B. Detecting Two Cups

The Probabilistic Hough Transform is first applied to detect the two inner circular boundaries of the cups using a *preset* poll size. The experimental method is similar to the case of detecting one object, except that an error is registered if any of the *two* highest peaks detected is not sufficiently close ($<\Delta d$) to any of the two reference peaks. The dashed curves in Fig. 4(a) and Fig. 4(b) show the average error rate in 10 000 trials as a function of the (preset) percentage of edge points used for the "clean" and noisy edge images, respectively.

Adaptive termination of voting is in this case based on monitoring the stability order of the set of peaks in the peak list that consists of the *two* highest peaks. The definition of an error is as above, i.e., the inner circular boundary of both cups must be detected. The solid curves in Fig. 4(a) and Fig. 4(b) show the average error rate in 10 000 trials as a function of the average percentage of edge points used for the clean and noisy edge maps respectively. The marks indicate operating points in which the stability order (of the set of two peaks) is 10.

If the number of objects to be detected is not specified, adaptive termination of voting is based on the *maximum stability order* of peak sets, stopping when it reaches a certain value, and taking the *maximum stability count* as an indication on the number of objects reliably detected when voting stopped. In this case, error registration depended on the maximum stability count. If it was 2 or more, an error was registered as above, i.e., if any of the two highest peaks detected was not sufficiently close ($<\Delta d$) to any of the two reference peaks. If the maximum stability count was 1, an error was registered only if the highest peak detected was not close enough to one of the two highest reference peaks. The average results of 10000 trials are summarized in Table II(a) and Table II(b) for the clean and noisy edge images respectively (Fig. 1(a) and Fig. 1(b)). The dotted curves in Fig. 4 show the average error rate as a function of the average poll size. The very low error rates (in comparison to the dashed and solid curves) should be interpreted in light of the different error definition.

IV. Conclusion

Methods for adaptive setting of the poll size in the Probabilistic Circular Hough Transform have been presented. Successful adaptation in an actual application has been experimentally demonstrated. In average, less votes are used than would be needed to achieve the same error rate with a fixed poll size.

The performance of object recognition algorithms in general, and of the Probabilistic Hough Transform in particular, depends on the task definition. Detecting one appearance of an object is easier than correctly detecting two appearances of that object. Detecting an unknown number of objects by Hough techniques using partial data is difficult in the presence of noise. The approach taken is to stop voting when any number of appearances of an object seem to have been reliably recognized, even though the existence of other appearances is not ruled out.

The suggested stopping rules are based on stability of just the *ranks* of the highest peaks in peak lists derived from the accumulator array. This leads to elegant stopping rules and reduces dependence on tuning parameters. The potentially valuable information in the absolute peak heights is however disregarded.

Analytic approach to the problem of adaptive termination of voting seems difficult, especially if the number of objects to be detected is not predefined. The equivalence of the Hough Transform to template matching [7] and its relation to model-based vision [3] provide valuable insight. Note that relying on detailed models of the image content and imaging process yields results that cannot be easily generalized. On the other hand, with little use of a priori knowledge powerful results are difficult to obtain. Due to the lack of theoretical sensitivity analysis, more experimentation is needed in order to characterize the performance of the algorithm in other applications. Our limited experience indicates that around a reasonable operating point, the sensitivity to algorithm internals such as the batch size, peak merging threshold and peak lists length, to edge detector properties and especially to random noise are sufficiently small to enable steady operation.

Acknowledgment

The authors are grateful to Mr. S. Du Pasquier for writing several computer programs, and to Dr. F. Ade for many helpful suggestions. N. Kiryati is grateful to Prof. Dr. O. Kübler for the support, hospitality, and stimulating discussions during his visits to the ETH.

References

[1] F. Ade, M. Peter, M. Rutishauser, M. Trobina, and A. Ylä-Jääski, "A 3-D vision system for deriving gripping information for a robot," in *Proc. 8th SCIA, Tromsø*, Norway, May 1993.

[2] G. Gerig, "Linking image-space and accumulator-space: A new approach for object recognition," in *Proc. 1st ICCV*, London, 1987, pp. 112–117.

[3] W. E. L. Grimson and D. P. Huttenlocher, "On the verification of hypothesized matches in model-based recognition," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, pp. 1201–1213, 1991.

[4] N. Kiryati, Y. Eldar, and A. M. Bruckstein, "Probabilistic Hough transform," *Pattern Recognit.*, vol. 24, pp. 303–316, 1991.

[5] V. F. Leavers, "Which Hough transform?" *CVGIP: Image Understanding*, vol. 58, pp. 250–264, 1993.

[6] J. Sheinvald, B. Dom, W. Niblack, and S. Banerjee, "Detecting parametrized curve segments using MDL and the Hough transform," in *Proc. CVPR'92*, Champaign, Illinois, June 1992, pp. 547–552.

[7] J. Sklansky, "On the Hough technique for curve detection," *IEEE Trans. Comput.*, vol. C-27, pp. 923–926, 1978.

[8] L. Xu, E. Oja, and P. Kultanen, "A new curve detection method: Randomized Hough transform (RHT)," *Pattern Recognit. Lett.*, vol. 11, pp. 331–338, 1990.

[9] A. Ylä-Jääski, "Contributions to a 3-D robot vision system: Grouping from sparse and incomplete data," *Acta Polytechnica Scandinavica*, Elect. Eng. Ser. No. 73, Helsinki, 1993.

Handwritten Character Classification Using Nearest Neighbor in Large Databases

Stephen J. Smith, Mario O. Bourgoin, Karl Sims, and Harry L. Voorhees

***Abstract*—We show that systems built on a simple statistical technique and a large training database can be automatically optimized to produce classification accuracies of 99% in the domain of handwritten digits. It is also shown that the performance of these systems scale consistently with the size of the training database, where the error rate is cut by**

Manuscript received August 1, 1992; revised April 12, 1994. Recommended for acceptance by Associate Editor B. Bhanu.

M. O. Bourgoin, K. Sims, and S. J. Smith are with the Thinking Machines Corporation, 245 First St., Cambridge, MA 02142 USA; e-mail: mob@think.com, karl@think.com, smith@think.com.

H. L. Voorhees is with TASC, 55 Walkers Brook Dr., Reading, MA 01867 USA; e-mail: hlv@tasc.com.

IEEE Log Number 9403774.

more than half for every tenfold increase in the size of the training set from 10 to 100,000 examples. Three distance metrics for the standard Nearest Neighbor classification system are investigated: a simple Hamming distance metric, a pixel distance metric, and a metric based on the extraction of penstroke features. Systems employing these metrics were trained and tested on a standard, publicly available, database of nearly 225,000 digits provided by the National Institute of Standards and Technology. Additionally, a confidence metric is both introduced by the authors and also discovered and optimized by the system. The new confidence measure proves to be superior to the commonly used Nearest Neighbor distance.

***Index Terms*— Optical character recognition, machine learning, machine vision, parameter optimization, Nearest Neighbor classification.**

I. Introduction

Some recent trends in machine intelligence and learning are based on the belief that statistical techniques may achieve superior performance on a wide range of perceptual tasks, compared to the artificial intelligence (AI) approaches of reasoning, expert systems, and model building [1], [6], [8]. Statistical techniques were used long before the advent of AI. What is it then that makes them more attractive today? We believe that one of the main reasons is the new possibility of processing very large amounts of data using supercomputers, especially massively parallel ones.

We have taken this statistical approach to learning by building simple k-Nearest Neighbor classification systems (k-NN) [3], [7], [9], [20] on large training databases and then providing these systems with the ability to optimize their own performance. By keeping these systems simple and running them on a supercomputer we are able to perform the many iterations of testing necessary for their optimization. These methods include gradient descent search for the relative weightings of penstroke features and usable terms in a confidence measure. As a result we have produced systems that have required a minimum of hand coding and that have been shown to be of nearly equal performance to the best currently available systems for handwritten digit recognition [22].

A. Results from the NIST OCR Conference

Though much work has been done on the automatic recognition of handwritten characters it has often been difficult to compare performance results [2], [13], [19]. To address this problem a conference was sponsored by the National Institute of Standards and Technology (NIST) at which twenty-nine groups from Europe and North America came together to compare the performance of their OCR systems on a common set of segmented handwritten characters. Both commercial and research systems were compared. The discriminant functions used ranged from neural networks to k-NN classifiers and included many proprietary algorithms that were not disclosed.

Many of the systems were trained on a database of segmented characters collected from 2,100 U.S. Census employees [12], which consisted of 223,125 classified digits, 44,951 upper-case alphas, and 45,313 lower case alphas. All systems were tested on a database collected from 500 high school students. The high school database proved to be a fairly difficult test set, as even human readers could do no better than a 1.5% error rate (e.g., this would correspond to 75 character errors on this page).

The results of the NIST test showed that systems that were trained solely on the Census database had poorer performance than those that were trained on data sets incorporating large additional databases. One possible reason for this was that the high school database appeared to be both more difficult than and significantly different from the Census database. The proportion of crossed sevens, for instance, was much higher in the high school database than in the Census database. It must also be pointed out that, though the performance analysis was rigorous, the actual collection of classification categories was unproctored by NIST (a full description of the experimental setup and test results can be found in [22]). Thus specific performance results in this conference may not completely reflect performance on other databases or in other domains.

TABLE I
Selected NIST Digit Results on High School Test Database

Method	Error Rate 0% Reject (rank)	Error Rate 50% Reject (rank)
Human	1.5%	*
Proprietary	1.56% (1)	*
k-NN	3.16% (2)	0.11% (7)
k-NN	3.35% (3)	*
Proprietary	3.43% (5)	0.32% (17)
Neural Net	3.49% (6)	0.38% (19)
Penstroke**	3.85% (8)	0.08% (6)
Pixel Distance**	4.89% (22)	0.06% (2)

* No confidence measure supplied.
** Techniques described in this correspondence.

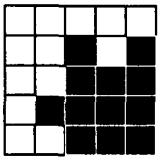
The k-NN metrics presented in this paper (penstroke and pixel distance) were used in the NIST conference and were trained solely on the Census database. They nonetheless showed good performance compared with the other 46 systems represented. The penstroke extraction algorithm placed 8th overall with an error rate of 3.85% at zero rejection and the pixel distance algorithm was 22nd with an error rate of 4.89% at zero rejection. The confidence measures for the penstroke and pixel distance metrics proved to be among the best, however, as the penstroke metric moved from 8th to 6th and the pixel distance metric improved from 22nd to 2nd as the rejection criterion was increased to 50% (see Table I).

B. Preprocessing and Normalization

In the experiments reported here only the NIST digit database collected from Census workers was used. The original database consisted of 128×128 one bit, unnormalized images along with a classification of the image (not necessarily the correct one) and the ID of the author. On average the automatic segmentation system used to extract the character images from the image blocks was able to extract only 106 out of the 133 possible digit images per author. To normalize the original images by size, a minimal square bounding box was fit to the set of nonzero pixels in each image, and the containing area was downsampled to a 32×32 pixel bitmap. No attempt was made to normalize for aspect ratio, shear, rotational variations, or pen thickness, nor to filter out random noise, since our experiments concerned the effect of database size rather than preprocessing.

C. Testing Procedures

The performance results described in this paper were obtained via a validation test where the testing and training data sets were always disjoint. For the penstroke metric, test set validation was done using held-aside data. For the Hamming and pixel distance metrics, N-way cross-validation was used. In the latter case, for each example in the database, the example and all other examples from the same author were removed from the database and the remaining examples were used as the training set. Nearly the entire database could thus be used for both testing and training without any contamination of the test set from the training set. Examples from the same author as the test example were removed from the training database because it was found that their inclusion unfairly increased the tested performance



Bitmap B

3	2	1	2	1
2	1	1	1	1
2	1	1	1	2
1	1	2	2	3
2	1	1	2	3

Distance Map D(B)

Fig. 1. Example Distance Map (Distance values are rounded up to integers, for clarity).

of the system. All performance tests among the three systems were conducted by using at least 30,000 test examples.

II. k-Nearest Neighbor Distance Metrics

Three classification metrics were used in our experiments. All were built on a simple k-NN classification scheme. Using a massively parallel supercomputer (the Connection Machine CM-2), the distance metric is computed in parallel between a test image and all training examples, which are distributed across the processing elements. For single neighbor ($k = 1$) classification, the example with the smallest distance is selected using a reduction operator, and its associated classification determines the classification of the test image.

A. Hamming Metric

The Hamming metric is the simplest of the three we tested; it counts the number of mismatched pixels. On the CM-2, this is trivially computed in parallel as the sum of the pixel-wise exclusive or of the two bitmaps B_1 and B_2:

$$\sum B_1 \oplus B_2. \tag{1}$$

B. Pixel Distance Metric

The second metric tested, pixel distance, is slightly more sophisticated. For mismatched pixels between the test and training images, it takes into account the distance to the nearest pixel of the same color (i.e., foreground or background). We use an integral approximation to the Euclidean distance between pixel centers, truncated to 4 bits to conserve memory. The total difference between two bitmaps B_1 and B_2 is:

$$\sum (B_1 \oplus B_2)(D(B_1) + D(B_2)) \tag{2}$$

where $D(B_i)$ is the distance map representing distances to the nearest pixel of a different color (Fig. 1). Descriptions of similar techniques can be found in [4], [5], [10], [11], [18], [21].

C. Penstroke Metric

The third metric tested, penstroke, makes use of a priori information regarding the manner in which the digit images were formed, by extracting a set of penstroke features from each bitmap. The distance is computed using a weighted value of penstroke feature parameters.

Penstroke features are extracted by thinning the image [17], segmenting the resulting skeleton at intersections, and breaking the resulting contours at significant maxima of curvature. The curvature threshold is made proportional to the length of the contour, so that longer strokes are more easily broken, and the results are less scale dependent. In practice, several penstroke features may end up representing a single real penstroke due to spurious intersections or kinks; we made no attempt to smooth or rejoin the computed penstroke features.

Each penstroke feature is abstracted into an "arc" representation, characterized by the following parameters: length, center point of contour, direction vectors from center point to each endpoint, and number of endpoint connections: 0 (free endpoint), 1 (kink), or 2+ (intersection).

The distance between two digits represented as sets of arcs is computed as the minimum sum of the difference between each possible pairing of arc features between the two sets (since only 3 to 4 arcs are extracted from each image, on average, an exhaustive algorithm is feasible). The difference between two arcs is the weighted sum of squares of the attribute differences; the actual weights were determined by gradient descent, as described in Section IV-A. Since a given arc may match better in one orientation than another, the endpoints of the arcs are compared in both the original and reversed order and the minimum distance of the two is retained.

TABLE II
Database Size vs. Performance

Training Examples	Hamming Error %	Pixel Distance Error %
10	49.05	39.34
30	40.96	32.13
100	29.78	23.44
300	22.19	15.79
1,000	13.71	9.26
3,000	8.38	5.18
10,000	5.05	3.12
30,000	3.37	2.08
100,000	2.29	1.39
223,125	1.86	1.12

III. Performance

The three distance metrics achieved zero-rejection error rates of 1.9% for Hamming, 1.1% for pixel distance and 1.0% for penstroke on the test data with the full training database of 223,125 examples. Though the Hamming metric had nearly twice the error rate of the other methods, it ran nearly ten times faster than and in only 1/10th the memory space of the other algorithms. It is may thus still be of interest for commercial systems.

A. Performance and Database Size

Since these three systems performed remarkably well for the simplicity of their algorithms, it was considered that the training database itself and its size were of critical importance to their overall performance. This issue was investigated and the results are presented in Table II.

The smaller-sized training sets were created by taking the first n examples (where n is the size of the database) in the full training set that were not of the same author as the test example, and such that each class was equally represented (e.g., if the database size was 100 there would be exactly 10 zeros, 10 ones, etc.). The training databases were constructed in this way so that new authors were also added to the database as the database grew, thus the diversity of the database increased as well as its size (up to 2,100 authors or approximately 100 examples per author). The same test database of approximately 50,000 examples was used for all the experiments.

For every tenfold increase in database size the error rate is cut by half or more though the performance seems to be leveling off slightly for the larger database sizes. Although we lacked the data to do so, it would be interesting to see whether performance effectively reaches a plateau as the database size is increased further; this level would serve as a measure of a given metric's peak performance. It is also interesting to note the power of very small database sizes. With only a single randomly selected example representing each class the pixel

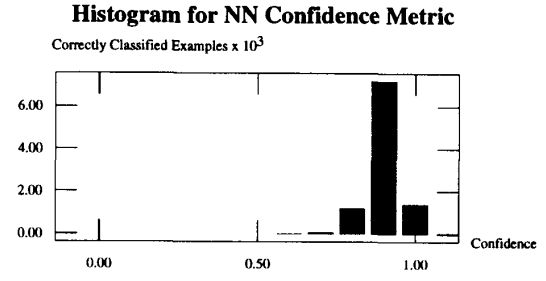


Fig. 2. NN confidence—correct examples.

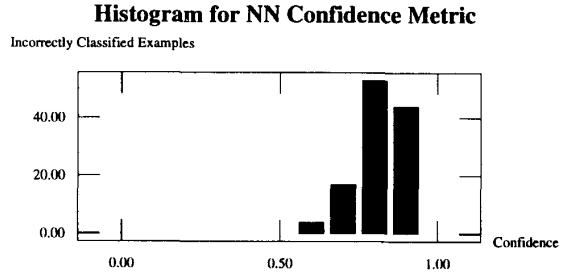


Fig. 3. NN confidence—incorrect examples.

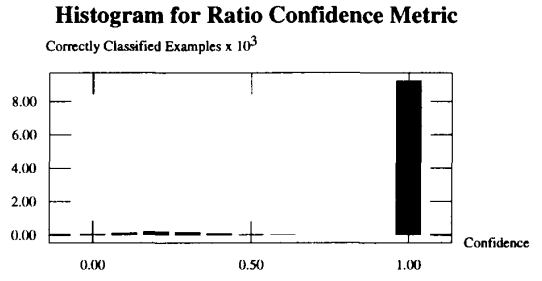


Fig. 4. Ratio confidence correct examples.

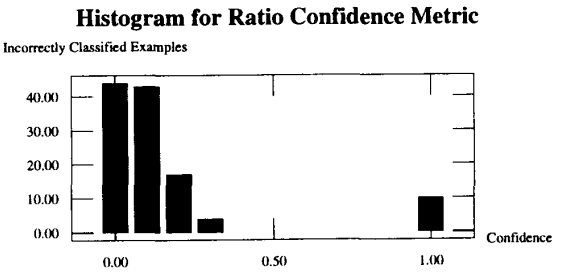


Fig. 5. Ratio confidence incorrect examples.

distance metric achieves over 60% correct (surprising, since random guessing would yield only 10% correct).

B. Confidence Measures

It was initially assumed that the distance between the test and the nearest training example would be a good indication of the confidence in the classification (i.e. the larger the distance the lower the confidence). Though there is a relationship between the confidence and the k-NN distance, it was surprisingly weak. Figures 2 and 3 show the confidence distributions of the correctly and incorrectly classified examples, based on k-NN distance. The k-NN confidence measure was calculated as:

$$1 - \frac{D_1}{K} \tag{3}$$

where D_1 is the distance to the nearest neighbor, computed using the pixel distance metric, and K is a constant, greater than the maximum distance (we used 1,000), to insure that the measure lies on the range 0 to 1. If this were a good measure of confidence, correctly classified examples would be distributed around a higher confidence value than incorrect examples. In fact, the two distributions are quite similar.

Inspection of the data on nearest neighbors and distances revealed that the 15 nearest neighbors to a correctly classified test example were often all of the same class as the nearest neighbor and that the distance to the nearest neighbor did not seem to be a particularly good predictor of errors. The test cases that were most likely to be incorrectly classified were better represented by closeness between competing categories. For instance if the first and second nearest neighbors of an image are distances of 100 and 101 away but represent different classes we would have less confidence in our classification than for an image whose nearest neighbor was a distance of 900 away yet all of the 15 nearest neighbors predicted the same category.

To incorporate this knowledge of closely competing categories into our confidence measure, we considered the following confidence measure:

$$1 - \frac{D_1}{\overline{D}_1} \tag{4}$$

where D_1 is again the distance to the nearest neighbor, and where $\overline{D}_1$ represents the distance to the nearest neighbor of a class other than the class of the nearest neighbor. We call this the ratio confidence measure.

Since D_1 can never be less than $\overline{D}_1$ the confidence varies between 0 (when there is a tie between two competing classes) and 1 (when the nearest competing class is an infinite distance away from the test example).

Figures 4 and 5 display the distributions of the correctly and incorrectly classified test examples with respect to the ratio confidence measure, again computed using the pixel distance metric. In this case, the two distributions are well separated, providing much better discrimination than the k-NN confidence measure. In fact, the performance of our system relative to others, as reported in the right hand column of Table I, suggests that this may indeed by a valuable measure of confidence.

IV. Machine Learning and Optimization

Because our systems were simple we were able to turn the complicated task of "learning" into one of optimization and self adaptation in a parameter space. We did this in a number of areas including finding optimal weights for each feature in the penstroke metric and finding a confidence metric that is slightly improved over the ratio metric.

A. Adapting Feature Weights for Classification

The adaptation of weights in the penstroke metric was accomplished using a gradient descent technique. At each iteration, a parameter was adjusted, and the resulting performance of the system was measured by performing classifications on a selected subset of the training database. If the performance improved, the new value was kept. Several passes of scaling each parameter up and down

by successively smaller amounts were performed until the system reached a stable optimum.

Though this was a computationally expensive approach, in that a classification run had to be made to test each new set of parameters, it could be accomplished in a reasonable amount of time in parallel on the Connection Machine and resulted in a significant performance increase over either uniform or hand coded values. When the feature weight optimization was run on a subset of 270 particularly difficult examples the error rate was decreased from 42% to 27%. Similar work has been performed on optimization of feature weights for k-NN classification systems for both classical datasets [14], [15] and free text [16].

B. Optimizing a Confidence Metric

Since the ratio confidence metric outperformed the k-NN confidence metric it was hypothesized that including even further information about the nearness of other different class neighbors might be even better. To test this, an experiment was performed on the penstroke metric to optimize the contributions of the k-NN distance and the ratios of the nearest five examples of the same as well as different classes. We used a confidence measure comprised of the following terms:

$$1 - wD_1/K - \sum_{i=1}^{5} x_i D_1/\overline{D_i} + \sum_{i=2}^{6} y_i (D_1/D_i) \qquad (5)$$

where w, x_i, and y_i are weights corresponding to the following confidence components:

1) k-NN confidence described in Section III-B;
2) ratios of the distance of the nearest neighbor to the other five nearest neighbors of a different class.
3) ratios of the distance of the nearest neighbor to the other five nearest neighbors of the same class.

Initially each of these elements of the total confidence was assigned an equivalent weight. Using gradient descent, the following sets of weights were determined: $w = 0.x_1 = 1.0.x_i (i = 2, \cdots, 5)$ and $y_i (i = 2, \cdots, 6)$ were all approximately 0.15.

The experiment confirms ours previous results that the k-NN distance is not a useful component of the confidence metric, while the ratio measure, $D_1/\overline{D}_1$, is particularly useful. While the incorporation of the additional nonzero weighted terms improved performance slightly, it is possible that this second order effect is particular to our training database.

V. Conclusion

Our initial hope for this research was to show that acceptable classification performance could be achieved through large training databases, k-NN classification and the simple optimization of parameterized feature spaces. What we have found is that trivial distance metrics (Hamming distance) for k-NN classification and a large database alone provide high performance in the domain of handwritten digit recognition. Beyond that, we have shown that there is good reason to believe that performance will continue to improve as the training database grows even larger.

In some ways, this is an obvious result. If the database is large enough it will eventually saturate the space of all possible bitmaps and the system could only fall short of perfect performance due to errors or noise in the training database. What is remarkable is that such high performance is achieved not with the example database required to saturate the search space, but rather with less than 225,000 examples. This result suggests, at least in this domain, that researchers might better spend their time collecting data than writing code.

With that said it is also clear that further increases in performance can be had by improved distance metrics (pixel distance and penstroke) and by adapting feature weights and confidence metrics to the task at hand. Fortunately the latter can be done automatically.

Acknowledgment

The authors would like to recognize the previous, unpublished work on character recognition by the Thinking Machines Classification project. This group included: M. Drumheller, D. Fritzsche, J. Hutchinson, B. Kahle, P. Oppenheimer, T. Poggio, and L. Tucker.

References

[1] D. Aha, D. Kibler, and M. Albert, "Instance-based learning algorithms," *Machine Learning*, vol. 6, pp. 37–66, 1991.
[2] P. Ahmed and C. Y. Suen, "Computer recognitions of totally unconstrained handwritten zip codes," *Int. J.Pattern Recognition and Artificial Intell.*, 1, 1987.
[3] C. Atkeson, "Roles of knowledge in motor learning," MIT AI Lab Tech. Rep. 942, 1986.
[4] G. Borgefors, T. Hartmann, and S. L. Tanimoto, "Parallel distance transforms on pyramid machines: Theory and implementation," *Signal Processing*, vol. 21, no. 1, pp. 61–86, 1990.
[5] D. J. Burr, "Elastic matching of line drawings," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-3, no. 6, pp. 708–713, 1981.
[6] K. A. Church, "A stochastic parts program and noun phrase parser for unrestricted text," unpublished manuscript, AT&T Bell Labs, Murray Hill, NJ, 1986.
[7] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inform. Theory*, vol. 13, pp. 21–27, 1967.
[8] R. Creecy, B. Masand, S. Smith, and D. Waltz, "Trading mips and memory for knowledge engineering," *Commun. ACM*, vol. 35, no. 8, pp. 48–64, Aug. 1992.
[9] B. Dasrathy, Ed., *Nearest Neighbor Pattern Classification*. Los Alamitos, CA: IEEE Computer Society Press, 1990.
[10] P. Danielsson, "Euclidean distance mapping," *Computer Graphics and Image Processing*, vol. 14, pp. 227–248, 1980.
[11] G. Hinton, C. Williams, and M. Revow, "Adaptive elastic models for handprinted character recognition," *Advances in Neural Information Processing Systems 4*, J. Moody, S. Hanson, and R. Lippmann, Eds. San Mateo, CA: Morgan Kauffmann, 1992.
[12] M. D. Garris and R. A. Wilkinson, "NIST special database 3. Handwritten segmented characters," *NIST*, Gaithersburg, MD.
[13] S. Kahan, T. Pavlidis, and H. Baird, "On the recognition of printed characters of any font and size," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-9, no. 2, pp. 274–287, 1987.
[14] J. Kelly and L. Davis, "A hybrid genetic algorithm for classification," in *Proc. Twelfth Int. Joint Conf. on Artificial Intell.*, Sydney, Australia, Aug. 1991.
[15] ——, "Hybridizing the genetic algorithm and the K nearest neighbors classification algorithm," in *Proc. Fourth Int. Conference on Genetic Algorithms*, San Diego, CA, July, 1991.
[16] B. Masand, "Effects of query and database sizes on classification of news stories using memory based reasoning," presented at the *AAAI Spring Symposium on Case Based Reasoning*, Palo Alto, CA, Apr. 1993.
[17] T. Pavlidis, *Algorithms for Graphics and Image Processing*. Rockville, MD: Computer Science Press, 1982.
[18] A. Rosenfeld and J. L. Pfaltz. "Distance functions on digital pictures," *Pattern Recognition*, vol. 1, pp. 33–61, 1968.
[19] S. Smith, "A handwritten character recognition system for the connection machine CM-2 supercomputer," in *Supercomputing Symp. '92*, Ottawa, Canada, 1992, pp. 377–389.
[20] C. Stanfill and D. Waltz, "Toward memory based reasoning," *Commun. ACM*, vol. 29, no. 12, pp. 1213–1228, 1986.
[21] B. Widrow, "The 'Rubber-mask' technique I. Pattern measurement and analysis," *Pattern Recognit.*, vol. 5, pp. 175–197, 1973.
[22] R. A. Wilkinson, J. Geist, S. Janet, P. Grother, C. Burges, R. Creecy, B. Hammond, J. Hull, N. Larsen, T. Vogl, C. Wilson, "The first census optical character recognition system conference," Nat. Inst. of Standards and Technol. Tech. Rep. #NISTIR 4912, Gaithersburg, MD, Aug. 1992.