds for the skin-color data (25 million pixels) and 2106 seconds for the nonskin-color data (90 million pixels) in one common personal computer. As for the EM algorithm, it takes 17410 and 175021 seconds respectively. In these examples, the modified SOMN is 30-80 times faster than the EM algorithm. And with the growth of the data size, the modified algorithm may save more time. It is much faster than the EM algorithm because:

1. The modified algorithm is stochastic gradient descent, whilst the EM algorithm is deterministic gradient descent. The stochastic gradient methods generally outperform the deterministic gradient methods in terms of better solvability and improved convergence [9]. This can be observed directly in the learning curves of both algorithms in Fig.4. Fig.4 shows the average log-likelihood curves of the modified algorithm and the EM algorithm during the first 20 training epochs of the skin-color dataset. As can be seen, the convergence properties of these two methods differ markedly. The modified SOMN quickly converges to near the final result in 2-3 epochs, and then fluctuates slightly, but generally converges to better solution. As for the EM algorithm, it converges smoothly but slowly, and it takes at least 20 epochs to approximate the learning result of the modified SOMN. It is obvious that the modified SOMN has higher convergent speed and better ability to escape from local minima.

![Figure 4. Comparison of convergence properties of the modified SOMN and the EM algorithm.](image)

2. The modified SOMN also is a nonlinear mapping network. It has the prominent properties of the neural networks such as good generalization ability (or, good nonlinear interpolation of the input data). Therefore, at each iteration step, it can randomly choose a very small part of the dataset to train the network. Thus, the algorithm requires much lower computational cost.

According to the obtained skin and nonskin color models, the Bayesian decision rule is applied to classify the pixels in the testing set. The segmentation performance is measured in terms of the true positive rate (TPR) and the false positive rate (FPR). The TPR is the percentage of skin pixels correctly classified; the FPR is the percentage of non-skin pixels incorrectly classified. Our typical detection results are TPR=80.01% with FPR=8.21%, or TPR=90.4% with FPR=14.21%. These results are comparable with histogram
model-based classifiers and outperform the GMM-based classifiers. And our detection method is far superior in storage requirement and computation efficiency [15].

When applied to the Skin Segmentation Dataset [14, 16], which contains 50859 skin samples and 194198 non-skin samples, our method can achieve a detection rate with TPR=99.89% and FPR=0.10%. This agrees with the conclusion of [9, 10], which states that: when only a small dataset is used to train the SOMN, it still can yield stable and accurate estimates.

IV. CONCLUSION

The SOMN is an extension to the SOM for solving general mixture distributions. It is designed to learn arbitrarily complex density functions, and it also provides a new explanation of the role of neighborhood functions used in the common SOM. However, in the SOMN, the learning formula of the mixing parameters is not convergent, so that it cannot fully achieve the desired performance. To overcome the limitations of the SOMN, we re-derive an updating formula for the mixing parameters based on the criterion of minimizing the KLD and the maximum-likelihood principle. This greatly improves the stability and applicability of the SOMN. Moreover, we further simplify the competitive and cooperative mechanism used in the SOMN, and apply the posterior probability to approximate the mixing role of the neighborhood function. This lets the modified SOMN has an explicit objective function, which helps to provide a general proof of convergence. Formal analysis and experiments show that the modified algorithm has advantages such as fewer local optima, faster convergence speed, and better generalization ability, and it can learn arbitrary density functions accurately and effectively. When used for density estimation, it outperforms the EM algorithm, the SOMN, and many other algorithms in estimation accuracy, and is far superior to the EM algorithm in terms of learning speed and computational cost. Experimental results show that when used to estimate large datasets, the modified algorithm is 30-80 times faster than the EM algorithm at least. Owing to its outstanding density estimation performance, the modified algorithm is very helpful to the construction of optimal classifiers. In the Wine Dataset classification example, our classifier can achieve 100% correct classification.

REFERENCES