Representation and Recognition of Handwritten Digits using Deformable Templates

Anil K. Jain
Department of Computer Science
Michigan State University
East Lansing, Michigan, USA
jain@cps.msu.edu

Douglas Zongker
Department of Computer Science
Michigan State University
East Lansing, Michigan, USA
zongker@cps.msu.edu

August 9, 1996

Abstract

We investigate the application of deformable templates to recognition of handprinted digits. Two characters are matched by deforming the contour of one to fit the edge strengths of the other, and a dissimilarity measure is derived from the amount of deformation needed, the goodness of fit of the edges, and the interior overlap between the deformed shapes. Classification using the minimum dissimilarity results in recognition rates up to 99.25% on a 2,000 character subset of NIST Special Database 1. Multidimensional scaling is also applied, using the dissimilarity measure as a distance, to embed the patterns as points in low-dimensional spaces. The nearest neighbor classifier is applied to the resulting pattern matrices. Methods to reduce the computational requirements, the primary limiting factor of this method, are discussed.

1 Introduction

Automatic recognition of handprinted characters has long been a goal of many research efforts in the pattern recognition field. The subproblem of digit recognition is also seen as important, not only because advances in it are expected to lead to advances in the general case, but also because of its immediate applicability to a number of fields, the most frequently cited of which is the reading of Postal ZIP codes from mail pieces.

Figure 1: Sample digit images from the NIST SD-1 data set.

The challenges in handwritten digit recognition arise not only from the different ways in which a single digit can be written, but also from the varying requirements imposed by the specific applications. The primary performance measures are classification accuracy and recognition speed—a system for reading ZIP codes from envelopes may not be appropriate for reading amounts from checks, due to the differing volumes and costs of error.

A number of schemes for digit classification have been reported in the literature. They differ in the feature extraction and classification stages employed. Many methods for extracting features from character images have been proposed. The proposed features include counts of topological features (crossings, endpoints, holes, etc.) and various mathematical moments. While these ad hoc features have performed well in many tests, they are neither intuitive nor, in many cases, generally applicable to other character sets. Classification methods used for digit recognition include nearest neighbor classifiers and multilayer perceptron networks. There has also been a recent trend to combine the outputs of multiple classifiers [12].

A more intuitive alternative to these feature extraction models is the use of deformable templates, where an image deformation is used to match an unknown image against a database of known images. We have investigated the use of image deformation to handprinted digit recognition. Therefore, our literature review includes only similar approaches; for a wider survey of digit recognition in general, refer to the recent paper by Trier *et al.* [13].

The goal of this paper is to investigate the deformation of character images as a source of information for recognition. We show that a combination of the deformation energy required to match two character images and the template matching coefficients of the resulting binary images form a good measure of dissimilarity between images. We have used this dissimilarity measure for classification of unknown images. After the literature review, we present our deformation model, discuss the use of this model for feature *extraction*, and present results with this method on a 2,000 image NIST SD-1 handwritten digit data set.

2 Deformable Models for Digit Recognition

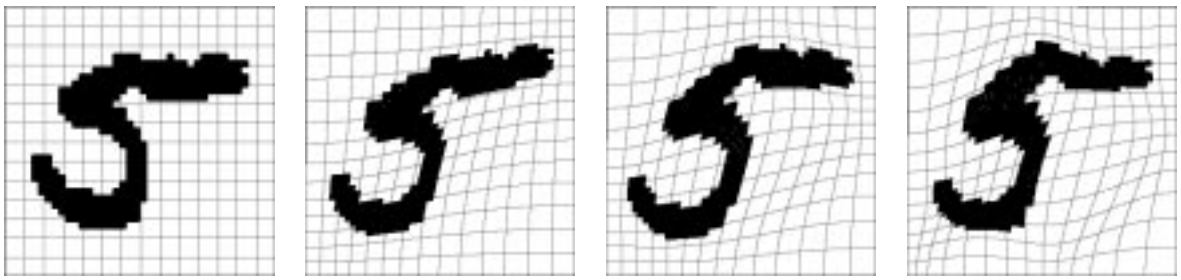
A number of studies have been reported in the literature which have applied deformable models to digit recognition. Research in this area has concentrated on taking a skeletonized digit image, representing it with a number of curve segments, and then altering the curve parameters to deform the image. Nishida [8] proposes a grammar-like model for applying deformations to structures composed of primitive strokes. Lam and Suen [7] use a two-stage method for recognition, in which samples are first classified by their structure using a tree classifier. Samples which can not be satisfactorily assigned to a class in this way are passed to a slower relaxation matching algorithm which uses deformation to match the sample to each template. They report a 93.15% recognition rate, with a 4.60% rejection rate on a 2000-sample database taken from USPS ZIP code images. Cheung *et al.* [2] model characters with a spline, and assume that the spline parameters have a multivariate Gaussian distribution. A Bayesian approach is then used to determine the character class, with the model parameters as prior and the image data parameters as likelihood. This method achieved a 95.4% recognition rate on the NIST SD-1 handprinted digit set. Revow *et al.* [9] model digits as ink-generating Gaussian "beads" strung along a spline outline. Characters are matched through deformation of the spline and adjustment of the bead parameters. Their best result reported is 99.00% recognition accuracy on a 2,000 character set with no rejections.

Simard *et al.* [10] present a digit recognition system based on an efficient distance measure that is locally invariant to transformations such as translation, rotation, scaling, stroke thickness, and others. Efficiency is further improved by using a multiresolution algorithm to differentiate very dissimilar patterns using a simpler, coarser distance measures. On a NIST-provided set of 60,000 training patterns and 10,000 test patterns, this method reached an 0.7% error rate.

Casey [1] gives a method for linear transformation of digit images, based on moment normalization, for removing some skew and orientation variation. This is used as a pre-processing step by Gader *et al.* [3] for a digit recognition system based on binary template matching. The authors report recognition rates in the range of 94.03–96.39%, with error rates in the range 0.54–1.05%.

Wakahara [15] uses iterated local affine transformation (LAT) operations to deform binary images to match prototype digit images. This method correctly identified 96.8% of a 2400-sample database, with a substitution error rate of 0.2% and a reject rate of 3%.

The deformation and matching technique used in this paper was proposed by Jain *et al.*



(a) (b) (c) (d)

Figure 2: Deformations of a sample digit image. (a) original image; (b) $M = N = 1$; (c) $M = N = 2$; (d) $M = N = 3$.

[5]. In this approach, the image is considered to be mapped to the unit square $\mathcal{S} = [0, 1]^2$. The deformation is then represented by a displacement function $D(x, y)$. These displacement functions are continuous and are zero on the edges of the unit square. The mapping $(x, y) \mapsto (x, y) + D(x, y)$ is thus a deformation of $\mathcal{S}$, a smooth mapping of the unit square onto itself. The space of displacement functions has an infinite orthogonal basis:

$$\mathbf{e}^x_{mn}(x, y) = (2\sin(\pi nx)\cos(\pi my), 0) \tag{1}$$

$$\mathbf{e}^y_{mn}(x, y) = (0, 2\cos(\pi mx)\sin(\pi ny)) \tag{2}$$

for $m, n = 1, 2, \ldots$. Low values of m and/or n correspond to lower frequency components of the deformation in the x and y directions, respectively. Figure 2 shows a series of deformations using progressively higher-order terms. Note that the deformation gets more severe as higher-order terms are included in the expansion. A parameter vector ξ can be used to represent a specific deformation function with this basis:

$$D_\xi(x, y) = \sum_{m=1}^{\infty} \sum_{n=1}^{\infty} \frac{\xi^x_{mn}\mathbf{e}^x_{mn} + \xi^y_{mn}\mathbf{e}^y_{mn}}{\lambda_{mn}}. \tag{3}$$

The parameters $\lambda_{mn} = \alpha\pi^2(n^2 + m^2)$ serve as normalizing constants.

3 Methodology

The basic goal is to determine the dissimilarity between two digit images using a deformable template approach. This is achieved by transforming one image into a template, and de-

forming it to fit the other image as closely as possible. The dissimilarity measure is defined in terms of (i) how well the deformed template fits the target image, and (ii) how much deformation was required.

In practice, we truncate the infinite series expansion of Eq. (3) to get a finite-length parameter vector ξ:

$$D_\xi(x,y) = \sum_{m=1}^{M}\sum_{n=1}^{N} \frac{\xi_{mn}^x \mathbf{e}_{mn}^x + \xi_{mn}^y \mathbf{e}_{mn}^y}{\lambda_{mn}}. \tag{4}$$

The template is fit to the target image using a Bayesian approach, as in [5]. The prior is a function of ξ; a measure of how much deformation of the template is required.

We use M and N equal to 3. This choice allows a sufficiently wide range of possible deformations, while keeping the number of parameters, and hence the computational requirements, reasonable. The parameter vector ξ consists of 9 ordered pairs. A probability density is assumed for the components of ξ. For simplicity, we assume that the terms ξ_{mn} are independent of each other, that the x and y components are independent, and that they are each Gaussian distributed with mean zero and variance σ^2. This leads to the following prior distribution on the parameter vector:

$$P(\xi) = \kappa \frac{1}{(2\pi\sigma^2)^{MN}} \exp\left\{ -\frac{1}{2\sigma^2}\left[\sum_{m=1}^{M}\sum_{n=1}^{N} ((\xi_{mn}^x)^2 + (\xi_{mn}^y)^2) \right] \right\}, \tag{5}$$

where κ is a normalizing constant.

The likelihood is determined by how well the template contour fits the edge location and direction of the target image (as determined by the Canny edge detection operator). This is given by an energy function defined at the points of the deformed template T_ξ, in terms of the deformation vector and the target image Y:

$$E(T_\xi, Y) = \frac{1}{n_T} \sum \left(1 + \Phi(x,y)\,|\cos(\beta(x,y))|\right). \tag{6}$$

Here, n_T is the number of pixels in the template outline, $\Phi(x,y)$ is an edge potential function (lowest near edge pixels in the target image Y), and $\beta(x,y)$ is the angle between the tangent direction of the template at (x,y) and the tangent direction of the nearest edge in the target direction.

We combine the prior probability and likelihood using Bayes rule to derive the following objective function, which we attempt to minimize:

$$O(T_\xi, Y) = E(T_\xi, Y) + \gamma \sum_{m=1}^{M} \sum_{n=1}^{N} \left((\xi_{mn}^x)^2 + (\xi_{mn}^y)^2 \right), \tag{7}$$

where γ provides a relative weighting between the two penalty terms.

The output of the deformation process is a single objective function value in the range $[0, 1]$, with zero indicating a perfect match with no deformation. It is important to note that this objective value is not symmetric, that is, the objective value from matching a template derived from image i to image j will not necessarily be the same as that of matching template j to image i.

The above process deforms the template so that it corresponds as closely as possible to edges in the target images. In practice, however, this is not sufficient for matching, as templates of topologically simple characters such as '1' and '0' can often be mapped on to the edges of any target image. Because of this, we also calculate binary matching coefficients between the target image and the interior of the deformed outline. The Jaccard measure (selected on the basis of its good performance in the evaluation of Tubbs [14]) is used to gauge the similarity between two binary digit images. The Jaccard measure J_{ij} between two binary images i and j is defined as

$$J_{ij} = \frac{b_{01} + b_{10}}{b_{00} + b_{01} + b_{10}}, \tag{8}$$

where b_{00} is the number of points which are object pixels in both images, and b_{10} and b_{01} count the pixels which are background in one image and object in the other. Note that this measure is actually the standard Jaccard measure [4] subtracted from one, so that *lower* values indicate better matches, just as in the objective function defined in Eq. (7).

The dissimilarity between two binary images (a template i and a target image j) is now computed as a weighted sum of the two dissimilarities defined in Eqs. (7) and (8).

$$D_{ij} = \alpha O_{ij} + (1 - \alpha) J_{ij}, \qquad 0 \leq \alpha \leq 1. \tag{9}$$

(Note that $O(T_\xi, Y)$ in Eq. (7) is here denoted as O_{ij}.) The weight α needs to be specified by the user. With this measure, a smaller value of D_{ij} indicates more similar images. Figure 3 shows the results of two deformations, one with images from the same class and one with images of differing classes. Table 1 gives the value of D_{ij} for these two pairs, with various weight values.

$O_{ij} = 0.0922$ $J_{ij} = 0.4403$ (a)

$O_{ij} = 0.1942$ $J_{ij} = 0.5818$ (b)

Figure 3: Deformed template superimposed on target image, with dissimilarity measures. (a) Template from the same class as target; (b) template from a different class.

α	D_{ij} pair (a)	D_{ij} pair (b)
1	0.0922	0.1942
1/2	0.2662	0.3880
0	0.4403	0.5818

Table 1: Dissimilarity values for the image pairs of Figure 3, for various values of weight α.

4 Multidimensional Scaling for Feature Extraction

At this point we have defined two dissimilarity measures O_{ij} and J_{ij} between a pair of character images, and can calculate an $n \times n$ proximity matrix for a set of n input images. To apply many standard pattern classification techniques, however, we need an $n \times d$ *pattern* matrix—a set of d features for each of the n patterns.

Multidimensional scaling [6] is a well-known technique to obtain an appropriate representation of the patterns from the given proximity matrix. Given an $n \times n$ input matrix of interpattern distances, multidimensional scaling creates an $n \times d$ pattern matrix; embedding the n patterns as points in a d-dimensional space, trying to keep the distances between patterns as close to the input dissimilarity matrix as possible. For a given d, the algorithm minimizes a stress value, which measures the similarity between the given proximity matrix and the interpoint distances of the output pattern matrix. The pattern matrices produced by two sample multidimensional scaling runs (corresponding to the starred entries of the table in Figure 6) are shown in Figures 4 and 5.

It is expected that given a meaningful set of interpattern distances as input, the mul-

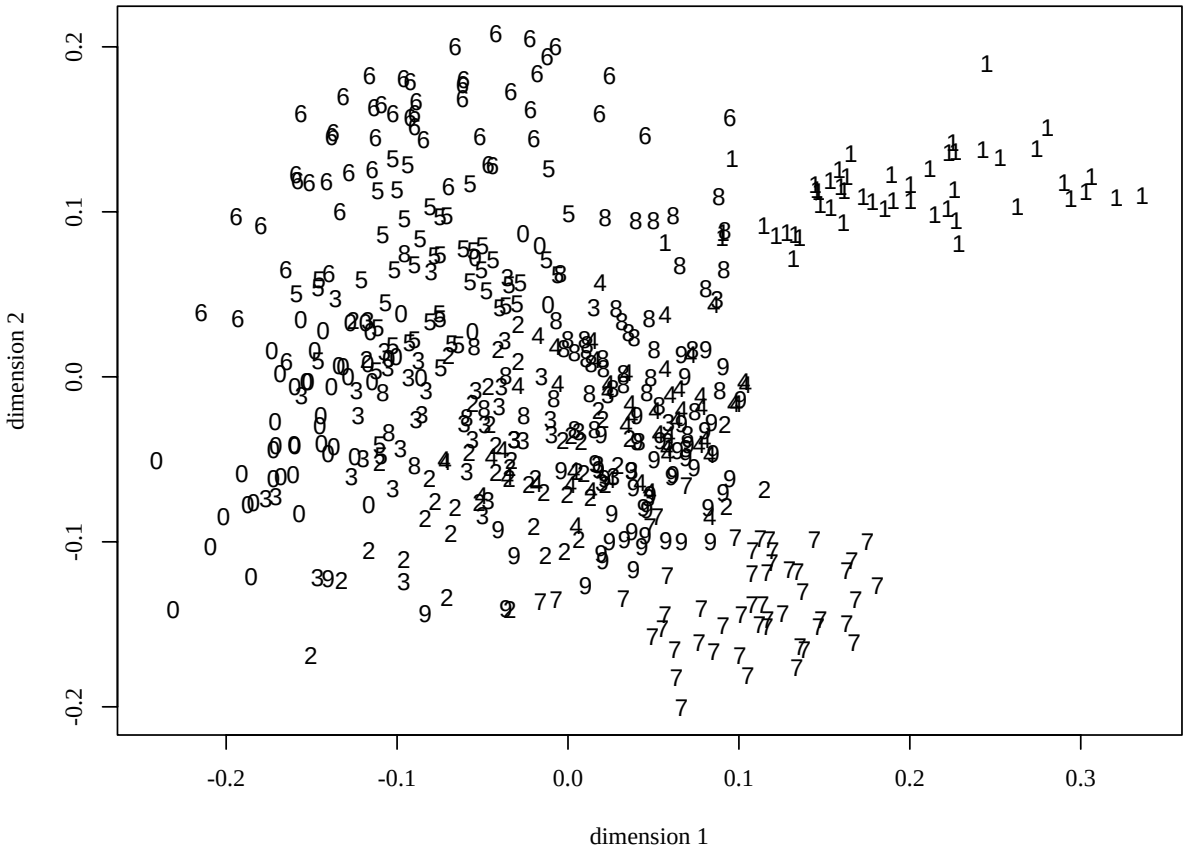


Figure 4: Two-dimensional pattern matrix produced by multidimensional scaling, with $\alpha = 1/2$.

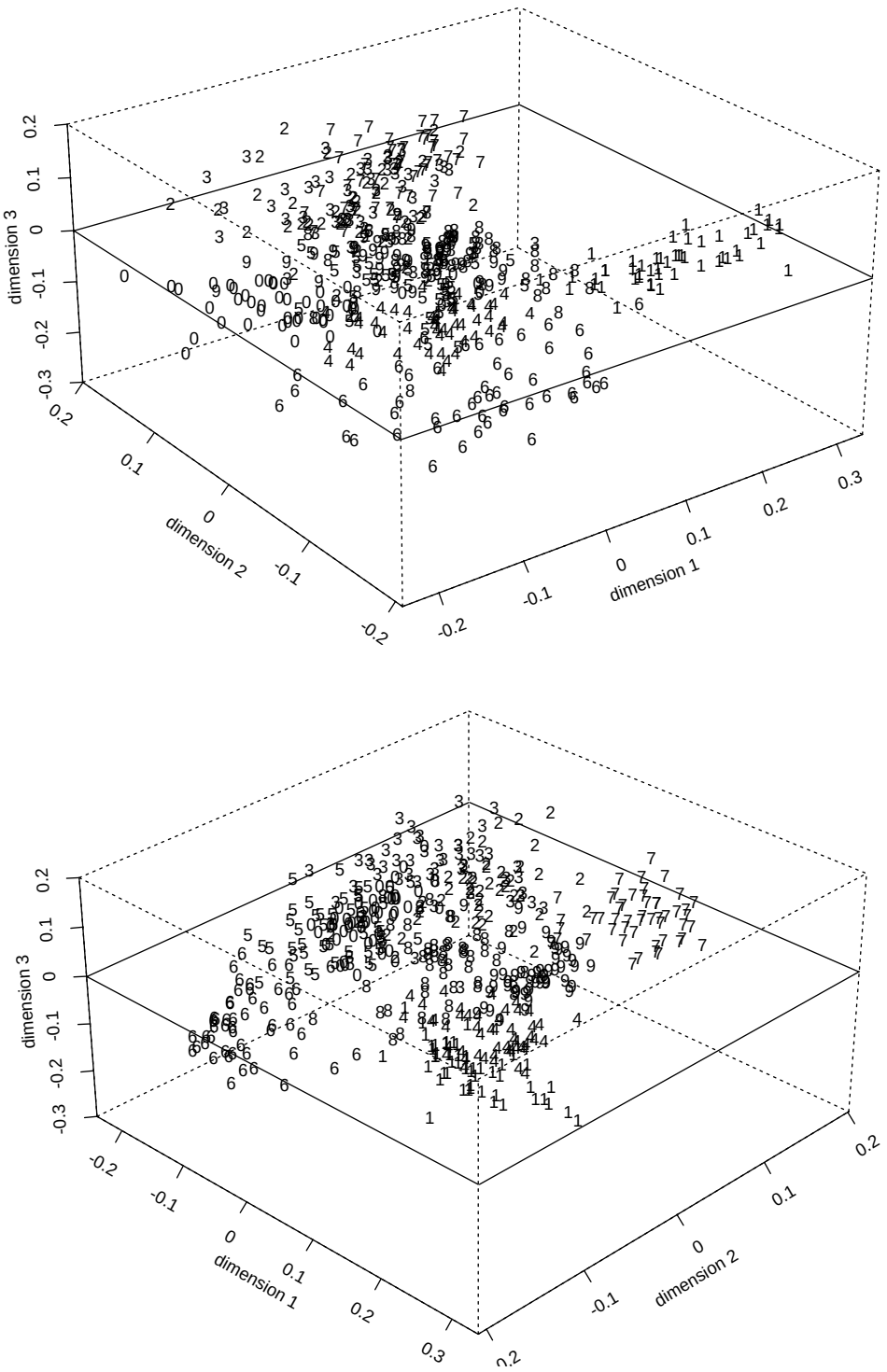


Figure 5: Three-dimensional pattern matrix produced by multidimensional scaling, with $\alpha = 1/2$, from two different perspectives.

tidimensional scaling algorithm [11] will generate a pattern matrix that represents pattern classes as compact and isolated clusters in a feature space. We have applied multidimensional scaling to the dissimilarity matrices produced by the deformable template method, and used a nearest-neighbor (NN) classifier to evaluate the quality of the resulting pattern matrix or the representation space.

The stress values obtained using this procedure for different values of d (dimensionality of the representation space) are given in Table 2 and plotted in Figure 6. Three different values of α were used: 0, 1, and 1/2. These correspond to using the objective function value O_{ij} only, the Jaccard measure J_{ij} only, and an equally weighted sum of the two. Each dissimilarity matrix was averaged with its transpose to produce a symmetric distance matrix. Due to computational limitations, only 500 of the 2,000 patterns in the database were used in this analysis. So, an attempt was made to embed 500 patterns in feature spaces of dimensionality ranging from 2 to 9. Stress generally decreases as d increases over this range. It is generally suggested that a stress value below 0.05 corresponds to a "good" representation. The quality of the derived representation will be determined based on the classification results in the next section.

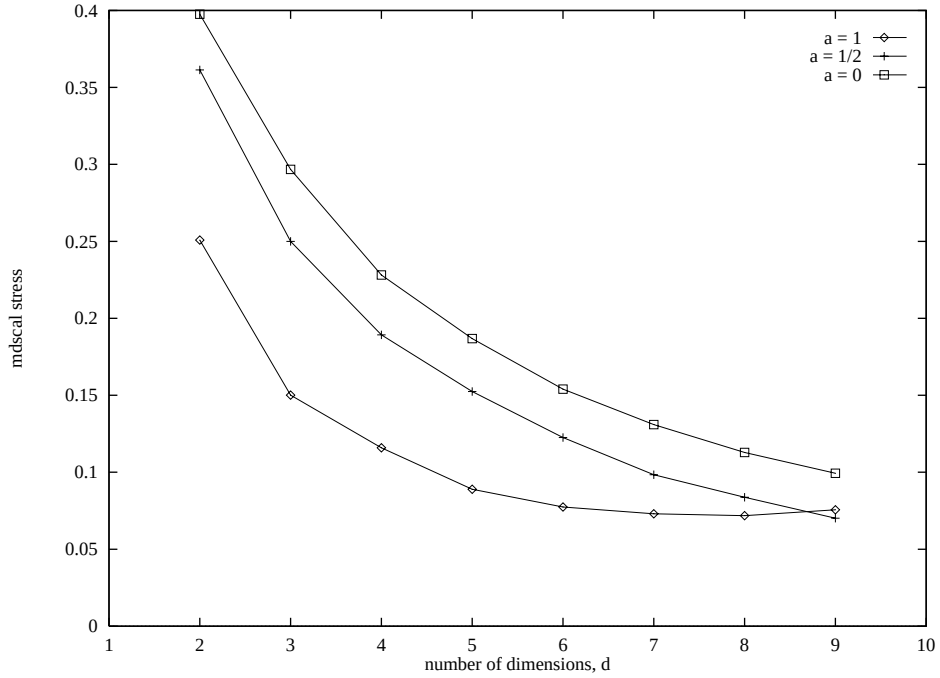


Figure 6: Plot of multidimensional scaling stress vs. number of features.

	# of dimensions, d							
α	2	3	4	5	6	7	8	9
1	0.2509	0.1501	0.11588	0.08897	0.07742	0.07300	0.07179	0.07561
1/2	0.3614*	0.2500*	0.18922	0.15244	0.12255	0.09837	0.08375	0.07016
0	0.3976	0.2968	0.22817	0.18680	0.15400	0.13094	0.11287	0.09934

Table 2: Multidimensional scaling stress values, for various dissimilarity measures and dimensionalities. Pattern matrices for the entries marked with * are plotted in Figures 4 and 5.

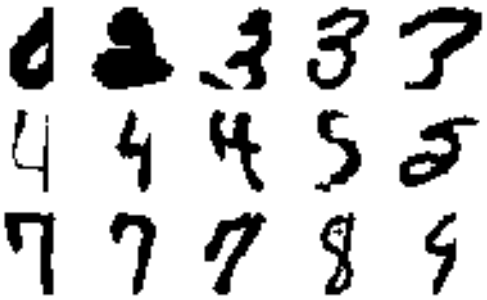
5 Classification Results

All results presented here are based on a 2,000 character sample from NIST Special Database 1. Each character is a 32×32 binary image. A 4-pixel-wide border was placed around each image to allow the deformation process some room to adjust the template in, so the actual image size used was 40×40 pixels.

We use the dissimilarity value D_{ij} in Eq. (9) to classify each target image. A leave-one-out approach is used, with two different ways of calculating the dissimilarity value. In the first ("asymmetric"), the unknown image is classified by taking it as the target image, and each of the other 1,999 images as templates in turn. The unknown image is assigned to the class of the template with the minimum dissimilarity value. The second ("symmetric") method also compares the unknown image with the other 1,999 images but instead of treating the unknown image as the target and the known image as the template, it performs the deformation both ways and averages the results. While the second method gives better results, it has the disadvantage of requiring twice as many deformations to classify an unknown image. Table 3 gives the classification accuracies for different values of the weight α.

	correct classifications	
α	asymmetric	symmetric
1	952/2000 (47.60%)	1873/2000 (93.65%)
1/2	**1957/2000 (97.85%)**	**1985/2000 (99.25%)**
0	1951/2000 (97.55%)	1971/2000 (98.55%)

Table 3: Classification accuracies using the dissimilarity value D_{ij}.



6	6	2	8	7
6	9	7	9	2
9	9	9	9	4

(a) (b)

Figure 7: Misclassified digits by the best classifier of Table 3. (a) The fifteen input images that were misclassified; (b) the classes assigned by the classifier.

The 15 images misclassified using the symmetric dissimilarity with $\alpha = 1/2$ are shown in Figure 7. Some of these images are very difficult to classify, even by a human expert.

	# of dimensions, d							
α	2	3	4	5	6	7	8	9
1	0.430	0.710	0.720	0.846	0.884	0.902	0.894	0.912
1/2	0.552	0.804	0.894	0.922	0.958	0.960	0.960	0.970
0	0.526	0.786	0.904	0.938	0.942	0.960	0.946	0.962

Table 4: Results of 1NN classifier applied to the pattern matrix derived from multidimensional scaling.

Classification was also done by using a nearest-neighbor algorithm on the pattern matrix produced by the multidimensional scalings of Section 4. A leave-one-out approach was used. These results are given in Table 4 and plotted in Figure 8. The best 1NN recognition rate obtained was 97.0%, using $\alpha = 1/2$, with 9 dimensions. While this technique is impractical for use as a classifier in a production system (the computationally expensive multidimensional scaling algorithm would have to be applied for each digit to be classified), it does illustrate the existence of a relatively small set of features that give good classification performance with a simple classifier such as 1NN. These results should motivate us to search for a good representation space for handwritten digits.

The computational requirements of our deformable template approach to digit classification are high. To classify a single character against the database of 2,000 images, using

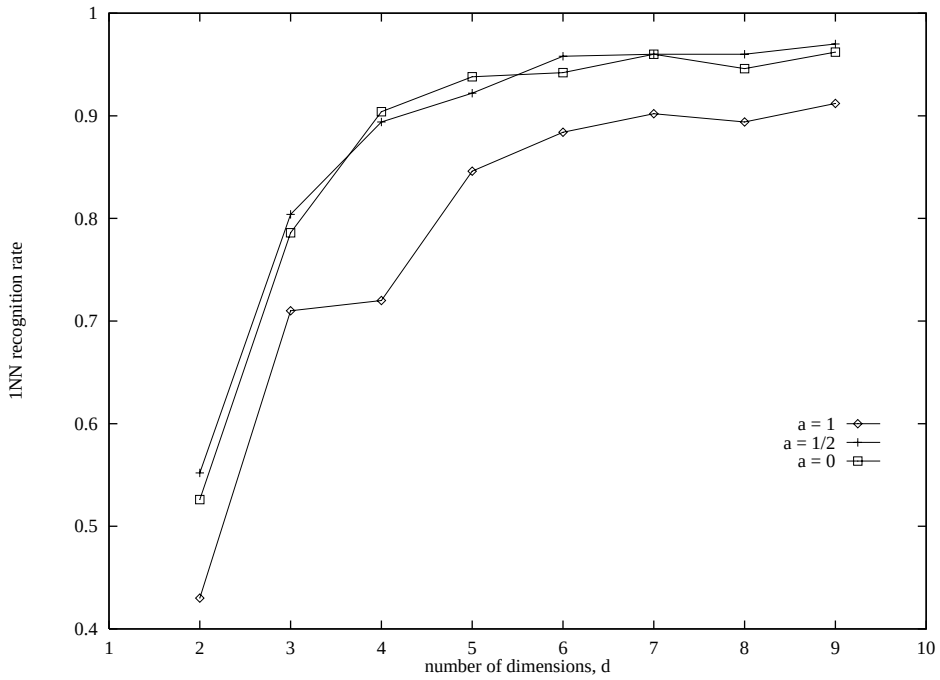


Figure 8: Plot of 1NN recognition accuracy vs. number of dimensions.

the asymmetric dissimilarity would require running the deformation process 2,000 times, which takes approximately 64 CPU minutes on a SPARCstation 20/61. Use of the symmetric dissimilarity doubles the necessary computational effort. To use the NN method as a classifier would require additionally rerunning the multidimensional scaling process for the 2,000 database images plus the test images. Obviously, this does not make for a practical classifier.

One way to reduce this computational burden would be to reduce the size of the training set, by selecting a small number of images to serve as prototypes for the whole class. One approach would be to cluster the patterns of each class, and select a representative of each cluster. To implement this strategy, we performed a complete-link hierarchical clustering on the patterns of each class, independently. The resulting dendrogram was cut to form p clusters. To choose a representative from each resulting cluster, the sum of dissimilarities from each member to all other members of the cluster was computed. The member with the minimum such sum is chosen. In this way, p prototype images from each class are chosen, $10p$ images in all for the digit database. A sample dendrogram is shown in Figure 9.

The prototype set is tested using the minimum dissimilarity classification method, as in Table 3. The symmetric dissimilarity value is used. Instead of a leave-one-out method as

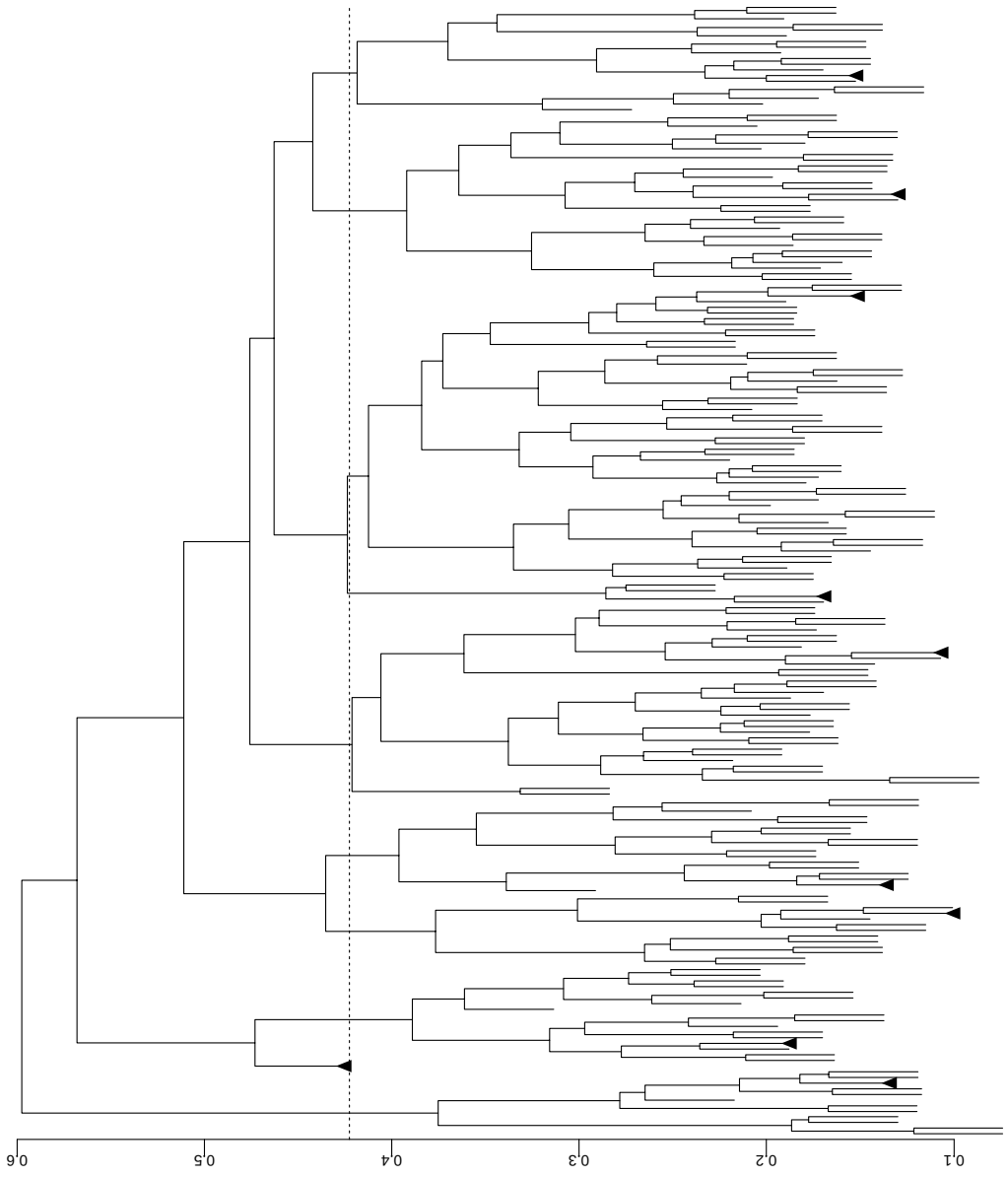


Figure 9: Sample dendrogram for the '4' class, $p = 10$, $\alpha = 1/2$. The dotted line indicates where the cut was made to form 10 clusters. The leaf nodes corresponding to the selected prototypes are marked with triangles.

α	# of prototypes per class, p 5	10	20	30	full database ($p = 200$)
1	0.843	0.890	0.925	0.938	0.937
1/2	0.950	0.963	0.982	0.988	0.993
0	0.929	0.940	0.971	0.975	0.986

Table 5: Classification accuracy of least dissimilar prototype pattern matching.

above, a holdout method is used—the prototypes form the training set, and the remaining images form the test set. The classification accuracy using this method, for different values of p, is shown in Table 5.

Comparing the case of 30 prototypes per class to the use of the full database (equivalent to 200 prototypes per class), the recognition rate drops slightly for two of the three values of α tested. However, the computational effort required to classify an unknown character with 30 prototypes per class is only 15% of that needed when the full database is used. It should also be noted that the reduced databases are tested using a holdout method, with separate training (selected prototypes) and test patterns, while the recognition rate reported for the full database uses a leave-one-out method.

6 Summary

We have used a deformable template approach for the purpose of handprinted digit recognition. The deformation system used represents one binary image in terms of its contour, and then iteratively computes parameters of a continuous displacement function in order to map the contour template as closely as possible onto the edges of the other binary target image.

Two dissimilarity measures between character image pairs have been defined: a measure of the amount of deformation needed, and the Jaccard binary matching coefficient between the target image and the deformed template image. Classifying each image using the minimum dissimilarity to all the other templates produced over 99% accuracy on a 2,000 image database.

Future work will focus on reducing the computational requirements of this method, through faster deformation software and better selection of representative prototypes from the training set.

References

[1] R. G. Casey. Moment normalization of handprinted characters. *IBM Journal of Research and Development*, pages 548–557, November 1970.

[2] K.-W. Cheung, D.-Y. Yeung, and R. T. Chin. A unified framework for handwritten character recognition using deformable models. In *Proc. Second Asian Conference on Computer Vision*, volume I, pages 344–348, 1995.

[3] P. Gader, B. Forester, M. Ganzberger, A. Gillies, B. Mitchell, M. Whalen, and T. Yocum. Recognition of handwritten digits using template and model matching. *Pattern Recognition*, 24(5):421–431, 1991.

[4] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, 1988.

[5] A. K. Jain, Y. Zhong, and S. Lakshmanan. Object matching using deformable templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(3), March 1996.

[6] J. B. Kruskal. Multidimensional scaling and other methods for discovering structure. In K. Enslein, A. Ralston, and H. S. Wilf, editors, *Statistical Methods for Digital Computers*, pages 296–339. John Wiley & Sons, 1977.

[7] L. Lam and C. Y. Suen. Structural classification and relaxation matching of totally unconstrained handwritten zip-code numbers. *Pattern Recognition*, 21(1):19–31, 1988.

[8] H. Nishida. A structural model of shape deformation. *Pattern Recognition*, 28(10):1611–1620, 1995.

[9] M. Revow, C. K. I. Williams, and G. E. Hinton. Using generative models for handwritten digit recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(6):592–606, June 1996.

[10] P. Y. Simard, Y. Le Cun, and J. S. Denker. Memory-based character recognition using a transformation invariant metric. In *Proc. 12th International Conference on Pattern Recognition*, pages 262–267, October 1994.

[11] Statistical Sciences, Inc. *S-PLUS 3.2*, 1993.

[12] C. Y. Suen, R. Legault, C. Nadal, M. Cheriet, and L. Lam. Building a new generation of handwriting recognition systems. *Pattern Recognition Letters*, 14:303–315, April 1993.

[13] Ø. D. Trier, A. K. Jain, and T. Taxt. Feature extraction methods for character recognition–a survey. *Pattern Recognition*, 29(4):641–662, 1996.

[14] J. D. Tubbs. A note on binary template matching. *Pattern Recognition*, 22(4):359–365, 1989.

[15] T. Wakahara. Shape matching using LAT and its application to handwritten numeral recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):618–629, June 1994.