Handwritten digit recognition: Benchmarking of state-of-the-art techniques

Article *in* Pattern Recognition · October 2003

DOI: 10.1016/S0031-3203(03)00085-2 · Source: DBLP

4 authors, including:

Cheng-Lin Liu
Chinese Academy of Sciences
266 PUBLICATIONS **6,738** CITATIONS

SEE PROFILE

Hiromichi Fujisawa
Waseda University
82 PUBLICATIONS **2,279** CITATIONS

SEE PROFILE

PERGAMON

Pattern Recognition 36 (2003) 2271–2285

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Handwritten digit recognition: benchmarking of state-of-the-art techniques

Cheng-Lin Liu*, Kazuki Nakashima, Hiroshi Sako, Hiromichi Fujisawa

Central Research Laboratory, Hitachi, Ltd. 1-280 Higashi-koigakubo, Kokubunji-shi, Tokyo 185-8601, Japan

Received 13 June 2002; received in revised form 23 December 2002; accepted 28 January 2003

Abstract

This paper presents the results of handwritten digit recognition on well-known image databases using state-of-the-art feature extraction and classification techniques. The tested databases are CENPARMI, CEDAR, and MNIST. On the test data set of each database, 80 recognition accuracies are given by combining eight classifiers with ten feature vectors. The features include chaincode feature, gradient feature, profile structure feature, and peripheral direction contributivity. The gradient feature is extracted from either binary image or gray-scale image. The classifiers include the k-nearest neighbor classifier, three neural classifiers, a learning vector quantization classifier, a discriminative learning quadratic discriminant function (DLQDF) classifier, and two support vector classifiers (SVCs). All the classifiers and feature vectors give high recognition accuracies. Relatively, the chaincode feature and the gradient feature show advantage over other features, and the profile structure feature shows efficiency as a complementary feature. The SVC with RBF kernel (SVC-rbf) gives the highest accuracy in most cases but is extremely expensive in storage and computation. Among the non-SV classifiers, the polynomial classifier and DLQDF give the highest accuracies. The results of non-SV classifiers are competitive to the best ones previously reported on the same databases.

Keywords: Handwritten digit recognition; The state of the art; Feature extraction; Pattern classification; Discriminative learning; Support vector classifier

1. Introduction

Handwritten digit recognition is an active topic in OCR applications and pattern classification/learning research. In OCR applications, digit recognition is dealt with in postal mail sorting, bank check processing, form data entry, etc. For these applications, the performance (accuracy and speed) of digit recognition is crucial to the overall performance. While in pattern classification and machine learning communities, the problem of handwritten digit recognition is a good example to test the classification performance [1].

The performance of character recognition largely depends on the feature extraction approach and the classification/learning scheme. [1] For feature extraction of character recognition, various approaches have been proposed [2]. Many experiments have shown that the stroke direction feature [3,4] is one of the most efficient features for handwritten character recognition [5]. This feature is basically statistical and is represented in histograms of local direction elements. Further, the local direction and the element location can be blurred to improve the discrimination ability [6]. Further improvements can be achieved by adding curvature

* Corresponding author. Tel.: +81-42-323-1111; fax: +81-42-327-7778.

E-mail address: liucl@crl.hitachi.co.jp (C.-L. Liu).

[1] We treat the classification and learning in a single issue because the learning algorithm is dependent on the classification scheme, and the classification performance largely depends on the learning algorithm.

0031-3203/03/$30.00
doi:10.1016/S0031-3203(03)00085-2

feature [7] or local structure features [8–10]. The statistical features and local structural features are stored in a feature vector, which corresponds to a point in the feature space.

The task of classification is to partition the feature space into regions corresponding to source classes or assign class confidences to each location in the feature space. Statistical techniques [11–13] and neural networks [14,15] have been widely used for classification due to the implementation efficiency. Statistical classifiers are divided into parametric and non-parametric ones. They include the linear discriminant function (LDF), the quadratic discriminant function (QDF), the nearest-neighbor (1-NN) and k-NN classifiers, the Parzen window classifier, etc. In character recognition community, the modified quadratic discriminant function (MQDF2) proposed by Kimura et al. has been widely used since it reduces the complexity and improves the classification performance of QDF [16,17]. Neural networks for classification include the multilayer perceptron (MLP) [18], the radial basis function (RBF) network [14,19,20], the polynomial classifier [21,22], etc.

The main difference between statistical and neural classifiers is that the parameters of neural networks are optimized in discriminative supervised learning with aim to separate the patterns of different classes. When the network structure is appropriately designed and the training sample size is large, neural networks are able to give high classification accuracy to unseen test data. The so-called discriminative models also include the learning vector quantization (LVQ) classifier, which employs the same classification rule as the 1-NN classifier but the prototypes are designed in discriminative learning [23].

In recent years, a new type of classifier, support vector machine (SVM) [24] is receiving attention increasingly. SVM is based on the statistical learning theory of Vapnik [25] and quadratic programming optimization. An SVM is basically a binary classifier and multiple SVMs can be combined to form a classification system for multi-class classification. We generally call an SVM classification system as an SV classifier (SVC). The superior classification performance of SVC has been justified in numerous experiments, particularly in high dimensionality and small sample size.

This paper reports the new results of handwritten digit recognition using state-of-the-art feature extraction techniques and learning/classification algorithms. We did not test all possible techniques but chose the ones that are widely acknowledged to give high performance, i.e., those represent the state of the art. The algorithms were experimented on well-known databases so as to be compared with previous results. The intention of this work is to investigate what accuracy the state-of-the-art techniques can achieve in handwritten digit recognition and to provide a baseline for evaluation of future works.

The tested databases are CENPARMI [26], CEDAR [8], and MNIST [1]. These databases have been widely used in classifier design and evaluated in character recognition and classification/learning research. Every database is partitioned into a standard training data set and a test data set such that the results of different algorithms can be fairly compared.

The tested classifiers include an MLP, an RBF classifier, a quadratic PC, an LVQ classifier, a discriminative learning QDF (DLQDF) classifier, and two SVCs (with polynomial kernel and RBF kernel, respectively). The MLP, RBF classifier, PC, and LVQ classifier have been evaluated on special digit database and feature representation [29]. It was shown that these discriminative models give higher classification accuracies than statistical classifiers when trained on large sample size. The DLQDF is a new classifier which combines the advantages of statistical techniques and neural networks to achieve both high accuracy and resistance to out-of-class patterns (in brief, outliers) [27]. In addition to the above seven classifiers, the k-NN classifier is used as a standard for benchmarking other classifiers.

The features used in experiments are basically direction features extracted by different approaches. Possibly, the direction feature can be extracted from binary image, skeleton, contour, and gradient maps. The gradient can in turn be computed with various operators, such as Roberts, Sobel, and Kirsh. The contour direction (also called chaincode) feature and the gradient feature have been widely adopted. The extraction of chaincode feature is computationally efficient while the gradient feature can be extracted from either binary image or gray-scale image. As complementary feature, we only tested the profile structure feature [9] which incurs low additional cost to the dimensionality.

Some results of this paper have been presented previously in Ref. [28]. The rest of this paper is organized as follows. Section 2 introduces the digit databases and review previous recognition results on them. Section 3 describes the feature extraction techniques and Section 4 introduces the classification algorithms. Section 5 presents the recognition results and Section 6 makes concluding remarks.

2. Databases and previous results

The CENPARMI digit database [26] was released by CENPARMI, Concordia University. It contains 6000 digit images collected from the envelope images of USPS, scanned in 166DPI. In this database, 4000 images (400 samples per class) are specified for training and the remaining 2000 images (200 samples per class) are for testing. Some test images are shown in Fig. 1.

The CEDAR digit database is contained in the CEDAR CDROM-1, released by CEDAR, SUNY Buffalo. The images were scanned in 300 DPI. The training data set (*br*) contains 18468 digit images (the number of samples varies from class to class, as for the test data set). The test data set (*bs*) contains 2711 digit images. Since some images in the test set were poorly segmented, a subset (*goodbs*) containing 2213 well-segmented images was figured out for testing. Some images of *goodbs* data set are shown in Fig 2.

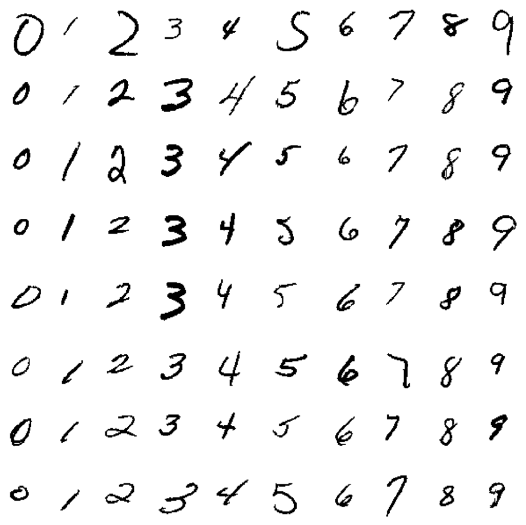

Fig. 1. Sample images of CENPARMI data.

Fig. 3. Sample images of MNIST data.

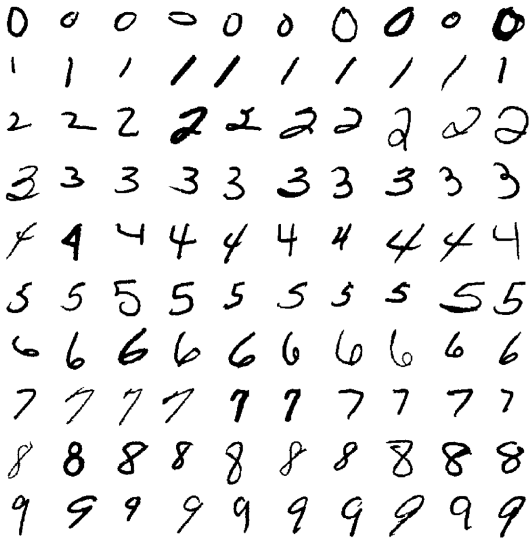

Fig. 2. Sample images of CEDAR digit data.

The MNIST (modified NIST) database [1] was extracted from the NIST special databases SD3 and SD7. SD3 and SD7 were released as the training and test data sets, respectively, for the First Census OCR Systems Conference. The two databases, however, were written by totally different sources of writers and show different styles. To overcome this problem, the training and test data sets of MNIST database were composed of samples from both SD3 and SD7. The sample binary images were normalized into 20 × 20 gray-scale images with aspect ratio preserved, and the normalized image is located in a 28 × 28 plane. Some images are shown in Fig. 3. The normalized image data are available at the homepage of LeCun [30]. The numbers of training samples and test samples are 60 000 and 10 000, respectively. Because the original sample images are not available, and some of our feature extraction algorithms are designed for binary images, we re-binarized the normalized images to generate binary images and trim to 20 × 20 size. We run classification experiments on features extracted from the binary images and from the gray-scale images.

Table 1
Previous results on CENPARMI database

Method	Correct (%)	Error (%)	Reject (%)
*Suen et al. [26]	93.05	0	6.95
*Franke et al. [32]	98.50	1.50	0
Lee [33]	97.80	2.20	0
*Gader et al. [34]	98.30	1.70	0
Liu et al. [9]	98.00	2.00	0
Hwang et al. [35]	97.90	2.10	0
Franke [36]	97.60	2.40	0
*Suen et al. [37]	**98.85**	1.15	0
Oh et al. [38]	97.85	2.15	0
Liu et al. [39]	**98.45**	1.55	0

*Multiple classifiers.

The CENPARMI database has been very widely tested and compared in character recognition community. Some good results have been reviewed in Ref. [31]. Table 1 lists some sources of high accuracies achieved on this database. The results are given in terms of the correct rate, error rate and reject rate. Except the result of Ref. [26] which makes rejection, the correct rates of other works range from 97.60% to 98.85%. Some high accuracies were given by multiple classifier systems (MCSs) [26,32,34,37], among which [37]

Table 2
Previous results on CEDAR database

Method	Correct (%)	Error (%)	Reject (%)
Lee et al. [8]	**98.87**	1.13	0
*Ha et al. [40]	99.09	0.91	0
*Suen et al. [37]	**99.77**	0.23	0
*Filatov et al. [41]	99.54	0.46	0
Cai et al. [42]	98.40	1.60	0
Oh et al. [38]	98.73	1.27	0

*Multiple classifiers.

Table 3
Previous results on MNIST database

Method	Correct (%)	Error (%)	Reject (%)
LeNet-4 [1]	98.90	1.10	0
LeNet-5 [1]	99.05	0.95	0
*Boosted LeNet-4 [1]	99.30	0.70	0
SVC-poly [43]	98.60	1.40	0
Virtual SV [43]	99.00	1.00	0
Pairwise SVC [44]	98.48	1.52	0
Dong et al. [45]	99.01	0.99	0
Mayraz et al. [48]	98.30	1.70	0
Belongie et al. [47]	99.37	0.63	0
Teow et al. [46]	**99.41**	0.59	0

*Multiple classifiers.

gave the highest accuracy 98.85% using multiple neural networks trained with very large sample size. The highest accuracy of individual classifier is 98.45% [39].

Table 2 lists selected previous results on the CEDAR database. The highest accuracies were still given by MCSs [37,40,41]. The result of Ref. [37] was produced by training with very large sample size. The highest accuracy of MCS and individual classifier are 99.77% and 98.87%, respectively.

The MNIST database was mainly used for evaluation of classification and machine learning algorithms. The normalized gray-scale images were directly used in training and classification without feature extraction so as to fairly compare the classification algorithms. Recently, Dong et al. extracted gradient features on this database [45], whereas Teow and Loe extracted features from gray-scale images [46]. Some high accuracies on the MNIST database are listed in Table 3. Except [45–47], all the results were obtained on normalized images without heuristic feature extraction. However, the LeNet-4 and LeNet-5 [1] are multilayer convolutional networks, in which the local receptive fields function as trainable feature detectors. The highest accuracy was given by Teow and Loe using trio-wise linear SVC on direction and stroke end features [46]. On the other hand, the result of Belongie et al. was achieved using flexible image matching with 10 000 templates [47].

3. Feature extraction

The features used in our experiments are basically the variants of direction feature: chaincode, gradient feature with Sobel and Kirsh operators, and peripheral direction contributivity (PDC) [49]. Direction features well characterize the within-class shape invariance and between-class difference by representing the local stroke directions. They have been widely used in character recognition and yielded high performances in the last decade. Further improvements are usually made by adding complementary structural features to the direction features. Some structural features complement to direction feature and have low dimensionalities, such as the structural and concavity features [50] and the profile structure feature [9]. In the following we will introduce the extraction approaches of various direction features and the profile structure feature.

The direction feature is extracted in three steps: image normalization, direction assignment, and feature measuring. In normalization, the character image is scaled into a standard size. The slant normalization is not performed. It was shown that the slant is better estimated and corrected in context [51]. In our experiments, the size of normalized image was set to 35×35. When the character image is not square, i.e., the x-dimension is different from the y-dimension, the aspect ratio (ratio of the short dimension to the long dimension, < 1) of the normalized image is mapped from the original aspect ratio [52]:

$$r' = \sqrt{\sin\left(\frac{\pi}{2}r\right)},$$

where r' and r are the aspect ratios of normalized image and original image, respectively. In the standard square plane of normalized image, the long dimension is filled while the short dimension is centered. This strategy of aspect ratio adaptive normalization was shown to benefit the classification performance [52].

The assignment of direction codes to pixels can be viewed as directional decomposition of image into directional sub-images. For feature measuring, each directional sub-image is partitioned into uniform zones and the intensity of each zone is accumulated as a measurement, or blurring masks are convolved with the directional sub-image to give measurements. The convolution with blurring masks is equivalent to low-pass filtering and sampling. Usually, a Gaussian mask is used:

$$h(x, y) = \frac{1}{2\pi\sigma_x^2} \exp\left(-\frac{x^2 + y^2}{2\sigma_x^2}\right).$$

The variance parameter σ_x is related to the interval between blurring masks, which can be viewed as the sampling interval and is the reciprocal of sampling frequency. An empirical formula was given in Ref. [9]:

$$\sigma_x = \frac{\sqrt{2}t_x}{\pi},$$

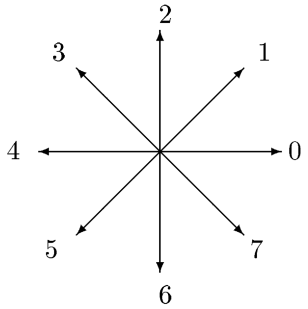

Fig. 4. Eight directions of chaincodes.

-1	0	1
-2	0	2
-1	0	1

1	2	1
0	0	0
-1	-2	-1

Fig. 5. Sobel masks for gradient.

where t_x is the interval between blurring masks in x- and y-axis. At the center (x_0, y_0) of blurring mask, the convolution gives a measurement

$$F(x_0, y_0) = \sum_x \sum_y f(x, y)h(x - x_0, y - y_0).$$

In chaincode feature extraction, the contour pixels of normalized image are assigned 8-direction codes (Fig. 4) and the contour pixels of each direction are assigned to a direction plane. The assignment of chaincodes can be accomplished in a raster scan without contour tracing, wherein a pixel of multiple connectivity can be assigned multiple chaincodes [9]. If 4-orientation feature is to be extracted, the direction planes of each pair of opposite directions are merged into one orientation plane, and blurring is performed on four planes.

For gradient feature extraction, a gradient operator is used to compute the gradient (components in two axes, or strength and direction). The gradient image is decomposed into four orientation planes or eight direction planes. The gradient operators used in feature extraction of character recognition include the Roberts operator [7,45], the Sobel operator [50,53] and the Kirsh operator [33]. The Kirsh operator is unique in that four gradient components are computed while other operators compute two components. We selected the Sobel operator and the Kirsh operator for experiments.

The Sobel operator has two masks to compute the gradient components in two axes. The masks are shown in Fig. 5 and the gradient $\mathbf{g}(x, y) = [g_x, g_y]^T$ at location (x, y) is computed by

$$\begin{aligned} g_x(x, y) = {} & f(x + 1, y - 1) + 2f(x + 1, y) \\ & + f(x + 1, y + 1) - f(x - 1, y - 1) \\ & - 2f(x - 1, y) + f(x - 1, y + 1), \end{aligned}$$

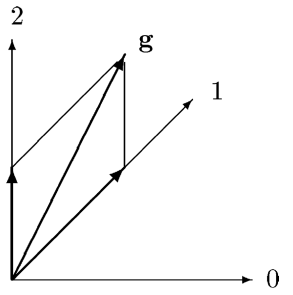

Fig. 6. Decomposition of gradient direction.

5	5	5
-3	0	-3
-3	-3	-3

-3	-3	-3
-3	0	-3
5	5	5

Horizontal edge masks

-3	5	5
-3	0	5
-3	-3	-3

-3	-3	-3
5	0	-3
5	5	-3

Righ-diagonal edge masks

Fig. 7. Kirsh masks for horizontal and diagonal edges.

$$\begin{aligned} g_y(x, y) = {} & f(x - 1, y + 1) + 2f(x, y + 1) \\ & + f(x + 1, y + 1) - f(x - 1, y - 1) \\ & - 2f(x, y - 1) + f(x + 1, y - 1). \end{aligned}$$

The gradient strength and direction can be computed from the vector $[g_x, g_y]^T$. For character feature extraction, the range of gradient direction is partitioned into a number (say, 8 or 16) of regions and each region corresponds to a directional sub-image. Each pixel is assigned to a direction region and the gradient strength contributes to the intensity of the corresponding sub-image. This strategy has been adopted in previous works [50,53]. In our experiments, however, we adopt another strategy, which decomposes each gradient vector into components in standard chaincode directions. This strategy was previously used in on-line character recognition [54]. We decompose the gradient vectors into eight chaincode directions. If a gradient direction lies between two standard directions, it is decomposed into two components in the two standard directions, as shown in Fig. 6.

The Kirsh operator has eight masks to compute the gradient components in eight directions. Fig. 7 shows four masks, and the other four are symmetric with these. For 4-orientation feature extraction, the edge strengths in

A_0	A_1	A_2
A_7	(i,j)	A_3
A_6	A_5	A_4

Fig. 8. Neighborhood configuration of pixel (i, j).

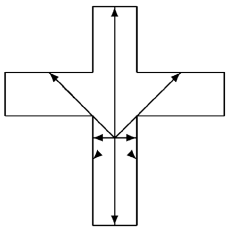

Fig. 9. Calculation of direction contributivity.

four orientations (horizontal, vertical, left-diagonal and right-diagonal) are computed by

$$G(i,j)_H = \max(|5S_0 - 3T_0|, |5S_4 - 3T_4|),$$

$$G(i,j)_V = \max(|5S_2 - 3T_2|, |5S_6 - 3T_6|),$$

$$G(i,j)_L = \max(|5S_3 - 3T_3|, |5S_3 - 3T_3|),$$

$$G(i,j)_R = \max(|5S_1 - 3T_1|, |5S_1 - 3T_1|),$$

where

$$S_k = A_k + A_{k+1} + A_{k+2},$$

$$T_k = A_{k+3} + A_{k+4} + A_{k+5} + A_{k+6} + A_{k+7}$$

and the neighboring pixels are denoted as in Fig. 8.

The advantage of Kirsh operator lies in that it directly gives the strengths of eight directions. However, since the masks are not orthogonal, i.e., the mask of a specific direction does not give zero gradient strength to the edge in perpendicular direction, the directional elements are not well separated.

In PDC feature extraction, the edge pixels of character image are decomposed into hierarchical profiles and the directionality of edge pixels is determined by run-lengths in stroke area [49]. As shown in Fig. 9, to compute the directionality of a stroke pixel (x, y), the distances from this point to the nearest edge in eight directions are used to compute the four direction components d_m, $m = 0, 1, 2, 3$,

$$d_m = \frac{l_m + l_{m+4}}{\sqrt{\sum_{i=0}^{3} (l_i + l_{i+4})^2}}.$$

The feature measurements are extracted from the direction components of edge pixels of hierarchical profiles. The profiles are composed of the edge pixels crossed by rays starting from four boundaries of image, two hierarchies from each boundary. The outmost pixels compose the first hierarchy and the pixels next to the outmost constitute the second hierarchy. Each of the eight profiles is partitioned into uniform intervals and the four direction components in each interval are counted.

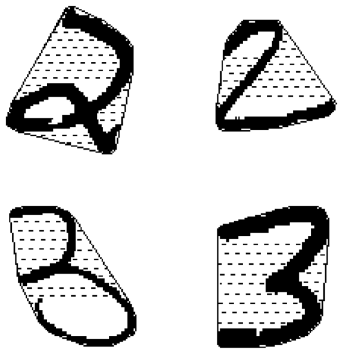

Fig. 10. Extraction of concavity feature.

Now let us introduce the profile structure features which are used as complements to the direction feature. We use the horizontal crossing counts and peripheral concavity measurements as structural features [9]. The horizontal crossing counts as well as the left and right concavity measurements are sampled by 1D Gaussian blurring. The concavity is represented by the distance between the outmost edge and the convex hull, as shown in Fig. 10. This feature is fairly stable against the image skew.

Unlike other features are designed specially for binary images, the gradient feature can be extracted from either binary or gray-scale images. We use the Sobel operators to extract gradient feature from both binary and gray-scale images. In the character databases we are to test, the samples of MNIST are normalized gray-scale images. For CENPARMI and CEDAR databases, we generate binary image as well as gray-scale image in normalization. The gray-scale image is generated by interpolation, wherein a stroke pixel in the original binary image gives continuous gray levels to the corresponding pixels in the normalized image. To differentiate from original gray-scale images, we refer to the gray-scale images normalized from binary images as *pseudo-gray* images. For the MNIST database, we test gradient features from pseudo-gray images as well as the original 20×20 gray-scale images.

In summary, we use ten feature vectors in character classification. Except those extracted from the original gray-scale images of MNIST, all the features are extracted from 35×35 normalized (binary or pseudo-gray) images. The feature vectors are listed in the following:

- Feature vector **blr**: 4-orientation chaincode feature, 100D. On each of 4-orientation planes (merged from

8-chaincode sub-images), 5 × 5 Gaussian masks are uniformly imposed to compute 25 measurements.

- Feature vector **mul**: 4-orientation chaincode feature, plus 11 crossing counts, and 22 concavity measurements, 133D. The crossing counts and concavity measurements are sampled by 1D Gaussian blurring.
- Feature vector **e-blr**: 8-direction chaincode feature, 200D. On each of 8-direction planes, 5 × 5 Gaussian masks are uniformly imposed to compute 25 measurements.
- Feature vector **e-mul**: 8-direction chaincode feature, plus 11 crossing counts, and 22 concavity measurements, 233D.
- Feature vector **grd**: 4-orientation gradient feature, 100D. On each of 4-orientation gradient maps (merged from 8-direction gradient sub-images), 5 × 5 Gaussian masks are uniformly imposed to compute 25 measurements.
- Feature vector **e-grd**: 8-direction gradient feature, 200D. On each of 8-direction gradient maps, 5 × 5 Gaussian masks are uniformly imposed to compute 25 measurements.
- Feature vector **grg**: 4-orientation gradient feature from (pseudo-) gray-scale image, 100D.
- Feature vector **e-grg**: 8-direction gradient feature from (pseudo-) gray-scale image, 200D.
- Feature vector **Kir**: 4-orientation Kirsh feature, 100D. On each of 4-orientation Kirsh gradient maps, 5 × 5 Gaussian masks are uniformly imposed to compute 25 measurements.
- Feature vector **pdc**: 4-orientation PDC feature, 224D. Each of 8 profiles (4 boundaries, 2 hierarchies) is partitioned into 7 intervals, and the direction components of each interval are counted.

Before learning/classification, all the measurements in the feature vectors are transformed by variable transformation $y = x^{0.5}$ [55].

4. Classification algorithms

4.1. Neural classifiers

We use in experiments three neural classifiers, namely, an MLP with one hidden layer, an RBF classifier, and a quadratic PC on linear subspace.

For M-class classification, the MLP has M output units. On an input pattern $\mathbf{x} = (x_1, \ldots, x_d)^{\mathrm{T}}$, the output of class k, $k = 1, 2, \ldots, M$, is computed by

$$y_k(\mathbf{x}) = s\left[\sum_{j=1}^{N_h} w_{kj} s(\mathbf{v}_j^{\mathrm{T}}\mathbf{x} + v_{j0}) + w_{k0}\right] = s\left[\sum_{j=1}^{N_h} w_{kj} h_j + w_{k0}\right], \quad (1)$$

where N_h denotes the number of hidden units, w_{kj} and v_{ji} denote the connecting weights of output layer and hidden layer, respectively. $s(a)$ is a sigmoid activation function

$$s(a) = \frac{1}{1 + \mathrm{e}^{-a}}.$$

The error back-propagation (BP) algorithm [18] is used to learn the connecting weights by minimizing the mean-square error (MSE) over a set of N_x examples:

$$E = \frac{1}{N_x}\left\{\sum_{n=1}^{N_x}\sum_{k=1}^{M}[y_k(\mathbf{x}^n, \mathbf{w}) - t_k^n]^2 + \lambda \sum_{w \in W} w^2\right\}, \quad (2)$$

where λ is a coefficient to control the decay of connecting weights (excluding the biases); t_k^n is the desired value of class k, with value 1 for genuine class and 0 otherwise. The connecting weights are updated by stochastic gradient descent [56] and a momentum term is added to improve the convergence.

The RBF classifier has one hidden layer with each hidden unit being a Gaussian kernel:

$$h_j(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{x} - \mu_j\|^2}{2\sigma_j^2}\right). \quad (3)$$

Each output is a linear combination of the Gaussian kernels with sigmoid non-linearity to approximate the class membership:

$$y_k(\mathbf{x}) = s\left[\sum_{j=1}^{N_h} w_{kj} h_j(\mathbf{x}) + w_{k0}\right].$$

For parameter learning of RBF classifier, the kernel parameters are initialized from unsupervised clustering of the sample data and then the connecting weights and kernel center vectors are updated by gradient descent to minimize the MSE criterion (2). The supervised updating of center vectors can largely improve the classification accuracy [19,20].

The polynomial classifier (PC) is a single-layer network with the polynomials of the input measurements as inputs [21,22]. We use a PC with binomial inputs of the principal components on the linear subspace of feature space. Denote the principal components in the m-dimensional ($m < d$) subspace as $\mathbf{z}$, the output corresponding to class k is computed by

$$y_k(\mathbf{x}) = s\left[\sum_{i=1}^{m}\sum_{j=i}^{m} w_{kij}^{(2)} z_i(\mathbf{x}) z_j(\mathbf{x}) + \sum_{i=1}^{m} w_{ki}^{(1)} z_i(\mathbf{x}) + w_{k0}\right], \quad (4)$$

where $z_j(\mathbf{x})$ is the projection of $\mathbf{x}$ onto the jth principal axis of the subspace.

The connecting weights are updated by gradient descent to minimize the mean-square error (2). In implementation, the projected features are re-scaled as

$$z_j(\mathbf{x}) = \frac{(\mathbf{x} - \mu_x)^{\mathrm{T}}\phi_j}{\sqrt{\lambda_1}}, \quad j = 1, 2, \ldots, m, \quad (5)$$

where μ_x and ϕ_j, $j=1,2,\ldots,m$, denote the mean vector and the eigenvectors of the sample space, respectively; λ_1 is the largest eigenvalue corresponding to the eigenvector ϕ_1.

4.2. LVQ and DLQDF classifiers

LVQ classifiers take the 1-NN rule for classification but the prototypes are designed in discriminative supervised learning. As variations of Kohonen's LVQ, some prototype learning algorithms update the prototypes with aim to optimize an objective function. It was shown in a comparative study that this class of algorithms outperform Kohonen's LVQ [23]. We adopt the prototype learning algorithm with the minimum classification error (MCE) criterion [57,58]. The learning algorithm is outlined as follows, and more details can be found in Ref. [23].

For M classes $\{C_k \mid k=1,2,\ldots,M\}$, each class has n_k prototypes $\{\mathbf{m}_{kj} \mid j=1,2,\ldots,n_k\}$. A training data set $\{(\mathbf{x}^n, c^n) \mid n=1,2,\ldots,N\}$ (where c^n denotes the class label of pattern $\mathbf{x}^n$) is available to learn the prototypes via minimizing a loss function

$$L_0 = \frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{M} l_k(\mathbf{x}^n) I(\mathbf{x}^n \in C_k),$$

where $I(\cdot)$ is an indicator function which takes value 1 when the condition in parentheses is satisfied, otherwise takes value 0. The loss associated with a pattern is computed by

$$l_k(\mathbf{x}) = l_k(\mu_k) = \frac{1}{1+\mathrm{e}^{-\xi\mu_k}}$$

with

$$\mu_k(\mathbf{x}) = d(\mathbf{x}, \mathbf{m}_{ki}) - d(\mathbf{x}, \mathbf{m}_{rj}),$$

where $d(\mathbf{x}, \mathbf{m}_{ki})$ denotes the minimum distance to the prototypes of genuine class of training pattern, and $d(\mathbf{x}, \mathbf{m}_{rj})$ denotes the minimum distance to the closest rival class. The distance metric is the square Euclidean distance $d(\mathbf{x}, \mathbf{m}) = \|\mathbf{x}-\mathbf{m}\|^2$.

The loss function L_0 is minimized by stochastic approximation. The prototypes are initialized by k-means clustering. Then the training samples are fed to the classifier repeatedly, each time the prototypes are updated according to the partial derivatives:

$$\mathbf{m}_{ki} = \mathbf{m}_{ki} + 2\alpha(t)[\xi l_k(1-l_k)(\mathbf{x}-\mathbf{m}_{ki}) + \lambda(\mathbf{x}-\mathbf{m}_{ki})],$$

$$\mathbf{m}_{rj} = \mathbf{m}_{rj} - 2\alpha(t)\xi l_k(1-l_k)(\mathbf{x}-\mathbf{m}_{rj}), \tag{6}$$

where $\alpha(t)$ is a learning rate, which is sufficiently small and decreases with time. A regularization term is added to the closest genuine prototype so as to constrain the deviation of prototypes from samples.

The DLQDF was recently proposed by the authors to overcome the inadequate classification accuracy of MQDF2 while preserving the superior resistance to out-of-class patterns. It has the same form as MQDF2:

$$g_2(\mathbf{x}, \omega_i) = \sum_{j=1}^{k}\frac{1}{\lambda_{ij}}[(\mathbf{x}-\mu_i)^{\mathrm{T}}\phi_{ij}]^2 + \frac{1}{\delta_i}\left\{\|x-\mu_i\|^2 - \sum_{j=1}^{k}[(\mathbf{x}-\mu_i)^{\mathrm{T}}\phi_{ij}]^2\right\} + \sum_{j=1}^{k}\log\lambda_{ij} + (d-k)\log\delta_i, \tag{7}$$

where μ_i denotes the mean vector of class ω_i, λ_{ij} and ϕ_{ij}, $j=1,\ldots,d$, denote the eigenvalues and eigenvectors of the covariance matrix of class ω_i. The eigenvalues are sorted in decreasing order and the eigenvectors are sorted accordingly. k denotes the number of principal components, and the minor eigenvalues are replaced with a constant δ_i. Note the principal subspace of MQDF2 is class specific.

The DLQDF inherits initial parameters from MQDF2 and then all the parameters are optimized on training samples under the MCE criterion. Initially, the parameter δ_i of MQDF2 is set to the average of minor eigenvalues. Like for LVQ, the MCE criterion is regularized with the quadratic distance of the genuine class: [2]

$$L_1 = \frac{1}{N}\sum_{n=1}^{N}[l_c(\mathbf{x}^n) + \beta g_2(\mathbf{x}^n, \omega_c)], \tag{8}$$

where ω_c denotes the genuine class of training pattern, β is the regularization coefficient. In parameter updating by stochastic gradient descent, the Gram–Schmidt orthonormalization (GMO) procedure is embedded to maintain the ortho-normality of eigenvectors, under which the discriminant function adheres to the assumption of Gaussian density as MQDF2 does.

4.3. Support vector classifiers

Generally, an SVM solves a binary (two-class) classification problem, and multi-class classification is accomplished by combining multiple binary SVMs. An M-class problem can be decomposed into M binary problems with each separating one class from the others, or into $\binom{M}{2}$ binary problems with each discriminating between a pair of classes. The former scheme is adopted in our experiments.

On a pattern $\mathbf{x}$, the discriminat function of a binary SVM is computed by

$$f(\mathbf{x}) = \sum_{i=1}^{\ell} y_i\alpha_i k(\mathbf{x}, \mathbf{x}_i) + b, \tag{9}$$

[2] The learning algorithm of DLQDF has been changed slightly compared to the version presented in Ref. [27].

where ℓ is the number of learning patterns, y_i is the target value of learning pattern $\mathbf{x}_i$ (+1 for the first class and −1 for the second class), b is a bias, and $k(\mathbf{x}, \mathbf{x}_i)$ is a kernel function which implicitly defines an expanded feature space:

$$k(\mathbf{x}, \mathbf{x}_i) = \Phi(\mathbf{x}) \cdot \Phi(\mathbf{x}_i),$$

where $\Phi(\mathbf{x})$ is the feature vector in the expanded feature space and may have infinite dimensionality.

Two types of kernels, polynomial kernel and RBF kernel are frequently used. They are computed by

$$k(\mathbf{x}, \mathbf{x}_i, p) = (1 + \mathbf{x} \cdot \mathbf{x}_i)^p$$

and

$$k(\mathbf{x}, \mathbf{x}_i, \sigma^2) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{2\sigma^2}\right),$$

respectively. In our implementation, the pattern vectors are appropriately scaled for the polynomial kernel, with the scaling factor estimated from the self-inner products of sample vectors. While for the RBF kernel, the value of σ^2 is estimated from the variance of sample vectors.

The discriminant function (9) can be viewed as a generalized linear discriminant function with weight vector

$$\mathbf{w} = \sum_{i=1}^{\ell} y_i \alpha_i \cdot \Phi(\mathbf{x}_i).$$

The coefficients α_i, $i = 1, \ldots, \ell$, are determined on the learning patterns by solving the following optimization problem:

$$\begin{aligned} &\text{minimize} && \tau(\mathbf{w}) = \tfrac{1}{2}\|\mathbf{w}\|^2 \\ &\text{subject to} && y_i f(\mathbf{x}_i) \geqslant 1 - \xi_i, \quad \xi_i \geqslant 0, \quad i = 1, \ldots, \ell. \end{aligned}$$

This is a quadratic programming problem and can be converted into the following dual problem:

$$\begin{aligned} &\text{maximize} && W(\alpha) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{\ell} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) \\ &\text{subject to} && 0 \leqslant \alpha_i \leqslant C, \ i = 1, \ldots, \ell, \ \text{and} \sum_{i=1}^{\ell} \alpha_i y_i = 0, \end{aligned} \tag{10}$$

where C is a parameter to control the tolerance of classification errors in learning.

The solution of the quadratic problem (10) is not trivial on large sample size. Some algorithms have been proposed specially to solve the quadratic programming problem for SVM learning, among which are the sequential minimal optimization (SMO) of Platt [59] and the successive over-relaxation (SOR) of Mangasarian [60]. We tested both of them and found that SOR is faster than SMO in most cases. On a personal computer of 1.5 GHz CPU, the learning of a two-class SVM on MNIST image data ($\ell = 60\,000$, $d = 400$) costs less than three hours. On other feature vector representations of MNIST data, the two-class learning costs less than one hour.

After optimization, only a portion of the learning patterns have non-zero coefficient α_i. These patterns are called support vectors (SVs). In computing the discriminant function (9), the patterns of zero coefficient are not used. For multi-class classification, each class has an SVM to give a discriminant function $f_k(\mathbf{x})$, $k = 1, \ldots, M$, and the input pattern is assigned to the class of maximum discriminant function. Because the binary SVMs are learned on the same sample set (the target values vary with the target class), some SVs are shared by multiple classes. In classification, the kernel function of a shared SV is computed only once. We hence record the total number of distinct SVs and use it as an index of computational complexity in recognition.

5. Experimental results

5.1. Results on binary images

Presented in this section are the recognition accuracies obtained using (binary and pseudo-gray) images normalized from binary images. We combined the ten feature vectors and eight classifiers (k-NN, MLP, RBF, PC, LVQ, DLQDF, SVC-poly, and SVC-rbf) to give 80 classification accuracies to each of the three databases. On specific feature type and a database, the parameters of each classifier are trained on the training data set and then the trained classifier is used to classify the test data set. The artificial parameters in classifier training are set as follows.

Some training parameters such as the weight decay of neural classifiers are set to be training data independent. The weight decay was set to 0.05 for MLP, 0.02 for RBF classifier, and 0.1 for PC. The regularization coefficient of LVQ was set to $0.05/var$, where var is the estimated variance on training data partitioned by clustering. The regularization coefficient of DLQDF was set to $0.1/var$, where var is the class variance estimated on training data. The parameters of SVC-poly were set to $p = 5$ and $C = 1$, while the parameters of SVC-rbf were set to $\sigma^2 = 0.3var$ (var is the estimated variance of training data) and $C = 10$.

The performances of MLP and LVQ classifier are dependent on the random initialization of connecting weights or cluster centers. We trained each MLP or LVQ classifier twice with two different initialized parameter sets and retain the trained classifier which gives higher accuracy to the test data set.

To achieve good generalization performance, the size (parameter complexity) of classifiers is dependent on the training sample size. For each database, we did not attempt to set multiple sizes for a classifier so as to reach the highest accuracy. Rather, we set empirically an appropriate size for each classifier on a specific database. The classifier size of MLP and RBF classifier is indicated by the number of hidden units N_h, that of PC and DLQDF by the number of principal components m or k, that of LVQ classifier by the number of prototypes n_p for each class. The classifier sizes for the

Table 4
Parameter complexities of non-SV classifiers

Database	MLP (N_h)	RBF (N_h)	PC (m)	LVQ (n_p)	LQDF (k)
CENPARMI	40	50	60	4	20
CEDAR	80	80	60	8	30
MNIST	300	300	70	30	40

Table 5
Error rates (%) on the test data of CENPARMI database

	blr	mul	e-blr	e-mul	grd	e-grd	grg	e-grg	Kir	pdc	Aver	Rank
k-NN	2.35	2.20	2.10	1.85	2.35	1.80	2.65	2.35	5.65	3.00	2.630	8
MLP	2.15	2.05	2.05	1.75	2.10	1.40	2.05	1.70	2.90	2.00	2.015	7
RBF	1.75	1.65	1.65	1.40	1.80	1.25	1.55	1.55	3.25	1.95	1.780	5
PC	1.55	1.40	1.25	1.20	1.65	1.20	1.25	1.20	2.30	2.05	1.505	3
LVQ	2.15	1.50	1.80	1.50	2.60	1.60	1.85	1.40	3.50	2.10	2.00	6
LQDF	1.50	1.50	1.20	1.10	1.65	1.05	1.45	**0.95**	2.35	1.95	1.470	2
SVC-poly	1.55	1.40	1.45	1.45	1.70	1.45	1.35	1.20	1.85	1.70	1.510	4
SVC-rbf	1.10	1.00	1.20	**0.95**	1.30	1.15	1.10	**0.95**	2.05	1.55	1.235	1
Average	1.763	1.587	1.587	1.40	1.894	1.363	1.656	1.412	2.981	2.037		
Rank	7	4	4	2	8	1	6	3	10	9		

Table 6
Error rates (%) on the test data of CEDAR database

	blr	mul	e-blr	e-mul	grd	e-grd	grg	e-grg	Kir	pdc	Aver	Rank
k-NN	1.17	1.08	0.90	0.95	1.27	1.13	1.22	0.99	2.85	1.54	1.310	8
MLP	1.08	0.95	0.95	0.81	1.13	0.99	1.17	1.04	1.36	1.04	1.052	5
RBF	1.04	0.90	0.86	0.77	1.22	0.95	1.13	0.99	1.76	1.17	1.079	6
PC	0.81	**0.68**	0.81	0.77	0.99	0.86	1.08	0.99	1.68	1.08	0.975	4
LVQ	1.13	**0.68**	0.86	0.77	1.17	1.08	1.27	0.90	2.12	1.08	1.106	7
LQDF	0.81	0.86	0.90	0.72	0.95	0.72	0.86	0.86	1.13	1.13	0.894	3
SVC-poly	0.63	0.77	0.81	0.81	0.95	0.68	0.99	0.81	1.27	0.99	0.871	2
SVC-rbf	0.68	**0.54**	0.68	**0.54**	0.72	0.72	0.86	0.68	1.40	0.99	0.781	1
Average	0.919	0.808	0.846	0.768	1.050	0.891	1.073	0.907	1.696	1.127		
Rank	6	2	3	1	7	4	8	5	10	9		

non-SV classifiers on three databases are given in Table 4. For SVCs, the number of distinct SVs determines the classifier complexity. However, this number is not pre-specified. Rather, it depends on the training parameters and the training data. We will show later that this number is very large so that the SVC is much more expensive in storage and computation than other classifiers.

The k-NN classifier uses all the training patterns as the prototypes. The classification accuracy is influenced by the number of nearest neighbors k. We tried multiple values $k = 1, 3, 5, 7, 9$, and the tie of class scores was solved by the 1-NN rule. As result, the highest accuracy was mostly given by $k = 3$ or 5.

The error rates (of forced classification on test data set without rejection) on three databases are listed in Tables 5–7, respectively. For ordering the performance of different features and classifiers, the average error rate of each feature vector and each classifier is calculated. For each database, the lowest error rate of SVCs and that of non-SV classifiers are highlighted.

We can see from the results that on each database, either the average error rates or specific error rates are very low due to the discriminatory power of features and the regression capabilities of classifiers. Comparing the highest accuracies (corresponding to the lowest error rates) to those of previous results in Tables 1–3, our results are very competitive. In specific, the highest accuracy on the CENPARMI database (99.05% by either non-SV classifiers or SVCs) is apparently higher than the highest previous accuracy given by MCSs (98.85%). The accuracies on the CEDAR database

Table 7
Error rates (%) on the test data of MNIST database

	blr	mul	e-blr	e-mul	grd	e-grd	grg	e-grg	Kir	pdc	Aver	Rank
k-NN	1.58	1.37	1.35	1.26	1.45	1.26	1.47	1.35	3.12	1.76	1.597	8
MLP	1.14	1.08	0.98	0.88	1.03	0.89	1.03	0.92	1.46	1.24	1.065	6
RBF	0.97	0.89	0.87	0.81	1.00	0.90	0.95	0.85	1.38	1.06	0.968	4
PC	0.89	0.90	0.72	0.75	0.97	0.85	0.90	**0.69**	1.29	1.23	0.919	3
LVQ	1.34	1.22	0.94	1.02	1.27	1.05	1.31	1.09	2.12	1.49	1.285	7
LQDF	1.07	0.93	0.86	0.90	1.14	0.87	0.87	0.85	1.22	1.16	0.987	5
SVC-poly	0.91	0.86	0.83	0.76	1.05	0.79	0.90	0.70	1.23	0.97	0.90	2
SVC-rbf	0.84	0.87	0.72	0.74	0.90	0.75	0.76	**0.61**	1.16	0.88	0.823	1
Average	1.093	1.015	0.909	0.890	1.101	0.920	1.024	0.883	1.623	1.224		
Rank	7	5	3	2	8	4	6	1	10	9		

(99.32% by non-SV classifiers and 99.46% by SVCs) are much higher than the previous accuracies by individual classifiers. Actually, it is not fair to compare our results with those of Suen et al. [37] on CENPARMI and CEDAR test data sets because their classifiers were trained on 450 000 samples. On the MNIST database, the highest accuracies, 99.31% by non-SV classifiers and 99.39% by SVCs, are very close to the best previous results obtained on fine gray-scale images (the pseudo-gray images are coarse because they are normalized from 20×20 binary images).

To compare the feature vectors and classifiers, we can see that the order of feature vectors/classifiers varies with the specific classifier/feature vector. However, a good feature vector/classifier should outperform the others in majority of cases. We herein order the feature vectors/classifiers according to the average accuracy on various classifiers/feature vectors and it is approximately true that the order of average accuracy is consistent with the tendency of accuracies on specific classifier/feature vector.

In comparison of the feature vectors, it is evident that 8-direction features outperform 4-orientation features and the profile structure features are effective to improve the accuracy of direction feature. Comparing the 4-orientation features, it is evident that the chaincode feature and gradient features are superior to the Kirsh feature and PDC feature.

It is worthy further comparing the chaincode feature, the gradient feature from binary image (in brief, binary gradient feature) and the gradient feature from pseudo-gray images (in brief, pseudo-gray gradient feature). We can see that on the three databases, the binary gradient feature does not show advantage over the chaincode feature except that the 8-direction *e-grd* wins on CENPARMI data. This is because both of them are extracted from binary images and represent similar characteristics of the same signal. Comparing the binary and pseudo-gray gradient features, we can see the superiority of pseudo-gray gradient feature on CENPARMI and MNIST but the advantage is not shown on CEDAR database. We think this result is related to the fineness of images that features are extracted from. The resolution of character images in CEDAR database is high (300DPI), so normalization to either binary image or pseudo-gray image does not make significant difference in image fineness. The image resolutions of CENPARMI and MNIST are low such that the original image size is even smaller than the specified normalized image size (35×35), so normalization to pseudo-gray image give finer appearance than normalization to binary image, and consequently the pseudo-gray gradient feature yields higher recognition accuracies. The difference of accuracy between binary and pseudo-gray gradient features is prominent on MNIST database because the original binary images have very low resolution (20×20).

Unlike that the order of feature vectors varies largely with the data set to classify, the order of classifiers is fairly stable. In average sense, the SVC with RBF kernel (SVC-rbf) performs best unanimously while the *k*-NN classifier is consistently outperformed by the other classifiers. The SVC with polynomial kernel (SVC-poly) does not show advantage over the conventional neural PC and the DLQDF. The four classifiers, SVC-rbf, SVC-poly, PC, and LQDF are mostly among the top four ranks. Among other classifiers, we can see that in majority of cases, the RBF classifier outperforms the MLP and LVQ classifier. We believe from our experience that the artificial parameters in classifier training were set reasonably, though not optimally. Hence, the results of these experiments reasonably reflect the generalization performances of classifiers in handwritten character recognition.

From the results, it is not obvious what specific combination of feature vector and classifier gives the best performance. The highest classification accuracy is generally given by the combination of the features and classifiers that are good in average sense. For example, on CENPARMI database, the globally highest accuracies are given by feature vectors *e-mul* and *e-grg*, which are also ranked at the top in average sense. Regarding the classifiers that give the globally highest accuracies, the SVC-rbf, PC and DLQDF are also ranked at the top of classifiers in average sense. To figure out what feature vector and what classifier perform

Table 8
Error rates (%) on fine image features of MNIST database

MNIST	img	pca	grg	e-grg
k-NN	3.66	3.01	1.26	0.97
MLP	1.91	1.84	0.84	0.60
RBF	2.53	2.21	0.92	0.69
PC	1.64	N/A	0.83	**0.58**
LVQ	2.79	2.73	1.23	1.05
LQDF	1.97	N/A	0.94	0.71
SVC-poly	1.69	1.43	0.76	0.55
SVC-rbf	1.41	1.24	0.67	**0.42**

well universally, we pick up the ones that are ranked at the top for all the three databases. The feature vectors that are ranked top four in three cases are *e-blr*, *e-mul* and *e-grg*, while the classifiers that are ranked top four in three cases are SVC-rbf, SVC-poly, and PC.

5.2. Results on fine gray-scale images

We also experimented on features extracted from the original 20 × 20 gray-scale images of MNIST database. The images are finer than those normalized from 20 × 20 binary images because the binarization has lost some fineness. The 20 × 20 gray-scale image itself forms a feature vector of 400D, denoted by *img*. The 400D vector is reduced by principal component analysis (PCA) to give 80 principal components, denoted by *pca*. The Sobel gradient features are extracted from the 20 × 20 gray-scale images, in 4 orientations or 8 directions. The dimensionalities of 4-orientation and 8-direction gradient features are 100D and 200D, respectively. From each orientation or direction plane, 5 × 5 blurring masks are imposed uniformly to calculate 25 measurements. The 4-orientation and 8-direction gradient features are denoted by *grg* and *e-grg*, respectively.

The error rates on MNIST test data by combining four feature vectors and eight classifiers are shown in Table 8. We can see that by classification directly on the image data, the accuracy is not very high. Via dimensionality reduction by PCA, the accuracy is improved slightly. While on gradient feature extracted from fine gray-scale image, the accuracy is much higher than that from binary images and pseudo-gray images. The highest accuracies by SVCs and non-SV classifiers, 99.58% and 99.42%, are higher than the best of previous works (99.41%).

It is worth noting that the accuracies of the gradient feature extracted from fine gray-scale images are consistently higher than those of the features extracted from binary and pseudo-gray images in Table 7. This justifies the importance of the fineness of images in feature extraction. The gray-scale images transformed by LeCun et al. keep the fine appearance of character shapes even though the resolution is low. Via binarization and re-normalization to pseudo-gray images, we cannot recover the fineness of images.

Table 9
Classification times (ms) on fine image features of MNIST database

MNIST	img	pca	grg	e-grg
k-NN	98.66	14.85	16.67	37.75
MLP	0.80	0.17	0.24	0.44
RBF	1.03	0.23	0.34	0.58
PC	0.87	N/A	0.71	0.76
LVQ	1.43	0.13	0.31	0.78
LQDF	0.92	N/A	0.26	0.49
SVC-poly	16.1	6.41	3.88	5.90
SVC-rbf	62.8	17.7	12.5	21.9
Number of distinct SVs				
SVC-poly	12 765	12 689	4744	4521
SVC-rbf	20 885	17 154	7424	8030

In evaluating the performance of classifiers, the memory space and computation speed are also important factors. Table 9 gives the average CPU times in classifying a test pattern on MNIST fine image features, as well as the numbers of distinct SVs of SVCs. We can see that the *k*-NN classifier is the most expensive in storage and computation because all the training patterns are stored and compared with the input pattern. For SVCs, the number of SVs is very large, and the classification by SVCs costs much more CPU time than that by non-SV classifiers except *k*-NN. Particularly, the CPU time of SVC-rbf is comparable to that of *k*-NN classifier, and therefore is prohibitive for real-time applications. Also, compared to non-SV classifiers especially the PC, the SVC-poly shows poor performance-to-cost ratio.

6. Conclusion

The recognition results reported in this paper show what accuracy the current feature extraction and classification techniques can achieve in handwritten digit recognition on well-known databases. We combined ten feature vectors and eight classifiers to give 80 accuracies to the test data set of each database. The highest accuracy on each database is by far the best when compared to the previously reported results by individual classifiers. Even when compared to previous multiple classifier systems, our results are competitive. We are sure that the algorithms used in our experiments are not the best choices and are not implemented optimally. To evaluate other, possibly new algorithms, the reported results can serve the baseline.

In our experiments, the SVC with RBF kernel (SVC-rbf) gives the highest accuracy but is extremely expensive in memory space and computation. So, the target of future classifier design is to match with the accuracy of SVC-rbf at low complexity, via extracting more discriminatory features, devising new classification/learning schemes, combining multiple classifiers, etc. The reported results

show that the PC and DLQDF are good choices for applications since they give competitive accuracies at low complexity. The high performance of PC and DLQDF in this case only indicates that they are suitable for the problem of character recognition, while the SVCs are superior for many different recognition problems. Since no classifier is superior to others in all problems, for a specific application, we must compare the performance of a variety of classifiers and select or design classifiers (density models or discriminative models with appropriate constraints) aiming to solve the problem at hand.

Feature extraction is primarily important to the performance of character recognition. A powerful classifier such as the SVC may yield much different accuracies based on different features of patterns, as shown in our results. On appropriately extracted features, even a simple, low-complexity classifier can give very high recognition accuracy. Of course, on the same feature representation, different classifiers still give varying accuracies. The best final result is hence obtained by combining a good classifier and a good feature representation. We have shown in our results that some features and some classifiers are generally good, even though the order of performance varies with the data set. The features that universally perform well are the chaincode feature and gradient feature, and adding complementary structural features to them may further improve the accuracy. While the chaincode feature is extracted from binary images, the gradient feature is applicable to either binary or gray-scale images. Particularly, gradient feature extraction from gray-scale images is preferable for low resolution images.

References

[1] Y. LeCun, et al., Comparison of learning algorithms for handwritten digit recognition, in: F. Fogelman-Soulié, P. Gallinari (Eds.), Proceedings of the International Conference on Artificial Neural Networks, Nanterre, France, 1995, pp. 53–60.

[2] O.D. Trier, A.K. Jain, T. Taxt, Feature extraction methods for character recognition—a survey, Pattern Recognition 29 (4) (1996) 641–662.

[3] M. Yasuda, H. Fujisawa, An improvement of correlation method for character recognition, Trans. IEICE Japan J62-D (3) (1979) 217–224.

[4] Y. Yamashita, K. Higuchi, Y. Yamada, Y. Haga, Classification of handprinted Kanji characters by the structured segment matching method, Pattern Recognition Lett. 1 (1983) 475–479.

[5] F. Kimura, et al., Evaluation and synthesis of feature vectors for handwritten numeral recognition, IEICE Trans. Inform. Systems E79-D (5) (1996) 436–442.

[6] S. Tsuruoka, et al., Handwritten Kanji and hiragana character recognition using weighted direction index histogram method, Trans. IEICE Japan J70-D (7) (1987) 1390–1397.

[7] M. Shi, Y. Fujisawa, T. Wakabayashi, F. Kimura, Handwritten numeral recognition using gradient and curvature of gray scale image, Pattern Recognition 35 (10) (2002) 2051–2059.

[8] D.-S. Lee, S.N. Srihari, Handprinted digit recognition: a comparison of algorithms, in: Proceedings of the Third International Workshop on Frontiers of Handwriting Recognition, Buffalo, NY, 1993, pp. 153–164.

[9] C.-L. Liu, Y.-J. Liu, R.-W. Dai, Preprocessing and statistical/structural feature extraction for handwritten numeral recognition, in: A.C. Downton, S. Impedovo (Eds.), Progress of Handwriting Recognition, World Scientific, Singapore, 1997, pp. 161–168.

[10] L. Heutte, T. Paquet, J.V. Moreau, Y. Lecourtier, C. Olivier, A structural/statistical feature based vector for handwritten character recognition, Pattern Recognition Lett. 19 (7) (1998) 629–641.

[11] K. Fukunaga, Introduction to Statistical Pattern Recognition, 2nd Edition, Academic Press, New York, 1990.

[12] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, 2nd Edition, Wiley Interscience, New York, 2000.

[13] A.K. Jain, R.P.W. Duin, J. Mao, Statistical pattern recognition: a review, IEEE Trans. Pattern Anal. Mach. Intell. 22 (1) (2000) 4–37.

[14] C.M. Bishop, Neural Networks for Pattern Recognition, Clarendon Press, Oxford, 1995.

[15] L. Holmström, et al., Neural and statistical classifiers—taxonomy and two case studies, IEEE Trans. Neural Networks 8 (1) (1997) 5–17.

[16] F. Kimura, K. Takashina, S. Tsuruoka, Y. Miyake, Modified quadratic discriminant functions and the application to Chinese character recognition, IEEE Trans. Pattern Anal. Mach. Intell. 9 (1) (1987) 149–153.

[17] F. Kimura, M. Shridhar, Handwritten numeral recognition based on multiple algorithms, Pattern Recognition 24 (10) (1991) 969–981.

[18] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagation errors, Nature 323 (9) (1986) 533–536.

[19] D. Wettschereck, T. Dietterich, Improving the performance of radial basis function networks by learning center locations, in: J.E. Moody, S.J. Hanson, R.P. Lippmann (Eds.), Advances in Neural Information Processing Systems 4, Morgan-Kauffmann, Los Altos, CA, 1992, pp. 1133–1140.

[20] L. Tarassenko, S. Roberts, Supervised and unsupervised learning in radial basis function classifiers, IEE Proc.-Vis. Image Signal Process. 141 (4) (1994) 210–216.

[21] J. Schürmann, Pattern Classification: A Unified View of Statistical and Neural Approaches, Wiley Interscience, New York, 1996.

[22] U. Kreßel, J. Schürmann, Pattern classification techniques based on function approximation, in: H. Bunke, P.S.P. Wang (Eds.), Handbook of Character Recognition and Document Image Analysis, World Scientific, Singapore, 1997, pp. 49–78.

[23] C.-L. Liu, M. Nakagawa, Evaluation of prototype learning algorithms for nearest neighbor classifier in application to handwritten character recognition, Pattern Recognition 34 (3) (2001) 601–615.

[24] C.J.C. Burges, A tutorial on support vector machines for pattern recognition, Knowledge Discovery Data Mining 2 (2) (1998) 1–43.

[25] V. Vapnik, The Nature of Statistical Learning Theory, Springer, New Work, 1995.

[26] C.Y. Suen, et al., Computer recognition of unconstrained handwritten numerals, Proc. IEEE 80 (7) (1992) 1162–1180.

[27] C.-L. Liu, H. Sako, H. Fujisawa, Learning quadratic discriminant function for handwritten character recognition, Vol. IV, Proceedings of the Sixteenth International Conference on Pattern Recognition, Quebec City, Canada, 2002, pp. 44–47.
[28] C.-L. Liu, K. Nakashima, H. Sako, H. Fujisawa, Handwritten digit recognition using state-of-the-art techniques, Proceedings of the Eighth International Workshop on Frontiers of Handwriting Recognition, Canada, 2002, pp. 320–325.
[29] C.-L. Liu, H. Sako, H. Fujisawa, Performance evaluation of pattern classifiers for handwritten character recognition, Int. J. Document Analysis Recognition 4 (3) (2002) 191–204.
[30] Y. LeCun, MNIST OCR data, http://www.research.att.com/yann/exdb/mnist/index.html .
[31] C.Y. Suen, J. Kim, K. Kim, Q. Xu, L. Lam, Handwriting recognition—the last frontiers, Proceedings of the Fifteenth International Conference on Pattern Recognition, Vol. 4, Barcelona, Spain, 2000, pp. 1–10.
[32] J. Franke, L. Lam, R. Legault, C. Nadal, C.Y. Suen, Experiments with the CENPARMI database combining different classification approaches, Proceedings of the Third International Workshop on Frontiers of Handwriting Recognition, Buffalo, NY, 1993, pp. 305–311.
[33] S.-W. Lee, Multilayer cluster neural network for totally unconstrained handwritten numeral recognition, Neural Networks 8 (5) (1995) 783–792.
[34] P.D. Gader, M.A. Khabou, Automatic feature generation for handwritten digit recognition, IEEE Trans. Pattern Anal. Mach. Intell. 18 (12) (1996) 1256–1261.
[35] Y.-S. Hwang, S.-Y. Bang, Recognition of unconstrained handwritten numerals by a radial basis function neural network classifier, Pattern Recognition Lett. 18 (7) (1997) 657–664.
[36] J. Franke, Isolated handprinted digit recognition, in: H. Bunke, P.S.P. Wang (Eds.), Handbook of Character Recognition and Document Image Analysis, World Scientific, Singapore, 1997, pp. 103–121.
[37] C.Y. Suen, K. Liu, N.W. Strathy, Sorting and recognizing cheques and financial documents, in: S.-W. Lee, Y. Nakano (Eds.), Document Analysis Systems: Theory and Practice, Springer, Berlin, 1999, pp. 173–187.
[38] I.-S. Oh, J.-S. Lee, C.Y. Suen, Analysis of class separation and combination of class-dependent features for handwriting recognition, IEEE Trans. Pattern Anal. Mach. Intell. 21 (10) (1999) 1089–1094.
[39] C.-L. Liu, M. Nakagawa, Handwritten numeral recognition using neural networks: improving the accuracy by discriminative training, Proceedings of the Fifth International Conference on Document Analysis and Recognition, 1999, pp. 257–260.
[40] T. Ha, H. Bunke, Off-line handwritten numeral recognition by perturbation method, IEEE Trans. Pattern Anal. Mach. Intell. 19 (5) (1997) 535–539.
[41] A. Filatov, V. Nikitin, A. Volgunin, P. Zelinsky, The AddressScriptTM recognition system for handwritten envelops, in: S.-W. Lee, Y. Nakano (Eds.), Document Analysis Systems: Theory and Practice, Springer, Berlin, 1999, pp. 157–171.
[42] J.-H. Cai, Z.-Q. Liu, Integration of structural and statistical information for unconstrained handwritten numeral recognition, IEEE Trans. Pattern Anal. Mach. Intell. 21 (3) (1999) 263–270.
[43] C.J.C. Burges, B. Schölkopf, Improving the accuracy and speed of support vector learning machines, in: M. Mozer, M. Jordan, T. Petsche (Eds.), Advances in Neural Information Processing Systems 9, MIT Press, Cambridge, MA, 1997, pp. 375–381.
[44] U. Kreßel, Pairwise classification and support vector machines, in: B. Schölkopf, C.J.C. Burges, A.J. Smola (Eds.), Advances in Kernel Methods: Support Vector Learning, MIT Press, Cambridge, MA, 1999, pp. 255–268.
[45] J.X. Dong, A. Krzyzak, C.Y. Suen, A multi-net learning framework for pattern recognition, Proceedings of the Sixth International Conference on Document Analysis and Recognition, Seattle, 2001, pp. 328–332.
[46] L.-N. Teow, K.-F. Loe, Robust vision-based features and classification schemes for off-line handwritten digit recognition, Pattern Recognition 35 (11) (2002) 2355–2364.
[47] S. Belongie, J. Malik, J. Puzicha, Shape matching and object recognition using shape contexts, IEEE Trans. Pattern Anal. Mach. Intell. 24 (4) (2002) 509–522.
[48] G. Mayraz, G.E. Hinton, Recognizing handwritten digits using hierarchical products of experts, IEEE Trans. Pattern Anal. Mach. Intell. 24 (2) (2002) 189–197.
[49] N. Hagita, S. Naito, I. Masuda, Handprinted Kanji characters recognition based on pattern matching method, Proceedings of the International Conference on Text Processing, Tokyo, 1983, pp. 169–174.
[50] J.T. Favata, G. Srikantan, S.N. Srihari, Handprinted character/digit recognition using a multiple feature/resolution philosophy, Proceedings of the Fourth International Workshop on Frontiers of Handwriting Recognition, Taipei, 1994, pp. 57–66.
[51] A. de S. Britto Jr., et al., Improvement in handwritten numeral string recognition by slant normalization and contextual information, Proceedings of the Seventh International Workshop on Frontiers of Handwriting Recognition, Amsterdam, 2000, pp. 323–332.
[52] C.-L. Liu, M. Koga, H. Sako, H. Fujisawa, Aspect ratio adaptive normalization for handwritten character recognition, in: T. Tan, Y. Shi, W. Gao (Eds.), Advances in Multimodal Interfaces—ICMI 2000, Lecture Notes in Computer Science (1948), Springer, Berlin, 2000, pp. 418–425.
[53] G. Srikantan, S.W. Lam, S.N. Srihari, Gradient-based contour encoder for character recognition, Pattern Recognition 29 (7) (1996) 1147–1160.
[54] A. Kawamura, et al., On-line recognition of freely handwritten Japanese characters using directional feature densities, Proceedings of the Eleventh International Conference on Pattern Recognition, Vol. II, 1992, pp. 183–186.
[55] T. Wakabayashi, S. Tsuruoka, F. Kimura, Y. Miyake, On the size and variable transformation of feature vector for handwritten character recognition, Trans. IEICE Japan J76-D-II (12) (1993) 2495–2503.
[56] H. Robbins, S. Monro, A stochastic approximation method, Ann. Math. Stat. 22 (1951) 400–407.
[57] S. Katagiri, C.-H. Lee, B.-H. Juang, New discriminative training learning for minimum error classification, Proceedings of the IEEE Workshop Neural Networks for Signal Processing, Princeton, 1991, pp. 299–308.
[58] B.-H. Juang, S. Katagiri, Discriminative learning for minimization error classification, IEEE Trans. Signal Process. 40 (12) (1992) 3043–3054.

[59] J.C. Platt, Fast training of support vector machines using sequential minimal optimization, in: B. Schölkopf, C.J.C. Burges, A.J. Smola (Eds.), Advances in Kernel Methods: Support Vector Learning, MIT Press, Cambridge, MA, 1999, pp. 185–208.

[60] O.L. Mangasarian, D.R. Musicant, Successive overrelaxation for support vector machines, IEEE Trans. Neural Networks 10 (5) (1999) 1032–1037.

About the Author—CHENG-LIN LIU received the B.S. degree in electronic engineering from Wuhan University, the M.E. degree in electronic engineering from Beijing Polytechnic University, the Ph.D. degree in pattern recognition and artificial intelligence from the Institute of Automation, Chinese Academy of Sciences, in 1989, 1992 and 1995, respectively. He was a postdoctor fellow in Korea Advanced Institute of Science and Technology (KAIST) and later in Tokyo University of Agriculture and Technology from March 1996 to 1999. Since then, he has been a research staff and later a senior researcher at the Central Research Laboratory, Hitachi, Ltd. His current work is mainly in developing classification algorithms for OCR applications. His research interests include pattern recognition, artificial intelligence, image processing, neural networks, machine learning, and especially the applications to character recognition and document processing.

About the Author—KAZUKI NAKASHIMA received the B.E. degree from Nagoya Institute of Technology, Japan, in 1989, and the M.E. degree in Information Engineering from Nagoya University, Japan, in 1991. Since 1991, he has been working for Central Research Laboratory, Hitachi Ltd., Japan. He has been working in research and development on pattern recognition and image processing. He is a member of the Institute of Electronics, Information and Communication Engineers (IEICE) of Japan.

About the Author—HIROSHI SAKO received the B.E. and M.E. degrees in mechanical engineering from Waseda University, Tokyo, Japan, in 1975 and 1977, respectively. In 1992, he received the Dr.Eng. degree in computer science from the University of Tokyo. From 1977 to 1991, he worked in the field of industrial machine vision at the Central Research Laboratory of Hitachi, Ltd., Tokyo, Japan. From 1992 to 1995, he was a senior research scientist at Hitachi Dublin Laboratory, Ireland, where he did research in facial and hand gesture recognition. Since 1996, he has been with the Central Research Laboratory of Hitachi, Ltd., where he directs a research group of image recognition and character recognition as a chief researcher. Dr. Sako was a recipient of the 1988 Best Paper Award from the Institute of Electronics, Information, and Communication Engineers (IEICE) of Japan for his paper on a real-time visual inspection algorithm of semiconductor wafer patterns, and one of the recipients of the Industrial Paper Awards from the 12th IAPR International Conference on Pattern Recognition, Jerusalem, Israel, 1994 for his paper on a real-time facial-feature tracking technique. He has authored about 95 technical papers and about 100 patents in the areas of pattern recognition, image processing and neural networks. He is a member of the IEEE, IEICE, JSAI and IPSJ.

About the Author—HIROMICHI FUJISAWA received the B.E., M.E. and Doctor of Engineering degrees in Electrical Engineering from Waseda University, Tokyo, in 1969, 1971, and 1975, respectively. He joined Central Research Laboratory, Hitachi, Ltd. in 1974. He has engaged in research and development work on character recognition, document understanding including mailpiece address recognition and forms processing, and document retrieval. Currently, he is a Senior Chief Researcher at Central Research Laboratory. From 1980 to 1981, he was a visiting scientist at Computer Science Department of Carnegie Mellon University, Pittsburgh. Besides working at Hitachi, he has been a visiting lecturer at Waseda University (1985–1997) and at Kogakuin University, Tokyo (1998-present). He is an IAPR Fellow, IEEE Fellow, and a member of ACM, AAAI, Information Processing Society of Japan (IPSJ), and Institute for Electronics, Information and Communication Engineers (IEICE), Japan.