Classification of Invariant Image Representations Using a Neural Network

ALIREZA KHOTANZAD, MEMBER, IEEE, AND JIIN-HER LU, STUDENT MEMBER, IEEE

Abstract—**In this paper, a neural network (NN) based approach for classification of images represented by translation-, scale-, and rotation-invariant features is presented. The utilized network is a multilayer perceptron (MLP) classifier with one hidden layer. The back-propagation learning is used for its training. Two types of features are used: moment invariants derived from geometrical moments of the image, and the newly developed Zernike moment based features. Zernike moments are the mapping of the image onto a set of complex orthogonal polynomials. The performance of the MLP is compared to those of three other traditional statistical classifiers, namely, Bayes, nearest-neighbor, and minimum-mean-distance. Through extensive experimentation with noiseless as well as noisy binary images of all English characters (26 classes), the following conclusions are reached: 1) the MLP outperforms the other three classifiers, especially when noise is present, 2) the nearest-neighbor classifier performs about the same as the NN for the noiseless case, 3) the NN can do well even with a very small number of training samples, 4) the NN has a good degree of fault tolerance, and 5) the Zernike moment based features possess strong class separability power and are more powerful than moment invariants.**

I. Introduction

RECENT developments in the field of neural networks (NN's) have provided potential alternatives to the traditional techniques of pattern recognition. These nets, which are inspired from studies of biological nervous systems, are composed of many simple nonlinear computational elements (neurons or nodes) which are connected by links with variable weights. The inherent parallelism of these networks allows rapid pursuit of many hypotheses in parallel, resulting in high computation rates. Moreover, they provide a greater degree of robustness or fault tolerance than conventional computers because of the many processing nodes, each of which is responsible for a small portion of the task. Damage to a few nodes or links thus does not impair overall performance significantly.

Neural networks can perform different tasks, one of which is in the context of a supervised classifier. This is a decision-making process which requires the net to identify the class or category which best represents an input pattern. It is assumed that the net has already adapted to the classes it is expected to recognize through a learning process using labeled training prototypes from each category.

Pattern recognition is an essential part of any high-level image analysis system. The goal of a typical computer vision system is to analyze images of a given scene and recognize the content of the scene. Such systems are now in use in a variety of fields, among them robotics, military reconnaissance, remote sensing, document processing, and industrial automation. Most of these systems share a general structure which is composed of four building blocks. The first is image acquisition—converting a scene into an array of numbers that can be manipulated by the computer. The second is preprocessing, which involves removing noise, enhancing the picture, and, if necessary, segmenting the image into meaningful regions to be analyzed separately. The third is feature extraction, whereby the image is represented by a set of numerical "features" to remove redundancy from the data and reduce its dimension. The selected feature set must possess much of the useful information (in the sense of discriminability) present in the original data. Selection of "good" features is a crucial step in the process since the next stage sees only these features and acts upon them. "Good" features are those satisfying two requirements. (i) Small intraclass invariance—slightly different shapes with similar general characteristics should have numerically close values, and (ii) large interclass separation—features from different classes should be quite different numerically. Additionally, a flexible recognition system must be able to recognize an object regardless of its orientation, size, and location in the field of view. This requirement translates into rotation-, scale-, and translation-invariancy properties for the extracted features. The fourth building block is classification. This is the last stage of an image recognition system, where a class label is assigned to the unknown image/object by examining its extracted features and comparing them with class representations that the classifier has learned during its training stage. A conventional statistical classifier is a designed rule based on hypothesis generation and verification or other statistical analysis paradigms.

The main focus of this research is on the feature extraction and classification stages. Specifically, we will examine the performance of a neural network classifier known as multilayer perceptron (MLP) in conjunction with two invariant moment-based feature sets, namely

Manuscript received November 30, 1988; revised August 2, 1989. This work was supported in part by DARPA under Grant MDA 903-86-C-0182.

A. Khotanzad is with the Image Processing and Analysis Laboratory, Electrical Engineering Department, Southern Methodist University, Dallas, TX 75275.

J.-H. Lu was with the Image Processing and Analysis Laboratory, Electrical Engineering Department, Southern Methodist University, Dallas, TX 75275. He is now with the Image Recognition Equipment Corp., Richardson, TX 75081.

IEEE Log Number 9034984.

classical moment invariants and the newly developed Zernike moment features. In each case, we will compare the performance of the NN to those of three other conventional statistical classifiers, namely the Bayes, the minimum-mean-distance, and the nearest-neighbor classifier. We will also experimentally examine the noise tolerance of these different classifiers and explore the advantages of the NN-based system.

The organization of the paper is as follows. Section II discusses the moment-invariant features. In Section III, Zernike-moment-based features are defined. Section IV is devoted to the utilized neural network classifier and its back-propagation learning algorithm. In Section V the other three conventional classifiers are briefly described. Section VI reports the experimental results on a 26-class data set consisting of all the English characters. Section VII gives the conclusion of our study.

II. Moment-Invariant Features

Moment invariants are a set of nonlinear functions which are invariant to translation, scale, and orientation and are defined on geometrical moments of the image. They were first introduced by Hu [5]. Dudani *et al.* [3] successfully applied them to aircraft identification. Wong and Hall [14] used them to match radar images to optical images. Khotanzad and Hong [8] utilized them in texture classification.

Given a two-dimensional $M \times M$ image $\{ f(x, y); x, y = 0, \cdots, M-1 \}$, the $(p+q)$th geometrical moment is defined as

$$m_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{M-1} x^p y^q f(x, y) \quad \text{for } p, q = 0, 1, 2, \cdots. \tag{1}$$

To keep the dynamic range of m_{pq} consistent for different size images, the $M \times M$ image plane is first mapped onto a square defined by $x \in [-1, +1]$, $y \in [-1, +1]$. This implies that grid locations will no longer be integers but will have real values in the $[-1, +1]$ range. This changes the definition of m_{pq} to

$$m_{pq} = \sum_{x=-1}^{+1} \sum_{y=-1}^{+1} x^p y^q f(x, y). \tag{2}$$

To make these moments invariant to translation, one can define a central moment as

$$\mu_{pq} = \sum_{x=-1}^{+1} \sum_{y=-1}^{+1} (x - \bar{x})^p (y - \bar{y})^q f(x, y) \tag{3}$$

with

$$\bar{x} = \frac{m_{10}}{m_{00}} \quad \text{and} \quad \bar{y} = \frac{m_{01}}{m_{00}}.$$

Central moments can be normalized to become invariant to scale change by defining

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}, \qquad \gamma = \frac{p+q}{2} + 1. \tag{4}$$

A set of nonlinear functions defined on η_{pq} which are invariant to rotation, translation, and scale change has been derived [5]. They are

$$\begin{aligned}
\phi_1 &= \eta_{20} + \eta_{02} \\
\phi_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
\phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
\phi_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \\
\phi_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12}) \cdot [(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\
&\quad + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03}) \cdot [3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
\phi_6 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
&\quad + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{03} + \eta_{21}).
\end{aligned} \tag{5}$$

The numerical values of ϕ_1 to ϕ_6 are very small. To avoid precision problems the logarithms of the absolute values of these six functions, i.e. $\log |\phi_i|$, $i = 1, \cdots, 6$, are selected as features representing the image.

III. Zernike Moment Features

The definition of regular geometrical moments has the form of a projection of the $f(x, y)$ function onto the monomial $x^p y^q$. Unfortunately, the basis set $x^p y^q$ is not orthogonal. Consequently, features defined on functions of m_{pq} lack the optimality in information redundancy and other useful properties that might result from using orthogonal basis functions.

In [15], Zernike introduced a set of complex polynomials which form a complete orthogonal set over the interior of the unit circle, i.e. $x^2 + y^2 = 1$. Let the set of these polynomials be denoted by $\{ V_{nm}(x, y) \}$. The form of these polynomials is

$$V_{nm}(x, y) = V_{nm}(\rho, \theta) = R_{nm}(\rho) \exp(jm\theta) \tag{6}$$

where

- n positive integer or zero,
- m positive and negative integers subject to constraints $n - |m|$ even, $|m| \leq n$,
- ρ length of vector from origin to (x, y) pixel,
- θ angle between vector ρ and x axis in counterclockwise direction,
- $R_{nm}(\rho)$ radial polynomial defined as

$$R_{nm}(\rho) = \sum_{s=0}^{(n-|m|)/2} \frac{(-1)^s [(n-s)!] \rho^{n-2s}}{s! \left(\frac{n+|m|}{2} - s\right)! \left(\frac{n-|m|}{2} - s\right)!}.$$

Note that $R_{n,-m}(\rho) = R_{nm}(\rho)$.

These polynomials are orthogonal and satisfy

$$\iint_{x^2+y^2\le 1} [V_{nm}^*(x, y)] V_{pq}(x, y)\, dx\, dy = \frac{\pi}{n+1}\delta_{np}\delta_{mq} \tag{7}$$

with

$$\delta_{ab} = \begin{cases} 1 & a = b \\ 0 & \text{otherwise.} \end{cases}$$

Zernike moments are the projection of the image function onto these orthogonal basis functions. The Zernike moment of order n with repetition m for a continuous image function, $f(x, y)$, that vanishes outside the unit circle is

$$A_{nm} = \frac{n+1}{\pi} \iint_{x^2+y^2\le 1} f(x, y) V_{nm}^*(\rho, \theta)\, dx\, dy. \tag{8}$$

For a digital image, the integrals are replaced by summations to get

$$A_{nm} = \frac{n+1}{\pi} \sum_x \sum_y f(x, y) V_{nm}^*(\rho, \theta), \quad x^2 + y^2 \le 1. \tag{9}$$

To compute the Zernike moments of a given image, the center of the image is taken as the origin and pixel coordinates are mapped to the range of the unit circle, i.e. $x^2 + y^2 \le 1$. Those pixels falling outside the unit circle are not used in the computation. Also note that $A_{nm}^* = A_{n,-m}$.

The features defined on Zernike moments are derived by using rotational properties of these moments. Consider a rotation of the image through angle ϕ. The relationship between A'_{nm} and A_{nm}, the Zernike moment of the rotated image and the unrotated one, is [12]

$$A'_{nm} = A_{nm} \exp(-jm\phi). \tag{10}$$

This relation shows that Zernike moments have simple rotational transformation properties; each Zernike moment merely acquires a phase shift on rotation. This simple property leads to the conclusion that the magnitudes of the Zernike moments of a rotated image function remain identical to those before rotation. Thus $|A_{nm}|$, the magnitude of the Zernike moment, can be taken as a rotation-invariant feature of the underlying image function. Note that since $A_{n,-m} = A_{nm}^*$, then $|A_{nm}| = |A_{n,-m}|$; thus one can concentrate on $|A_{nm}|$ with $m \ge 0$ as far as the defined Zernike features are concerned. Table I lists the rotation-invariant Zernike features and their corresponding numbers from order 0 to order 12.

This rotation-invariant property is illustrated by an experiment. Fig. 1 shows a 64 × 64 binary image of character A and five rotated versions of it, with rotation angles of 30°, 60°, 150°, 180°, and 300°, respectively. Table II is a list of the magnitudes of their Zernike moments for orders 2 and 3, their respective sample mean, μ, sample standard deviation, σ, and $\sigma/\mu\%$, which indicates the percentage of spread of the $|A_{nm}|$ values from their corresponding means. It is observed that rotation invariancy is very well achieved since $\sigma/\mu\%$ values are very small. The reason for not obtaining exact invariances (i.e., σ/μ

TABLE I
LIST OF ZERNIKE MOMENTS AND THEIR CORRESPONDING NUMBERS OF FEATURES FROM ORDER 0 TO ORDER 12

Order	Moments	No. of Moments
0	A_{00}	1
1	A_{11}	1
2	A_{20}, A_{02}	2
3	A_{31}, A_{33}	2
4	A_{40}, A_{42}, A_{44}	3
5	A_{51}, A_{53}, A_{55}	3
6	A_{60}, A_{62}, A_{64}, A_{66}	4
7	A_{71}, A_{73}, A_{75}, A_{77}	4
8	A_{80}, A_{82}, A_{84}, A_{86} A_{88}	5
9	A_{91}, A_{93}, A_{95}, A_{97} A_{99}	5
10	$A_{10,0}$, $A_{10,2}$, $A_{10,4}$, $A_{10,6}$ $A_{10,8}$, $A_{10,10}$	6
11	$A_{11,1}$, $A_{11,3}$, $A_{11,5}$, $A_{11,7}$ $A_{11,9}$, $A_{11,11}$	6
12	$A_{12,0}$, $A_{12,2}$, $A_{12,4}$, $A_{12,6}$ $A_{12,8}$, $A_{12,10}$, $A_{12,12}$	7

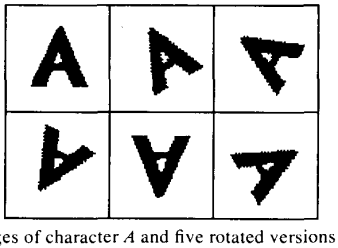


Fig. 1. The images of character A and five rotated versions of it. From top left to right, rotation angles are 0°, 30°, 60°, 150°, 180°, and 300°.

TABLE II
MAGNITUDES OF SOME OF THE ZERNIKE MOMENTS FOR ROTATED IMAGES SHOWN IN FIG. 1 AND THEIR CORRESPONDING STATISTICS

	$\lvert A_{20}\rvert$	$\lvert A_{22}\rvert$	$\lvert A_{31}\rvert$	$\lvert A_{33}\rvert$
0°	439.62	41.79	57.97	172.57
30°	436.70	40.20	63.82	171.96
60°	440.63	40.08	66.28	169.41
150°	438.53	41.55	65.47	170.83
180°	439.01	46.85	62.39	168.47
300°	438.43	39.19	65.77	170.84
μ	438.82	41.61	63.62	170.68
σ	1.32	2.74	3.12	1.53
σ/μ %	0.30	6.57	4.90	0.90

$= 0\%$) is that image function is digital rather than continuous.

The proposed Zernike features are only rotation invariant. To achieve scale and translation invariancy, the image is first normalized with respect to these variables. The Zernike features are then extracted from the scale- and translation-normalized images. The scale and translation normalization is carried out using the regular moments (i.e., m_{pq}) of the image.

Translation invariancy is achieved by transforming the image into a new one whose first-order moments, m_{01} and m_{10}, are both equal to 0. This is done by transforming the original $f(x, y)$ image into the $f(x + \bar{x}, y + \bar{y})$ image, where $\bar{x}$ and $\bar{y}$ are the centroid location of the original image computed as in (3). In other words, the origin is moved to the centroid before moment calculation.

Scale invariancy is accomplished by enlarging or reducing each object such that its zeroth-order moment, m_{00}, is set equal to a predetermined value, β. Note that in the case of binary images m_{00} is the total number of object pixels in the image. Let $f(x/a, y/a)$ represent a scaled version of the image function, $f(x, y)$. Then, the regular moment m_{pq} of $f(x, y)$ and m'_{pq}, the regular moment of $f(x/a, y/a)$, are related by

$$\begin{aligned} m'_{pq} &= \int_x \int_y x^p y^q f\left(\frac{x}{a}, \frac{y}{a}\right) dx\, dy \\ &= \int_x \int_y a^p x^p a^q y^q f(x, y) a^2\, dx\, dy \\ &= \int_x \int_y a^{p+q+2} x^p y^q f(x, y)\, dx\, dy \\ &= a^{p+q+2} \int_x \int_y x^p y^q f(x, y)\, dx\, dy \\ &= a^{p+q+2} m_{pq}. \end{aligned} \tag{11}$$

Since the objective is to have $m'_{00} = \beta$, one can let $a = (\beta/m_{00})^{1/2}$. Substituting $a = (\beta/m_{00})^{1/2}$ into m'_{00}, one obtains $m'_{00} = a^2 m_{00} = \beta$. Thus scale invariancy is achieved by transforming the original image function, $f(x, y)$, into a new function $f(x/a, y/a)$, with $a = (\beta/m_{00})^{1/2}$.

In summary, an image function, $f(x, y)$, can be normalized with respect to scale and translation by transforming it into $g(x, y)$, where

$$g(x, y) = f\left(\bar{x} + \frac{x}{a}, \bar{y} + \frac{y}{a}\right) \tag{12}$$

with $(\bar{x}, \bar{y})$ being centroid of $f(x, y)$ and $a = (\beta/m_{00})^{1/2}$; β a predetermined value for the number of object pixels in the image.

The scale and translation normalization process affects two of the Zernike features: $|A_{00}|$ and $|A_{11}|$. $|A_{00}|$ is going to be the same for all the images and $|A_{11}|$ is equal to 0. This is seen by noting that

$$\begin{aligned} A_{00} &= \frac{1}{\pi} \iint_{x^2+y^2 \le 1} g(x, y) R_{00}(\rho)\, dx\, dy \\ &= \frac{1}{\pi} \iint_{x^2+y^2 \le 1} g(x, y)\, dx\, dy = \frac{1}{\pi} m_{00}. \end{aligned} \tag{13}$$

Since $m_{00} = \beta$, it is evident that $|A_{00}| = \beta/\pi$ for all the normalized images. Therefore, $|A_{00}|$ is not taken as one of the features utilized in the classification.

In the case of A_{11},

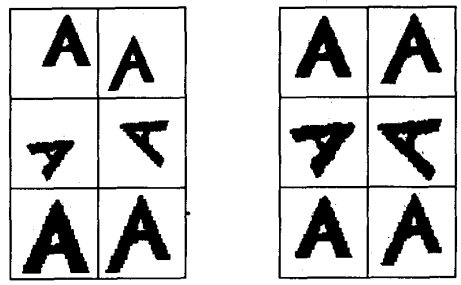
$$A_{11} = \frac{2}{\pi} \iint_{x^2+y^2 \le 1} g(x, y) R_{11}(\rho) \exp(j\theta)\, dx\, dy \tag{14}$$

$$\begin{aligned} \text{Re}[A_{11}] &= \frac{2}{\pi} \iint_{x^2+y^2 \le 1} g(x, y) R_{11}(\rho) \cos\theta\, dx\, dy \\ &= \frac{2}{\pi} \iint_{x^2+y^2 \le 1} g(x, y) \rho \cos\theta\, dx\, dy \\ &= \frac{2}{\pi} \iint_{x^2+y^2 \le 1} g(x, y) x\, dx\, dy = \frac{2}{\pi} m_{10} \end{aligned} \tag{15}$$

and

$$\begin{aligned} \text{Im}[A_{11}] &= \frac{2}{\pi} \iint_{x^2+y^2 \le 1} g(x, y) R_{11}(\rho) \sin\theta\, dx\, dy \\ &= \frac{2}{\pi} \iint_{x^2+y^2 \le 1} g(x, y)\, \rho \sin\theta\, dx\, dy \\ &= \frac{2}{\pi} \iint_{x^2+y^2 \le 1} g(x, y) y\, dx\, dy = \frac{2}{\pi} m_{01}. \end{aligned} \tag{16}$$

Since for all normalized images $m_{10} = m_{01} = 0$, then $|A_{11}| = 0$ for all of them, and $|A_{11}|$ will not be included as one of the utilized features. Thus in the experiments reported in this paper, the defined features start from the second-order moments.



(a) (b)

Fig. 2. (a) Six scaled, translated, and rotated images of character A. (b) The scale and translation normalized images of those shown in (a).

Fig. 2 shows six 64×64 scaled and translated images of the character A along with their scaled and translation normalized versions using $\beta = 800$.

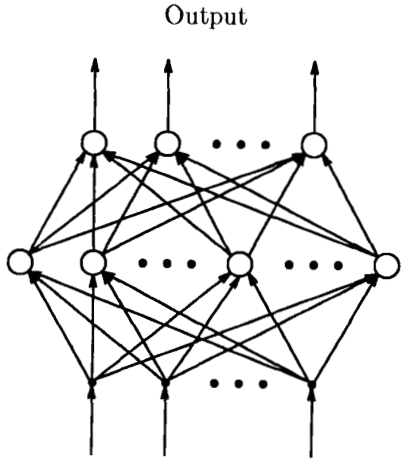


Fig. 3. A multilayer perceptron (MLP) with one hidden layer.

IV. Multilayer Perceptron Classifier

In this study we use a neural network topology known as multilayer perceptron, or MLP. A MLP is a feed-forward net with one or more layers of nodes between the input and output nodes. These in-between layers are called hidden layers. A MLP with one hidden layer is shown in Fig. 3. Connections within a layer or from higher to lower layers are not permitted. Each node in a layer is connected to all the nodes in the layer above it. Training is equivalent to finding proper weights for all the connections such that a desired output is generated for a corresponding input. Using MLP in the context of a classifier requires all output nodes to be set to 0 except for the node that is marked to correspond to the class the input is from. That desired output is 1. In our study, the inputs are either the moment-invariant features or the Zernike moment features extracted from the image.

MLP's were not used in the past because of a lack of effective training algorithms for them. This has recently changed due to development of an iterative gradient procedure known as the back-propagation algorithm [11]. According to this algorithm, which is outlined next, for each pattern in the training set, learning proper weights is conducted by 1) computing the discrepancy between the desired and actual outputs and 2) feeding back this error signal level by level to the inputs, changing the connection weights in such a way as to modify them in proportion to their responsibility for the output error. The major steps of the algorithm are as follows:

Step 1: Initialize all w_{ij}'s to small random values with w_{ij} being the value of the connection weight between unit j and unit i in the layer below.

Step 2: Present an input from class m and specify the desired output. The desired output is 0 for all the output nodes except the mth node, which is 1.

Step 3: Calculate actual outputs of all the nodes using the present value of w_{ij}. The output of node j, denoted by y_j, is a nonlinear function of its total input:

$$y_j = \frac{1}{1 + \exp\left(-\sum_i y_i w_{ij}\right)}. \tag{17}$$

This particular nonlinear function is called a sigmoid function.

Step 4: Find an error term, δ_j, for all the nodes. If d_j and y_j stand for, respectively, the desired and actual value of a node, then for an output node,

$$\delta_j = (d_j - y_j) y_j (1 - y_j) \tag{18}$$

and for a hidden layer node,

$$\delta_j = y_j (1 - y_j) \sum_k \delta_k w_{jk} \tag{19}$$

where k is over all nodes in the layer above node j.

Step 5: Adjust weights by

$$w_{ij}(n+1) = w_{ij}(n) + \alpha \delta_j y_i + \zeta\left(w_{ij}(n) - w_{ij}(n-1)\right) \tag{20}$$

where $(n+1)$, (n), and $(n-1)$ index next, present, and previous, respectively. The parameter α is a learning rate similar to step size in gradient search algorithms, and ζ is a constant between 0 and 1 which determines the effect of past weight changes on the current direction of movement in weight space. This provides a kind of momentum that effectively filters out high-frequency variations of the error surface.

Step 6: Present another input and go back to step 2. All the training inputs are presented cyclically until weights stabilize (converge).

In summary, the above algorithm is an iterative gradient descent procedure in the weight space which minimizes the total error between the desired and actual outputs of all the nodes in the system. It has been shown that a MLP with at most two hidden layers can form any arbitrarily complex decision region in a feature space [10]. However, no specific rule for selection of the number of nodes in the hidden layers has yet been developed.

V. Conventional Statistical Classifiers

To be an acceptable alternative to traditional classifiers, the NN must either outperform or at least do as well as them. In this study, three popular statistical classifier were selected for comparison of their performances to that of the suggested NN. These three classifiers are briefly discussed here.

The optimal (in the sense of probability of error) classifier is the so-called Bayes classifier. It is a parametric classifier and its utilization requires that the joint probability density of the used features of each class as well as the *a priori* probability of each class be available. Such information is rarely available and is normally estimated from the training samples. A usual practice is to assume a multivariate normal distribution for the features of each class and use the sample mean and covariance obtained

from training samples to characterize the underlying density. Although more elaborate schemes for estimation of the form of density function are available [2], we take the above approach for simplicity. Thus, the utilized Bayes classifier will be a suboptimal Bayes. It should also be noted that NN is a nonparametric classifier which does not require such statistical information. It extracts the needed knowledge regarding feature variations during its training.

The Bayes classification rule, under normal density assumption and equal *a priori* densities for all C classes, labels a test sample $\boldsymbol{X} = [x_1, x_2, \cdots, x_n]$, x_i being one of the n utilized features, to class i^* if

$$i^* = \text{Max}_i\, g_i(\boldsymbol{X}), \qquad i = 1, 2, \cdots, C \tag{21}$$

where $g_i(\boldsymbol{X})$ is the discriminant function of the ith class:

$$g_i(\boldsymbol{X}) = -\left[\ln|\Sigma_i| + (\boldsymbol{X} - \mu_i)^T \Sigma_i^{-1} (\boldsymbol{X} - \mu_i)\right] \tag{22}$$

with μ_i ($n \times 1$ vector) and Σ_i ($n \times n$ matrix) representing the sample mean vector and sample covariance matrix of the n features of class i, estimated from the available training samples.

The other two utilized classifiers are nonparametric ones. The first one is the nearest-neighbor rule. When an unknown sample $\boldsymbol{X}$ is to be classified, the nearest neighbor of $\boldsymbol{X}$ is found among the pool of all M available training samples and its label is assigned to $\boldsymbol{X}$. When the number of training samples is large, the probability of error for the nearest-neighbor rules has an upper bound of twice the Bayes error [2].

The distance between $\boldsymbol{X}$ and a training sample is measured using city block distance. This is a mapping from the n-dimensional feature space to a one-dimensional distance space. However, since the feature vector components have different dynamic ranges (see Table II), it is possible for one or a subgroup of them to dominate the distance measure. To prevent this from happening and in order to equally weight distances between each component of feature vectors, the features need to be normalized. The normalization consists of subtracting sample mean and dividing by standard deviation of the corresponding class. Let $\boldsymbol{t}_k^{(i)} = [t_{k1}^{(i)}, t_{k2}^{(i)}, \cdots, t_{kn}^{(i)}]$ and N_i denote the kth n-dimensional training feature vector of the ith class and the number of available training samples of class i respectively. The unknown test sample $\boldsymbol{X}$ is assigned to class i^*, where

$$i^* = \text{Min}_i\, d(\boldsymbol{X}, \boldsymbol{t}_k^{(i)}), \qquad i = 1, 2, \cdots, C$$
$$k = 1, 2, \cdots, N_i \tag{23}$$

$$d(\boldsymbol{X}, \boldsymbol{t}_k^{(i)}) = \sum_{m=1}^{n} |\tilde{x}_m - \tilde{t}_{km}^{(i)}| \tag{24}$$

$$\tilde{x}_m = \frac{x - \overline{t_m^{(i)}}}{\sigma_{t_m}^{(i)}} \qquad \tilde{t}_{k_m} = \frac{t_{k_m} - \overline{t_m^{(i)}}}{\sigma_{t_m}^{(i)}}$$

with $\overline{t_m^{(i)}}$ and $\sigma_{t_m}^{(i)}$ representing the sample mean and standard deviation of the mth element of the n-dimensional training feature vector of class i.

Equation (24) can be simplified to

$$d(\boldsymbol{X}, \boldsymbol{t}_k^{(i)}) = \sum_{m=1}^{n} \left\{ \frac{|x_m - t_{km}^{(i)}|}{\sigma_{t_m}^{(i)}} \right\} \tag{25}$$

which means that advance normalization of features is not necessary. The classification rule takes care of feature balancing through normalization by the standard deviation.

The last considered classifier is a weighted minimum-mean-distance rule. It characterizes each category by mean and standard deviations of the components of its training feature vectors. The weighted distance between an unknown sample $\boldsymbol{X}$ and the mean of the features of class i, $d(\boldsymbol{X}, i)$, is then measured. The weighting factor is the standard deviation of the respective feature. The unknown sample is then assigned to class i^* for which such distance is minimum, i.e.,

$$i^* = \text{Min}_i\, d(\boldsymbol{X}, i), \qquad i = 1, 2, \cdots, C \tag{26}$$

and

$$d(\boldsymbol{X}, i) = \sum_{m=1}^{n} \left\{ \frac{|x_m - \overline{t_m^{(i)}}|}{\sigma_{t_m}^{(i)}} \right\}. \tag{27}$$

Again, weighting by standard deviation is to balance the effect of all m feature vector components on distance d.

VI. Experimental Study

In this section, the results of applying the MLP neural network classifier to a 26-class problem are reported. Two sets of experiments are carried out. The moment-invariant features are used in the first set of experiments and the Zernike features are utilized in the second set. The performance on noiseless as well as noisy images with varying SNR's is examined. In addition, the three described traditional classifiers are applied to each of the considered problems and their performances are compared to that of the NN.

A. The Utilized Data Sets

The data base consists of 64×64 binary images of all 26 uppercase English characters (A to Z). Twenty-four different images from four slightly different silhouettes of each character are generated (for a total of 624). The set of 24 images per character consists of six images with arbitrarily varying scales, orientations, and translations from each of the four considered silhouettes per character. Fig. 4 shows the 24 images of character A. In Fig. 5 the four silhouettes of each of the other characters are shown.

In addition to the above noiseless image set, five other sets of noisy images with respective SNR's of 50, 25, 12, 8, and 5 dB are also constructed from the normalized images of the noiseless set. This is done by randomly selecting some of the 4096 pixels of a noiseless binary image and reversing their values from 0 to 1 or vice versa. The random pixel selection is done according to a uniform probability distribution between 1 and 4096. The SNR is

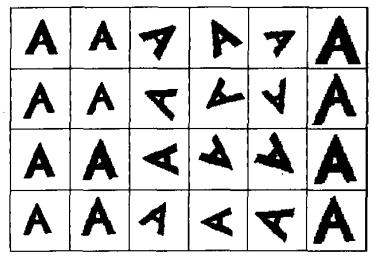


Fig. 4. The 24 scaled, translated, and rotated images of character *A* in the data set. Note the slight variations in shapes of the images shown in the first column.

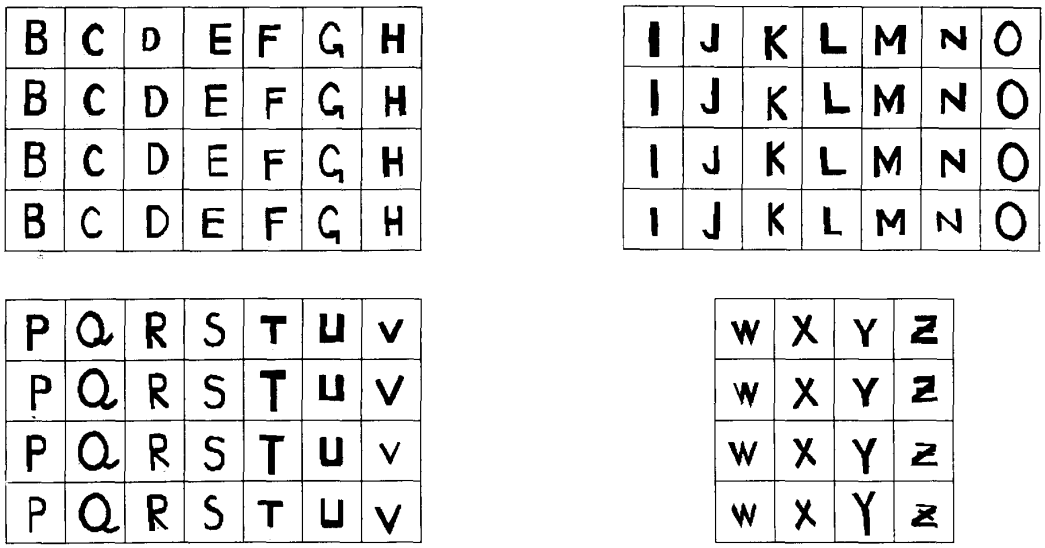


Fig. 5. Four out of the 24 images of letters *B* to *Z* in the data set. The remaining 20 images per character are rotated, scaled, and translated versions of the ones shown.

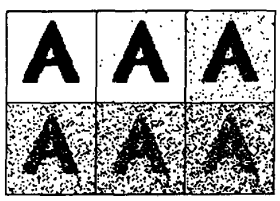


Fig. 6. One sample of image of character *A* with different levels of noise. From top left to right SNR is noiseless, 50, 25, 12, 8, and 5 dB.

computed using 20 log [4096 − L/L], where L is the number of pixels which are different between a noisy image and a clean version. Fig. 6 shows one image of character *A* with different SNR's.

B. General Description of the Experiments

An important parameter in any pattern classification problem is the estimate of the classification error. To compute it, the available samples must be divided into two sets: one for training and one for testing. In each of the following experiments two cases were considered. In the first case, the available samples from each character were divided into halves such that each half contained images of each sihouette. The first half was then used for training and the second half for testing. Therefore, there were 12 training images and 12 testing images per character. Referring to the images of Fig. 4, the images in the first three columns were used for training and those in the last three columns for testing.

In the second case, the training was limited to four unrotated images per character. These four were the four different silhouettes of each character (i.e., the images in the first column of Fig. 4). The remaining 20 images per class were used for testing. This way, the classifier does not see the rotated versions during learning but has to deal with them during testing.

The estimation of the error rate is done by finding the ratio of the number of misclassified test samples to the total number of tested samples.

In experiments with noisy images, the classifier is trained with noiseless images and tested with the noisy ones. Therefore, no noisy images are used for training.

Utilizing Zernike moment features requires the selection of the maximum order, i.e., the highest n. Two different synthesis-based methods for doing so are presented in [6] and [7]. In this study we experimented with $n = 2, \cdots, k$ and varied k from 5 to 12. Based on the entries of Table I, the number of features varies from 10 (corresponding to $k = 5$) to 47 (corresponding to $k = 12$).

The selected parameters for the MLP classifier were as follows: initial weight assignment from the $[-0.5, 0.5]$ interval, step size $\alpha = 0.2$, learning rate $\zeta = 0.7$, number of passes over the training data is 500, the number of input nodes is the same as the number of features in each case, the number of output nodes is 26, and, finally, the number of hidden layer nodes is varied from 5 to 200 in unequal increments. In experiments with Zernike features of order 11 and less, only 40, 45, and 50 hidden layer nodes are considered. This was due to the results of experimentation with order 12, which showed that the maximum classification accuracy happened in that range.

The training features are normalized to have zero mean and unit variance before being input to the MLP. This is necessary in order to ensure that a subgroup of the features do not dominate the weight adjustment process during training. The mth feature is normalized by

$$\tilde{t}_m = \frac{t_m - \bar{t}_m}{\sigma_{t_m}} \tag{28}$$

where $\bar{t}_m$ and σ_{t_m} are the sample mean and standard deviation of the mth training features of all the classes.

The MLP was simulated on a SEQUENT SYMMETRY S81 MIMD parallel computer utilizing six processors, each performing 3 MIPS. The learning for the case of 47 inputs, 50 hidden layer nodes, and 312 training samples took around 65 min.

C. Experiments with Moment-Invariant Features

In the first set of experiments, the six moment-invariant features were utilized. Fig. 7(a)–(f) shows the results of the experiments with the MLP classifier when the number of hidden nodes is varied. For the noiseless case shown in Fig. 7(a) and (d), the results when more than 20 hidden layer nodes were used are quite good. However, the performance significantly worsened when noise was added. The performance was so bad that the experiments were stopped at SNR of 25 dB, and lower SNR's were not considered. The performances of the other classifiers are presented in the form of legends on the corresponding graphs.

D. Experiments with Zernike Features

As the above results indicate, the moment invariant features are very sensitive to noise. In the second set of experiments, Zernike features were utilized. The performance of the second- through 12th-order Zernike features and the MLP classifier on different data sets is shown in Figs. 8 and 9 for the two learning cases. Note that 100% classification accuracies were achieved for noiseless, 50, 25, and 12 dB cases, while accuracies in 90% and 80% range were obtained for 8 and 5 dB sets respectively. It

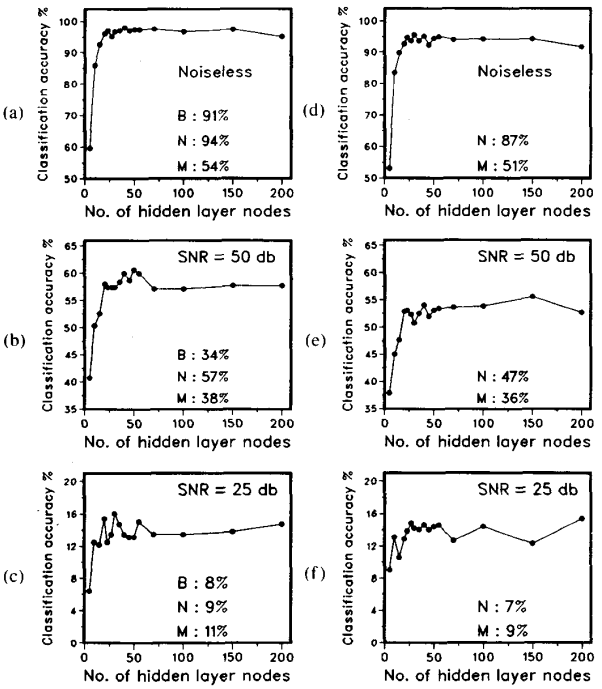


Fig. 7. Classification results using moment invariant features. B, N, and M stand for Bayes, nearest-neighbor, and minimum-mean-distance classifiers, respectively: (a)–(c) 12 samples per class for training and 12 for testing (case 1); (d)–(f) four samples per class for training and 20 for testing (case 2).

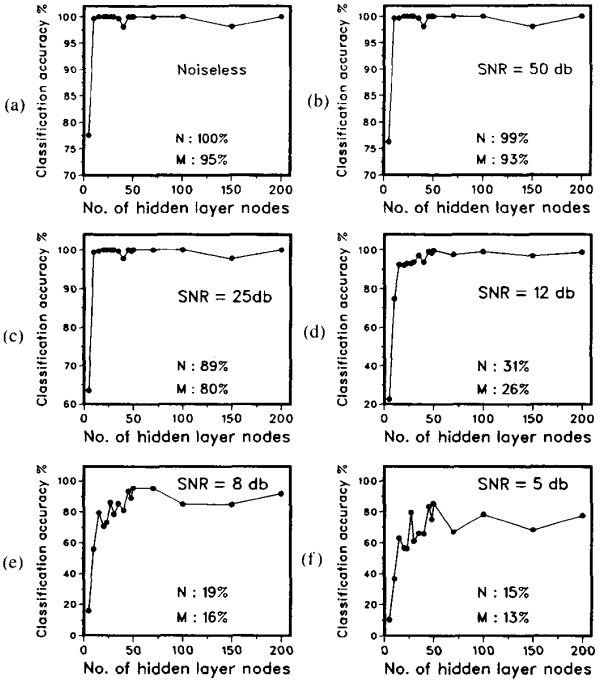


Fig. 8. Classification results using second- through 12th-order Zernike moments (47 features). Twelve images per class are used for training and the remaining 12 for testing. N and M stand for nearest-neighbor and minimum-mean-distance classifiers, respectively.

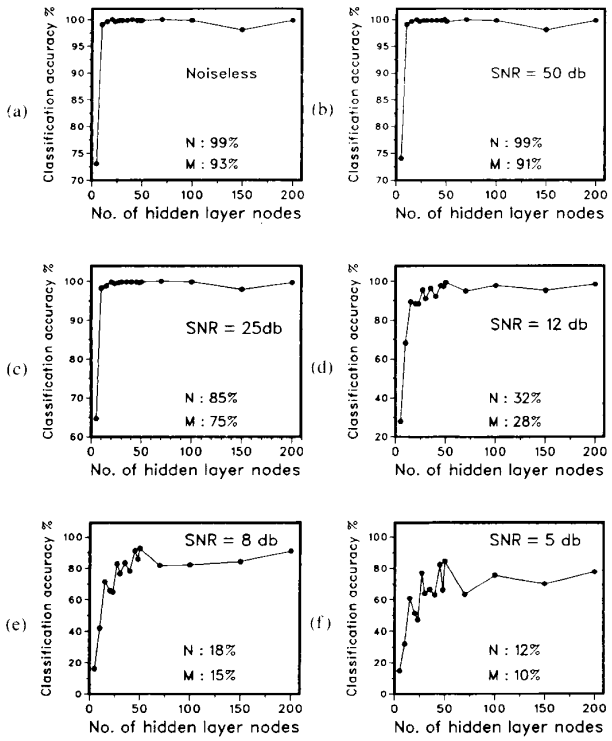


Fig. 9. Classification results using second- through 12th-order Zernike moments (47 features). Four images per class are used for training and the remaining 20 for testing. N and M stand for nearest-neighbor and minimum-mean-distance classifiers, respectively.

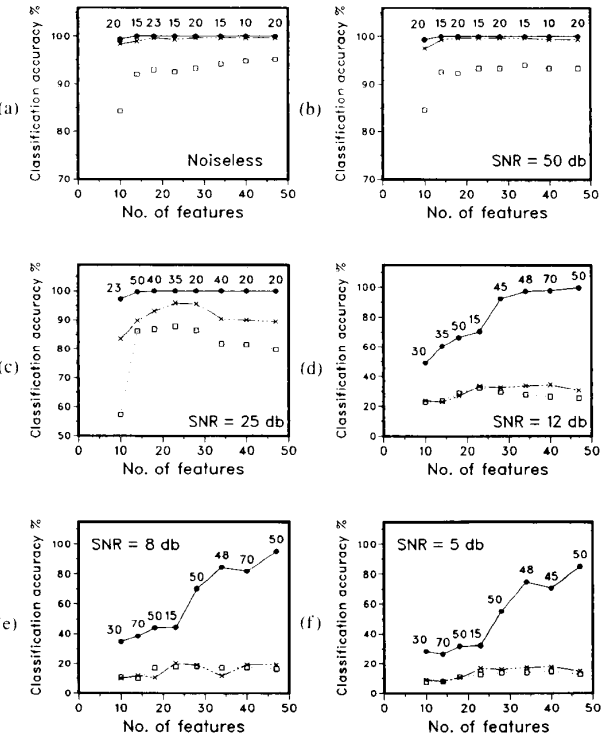


Fig. 10. Classification results using varying number of Zernike features. The numbers on top of the filled circles represent the corresponding number of the used hidden nodes. Twelve images per class are used for training and the remaining 12 for testing. The symbols ●, ×, and □ represent the MLP, nearest-neighbor, and minimum-mean-distance classifiers, respectively.

is clear that the sensitivity to noise was drastically reduced.

We also investigated the performance of Zernike features of orders lower than 12. Second- through kth-order features were utilized, with $k = 5, 6, \cdots, 11$. A summary of the obtained results is plotted in Figs. 10 and 11. For each case only the best classification accuracy obtained while varying the number of hidden layer nodes is plotted and the corresponding number of hidden nodes used is noted above each point.

The performances of the nearest-neighbor and minimum-mean-distance classifiers are also plotted. We were not able to use the Bayes classifier for all the cases. The reason is that the number of features used was too large while the number of training samples was small (12 for case 1 and 4 for case 2). Thus, the estimate of Σ_i, the covariance matrix of class i, becomes singular for some of the classes, which in turn makes the application of the Bayes classifier impossible. However, if only lower order moments are considered, the number of features decreases, enabling one to compute Σ_i's and applying the Bayes classifier. This was possible only when second- through fifth-order moments were considered (for case 1 only). The results are presented in Table III.

E. Effect of the Number of Passes Over Training Set

In the reported experiments, 500 passes over the training set were used to teach the MLP. The effect of varying this number is shown in Fig. 12. The classification accuracy as a function of the number of training passes is plotted on a log scale for the case of testing 12 dB images using a MLP with 50 hidden nodes trained with 47 features of clean images. If desired, an automatic mechanism for stopping the training before it reaches a preset maximum can be developed using similar information. One can test the recognition accuracy of the training samples at certain intervals during training and stop if it becomes sufficiently high. Another option is to keep track of the changes in the weights and stop when they become small. The trade-off in cutting the number of passes over the training set is the additional computation required by these methods.

F. Fault Tolerance

One of the advantages of NN's is that the processing is distributed among many nodes. This provides a good degree of fault tolerance and graceful degradation to the system. Even if some of the nodes fail to function properly, the effect on the overall performance of the system will not be appreciable. We examined this assertion by turning off m randomly selected hidden layer nodes and observing the resulting effect on the system performance.

The MLP with 30 hidden layer nodes and second- through 12th-order Zernike features using 12 training

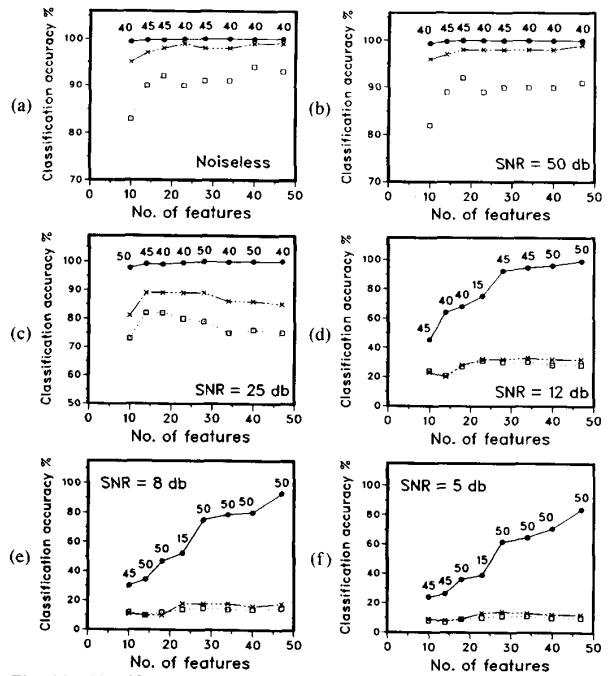


Fig. 11. Classification results using varying number of Zernike features. The numbers on top of the filled circles represent the corresponding number of the used hidden nodes. Four images per class are used for training and the remaining 20 for testing. The symbols •, ×, and □ represent the MLP, nearest-neighbor, and minimum-mean-distance classifiers, respectively.

TABLE III
CLASSIFICATION RESULTS USING BAYES CLASSIFIER AND SECOND- THROUGH FIFTH-ORDER ZERNIKE MOMENTS (TEN FEATURES)

SNR (dB)	Classification Accuracy %
Noiseless	96
50	90
25	59
12	10
8	8
5	7

Twelve images per class are used for training and the remaining 12 for testing.

samples in each class (case 1) was considered for this purpose. After training the net with all 30 nodes functioning, the classification of test images was carried out with m of the hidden nodes not functioning (i.e., their outputs were set to 0); m was varied from 1 to 20 in increments of 1 and for each m, 15 different combinations of m out of 30 nodes were considered. The result is plotted in Fig. 13, which shows a very graceful degradation of the performance.

VII. CONCLUSIONS

In all of the experiments, the MLP neural net performed better than the three conventional classifiers. The performance of the nearest-neighbor classifier was close

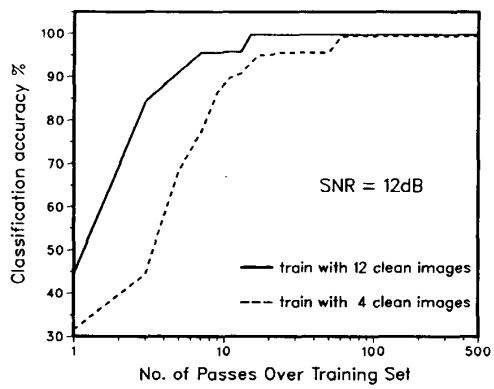


Fig. 12. Classification accuracy of 12-dB images as a function of the number of passes over the training set for both training cases.

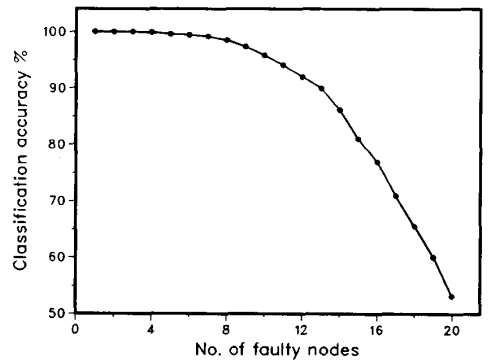


Fig. 13. Degradation in system performance as a function of faulty hidden layer nodes.

to that of the MLP for high SNR images, but the MLP did considerably better on images with low SNR. The minimum-mean-distance classifier performed almost as well as the nearest-neighbor for some cases, but its overall performance was lower than both the MLP and the nearest-neighbor. For those cases where the suboptimal Bayes classifier could be computed, it did worse than the nearest-neighbor.

In all of the examined cases, a number in the range of [20, 50] for hidden units gave the best (or very close to the best) classification accuracy. Utilizing more than 50 hidden layer nodes did not alter the results significantly, especially for high SNR images.

The performance of the NN classifier changed very little when the number of training samples per character was reduced from 12 to 4. However, the performance of the other classifiers got worse.

The NN exhibited a good degree of fault tolerance. Losing up to one fifth of the hidden layer nodes in a 30 hidden layer node network did not alter the performance by any significant amount.

The comparison of the Zernike moment features and the moment-invariant features shows that the former are superior. For noiseless images, the two performed almost the same and since the number of moment-invariant fea-

tures is lower, they may be preferable. However, when noise is added, the moment-invariant features become useless while Zernike features do very well under noisy conditions.

The number of Zernike features (highest order of Zernike moments) required for a good classification increases as SNR decreases. Using second- through 12th-order (47 features) gave good results for all cases. It is interesting to note that for low SNR images, increasing the Zernike features actually decreased the classification accuracy when conventional classifiers were used. This is due to the fact that higher order Zernike moments are more sensitive to noise [13]. However, note that the MLP classifier seems to be insensitive to this effect and its accuracy increases as more features are utilized.

With regard to the number of training samples, the MLP classifier performed about the same whether it was trained with 12 or four images per class, while the traditional classifiers suffered a slight performance degradation.

REFERENCES

[1] D. J. Burr, "Experiments on neural net recognition of spoken and written text," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 36, July 1988.

[2] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[3] S. A. Dudani, K. J. Breeding, and R. B. McGhee, "Aircraft identification by moment-invariants," *IEEE Trans. Comput.*, vol. C-26, pp. 39-45, Jan. 1977.

[4] E. Gullichsen and E. Chang, "Pattern classification by neural network: An experimental system for icon recognition," in *Proc. IEEE 1st Int. Conf. Neural Network* (San Diego, CA), June 21-24, 1987.

[5] M. Hu, "Visual pattern recognition by moment invariants," *IRE Trans. Inform. Theory*, vol. IT-8, pp. 179-187, Feb. 1962.

[6] A. Khotanzad and Y. H. Hong, "Rotation invariant pattern recognition using Zernike moments," in *Proc. 9th ICPR* (Rome, Italy), Nov. 14-17, 1988, pp. 326-328.

[7] A. Khotanzad and Y. H. Hong, "Zernike moment based rotation between invariant features for pattern recognition," in *Proc. SPIE Conf. Intelligent Robots and Computer Vision* (Cambridge, MA), Nov. 6-11, 1988.

[8] A. Khotanzad and Y. H. Hong, "Rotation and scale invariant features for texture classification," in *Proc. IASTED Int. Symp. Robotics and Automation* (Santa Barbara, CA), May 1987, pp. 16-17.

[9] A. Khotanzad and J. H. Lu, "Distortion invariant character recognition by a multilayer perceptron and back-propagation learning," in *Proc. IEEE 2nd Int. Conf. Neural Networks* (San Diego, CA), July 24-27, 1988, pp. 625-632.

[10] R. P. Lippmann, "An introduction to computing with neural nets," *IEEE ASSP Mag.*, vol. 4, pp. 4-22, Apr. 1987.

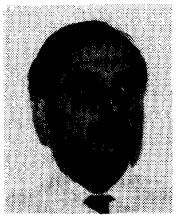
[11] D. E. Rumelhart and J. L. McClelland, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1, *Foundations*. Cambridge, MA: MIT Press, 1986.

[12] M. Teague, "Image analysis via the general theory of moments," *J. Opt. Soc. Amer.*, vol. 70, no. 8, pp. 920-930, Aug. 1980.

[13] C. H. Teh and R. T. Chin, "On image analysis by the methods of moments," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 10, July 1988.

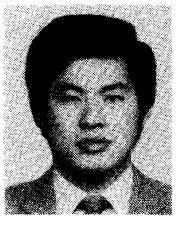
[14] R. Y. Wong, and E. L. Hall, "Scene matching with invariant moment," *Computer Graphics and Image Processing*, vol. 8, pp. 16-24, 1978.

[15] F. Zernike, *Physica*, vol. 1, pp. 689, 1934.



Alireza Khotanzad (S'79-M'84) was born in Tehran, Iran, in 1956. He received the B.S., M.S., and Ph.D. degrees in electrical engineering from Purdue University, West Lafayette, IN, in 1978, 1980, and 1983, respectively.

In 1984 he joined the faculty of the Department of Electrical Engineering, Southern Methodist University, Dallas, TX, where he is now an Assistant Professor. His research interests are in the areas of computer vision, pattern recognition, neural networks, and texture analysis.



Jiin-Her Lu (S'86) was born in Taiwan, Republic of China, in 1953. He received the B.S.E.E. degree from the Taiwan Provincial College of Marine and Oceanic Technology, in 1975 and the M.S. and Ph.D. degrees in electrical engineering in 1985 and 1989, respectively, from Southern Methodist University, Dallas, TX.

He joined Image Recognition Equipment Corp. in 1990. His research interests include image processing and analysis, computer vision, pattern recognition, signal processing, neural networks, and related applications.