CHAPTER 5

Combined Unsupervised and Supervised Learning Model

5.1 Introduction

If the unsupervised Kohonen's algorithm is to be used as a pattern classifier in which the cells or their responses are grouped into subsets, each of which corresponds to a discrete class of patterns, then the problem becomes a decision process and must be handled differently. The original map, like any classical vector quantization method is mainly intended to approximate input signal values, or their probability density function, by quantized code book vectors that are localized in the input space to minimize a quantization error functional. On the other hand, if the signal sets are to be classified into a finite number of categories, then several code book vectors are usually made to represent each class, and their identity within the classes is no longer important. In fact, only decisions made at class borders count. It is then possible to define effective values for the code book vectors such that they directly define near-optimal decision borders between the classes, even in the sense of classical Bayesian decision theory. These strategies and learning algorithms were introduced by T. Kohonen [76, 77, 79] and called as Learning Vector Quantization.

The work carried out in this chapter has resulted in the following papers:


Learning Vector Quantization [77] is a supervised learning technique that uses class information to move the code vectors slightly, so as to improve the quality of the classifier decision regions. An input vector $X$ is picked at random from the input space. If the class labels of the input vector $X$ and the winner node $w_c$ corresponding to a particular class agree, the vector $w_c$ is moved in the direction of the input vector $x$. If, on the other hand, the class labels of the input vector $X$ and the vector $w_c$ disagree, then $w_c$ is moved away from the input vector $x$.

5.2 Developed System

The combined unsupervised and supervised model mainly consists of two phases; the training phase, and recognition phase. Fig. 5.1 shows the block diagram of adaptive pattern classification, using a combination of unsupervised and supervised learning concepts. The training phase further consists of four stages. The feature extractor accepts the input space and transforms it to a one-dimensional vector of feature values (chapter 2). The output of the feature extractor is further processed using supervised neural network based feature selector (chapter 3). The devised feature selector model removes redundant features and returns the resultant features. The selected feature vector is fed as input to the Kohonen's neural network model. The developed combined classifier model comprises of unsupervised and supervised learning/training concepts. The first stage of the classifier assigns a given pattern (presented in the form of a feature vector) to one of a finite number of classes by applying the nearest neighbor rule. Both the Kohonen’s algorithm based (KA) classifier, and the modified Kohonen’s algorithm based (MKA) classifier models have been used (chapter 4). Kohonen has suggested that if the nodes of the feature map are used for pattern recognition, then their classification accuracy can be multiplied if the nodes are fine-tuned using supervised learning principle. The learning vector quantization uses supervised learning to modify the internal state of the network provided by the first stage and to remodel the features found in the training data. A fine-tuned map is autonomously organized by a cyclic process of comparing the input patterns to the centroid vectors at each node. Fine-tuning is achieved by selecting the training
vectors with known classification, and presenting them to the network to examine the cases of misclassification.

5.2.1 Type one Learning Vector Quantization (LVQ1)

If the class labels of the input vector $x(t)$ and winner node $w_c(t)$ match, then the weight of the winner node is updated as:

$$w_c(t+1) = w_c(t) + \alpha(t) \ (x(t) - w_c(t))$$

However, if the class labels do not match then the winner node is moved away from the input sample. Thus,

$$w_c(t+1) = w_c(t) - \alpha(t) \ (x(t) - w_c(t))$$

Where $\alpha(t)$ is the learning rate which decreases monotonically with the number of iterations. Since this is a fine tuning process $\alpha(t)$ must take an initial value closer to zero (say 0.01) and then it must decrease gradually with the number of iterations. For all other winner vectors, the weight values remain unaltered. That is, $w_c(t+1) = w_c(t)$.

The LVQ1 algorithm suffers from several short comings, and variants were developed in response to those drawbacks [67]. The LVQ2 algorithm, an improved version of LVQ1, adjusts the boundaries between categories to keep misclassifications to a minimum by training the interconnection weights in the portion of the structure that is characterized by a complete directed bipartite graph. To do so, LVQ2 uses competitive learning with the winner take all strategy to find the output node closest to the training pattern. If the training pattern's class differs from the output node's class, then LVQ2 finds the next best match. If a node other than the winning node is associated with the correct class, the algorithm moves that next best match closer to the training pattern and the originally more successful incorrect node(s) away from the input pattern. The process continues until optimum result is approximated.
5.2.2 Type two Learning Vector Quantization (LVQ2)

Algorithm:

1. Present the weight vectors of the centroids and the corresponding labels collected at the end of training of Kohonen's classifier.
2. Present each input vector $x(t)$ and its corresponding class label.
3. Compute the Euclidean distance between the input vector $x(t)$ and each of the ten centroids $w_c(t)$.
4. Determine the winner node $w_c(t)$ corresponding to the minimum distance.
5. Compare the class labels of the input vector and the winner.
6. If the class labels match, then the weight vector of the winner node is updated as
   $$ w_c^{*}(t+1) = w_c^{*}(t) + \alpha(t) (x(t) - w_c^{*}(t)) $$
7. If the class labels do not match, then the next nearest centroid node $w_{nc*}(t)$ is located and its class label is compared with that of the input. If the class labels match then the weight vector is updated as follows:
   $$ w_{nc*}(t+1) = w_{nc*}(t) + \alpha(t) (x(t) - w_{nc*}(t)) $$
   At the same time the winner node $w_c^{*}$ is moved away from the input vector by updating its weight value as follows;
   $$ w_c^{*}(t+1) = w_c^{*}(t) - \alpha(t) (x(t) - w_c^{*}(t)) $$
8. For all other centroids $w_c(t)$ the weight updation is as follows:
   $$ w_c(t+1) = w_c(t) $$
   That is, all other centroids are undisturbed.
   $\alpha(t)$ is the learning rate parameter whose initial value is chosen as 0.01. As the training proceeds its value gradually decreases.
9. A count of the number of samples correctly classified is obtained, and is used to compute the classification accuracy. The above process is repeated until the pre-specified classification accuracy is obtained.
5.2.3 Recognition Algorithm

The performance of the trained classifier is then evaluated by applying the unseen input samples. An input sample is mapped onto the combined classifier which returns the confidence values of that input sample belonging to each of the classes. The input sample is assigned to the class which has the highest confidence value.

Computation of Confidence Values:

The combined KA and LVQ2 classifier can be used to determine the class to which an input pattern belongs by applying the nearest neighbour rule. But instead of identifying only the class to which it belongs it would more useful to obtain a value that represents to what extent the input pattern matches each of the classes. This value is called the confidence value of the input pattern corresponding to that class.

After the training phase is completed a standard set of samples (machine printed numerals) is used to determine the centroid of each class. One machine printed sample belonging to each class is presented to the combined classifier and by applying Euclidean distance formula the winner node is determined. This winner node is termed as the centroid of that particular class. These centroids are used to calculate the confidence values for an input sample during the recognition/testing phase.

During the testing phase an input sample is mapped onto the classifier and the winner node is identified. The confidence values of the input sample are calculated by determining the Euclidean distance from the winner node to the centroids of the classes. This Euclidean distance from a winner node to a centroid is determined as follows:

\[ d[i] = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \]

Where,

- \( x_1 \) = centroid [i] DIV net-size.
- \( x_2 \) = centroid [i] MOD net-size.
- \( y_1 \) = winner DIV net-size
- \( y_2 \) = winner MOD net-size
For each input sample an array of 10 values is returned. Each value describes the extent to which an input sample belongs to a particular class. The confidence values are computed as follows:

\[
\text{Convall}_i = (100 - d[i]) \times 100 / (\text{dm})
\]

Where, \( \text{dm} = \text{maximum diagonal distance} = \sqrt{2} \times (\text{net-size-1}) \)

**Decision Rule:**

The performance of the system is evaluated based on the following criteria:

1. **Recognition Rate:** Percentage of samples recognized correctly.
2. **Substitution Rate:** Percentage of samples that were recognized incorrectly.
3. **Rejection Rate:** Percentage of samples that could not be assigned to any particular class.
4. **Classification accuracy:** Percentage of training samples that were recognized correctly.

**Algorithm:**

1. For all the input samples collect the confidence values from the combined classifier.
2. For all the samples compute the following:

   Let \( R(x) \) be a recognized character for sample \( x \) and \( j \) be the character with the highest confidence value, then if the maximum confidence value is greater than the threshold then increment \( \text{rec} \) (which is the number of samples recognized) else find the possible substitution by checking the second, third, and fourth confidence values. If these are greater than \( \lambda \) increment \( \text{sub} \) (number of samples substituted) otherwise increment \( \text{rej} \) (number of samples rejected).
\[ R(x) = j \text{ if conval}[j] = \max \text{ conval}[i] \text{ and conval}[j] \geq \lambda , \text{ for all } 0 \leq i \leq 9 \]

\[
\begin{align*}
\text{then} & \quad \text{rec} ++ \\
\text{else} & \\
\quad \text{if conval}[i] \geq \lambda \\
\quad \text{then} & \quad \text{sub} +++ \\
\quad \text{else} & \quad \text{rej} ++
\end{align*}
\]

Where conval[ ] is a set of ten confidence values for sample x.

Recognition (REC) = rec DIV total number of samples.

Substitution (SUB) = sub DIV total number of samples.

Rejection (REJ) = rej DIV total number of samples.

Reliability (REJ) = REC DIV (REC+SUB).

5.3 Experiments and Results

Fig. 5.1 shows the MKA and LVQ2 combined classifier model. Eight experiments have been conducted to investigate the classification accuracy of different techniques. The database of ten thousand samples has been used. Out of these, 5,000 samples are used for training and the remaining 5,000 samples are used for testing. The output layer of the Kohonen neural network has 20x20 nodes and the input layer has 10 nodes. The supervised LVQ2 technique is applied after collecting the centroids from the Kohonen classifier. The results shown in Table-5.1 justify the fact that the classification accuracy is improved when the feature map is fine-tuned by applying LVQ2 technique.

The following can be inferred from the results:

1. The feature selector based KA classification method resulted in a classification accuracy of 53.80% and the training time taken was 624 sec. When combined with LVQ2 technique the classification accuracy was raised to 100.00 % in 1200 sec. This enhanced classification accuracy increased the recognition rate to 84.00% from 78.00%.
2. The MKA classifier achieved a classification accuracy of 95.80% in 310 secs and it increased to 100.00% in 910 secs when combined with LVQ2 technique.

3. The combined MKA50 and LVQ2 technique achieved a classification accuracy of 100.00% in 912 secs as against 97.20% in 160 secs without LVQ2 technique. The combined classifier yielded a recognition rate is 99.60% and a rejection rate of 0.40%.

4. The MKA75 classifier yielded a classification accuracy of 96.30% in 205 sec. However the combined MKA75 and LVQ2 technique resulted in a classification accuracy of 100% and the training time obtained was 4232 secs.

5.4 Conclusion

Unconstrained hand written character recognition using KA, MKA, KANA and LVQ2 and MKA and LVQ2 techniques is presented in this chapter. The LVQ2 method applied after obtaining the classification map from the MKA technique resulted in best performance. In any pattern recognition system, the most important problem to address is feature extraction cum selection and correct classification. These two factors are well taken care of in this study. Instead of developing a single high performance classifier which is usually extremely difficult to design, we have built a two-stage classifier. The first is unsupervised Kohonen’s classifier followed by the supervised LVQ2 classification. Each classifier acting alone may not produce the desired performance, but the appropriate combination of classifiers produces a highly reliable performance. The results demonstrate that the combination of MKA and LVQ2 techniques results in a best classification accuracy at the same time minimizing the rejection rate.
Table- 5.1

Results of eight techniques for 5000 unconstrained handwritten numerals using
Pentium Processor

<table>
<thead>
<tr>
<th>Technique</th>
<th>REC (%)</th>
<th>SUB (%)</th>
<th>REJ (%)</th>
<th>REL (%)</th>
<th>Classification accuracy (%)</th>
<th>Training time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>KA</td>
<td>78.00</td>
<td>0.00</td>
<td>22.00</td>
<td>100.00</td>
<td>53.80</td>
<td>624</td>
</tr>
<tr>
<td>MKA</td>
<td>95.70</td>
<td>0.00</td>
<td>04.30</td>
<td>100.00</td>
<td>95.80</td>
<td>310</td>
</tr>
<tr>
<td>MKA 50</td>
<td>96.50</td>
<td>0.00</td>
<td>03.50</td>
<td>100.00</td>
<td>97.20</td>
<td>160</td>
</tr>
<tr>
<td>MKA 75</td>
<td>92.00</td>
<td>0.00</td>
<td>08.00</td>
<td>100.00</td>
<td>96.30</td>
<td>205</td>
</tr>
<tr>
<td>KA+LVQ2</td>
<td>84.00</td>
<td>0.00</td>
<td>16.00</td>
<td>100.00</td>
<td>100.00</td>
<td>120</td>
</tr>
<tr>
<td>MKA+LVQ2</td>
<td>96.50</td>
<td>0.00</td>
<td>03.50</td>
<td>100.00</td>
<td>100.00</td>
<td>610</td>
</tr>
<tr>
<td>MKA50+LVQ2</td>
<td>99.60</td>
<td>0.00</td>
<td>00.40</td>
<td>100.00</td>
<td>100.00</td>
<td>912</td>
</tr>
<tr>
<td>MKA75+LVQ2</td>
<td>97.60</td>
<td>0.00</td>
<td>02.40</td>
<td>100.00</td>
<td>100.00</td>
<td>4232</td>
</tr>
</tbody>
</table>
Learning vector quantization (LVQ)

Kohonen neural network structure

Feature selector

Input Layer

Modified feature extractor

Fig 5.1 Combined classifier