Randomized Inquiries About Shape; an Application to Handwritten Digit Recognition

Yali Amit * and Donald Geman †

November, 1994

TECHNICAL REPORT NUMBER 401
Department of Statistics
University of Chicago
Chicago, IL 60637

*Department of Statistics, University of Chicago, Chicago, IL, 60637; Email: amit@galton.uchicago.edu.
Supported in part by the ARO under grant DAAL-03-92-G-0322.

†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.

Abstract

We describe an approach to shape recognition based on asking relational questions about the arrangement of landmarks, basically localized and oriented boundary segments. The questions are grouped into highly structured inquiries in the form of a tree. There are, in fact, many trees, each constructed from training data based on entropy reduction. The outcome of each tree is not a classification but rather a distribution over shape classes. The final classification is based on an aggregate distribution.

The framework is non-Euclidean and there is no feature vector in the standard sense. Instead, the representation of the image data is graphical and each question is associated with a labeled subgraph. The ordering of the questions is highly constrained in order to maintain computational feasibility, and dependence among the trees is reduced by randomly sub-sampling from the available pool of questions.

Experiments are reported on the recognition of handwritten digits. Although the amount of training data is modest by today's standards, the rates we achieve are competitive with those reported elsewhere using neural network, nearest-neighbor, and other nonparametric classifiers.

Keywords. Shape recognition, handwritten digits, decision trees, stochastic indexing, computational learning.

1 Introduction

We explore a new approach for recognizing shapes by asking many questions about the spatial arrangement of local "landmarks." Like the parlor game "Twenty Questions," the questions are asked one at a time, answered "yes" or "no," and the ordering is adaptive. Consequently, the whole procedure is organized into a binary tree. There are, in fact, many trees, whose construction is randomized to reduce correlation, and which can be run in parallel. Each tree yields a distribution ("soft decision") rather than a classification; the final classification is based on an aggregate distribution. To date, our experiments involve deformable, planar shapes, specifically handwritten digits, but the method appears generic and should be adaptable to other visual recognition problems.

The trees are created off-line with training data. Each new question is chosen from a large pool in order to remove as much residual uncertainty as possible about the true hypothesis and taking into account the answers to the previous questions. (In our experiments, the data are images of individual handwritten numerals and the set $\mathcal{X}$ of possible hypotheses is simply $0, 1, ..., 9$.) Uncertainty is measured by the entropy of the pending posterior probability

distribution over the classes, i.e., the updated class likelihoods given the test results. Unlike the parlor game, however, the responses are highly ambiguous (due mainly to variations in the presentations of shapes) and therefore the true hypothesis cannot be determined without error by a deterministic process of elimination. Consequently, the most natural way to label the terminal nodes is by a *distribution* on $\mathcal{X}$ rather than a specific hypothesis in $\mathcal{X}$.

A departure from ordinary tree-structured methods is that our framework is non-Euclidean. In the standard approach to nonparametric learning and estimation, there is a training set; each instance is classified and assigned a numerical feature vector of fixed dimension. The questions are then scalar functions of this vector, perhaps simple ones such as "halfplane questions" which divide a single axis and thereby progressively subdivide the "feature space" into rectangular cells, or more complex ones involving linear or non-linear functions of many coordinates.

In our case each data point is an image, classified in $\mathcal{X}$. There is no natural feature vector, unless one takes this to be the raw image data themselves, i.e., the "vector" of grey levels. (This in fact is what is done in certain neural networks and nearest-neighbor schemes.) Nor do we extract numerical or "statistical" features in the usual sense. Instead, we consider an extension of the so-called "structural approach" based on a *graphical representation* of the *raw* image data. (There is no preprocessing.) We then construct *relational trees* based on probing this representation, progressively updating the distribution on $\mathcal{X}$ as more and more information about spatial relationships is accumulated.

The graphical representation is only implicit; it is never explicitly calculated and stored. The vertices are labeled by certain types of local binary patterns (landmarks), together with an image location. There is no sub-classification problem: the landmarks are deterministic functions of the data which *do not* have complex interpretations in terms of differential properties of curves ("concavity" etc.) or structural configurations ("ending," "turn" etc.). The edges of the representation graph are labeled by possible planar arrangements among the vertex locations, such as "north-south" and "next to." The arrangements need not necessarily involve nearby landmarks and can convey highly global information.

Questions are associated with the existence of *generic* subgraphs (partial representations), meaning that the vertex labels are location-independent. The high complexity of subgraph isomorphism is avoided by structuring the trees to limit the rate of growth of the subgraphs and maintain connectivity. In other words, at each node the set of questions investigated depends on the questions previously asked and their responses. More specifically, this set depends on the "pending subgraph" - the partial representation determined by all the previous "yes" answers along that branch back to the root. The new question may enlarge this

by at most one new vertex and one new relation. In effect, the questions are of the form: "Is (existing) vertex two northwest of (existing) vertex three?" or "Is there a vertex of type A south of (existing) vertex three?". Recognition is very fast since there is no on-line optimization, template-matching, or other computationally intensive procedures; we need only follow a pre-computed set of instructions to search for particular landmarks in particular regions.

By construction, the questions are scale-invariant, translation- invariant, and deformation-invariant. Partial rotation invariance is achieved by using internal coordinates, i.e., coordinate systems based on vertex locations.

The distribution associated with each terminal node is the empirical one generated by the training points which land there. Consequently, these trees are themselves distribution-valued questions, actually random measures in an appropriate stochastic framework. Each tree could itself be regarded as a classifier by plurality rule, i.e., taking the mode of the distribution. For any given training set, there is then a fundamental trade-off between the depth of the tree and its generality: deep trees have relatively peaked distributions, but have many nodes and require very large training sets in order to avoid over-dedication; shallow trees are obviously less decisive and accurate, but tend to be more representative of the global population from which the training set is sampled.

We attempt to overcome this dilemma by constructing many trees; hopefully an aggregate result reliably signals the true hypothesis. The trees are generated automatically by *randomly sub-sampling* from the pool of questions, exploring perhaps one to ten percent of the total number of allowable questions at each junction, and choosing the best question from this random subset. The fact that we can make many trees (at least 100) with any degree of "orthogonality" (statistical independence) is a consequence of the sub-sampling and the remarkable richness of the world of spatial relationships.

The manner in which the results are aggregated is ad hoc; in fact, we just sum the distributions and take the aggregate mode. We report experiments with several protocols for tree depth, randomization, etc. Even with training sets of order $10,000$, which are small by today's standards (see §7,8), we achieve recognition rates around 98% with no rejection. This is competitive with the best statistical, neural network, and other nonparametric classifiers; see §7 and [48]. Given more data, it is not necessary to construct new trees from scratch: one can simply "enrich" the terminal nodes (i.e., update the counts) and occasionally deepen the trees by splitting a terminal node if the amount of the data and degree of uncertainty warrant a new question. It should then be possible to exploit training sets of order $100,000$.

In the next section, §2, we identify and summarize some work related to ours. The general classification tree methodology, including the principle of entropy reduction, is reviewed

in §3. We introduce our relational framework in §4 and discuss distribution-valued trees, progressive learning, and the depth/generality tradeoff. Multiple trees are introduced in §5 as well as the role of random sampling of the questions. The issue of invariance with respect to deformations, distortions of data, and affine transformations is treated in §6. The character recognition problem is briefly outlined in §7 and some details about training and testing, together with experiments on handwritten digits, are presented in §8. Finally, a few speculative remarks are made in §9.

2 Tree-Based and Related Recognition Strategies

Single decision, search, and classification trees have already been used to a limited extent for pattern and object recognition ([11],[21]). In the context of rigid objects, Swain ([46]) constructs a tree based on high level features, entropy reduction, and object topology in order to recognize a small number of polyhedra. See also Spirkovska ([43]). Goad ([18]) uses search trees to match detected vs. predicted edges, and emphasizes pre-processing to reduce recognition time. Hansen and Henderson ([20]) use CAD models to generate a "strategy tree" based on features (e.g., edges and surface patches) which is used to generate hypotheses about object poses. Sossa and Horaud ([42]) explore a variation of geometric hashing based on relational graphs which capture the intrinsic topology of objects. Arkin et al ([4]) use decision trees for classifying convex planar shapes, asking whether individual pixels are "inside" or "outside." Wang and Suen ([47]) attempt to classify printed characters with trees. Decision trees have also been used for recognizing deformable shapes, for instance handwritten Chinese characters ([19]), biological shapes ([32]) and roads ([15], [16]), although in the latter work the tree is constructed *on-line*.

Our work has aspects in common with some of these approaches, but differs in several important respects, such as the use of multiple trees generated by randomization. In addition, our "features" are low level, robust and avoid subclassification; our trees are automatically generated from training data, not modeled by the user; and our questions are purely relational.

Besides classification trees, our method resembles other computational strategies for object recognition. For example, in the popular "hypothesis and test" paradigm, there is usually a feature-based representation for the objects (perhaps graphical), and two sub-processes are iterated: "indexing," in which a few detected features are used to elicit a candidate set of objects and/or poses; and "matching," in which a more global correspondence is investigated. In contrast to the repeated elicitation of candidate hypotheses, the indexing in our

approach, such as it exists, is dynamic and stochastic. We *gradually* formulate specific conjectures. There is an evolving distribution on hypotheses which is continually updated in light of the additional information derived from the most recent question. In fact, since these questions are associated with image locations (relative to established landmarks), our approach might be considered an example of "active vision" and proposals for aggregating the evidence accumulated by an active sensor.

The counterpart of object recognition is model registration. In ([2]) the issue of model registration for deformable objects is addressed using graphs describing the planar arrangement of local landmarks; see also ([3]). There the graphs are created by hand and are matched to the data using dynamic programming on decomposable subgraphs. The common premise here and there is that objects can be identified and discriminated through the planar arrangement of simple and robust local features.

3 Decision Trees

Decision trees are ubiquitous. We have already mentioned some applications to recognition. In addition, tree-structured protocols for searching, classifying, and other decision-making tasks appear in Information Theory, Combinatorial Optimization, Machine Learning and Statistics. For example, variations of the deterministic "Twenty Questions Game" correspond to well-known problems in coding theory and combinatorial optimization. In particular, the optimal strategy for choosing subsets is given by the *Huffman code* ([26], [49]) since the problem may be viewed as one of code construction under the constraint of minimizing the number of bits to examine during decoding. Generally, however, optimal strategies for choosing and ordering questions are virtually impossible to construct ([27]) and work centers on "greedy" strategies, sometimes called "splitting algorithms" ([12],[13],[33]).

In Machine Learning ([40]) and Statistics (e.g., CART [6]) the construction of decision trees is data-driven. The posterior distribution over hypotheses given certain test results is the *empirical* one determined by the training data. This is the approach taken here.

Papers dealing with the general methodology of tree construction include [6],[9],[22], [29], [37], [40], and [41]; see also the discussion in [21].

Nonparametric Tree Construction.

Nonparametric (=data-driven) construction of a (single) tree begins with:

- A set $\mathcal{X}$ of *hypotheses* (i.e., classification labels);

- A *training set* Ω_{tr} of data points with known labels $X(\omega) \in \mathcal{X}, \omega \in \Omega_{tr}$;
- A set $\mathcal{Q} = \{Q_1, ..., Q_N\}$ of *questions* (or tests or experiments), where each Q_n is a scalar function on Ω_{tr};
- A set of *decisions* $\mathcal{D}$.

In our case, the training set consists of binary images of isolated handwritten digits, the questions are relational functionals of the image data (specific examples are given later), and the decisions are distributions on $\mathcal{X}$.

When we write expressions such as $P(X = x)$, $P(Q_1 = q_1, ..., Q_N = q_n | X = x)$ and $P(X = x | Q_1, ..., Q_k)$ we are regarding X and $Q_1, ..., Q_N$ as random variables relative to the empirical distribution P on Ω_{tr}: $P(A) = \sharp A / \sharp \Omega_{tr}$, where $A \subset \Omega_{tr}$ and $\sharp A$ is the number of elements in A. Thus, for example, the *initial distribution* μ_0 over hypotheses is

$$\mu_0(x) = \frac{\sharp\{\omega \in \Omega_{tr} | X(\omega) = x\}}{\sharp \Omega_{tr}}.$$

Similarly,

$$P(X = x | Q_1 = q) = \frac{\sharp\{\omega \in \Omega_{tr} | X(\omega) = x, Q_1(\omega) = q\}}{\sharp\{\omega \in \Omega_{tr} | Q_1(\omega) = q\}}$$

the proportion of training points which answer "q" to question Q_1.

It will also be useful to imagine a much larger universe Ω of "data" from which the training set is randomly sampled or otherwise selected, each element having "true" label $X(\omega)$. In our case, Ω_{tr} is the set of images in our particular database of isolated handwritten digits and Ω is the set of *all* images of handwritten digits. Of course we hope that the distributional properties of the tree we construct, such as classification rates, *generalize* to Ω, or at least to "test sets" drawn from Ω.

The questions are asked sequentially and adaptively, meaning that at each stage we may utilize the results of previous responses in order to choose the next one. Let the index of the first question chosen be $\pi_1 \in \{1, ..., N\}$, the index of the second question $\pi_2 = \pi_2(q_1)$, which depends on the response q_1 to the first question, the index of the third question $\pi_3 = \pi_3(q_1, q_2)$, which depends on the first two responses, etc. The tree has Q_{π_1} at the root (or level-one) node, which branches into nodes corresponding to the possible answers to Q_{π_1}, etc. Our problem is then to ask these questions in an "optimal" order relative to some criterion, for instance asking as few as possible at a given accuracy level, or achieving the best accuracy with a given number of questions. Since these problems are intractable, we simply try to minimize uncertainty about X as much as possible at each step.

Therefore,

$$\pi_1 = \arg \min_{1 \le n \le N} H(X|Q_n).$$

In other words, we choose π_1 to minimize the expected amount of uncertainty about X given the answer to Q_{π_1}, which is the same as maximizing the expected gain in information. Now suppose the first k levels have been made, and we are at some level $k+1$ node, $k \ge 1$. Let $B_k = \{Q_{\pi_1} = q_1, ..., Q_{\pi_k} = q_k\}$, the set of data points in Ω_{tr} which respond $q_1, ..., q_k$ to the first k questions along the branch we are on. These data points are "sitting" at the current node under construction and will be used to determine the next split. Again, we choose the question which minimizes the expected uncertainty, but now under the "current" posterior $P(\cdot|B_k)$:

$$\begin{aligned} \pi_{k+1}(q_1, ..., q_k) &= \arg \min_{1 \le n \le N} H(X|B_k, Q_n) \\ &= \arg \min_{1 \le n \le N} \sum_{q \in \{0,1\}} P(Q_n = q|B_k) H(X|B_k, Q_n = q). \end{aligned}$$

Under the empirical distribution, this means choosing n to minimize:

$$- \sum_{q \in \{0,1\}} \frac{\sharp(\{Q_n = q\} \bigcap B_k)}{\sharp B_k} \sum_{x \in \mathcal{X}} \frac{\sharp(\{X = x, Q_n = q\} \bigcap B_k)}{\sharp(\{Q_n = q\} \bigcap B_k)} \log_2 \frac{\sharp(\{X = x, Q_n = q\} \bigcap B_k)}{\sharp(\{Q_n = q\} \bigcap B_k)}$$

where $\{Q_n = q\} = \{\omega \in \Omega_{tr} | Q_n(\omega) = q\}$, and similarly for $\{X = x, Q_n = q\}$.

We stop asking questions when one of the hypotheses becomes overwhelmingly likely. This is equivalent to stopping as soon as the entropy of the posterior distribution $P(\cdot|B_k)$ falls below a threshold. Such a node is declared to be "terminal," and labeled by one of the decisions $d \in \mathcal{D}$. If $\mathcal{D} = \mathcal{X}$, i.e., if we make a guess $\hat{X}$ about the true hypothesis at each terminal node, then each label will be associated with many terminal nodes, and following such a terminal node back to the root will produce one sequence of observations for which $\hat{X} = x$. For each x, the totality of such sequences determines a region of observation space - the event $\{\hat{X} = x\}$.

4 Relational Trees

4.1 Euclidean vs. Relational Framework

In the standard applications of the above construction above ([6], [40]), [21]) there is a "feature vector" $Y_1(\omega), ..., Y_M(\omega)$ of fixed dimension associated with each data point $\omega \in \Omega_{tr}$ and a set of questions is based on this vector. (In some cases the feature vector is categorical, e.g.,

in machine learning.) Often, the questions are of the form $Q_n(\omega) = I_{A_n}(Y_1(\omega), ..., Y_M(\omega))$ where A_n is a linear subspace of feature space and I_{A_n} is the indicator function. For example, in CART, the questions involve halfplanes, i.e., each question is of the form "Is $Y_n(\omega) < c$?" for some particular coordinate n and cutpoint c. Other examples involve linear combinations of the feature variables. In general all the questions can be asked at a given node. Many methods have been proposed for deciding when to stop splitting nodes, i.e., declare nodes to be terminal. For example, rather long trees may be grown and then pruned back. Generally the terminal nodes are assigned one of the possible classifications and it is well-known that good performance on the training set may not be reflected in results on test data. See [6].

It is possible, but awkward, to express our questions in such a framework. Perhaps the easiest way is the following. Recall that our data representation is based on a labeled graph, and our questions concern the existence of subgraphs. Now index a binary feature vector by 3-tuples from the set

$$\{vertex\ labels\}^2 \times \{image\ locations\}^2 \times \{relations\}.$$

If j is one of these 3-tuples, the corresponding feature, Y_j, assumes the value 1 if the indicated relation holds between the indicated vertex labels (landmarks) at the indicated locations, and $Y_j = 0$ otherwise. The subgraph questions are then of the form $Q_n = I_{A_n}(Y_1, ..., Y_M)$ where M is the size of the index set (very large) and A_n is, in general, an extremely complicated subset of $\{0,1\}^M$, partly because the questions involve matching *generic* subgraphs which involve landmarks and relations but no locations. Clearly, this representation is unnatural. Hence we view our approach as inherently relational and non-Euclidean.

Let $\mathcal{L}$ and $\mathcal{R}$ denote, respectively, the set of possible vertex and edge labels. These will be made precise momentarily. The corresponding set of generic labeled (or attributed) graphs is denoted $\mathcal{G}$; thus each $g \in \mathcal{G}$ consists of a set of vertices $\{v_1, ..., v_k\}$ with $v_i \in \mathcal{L}$ and a set of *ordered* edges $\{R(i_s, j_s)\}, R(i_s, j_s) \in \mathcal{R}, 1 \leq i_s, j_s \leq k, s = 1, ..., S$.

There is a question Q_g for every labeled graph $g \in \mathcal{G}$. The answers are $Q_g(\omega) = 1$ if g is a subgraph of $G(\omega)$ and $Q_g(\omega) = 0$ otherwise. In other words, we are asking whether or not the "configuration" g appears in the image ω. In principle, once $\mathcal{L}$ and $\mathcal{R}$ are specified, one could attempt to build trees exactly as above. But this would obviously not be computationally feasible since the set of possible questions is too large and checking subgraph isomorphism is too intensive.

The basic idea is to severely restrict the set of allowable questions at each junction. The restricted set depends on B_k. The entropy minimization problem is then highly constrained. We shall explain how this is accomplished after defining $\mathcal{G}$.

4.2 Landmarks

The possible vertex labels are defined based on the eight masks. Recall that our data points are *binary* images, say dark digits on a white background. (We will indicate how to accommodate grey levels in §6.1.) In the masks, the 1's represent digit pixels, the 0's represent background pixels and the 2's could be either digit or background. Loosely speaking, the masks are designed to detect locations where the tangents to object boundaries lie in one of four possible directions: north-south, east-west, northeast- southwest, and northwest-southeast. Four of the masks are shown below. The other four result from switching the 0's and 1's, i.e., switching the object-background orientation.

$$\begin{pmatrix} 0 & 0 & 2 & 1 & 1 \\ 0 & 0 & 2 & 1 & 1 \\ 0 & 0 & 2 & 1 & 1 \\ 0 & 0 & 2 & 1 & 1 \\ 0 & 0 & 2 & 1 & 1 \end{pmatrix} \begin{pmatrix} 2 & 2 & 1 & 1 & 2 & 2 \\ 2 & 2 & 1 & 1 & 1 & 2 \\ 0 & 0 & 2 & 1 & 1 & 1 \\ 0 & 0 & 0 & 2 & 1 & 1 \\ 2 & 0 & 0 & 0 & 2 & 2 \\ 2 & 2 & 0 & 0 & 2 & 2 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 2 & 1 & 1 & 2 & 2 \\ 2 & 1 & 1 & 1 & 2 & 2 \\ 1 & 1 & 1 & 2 & 0 & 0 \\ 1 & 1 & 2 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 & 0 & 2 \\ 2 & 2 & 0 & 0 & 2 & 2 \end{pmatrix}$$

Figure 1: The first four masks used to identify landmarks

Clearly this is a very primitive set of masks; plenty of additional possibilities (e.g., including more directions) could be tried.

There are sixteen vertex labels corresponding to sixteen landmark types, $L_1, ..., L_{16}$. A pixel u is lighted as an instance of one of the first eight landmarks if there is a perfect match between the corresponding mask and image data when the mask is centered at u. The second group of landmarks is a "robust" version of the first group. Occurrence of one of these means the same as above, except that there needn't be a perfect match; specifically, the number of matching zeros and the number of matching ones must both exceed some threshold. In Figure 2 we show an image together with all the instances of landmarks L_4 and L_{12}.

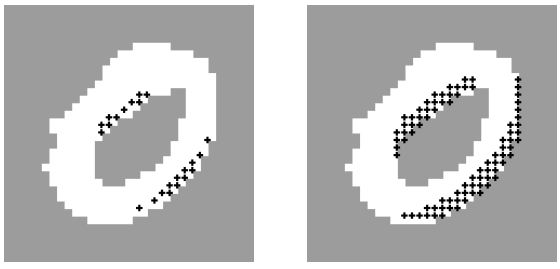

Figure 2: Instances of landmarks 4 and 12.

These landmarks are very crude and do not represent distinguishing or complex properties of object boundaries. In general, many (sometimes all) of the sixteen landmarks occur at least somewhere in every image and there is consequently very little information in mere

existence. On the other hand, there is no sub-classification problem: the occurrence or non-occurrence of a landmark at an image location is unambiguous.

In order to simplify the book-keeping involved in storing instances edges, we have clustered the instances of individual landmarks. All the pixels in disjoint blocks which light up for a given landmark are represented by one pixel at the center of mass. In Figure 3 we show the locations of the landmarks in Figure 2 after clustering.

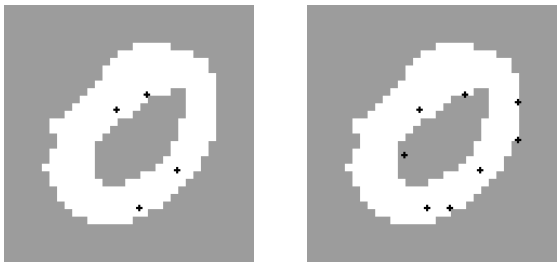

Figure 3: Instances of landmarks 4 and 12 after clustering.

Finally, there is one more landmark, L_0, which is the "OR landmark" and occurs at pixel u if *any* of the sixteen others occur there. It is informative to know where there is stroke and where there is background.

4.3 Relations.

Basically, the edge labels are the eight oriented relationships "south," "southeast," "east," etc., with each interpreted at three "tolerances," yielding twenty-four possible binary relations. We abbreviate these as $\mathcal{R} = \{S_1, S_2, S_3,\ SE_1, SE_2, SE_3, ...\}$. The precise definition depends on the orientation of the vector between two landmarks locations.

Note: We shall describe the set-up in an absolute coordinate system. In order to achieve rotation invariance, it is necessary to use internal coordinates based on relative vertex locations. For simplicity we shall omit the details.

An edge $R \in \mathcal{R}$ exists between an instance of landmark L_i at location $u = (u_1, u_2)$ and an instance of landmark L_j at location $w = (w_1, w_2)$ if the vector $V = (w_1 - u_1, w_2 - u_2)$ lies in a polar wedge associated with R. The three tolerances are $\pm\pi/4, \pm\pi/8$ and $\pm\pi/16$. Thus, for example, there is an edge of type S_1 if the angle between the vectors V and $(0, -1)$ falls in $(-\pi/4, \pi/4)$ (i.e., L_j is roughly "below" L_i); of type S_2 if this angle falls in $(-\pi/8, \pi/8)$; and of type S_3 if it falls in $(-\pi/16, \pi/16)$ (i.e., L_j lies almost directly south of L_i). Notice that

two landmarks may be linked by several distinct edge types, and that edges exist between *every* pair of landmarks.

The graph $G(\omega)$ assigned to an image ω is the one determined by *all* (clustered) landmark locations and *all* existing relations. The amount of information extracted with these ingredients is immense. Moreover, one can also consider ternary relations (involving the triangle formed by three landmark locations) and relations between relations, such as parallel, co-linear, etc., ("perceptual groupings"- [34]). We have only considered binary relations. Notice that any graph $g \in \mathcal{G}$ may have many "instances" in ω, i.e., there may be many copies of g in $G(\omega)$; see Figure 4.

4.4 Growth Constraints

The pool of questions available at the top node of the tree ask whether a specific relation exists between two specific types of landmarks. (This is the minimal configuration with any discriminatory power.) In other words, the first question is Q_{π_1} where π_1 is a graph with two vertices and one edge, hence of the form $(v_1 = L_i | R_k | v_2 = L_j)$ where $R_k \in \mathcal{R}$. If $Q_{\pi_1} = 0$, we choose again from such graphs. If $Q_{\pi_1} = 1$, there are one or more instances of the pending graph, π_1. In principle all these instances should be stored and any future question should be answered for every instance. (In practice, however, we maintain a limit on the number of kept instances, a form of pruning.) We now choose choose $\pi_2(1)$ from among extensions of π_1 to *connected* graphs with either: (i) Two vertices and two edges, or (ii) Three vertices and two edges.

In general terms, let t be a level $k+1$ node and let $\mathcal{G}_t$ be the set of allowed choices. The graphs at the ancestor nodes are $\pi_j, j = 1, ..., k$; the one with the largest index j^* with $Q_{\pi_j^*} = 1$ is the "pending graph," i.e., the largest *confirmed* graph. Any $g \in \mathcal{G}_t$ must expand $g_{\pi_j^*}$ by either (i) one edge or (ii) one vertex and one edge involving this vertex. In this way, the pending graph grows slowly and remains connected. In practice, these constraints reduce the search to several hundred questions. In Figure 4 we show two instances of one graph g found in in an image of an "8", the same graph found in an image of a "2", together with the description of the labels of this specific pending graph.

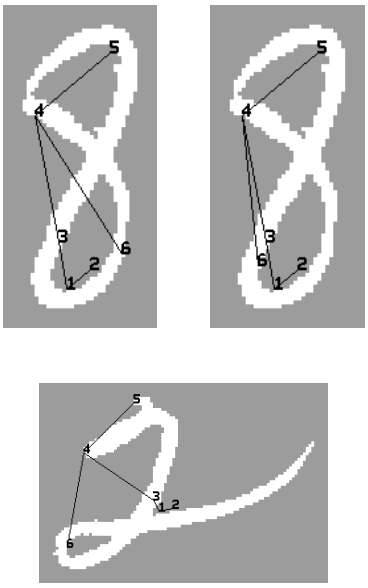

Figure 4: Two instances of a graph found in an 8, and one instance found in a 2. The labeled vertices and edges are:
$(v_1 = L_7|NE_1|v_2 = L_{16}), (v_1 = L_7|N_1|v_3 = L_4), (v_3 = L_4|NW_2|v_4 = L_2), (v_4 = L_2|NE_1|v_5 = L_{10}), (v_4 = L_2|S_1|v_6 = L_5)$.

NOTE: Some degree of connectivity is crucial for performance. For example, if the pending graph consists entirely of disconnected subgraphs of size two (e.g., if the procedure at the root node is repeated at every node) then the results are relatively poor. It is the gradual accumulation of relationships among a growing, but relatively small, number of landmarks that provides discriminating power.

4.5 Generalization

Obviously performance and other attributes of the system depend on the size and choice of the training set. We do not know how performance is affected by the size of Ω_{tr} since we have not done systematic experiments (such as those in [44]) with the database we have (see §8), let alone with very large ones.

Another issue is the extent of performance deterioration from training to test data. This will be pronounced *unless* the trees are very shallow, so that every question is chosen based on a population of data points sufficiently large that the same question would be chosen for any other large sample of data points with the same "history." In this case, the performance of the individual classifiers is generally mediocre, unless, of course, the training set is extremely large, in which case the trees can be both general and deep. One alternative is to construct trees with one training set and then enrich them (i.e., update the counts) with considerably more data. In this case the performance fall-off is largely eliminated. A closely related

issue is the extent to which equally-sized training sets might yield different classifiers, i.e., the problem of "over-dedication." (See the discussion in [17] about the "bias-variance" dilemma.)

Suppose a tree is constructed based on Ω_{tr}. Thus, for each data point $\omega \in \Omega_{tr}$, we define a probability distribution $\mu(\omega) = \{\mu(x,\omega), x \in \mathcal{X}\}$ where

$$\mu(x,\omega) = \frac{\sharp\Omega(x,\omega)}{\sharp\Omega(\omega)}$$

Here $\Omega(\omega)$ is the portion of the training set which lands in the same terminal node of the tree as does ω, and $\Omega(x,\omega)$ is the set of points in $\Omega(\omega)$ which have label x. We regard μ as a distribution-valued random variable on Ω_{tr}. Then, almost by definition, for any distribution ν on the classes, $P(X = x|\mu = \nu) = \nu(x)$. In other words, given we are in the set of data points which land in *any* terminal node with distribution ν, the conditional probability under P that the label is x is $\nu(x)$. Still fixing the tree, and given another, larger set Ω, μ is again a random distribution on Ω, and one may ask whether the resulting conditional distribution of X given μ is still μ itself, i.e., whether $\mu(\omega)$ still represents the correct posterior distribution on the labels of data points, but now across the larger domain, most of which may not have been used in the construction of the tree. If this property remains valid than indeed the tree "generalizes" from Ω_{tr} to Ω.

Again, shallow trees are more general than deep ones, and questions made with relatively few data points will tend to be over-dedicated to the training set, resulting in distributions which can be significantly different from those obtained by enrichment.

4.6 Progressive Learning

We therefore consider a two-step learning procedure consisting of (i) initial construction and (ii) enrichment and deepening. In the first step we are building the basic classifier and obtaining an initial estimate of the "parameters," namely the weights at the terminal nodes. In the second step we are updating our estimates based on new data, and enlarging the tree by splitting some terminal nodes. Each new (classified) data point adds one count (to each tree). The procedure is computationally efficient since the initial construction is based on very "conservative" stopping rules and hence can be accomplished with moderately-sized training sets.

4.7 On-Line Execution

The training (=tree construction) may be intensive, but the on-line recognition is extremely simple: we need only execute the strategy, i.e., follow the instructions. *There is no on-line optimization.* Moreover, since we traverse exactly one path of the tree it doesn't matter how many different questions are represented in the entire tree: we must only ask the those encountered on the path dictated by the data.

If the classification is based on a single tree then the obvious decision rule is to classify a terminal node by the mode of the (empirical) distribution $\mu(\omega)$ - that class which has the most representatives at that node. The performance of this classifier was not satisfactory (see §8.6), which led us to consider multiple trees.

5 Multiple Trees

The idea of using multiple decision trees is relatively new although it appears to be quickly gaining popularity. Some authors have suggested using multiple "pruned" or "shrunken" subtrees of one single decision tree ([10]) . Others have created mutiple decision trees through interaction with the user ([30]). A very recent idea is to use mutiple "decision stumps", namely depth two trees; see [39]. In [7], multiple trees are created by generating bootstrap replicates of the learning set and creating a decision tree for each such replicate. Most of these papers are dealing with relatively small data sets, fixed size feature vectors, and questions or splits of the single coordinate type. All accounts of multiple trees point to an increase in performance and stability of the classifier.

In our context the data sets are very large and the family of allowable relational questions is immense. This suggests yet another mechanism for generating multiple trees: choosing a random subset of the allowable questions at each node and selecting the best question from this subset. This is an automatic procedure which appears to promote "orthogonality" among the trees. It should be emphasized that this is *not* a procedure which simply picks a question at random at each node, thereby generating completely random trees, as suggested in Mingers ([36]) and criticized in Buntine and Niblett ([8]).

Our procedure is the following.

- Start with a training set Ω_{tr} of modest size and separately create many trees of modest depth;
- Promote "orthogonality" among the trees by generating a small *random subset* from the pool of questions at every node and choosing from that subset the question with

most information gain ;

- Declare nodes to be terminal when the number of data points falls below a threshold, the node is sufficiently "pure," or a maximum depth is reached;
- Label each terminal node by the histogram of counts from Ω_{tr};
- Enrich the terminal nodes with any additional training data by updating the counts, occasionally expanding nodes when there is enough data and uncertainty to warrant a new question;
- Classify using the mode of the *aggregate* distribution.

5.1 Individual vs. Simultaneous Construction

We have chosen to make the trees individually, using the same or separate protocols for each tree. In particular, the questions chosen at the nodes of tree j have no impact on those chosen for tree l. Consequently, the trees are constructed one at a time, without reference to those already built, and following a protocol which incorporates the constraints on graph-growing described in §4.4 as well as random sub-sampling of the questions meeting those constraints; an example will be given in §8. (In principle, all the trees could be constructed in parallel.) For each sampled question, the resulting average empirical entropy over the classes is computed and the question is chosen for which this is smallest, as described in §3.

Another possibility is to construct the trees simultaneously, in an interdependent fashion, where the criterion itself for selecting a question in any one tree will depend on the partial construction of all other trees. For example, one might build all the trees to depth one, then all to depth two, etc., or randomly select any (current) terminal node for splitting, allowing the trees to grow at different rates. The entropy criterion would be replaced by one which takes into account the *overall* system performance based on the pending terminal distributions. One example would be the overall misclassification rate. Another is the sum over all training points of the aggregate mass (see §5.3 below) placed on the label of the training point. Experiments in this domain are still preliminary.

5.2 Orthogonality

Conditional on X, suppose the trees were actually independent as random variables on a suitably large space of images Ω; then if each tree individually had any discriminating power,

the overall performance could be made arbitrarily high. Obviously this is not possible and the trees are in fact quite dependent, even conditionally given X.

Our approach is not based on assessing statistical independence with an appropriate stochastic framework. Instead, we simply try to make the trees as "different" as possible by trying to avoid asking the same questions in comparable situations (similar node histograms) from tree to tree.

We have experimented with several alternatives; the one which seems to give the best results is to use random sub-sampling of the questions. Suppose we are at a (current) terminal node t and let $\mathcal{G}_t$ be the set of graphs we are allowed to consider (§4.4). The corresponding set of possible questions is $\mathcal{Q}_t = \{Q_g | g \in \mathcal{G}_t\}$, the number of which depends on the depth of node t (more specifically the number of "yes" answers in the node history) because the question are tied to existing vertices in the pending graph; the more existing vertices the more possible questions. We randomly select a fixed percentage of the questions in $\mathcal{Q}_t$, for example one to ten percent, and choose from among these the one with highest information gain. Invariably, the discriminating power of each individual tree is reduced. This is to be expected since there is necessarily a *smaller* reduction in entropy at each node than would be the case if all the allowable questions were permitted to actually be used. However, the aggregate performance of the family of trees can be substantially improved.

5.3 Aggregating Results

The classification is made as follows. Suppose K trees have been made. Given a test set Ω_{test}, each $\omega \in \Omega_{test}$ is dropped down each tree, yielding distributions $\mu_1(\omega), ..., \mu_K(\omega)$. Averaging over these we obtain the aggregate distribution:

$$\bar{\mu}(x, \omega) = \frac{1}{K} \sum_{k=1}^{K} \mu_k(x, \omega), \ x \in \mathcal{X}$$

The estimated label is then

$$\hat{X}(\omega) = \arg \max_{x \in \mathcal{X}} \bar{\mu}(x, \omega).$$

The classification rate (at zero percent rejection) is the proportion of points in Ω_{test} for which $X(\omega) = \hat{X}(\omega)$.

Various criteria can be used to measure the confidence in the classification. For example, if the ratio of the weight $\bar{\mu}(\hat{X}(\omega), \omega)$ of the largest class to that of the second largest class exceeds some threshold, then the classification is accepted; otherwise it is discarded and the data point is considered to be unclassified. In §8 we will report some typical values for zero percent rejection and about six percent rejection.

5.4 Robustness

One additional advantage of using multiple trees is the insensitivity to changes in the protocol for individual tree construction. Making what appear to be relatively major changes in various "parameters," such as stopping criteria, sampling schemes, landmark masks, and splitting criteria, has virtually no effect on the overall classification rate.

6 Invariance

Our experiments concern the classification of handwritten digits based on high resolution binary images. We believe the strategy could be adapted to other visual recognition problems. Any truly generic and extendible strategy must accommodate a wide range of imaging scenarios, including variations in shape, pose, and lightning, and the presence of clutter, occlusion, and noise.

6.1 Data Distortion

We refer to transformations of the raw image data, due, for example, to changes in lighting, and to other distortions caused by image noise, blur, occlusion, etc. Due to our high quality data, these issues have not been seriously addressed. Still, the image attributes we utilize are local and elementary, basically just locations of oriented boundaries; the masks are designed to accommodate considerable variation in boundary presentation and appear to be stable against various forms of "binary noise." Extensions to grey-level images should therefore be possible by replacing the matched filters with relational masks depending only on *relative* brightness, i.e., comparisons.

Finally, the use of multiple trees provides a natural mechanism for dealing with occlusion and clutter. In fact, we have done some preliminary experiments in which trees are constructed based on differing portions of the image data, for instance using only half the image (top, left, etc.). The results are encouraging.

6.2 Shape Variation

By construction, the questions are deformation-invariant, in the sense that the responses are invariant to "rubber sheet" transformations, at least those which preserve the essence of the shape, i.e., keep a "4" looking like a "4".

6.3 Affine Transformations

The method is also translation-invariant by construction. Rotation-invariance can be included by aggregating the questions into invariant classes and using internal coordinates. For example, full rotation-invariance can be achieved by replacing the first question by an invariant version in which the entire structure (the two participating landmarks and the relation between them) is said to exist when the landmarks and relation are rotated through multiples of some small angle. Once the pending graph is non-empty, future questions are then based on the coordinate system determined by the locations of the first two vertices. Naturally there is a fall-off in performance, even beyond such fundamental confusions as "6"and "9". Partial rotation-invariance results from using a coordinate frame based on two existing vertices when searching for a new landmark.

The size range in the database is considerable, some digits being up to four times larger than others in both height and width. Nontheless, we do not assume we know the size, nor do we estimate it. In particular, neither the clustering nor the masks are adapted to the size of the digit. Some adjustment for (estimated) scale would likely be helpful, even necessary for attempting to recognize multiple objects (e.g., reading zip codes), but there does not seem to be any simple way to accomplish this without confronting the recognition/segmentation dilemma.

7 Character Recognition

The problem has many variations and the literature is immense. Surveys include [38] and [45]. Even the problem of *handwritten* character recognition (as opposed to machine printed characters) has many facets, depending on whether recognition is on-line or off-line, whether the characters are isolated or touching, binary or grey-level, etc. The most difficult problem is the recognition of unconstrained script, i.e., "reading" unsegmented alphanumeric character strings based on pictures of ordinary handwriting. Zip codes ([1], [35]) and hand-drawn checks also present a formidable challenge.

The problem we consider is easier, mainly because the digits are segmented from the background and from each other. Even this problem has drawn enormous attention, including a competition sponsored by the National Institute of Standards and Technology ([48]).

For nonparametric statistical methods see [5],[23] and [31]. Usually the techniques involve a neural network or some other form of nonlinear discriminant analysis, and do not address the description of shape in any direct way. An example of a model-based statistical approach

is found in [24], in which shape is explicitly modeled by crafting prototypes for each class.

It is somewhat difficult ([1], [28]) to assess the state-of-the-art; for example, some "test sets" are widely considered to be more difficult than others ([44]). 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 produced error rates of less than five percent; see [48]. Several of the top methods were based on nearest neighbor and neural network classifiers. The test set was considered quite difficult and recognition rates reported elsewhere are generally higher. For example, using a nearest-neighbor system, the study in [44] achieves rates of 96.88% with $10,000$ training points, 97.92% with $30,000$ training points, and 98.61% with $100,000$ training points. With smaller training sets (around 5000), the rates are $95-96\%$ with nearest-neighbor and feed-forward neural networks ([25]).

8 Experiments

8.1 Data

The experiments described were carried out with the USPS database produced and distributed by CEDAR, SUNY Buffalo. The training data consists of 19046 binary digits segmented from zipcodes; the test set consists of 2213 segmented binary digits. The images vary widely in dimensions, from approximately 25×25 to 100×100. No preprocessing step was implemented to normalize or otherwise alter the data. The distribution over classes in Ω_{tr} is not uniform, but our results are nearly identical when uniform subsets are selected for training and testing. Figure 5 shows some digits from the test set.

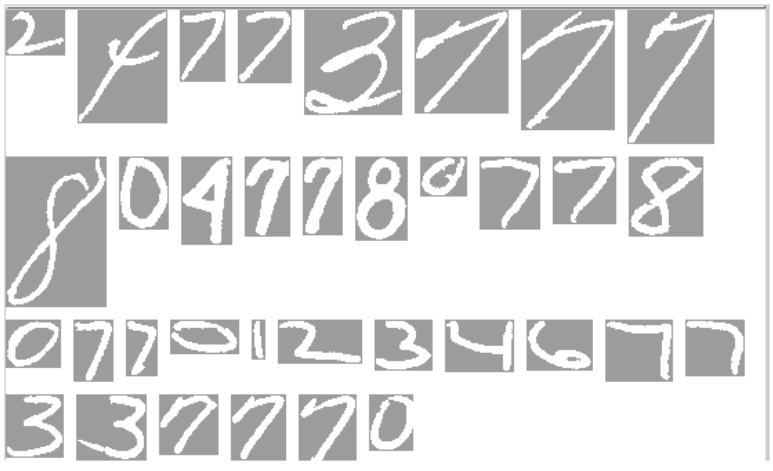

Figure 5: The first 40 images of the USPS test dataset.

8.2 Questions

Recall that $\mathcal{L}$ and $\mathcal{R}$ are, respectively, the set of vertex and edge labels. For the particular examples described in §4, $\sharp\mathcal{L} = 17$ and $\sharp\mathcal{R} = 24$. If there are V vertices in the pending graph, there are $V(V-1) \times \sharp\mathcal{R}$ possible questions of the first type and $V \times \sharp\mathcal{R} \times \sharp\mathcal{L}$ possible questions of the second type. These numbers are manageable and it is feasible to construct trees of considerable depth, say up to 20, even without random sub-sampling. Due to the stopping rules, the trees are in fact less deep and highly unbalanced.

8.3 Stopping Rules

A stopping rule is used to determine when a node in the graph cannot be split any further. The trees are grown to a maximum depth D. In addition, a node is not split unless the number of data points exceeds a minimum value M and the proportion of data points in the largest class is less than P. Typical values are $D = 12, M = 50$ and $P = 90$. This results in trees with approximately $200 - 300$ nodes.

8.4 Enrichment

With the USPS database we have enriched trees initially made with approximately $10,000$ data points by sending down an additional $10,000$. Unless the stopping rules are extremely conservative, the classification rates based on individual trees and on the initial $10,000$ data points are higher than the rates on the test data. These two rates become virtually identical when the distributions are updated; however, the overall classification rate is only slightly improved. We expect additional improvement with sufficient data to warrant expanding the trees, and work is underway using the recently acquired NIST database ([14]).

8.5 Computation

For both training data and test data the landmark locations were precalculated and stored. No effort has been made to accelerate any of the computations. For example, it would be straightforward to search simultaneously for all the landmarks rather than one by one. Similarly, one could optimize the question-answering routine. Depending on the protocol, each tree requires about $30 - 60$ minutes to build on a Sun Sparc 10. The computation time for classification is approximately 100 data points per tree per second.

8.6 Test Results

The best performance we could achieve with a single tree was approximately 92%, and we abandoned this approach. In Figure 6 we show the results for four groups of approximately fifty trees constructed under various protocols and based on $10,000$ training points.

One protocol involves random sampling from the landmark types only, specifically choosing three of the landmarks at each node, but investigating all possible relations. The classification rate for this group reaches 97.4%, but is over 97% by the fifteenth tree. Moreover when tested on the remaining 10000 training data points (before the enrichment step) the classification rate was just over 98%. Another protocol is the same, except that the entropy criterion is only applied to the two leading classes rather than all ten classes. The other two protocols involve variations in the stopping criteria, landmark masks, sampling schemes and the use of ternary questions.

Combining several groups results in rates around 98%. If we demand that the mode $\hat{X}$ have at least twice the weight of the next largest class, then 6.3% of the test data is rejected and the classification rate on the remaining data is 98.8%.

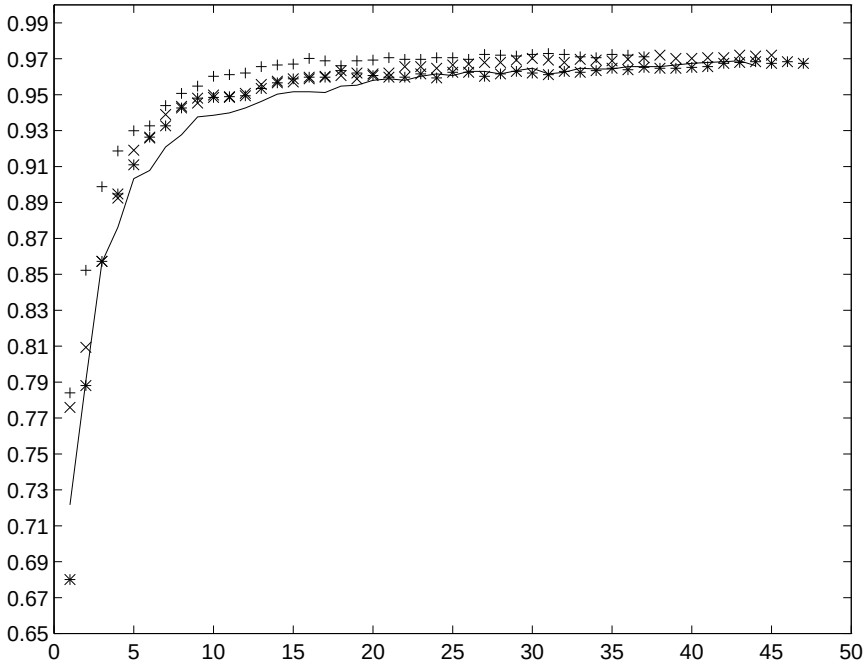

Figure 6: Cumulative classification rates with four different protocols.

9 Discussion

We have reported our progress on an approach to shape recognition based on asking questions about the spatial arrangement of oriented boundary segments. The questions are organized into many highly structured sequences corresponding to branches on many decision trees. The outcome of each tree is a distribution over the possible shape classes. The use of trees for classification is not new, especially within a standard framework with feature vectors. Nor is data-driven tree construction based on entropy reduction. So far as we know, however, the use of a graphical or relational framework is new, as well as the use of randomization in the construction of multiple, distribution-valued trees.

The current implementation suffers several limitations and does not extend *as it stands* to the recognition of multiple objects in grey-level images. Nonetheless, we are encouraged by the recognition rates on binary images of isolated handwritten numerals, especially in view of the relatively small training sets and the stability of the procedure. We believe that computational strategies based on the systematic accumulation of relationships hold considerable promise for general recognition.

Acknowledgement. The interest of the second author in the "twenty questions" paradigm originates in unpublished work with E. Bienenstock, S. Geman, and D.E. McClure on invariant recognition of rigid objects, such as vehicles and character fonts. Both authors would also like to thank Stu Geman for stimulating discussions about competing recognition paradigms.

References

[1] Ahmed, P. and Suen, C.Y., "Computer recognition of totally unconstrained handwritten ZIP codes," Inter. J. Pattern Recog. and Artif. Intell., 1, 1-15, 1987.

[2] Amit, Y. and Kong, A., "Graphical templates for image matching," Technical report no. 373, Dept. of Statistics, University of Chicago, 1993.

[3] Amit, Y., "Graphical shape templates for deformable model registration with application to MRI brain scans," Technical Report, Department of Statistics, University of Chicago, 1994.

[4] Arkin, E., Meijer, H., Mitchell, J., Rappaport, D., and Skiena, S., "Decision trees for geometric models," Proc. Ninth ACM Symp. on Computational Geometry, 1993.

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

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

[7] Breiman, L., "Bagging Predictors," Technical Report no. 421, Department of Statistics, University of California, Berkeley, 1994.

[8] Buntine, W. and Niblett, T., "A further comparison of splitting rules for decision-tree induction," *Machine Learning*, 8, pp. 75-85, 1992.

[9] Chou, P.A., "Optimal partitioning for classification and regression trees," IEEE Trans. PAMI, 13, 340-354, 1991.

[10] Clark, L. A. and Pergibon, D., "Tree-Based Models," in *Statistical Models in S* ed. Chambers, J. M., and Hastie, T. J., Wadsworth & Brooks, Pacific Grove, CA, 1992.

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

[12] Garey, M.R., "Optimal binary identification procedures," SIAM J. Appl. Math., 23, 173-186, 1972.

[13] Garey, M.R. and Graham, R.L., "Performance bounds on the splitting algorithm for binary testing," Acta Informatica, 3, 347-355, 1974.

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

[15] Geman, D. and Jedynak, B., "Shape recognition and twenty questions," Technical Report No. 2155, INRIA-Rocquencourt, November, 1993.

[16] Geman, D. and Jedynak, B., "Tracking roads in satellite images by playing twenty questions," Technical Report, University of Massachusetts, Dept. of Mathematics, 1994.

[17] Geman, S., Bienenstock, E. and Doursat, R., "Neural networks and the bias/variance dilemma," Neural Computation, 4, 1-58, 1992.

[18] Goad, C., "Special purpose automatic programming for three- dimensional model-based vision,"Proc. ARPA Image Understanding Workshop, 94-104, 1983.

[19] Gu, Y.X., Wang, Q.R., and Suen, C.Y., "Application of multilayer decision tree in computer recognition of Chinese characters," IEEE Trans. PAMI, 5, 83-89, 1983.

[20] Hansen, C. and Henderson, T., "Towards the automatic generation of recognition strategies," Second International Conf. on Computer Vision, IEEE, 275-279, 1988.

[21] Haralick, R.M. and Shapiro, L.G., *Computer and Robot Vision*, Vol. I, Addison-Wesley, Reading, Massachusetts, 1992.

[22] Hartmann, C., Varshney, P., Mehrotra, K. and Gerberich, C., "Application of information theory to the construction of efficient decision trees," IEEE Trans. Info. Theory, 28, 565-577, 1982.

[23] Hastie, T., Buja, A. and Tibshirani, R., "Penalized discriminant analysis," Preprint, Department of Statistics, University of Toronto, 1993.

[24] Hastie, T. and Tibshirani, R., "Handwritten digit recognition via deformable prototypes," Preprint, Department of Statistics, University of Toronto, 1993.

[25] Hecht-Nielsen, R., *Neurocomputing*, Addison- Wesley, Reading, Massachusetts, 1990.

[26] Huffman, D.A., "A method for the construction of minimum redundancy codes," Proc. I.R.E., 40, 1098-1101, 1952.

[27] Hyafil, L. and Rivest, R., "Constructing optimal binary decision trees is NP-complete," Information Processing Letters, 5, 15-17, 1976.

[28] Kahan, S., Pavlidis, T. and Baird, H., "On the recognition of printed characters of any font and size," IEEE Trans. PAMI, 9, 274-287, 1987.

[29] Kurzynski, M.W., "The optimal strategy of a tree classifier," Pattern Recognition, 16, 81-87, 1983.

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

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

[32] Lin, Y.K. and Fu, K.S., "Automatic classification of cervical cells using a binary tree classifier," Pattern Recognition, 16, 69-80, 1983.

[33] Loveland, D.W., "Performance bounds for binary testing with arbitrary weights," Acta Informatica, 22, 101-114, 1985.

[34] Lowe, D.G., *Perceptual Organization and Visual Recognition*, Kluwer Academic Publishers, Boston, 1985.

[35] Mitchell, B. and Gilles, A., "A model-based computer vision system for recognizing handwritten ZIP codes," Machine Vision and Applications, 2, 231-243, 1989.

[36] Mingers, J., "An empirical comparison of selection measures for decision-tree induction," *Machine Learning*, 3, pp. 319-342, 1989.

[37] Miyakawa, M., "Criteria for selecting a variable in the construction of efficient decision trees," IEEE Trans. Computers, 38, 130-141, 1989.

[38] Mori, S., Suen, C.Y. and Yamamoto, K., "Historical review of OCR research and development," Proc. IEEE, 80, 1029-1057, 1992.

[39] Oliver, J.J. and Hand D., "Averagin over decision stumps," *Proc. ECML-1994*, ed. Bergadano, F. and De Raedt, L., Springer Verlag, Berlin, 1994.

[40] Quilan, J.R., "Induction of decision trees," Machine Learning, 1, 81-106, 1986.

[41] Quinlan, J.R. and Rivest, R.L., "Inferring decision trees using minimum description length principle," Information and Computation, 80, 227-248, 1989.

[42] Sossa, H. and Horaud, R., "Model-indexing: the graph-hasing approach, IEEE Conf. Comp. Vision Patt. Recog., Champaign, Ill., June, 1992.

[43] Spirkovska, L., "Three-dimensional object recognition using similar triangles and decision trees," Pattern Recognition, 26, 727-732, 1993.

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

[45] Suen, C.Y., Nadal, C., Legault, R., Mai, T.A. and Lam, L., "Computer recognition of unconstrained handwritten numerals," Proc. IEEE, 80, 1162-1180, 1992.

[46] Swain, M., "Object recognition from a large database using a decision tree," Proc. of the DARPA Image Understanding Workshop, Morgan Kaufman, 1988.

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

[48] Wilkinson, R.A., Geist, J., Janet, S., Grother, P., Gurges, C., Creecy, R., Hammond, B., Hull, J., Larsen, N., Vogl, T., and Wilson, C., "The first census optical character recognition system conference," Nat. Inst. of Standards and Technol. Tech. Rep. no. NISTIR 4912, Gaithersburg, MD, Aug. 1992.

[49] Zimmerman, S., "An optimal search procedure," Am. Math. Monthly, 66, 690-693, 1959.