A Single Loop EM Algorithm for the Mixture of Experts Architecture

Yan Yang and Jinwen Ma*

Department of Information Science, School of Mathematical Sciences & LMAM
Peking University, Beijing, 100871, P. R. China
jwma@math.pku.edu.cn

Abstract. The mixture of experts (ME) architecture is a powerful neural network model for supervised learning, which contains a number of "expert" networks plus a gating network. The expectation-maximization (EM) algorithm can be used to learn the parameters of the ME architecture. In fact, there have already existed several methods to implement the EM algorithm, such as the IRLS algorithm, the ECM algorithm, and an approximation to the Newton-Raphson algorithm. The differences among these implementations rely on how to train the gating network, which results in a double-loop training procedure, i.e., there is an inner loop training procedure within the general or outer loop training procedure. In this paper, we propose a least mean square regression method to learn or compute the parameters for the gating network directly, which leads to a single loop (i.e., there is no inner loop training) EM algorithm for the ME architecture. It is demonstrated by the simulation experiments that our proposed EM algorithm outperforms the existing ones on both speed and classification accuracy.

Keywords: The mixture of experts (ME) architecture, The EM algorithm, Gating network, Single loop, Least mean square regression.

1 Introduction

For a supervised learning problem, it is common to train a single or multilayer neural network to model and solve it. But if our aim is to perform different subtasks on different occasions, the learning algorithm like the error backpropagation procedure will generate strong interference effects that lead to slow convergence and poor generalization. In order to overcome this difficulty, Jacobs, Jordan, Nowlan and Hinton [1] established the mixture of experts (ME) architecture which consists of some different "expert" networks plus a gating network. Actually, since a complex problem can be often divided into certain subproblems, the ME architecture can be used to train these subproblems, respectively, and the gating network determines which expert network should be used for each subtask.

The well-known Expectation-Maximization (EM) algorithm [2] is designed to solve the maximum likelihood estimation problem for a probability model in which some random variable can be observed, while the other random variable

* Corresponding author.

W. Yu, H. He, and N. Zhang (Eds.): ISNN 2009, Part II, LNCS 5552, pp. 959–968, 2009.

cannot be observed. Actually, the EM algorithm has already been implemented on finite mixtures in which the random variables of the components are observable but the random variables to reflect the classification of the component data are unobservable or hidden [3]. Recent theoretical analysis has proved that the EM algorithm for Gaussian mixtures or the mixtures of densities from a class of exponential families tends to be asymptotically superlinear and correct when the overlap of densities in the mixture tends to zero [4],[5],[6]. As a generalization of the Gaussian mixture model, the ME architecture can be certainly trained by the EM algorithm. As matter of fact, there have already existed several versions of the EM algorithm for the ME architecture in literature. The EM algorithm for the ME architecture implements a double-loop iteration at each step. That is, one general or outer loop is to learn the parameters of the experts, while an inner loop is to learn the parameters of the gating network within the outer loop. In fact, the differences among these EM implementations rely on how to train the gating network, i.e., the inner loop for the learning of the parameters in the gating network.

In the early stage, the gating network inner loop in the EM algorithm was implemented by the iteratively reweighted least squares (IRLS) algorithm [7], [8]. Although the IRLS algorithm is very popular, it often leads to an unstable performance. In order to overcome this weakness, Chen et al. [9] applied the Newton-Raphson algorithm to the inner loop, where a Hessian matrix and its inverse were needed to be computed. Ng and McLanchlan [10] further proposed an ECM algorithm for the inner loop. It was shown by the experiments that the Newton-Raphson algorithm has a better accuracy than the IRLS algorithm, but its convergence is much slower. On the other hand, the ECM algorithm is superior to the IRLS algorithm on some simulated and real data sets. It is also shown by the experiments that if the learning rate in the inner loop is not very small, the IRLS algorithm performs poorly. It is clear that, by each of these methods, the inner loop needs to implement an iterative learning algorithm for a number of iterations, which make the EM algorithm be very time-consuming.

In this paper, we apply the least mean square method to learning or computing the optimal parameters of the gating network directly. In the inner loop, we solve a regression solution of the parameters for the gating network. The updates of the other parameters keep the same as those of the other EM algorithms. Thus, our training procedure of the ME architecture for the gating network parameters becomes one-step-update, i.e., no inner loop training is needed. For convenience, we refer to this kind of the EM algorithm as the single loop EM algorithm. It is demonstrated by the experiments on both synthetic and real data sets that our single loop EM algorithm is considerably faster than all the others, with a better accuracy on parameter estimation.

The rest of the paper is organized as follows. Section 2 briefly describes the EM algorithm for the ME architecture. In Section 3, we propose the single loop EM algorithm. Furthermore, the experimental results on several data sets are presented in Section 4. Finally, we conclude briefly in Section 5.

2 The EM Algorithm for ME Architecture

We consider the following ME architecture model:

$$P(y|x) = \sum_{j=1}^{K} P(j|x)P(y|x,\theta_j,\Sigma_j) = \sum_{j=1}^{K} g_j(x,\theta_0)P(y|x,\theta_j,\Sigma_j), \tag{1}$$

where

$$g_j(x,\theta_0) = \frac{e^{s_j(x,\theta_0)}}{\sum_{j=1}^{K} e^{s_j(x,\theta_0)}}, \tag{2}$$

$$P(y|x,\theta_j,\Sigma_j) = \frac{1}{(2\pi)^{m/2}|\Sigma_j|^{1/2}} \exp\{-\frac{1}{2}[y - f_j(x,\theta_j)]^T \Sigma_j^{-1}[y - f_j(x,\theta_j)]\}, \tag{3}$$

K is the number of the mixture components or experts, $x \in \mathbb{R}^n$ denotes a sample vector, and $y \in \mathbb{R}^m$ is the output vector. The parameter vector Θ consists of θ_0, θ_j, and the covariance matrices Σ_j, which are assumed positive definite.

For simplicity, the functions f_j are assumed to be linear in the parameters [8] as follows:

$$f_j(x) = X^T\theta_j,$$

where

$$X^T = \left\{ \begin{array}{ccccccc|ccccc} x^T & 0 & \cdots & \cdots & 0 & | & 1 & 0 & \cdots & \cdots & 0 \\ 0 & x^T & 0 & \cdots & 0 & | & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & & & \vdots & | & \vdots & & & & \vdots \\ 0 & \cdots & \cdots & 0 & x^T & | & 0 & \cdots & \cdots & 0 & 1 \end{array} \right\}.$$

The i-th output of the gating network, $g_i(x,\theta_0)$, is given as the multinomial softmax function with s_i being linear in the parameters by

$$s_i(x,\theta_0) = x^T\theta_{0i}.$$

For convenience of expression, we modify the above equation as follows:

$$s_i(x,\theta_0) = [x^T 1]\theta_{0i},$$

which is more popular in regression theory.

The multinomial softmax function contains K parameter vectors which are not independent. In fact, there are only $K-1$ independent parameter vectors due to the constraint $\sum_{i=1}^{K} g_i = 1$. Specifically, the multinomial softmax function can be expressed as follows:

$$g_i(x,\theta_0) = \left\{ \begin{array}{ll} \dfrac{e^{s_i}}{1 + \sum_{j=1}^{K-1} e^{s_j}}, & i \neq K \\ \dfrac{1}{1 + \sum_{j=1}^{K-1} e^{s_j}}, & i = K \end{array} \right. \tag{4}$$

where $\theta_0 = [\theta_{01}^T, \cdots, \theta_{0(K-1)}^T]^T$.

Given K and independently and identically distributed (i.i.d.) samples $\{x^{(t)}, y^{(t)}\}_1^N$, we can estimate Θ by maximizing the log-likelihood:

$$l(\Theta, \mathcal{Y}) = \sum_{t=1}^{N} ln \sum_{j=1}^{K} g_j(x^{(t)}, \theta_0) P(y^{(t)}|x^{(t)}, \theta_j, \Sigma_j). \tag{5}$$

In order to do so, we can implement the EM algorithm for the ME architecture and obtain the so-called Q function:

$$Q(\Theta|\Theta^{(k)}) = \sum_{t=1}^{N}\sum_{j=1}^{K} h_j^{(k)}(t) \log g_j(x(t)) + \sum_{t=1}^{N}\sum_{j=1}^{K} h_j^{(k)}(t) \log P(y^{(t)}|x^{(t)}, \theta_j, \Sigma_j), \tag{6}$$

where

$$h_j^{(k)}(t) = \frac{g_j(x^{(t)}, \theta_0) P(y^{(t)}|x^{(t)}, \theta_j^{(k)}, \Sigma_j^{(k)})}{\sum_{i=1}^{K} g_i(x^{(t)}, \theta_0) P(y^{(t)}|x^{(t)}, \theta_i^{(k)}, \Sigma_j^{(k)})}. \tag{7}$$

Under the EM framework for each iteration, we need to compute the E (Expectation) step and the M (Maximization) step alternatively. Specifically, we need to get the Q function with the estimated parameters Θ_k in the E-step, and solve the parameter maximum of the Q function in the M-step. Although the maximum solution can be easily solved on the second term of the Q function, it is quite difficult to solve the maximum solution on the first term of the Q function. Therefore, we need to apply some learning algorithm to help solve the maximizing problem in the M-step.

In fact, we can update Σ_j and θ_j by the following equations:

$$\Sigma_j^{(k+1)} = \frac{1}{\sum_{t=1}^{N} h_j^{(k)}(t)} \sum_{t=1}^{N} h_j^{(k)}(t)[y^{(t)} - f_j(x^{(t)}, \theta_j)][y^{(t)} - f_j(x^{(t)}, \theta_j)]^T, \tag{8}$$

$$\theta_j^{(k+1)} = (R_j^{(k)})^{-1} c_j^{(k)}, \tag{9}$$

where

$$c_j^{(k)} = \sum_{t=1}^{N} h_j^{(k)}(t) X_t (\Sigma_j^{(k)})^{-1} y^{(t)}, \tag{10}$$

$$R_j^{(k)} = \sum_{t=1}^{N} h_j^{(k)} X_t (\Sigma_j^{(k)})^{-1} X_t^T. \tag{11}$$

As for θ_0, Jordan and Xu [8] proposed the IRLS algorithm, and also proved that the complete EM algorithm can be viewed as a modified gradient scent algorithm for maximizing the log-likelihood function l. On the other hand, Chen et al. [9] proposed an approximation to the Newton-Raphson algorithm based on a so-called generalized Bernoulli density. Moreover, Ng and McLachlan [10] proposed an ECM algorithm to maximize the first term of the Q function.

These three algorithms for θ_0 are all iteratively learning procedures, and the purpose of the inner loop is to find a numerical solution which maximize the first term of the Q function. In the inner loop, a Hessian matrix or an approximate Hessian matrix and its inverse are needed to be computed. Thus, the procedure will be time consumptive.

In order to overcome the weaknesses of these algorithms, especially on time-consuming, we utilize the least mean square method to maximize the first term of the Q function to directly get the update of θ_0. That is, we use a direct update of the parameters of the gating network instead of the inner learning procedure. The new EM algorithm will be derived and demonstrated in the following sections.

3 The Single Loop EM Algorithm

We consider the first term of the Q function:

$$l_g = \sum_{t=1}^{N} \sum_{j=1}^{K} h_j^{(k)}(t) \log g_j(x(t)), \tag{12}$$

and get its derivatives with respective to the components of θ_0 as follows:

$$\frac{\partial l_g}{\partial \theta_0^{(k)}} = \sum_{t=1}^{N} \sum_{j=1}^{K} [h_j^{(k)}(t) - g_j(x^{(t)}, \theta_0^{(k)})] \frac{\partial s_j}{\partial \theta_0^{(k)}}. \tag{13}$$

In order to get the maximum solution of l_q, we try to get the zero point of Eq.(13). In general, this kind problem can be solved by the gradient ascent or Newton's method. However, we can apply the least mean square method on solving this problem.

It is clear that if $g_j(x^{(t)}, \theta_0^{(k)}) - h_j^{(k)}(t) = 0$ holds for $t = 1, \cdots, N$ and $j = 1, \cdots, K$, Eq.(13) becomes zero and Eq.(12) probably reaches a maximum. Hence, we can try to solve a solution of the following equations:

$$\begin{cases} \frac{e^{s_1}}{1+\sum_{j=1}^{K-1} e^{s_1}} = h_1^{(k)}(t), \\ \quad \vdots \\ \frac{1}{1+\sum_{j=1}^{K-1} e^{s_j}} = h_K^{(k)}(t). \end{cases} \tag{14}$$

In order to do so, we divide the first $K-1$ equations by the last equation on both sides, and obtain:

$$\begin{cases} e^{s_1} = \frac{h_1^{(k)}(t)}{h_K^{(k)}(t)}, \\ \quad \vdots \\ e^{s_{K-1}} = \frac{h_{K-1}^{(k)}(t)}{h_K^{(k)}(t)}. \end{cases} \tag{15}$$

Thus, we further obtain the following equations:

$$[x(t)^T 1]\theta_{0j} = \log(\frac{h_j^{(k)}(t)}{h_K^{(k)}(t)}), j = 1, \cdots, K-1. \tag{16}$$

Finally, we obtain the least mean square solution of Eq.(16) as follows:

$$\theta_{0j}^{(k+1)} = (Y^T Y)^{-1} Y^T H_j^{(k)}, j = 1, \cdots, K-1, \tag{17}$$

where

$$Y = \begin{Bmatrix} x^T(1) & 1 \\ \vdots & \\ x^T(N) & 1 \end{Bmatrix},$$

and

$$H_j^{(k)} = (\log\frac{h_j^{(k)}(1)}{h_K^{(k)}(1)}, \cdots, \log\frac{h_j^{(k)}(N)}{h_K^{(k)}(N)})^T.$$

By taking in Eq.(17) as the update of θ_0 and combining with Eqs (8) & (9), we construct a new EM algorithm for the ME architecture. In the Expectation (E) step, Eq.(7) is implemented to compute the corresponding posterior probability. In the Maximization (M) step, Eq.(8), (9) & (17) are implemented to compute the parameters that maximize the Q function. Since we compute the update of θ_0 directly, we call it the single loop EM algorithm.

In the next section, the single loop EM algorithm will be tested and compared with the existing EM algorithms.

4 Experimental Results

In this section, we conduct several experiments on synthetic and real-world data sets to test our single loop EM algorithm for the ME architecture. Firstly, in order to compare with the EM algorithm using the IRLS algorithm, we conduct the experiment on the same data set given and used in [8]. Secondly, we conduct the experiment on the Iris data set and compare the result with the EM algorithm using the Newton-Raphson algorithm. Finally, we conduct the experiments on the Thyroid data set and the Leptograpsus Crab data set. The experimental results are compared with those of the ECM algorithm [10].

When the ME architecture is used for multi-task classification in the above cases, we assume that the output vector y is a discrete binary vector. For example, if there are four distinct categories, y may be one of the vectors: $[0, 0]$, $[0, 1]$, $[1, 1]$ and $[1, 0]$. It is reasonable that the elements of y is independent, so that the covariance matrices $\Sigma_j, j = 1, \cdots, K$ in Eq.(3) are diagonal matrices.

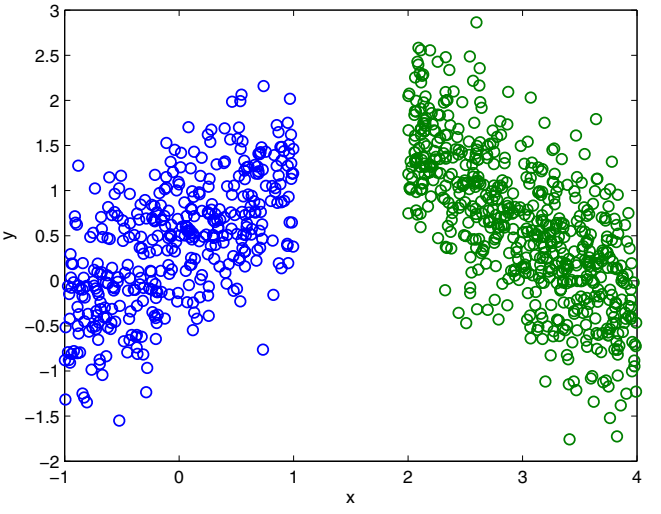

Fig. 1. The synthetic data set used in [8]

4.1 On a Synthetic Data Set

For convenience of comparison, we here use the synthetic data set used in [8], which is sketched in Fig.1. The data points were generated from the piecewise linear function $y = a_1x+a_2+n_t$, $x \in [x_L, x_U]$ and $y = a'_1x+a'_2+n_t$, $x \in [x_L, x'_U]$, where n_t is a Gaussian random variable with zero mean and variance $\sigma = 0.3$. The data points sketched in Fig. 1 are generated using the following parameter values $a_1 = 0.8$, $a_2 = 0.4$,$x_L = -1.0$, $x_U = 1.0$, $a'_1 = -1.0$, $a'_2 = 3.6$, $x'_L = 2.0$, $x'_U = 4.0$. The training data set contains 1000 data points.

We implement our single-loop EM algorithm for the two-expert ME architecture on the synthetic data set. Fig. 2 shows the evolution of the average log likelihood, the parameters of the expert networks and the variances. It can be observed that the obtained values of the parameters are very close to the real ones. The average log likelihood converges faster than that exhibited in [8]. The most important thing is that, no learning rate is needed in the single loop algorithm.

4.2 On the Iris Data Set

We further apply our single loop EM algorithm to classification of the Iris data set which is a typical real-world data sets [12]. The Iris data set contains 150 sample points, each category 50 points. We randomly chose 90 sample points as training data set, each category 30 points. The remaining data are left for test. We implement the single loop EM algorithm for the three-expert ME architecture on the training set of the Iris data for 10 times and list the average results in Table 1. It can be observed that 2.0 test data points are misclassified. The average accuracy rate reaches at 96.67%. The average training procedure only needs 3 epochs.

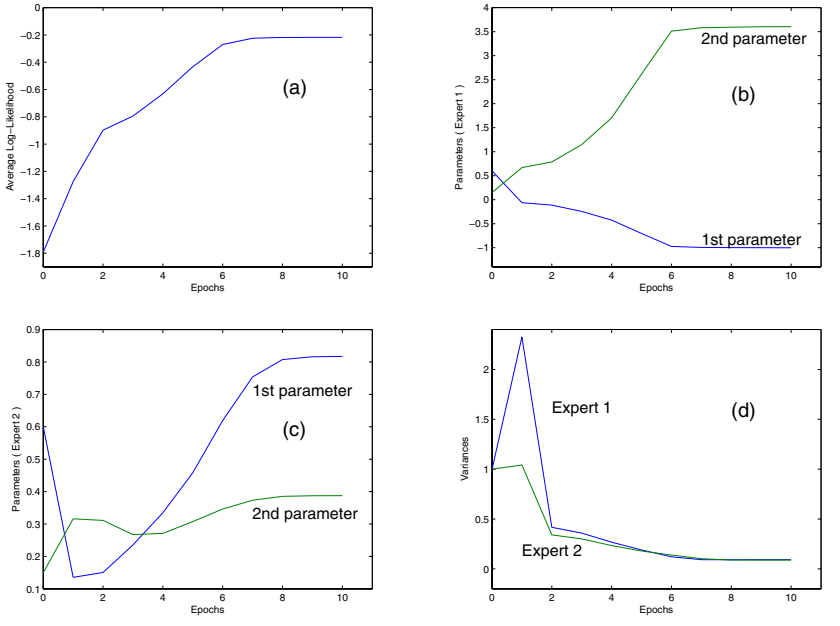

Fig. 2. The performance of the single-loop EM algorithm: (a). The evolution of the average log likelihood; (b). The evolution of the parameters for expert 1; (c). The evolution of the parameters for expert 2; (d). The evolution of the variances.

Table 1. The performance of the single loop EM algorithm on the Iris data set. The experimental results of the EM algorithms with the Newton-Raphson, Approximation, BFGS, and IRLS algorithms are simply copied from [9].

Algorithm	Error no.	Accuracy Rate	Epochs
Newton-Rapshon	4.0	93.33%	8.0
Approximation	4.2	93.00%	19.2
BFGS	4.2	93.00%	23.4
IRLS	6.8	88.67%	25.0
Single-Loop	2.0	96.67%	3.0

In comparison with the experimental results of the EM algorithms using the IRLS and Newton-Raphson algorithms on the same training data set, the error number of our EM algorithm is less than that of the EM algorithm using the IRLS algorithm, 6.8 in average, and that of the EM algorithm using the Newton-Raphson algorithm, 4.0 in average. Furthermore, our algorithm is much faster than both of them, because no inner loop procedure needs to be taken. Besides, we only have 3 epochs of the entire iteration procedure. In fact, the EM algorithm using the Newton-Raphson algorithm takes 8.0 epochs, while the EM algorithm using the IRLS algorithm even takes 25.0 epochs.

4.3 On the Thyroid Data Set

The task of the Thyroid problem is to decide whether a patient's thyroid has over-function, normal-function or under-function. The data set consists of 3772 patterns for training and 3428 patterns for testing. There are 21 features of each pattern, 15 of which are binary and 6 of which are continuous ones.

In our experiment, the ME architecture consists of 8 experts. The input vector x is a 21−dimensional vector. The output vector y is a 2−dimensional vector which is supposed to be $[0,0]$, $[0,1]$, or $[1,1]$. All the parameters are initiated with random values drawn from a uniform distribution on the unit interval.

We repeat the algorithm for 10 times with randomly chosen initial parameters. Each time we run the algorithm for 5 epochs. The average results are given in Table 2, being compared with the results of the ECM algorithm and the IRLS algorithm. From Table 2, it can be observed that the single-loop algorithm outperforms the ECM algorithm and the IRLS algorithm on both speed and classification accuracy.

Table 2. The performance of the single loop EM algorithm on the Thyroid data set. The experimental results of the ECM and IRLS algorithms are simply copied from [10].

	Training Set		Test Set		
Algorithm	Error no.	Accuracy Rate	Error no.	Accuracy Rate	Epochs
IRLS	70.0	98.14%	101.0	97.05%	60
ECM	42.0	98.89%	81.0	97.64%	60
Single-Loop	3.1	99.92%	5.6	99.84%	5

4.4 On the Leptograpsus Crab Data Set

The Leptograpsus Crab data set contains 5 morphological measurements of 200 crabs. They have two color forms and two sexes. So there are four categories of the crabs and each category contains 50 crabs.

In our experiment, a ME architecture with 2 experts is adopted. The input vector x is a 5−dimensional vector and the output vector y is a 2−dimensional vector. y is supposed to be one of the four vectors: $[0,0]$, $[0,1]$, $[1,1]$ and $[1,0]$. All the parameters are initiated with random values drawn from a uniform distribution on the unit interval.

We randomly choose 20 examples of each category to the train the ME architecture and the remaining examples are left for test. The algorithm is repeated 10 times and only 4 epochs are taken in each trial. It is found by the experiments on the Leptograpsus Crab data set that the average accuracy rate of the single loop algorithm is 96.38%, which is considerably higher than 91.67% of the IRLS algorithm and 94.17% of the ECM algorithm for the same Leptograpsus Crab data set reported in [10].

5 Conclusions

We have proposed a single loop EM algorithm for the mixture of experts architecture with the help of the least mean square regression method. The single loop EM algorithm makes the M-step of the EM algorithm simple and easy and therefore speed up the convergence considerably. The experiments on both synthetic and real-world data sets shows that the proposed single loop EM algorithm considerably faster than the existing ones, also with a better accuracy on parameter estimation.

Acknowledgments

This work was supported by the Natural Science Foundation of China for grant 60771061.

References

1. Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E.: Adaptive mixtures of local experts. Neural Computation 3, 79–87 (1991)
2. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximun likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Soceity B 39, 1–38 (1977)
3. Redner, R.A., Walker, H.F.: Mixture densities, maximum likelihood, and the EM algorithm. SIAM Review 26, 195–239 (1984)
4. Ma, J., Xu, L., Jordan, M.I.: Asymptotic convergence rate of the EM algorithm for Gaussian mixtures. Neural Computation 12, 2881–2907 (2000)
5. Ma, J., Xu, L.: Asymptotic convergence properties of the EM algorithm with respect to the overlap in the mixture. Neurocomputing 68, 105–129 (2005)
6. Ma, J., Fu, S.: On the correct convergence of the EM algorithm for Gaussian mixtures. Pattern Recognition 38(12), 2602–2611 (2005)
7. Jordan, M.I., Jacobs, R.A.: Hierachical mixtures of experts and the EM algorithm. Neural Computation 6, 181–214 (1994)
8. Jordan, M.I., Xu, L.: Convergence Results for the EM Approach to Mixtures of Experts Architectures. Neural Computation 8(9), 1409–1431 (1995)
9. Chen, K., Xu, L.: Improved learning algorithms for mixture of experts in multiclass classification. Neural Networks 12(9), 1229–1252 (1999)
10. Ng, S.K., McLachlan, G.J.: Using the EM Algorithm to Train Neural Networks: Misconceptions and a New Algorithm for Multiclass Classification. IEEE transactions on neural networks 15(3), 738–749 (2004)
11. Ng, S.K., McLachlan, G.J.: Extension of Mixture-of-experts networks for binary classification of hierachical data. Artificial Intelligence in Medicine 41, 51–67 (2007)
12. UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, `http://www.ics.uci.edu/~mlearn/MLRepository.html`