Pattern Recognition, Vol. 26, No. 4, pp. 543-551, 1993

A NEW SHAPE REPRESENTATION SCHEME AND ITS APPLICATION TO SHAPE DISCRIMINATION USING A NEURAL NETWORK

NIKHIL R. PAL,†§ PRATIK PAL‡ and ANUPAM K. BASU‡
† Electronics and Communication Sciences Unit, Indian Statistical Institute, 203 B.T. Road, Calcutta 700035, India
‡ Tata Consultancy Services, 2 Hare Street, Calcutta 700001, India

Abstract—A new method of shape representation and feature extraction is suggested. A shape is approximated by a constant-point polygon in which between any two adjacent vertices of the polygon, the number of points on the contour of the shape is constant. This representation is applicable for both concave and convex shapes and there is no chance of missing any spikes on the boundary. The sequence of the angle of variation between two consecutive line segments is taken as the primary feature (representation of the shape). This sequence is then modelled by an autoregressive (AR) process and the least square error estimate of the AR coefficient vector is used as input to a multilayer perceptron (MLP) network for learning and classification. Robustness of the shape representation scheme and the MLP classifier is also investigated empirically. Adaptive AR modelling is used for estimating the numerically stable and robust coefficient vector.

Shape recognition Neural networks Multilayer perceptron Autoregressive model

1. INTRODUCTION

Recognition of shapes is an important problem in computer vision. Its applications vary from detection of aircraft to identification of cancerous blood cells. Several algorithms[1-14] are available in the literature, some of which work only for convex shapes while others fail for open shapes. This report reviews some of the existing methods of feature extraction and classification of shapes. Attempts have then been made to develop a new feature extraction and classification techniques for two-dimensional (2D) closed shapes without any restriction on the convexity/concavity of the shape. A new primary feature, called angle of variation has been suggested based on a constant-point polygonal approximation of the shape. The constant-point polygonal approximation is an approximation in which between two consecutive vertices of the polygon, a constant number of points (pixels) exists on the contour. The sequence of angle of variation between two consecutive segments is taken as the primary feature for the shape. The angle sequence is then modelled as a circular autoregressive (AR) process. The AR coefficient vector is then estimated using the least square error (LSE) estimate. These AR vectors can be used to classify shapes using the minimum distance classifier. When the class boundaries are complicated the minimum distance classifier may not function well. Here this feature has been used as input to a multilayer perceptron network for learning and classification. An investigation has been carried out varying the number of layers, number of neurons in the intermediate layers and the dimensionality of feature vectors. The method has been tested on four classes of shapes and the results are found to be quite satisfactory. To make the AR coefficients numerically more accurate and robust an adaptive formulation of the AR process has been used.

2. SOME WORKS IN SHAPE RECOGNITION AND THEIR LIMITATIONS

2.1. *Method of Dubois and Glanz*[1]

In this approach, the boundary of a shape is approximated by an ordered sequence of N angularly equispaced radius vectors projected between the centroid and the boundary as shown in Fig. 1. The boundary approximation can be improved by increasing the number of radius vector projections, N. Here the radius vector length is a function of the angle of projection $\phi = 2\pi t/N$, where $t = 1, \ldots, N$ and $r(\phi)$ forms a one-dimensional boundary approximation. This boundary approximation is however restricted to boundaries which are convex. Since their shape description technique is based on an AR model, the function $r(\phi)$ should be single valued. However, $r(\phi)$ becomes multivalued when the radius vector intersects the boundary more than once, as in the case of a non-convex boundary. In order to represent such non-convex boundaries, Dubois and Glanz suggested an algorithm that "unwraps" the boundary and stretches the multivalued function $r(\phi)$ to produce a one-dimensional function $ru(i)$. As Fig. 2 shows, the new scheme searches sequentially along the boundary until

§ Present address: University of West Florida, Division of Computer Science, 11000 University Parkway, Pensacola, FL 32514-5750, U.S.A.

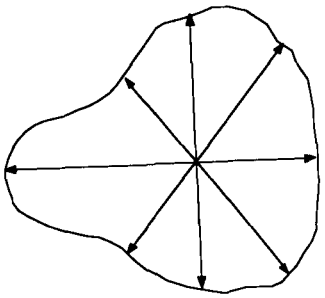

Fig. 1. Shape representation using radius vectors.

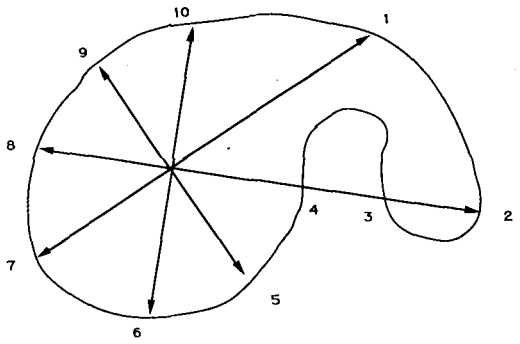

Fig. 2. Extraction of radius vectors for non-convex shapes.

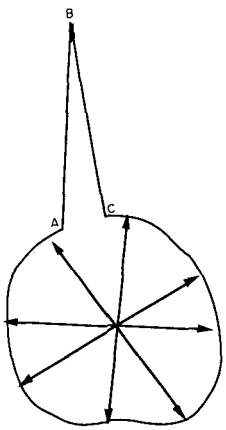

Fig. 3. Missing of spike by a boundary sampler.

a radius vector crossing is detected and measures the length. To determine the next ordered length, the sequential boundary search is continued until another radius vector crossing is detected. Thus the radius lengths are ordered according to the order of detection i, by the sequential boundary follower. This new scheme also produces a periodic or circular time series but the period Nu of the new time series is longer than the period N of $r(\phi)$. This sequence, $ru(i)$, is then modelled as a circular AR process of some order p and then the AR coefficient vector is used for the classification purpose.

Regarding this approach the following points are noted.

(1) This method of unwrapping does not preserve the sign of the angular change between two consecutive boundary samples. This loss of phase information can lead to a single unwrapped function coming from a number of different shapes.

(2) The centroid of a non-convex shape may fall on the boundary of the shape.

(3) A spike on the boundary of the shape can be missed out by this method of boundary sampling (Fig. 3).

(4) This method is not applicable for open shapes.

2.2. *Method of Das et al.*[2]

Das *et al.* approximated a shape by N samples of its original boundary sequence. The x and y Cartesian coordinates of these samples, with reference to the object centroid, serve as elements of a bivariate series. This bivariate series is then represented by a circular autoregressive (CAR) model characterized by a set of unknown coefficient matrices and an independent vector noise sequence. They used the following model:

$$\mathbf{Y}(k) = \sum_{j=1}^{p} A_j Y(k-j) + \sqrt{\beta}\, V(k), \quad k = 1, 2, \ldots, N$$

$$X(k) = \alpha + Y(k)$$

where $\{A_j, j = 1, 2, \ldots, p\}$ are 2×2 coefficient matrices and α is the 2×1 process mean vector; i.e. $E(X_k) = \alpha$. The modelling error is taken as zero mean white noise and is represented by $\sqrt{\beta}\, V(k)$ with β as the covariance matrix.

Several classification features which are functions of the coefficient matrices and the residual error covariance matrices are used for shape recognition. However, the following comments are relevant.

(1) This method does not work with open shapes.

(2) Moreover, it involves estimation of many parameters, for example, even when the order of the CAR process is three, the number of coefficients to be determined is 12.

(3) The coefficient matrices A_j are not rotation invariant. Hence these cannot be used *directly* for classification.

2.3. *Method of Gupta et al.*[5]

Gupta *et al.* used a multilayer perceptron network for shape classification. The normalized contour sequence has been used as the input, where a contour sequence is an ordered sequence that represents the Euclidean distance between the centroid and all the boundary pixels. They referred to the network as a three-layer perceptron, but effectively it is a four-layer perceptron. This is because every component of the input vector (contour sequence) is applied to every neuron in the first hidden layer. This is possibly required to make the input rotation invariant. In this

experiment they used 48 neurons in the first hidden layer, so the number of neurons in the input layer has also to be 48. The number of neurons used in the second layer was 26 and that in the output layer was 4. Thus the total number of neurons was 126 and the network required updating of $48*48 + 48*26 + 26*4 = 3656$ weights in every iteration. Moreover, if the number of neurons in the input layer is reduced, the shape information will be lost as reduction of the number of neurons in the input layer implies reduction of duration of the contour sequence.

3. PROPOSED METHOD

In this report a new scheme for shape representation has been suggested and feature vectors derived from this have then been used as input for classification. Usually the feature vectors derived from a training set of shapes are used to compute the representative vectors for each class. Unknown shapes are then classified using these representatives. Here we have used a robust technique of classification using a neural network. First, we shall discuss the proposed representation scheme and then the classification technique.

3.1. *Representation and feature extraction*

3.1.1. *Angle of variation.* The entire boundary of a 2D shape is divided into constant point segments and then each segment is approximated by a line segment joining the two end points. As the boundary is traversed clockwise (or anticlockwise), the change of angle between consecutive segments serves as our primary feature. Suppose there are N points on the boundary and we choose to segment the entire boundary into n equal segments. Our concept of constant point segments means that there will be (N/n) points between the initial and final points of each segment, though the segment lengths (i.e. the Euclidean distance between the initial and final points) may be unequal. This will do some sort of polygonal approximation of the shape. The reason for not considering equal length segments is given as follows. Consider the shape as shown in Fig. 3. If we choose each segment to be of p units of length then while segmenting the contour, the probability of missing out the spike ABC is quite high which in turn leads to a loss of shape information. However, if we decide to segment the boundary into n segments such that there will be p pixels between the initial and final points of each segment then there will be no loss of shape information unless p is very high.

The process of finding out the angle of variation is described below. Consider three segments (AB, BC, CD) of equal length. Suppose the change of angle between segments AB, BC is θ_1, and that between BC, CD is θ_2, serve as our primary feature. Then these angles are found as follows. Assume that $\overline{AB}$ and $\overline{BC}$ are vectors. The dot product of the vectors is given as

$$\overline{AB}\cdot\overline{BC} = |AB||BC|\cos\theta_1.$$

So

$$\cos\theta_1 = \overline{AB}\cdot\overline{BC}/|AB||BC|. \tag{1}$$

If the coordinates of A, B, C are $(x1, y1)$ $(x2, y2)$ and $(x3, y3)$, respectively, then the position vector of $\overline{AB}$ is $(x2-x1)\hat{i} + (y2-y1)\hat{j}$ and the position vector of $\overline{BC}$ is $(x3-x2)\hat{i} + (y3-y2)\hat{j}$, where $\hat{i}$ and $\hat{j}$ are the unit vectors in the x and y directions, respectively. So

$$\overline{AB}\cdot\overline{BC} = (x3-x2)(x2-x1) + (y3-y2)(y2-y1)$$

and

$$\cos\theta_1 = ((x3-x2)(x2-x1) + (y3-y2)(y2-y1))/\sqrt{((x3-x2)^2 + (y3-y2)^2)}\sqrt{((x2-x1)^2 + (y2-y1)^2)}. \tag{2}$$

Proposition. Angle of variation is rotation, translation and size invariant.

Proof. Consider the two shapes shown in Fig. 4 where the second shape is a rotated version of the first one. Suppose the polar coordinates of A, B, C are $(r1, \alpha1)$, $(r2, \alpha2)$ and $(r3, \alpha3)$, respectively. Then the slope of AB is

$$m_1 = \frac{(r2\sin\alpha2 - r1\sin\alpha1)}{(r2\cos\alpha2 - r1\cos\alpha1)}$$

and the slope of BC is

$$m_2 = \frac{(r3\sin\alpha3 - r2\sin\alpha2)}{(r3\cos\alpha3 - r2\cos\alpha2)}. \tag{3}$$

Thus

$$\tan\theta_1 = \frac{m1 - m2}{1 + m1*m2} = \frac{r2r3\sin(\alpha2-\alpha3) + r1r3\sin(\alpha3-\alpha1) + r1r2\sin(\alpha1-\alpha2)}{r2r3\cos(\alpha2-\alpha3) - r2^2 - r1r3\cos(\alpha1-\alpha3) + r1r2\cos(\alpha1-\alpha2)}. \tag{4}$$

Now if the shape is rotated by an angle θ with respect to an axis passing through the centroid, then the rotated coordinates of A, B, C are $(r1, \alpha1 - \theta)$, $(r2, \alpha2 - \theta)$ and $(r3, \alpha3 - \theta)$. If the angle between AB and BC is $\theta2$, then it can be shown in a similar manner that $\tan\theta2$ is given by the same formula as $\tan\theta1$, i.e. $\tan\theta2 = \tan\theta1$. Hence it is rotation invariant. Moreover, a bigger shape will have all its coordinates scaled by a factor (say α) over all the coordinates of the smaller shape. Hence the slopes $m1$ and $m2$ remain the same which results in the same angle sequence for both

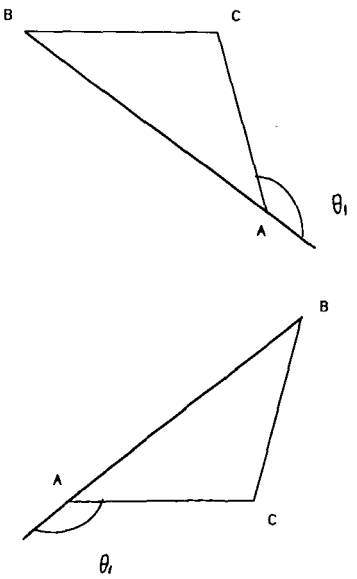

Fig. 4. Illustration of rotation invariance for angle of variation.

the larger and the smaller shapes. Thus it is also size invariant.

The translation invariance is obvious.

3.1.2. *Finding out the feature vectors.* The way in which the angle sequence has been generated, it is clear that the ith angle is dependent on some immediately previous angles. Hence such a sequence can be modelled as an AR process.

3.1.3. *AR model.* An AR model is a parametric equation that expresses each sample of an ordered set of data samples as a linear combination of a specified number of previous samples plus an error term. If the sequence is assumed to be a circular one, then it is invariant to rotation, translation. Hence the model parameters (CAR coefficients) are also so and can be used as shape descriptors. The specific form of the model used is

$$y_t = \alpha_0 + \sum_{j=1}^{m} \alpha_j y_{t-j} + w_t \tag{5}$$

where y_t is the current angle; y_{t-j} the angle detected j angles before the current angle; $\alpha_1, \alpha_2, \ldots, \alpha_m$ the unknown AR coefficients to be estimated from the observed time series data; m the model order; α_0 the unknown constant to be estimated, and $\{w_t\}$ a random sequence of independent, zero mean noise with variance β.

The AR model parameters are estimated using the conventional least square error method. The least square error estimates are those estimates which minimize the error of prediction. This can be found as follows. Let

$$y_i = \alpha_0 + \sum_{k=1}^{m} \alpha_k y_{i-k}, \quad i = m+1, m+2, \ldots . \tag{6}$$

In matrix form the system of equations can be written as

$$\begin{bmatrix} y_{m+1} \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & y_m & \cdots & y_1 \\ 1 & y_{m+1} & \cdots & y_2 \\ \vdots & \vdots & \vdots & \\ 1 & y_{n-1} & \cdots & y_{n-m} \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_m \end{bmatrix} \tag{7}$$

i.e. $Y_n = X_n \beta_n$ where

$$\beta_n = \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_m \end{bmatrix}.$$

The least square error estimate of β_n is

$$\text{or } \hat{\beta}_n = (X_n^T X_n)^{-1}(X_n Y_n). \tag{8}$$

Solving the above equation, we can get the AR model parameters which may serve as the ultimate features for classification of shapes. It may be noted here that if there are n angles and p is the order of the model then $(n-p)$ angles can be predicted using equation (5). In order to predict the first p values one can assume p previous values as zero or one may view the sequence as a circular one. The latter choice is more logical as this will make the process starting point invariant for the polygon approximated shape.

However, in the case of angle of variation if the length of a segment is k (i.e. k points or pixels lie on the contour between the two end points of a line segment) then $(k-1)$ polygonal approximations of the shape are possible depending on the starting point (a similar situation exists in many other shape representation schemes). The angle sequence generated for each of these may be little different. But this difference will decrease with the reduction in k. Experimentally, it has been found that the AR vectors generated by the $k-1$ different angle sequences are slightly different. This difference can be possibly reduced in two ways. One can estimate the AR vector considering all the sequences or repeat an angle sequence a large number of times and the AR vectors are estimated using that. This will also make the coefficient vectors more robust to noise. But how many times should we repeat the sequence? One solution may be to compute the AR vector first with one sequence, and then with two sequences, three sequences and so on, until the coefficient vector stabilizes. This will involve a lot of computation, as every time one needs to invert a matrix. Accuracy of the estimate may also be reduced. To overcome this we suggest use of the adaptive AR model. In this model one can add observations one by one and the estimates of the AR coefficients can be updated without recomputing the matrix inverse repeatedly. The model is developed as follows.

3.1.4. *Adaptive AR model*: Let the model order be p, $y_i (i = 1, 2, \ldots, n)$ be the observed sequence and α_i $(i = 0, 1, 2, \ldots, m)$ the model coefficients. If we have only n observations then the LSE estimate of the coefficient vector is given by equation (8). Now if a new sample is

used then the transformed system of equations is

$$\begin{bmatrix} y_{m+1} \\ \vdots \\ y_{n+1} \end{bmatrix} = \begin{bmatrix} 1 & y_m & \cdots & y_1 \\ 1 & y_{m+1} & \cdots & y_2 \\ \vdots & & & \\ 1 & y_{n-1} & \cdots & y_{n-m} \\ 1 & y_n & \cdots & y_{n+1-m} \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \\ \vdots \\ \alpha_m \end{bmatrix} \quad (9)$$

i.e.

$$\hat{\beta}_{n+1} = (X_{n+1}^{\mathrm{T}} X_{n+1})^{-1} (X_{n+1}^{\mathrm{T}} Y_{n+1})$$

where

$$x_{n+1} = \begin{pmatrix} X_n \\ XX \end{pmatrix}$$

and

$$XX = (1, y_n, \ldots, y_{n+1-m}).$$

After some algebraic manipulation one gets $\hat{\beta}_{n+1}$ as follows:

$$\hat{\beta}_{n+1} = \hat{\beta}_n + (X_{n+1}^{\mathrm{T}} X_{n+1})^{-1} XX^{\mathrm{T}} [y_{n+1} - XX\hat{\beta}_n] \quad (10)$$

$$= \hat{\beta}_n + K_{n+1} [y_{n+1} - \hat{y}_{n+1}] \quad (11)$$

where

$$K_{n+1} = \frac{P_n XX^{\mathrm{T}}}{1 + XXP_n XX^{\mathrm{T}}}$$

and

$$P_n = (X_n^{\mathrm{T}} X_n)^{-1}.$$

An initial estimate can be obtained by taking a reasonable number of points and then each new point is added until convergence is achieved. This method has been found to be more accurate and numerically more stable than the previous model.

3.2. *Classification*

These AR vectors can be used to classify shapes. In fact a data base can be created with representative AR vectors (one for each shape) from a set of known shapes. Then this data base can be used to classify an unknown shape using minimum distance or some other criterion. When the class boundaries are simple this works fine but with complicated structure of classes the performance of such classifiers may fall down. To overcome this a neural network classifier can be used. The advantage with a neural network is that it can test competing hypotheses in parallel, thereby providing output in real time. Moreover, it has the ability to learn and separate complicated and concave/convex class boundaries. In the next section we concentrate on a neural network classifier.

4. A NEURAL NET APPROACH FOR SHAPE CLASSIFICATION

Human intelligence, discriminating power, etc. are attributed to the massively connected network of biological neurons in the human brain. Recently, several attempts have been made to emulate the biological neural network by artificial analogue neurons.

4.1. *The artificial neural net*

The artificial neuron has been designed to mimic the first-order characteristics of the biological neuron. In essence a set of inputs are applied, each representing the output of another neuron. Each input is multiplied by a corresponding weight, analogous to a synaptic strength and all of the weighted inputs are summed to determine the activation level of the neuron. Figure 5 represents the idea.

In the diagram $X = (x1, x2, \ldots, xn)$ is the set of inputs and $W^{\mathrm{T}} = (w1, w2, \ldots, wn)$ the set of weights. Hence NET $= \sum w_i x_i = XW$. The NET signal is usually further processed by an activation function, F, to produce the neuron's output signal OUT. This may be a simple function like

$$\mathrm{OUT} = K(\mathrm{NET})$$

where K is such that

$$\mathrm{OUT} = 1 \quad \text{if NET} > \text{Threshold}$$

$$= 0 \quad \text{otherwise.}$$

It may also be a sigmoidal function, for example

$$\mathrm{OUT} = 1/(1 + \mathrm{e}^{-\mathrm{NET}}).$$

This type of sigmoidal function enables the neuron

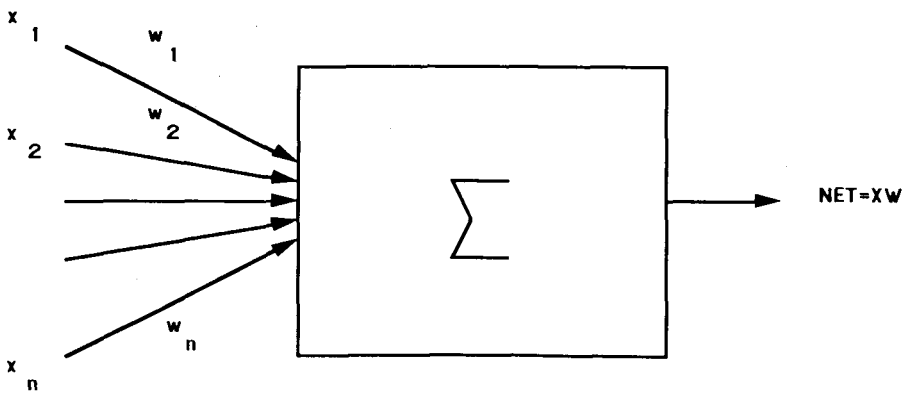

Fig. 5. Schematic diagram of functioning of a neuron.

to function with appropriate gain over a wide range of input levels. Neural networks have been extensively used for pattern classification.[5,11–14] The following section describes the perceptron network commonly used for classification.

4.1.1. *Single layer perceptron.* A single layer perceptron consists of only a single layer of neurons. The nodes compute a weighted sum of the input elements and apply an activation function on the weighted sum which determines the output of the nodes. This perceptron divides the space spanned by the input into two regions separated by a hyperplane or a plane in two dimensions. The connection weights and the thresholds in a perceptron can be fixed or adapted. However, the single layer perceptron cannot simulate a simple EXCLUSIVE-OR function. Thus no matter what weights and thresholds are assigned, the EX-OR function cannot be realized. Such functions are said to be linearly inseparable and set definite bounds on the capabilities of single layer networks. This problem can be circumvented by a multilayer perceptron network (MLP).

The linear inseparability problem of single layer networks could be overcome by adding more layers. These multilayer networks can perform more general classifications, separating points that are contained in concave/convex regions. The architecture of a three layer MLP is shown in Fig. 6. In an MLP there is no connection among the neurons within a layer, but every neuron in layer i is connected to all neurons in layer $i+1$.

4.1.2. *Perceptron learning.* The learning ability of an artificial neural network is its most intriguing property. Learning can either be supervised or unsupervised. Supervised learning requires an external "teacher" that evaluates the behaviour of the system and directs subsequent modifications. Perceptron learning is of the supervised type.

A perceptron is trained by presenting a set of patterns to its input layer, one at a time and adjusting the weights until the desired output occurs for each of them in the output layer. The backpropagation algorithm is used to train the perceptron. The algorithm is discussed below.

Consider the network depicted in Fig. 6. The total input to the ith neuron of any layer is

$$\mathrm{NET}_i = \sum_j w_{ij} O_j \tag{12}$$

where O_j is the output of the jth neuron in the previous layer and W_{ij} the connection weight between the ith neuron in one layer and the jth node of the previous layer. The output of node i is $O_i = F(\mathrm{NET}_i)$ where F is the activation function. For a sigmoidal activation function F can be taken as

$$O_i = 1/(1 + e^{-(\mathrm{NET}_i - \theta_i)}). \tag{13}$$

The parameter θ_i serves as the threshold bias. In the learning phase of such a net, we present the pattern $X = \{x_k\}$, where x_k is the kth component of the feature vector X and ask the net to adjust the weights in all the connecting links such that the desired outputs $\{t_k\}$ are obtained at the output nodes. In fact we ask the net to find a single set of weights and biases that will satisfy all the pairs presented to it. In general, the output $\{O_k\}$ will not be the same as the desired output $\{t_k\}$. For a pattern the square error can be written as

$$E = \sum_k (t_k - O_k)^2. \tag{14}$$

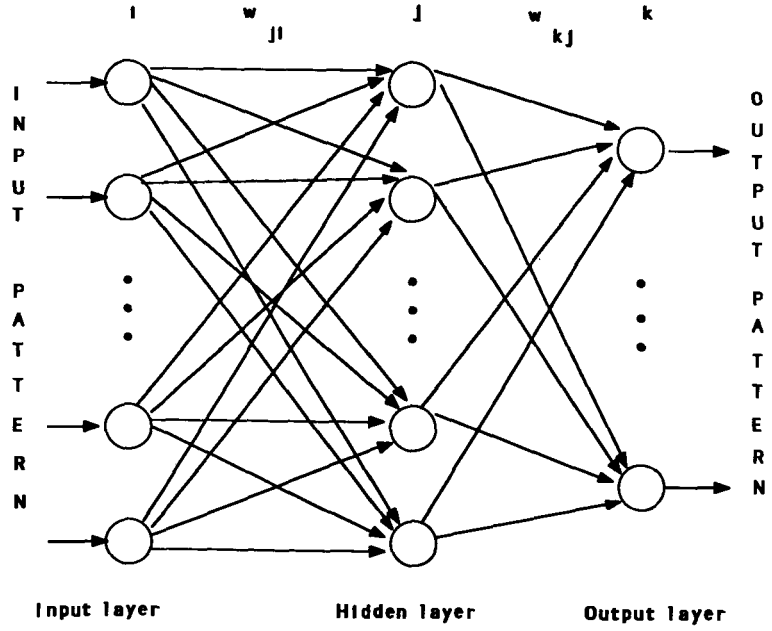

Fig. 6. A three layer perceptron.

In the backpropagation algorithm the adjustment of weights is done in a manner to reduce the error E gradually. It may be mentioned here that $t_k = 1$ for the class in which X belongs and $t_k = 0$, otherwise. In order to minimize E the gradient descent search is applied here. In other words, the incremental change ΔW_{jk} is taken as proportional to the negative gradient $(-\partial E/\partial W_{jk})$. Therefore, the updating rules can be written as $W_{jk}(t+1) = W_{jk}(t) - \eta\, \partial E/\partial W_{jk}$ (η is the step length of learning). After a little algebraic manipulation the updating rule for the output layer becomes

$$W_{jk}(t+1) = W_{jk}(t) + \eta \delta_j O_k$$

where

$$\delta_j = (t_j - O_j) O_j (1 - O_j).$$

For the other layers

$$W_{jk}(t+1) = W_{jk}(t) + \eta \delta_j O_k$$

where

$$\delta_j = O_j (1 - O_j) \sum_k \delta_k W_{kj}.$$

A detailed derivation of this result can be found in reference (14).

5. IMPLEMENTATION

While finding out the angle of variation of a shape, we divided the contour of the shape into equal length segments. However, if there are N points on the boundary of the contour, and if we decided to segment the contour into n constant-point segments then the ratio (N/n), i.e. the number of points on the boundary between the initial and final points of the segment may be fractional. Under this situation the coordinates of the end points can be computed as follows.

Let $p = \text{int}(N/n)$. If P is the starting point of the segment, then the next required point will lie between the $(P+p)$th and $(P+p+1)$th points. If $(x1, y1)$ and $(x2, y2)$ are the coordinates of the $(P+p)$th and $(P+p+1)$th points, respectively, then the coordinates (x, y) of the end point of the segment can be obtained as follows.

If $m = N/n - \text{int}(N/n)$ then $x = mx2 + (1-m)x1$ and $y = my2 + (1-m)y1$.

The investigations for the AR vectors (obtained from the angle of variation) were carried out on four different shapes (Fig. 7) of various sizes. We confined our study to model orders 2, 3 and 4 because for higher order models the computation becomes quite extensive. Experiments have also been carried out with noisy shapes. From each reference shape, various noisy shapes are generated by adding different degrees of random noise to the contour points of the shape.[5] Each contour point is assigned a probability p of retaining its original coordinates in the image plane and probability $q = (1-p)$ of being randomly assigned the coordinates of one of its eight neighbouring pixels.

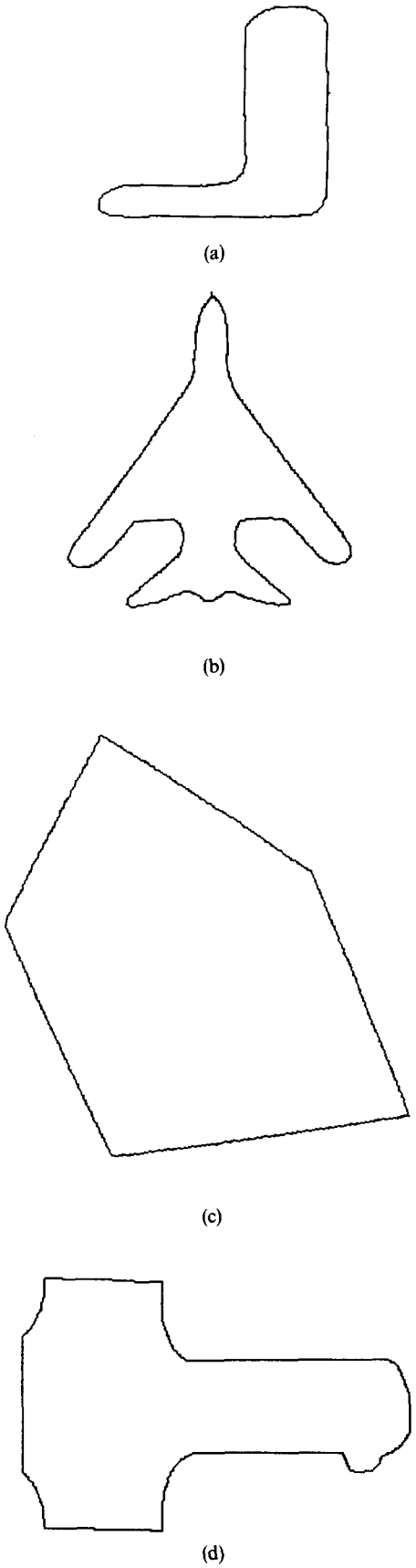

Fig. 7. Uncorrupted shapes.

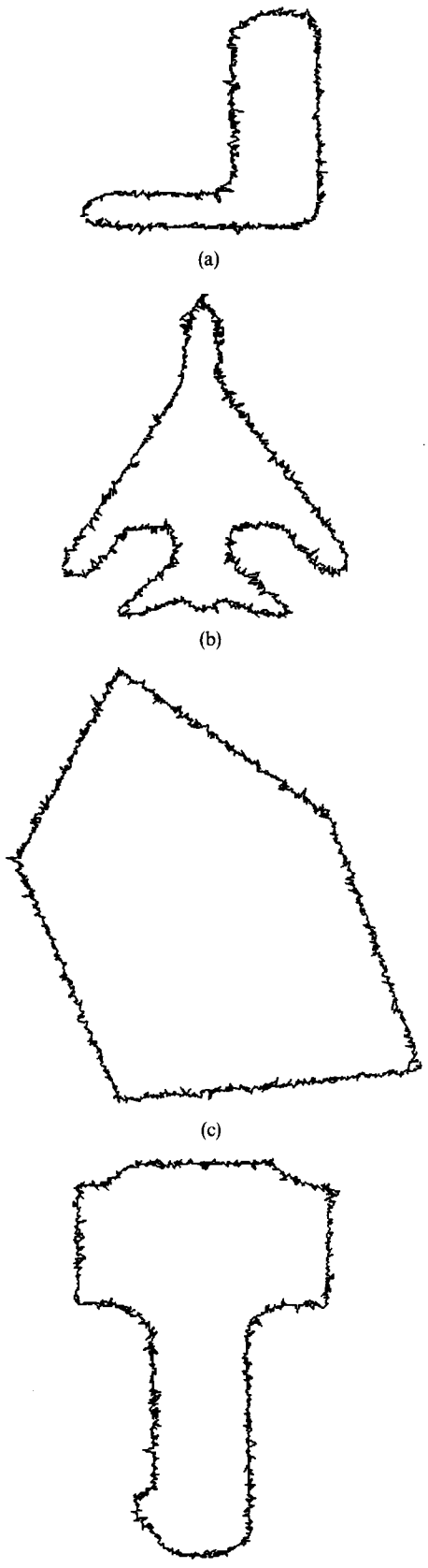

Fig. 8. Noise corrupted shapes.

The degree of noise is increased by increasing the noise level q. Noise may be further increased by repeating the process several times. Addition of noise through this procedure distorts the contour and thus changes the overall shape. Figure 8 depicts some noisy shapes. As an illustration the AR vectors for the original shapes and two noisy versions of them are shown in Table 1. Experimental investigation shows that AR vectors obtained after iterative refinement using the adaptive model are more consistent.

The Multilayer Perceptron Network is then trained with these input feature vectors for classification purpose. The number of layers and the number of neurons in the hidden layers are varied. The number of neurons in the input layer varies with the dimension of the input vectors. On the other hand, the number of output neurons is the same as the number of classes, i.e. four (as four shapes are considered here). In order to train the neural net, ten noisy samples of each shape are generated (i.e. total of 40 samples). The network is then trained with the feature vectors generated from these noisy samples, which form the training set. The training set is cyclically inputted to the network and iterated a number of times, until the weights stabilize. After this, unknown shape vectors are classified using the trained net. The recognition scores for three and four layer perceptrons are given in Table 2.

Table 1. AR vectors for original and noise corrupted shapes

Shape No.	Original shape	Noisy shape 1	Noisy shape 2
1	0.1470	0.1877	0.2067
	−0.1647	−0.1805	−0.1708
	−0.1012	−0.1134	−0.1149
	8.9116	10.2215	10.4232
2	0.2233	0.2268	0.1502
	−0.2330	−0.2443	−0.1695
	0.1527	0.1654	0.1414
	24.5230	24.7038	26.0620
3	0.6888	0.6105	0.4704
	−0.2779	−0.3294	−0.0062
	−0.0542	−0.0406	−0.2404
	7.7550	9.1158	11.4332
4	0.0387	0.0355	−0.0163
	−0.1198	−0.1452	−0.1396
	0.0624	0.0720	−0.0429
	18.1356	21.2766	20.3376

Table 2. Recognition scores

No. of layers	Model order	No. of hidden neurons	Percentage score
3	3	2	97.50
		3	100.00
		4	100.00
4	3	3,3	99.00

As can be seen from Table 2, the recognition score increases with the increase in the number of hidden neurons. In our investigation, we found that the recognition score does not increase if the number of neurons is increased beyond four (with AR process of order 3). Increasing the number of layers of the perceptron beyond three, does not increase the recognition score. The AR process of order 3 seems to be adequate for representing shapes under the proposed scheme. It may be noted that for a three layer perceptron with an AR process of order 3 the total number of weights to be updated in every iteration is only 32 even when we take four neurons in the intermediate layer. This is indeed very low compared to 3656.[5] Moreover, the total number of neurons in this case is only 12.

6. CONCLUSION

A review of some shape analysis techniques has been provided. A new scheme for shape representation using a constant-point polygonal approximation which can be applied to both concave and convex closed simple shapes has been suggested. The approximation scheme is such that it preserves the structural information of the shape and there is no chance of missing any spike on the contour. The sequence of angle of variation between two consecutive linear segments is then taken as the primary feature. This angle sequence is rotation, translation and size invariant. Autoregressive models are then used to find features for shape analysis. A multilayer perceptron network has been used for classification. The network is initially trained with the AR coefficient vectors derived from a set of shapes (including some noisy shapes) and then the trained net is used for recognition of unknown shapes. Application of the neural network enables one to separate classes which have complicated boundaries. Robustness of the proposed shape representation scheme and the effectiveness of the neural net have been established experimentally.

REFERENCES

1. S. R. Dubois and F. H. Glanz, An autoregressive model approach to two dimensional shape classification, *IEEE Trans. Pattern Analysis Mach. Intell.* **8**(1), 55–66 (1986).
2. M. Das, M. J. Paulik and N. K. Loh, A bivariate autoregressive modeling technique for analysis and classification of planar shapes, *IEEE Trans. Pattern Analysis Mach. Intell.* **12**(1), 97–103 (1990).
3. L. Gupta and M. D. Srinath, Contour sequence moments for the classification of closed planar shapes, *Pattern Recognition* **20**, 267–272 (1987).
4. Z. You and A. K. Jain, Performance evaluation of shape matching via chord length distribution, *Comput. Graphics Image Process.* **28**, 185–198 (1984).
5. L. Gupta, M. R. Sayeh and R. Tamanna, A neural network approach robust shape classification, *Pattern Recognition* **23**, 563–568 (1990).
6. B. Bhanu and O. D. Faugeras, Shape matching of two-dimensional objects, *IEEE Trans. Pattern Analysis Mach. Intell.* **6**(2), 137–156 (1984).
7. E. Persoon and K. S. Fu, Shape discrimination using fourier descriptor, *IEEE Trans. Syst. Man Cybern.* 7(3), 170–179 (1977).
8. T. Pavlidish, A review of algorithms for shape analysis, *Comput. Graphics Image Process.* 7, 243–258 (1978).
9. R. L. Kashyap and R. Chellappa, Stochastic model for closed boundary analysis: representation and reconstruction, *IEEE Trans. Inf. Theory* **IT-27**(5), 627–637 (1981).
10. C. T. Zahn and R. Z. Roskies, Fourier descriptors for plane closed curves, *IEEE Trans. Comput.* **C-21**, 269–281 (1972).
11. R. P. Lippman, An introduction to computing with neural nets, *IEEE Trans. Acoust. Speech Signal Process.* 4–22 (April 1987).
12. S. E. Troxel, S. K. Rogers and M. Kabrisky, The use of neural networks in PSRI target recognition, *IEEE Int. Conf. Neural Networks*, San Diego, California, 24–27 July (1988).
13. M. R. Sayeh and J. Y. Han, Pattern recognition using a neural network, *Advances in Intelligent Robotics Systems, SPIE's Cambridge Symp. Optical and Optoelectronic Engineering*, Cambridge, Massachusetts (1987).
14. Y. H. Pao, *Adaptive Pattern Recognition and Neural Network*. Addison-Wesley, Reading, Massachusetts (1989).

About the Author—NIKHIL R. PAL obtained the B.Sc. (Hons.) in physics and the M.B.M. (operations research) in 1979 and 1982, respectively, from the University of Calcutta. He received the M.Tech. and Ph.D. in computer science from the Indian Statistical Institute, Calcutta, in 1984 and 1991, respectively. At present he is with the Electronics and Communication Sciences Unit of the Indian Statistical Institute, and currently visiting the University of West Florida, Pensacola, Florida, He is also a guest lecturer at the University of Calcutta. His research interests include image processing, pattern recognition, artificial intelligence, fuzzy sets and systems, uncertainty measures and neural networks.

About the Author—PRATIK PAL obtained his B.Tech. degree in electronics from the Jadavpur University in 1989 and M.Tech. degree in computer science from the Indian Statistical Institute, Calcutta, in 1991. Currently he is associated with Tata Consultancy Services, Calcutta. His research interests include shape analysis, neural networks and software engineering.

About the Author—ANUPAM K. BASU obtained his M.Stat. degree in 1989 and M.Tech. in computer science in 1991 both from the Indian Statistical Institute, Calcutta. He is currently associated with Tata Consultancy Services, Calcutta. His research interests include shape analysis, neural networks and software engineering.