A fuzzy-soft learning vector quantization for control chart pattern recognition

MIIN-SHEN YANG* and JENN-HWAI YANG†

This paper presents a supervised competitive learning network approach, called a fuzzy-soft learning vector quantization, for control chart pattern recognition. Unnatural patterns in control charts mean that there are some unnatural causes for variations in statistical process control (SPC). Hence, control chart pattern recognition becomes more important in SPC. In order to detect effectively the patterns for the six main types of control charts, Pham and Oztemel described a class of pattern recognizers for control charts based on the learning vector quantization (LVQ) such as LVQ, LVQ2 and LVQ-X etc. In this paper, we propose a new supervised LVQ for control charts based on a fuzzy-soft competitive learning network. The proposed fuzzy-soft LVQ (FS-LVQ) uses a fuzzy relaxation technique and simultaneously updates all neurons. It can increase correct recognition accuracy and also decrease the learning time. Comparisons between LVQ, LVQ-X and FS-LVQ are made. Numerical results show that the proposed FS-LVQ has better accuracy and less learning epochs for all neurons being completely learned than LVQ and LVQ-X. Overall, FS-LVQ is highly recommended to be used as a control chart pattern recognizer.

1. Introduction

Quality and productivity are two essential factors for survival in the global economy. Statistical process control (SPC) is a method to improve the quality of products and reduce rework and scrap so that the quality and productivity expectations can be met. Control charting is the most important part of SPC. Shewhart (1931) control charts have been the most popular and widely used charts in industry to provide the capability to detect unnatural process behaviour. The most typical form of control chart consists of a centre line corresponding to the average statistical level and two control limits normally located at $\pm 3\sigma$ of this statistic, where $\sigma$ is a measure of the spread, or standard deviation in a distribution. A control chart can be used to indicate whether a manufacturing process is in control or not (see Montgomery, 1997). However, these Shewhart-type control charts do not provide any pattern-related information because they focus only on the latest plotted data points and seem to discard useful information contained in previous points. Although other improved control chart varieties give more powerful detection ability, such as the CUSUM chart etc, they still lack a pattern recognition capability (see Cheng 1997, and Guh and Hsieh 1999).

In general, there are six main types of patterns in control charts, i.e. normal, upward trend, downward trend, upward shift, downward shift and cycle (see Grant...
and Leavenworth 1988). These patterns can present the long-term behaviour of a process. Thus, the construction of a pattern recognition system for control charts is quite important. Recently, there have been many studies on control chart pattern recognition (see Chang and Aw 1996, Cheng 1997 and Smith 1994). In order to detect effectively these six main control chart patterns, Pham and Oztemel (1994) described a class of pattern recognizers for control charts based on supervised learning vector quantization (LVQ) such as LVQ, LVQ2 and LVQ-X etc. In this paper, we propose a new supervised LVQ for control charts based on a fuzzy-soft competitive learning network.

Kohonen’s (1990) self-organizing map (SOM) is a well-known unsupervised learning neural network. It has been used as a (sequential) learning algorithm for vector quantizer design. The traditional $k$-mean (or so-called hard $c$-mean) has been used as a batch algorithm for vector quantizer design. We know that the fuzzy $c$-mean (FCM) is a fuzzy extension of the hard $c$-mean (see Bezdek 1981, Yang 1993). Combining FCM with Kohonen’s SOM, Bezdek and Pal (1995) and Tsao et al. (1994) proposed the so-called fuzzy Kohonen’s clustering network and fuzzy unsupervised learning vector quantization. However, they are still batch-type algorithms. Recently, Wu and Yang (2002) proposed a fuzzy-soft competitive learning network by modifying the neighbourhood interaction function and the learning rate in Kohonen’s SOM. With a fuzzy-soft relaxation technique, using the FCM membership functions as a kernal type of neighbourhood interaction functions, they created a fuzzy-soft unsupervised learning vector quantization method (FS-U-LVQ). This FS-U-LVQ has a more reasonable learning rate and better performance than the other competitive learning algorithms. Of course, it is a sequential type of unsupervised learning that is different from the batch type of Bezdek and Pal (1995) and Tsao et al. (1994) in which they do not have sequential learning behaviour.

In this study, we used the fuzzy-soft competitive learning network to create a supervised fuzzy-soft learning vector quantization (FS-LVQ). The proposed FS-LVQ is applied to control chart pattern recognition. Since the proposed FS-LVQ uses a fuzzy relaxation technique and updates all neurons simultaneously, it can increase the correctly recognized accuracy and also decrease the learning time with less learning epochs. Comparisons between LVQ, LVQ-X and FS-LVQ are made, and numerical comparisons give impressive results. Section 2 presents control chart pattern recognizers using the LVQ network. A supervised FS-LVQ for control charts is created in section 3. Some numerical results and comparisons between LVQ, LVQ-X and FS-LVQ are made in section 4. Conclusions are made in section 5.

2. Control chart pattern recognizers using LVQ

In order to ensure that products have an acceptable quality in the industry, control charting is a tool frequently used to monitor whether the production process is in control. This is an important part of SPC. Furthermore, a point located outside the control limits is interpreted as the process being out of control. We know that, in a normal case, the points plotted on the control chart should not have any pattern. In many cases, a control chart that presents some kind of pattern may provide some information about the process. If control chart patterns can be detected, actions can be taken as early as possible to avoid an out-of-control process. However, most control charting techniques lack this pattern recognition ability.

In general, there are six main patterns with one normal pattern and five abnormal patterns, i.e. upward shift, downward shift, upward trend, downward trend and cycle
The normal pattern is due to random noise and is inevitable. It is therefore called ‘unassignable cause’. Shift patterns may result from new workers, raw materials, machines or methods. Trend patterns are usually due to gradual tool wear. A cyclic pattern is caused by systematic environmental changes such as temperature, operator fatigue or some other equipment variable. Owing to the above assignable causes, we would like to be able to detect these patterns before the process becomes out of control. Recently, there have been many studies on pattern recognition design for control charts. In this paper, we will focus on a pattern recognizer for control charts using learning vector quantization and its improvement types to achieve this objective.

In recent years, neural network techniques have been widely applied to pattern recognition. Learning vector quantization (LVQ) is one of the competitive learning networks first presented by Kohonen in 1984. As its name indicates, the LVQ model is used to classify patterns through learning. Essentially, learning will help us to find representative prototypes called ‘reference vectors’ so that we can obtain a correct category after inputting a pattern into the network.

The LVQ network structure includes three layers: an input layer, which conveys input data to the network; a hidden layer, which deals with the actual data information, and an output layer, which generates a category for the input (see figure 2). The input layer is connected to the hidden layer through weights, the so-called reference vectors, which are updated through the competitive learning process. The hidden layer to output layer are partially connected and the weights are fixed at 1.

2.1. Original LVQ learning procedure

In order to obtain a better classification performance, the learning procedure is a very important stage. In general, the Euclidean distance is adopted as a basic rule of competition. The distance $d_i$ between the reference vector $Z_i$ of neuron $i$ and the training vector $X$ is given by

$$d_i = \|Z_i - X\| = \sqrt{\sum_j (Z_{ij} - X_j)^2},$$

(1)

Figure 1. Six main patterns.
where $Z_{ij}$ and $X_j$ are the $j$th elements of $Z_i$ and $X$, respectively. The neuron wins the competition when it has the minimum distance and only this neuron winner is permitted to modify the connection weights in each learning iteration.

The learning formula for updating the reference vector is given as follows.

If the neuron is in the correct category, then

$$Z_i(t + 1) = Z_i(t) + \lambda(t) h_i(t)(X(t) - Z_i(t)),$$

(2)

and if the neuron is in the wrong category, then

$$Z_i(t + 1) = Z_i(t) - \lambda(t) h_i(t)(X(t) - Z_i(t)),$$

(3)

where

$$h_i(t) = \begin{cases} 1, & \text{if the } i\text{th neuron is a winner,} \\ 0, & \text{otherwise,} \end{cases}$$

(4)

denotes the degree of excitation of the neurons. $\lambda(t)$ is the learning rate which is decreasing monotonically. The main idea behind equations (2) and (3) is that the weight is updated to be close to the training vector if the winning neuron is in the correct category, and far away from the training vector if the winning neuron is in the wrong category. Equation (4) shows that only the winning neuron is qualified to be modified with the so-called ‘winner-take-all’ competition.

2.2. LVQ-X learning procedure

Notice that in LVQ some neurons may win too often while others are always inactive. This means that only a few neurons have been learning. This situation may occur when the initial vector is far from the training vector or there are some degrees of similarities between the patterns for classification.
To solve this kind of problem, Pham and Oztemel (1994) proposed an extended version of the LVQ learning procedure, called an LVQ-X learning procedure. This approach uses two weight vectors to be modified at the same time. The first is called the 'global winner', which is the one globally nearest to the training vector. The second is called the 'local winner' and is the one nearest to the training vector in the correct category. The weights are modified as follows.

If the global winner is also the local winner, then
\[ Z_i(t + 1) = Z_i(t) + \lambda(t) h_i(t)(X(t) - Z_i(t)), \]
if the global winner and the local winner are different, then
\[ Z_i(t + 1) = Z_i(t) - \lambda(t) h_i(t)(X(t) - Z_i(t)) \]  
(5)

(6)

(7)

(8)

This approach gives an opportunity for the correct neuron to win in the next iteration. In Pham and Oztemel (1994), numerical comparisons showed that LVQ-X has a better classification accuracy within a shorter training time than LVQ and its two varieties of LVQ2 and LVQ with a conscience mechanism.

3. A new LVQ learning procedure

There are, at most, two weights to be modified for the LVQ and LVQ-X methods as discussed in section 2. In this section, we propose a new supervised LVQ based on a fuzzy-soft competitive learning network. The proposed fuzzy-soft LVQ (FS-LVQ) uses a fuzzy relaxation technique and updates all weights simultaneously according to the information between the reference and training vectors. This method results in an increase in recognition accuracy and a decrease in learning time.

Kohonen’s SOM is a two-layer feedforward competitive learning network. The SOM learning uses a neighbourhood interaction set to approximate the lateral neural interaction phenomenon. At each iteration \( t \) with a training vector \( X(t) \), the SOM updates the reference vector \( Z_i(t + 1) \) for each neuron \( i \) with the following learning formula
\[ Z_i(t + 1) = Z_i(t) + \alpha_i(t) h_i(t, j^*)(X(t) - Z_i(t)), \quad i = 1, \ldots, c, \]
\[ j^* = \arg \min_{1 \leq j \leq c} \|X(t) - Z_j(t)\|, \]
where \( \alpha_i(t) \) is a learning rate and \( h_i(t, j^*) \) is a neighbourhood function of the winner \( j^* \). In 1992, Yair et al. proposed a soft competition scheme to modify the SOM learning using the following learning formula
\[ Z_i(t + 1) = Z_i(t) + \alpha_i(t) P_i(t)(X(t) - Z_i(t)), \quad i = 1, \ldots, c \]
in which \( \alpha_i(t) \) and \( P_i(t) \) are given by
\[ \alpha_i(t) = 1/n_i(n) \]
and

\[ P_i(t) = \frac{e^{-\beta(t)}\|X(t) - Z_i(t)\|^2}{\sum_{j=1} e^{-\beta(t)}\|X(t) - Z_j(t)\|^2} . \]  

(12)

In order to have a more reasonable learning rate and better performance, Wu and Yang (2002) recently proposed a fuzzy-soft competitive learning network by modifying the learning formula to be

\[ Z_i(t + 1) = Z_i(t) + \alpha_i(t) h_i(t)(X(t) - Z_i(t)) , \quad i = 1, \ldots, c \] 

(13)
in which

\[ \alpha_i(t) = \frac{\alpha_0}{\alpha_i(t-1) + h_i(t)} , \] 

(14)

and

\[ h_i(t) = \left( \frac{\mu_i(X(t))}{\max_{1 \leq j \leq c} \mu_j(X(t))} \right)^{(1+f(t)/c)} . \] 

(15)

with

\[ \mu_i(X(t)) = \left( \sum_{k=1}^c \frac{\|X(t) - Z_i(t)\|^{2/(m-1)}}{\|X(t) - Z_k(t)\|^{2/(m-1)}} \right)^{-1} . \] 

(16)

Note that \( \mu_i(X(t)), \ i = 1, \ldots, c \) are the well-known FCM membership functions. In general, \( m = 2 \) is chosen. The function \( f(t) \) is a monotonically increasing function of \( t \) that controls the approximated excitation of neurons. In general, \( f(t) \) can be chosen with \( \sqrt{t}, t, \) and \( t^2 \) etc. This learning algorithm is called a fuzzy-soft unsupervised LVQ (FS-U-LVQ). Wu and Yang (2002), described its properties and showed impressive numerical results.

We used these fuzzy-soft competitive learning equations (13)–(16) to create a supervised fuzzy-soft LVQ (FS-LVQ) using the following update equations.

If the neuron \( i \) is in the correct category, then

\[ Z_i(t + 1) = Z_i(t) + \alpha_i(t) h_i(t)(X(t) - Z_i(t)) , \] 

(17)

if the neuron \( i \) is in the wrong category, then

\[ Z_i(t + 1) = Z_i(t) - \beta \alpha_i(t) h_i(t)(X(t) - Z_i(t)) , \] 

(18)

where the learning rate \( \alpha_i(t) \) and the neighbourhood function \( h_i(t) \) are given by equations (14), (15) and (16). Obviously, \( h_i(t) \) will be similar to winner-take-all learning as \( t \) is large enough. That is

\[ \lim_{t \to \infty} h_i(t) = \begin{cases} 1, & \text{if neuron } i \text{ is winner,} \\ 0, & \text{otherwise.} \end{cases} \]

\( f(t) \) may help us to control how fast FS-LVQ becomes LVQ. We use \( f(t) = \sqrt{t} \) in this paper. The learning rate \( \alpha_i(t) \) is decreasing according to the degree of excitation for individual neurons, which can be described as
\[
\alpha_i(t) = \frac{\alpha_0}{\alpha_i(0) + \sum_{j=1}^{t} h_i(j)},
\]

(19)

where \( \alpha_i(0) \) controls the starting value of the learning rate for separated neurons and \( \alpha_0 \) will adjust the rate of decrease. Clearly, the greater the excitation they have accumulated, the faster the decrease in the learning rate. Parameter \( \beta \) is used because in supervised learning we have information about the training sets. That is, we know in advance what the training vector is so that we can eliminate negative corrections. We take \( \beta = 0.05 \) here.

4. Experimental results and comparisons

4.1. Network configuration

The neural network structure in this paper adopted \( N \) neurons in the input layer with \( N = 20, 25, \ldots, 60 \). It is mentioned that Pham and Oztemel (1994) used \( N = 60 \) in their numerical results with the various LVQ networks. However, \( N = 60 \) may be too large for practical use, so we considered \( N \) from 20 to 60 here. Each neuron has an input point with the most recent mean value in the process variable to be controlled. These \( N \) input points are used to represent a control chart pattern. There are 36 neurons in the hidden layer to give six reference vectors per category to produce random variation. There are 6 neurons in the output layer for the six main types of patterns in the control charts. The initial values for all of the reference vectors are set randomly from \(-1\) to \(1\). The learning rates for the algorithms LVQ, LVQ-X and FS-LVQ are typically chosen to decrease from 0.05 to 0.01.

4.2. Training sets

Reviewing the patterns in figure 1, we have six models and training sets for the six main types of patterns as follows.

(a) Normal pattern:

\[ x(t) = n(t), \quad n(t) \sim N(0, 1), \]

where \( x(t) \) is the sample value at time \( t \) and \( t = 1, 2, \ldots, N \) are chosen as the sample points.

(b) Upward and downward shift pattern:

\[ x(t) = n(t) \pm u \times s \]

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Upward shift</td>
<td>1</td>
</tr>
<tr>
<td>Downward shift</td>
<td>0</td>
</tr>
<tr>
<td>Upward trend</td>
<td>0</td>
</tr>
<tr>
<td>Downward trend</td>
<td>0</td>
</tr>
<tr>
<td>Cycle</td>
<td>0</td>
</tr>
<tr>
<td>Normal</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 1. Six patterns and their output categories.
\[ u = \begin{cases} 0, & \text{before shifting,} \\ 1, & \text{after shifting,} \end{cases} \]

where \( s \) is the shift quantity that we take randomly from 0.5 to 3 for the upward shift and from \(-0.5\) to \(-3\) for the downward shift.

(c) Increasing and decreasing trend pattern:
\[ x(t) = n(t) \pm d \times t, \]

where \( d \) is the trend slope randomly selected from 0.01 to 0.06 for the upward trend and from \(-0.01\) to \(-0.06\) for the downward trend.

(d) Cyclic pattern:
\[ x(t) = n(t) + a \times \sin(2\pi t/\Omega), \]

where \( a \) is the amplitude randomly selected from 0.5 to 3 and \( \Omega \) is the cycle length. We take \( \Omega = 8 \) here.

4.3. Results and comparisons

As mentioned above, all of the 36 reference vectors and components were randomly chosen from \(-1\) to 1. The training sets were generated using the formula in section 4.2, using the same initial values and training sets but different learning rules for LVQ, LVQ-X and FS-LVQ. For testing the performance we sequentially input a pattern into the network and used equation (1) to determine which neuron wins the current competition and then we activate the winner to 1 and the others to 0. The output vectors indicate the identified pattern (see table 1).

In order to get a better prototype, training sets are an important factor in addition to the learning rules. This is a fundamental concept about supervised learning. We used 450 samples for each category. In some cases, if there were not enough training sets or it was difficult to collect the training sets, we may use fewer samples and reuse them several times. After finishing the training stage, we examined the performance using 150 samples for each category. The accuracy was calculated as the following equation:

\[
\text{Accuracy} \% = \frac{\text{Number of correctly classified patterns}}{\text{Total numbers of testing patterns}}.
\]

We implemented three algorithms LVQ, LVQ-X and FS-LVQ for the same training sets and testing samples with different sizes \( N \) from 20 to 60. In ten changing training sets and testing samples, we implemented three algorithms for these ten sample sets using different initial values with different sizes \( N \) and then gave the individual accuracy, average and variance as shown in table 2. Figure 3 gives the average accuracy of LVQ, LVQ-X and FS-LVQ with \( N = 20, 25, \ldots, 60 \), and figure 4 gives the variances. We found that the accuracy functions are all increasing in size \( N \). The accuracy of FS-LVQ is best. The variance of FS-LVQ is the smallest and also most stable for the difference size \( N \) in all three algorithms.

Overall, FS-LVQ has better accuracy than LVQ and LVQ-X. It is worth mentioning that LVQ can easily suffer from the problem where some neurons tend to win too often while others are inactive. Although LVQ-X can improve it, it still has the same problem as LVQ. In order to have nearly all neurons completely learned, LVQ and LVQ-X need to have more learning epochs. In our simulation, LVQ needs to have at least ten learning epochs and LVQ-X needs to have at least five learning
<table>
<thead>
<tr>
<th></th>
<th>$N = 20$</th>
<th></th>
<th>$N = 30$</th>
<th></th>
<th>$N = 40$</th>
<th></th>
<th>$N = 50$</th>
<th></th>
<th>$N = 60$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LVQ</td>
<td>LVQ-X</td>
<td>FS-LVQ</td>
<td>LVQ</td>
<td>LVQ-X</td>
<td>FS-LVQ</td>
<td>LVQ</td>
<td>LVQ-X</td>
<td>FS-LVQ</td>
</tr>
<tr>
<td>1</td>
<td>72.44</td>
<td>72.78</td>
<td>83.11</td>
<td>78.56</td>
<td>82.89</td>
<td>87.33</td>
<td>76.00</td>
<td>82.78</td>
<td>87.56</td>
</tr>
<tr>
<td>2</td>
<td>73.78</td>
<td>75.22</td>
<td>83.33</td>
<td>77.56</td>
<td>81.00</td>
<td>86.33</td>
<td>78.89</td>
<td>82.89</td>
<td>88.00</td>
</tr>
<tr>
<td>3</td>
<td>75.22</td>
<td>78.89</td>
<td>86.22</td>
<td>74.78</td>
<td>80.89</td>
<td>85.11</td>
<td>73.33</td>
<td>82.56</td>
<td>87.56</td>
</tr>
<tr>
<td>4</td>
<td>72.00</td>
<td>74.67</td>
<td>84.67</td>
<td>74.67</td>
<td>80.89</td>
<td>85.00</td>
<td>73.11</td>
<td>79.11</td>
<td>84.67</td>
</tr>
<tr>
<td>5</td>
<td>72.00</td>
<td>78.89</td>
<td>85.78</td>
<td>75.22</td>
<td>79.67</td>
<td>86.67</td>
<td>78.11</td>
<td>81.33</td>
<td>88.67</td>
</tr>
<tr>
<td>6</td>
<td>75.56</td>
<td>78.33</td>
<td>85.78</td>
<td>74.67</td>
<td>80.56</td>
<td>86.33</td>
<td>76.67</td>
<td>78.56</td>
<td>86.67</td>
</tr>
<tr>
<td>7</td>
<td>70.78</td>
<td>75.78</td>
<td>84.89</td>
<td>76.33</td>
<td>80.44</td>
<td>87.22</td>
<td>79.00</td>
<td>82.00</td>
<td>88.56</td>
</tr>
<tr>
<td>8</td>
<td>72.56</td>
<td>74.84</td>
<td>85.11</td>
<td>74.56</td>
<td>80.78</td>
<td>87.22</td>
<td>72.78</td>
<td>79.78</td>
<td>85.67</td>
</tr>
<tr>
<td>9</td>
<td>72.00</td>
<td>76.78</td>
<td>84.33</td>
<td>78.89</td>
<td>81.89</td>
<td>86.33</td>
<td>73.56</td>
<td>80.33</td>
<td>87.00</td>
</tr>
<tr>
<td>10</td>
<td>74.00</td>
<td>78.89</td>
<td>85.89</td>
<td>76.22</td>
<td>81.89</td>
<td>87.22</td>
<td>71.89</td>
<td>77.44</td>
<td>85.11</td>
</tr>
<tr>
<td>average</td>
<td>73.03</td>
<td>76.01</td>
<td>84.91</td>
<td>76.15</td>
<td>81.09</td>
<td>86.48</td>
<td>75.33</td>
<td>80.68</td>
<td>86.95</td>
</tr>
<tr>
<td>variance</td>
<td>2.787</td>
<td>4.281</td>
<td>1.292</td>
<td>2.352</td>
<td>0.832</td>
<td>0.845</td>
<td>6.872</td>
<td>3.024</td>
<td>1.989</td>
</tr>
</tbody>
</table>

Table 2. Comparisons of different recognizers with $N = 20, 30, 40, 50, 60$. 

Fuzzy-soft learning vector quantization.
epochs to ensure nearly all neurons are completely learned. However, the FS-LVQ needs only one learning epoch for all neurons to be completely learned because it uses a fuzzy relaxation technique and updates all neurons simultaneously. Thus, FS-LVQ can decrease the learning time with less learning epochs and increase the accuracy better than LVQ and LVQ-X.

5. Conclusion
The proposed new approach modifies all neurons according to the information between training sets and reference vectors. This makes the learning procedure more efficient. In fact, a larger pattern size will increase the accuracy because it has more information, but this is not practical in some cases. FS-LVQ produced better performance than the traditional learning vector quantization such as LVQ, LVQ2 and LVQ-X etc. Thus, the supervised competitive learning network with FS-LVQ is highly recommended for use as a pattern recognizer for control charts.

We mention that all LVQ techniques are necessary to assign a fixed size $N$ in order to contain complete unnatural patterns. For incomplete unnatural patterns that commenced after the initial analysis window, Guh and Tannock (1999) used a
Back-Propagation Network (BPN) for the recognition of these control chart concurrent patterns. This BPN has a good network structure to detect incomplete unnatural patterns. However, this BPN approach to control chart pattern recognition is slow and complicated. Recently, Yang and Yang (2002) have designed a simple mechanism for incomplete control chart pattern recognition based on the statistical correlation coefficient. In our further research, we will consider these pattern recognition methods for the recognition of multivariate control chart patterns.

Acknowledgements

The authors are grateful to the anonymous referees for their valuable comments. This work was supported in part by the National Science Council of Taiwan under grant NSC-89-2118-M-033-004.

References


