Joint Induction of Shape Features and Tree Classifiers

Donald Geman * Yali Amit † Kenneth Wilder ‡

May 1996

Abstract

We introduce a very large family of binary features for two-dimensional shapes. The salient ones for separating particular shapes are determined by inductive learning during the construction of classification trees. There is a feature for every possible geometric arrangement of local topographic codes. The arrangements express coarse constraints on relative angles and distances among the code locations and are nearly invariant to substantial affine and non-linear deformations. They are also partially ordered, which makes it possible to narrow the search for informative ones at each node of the tree. Different trees correspond to different aspects of shape. They are statistically weakly dependent due to randomization and are aggregated in a simple way. Adapting the algorithm to a shape family is then fully automatic once training samples are provided. As an illustration, we classify handwritten digits from the NIST database; the error rate is .7%.

*Department of Mathematics and Statistics, University of Massachusetts, Amherst, MA 01003; Email:geman@math.umass.edu. Supported in part by the NSF under grant DMS-9217655, ONR under contract N00014-91-J-1021, and ARPA contract MDA972-93-1-0012.

†Department of Statistics, University of Chicago, Chicago, IL, 60637; Email: amit@galton.uchicago.edu. Supported in part by the ARO under grant DAAH04-96-1-0061.

‡Department of Mathematics and Statistics, University of Massachusetts, Amherst, MA 01003; Email: wilder@math.umass.edu

1 Introduction

We re-visit the problem of finding good features for separating two-dimensional shape classes in the context of invariance and inductive learning. The feature set we consider is virtually infinite. The salient ones for a particular shape family are determined from training samples during the construction of classification trees. We experiment with isolated handwritten digits. Off-line recognition has attracted enormous attention, including a competition sponsored by the National Institute of Standards and Technology (NIST) ([1]), and there is still no solution that matches human performance. Many approaches today are based on non-parametric statistical methods such as neural networks ([2],[3]), discriminant analysis ([4],[5]), nearest-neighbor rules with different metrics ([6], [7],[8]), and classification trees ([9],[10]). Hybrid and multiple classifiers are also effective ([11],[12]). In many cases the feature vector does not explicitly address "shape."

Our features are shape-based and bear a resemblance to the geometric invariants which have been proposed for recognizing rigid shapes and three-dimensional objects. (See [13] for a review and bibliography.) Typically this involves extracting boundary information, computing tangents and identifying distinguished points, such as those of high curvature or inflections. Invariant geometric relations are then determined among these special points; see e.g.,[14],[15], [16]. Many authors report much better results with these and "structural features" than with normalized bit maps.

Our features also involve geometric relations among points, but not distinguished points. Instead, the pixels are coded in a very coarse manner based on the image topography in their immediate neighborhood. We refer to these codes as "tags." They are much too common and primitive to be informative about shape. The features we use are arrangements of tags involving relative angles and distances. Although not strictly invariant to affine transformations, they are "semi-invariant" in a sense that will be made precise in §3. On the other hand, making these two compromises - giving up discriminating points and strict invariance - allows us to define a very large family of features, which has a natural partial ordering corresponding to increasing detail and structure. In addition, the semi-invariance extends to significant non-linear deformations.

The most informative features for separating particular shape classes are singled out

by recursively partitioning training data using the standard splitting criterion of entropy reduction. The partial ordering makes this computationally feasible. The choice of features used by the classifier is part of the training process and there is no dedicated modeling. As a result, the method is very portable; for example, it has been applied to recognizing deformed LaTeX symbols (293 classes) [17] and rigid three-dimensional objects [18].

The classifier is constructed from multiple classification trees. Randomization prevents the same features from being chosen from tree to tree and guarantees weak dependence. These statistical issues are analyzed in [17] and [19], together with semi-invariance, generalization and the bias/variance tradeoff. The purpose here is to introduce the features, outline the algorithm and experiment with handwritten digits. The training and test sets are taken from the "NIST Special Database 3" [20]. In terms of speed and accuracy we achieve results which are comparable to the best of those reported elsewhere.

2 Tags

The first step in the algorithm involves assigning each pixel in the image one or more "tags" characterizing the local topography of the intensity surface in its neighborhood. Instead of manually characterizing the local configurations of interest, for example trying to define local operators to identify gradients, we adopt an information-theoretic approach and "code" a micro-world of sub-images through tree-structured vector quantization. However, other types of local primitives might work equally well; for example, our tags are similar to oriented edge fragments. The discriminating power derives from spatial relationships among the primitives rather than from the primitives themselves.

A large sample of 4×4 sub-images is randomly extracted from the training data. The corresponding shape classes are irrelevant and not retained. This family of training sub-images is then clustered by growing a decision tree. At each node there are 16 possible "questions": *"Is site* (i, j) *black?"* for $i, j = 1, 2, 3, 4$. The question chosen is the one which divides the the sub-images at the node as equally as possible into two groups. There is a tag type for each node of the resulting tree, except for the root. Thus, there are $2+4+8 = 14$ tags for a depth 3 tag tree and 62 tags for a depth 5 tag tree, the one we use in our experiments.

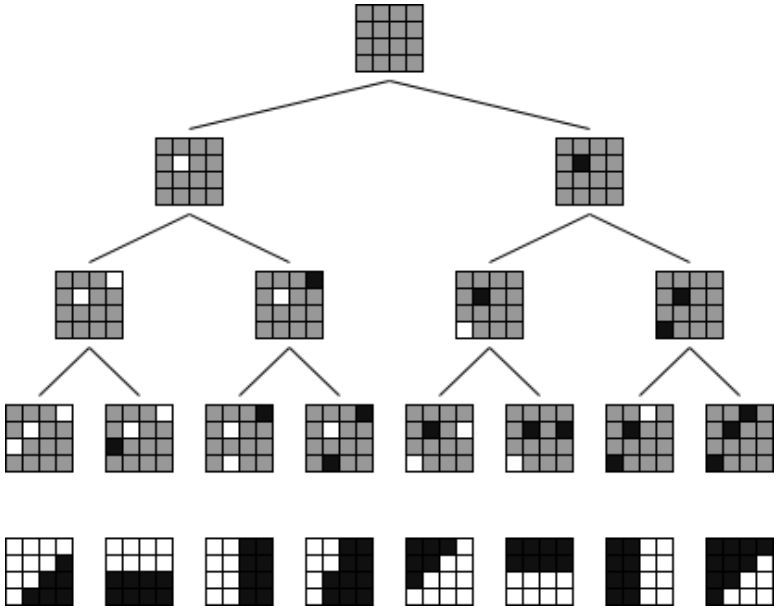

Figure 1: First three tag levels with most common configurations.

Depth 5 tags provide the most detailed descriptions of the topography. Observe that the tag corresponding to an internal node represents the union of those in the subtree rooted at that node. Each pixel in an image is then assigned all the tags encountered as the 4×4 sub-image which contains that pixel in the upper left-hand corner proceeds down the tag tree.

For efficiency, the population is restricted to sub-images containing at least one black and one white site within the center four. Obviously this concentrates the processing in the neighborhood of boundaries. In the grey-level context it is useful to consider more general tags, allowing variations on the concept of local homogeneity [18] and other local attributes of the intensity surface.

The first three levels of the tag tree constructed from the NIST data are shown in Figure 1. We also display the most common configuration found at each of the eight depth 3 nodes. Note that there are many other 4×4 configurations consistent with these nodes. In Figure 2 we show all instances of four of the depth 3 tags in four images of a particular digit as well as all instances of four of the depth 5 tags in the second row; the depth 5 tags are refinements of the depth 3 tags. Notice that each instance of a depth 5 tag is also an instance of its

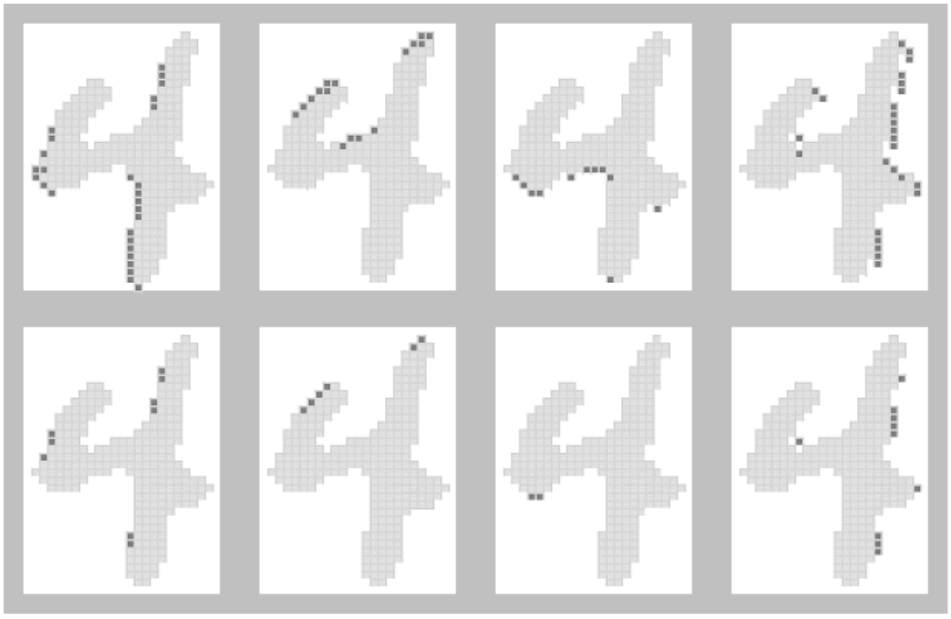

Figure 2: Top: All instances of four depth 3 tags. Bottom: All instances of four depth 5 tags.

ancestor depth 3 tag.

The fixed size neighborhood conveys more or less the same information in a certain range of resolutions relative to the shapes under consideration. In our experience this range is roughly 10×10 to 70×70. The more uniform the resolution across training and test sets the better the ultimate performance of the classifier. A multi-resolution approach has been investigated in the context of grey-level images and 3D objects in [18].

3 Features: Tag Arrangements

The shape features involve geometric arrangements of the tags defined in terms of the angles of the vectors connecting their locations. There is one feature for each possible tag arrangement. An arrangement may contain any number of tags and any subset of all possible pairwise relations among these tags. More formally, then, a tag arrangement is an attributed graph; the vertices of the graph are labeled by the tag types and the edges by the angles (see below) between the two corresponding vertices. Each such arrangement is either present or

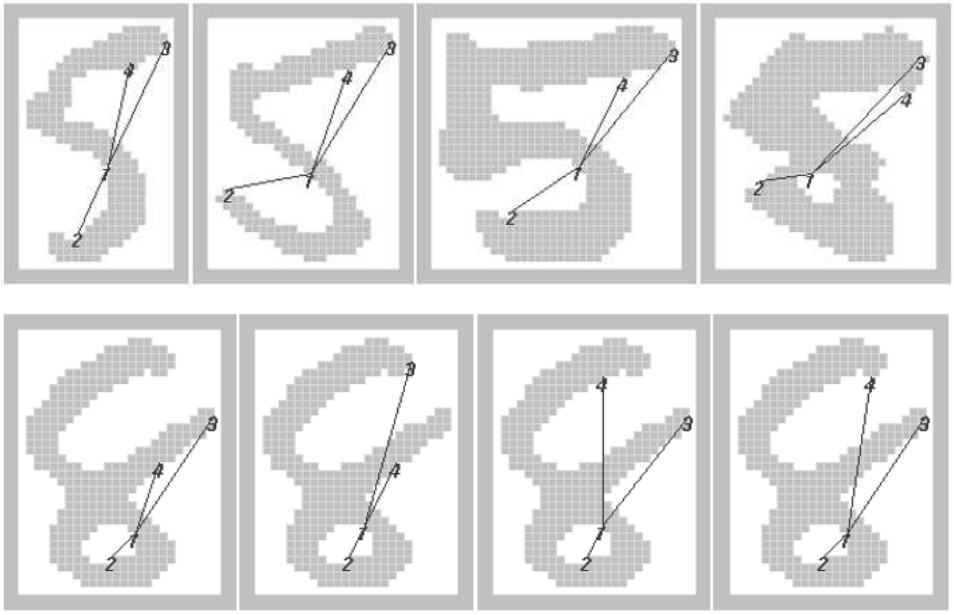

Figure 3: Top row: Instances of a geometric arrangement in several 5's. Bottom row: Several instances of the geometric arrangement in one 8.

not present in the image. The features are therefore binary. "Present in the image" means there is at least one set of tags of the prescribed type in the image whose locations satisfy the indicated relationships.

There are eight possible relations between any two locations u and v corresponding to the eight compass headings "north," "northeast," "east," etc. The two points satisfy relation k ($k = 1, ..., 8$) if the angle of the vector $u - v$ is within $\pi/4$ of $k * \pi/4$. Figure 3 shows several digits which contain a specific geometric arrangement of four tags. Since an arrangement involves only relative angles and no fixed locations, it may appear many times in a given image as illustrated in the bottom row of Figure 3.

It is clear from the description of these geometric relations that even in the continuum they are invariant only to "neighborhoods" in the space of affine transformations (in particular rotation, skew and shear) as opposed to algebraic invariants ([13]), which are truly invariant in the continuum. On the other hand, since the tag arrangements are defined in terms of coarse tolerances on the angles, they are also invariant to substantial nonlinear deformations and very robust with respect to discretization noise and other forms of

degradation.

There is a partial ordering of the arrangements under which a specific arrangement precedes any of its extensions involving additional tags and/or relations. The partial ordering corresponds to a hierarchy of structure. Small arrangements with few tags produce coarse splits of shape space. As the arrangements increase in size (say the number of tags plus relations), they contain more and more information about the images which contain them.

An important property of this ordering is the following. Given an arrangement, define a *minimal extension* of it as one which contains that arrangement together with either one additional tag in relation to an existing one or a new relation between two existing tags. Then given that two images of the *same class* share a common feature (i.e., the same arrangement is present in both images), and given a minimal extension of that feature, it is very likely that either both images contain the extension or neither image does. This property, which we call *semi-invariance*, is discussed in more detail in [17]. On the other hand, given two images of *different* classes which share a common feature, there are relatively many more minimal extensions which split these. The goal of course is to find a minimal extension which keeps most images of the same class together while at the same time separates the dominating classes from one another.

Metric relations can also be employed, for example ratios of distances between two pairs of tags. In order to make the feature binary one employs a quantization of the range of ratios, much like the quantization of the range of angles described above. Thus, for example, $|u-v| < |u-w|$ is a ternary relation among three locations u, v, w. Indeed it is conceivable that the algebraic functions used to define pure invariants could be similarly quantized to provide additional binary features. We have not yet explored this possibility.

The number of features described here is clearly unlimited. Even if the size of the arrangements is limited in advance (say to at most twenty tags), the entire family still cannot be computed for a collection of training data and subsequently used in conjunction with a standard classifier. Moreover most of these features may lack any discriminating power for the particular problem at hand. How can this source of information be accessed efficiently? One very natural way is recursive partitioning based on the partial ordering.

4 Recursive Partitioning of Shape Space

A tree is built as follows. At the root loop through only the simplest arrangements involving two tags and a relation. Each one splits the data into two subsets: those with the arrangement and those without it. Now compute the reduction in mean uncertainty resulting from each split. Uncertainty is measured by Shannon entropy and estimated using the training data - the standard procedure in pattern recognition and machine learning. Choose that arrangement which yields the best split and denote the chosen feature by $\mathbf{A_0}$. Those data points for which $\mathbf{A}_0$ is not present are in the "no" child node. To split this node search again through arrangements involving two tags and a relation. Those data points for which $\mathbf{A}_0$ is present are in the "yes" child node and have one or more instances of $\mathbf{A}_0$, the *pending arrangement*. Now search among minimal extensions of $\mathbf{A}_0$ and choose the one which leads to the greatest reduction in uncertainty about class given the existence of $\mathbf{A}_0$. Note that only a very small fraction of the total number of features (splits) is considered at a node, namely those which minimally extend the pending feature. Note also that an extension $\mathbf{A_1}$ of $\mathbf{A_0}$ is considered present in an image if *any* one of the instances of $\mathbf{A_0}$ can be extended to satisfy the relations defined by $\mathbf{A_1}$.

Given we are at a node t in the tree, the *pending arrangement*, say $\mathbf{A}_p$, is the largest arrangement found along the path from the root to t; it is of course the same for all data points in the node t. Note that not every step along this path involves an additional tag or relation because some of the nodes might be "no" children. The arrangement $\mathbf{A}_t$ chosen to split node t minimizes the mean entropy on class among minimal extensions of $\mathbf{A}_p$. Continue in this fashion until a stopping criterion is satisfied, e.g., the number of data points falls below a threshold. The digits in Figure 3 were taken from a depth 4 node of a tree; the history of the node is "yes,yes,no,yes" accounting for the existence of four tags in the pending arrangement.

Figure 4 depicts how the data is split at a node of a tree, again grown as part of our experiments on handwritten digit classification. The eight images are representative of those at the node. The four images on the left answer "no" to a query involving the existence of a fifth tag in a relationship to the fourth tag. The four images on the right answer "yes." In each case an instance of the pending arrangement is shown.

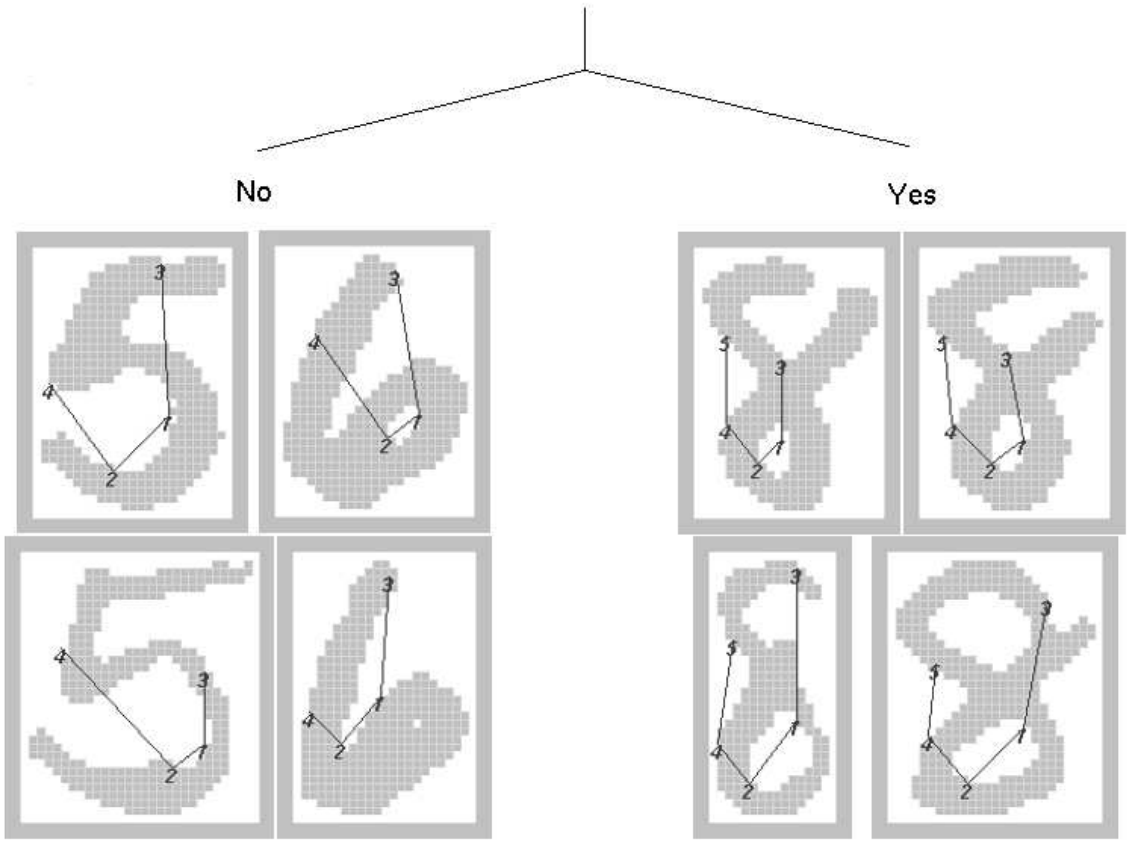

Figure 4: Example of node-splitting in a typical digit tree; the query involves adding a fifth tag (vertex) to the pending arrangement. Specifically, the proposed arrangement adds a fifth vertex and a fourth relation to the existing graph which has four vertices and three relations.

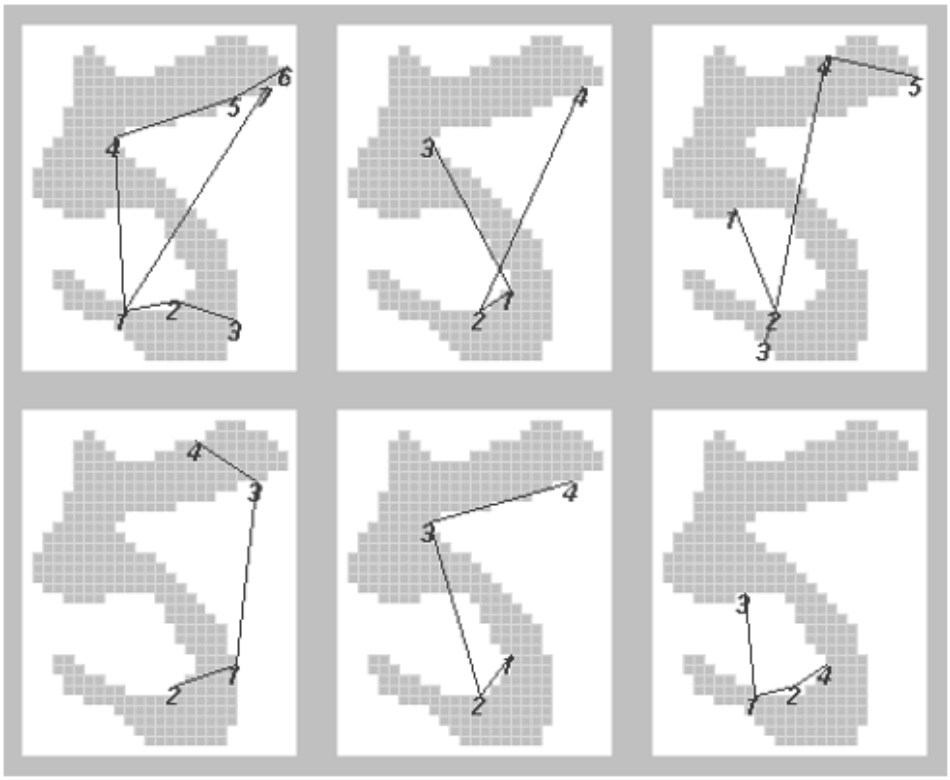

Figure 5: Arrangements found in an image at terminal nodes of six different trees.

5 Multiple Randomized Trees

Despite the fact that only minimal extensions of the pending arrangement are entertained at each node, the procedure above is still not practical due to the number of candidates. At the root node there are $62 \times 62 \times 8 = 30752$ arrangements. The number of minimal extensions in the internal nodes is also very large. The solution is simple: Instead of searching among *all* the admissible arrangements at each node, we restrict the search to a *small random subset* and choose the one among these which yields the best split.

Because the set of tag arrangements is so large, and because different ones address different aspects of shape, separate trees provide separate structural descriptions, characterizing the shapes from different "points of view". This is visually illustrated in Figure 5 where the same image is shown with an instance of the pending arrangement at the terminal node for six different trees.

Consequently, aggregating the information provided by a family of trees should yield more accurate and more robust classification. There are many ways this can be done, most of which are impractical due to the limited amount of training data. We have chosen a simple method of aggregation which has proven successful.

Each tree may be regarded as a discrete random variable T on the space of images $\mathbf{X}$. Each terminal node corresponds to a different value of T. Let $T_1, ..., T_N$ be the set of trees. For each terminal node of each tree we compute the empirical distribution on the classes $\{0, ..., 9\}$ based on the training data which reaches that node. Now given an image $\mathbf{x}$ and a tree T_n, let

$$\mu_n(\mathbf{x}) = (\mu_n(\mathbf{x}, 0), ..., \mu_n(\mathbf{x}, 9))$$

be the distribution stored at the leaf of T_n reached by $\mathbf{x}$. In addition, let

$$\bar{\mu}(\mathbf{x}) = \frac{1}{N} \sum_{n=1}^{N} \mu_n(\mathbf{x}).$$

In order to classify an image $\mathbf{x}$ in the test set we simply compute $\bar{\mu}(\mathbf{x})$ and take the mode of this distribution as the estimated class. Our first experiments with such multiple, randomized trees were reported in [21] and there is a statistical analysis of the dependence structure among the trees in [17]. Alternative ways of producing and aggregating multiple trees are proposed in [22],[23], [24],[25] and [19].

Various rejection criteria can also be defined in terms of $\bar{\mu}$. For example an image $\mathbf{x}$ is classified only if the value at the mode of $\bar{\mu}(\mathbf{x})$ exceeds some threshold or exceeds some multiple of the value at the second mode.

6 Handwritten Digit Recognition

Although classification rates for handwritten digit recognition are inching up, it is somewhat difficult to assess the state-of-the-art. One reason is that many different "test sets" have been proposed and some are widely considered to be more difficult than others ([2],[8]). At the NIST competition, the best recognition rate at zero percent rejection was 98.44%, but the next best was 96.84%, and only about half the systems had error rates of less than five percent; see [1]. The test set ("NIST Test Data 1") was considered quite difficult

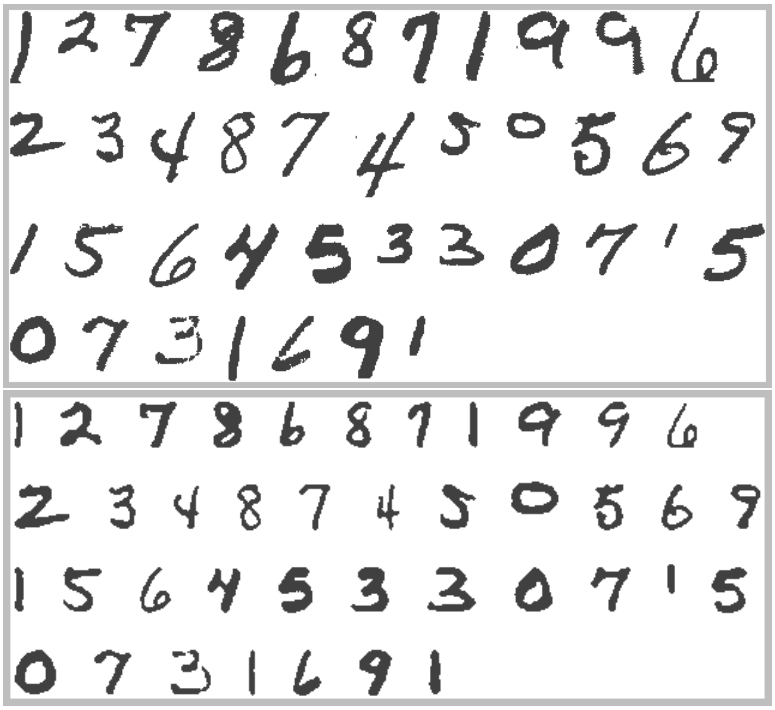

Figure 6: Random sample of test images before and after pre-processing.

and recognition rates reported elsewhere, for instance on portions of the NIST training set ("NIST Special Database 3" [20]) or on the USPS test set [26], are generally higher. For example, using a nearest-neighbor system, the study in [8] achieves 98.61% with 100,000 training points on the NIST training database. The best reported rates seem to be those obtained by the ATT research group, up to 99.3% by training and testing on composites of the NIST training and test sets; see [2].

Our experiments are based on portions of the NIST Special Database 3 which consists of approximately 223,000 binary images of isolated digits written by more than 2000 writers. The images vary widely in dimensions, ranging from about twenty to one hundred rows, and also vary in stroke thickness and other attributes. We used 100,000 digits for training and another 50,000 for testing. There is no overlap of writers. A random sample from the test set is shown in Figure 6 (top).

The studies mentioned above utilize pre-processing, such as thinning, slant correction and size normalization. Many also utilize post-processing; for example, in [27] it is shown how to use additional training data to make a second classifier more or less dedicated to the mistakes

and marginal decisions of the original classifier; see also [2]. This procedure ("boosting") can then be iterated. Unfortunately, this requires very large training sets and/or the ability to create artificial data. In order to compare our method with those cited above, we did several experiments using only pre-processing. Specifically, the images in the training set were converted to a "reference pose" by slant-correcting and a crude form of scaling. The "slant" of an image is taken as the slope of the (regression) line fitted to the set of stroke pixels using least-squares; the slant is "corrected" by then applying a linear transformation to each row to bring the regression line to a vertical orientation. Images with fewer than 32 rows were not scaled; the others were down-sampled to exactly 32 rows by blurring and sub-sampling on a regular lattice, thereby preserving the aspect ratio of the original. As is well-known, there is virtually no loss of information at this resolution for character images. In Figure 6 we show the result of slant-correcting and scaling.

The best error rate we achieved with a *single* tree was about 7%. In contrast to standard recursive partitioning, we did not grow a deep tree and "prune back" ([28]), which is an approach designed to avoid "over-fitting." In the context of multiple trees this problem appears to be of less importance. We stop splitting when the number of data points in the second largest class falls below ten.

Twenty-five trees $T_1, ..., T_{25}$ were made. The depth of the terminal nodes (i.e., number of questions asked per tree) varies widely, the average over trees being 8.8; the maximum depth is 20. The average number of terminal nodes is about 600. We pre-processed (scaled and slant-corrected) the test set in the same manner as the training set. For a single tree T_n we classify an image $\mathbf{x}$ based on the mode of $\mu_n(\mathbf{x})$, the probability distribution at the terminal node where $\mathbf{x}$ lands. The average classification rate per tree is about 91%.

By aggregating $T_1, ..., T_{25}$, which means classifying based on the mode of the *aggregate* distribution $\bar{\mu}(\mathbf{x})$, the classification rate climbs to 99.2% (with no rejection). In Figure 7 we show a random sample of the digits we classify incorrectly. The classification rate as a function of the rejection rate is also of interest. Rejection is based on the ratio α between the mode and the next highest class in the aggregate distribution. For example, the classification rate is 99.5% with one percent rejection and 99.8% with three percent rejection. Finally, doubling the number of trees makes the classification rates $99.3\%, 99.6\%$ and 99.8% at zero,

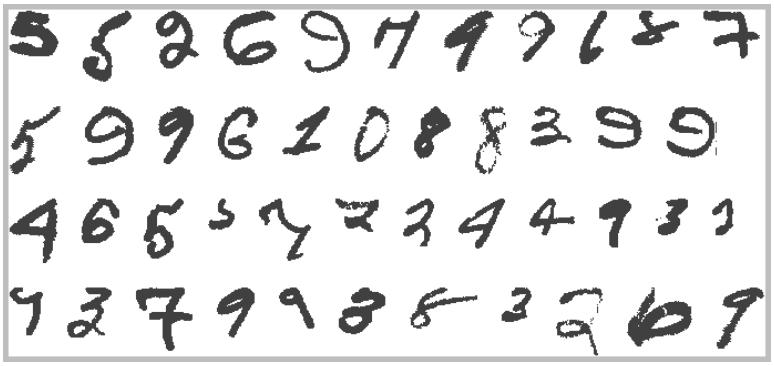

Figure 7: Random sample of incorrectly classified digits.

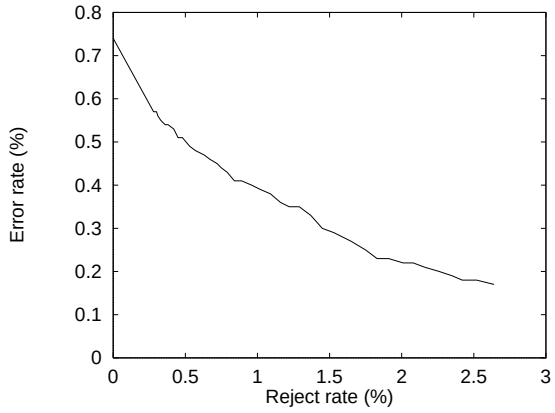

Figure 8: Error rate vs. Reject rate

one and two percent rejection, respectively; the error rate vs. rejection rate curve is shown in Figure 8.

We performed a second experiment in which the test data was not pre-processed in the manner of the training data; in fact, the test images were classified without utilizing the pixel dimensions of the digits. (In particular, the size of the bounding box is irrelevant.) Instead, each test image was classified with the same set of trees at two resolutions (original and halved) and three (fixed) slants. The highest of the resulting six modes determines the classification. (This might be viewed as a very crude version of "active testing," in which a discrete set of "poses" is searched for a good match with a reference pose.) The classification rate was 98.9%.

There are other properties of classifiers besides accuracy which are important, such as

training time, storage requirements, and computational complexity. Nearest-neighbor classifiers require no training but considerable memory and are relatively slow; neural networks are slow to train but fast on line. Our classifier is relatively easy to train, requiring about $25MB$ of memory during training, due primarily to storing the tags for all the images and storing the instances of the pending arrangements. The memory requirements for testing are negligible.

The key computational component in classification is "instance checking." On the average a digit will have about ten instances of each tag and ten instances of the pending tag arrangement at each node. Therefore, at each node, we perform about 100 relation checks in order to determine whether or not a minimal extension involving a particular tag and angle relation is present. Aggregating this over twenty-five trees, and taking the average tree depth as ten, leads to approximately $25,000$ "checks" per digit. Each of these involves determining an arctangent from a pre-computed table in order to check an angle condition between two tag locations. (It should be noted that two instances of a tag arrangement are declared the same if all the corresponding vertices have approximately the same image location. However, the computation for "instance clustering" is small compared with checking.) On a typical workstation one can then classify at least $15 - 50$ digits per second without special efforts to optimize the code; the time is approximately equally divided between extracting the tags from an image and sending it down the trees.

Finally, additional experiments are reported in [19]. Some of these utilize other training and test sets, such as the USPS database and a composite of NIST Special Database 3 and NIST Test Data 1; the results are comparable. Other experiments involve a post-processing step applied to the reject set, utilizing the full aggregated distribution $\bar{\mu}(\mathbf{x})$ in a nearest-neighbor context. Since this ten-dimensional output vector is more informative than the mode alone, and since the reject set is small, the classification rate can be improved to 99.5% with almost no increase in the average amount of computation per digit.

7 Conclusion

We have reported our progress on classifying handwritten digits using a new set of features as candidates for splitting rules in recursive partitioning. The first step is transforming the image by replacing pixel values by five bit codes which characterize the local topography in the vicinity of boundaries. In contrast to work on distinguished point features, the codes are primitive and redundant; in particular, the points themselves cannot disambiguate among shapes. All the discriminating power derives from spatial relationships among the codes. There is a binary feature for each such arrangement.

It is well-know that decision trees offer a powerful mechanism for feature selection but the standard method for constructing trees is not practical because our feature set is virtually infinite. The remedy is to utilize the natural partial ordering on arrangements together with randomization, which has the added benefit of reducing the statistical dependence from tree to tree.

There is one general algorithm for tree construction based on the entire feature set. The particular subset of features used is problem-dependent and determined inductively from training samples. There is no explicit modeling. We apply it to a training set of handwritten digits but nothing would change if the images contained other shapes, or hundreds of shape classes. Perhaps the main reason is the property of semi-invariance to significant linear and nonlinear deformations, which is a generic attribute of the entire feature set. Features which separate two classes based on a small number of samples will also separate new samples.

The speed and error rates we have achieved on the NIST database are about the same as those for advanced neural networks (without special hardware). The advantages here are simplicity, automation and portability, which is due to built-in invariance and the rich world of spatial relationships. We are currently extending the method to other visual recognition problems involving grey-level images and structured backgrounds.

Acknowledgment. We are grateful to George Nagy for sharing his insights about decision trees and the OCR problem, and for valuable advice on the presentation of our work.

References

[1] R. A. Wilkinson, J. Geist, S. Janet, P. Grother, C. Gurges, R. Creecy, B. Hammond, J. Hull, N. Larsen, T. Vogl, , and C. Wilson, "The first census optical character recognition system conference," Tech. Rep. NISTIR 4912, Nat. Inst. of Standards and Technol., Gaithersburg, MD, 1992.

[2] L. Bottou, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, L. D. Jackel, Y. LeCun, U. A. Muller, E. Sackinger, P. Simard, and V. Vapnik, "Comparison of classifier methods: a case study in handwritten digit recognition," in *Proc. IEEE Inter. Conf. on Pattern Recognition*, pp. 77–82, 1994.

[3] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network," in *Advances in Neural Information* (D. S. Touresky, ed.), vol. 2, Denver: Morgan Kauffman, 1990.

[4] B. Boser, I. Guyon, and I. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of COLT II Philadelphia, Pa.*, 1992.

[5] T. Hastie, A. Buja, and R. Tibshirani, "Penalized discriminant analysis," *Annals of Statistics*, vol. 23, pp. 73–103, 1995.

[6] Z. M. Kovacs-V and R. Guerrieri, "Massively- parallel handwritten character recognition based on the distance transform," *Pattern Recognition*, vol. 28, pp. 293–301, 1995.

[7] P. Y. Simard, Y. L. LeCun, and J. S. Denker, "Memory-based character recognition using a transformation invariant metric," in *IEEE Inter. Conf. on Pattern Recog.*, pp. 262–267, 1994.

[8] S. Smith, M. O. Bourgoin, K. Sims, and H. L. Voorhees, "Handwritten character classification using nearest neighbor in large databases," *IEEE Trans. PAMI*, vol. 16, pp. 915–919, 1994.

[9] S. Shlien, "Multiple binary decision tree classifiers," *Pattern Recognition*, vol. 23, pp. 757–763, 1990.

[10] Q. R. Wang and C. Y. Suen, "Analysis and design of a decision tree based on entropy reduction and its application to large character set recognition," *IEEE Trans. PAMI*, vol. 6, pp. 406–417, 1984.

[11] T. K. Ho, J. J. Hull, and S. N. Srihari, "Decision combination in multiple classifier systems," *IEEE Trans. PAMI*, vol. 16, pp. 66–75, 1994.

[12] T. S. Huang and C. Y. Suen, "Combination of multiple experts for the recognition of unconstrained handwritten numerals," *IEEE Trans. PAMI*, vol. 17, pp. 90–94, 1995.

[13] T. H. Reiss, *Recognizing Planar Objects Using Invariant Image Features*. Lecture Notes in Computer Science no. 676, Berlin: Springer Verlag, 1993.

[14] J. L. Mundy and A. Zisserman, *Geometric Invariance in Computer Vision*. Cambridge: MIT Press, 1992.

[15] M. Sabourin and A. Mitiche, "Optical character recognition by a neural network," *Neural Networks*, vol. 5, pp. 843–852, 1992.

[16] K. Cho and S. M. Dunn, "Learning shape classes," *IEEE Trans. PAMI*, vol. 16, pp. 882–888, 1994.

[17] Y. Amit and D. Geman, "Shape quantization and recognition with randomized trees," *Neural Computation*, 1997.

[18] B. Jedynak and F. Fleuret, "3d object recognition from geometric queries," in *Proc. Image'Com 96*, (Bordeaux, France), 1996.

[19] K. J. Wilder, *Decision tree algorithms for handwritten digit recognition*. PhD thesis, Department of Mathematics and Statistics, University of Massachusetts, Amherst, MA., 1997.

[20] M. D. Garris and R. A. Wilkinson, *NIST special database 3. Handwritten segmented characters*. NIST, Gaithersburg, MD.

[21] Y. Amit and D. Geman, "Randomized inquiries about shape; an application to handwritten digit recognition," Tech. Rep. 401, Department of Statistics, University of Chicago, 1994.

[22] L. Breiman, "Bagging predictors," Tech. Rep. 421, Department of Statistics, University of California, Berkeley, 1994.

[23] S. W. Kwok and C. Carter, "Multiple decision trees," in *Uncertainty and Artificial Intelligence* (R. D. Shachter, T. S. Levitt, L. Kanal, and J. F. Lemmer, eds.), North-Holland, Amsterdam: Elsevier Science Publishers, 1990,.

[24] J. J. Oliver and D. Hand, "Averaging over decision stumps," in *Proc. ECML-1994* (F. Bergadano and L. De Raedt, eds.), (Berlin), Springer Verlag, 1994.

[25] T. G. Dietterich and G. Bakiri, "Solving multiclass learning problems via error-correcting output codes," *J. Artificial Intell. Res.*, vol. 2, pp. 263–286, 1995.

[26] J. J. Hull, "A database for handwritten text recognition research," *IEEE Trans. PAMI*, vol. 16, pp. 550–554, 1994.

[27] H. Drucker, R. Schapire, and P. Simard, "Boosting performance in neural networks," *Int. J. of Pattern Recog. and A.I.*, vol. 7, pp. 705–719, 1993.

[28] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees.* Belmont, CA.: Wadsworth, 1984.