Transactions on Information and Communications Technologies vol 19, © 1997 WIT Press, www.witpress.com, ISSN 1743-3517

The performance of various learning rates for an unsupervised neural network

C.K. Lee, C.H. Chung
Department of Electronic Engineering, Hong Kong Polytechnic, Hung Hom, Hong Kong.

Email:encklee@hkpucc.polyu.edu.hk

Abstract

In this paper, we shall investigate the effect of *learning rate* and *threshold* to an unsupervised neural network when applied to an inspection process. The network we use is the *Learning by Experience* (LBE)[1]. Here, we analyse the effect based on a performance index. Experimental results are included when this neural network is applied to IC leadframe inspection.

1. Introduction

Whenever we want to apply an unsupervised neural network for inspection, we first have to set up the initial values of some parameters before we can progress. In our case, these parameters are the learning rate and threshold. For the learning rate, its purpose is to adapt the weight vector to a new pattern. The threshold means the acceptance criterion for a certain part. This paper will discuss the adjustments of these 2 parameters when the neural network is applied for IC leadframe inspection.

An IC leadframe is used to provide the mechanical support of the IC die and connections between the die and the pins of the integrated circuit. If there is any defects on the leadframe, the support of the integrated circuit or the connections will be weak or faulty. Therefore, we need to inspect the leadframe. There are various types of defects, which can be roughly classified as 2-dimensional and 3-dimensional ones. A complete inspection of all of them is time consuming and not practical. Therefore, common and critical defects are chosen here. In this paper we will concentrate on 2-dimensional shape distortion defects, which include planar rotation and width mismatches. They are the most common defects and their occurrences are easily to detect. To increase the production efficiency, an automatic inspection system is needed for this task. Here, we will use an unsupervised neural network, the LBE network, to perform the inspection because here we want the inspection process to adjust the acceptance criterion by itself. For the mechanical alignment of the leadframe,there is a pair of guide pins on it, and with them, the leadframe can be accurately positioned. First of all, the image of a whole leadframe is captured. From this image, small portions are then extracted. Those locations that defects commonly occur will be chosen for the extraction, and these extracted images are then input to the LBE network for inspection. After the inspection process, those defective leadframes will be differentiated from good ones.

In the learning process of an unsupervised neural network, memory is used to store the learned patterns inside the network. Commonly, it is represented by the weight vector to the output neuron. Hence, this weight vector will act as the centre of the cluster, and the boundary of this cluster will determine the class that is denoted by that particular output neuron[2]. In one way, the radius of the cluster comes from the threshold of the network. Throughout the learning process, the weight vector to the output neuron will be updated. Thus, the amount of learning at each iteration is governed by the learning rate. A large learning rate means that internal weights will adapt to the input pattern quickly. That is, a learning rate equal to 1 will change the internal weight equal to the new pattern just learned while a zero learning rate will not change it at all. Therefore, the learning capability of the network is minimized. Hence, the learning rate has to be set between 0 and 1. As the internal weight vector being updated, the cluster centre of the output neuron drifts accordingly, and the extent is proportional to the learning rate. This phenomenon can be explained graphically as follows. Fig. 1 shows an output neuron centred at W_0 with the cluster boundary C_0, which is the radius equal to the threshold. A new input pattern X_1 which lies within the cluster boundary, will be classified by that neuron, and the internal weight of that neuron will be updated. The new cluster centre is then located at W_1, which is represented by the dashed circle. The change of the distance, D, will then depend on the learning rate. Therefore, the centre of the output neuron drifts as the network learns. As the network continues to learn, the internal weights will drift continuously. A large learning rate will change them rapidly. If the cluster centre is too close to a pattern that is originally belonging to another cluster, then after a rapid change, this particular pattern will be grouped to it. Therefore, it may increase the chance of classifying defective parts as good ones. On the other hand, a small learning rate will slow down the learning process, which in turn will decrease the learning capability of the network. As a result, the learning rate is an important factor which will affect the inspection process. Hence, we need to determine the optimal learning rate under different operating conditions.

Apart from the learning rate, we have to consider one more parameter, the threshold. Usually, an unsupervised neural network will behave as follows. If the difference between the input and stored patterns is smaller than a threshold, it will be classified to the same class; otherwise a new class will be created to accommodate this new pattern. The network will then update the internal weight vector to reflect the arrival of this new pattern. Therefore, the classification of patterns depends on a set value of the threshold, which is related to a physical quantity among input patterns. If a tight threshold is set, we will have a network that tends to form more classes, which in turn needs more output neurons, and the network becomes memory full easily [3]. On the contrary, a loose threshold will group patterns with a large difference into the same class. Therefore, we have to determine an optimal threshold in order to compromise these two extreme effects.

The fitness or performance of a classification from an unsupervised network can be measured by a performance index. This index tries to quantify the error between the classification centres and the actual input patterns. A smaller performance index represents a smaller error between them, hence a better classification. Therefore, a smallest performance index will correspondingly locate the optimal values of the threshold and learning rate. In this paper, we shall propose a mechanism to establish a performance index and then base on this index to adjust the learning rate and the threshold. We propose to use fuzzy logic [4,5] to adjust these two parameters from the performance index formulated.

The layout of the paper is as follows. In Section 2, we shall describe the structure of the LBE network and its learning algorithm. Then we will describe the establishment of a performance index in Section 3. It is then followed by the investigation of the learning rate based on this performance index in Section 4. Then we reverse the process and evaluate the effect of the learning rate and threshold when using this performance index as a measure criterion. A fuzzy engine which is used as the adaptation mechanism will be described in Section 5. A conclusion will appear in Section 6.

2. Learning By Experience (LBE) Network

2.1 Structure of the LBE Network

Unlike other unsupervised learning neural networks, the main idea of LBE network is not the way of updating the weights of the winner, but instead, is to use memory to let output neurons know whether the applied input pattern belonging to the stored patterns or not. In fact, it employs the competitive learning as the updating rule for the winner.

First, a memory is added to each output neuron to store the strength, SS_j. This strength is used to compare the current input pattern with the stored patterns. As analog signals are applied to the network, the strength function is represented by the mean square error between the input pattern and the weight. Therefore, the lower the strength, the closer is the 2 compared patterns. The structure of the LBE network is shown in Fig. 2.

It is a two-layer, feed forward type network. The lower layer contains the input neurons and the upper layer consists of the output neurons. The connections among the lower layer and the upper layer store the weights, W_{ij}, which in turn store the learnt patterns. The connections within the upper layer indicate the output neurons, each receiving SS_j as the inhibitory input, from all the other output neurons. Also, each output neuron has a memory to store its experience. Once the output neuron has become the winner, the current strength SS_j will be stored into its memory. This memory is used to determine whether any applied pattern is associated to this output neuron.

When an input pattern is applied to the network, X_i will be equal to P_i, the input pattern. Then S_j will be built up through the network connections. When S_j is obtained, it will be compared with the strength stored in its internal memory M_j to work out SS_j. Finally the output neuron which has the minimum strength will become the winner, and its internal memory and the weights will be updated only.

2.2 The Learning Algorithm

2.2.1 Step 1 - the initialization phase

All weights, W_{ij}, are set to 0. All memory locations, M_j, are set to $(N + 1)$, where

N is the number of output neurons. The *learning rate* can be set to any value from 0 to 1. The *mean square error* can be set to any value from 0 to 1. This is the maximum allowable error between two input patterns to be classified to the same class.

2.2.2 Step 2 - the weight updating phase

Apply the pattern to the input neuron X_i of the network, then S_j is calculated as

$$S_j = \sum_{i=0}^{M} \frac{(W_{ij} - X_i)^2}{N} \tag{2}$$

where i = index of the input neurons,
j = index of the output neurons,
M = the no. of input neurons.

The final strength, SS_j, for competition within the upper layer is obtained as

$$SS_j = \begin{cases} S_j & \text{if } S_j \le M_j + \textit{mean square error} \\ \infty & \textit{otherwise} \end{cases} \tag{3}$$

The winner is searched among all the output neurons, and it is the one which has the minimum strength. Two cases will occur. They are: (1) *normal case* -- the winner with SS_j not equal to ∞. This means the input pattern is recognized; (2) *memory full case* -- the winner with SS_j equal to ∞. This means that there is no pattern stored in this network matched with this pattern and the network has no more unused output neuron to store this new pattern, which is a memory catastrophe. For normal case, the internal memory and the weights of the winner are updated as:

If $M_j = (N + 1)$ then

$$W_{ij} = X_i, \tag{4}$$

else

$$W_{ij} = W_{ij} + L \times (X_i - W_{ij}) \tag{5}$$

where j = index of the winner
L = learning rate

Next, we compute the S_j again and use these new weights, W_{ij}, to update the memory M_j as

$$M_j = S_j \tag{6}$$

and the output is given as

$$Y_j = \begin{cases} 1 & \text{if } j = index\ of\ winner \\ 0 & otherwise \end{cases} \tag{7}$$

3. Performance Index of the LBE Network

In industrial applications, when we want to classify patterns, such as the inspection of the IC leadframe, some patterns from a product are extracted first. These patterns are then input to an intelligent system for distinguishing as defective and non-defective. Then, we have the crucial question: Has the recognition system done a satisfactory job? Hence, here we suggest to establish a performance index to measure job done.

When an unsupervised neural network is used, some non-defective patterns are stored in each output neuron first. Then more patterns from different products are input to the network for classification. For those patterns not deviated from the good one too much, it will be grouped by an output neuron denoting that particular class. While patterns having too much deviation, the network will reject them, which in our case will be signified as a memory full phenomenon. The extent to which patterns deviate from the good one as being rejected is determined by the threshold of the network. If the threshold is set too small, the network will allow only a small difference between input patterns and patterns stored in the network. Therefore the network tends to reject more input patterns and form more clusters. A large threshold will let the network have a larger error. However, a defective one may be wrongly classified as non-defective. An extremely large threshold will allow all the patterns group as only one group, i.e., denoted by one output neuron only. Thus, it will then lose the classification function. There are another parameter which also will affect the inspection of input patterns -- the learning rate. The learning rate is used to control how fast the network adapts to input patterns. A small learning rate will put the network to follow the input patterns very slowly and thus will decrease its learning capability. On the contrary, for a large learning rate the network will adapt to the input pattern very fast. Hence, a large learning rate will increase the chance of classifying defective parts as good ones. Therefore, there are two parameters that will affect the classification of the input patterns. In order to control the learning process or to obtain the best classification of input patterns, we need to have to a yardstick for classification results. Then, we control the two parameters to have the best classification.

The fitness of a classification can be measured using a performance index. In the learning process of an unsupervised neural network, learned patterns are denoted by the weight vectors to output neurons. This weight vector also acts as the centre of the cluster, and with the threshold, this cluster centre will group all similar patterns. On the side, we can consider that the difference between the actual input pattern and the cluster centre is the error of the classification. Without much arguments, we use the Euclidean distance to represent this difference. Therefore, we define a performance index as

$$P = \sum_{i=1}^{c} \sum_{k=1}^{n} \|x_k - v_i\|^2 + \sum_{l=1}^{c} \sum_{j=1}^{m} \|x_j - v_l\|^2 \tag{8}$$

where

x_k is the vector representation of the input pattern,
v_i is the weight vector of the cluster centre,
c is the number of clusters,
n is the number of patterns stored in cluster i,
m is the number of patterns being unclassified,
and $\|x_k - v_i\|$ represents the Euclidean distance between x_k and v_i

The first term in Equ. 8 sums up all the differences between the learned patterns which are stored in output neurons, and it only has values on those input patterns which are classified successfully. For those unclassified patterns, they are not in favour to the classification process, so we need to add some penalty to the performance index from them. Therefore, we add up all the distances from each unclassified pattern to each output neuron. Since the performance index is the sum of all the errors between input patterns and patterns stored in the network, the smaller the performance index, the better the classification. At this stage, we move on to elaborate the incorporation of this performance index to the LBE network for classification.

4. The Change of the Learning Rate and Threshold With the Performance Index

In Section 3, we have discussed that the learning rate and the threshold will affect the classification of input patterns. Apart from assigning values to these two parameters, we also want to obtain the best classification, which corresponds to the smallest performance index. Therefore, in this section, we will investigate the effect of the learning rate and threshold on the performance index. The result can thus guide us to set an optimal value for them. At this stage, we have no intention to provide a mathematical model of changing these 2 parameters on the performance index, instead, we shall provide some simulations to demonstrate this effect when inspecting IC leadframes. Fig. 3 shows an image of the whole IC leadframe. Two locations are selected for the inspection of defects. Also we extract another 13 images at these two locations from different leadframes. Each image is 24×24 pixels in size and they are shown in Fig. 4. Here, image 0 and image 9 are extracted from a good IC leadframe. Image 1 to 8 are deviated from image 0 by rotating some angles clockwise or anti-clockwise. Image 1 differs from image 0 by rotating 2^o clockwise, whereas image 2, 5, 6, 7 and 8 have differences by rotating 4, 6, 8, 10 and 20 degrees respectively. While image 10 and 11 are 3 and 6 pixels thicker than image 9 and image 12 are 3 pixels thinner.

Now we vary the learning rate from 0.1 to 0.9 in steps of 0.1, and notice the change in the performance index. The network is configured to have 2 output neurons since these images will be grouped into two categories only if they all are good. The threshold is set as 0.03. Fig. 5 shows the performance index against the learning rate. From it, we notice that the performance index increases as the learning rate increases. But, as the learning rate increases, the memory wash away effect will also increase as drifting of the cluster centre becomes more rapidly. The distance between each input pattern and the cluster centre changes all the time. When the learning rate is larger than 0.6, some previously classified images become unclassified, leading to the increase in the performance index. As a result, the performance index will increase as the learning rate increases.

Now we vary the threshold instead of the learning rate. The learning rate is now set as 0.1 to make its effect on the performance index become pronouncing. We change the threshold from 0.005 to 0.03 in steps of 0.005. Fig. 6 shows the performance index against the threshold. Here, we find that the performance index decreases as the threshold increases. The reason is that, for a small threshold, the error allowed for each output neuron is smaller. In fact, the number of unclassified images increases, so as the performance index. As the threshold is increased gradually, the number of unclassified images reduces. It also occurs in the value of the performance index. When the threshold is 0.03, all unclassified images are successfully classified. Therefore, the performance index is the smallest when the threshold is 0.03. We then vary the learning rate and the threshold to obtain 54 sets of combinations and notice the change in the performance index. The results are shown in Table 1, and these 6 sets of results are plotted 3-dimensionally in Fig. 7. From it, we find that the smallest performance index (corresponding to the best classification in a mathematical sense) is obtained by setting the threshold to 0.03 and the learning rate to 0.1. From these simulation results, we come to know that a large threshold with a small learning rate can give a better classification. However, for a large threshold, those defective images such as image 8 and 11, are classified as good (they are defective because they deviate from their good counterparts too much). Therefore, we need to decrease the threshold to 0.025 instead of 0.03, whereas the learning rate is set to 0.1. The new classification results are shown in Table 2. An * in the table indicates that the image is unclassified and thus will be rejected by the network. Therefore, there is no simple method to determine the learning rate and threshold for the best classification. Nevertheless, we will describe the use of fuzzy logic to determine the learning rate and threshold in the following section.

Same simulations are performed using the Fuzzy C-Means algorithm [5,6] in order to have a comparison between it and the LBE network when inspecting IC leadframes. We choose it because it is a well-established algorithm for clustering applications. In this application, we need to form some clusters that can properly represent different classes of input patterns. The criterion is that these classes are required to form a partition of input patterns such that the degree of association is strong for data within the same cluster and weak in other clusters. However, this requirement is too stringent in many practical applications, and thus it is more desirable to replace it with a weaker requirement. Therefore, fuzzy partition is used to replace the crisp partition. As a result, fuzzy partition is superior to crisp partition in many practical clustering applications. For the integrity of this paper, we will describe the Fuzzy C-Means algorithm here, which is excerpted from [5].

4.1 Fuzzy C-Means Algorithm

Given a set of data $X = \{x_1, x_2, ..., x_n\}$, where x_k, in general, is a vector $x_k = \{x_{k1}, x_{k2}, \ldots, x_{kp}\}$, the fuzzy clustering is to find a fuzzy pseudopartition and the associated cluster centers by which the structure of the data is represented as best as possible. This requires some criteria expressing the general idea that associations are strong within clusters and weak between clusters. To solve the problem of fuzzy clustering, we need to formulate a criterion in term of a performance index. Usually, the performance index is based upon cluster centers. Given a pseudopartition $P = \{ A_1, A_2, \dots, A_c \}$, the c cluster centers, $v_1, v_2, \dots, v_c$ associated with the partition are given by the following equation

$$v_i = \frac{\sum_{k=1}^{n} \left[A_i(x_k) \right]^m x_k}{\sum_{k=1}^{n} \left[A_i(x_k) \right]^m} \tag{9}$$

where $m > 1$ is a real number that governs the influence of membership grades. The performance index of a fuzzy pseudopartition P, $J_m(P)$, is defined in terms of the cluster centers by Equ. 10 as below,

$$J_m(P) = \sum_{k=1}^{n} \sum_{i=1}^{c} \left[A_i(x_k) \right]^m \| x_k - v_i \|^2 \tag{10}$$

$\| x_k - v_i \|$ represents the distance between x_k and v_i. This performance index measures the weighted sum of distances between cluster centers and elements in the corresponding fuzzy clusters. The Fuzzy C-Means algorithm is to find a fuzzy pseudopartition P that minimizes the performance index $J_m(P)$. The step of the algorithm is listed below

Step 1. Let $t = 0$. Select an initial fuzzy pseudopartition $P^{(0)}$.

Step 2. Calculate the c cluster centers $v_1^{(t)}, \dots, v_c^{(t)}$ with Equ. 9 for $P^{(t)}$ and the chosen value of m.

Step 3. Update $P^{(t+1)}$ as follows: For each $x_k \in X$, if $\| x_k - v_i^{(t)} \|^2 > 0$, then define

$$A_i^{(t+1)}(x_k) = \left[\sum_{j=1}^{c} \left(\frac{\| x_k - v_i^{(t)} \|^2}{\| x_k - v_j^{(t)} \|^2} \right)^{\frac{1}{m-1}} \right]^{-1} \tag{11}$$

if $\| x_k - v_i^{(t)} \|^2 = 0$ for some $i \in I$, then define $A_i^{(t+1)}(x_k)$ for $i \in I$ by any nonnegative real numbers satisfying $\sum_{i \in I} A_i^{(t+1)}(x_k) = 1$, and define

$A_i^{(t+1)}(x_k) = 0 \quad \text{for } i \notin I.$

Step 4. Compare $P^{(t)}$ and $P^{(t+1)}$. If $|P^{(t+1)} - P^{(t)}| \le \varepsilon$, then stop; otherwise, increase t by one and return to step 2, where ε is a small positive number serving as a stopping criterion. $|P^{(t+1)} - P^{(t)}|$ denotes a distance between $P^{(t)}$ and$P^{(t+1)}$. An example of this distance is

$$|P^{(t+1)} - P^{(t)}| = \max_{i,k} \left| A_i^{(t+1)}(x_k) - A_i^{(t)}(x_k) \right|$$

The same input patterns in Section 4 are input to the Fuzzy C-Means algorithm for classification. The classification result is tabulated in Table 3. From it, we find that the Fuzzy C-Means algorithm classifies both image 8 and 11 as good ones even though they are defective. Since the Fuzzy C-Means algorithm will classify all patterns and will not reject any patterns, all patterns will eventually be grouped together. Then we increase the number of clusters to 3 to hope that the algorithm will classify those defective parts into a single cluster. The results are shown in Table 4. Here, image 8 and 11 are still wrongly classified as non-defective, and image 2, 5 to 7 are classified to cluster 2 rather than cluster 0. Therefore, it cannot distinguish between defective and non-defective images in this application even though we increase the number of clusters. Thus, we find that the LBE network is better than the Fuzzy C-Means algorithm for this application because it needs to group all defective ones as a single group.

5. Fuzzification of the Variables

As described in Section 4, the fitness of the classification or the performance index depends on two factors, the learning rate and the threshold. In order to obtain a better classification, we have to determine a required value for the threshold and learning rate. From Section 4, we know that a better classification may be acquired from a lower learning rate but a larger threshold, and there is no simple method to determine how small or how large their values. As a result, we use a fuzzy engine to dynamically adjust their values from the information of the performance index. Putting it in other words, the goal of the fuzzy engine is to obtain a classification with the lowest performance index.

First, we list the abbreviations used as belows:

ZE:	ZERO
SM:	SMALL

ME: MEDIUM
LA: LARGE
VL: VERY LARGE

We use the performance index as the input fuzzy variable to an inference engine to determine the values of the threshold and learning rate. Hence, we start to construct the fuzzified membership graphs of these variables.

5.1 Performance index

First we need to normalize the performance index using Equ. 12.

$$P_{ratio} = \frac{P}{\sum_{i=1}^{c} \sum_{j=1}^{n} \|x_j - v_i\|^2} \qquad (12)$$

where

P is the original performance index given by Equ. 8,
P_{ratio} is the normalized performance index,
x_j is the vector representation of the input pattern,
v_i is the weight vector of the cluster centre,
c is the number of clusters,
n is the number of input patterns,
and $\|x_j - v_i\|$ represents the Euclidean distance between x_j and v_i

After normalization, the performance index ranges from 0 to 1, so we divide the it into 5 fuzzy set values evenly, namely:- **ZERO, SMALL, MEDIUM, LARGE, VERY LARGE**. Fig. 8 shows the membership function of the normalized performance index.

5.2 Threshold

From the input fuzzy variable -- normalized performance index, an inference engine can determine the value of threshold. From the simulation results in Section 4, we observe that the threshold required to classify the input patterns is about 0.02. In addition, too large a threshold such as 1 is not practical in real applications. Therefore, we limit the threshold ranging from 0 to 0.1. Thus we divide it into 5 fuzzy set values namely**:- ZERO, SMALL, MEDIUM, LARGE** and **VERY LARGE**. Fig. 9 shows its membership function.

5.3 Learning Rate

From the simulation results in Section 4, a large learning rate will increase the performance index, while a learning rate equal to 1 is not practical in real applications. Therefore, we limit its value ranging from 0 to 0.2, and we divide it into 5 fuzzy set values namely:- **ZERO, SMALL, MEDIUM, LARGE** and **VERY LARGE**. Fig. 10 shows its membership function.

After defining the ranges of the fuzzy variables, we have to build the rule matrix of the inference engine. There is only 1 input variable and five rules for each output variable. If the performance index is large, we have to increase the threshold. We tabulate the rule matrix for the threshold as Table 5. On the other side, if the performance index is large, we have to reduce the learning rate. Therefore, the rule matrix for the learning rate is shown in Table 6. With these rules, the inference engine first finds out the scalar activation value w_i for each rule. From these activation values, we can then work out the output fit vector, and the values of the threshold and learning rate are thus calculated using a defuzzification algorithm - the centroid of the output fit vector.

The same input patterns used in Section 4 is again input into the LBE network with the fuzzy inference engine. The initial values of the threshold and learning rate are set as 0.001 and 0.0001 respectively. The simulation results are the same as the one using the original LBE network without the fuzzy engine, that is, image 8 and 11 are rejected by the network since these two images deviate from the non-defective too much. The final threshold and learning rate are 0.025 and 0.1 respectively, and the performance index is 237.64. We try different initial settings of the threshold and learning rate, but the same results are obtained. Even we try to set the initial threshold to 0.1, which is larger than the one required to classify all input patterns including the defective ones, we again get the same results with the final threshold equal to 0.025 and the learning rate equal to 0.1. At this stage, the performance index equals 237.78. We can observe from the simulation results that the fuzzy inference engine automatically adjusts the threshold and learning rate from the performance index in spite of all these initial settings.

We extend the application of the fuzzified LBE network to increase the number of images to 18, which are extracted from 5 different locations. Therefore, the number of clusters has been increased to 5. Fig. 11 shows the original IC leadframe with 5 different locations, from which the other images are extracted. These 5 images are 0, 9, 13, 15, and 16. In Fig. 12, all these 18 images are shown. The first

13 images are the same as the one used in Section 4. Image 14 is 2^o rotated clockwise from image 13, and image 17 is 5^o rotated clockwise from image 16. Therefore, we shall expect that image 14 and 17 should be classified to image 13 and 16 respectively while the others remain unchanged. The initial settings of the threshold and learning rate are both at 0.0001. The classification results are tabulated in Table 7. The simulation results meet our expectation. The final performance index is 604.6 and the final threshold and learning rate is 0.025 and 0.1 respectively. Therefore, the fuzzy engine incorporated in the LBE network can be used to dynamically adjust the threshold and learning rate during the learning process so as to obtain a better classification result.

6. Conclusion

In this paper, we have illustrated a relationship among the learning rate, threshold and the performance index of an unsupervised neural network. The performance index is found to be proportional to the learning rate. We can obtain a small performance index with a small learning rate, but the learning capability of the network will be reduced. In addition, we can obtain a smaller performance index with a larger threshold. Therefore, we need to set an optimal learning rate and threshold in order to obtain the best classification. We demonstrate the use of an fuzzy inference engine to control the threshold and learning rate from the performance index. We also apply the LBE network with fuzzy inference engine for inspecting IC leadframes. From the simulations, the best classification is not solely represented by the performance index. Since the performance index is the smallest when all the patterns are being classified. But then, all the defective images will be wrongly classified to be non-defective. Therefore, classification so far obtained may not be the one with the smallest performance index, but it is the best classification result according to the input patterns with a satisfactory performance index. In addition, a comparison with the Fuzzy C-Means algorithm has been illustrated. We find that the LBE network is better than Fuzzy C-Means algorithm in this application of IC leadframe inspection.

References

1. C. K. Lee, C. H. Chung, "An unsupervised neural network with a memory replacement effect," *IEEE ICNN'94*, June 26 - July 2, 1994, vol. 2, pp. 675-680, Orlando, USA.
2. C. W. Therrien, *Decision Estimation and Classification: An Introduction to Pattern Recognition and Related Topics*. John Wiley & Sons, 1989.
3. C. K. Lee, C. H. Chung, "An unsupervised neural network for machine part

recognition with constraint release," *IEEE ETFA'94*, 6-10 Nov., 1994, pp. 166-173, Tokyo, Japan.

4. B. Kosko, *Neural Networks and Fuzzy Systems, A Dynamical Systems Approach to Machine Intelligence*, Prentice Hall, 1994.
5. George J. Klir, Bo Yuan, *Fuzzy Sets and Fuzzy Logic, Theory and Applications*, Prentice Hall, 1995.
6. Timothy J. Ross, *Fuzzy Logic with Engineering Applications*, McGraw-Hill, 1995.

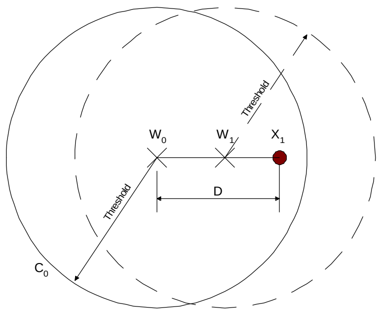

Fig. 1 The drifting of weights as the input pattern changes.

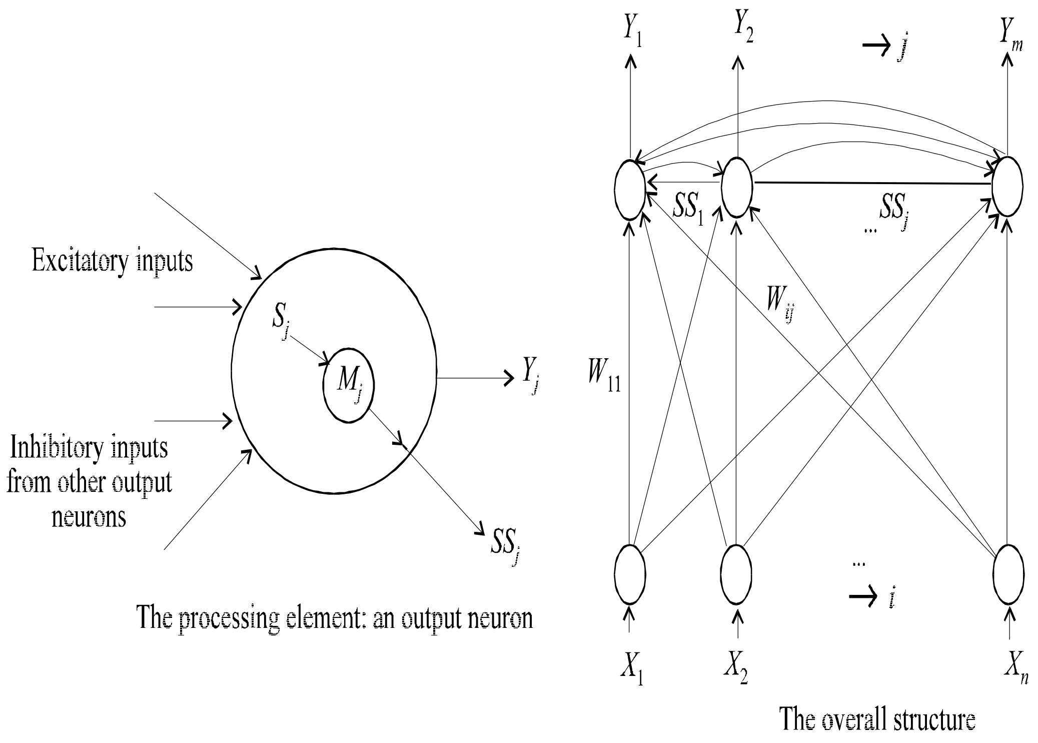

Fig. 2 The structure of the neural network

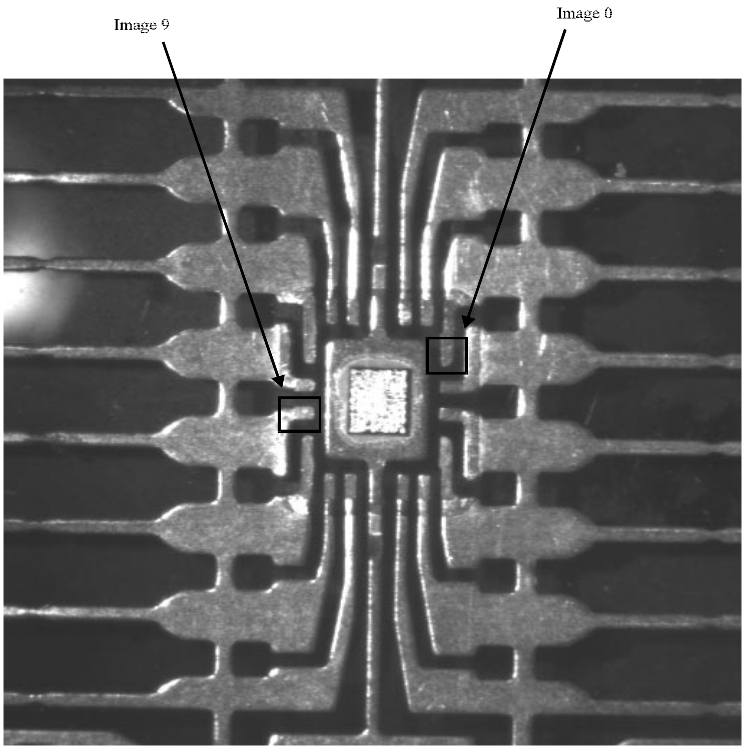

Fig. 3 An IC leadframe with the 2 locations

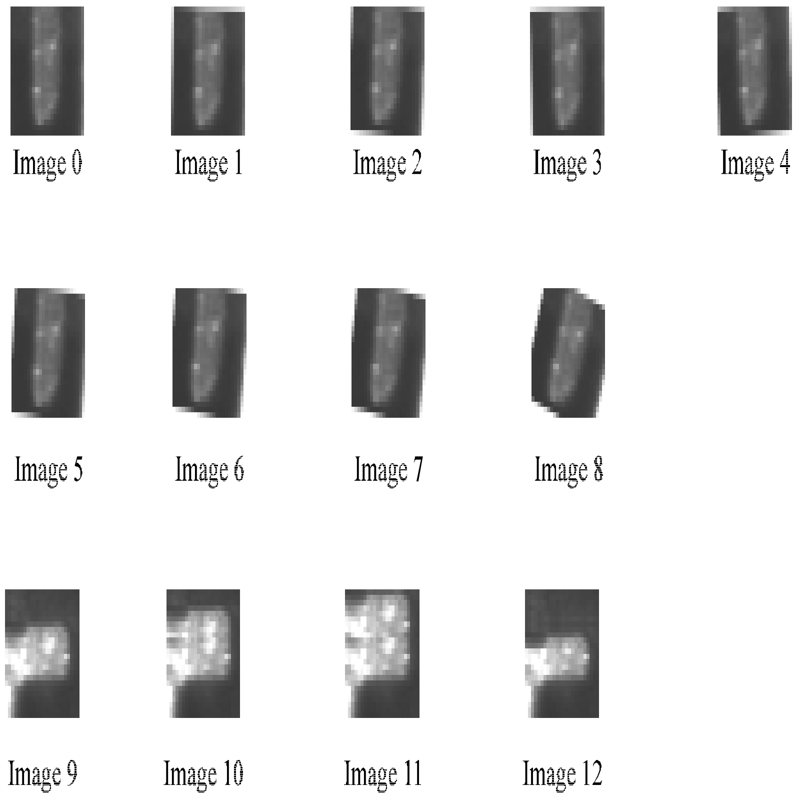

Fig. 4 Portions of the leadframe for inspection

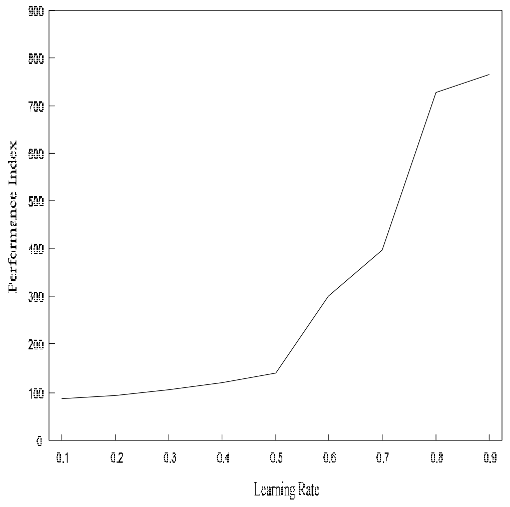

Fig. 5 Performance index vs the learning rate

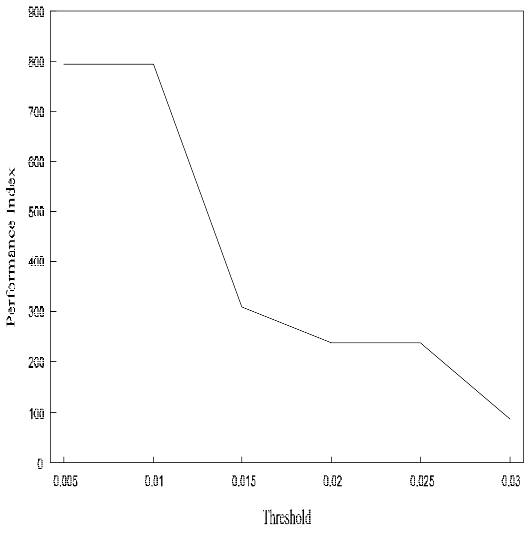

Fig. 6 Performance index vs the threshold

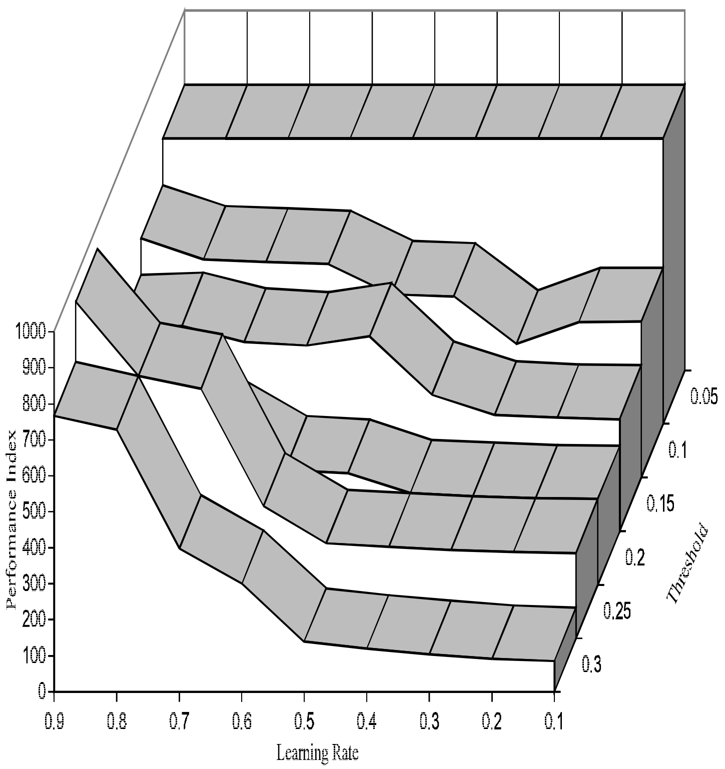

Fig. 7 Performance index vs the learning rate and threshold

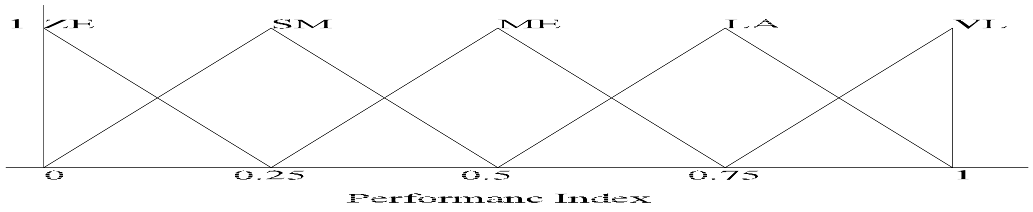

Fig. 8 Membership graph for the performance index

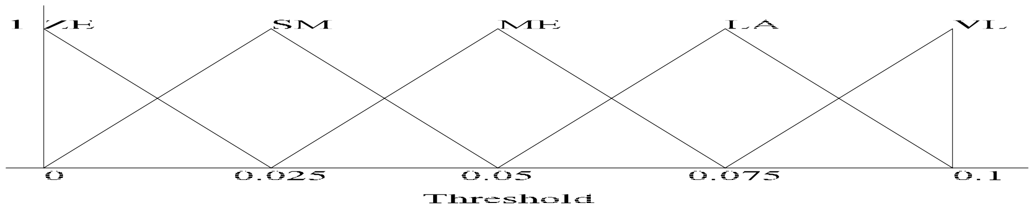

Fig. 9 Membership graph for the threshold.

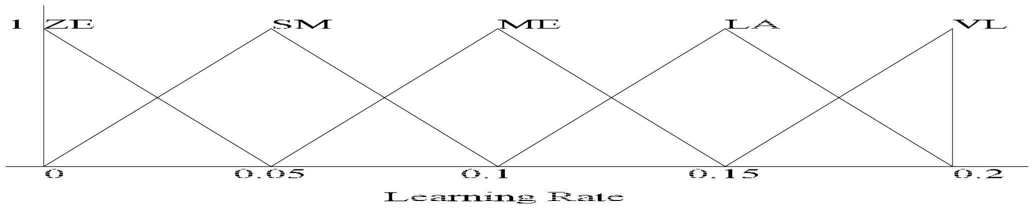

Fig. 10 Membership graph for the learning rate

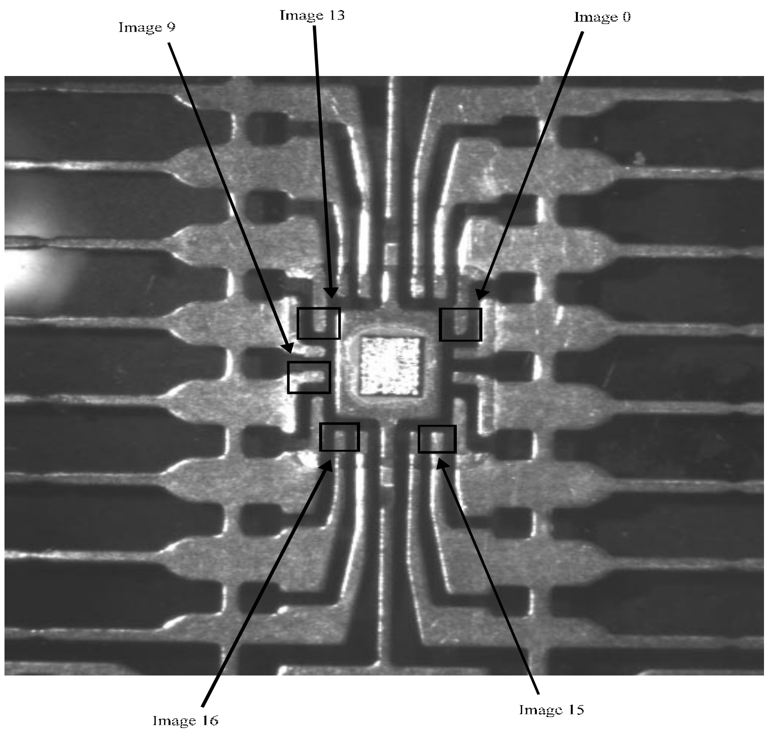

Fig. 11 An IC leadframe with 5 locations for inspection

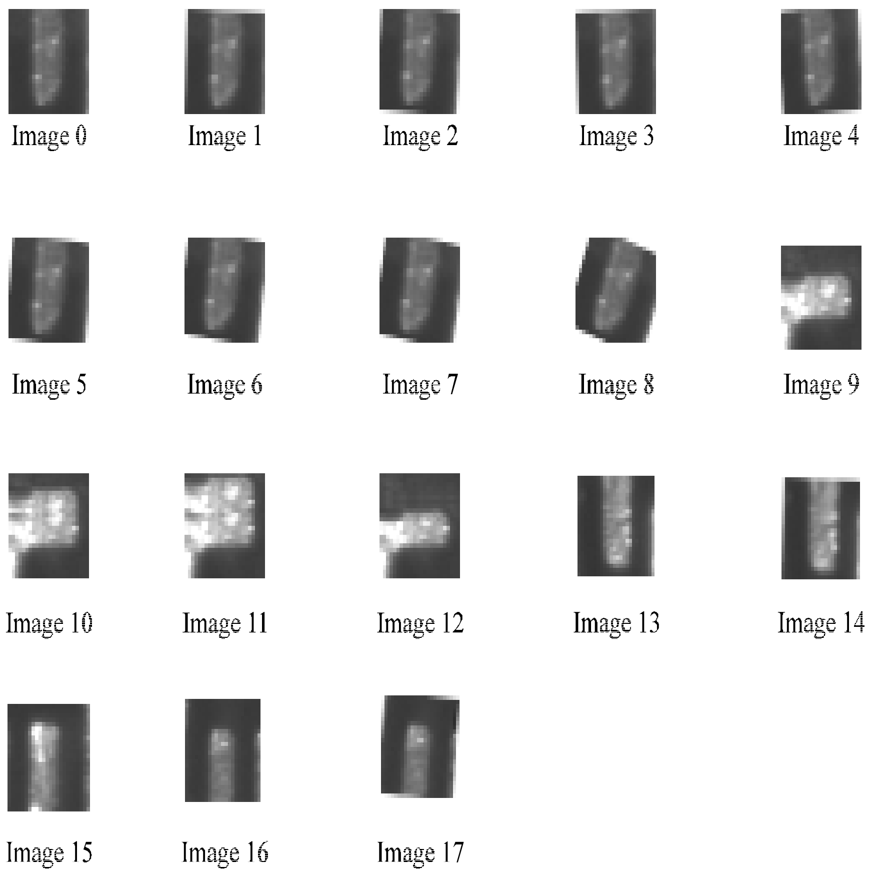

Fig. 12 18 portions of the leadframe for inspection

Performance Index	Learning Rate								
Threshold	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0.005	793.7	793.7	793.7	793.7	793.7	793.7	793.7	793.7	793.7
0.01	433.5	432.5	371.3	503.4	508.9	593.1	598.6	605.5	663.4
0.015	311.1	315.1	322.3	378.7	541.5	516.1	524.5	569.0	562.2
0.02	237.5	239.9	245.2	253.3	309.8	319.3	419.7	548.2	564.2
0.025	237.5	239.9	245.2	253.3	263.2	365.8	694.0	727.6	932.7
0.03	85.3	91.8	103.6	119.7	138.8	300.1	397.6	727.6	766.9

Table 1 Performance index vs the threshold and learning rate

Image	0	1	2	3	4	5	6	7	8	9	10	11	12
Class	0	0	0	0	0	0	0	0	*	1	1	*	1

Table 2 Classification results using the LBE network

Image	0	1	2	3	4	5	6	7	8	9	10	11	12
Class	0	0	0	0	0	0	0	0	0	1	1	1	1

Table 3 Classification results using the Fuzzy C-Mean

Image	0	1	2	3	4	5	6	7	8	9	10	11	12
Class	0	0	2	0	0	2	2	2	2	1	1	1	1

Table 4 Classification results using the Fuzzy C-means with 3 clusters

Performance Index	ZE	SM	ME	LA	VL
Threshold	SM	SM	SM	ME	ME

Table 5 Rule matrix for the threshold

Performance Index	ZE	SM	ME	LA	VL
Learning Rate	ME	ME	ME	SM	SM

Table 6 Rule matrix for the learning rate

Image	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Class	0	0	0	0	0	0	0	0	*	1	1	*	1	2	2	3	4	4

Table 7 Classification results using the LBE network