Geometric Decision Trees for Optical Character Recognition

(Extended Abstract*)

George N. Sazaklis† Esther M. Arkin‡ Joseph S. B. Mitchell§

Steven S. Skiena¶

State University of New York, Stony Brook, NY 11794

Abstract

A fundamental problem in computer vision is identifying which of a given set of geometric models is present in an image. We consider an approach to model recognition based on computing efficient strategies (*decision trees*) for "probing" a scanned image of a typeset document, in order to perform fast and effective optical character recognition (OCR). We consider a "probe" to be a simply computed local operator that can be applied to discriminate between two sets of possible models. By carefully constructing effective probes, and assembling them into a geometric decision tree, we have devised, implemented, and compared a variety of methods to perform OCR. In this paper, we present algorithms for probing strategies and decision tree construction, and we report experimental results on the effectiveness of these algorithms in identifying English characters and numerals in scanned images of printed pages of text. These algorithms are implemented as part of a system used by a document processing company (Syngen Corp.).

1 Introduction

The field of optical character recognition (OCR) strives to build systems enabling a machine to "read" pages of printed text that have been scanned into a digital file format. Extensive research has been invested in OCR since the 1960's, but the problem is still largely unsolved. Many commercial OCR packages exist, but most of them encounter problems in multi-font or low-quality documents. Moreover, most approaches are based on statistical decision theory that requires a set of training data to "tune" the classifier. However, such an extensive training set is not always available, as when one is scanning a book, or working on a document with an unknown font.

We report on our experience doing OCR using geometric probes, arranged in a *decision tree classifier*. Each probe consists of a simple local operator (e.g., summing the intensity values of the set of pixels that lie within a selected small box), and a comparison of the resulting value with threshold values that determine which branch of the decision tree is to be taken next.

This work can be considered a follow-up to Arkin et al. [1, 2], who studied the geometric decision tree problem. Here, we investigate the practical aspects of applying geometric decision trees to a real problem (OCR) and provide the experimental analysis that permits them to be further developed for other applications.

The "probing paradigm" has been applied to model-based object recognition; see [6], and references cited therein. In effect, the probing schemes serve to "factor out" the effect of translation and rotation, reducing the final decision problem to that of this paper. For previous work on the decision tree classifier paradigm, see [3, 4, 11]. For general background on the extensive field of OCR, we refer the reader to the survey by Mori, Suen and Yamamoto [7]. The method we are employing most closely resembles the "peephole" method that was among the first approaches to the OCR problem [5].

Main Results

(1). We formalize the decision tree methodology, as applied to OCR, and describe and analyze heuristic algorithms designed to produce decision trees having low height. In particular, we conduct an experimental investigation into the methods based on greedy set cover heuristics, which were introduced in previous theoretical work of Arkin et al. [1, 2].

(2). Our decision-tree construction algorithm is based on an error model, which provides us with the probabilistic information required to assess the reliability of local probes. A detailed description of our error model is provided in the full paper.

(3). We develop a method of *verifying* a hypothesis given by a recognizer, in order to make more stringent demands on the level of certainty. Our verifier is based on set cover heuristics, and uses a select set of probes in order to raise the recognition confidence.

(4). We report on experimental results and on the practical effectiveness of geometric decision trees for use in scanned document processing. Our implementation is part of a system being tested now for use within a full-scale production environment to scan and read consumer surveys, financial account documents, and other forms.

*Full paper at http://ams.sunysb.edu/~jsbm/jsbm.html

†sazaklis@cs.sunysb.edu; Dept. Computer Science. Supported in part by a grant from Syngen Corp. and by SPIR, College of Engineering and Applied Science, SUNY Stony Brook.

‡estie@ams.sunysb.edu; Dept. Applied Math & Statistics.

§jsbm@ams.sunysb.edu; Dept. Applied Math & Statistics.

¶skiena@cs.sunysb.edu; Dept. Computer Science.

Computational Geometry 97 Nice France

2 Methodology

We assume that each model in the library S is given in a fixed position, orientation, and scale, as a greyscale array of pixels. From the set of models, we construct a decision tree (a *probe tree*), which can be used to perform recognition of scanned images. We assume that the scanned image has been segmented into subimages that each contain exactly one instance of one model. While segmentation can itself be a challenging task, for this work we chose to concentrate on classification and keep the segmentation issue separate.

As in [1, 2], we consider a *probe* to be a function that maps an image to a real number, its *outcome*, typically by computing a simple local function of the image values in one or more selected locations or neighborhoods within the image. The probe outcome must be simple to compute. By normalization, we can assume that all outcome values are in [0, 1]. The outcome of a probe when applied on a model is a random variable. The support interval of its distribution is determined by the error model and is called the *outcome interval*.

At each node v of the probe tree there is an associated subset $S_v \subseteq S$ of the models. In the tree construction algorithm, our goal is to partition S_v into roughly equal-size subsets by means of selecting an effective probe, π_v, to apply at node v. We determine one or more *classification thresholds* for each probe that we use. The classification thresholds partition [0, 1] into intervals, each of which corresponds to a child of the current node. While the thresholds partition the real numbers, we may not end up with a partition of the models S_v: Some models may be assigned to two or more children of v. In particular, since the outcome interval I_m for a model m contains the set of values that π_v might produce, m is included in any child whose interval overlaps with I_m. The optimal classification intervals for each candidate probe, together with the model subsets for each child are computed using dynamic programming. The "best" split is a set of k points (with $k < 5$, typically) in the interval arrangement that generate $k+1$ children, while minimizing the model overlap between different children. We have found that minimizing the objective function $\sum_{k=1}^{n} |S_i|^3$ is an effective means of minimizing model overlap, where S_i denote the model sets of the children. (While here we use a sum of cubes, any sum of convex functions will suffice). A dynamic programming algorithm ($O(n^2)$) is used to select the optimal classification intervals.

We then apply a *greedy* strategy, selecting the probe whose corresponding split of the models is the most balanced. The greedy heuristic for constructing decision trees was discussed and analyzed in [2], where it was shown that the greedy strategy to split a set of models produces a tree within a $\lceil \lg k \rceil$ factor of optimal, when we have $|M| = k$ models. While greedy probes produce short trees, there is a tradeoff between partitioning and accuracy. Currently, we choose the most reliable probe among those that are greedy enough (have balanced children).

The resulting decision tree can then be used for character classification. As we see our tree growing strategy is top-down, because we want to use only local information to make a decision on a tree node.

During classification, we descend to the child (or children) of v whose interval contains the outcome of probe π_v.

Currently we use four distinct types of probes: *Bounding Box* probes measure the width or the height of the bounding box of the character. *Single-pixel* probes consist of a lone pixel, while *Rectangle Probes* are a generalization of single-pixel probes. with the probe calculating the average intensity over a rectangular area. Finally, *Pixel Set* probes are even more general. They may consist of any collection of pixels, equipped with signs (to specify whether the pixel intensity has to be inverted before taking the average). The *Pixel Set* type can be used to "home in" on where the essential geometrical differences are between two shapes.

3 Verifier

Because of distortions in the input image, a local operator may make mistakes during recognition. We can overcome this danger, however, if we use the redundancy present in the character image, to *verify* the classifier's suggestion.

The concept of *verification* is to use additional probes in order to obtain further evidence supporting the suggestion made by the decision tree.

To verify a suggestion m, made by the probe tree, we apply a sequence of probes to the sample, that can distinguish between model m and any other model in our alphabet. We say a probe p *covers* model c against c', if and only if it can distinguish between c and c'. For probe p to be used as a verification probe between c and c', it has to cover c against c' and be independent (disjoint) of any probes used so far in the decision tree for c, so that it does not repeat any errors made by the tree probes.

To build our verifier, we construct the so called *ambiguity graph*. The nodes are the models, while there is an edge between two models, if they are not discriminated by the current set of verifying probes. The edge is labeled by those probes that can do the discrimination, and break the ambiguity. This way, the verifier construction problem is transformed into a set covering problem. To construct the verifier with the minimum number of probes, we apply the greedy heuristic, selecting the probe that breaks the highest number of unresolved ambiguities.

As a different verifier variety, we can have a separate set of probes for each model, producing many set covers, independent from one another. The second flavor is called the "multiple cover" verifier.

Finally, we can ask for multiple coverage ("C-coverage") to increase the confidence of the final decision, where C is a parameter that determines the minimum number of probes needed to break the ambiguity between two models.

4 Experimental Results

Our experiments were done on two fonts, OCR-font and Times-Roman 10pt. The OCR-font data was *real* data provided to us by Syngen Corp. As the OCR-font was designed to be machine-readable, it draws the numbers using only thick, constant width, horizontal and vertical lines, except "7", which has a tilted stroke. The experiment was conducted on 5183 number sequences, each segmented into 7 numerals, for a total of 36281 characters. The results for each tree individually without the verifier and for the system as a whole (with verification) appear in Table 1.

	Rec. Rate	Error Rate	Rejections
Tree1 Only	99.94 %	0.05 %	0.01 %
Tree2 Only	99.96 %	0.03 %	0.00 %
Whole system	99.87 %	0.00 %	0.13 %

Table 1: Rates for OCR-font

Decision Tree	Verifier	Originals		Photocopy		2nd Generation		Fax	
		Rec.	Error	Rec.	Error	Rec.	Error	Rec.	Error
Single Pixel	Y	99.87 %	0.00 %	96.20 %	0.00 %	72.57 %	0.00 %	42.95 %	0.07 %
	N	99.90 %	0.10 %	98.87 %	1.12 %	92.63 %	6.83 %	82.73 %	16.48 %
Pixel Set	Y	99.15 %	0.00 %	94.03 %	0.00 %	80.99 %	0.00 %	55.32 %	0.20 %
	N	99.84 %	0.13 %	98.13 %	1.85 %	94.04 %	5.74 %	76.17 %	19.76 %

Table 2: Experimental results for the Times-Roman font, both with ("Y") and without ("N") the verifier.

Trees	1-cover Ambiguity Graph		2-cover Ambiguity Graph		1-cover Mult. Verifiers		2-cover Mult. Verifiers	
	CR	WA	CR	WA	CR	WA	CR	WA
Original Print								
Single Pixel	0.03 %	0.00 %	0.12 %	0.00 %	0.02 %	0.00 %	0.30 %	0.00 %
Pixel Set	0.43 %	0.00 %	0.68 %	0.00 %	1.89 %	0.00 %	0.77 %	0.00 %
Photocopy (1st)								
Single Pixel	2.48 %	0.00 %	5.42 %	0.00 %	4.92 %	0.00 %	5.82 %	0.00 %
Pixel Set	4.16 %	0.00 %	6.20 %	0.00 %	7.39 %	0.00 %	8.55 %	0.00 %
Photocopy (2nd)								
Single Pixel	9.09 %	0.96 %	34.79 %	0.03 %	21.56 %	0.00 %	24.00 %	0.00 %
Pixel Set	10.67 %	0.07 %	13.81 %	0.00 %	40.37 %	0.00 %	24.92 %	0.00 %

Table 3: Verification rates for different strategies on originals, copies, and second-generation copies.

Figure 1: Left to right: Original, first-generation copy, second-generation copy, faxed.

Times-Roman, on the other hand, presents a greater challenge. As a font, it is considered non-trivial to be processed by OCR systems, as it has stroke width variability and serifs. We chose a size of 10pt, as it is more challenging than the more commonly used size of 12pt.

Printed pages with uppercase, lowercase letters as well as numbers were scanned on a flatbed Ricoh IS-60 scanner at 300 dpi, as 8-bit greyscale images. We also scanned first and second-generation photocopies, as well as faxed images to test our system on badly distorted documents. In Figure 1 we present typical examples of characters for our 4 document quality categories. We see that the probe tree has to deal with quite disparate distortions on the 4 documents: On photocopied images, individual pixels are flipped, smudges make recognition difficult, while defects accumulate in higher generations. Faxed data is the most challenging, since many characters are elongated vertically, due to the mechanical inaccuracies of the sending device, which scans the page; the image also suffers from ink dispersion that occurs at the receiving end.

Character models and probe trees were constructed as was outlined previously. Table 2 displays the recognition and error rates for different choices of probe trees and sources of scanned images. The remaining percentage in each case is due to rejections either from the tree or the verifier. The sample size for these experiments was 15692 characters, distributed equally among the 26 uppercase, the 26 lowercase and the 10 numerals.

Table 3 reports our verification rates for different verifier choices, different probes and sources of scanned data. In each table cell, two error measures are shown: The percentage of Correct but Rejected (CR) samples as well as the percentage of Wrong but Accepted (WA) samples. Our verifiers tend to be aggressive with rejections, since the cost of accepting a wrong classification is much higher than rejecting a correct one.

References

[1] E. Arkin, M. Goodrich, J. Mitchell, D. Mount, C. Piatko, and S. Skiena. Point probe decision trees for geometric concept classes. In *Proc. 3rd WADS*, pp. 95–106, 1993.

[2] E. Arkin, H. Meijer, J. Mitchell, D. Rappaport, and S. Skiena. Decision trees for geometric models. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, 369–378, 1993.

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

[4] P. Chou. Optimal partitioning for classification and regression trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):340–354, 1991.

[5] ERA. An electronic reading automaton. *Electronic Engineering*, pages 189–190, April 1957.

[6] R. Freimer, S. Khuller, J. Mitchell, C. Piatko, K. Romanik, and D. Souvaine. Localizing an object with finger probes. In R. Melter and A. Wu, eds., *Proc. of Vision Geometry III*, pp. 272–283, SPIE, Nov., 1994.

[7] S. Mori, C. Suen, and K. Yamamoto. Historical review of OCR research and development. *IEEE Proceedings*, 80:1029–1058, July 1992.

[8] S. Mori, K. Yamamoto, and M. Yasuda. Research on machine recognition of handprinted characters. *IEEE Trans. PAMI*, 6:386–405, 1984.

[9] G. Nagy. State of the art in pattern recognition. *Proc. IEEE*, 56:836–860, 1968.

[10] K. Romanik and C. Smith. Testing geometric objects. *Comput. Geom. Theory Appl.*, 4:157–176, 1994.

[11] S. Safavian and D. Landgrebe. A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(3):660–674, 1991.