Table 5: Summary of time/space demands when 100,000 prototypes are used.

classifier	prototype/feature storage	run time per image
nearest-neighbor	28,800 K bytes	18 seconds
decision tree	166 K bytes	0.024 seconds
perfect metrics	1,110 K bytes	0.096 seconds

Table 6: Distribution on the parameters of the image defect model.

Parameter	Distribution	Units
randomized per-character ...		
`resn`	fixed (= 300)	pixels/inch
`size`	fixed (= 5,7,9,11,13 for training, 6,8,10,12,14 for testing)	points (1/72 inch)
`blur` standard error of the Gaussian blurring kernel	normal ($\mu = 0.7, \sigma = 0.3$)	pixels
`thrs` binarization threshold	normal ($\mu = 0.25, \sigma = 0.04$)	intensity
`skew` skew	normal ($\mu = 0, \sigma = 1.33$)	degrees
`xscl` horizontal scaling	uniform in [0.85,1.15]	dimensionless
`yscl` vertical scaling	normal ($\mu = 1, \sigma = 0.0167$)	dimensionless
`xoff` horizontal translation	uniform in [-0.5,0.5]	pixels
`yoff` vertical translation	normal ($\mu = 0, \sigma = 0.06$)	ems
randomized per-char and per-pixel ...		
`sens` pixel sensor sensitivity	normal ($\mu = 0.125, \sigma = 0.04$)	intensity
`jitt` jitter	normal ($\mu = 0.2, \sigma = 0.1$)	pixels

[17] I.K. Sethi, G.P.R. Sarvarayudu, Hierarchical Classifier Design Using Mutual Information, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI–4**, 4, July 1982, pp. 441–445.

[18] V. Vapnik, *Estimation of Dependences Based on Empirical Data*, Springer–Verlag, 1982.

[19] Q.R. Wang, C.Y. Suen, Large Tree Classifier with Heuristic Search and Global Training, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI–9**, 1, January 1987, pp. 91–102.

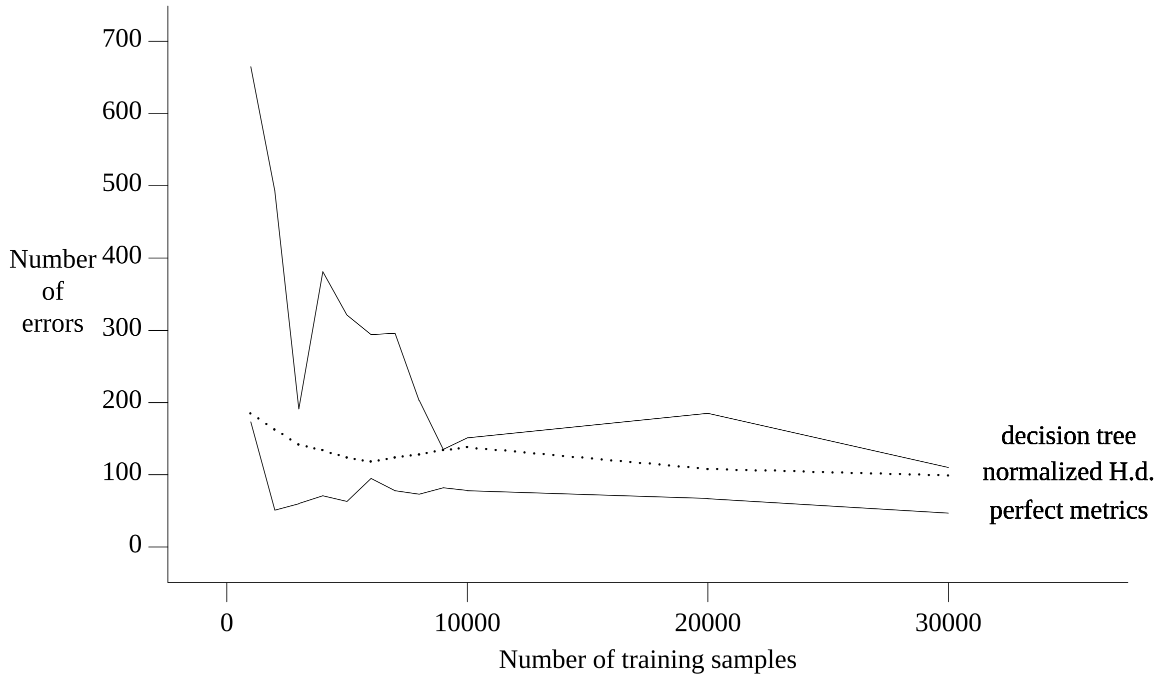

Figure 10: Comparison of number of errors by three classifiers. Rejects are not included.

on Pattern Analysis and Machine Intelligence, **PAMI–9**, 1, January 1987, pp. 103–112.

[8] G.W. Gates, The Reduced Nearest Neighbor Rule, *IEEE Transactions on Information Theory*, **IT–18**, May 1972, pp. 21–27.

[9] P.E. Hart, The Condensed Nearest Neighbor Rule, *IEEE Transactions on Information Theory*, **IT–14**, May 1968, pp. 515–516.

[10] T.K. Ho, H.S. Baird, Perfect Metrics, *Proceedings of the Second International Conference on Document Analysis and Recognition*, Tsukuba Science City, Japan, October 20–22, 1993, pp. 593–597.

[11] M.W. Kurzynski, The Optimal Strategy of a Tree Classifier, *Pattern Recognition*, **16**, 1, 1983, pp. 81–87.

[12] B.M.E. Moret, Decision Trees and Diagrams, *ACM Computing Surveys*, **14**, December 1982, pp. 593–623.

[13] S.V. Rice, J. Kanai, T.A. Nartker, An Evaluation of OCR Accuracy, in *Information Science Research Institute, 1993 Annual Research Report*, University of Nevada, Las Vegas, 1993, pp. 9–20.

[14] G.L. Ritter, H.B. Woodruff, S.R. Lowry, T.L. Isenhour, An Algorithm for Selective Nearest Neighbour Decision Rule, *IEEE Transactions on Information Theory*, **IT–21**, November 1975, pp. 665–669.

[15] M. Sabourin, A. Mitiche, D. Thomas, G. Nagy, Hand–Printed Digit Recognition using Nearest Neighbour Classifiers, *Proceedings of the Second Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, April 26–28, 1993, pp. 397–409.

[16] J. Schuermann, W. Doster, A Decision Theoretic Approach to Hierarchical Classifier Design, *Pattern Recognition*, **17**, 3, 1984, pp. 359–369.

Table 4: Results of perfect–metric classification on test set.

training set	number of prototypes (,000)	number of features	truth = c total = 50,000		truth = e total = 50,000		Total total = 100,000	
			# err	# rej	# err	# rej	# err	# rej
1	1	5	34	204	139	202	173	406
2	2	6	14	66	37	140	51	206
3	3	7	19	13	41	112	60	125
4	4	9	34	8	37	50	71	58
5	5	14	16	10	47	27	63	37
6	6	22	31	14	64	12	95	26
7	7	32	26	6	52	22	78	28
8	8	45	15	1	58	12	73	13
9	9	32	29	7	53	16	82	23
10	10	54	20	2	58	16	78	18
11	20	470	14	6	53	47	67	53
12	30	761	16	11	31	42	47	53

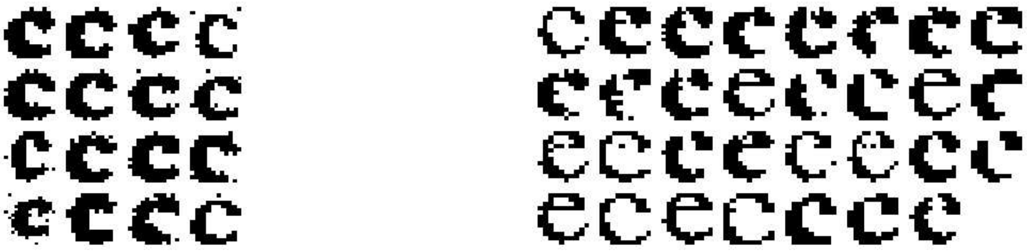

Figure 8: Images misclassified when 30,000 prototypes were used to construct a perfect metric (out of the 100,000 tested, shown normalized). On the left: samples of 'c's; right: samples of 'e's.

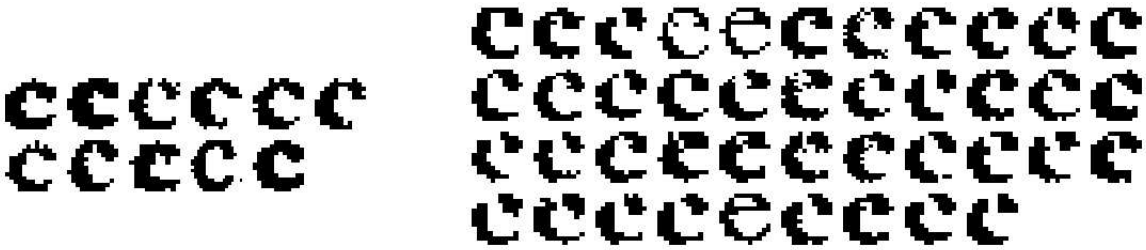

Figure 9: Images that were considered ambiguous when 30,000 prototypes were used to construct a perfect metric (out of the 100,000 tested, shown normalized). On the left: samples of 'c's; right: samples of 'e's.

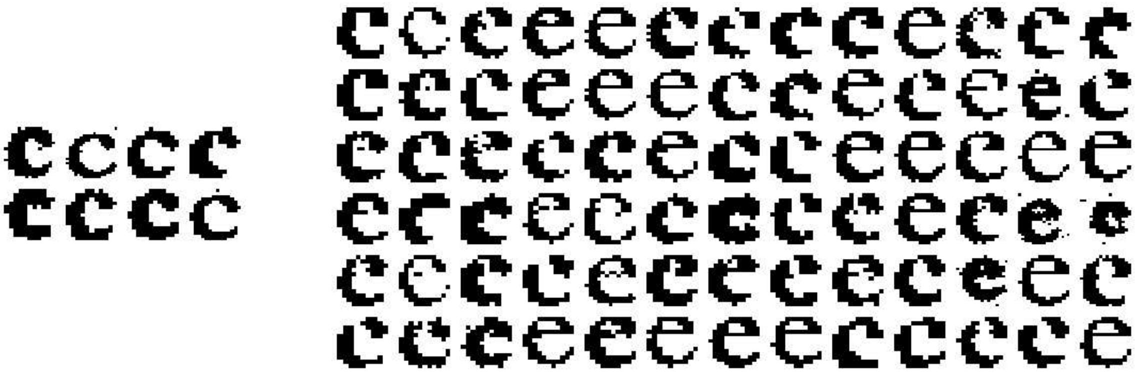

Figure 7: Images misclassified when 60,000 prototypes were used to derive a decision tree (out of the 100,000 tested, shown normalized). On the left: samples of 'c's; right: samples of 'e's.

Appendix

The defect model parameters (and their units) include: `size`, the nominal text size of the output (units of points); `resn`, "resolution," the output spatial sampling rate (pixels/inch); `skew`, rotation (degrees); `xscl` and `yscl`, multiplicative scaling factors (horizontally and vertically); `xoff` and `yoff`, translation offsets (pixels); `jitt`, jitter, the distribution of per–pixel discrepancies of the pixel sensor centers from an ideal square grid; `blur`, defocusing, modeled as a Gaussian point–spread function's standard error (pixels); `sens`, sensitivity, the distribution of per–pixel additive noise, expressed as the standard error of a normal distribution with zero mean, (intensity units); and `thrs`, the binarization threshold (intensity). When the model is simulated, the parameters take effect in the order given above: the ideal input image is first rotated, scaled, and translated; then the output resolution and per–pixel jitter determine the centers of each pixel sensor; for each pixel sensor the blurring kernel is applied, giving an analog intensity value to which per–pixel sensitivity noise is added; finally, each pixel's intensity is thresholded, giving the output image. Table 6 summarizes the distribution on the parameters used in this experiment.

References

[1] H.S. Baird, R. Fossey, A 100–Font Classifier, *Proceedings of the first International Conference on Document Analysis and Recognition*, St.–Malo, France, September 20–October 2, 1991, pp. 332–340.

[2] H.S. Baird, Document Image Defect Models, in H.S. Baird, H. Bunke, K. Yamamoto (Eds.), *Structured Document Image Analysis*, Springer–Verlag, 1992.

[3] H.S. Baird, Document Image Defect Models and Their Uses, *Proceedings of the Second International Conference on Document Analysis and Recognition*, Tsukuba Science City, Japan, October 20–22, 1993, pp. 62–67.

[4] R.G. Casey, G. Nagy, Decision Tree Design Using A Probabilistic Model, *IEEE Transactions on Information Theory*, **IT–30**, 1, January 1984, pp. 93–99.

[5] T.M. Cover, P.E. Hart, Nearest Neighbor Pattern Classification, *IEEE Transactions on Information Theory*, **IT–13**, 1, January 1967, pp. 21–27.

[6] J.H. Friedman, J.L. Bentley, R.A. Finkel, An Algorithm for Finding Best Matches in Logarithmic Expected Time, *ACM Transactions on Mathematical Software*, **3**, 3, September 1977, pp. 209–226.

[7] K. Fukunaga, D.M. Hummels, Bias of Nearest Neighbor Error Estimates, *IEEE Transactions*

Table 3: Results of decision–tree classification on test set.

training set	number of prototypes (,000)	number of internal nodes	maximum number of levels	truth = c total = 50,000		truth = e total = 50,000		Total 100,000	
				# err	# rej	# err	# rej	# err	# rej
1	1	3	3	30	0	635	0	665	0
2	2	4	3	16	0	477	0	493	0
3	3	6	3	22	0	169	0	191	0
4	4	6	3	17	0	364	0	381	0
5	5	7	4	36	0	285	0	321	0
6	6	8	4	38	0	256	0	294	0
7	7	10	5	30	0	266	0	296	0
8	8	10	4	24	0	179	0	203	0
9	9	8	4	15	0	120	0	135	0
10	10	8	4	13	0	138	0	151	0
11	20	18	6	16	0	169	0	185	0
12	30	25	7	10	0	100	0	110	0
13	40	28	7	14	0	81	0	95	0
14	50	31	7	14	0	94	0	108	0
15	60	39	8	8	0	78	0	86	0

ble 5 summarizes their space and time demands (when training set 10 with 100,000 training prototypes was used). The run time was measured on a Silicon Graphics Computer Systems Power Series Model 4D/480S.

The accuracies achieved (99.9%) are remarkably high, considering the well–known practical difficulty of the problem. Apparently the perfect metrics method was more accurate than the nearest–neighbors and decision–tree methods. However, if we count rejects as errors, the differences were not significant[3] at 95% statistical confidence.

It would be surprising if this remarkable consistency were a coincidence: therefore, we speculate that the asymptotic accuracy of all three methods is determined more by the characteristics of the training data that they shared, than by the details of their methodology which differ. In other words, we believe that, as long as the training data are representative and sufficiently many, a wide range of classifier technologies can be trained to equally high accuracies. Moreover, impractically expensive computing resources need not be required for this to succeed.

It should be borne in mind that some aspects of the classifiers' performance may depend largely on the nature of the imaging defects. For instance, an asymmetry in errors could be due to the defect model rather than the classifiers: if defects such as scratches and coffee stains were modeled, we might have seen more 'c's classified as 'e's.

We have noted that there is little overlap among the errors made by the classifiers. This suggests that further performance improvements might be possible, for example, through combining their results.

Our results suggest a promising two–part research strategy to build highly accurate classifiers. On one hand, more research is called for to develop image defect models that fit reality as closely as possible. On the other hand, we should further investigate methods for constructing high–performance classifiers given the precise problem definitions that image defect models allow.

Acknowledgements

Stimulating comments by David Ittner, George Nagy, and Dz–Mou Jung are gratefully acknowledged.

[3] We estimate the 95% statistical confidence intervals of errors plus rejects, after training on 30,000 samples and testing on 100,000 samples, to be as follows (in per cent): nearest–neighbors with normalized Hamming distance, [0.08,0.12]; decision trees, [0.09,0.13]; and perfect metrics, [0.08,0.12]. Note that each pair of intervals overlaps.

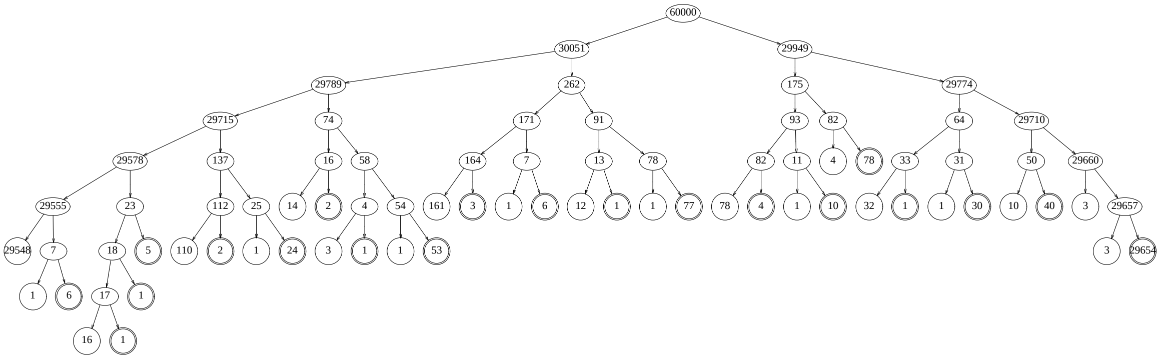

Figure 6: Decision tree generated with training set 15. The number inside each node indicates how many training samples are assigned to that node. The terminal nodes containing 'c's are drawn as circles and those containing 'e's double circles.

the training samples are in a single undifferentiated set. A feature is generated and all training samples that it discriminates are removed from the set. The algorithm iterates until all training samples of each class are discriminated or no more features can be found to discriminate any of the remaining samples. [2] In the resulting perfect–metric classifier all the features are evaluated (by contrast to the decision–tree classifier, in which only a fraction are evaluated).

Using this method, we obtained a classifier for each of the training sets 1 through 12. Table 4 summarizes their performance on the test set. We treat all ambiguities as rejects.

Errors and rejects (ambiguities) decline rapidly through 5000 prototypes, and then stabilize at values lower than those for nearest–neighbors and decision trees. The asymmetry in errors is also less obvious. Of the 47 errors, 22 are identical to those in Figure 4 and 17 are identical to those in Figure 7. Only three errors are common to all three methods.

Figure 8 shows the errors of the classifier trained with 30,000 prototypes (training set 12) and Figure 9 shows the rejects.

6 Conclusions

We have experimentally estimated the asymptotic accuracy of three trainable classification methods, as applied to the same precisely specified recognition problem. Through the use of a parameterized image defect model, we were able to supply a training set that was representative and of unlimited size.

In all three methods, the capacity of the classifier was permitted to expand during training. For all three, under training with larger and larger training sets, accuracy rose to an apparent asymptote (Figure 10); this occurred more rapidly for some methods than others. None of the methods required exorbitant time or space resources to approach the asymptote closely.

In the nearest-neighbor trials, the estimated accuracy seems to approach an asymptote, clearly enough; but, we would prefer to support this sort of subjective judgement with a principled statistical analysis. In other future work, we hope to extend the decision-tree and perfect-metric trials to larger training sets, to increase our confidence that we have observed their asymptotic behavior.

In other important respects, however, such as time/space demands at runtime, the three classification methods are quite dissimilar. Ta-

[2] This occurs when the two classes share some identical samples.

will see, they are not exorbitant). We also do not aim to construct worst–case optimal trees; rather, we choose a greedy heuristic that is sensitive to the quality of training data and is otherwise simple to implement. It is, like most such heuristics, correct on the training set by construction.

We construct a deterministic, non–backtracking, binary classification tree in which each interior node owns a linear discriminant function, and each leaf owns a single class ('e' or 'c'). At the outset, the tree consists of a single leaf which owns all the training data.

Each "mixed" leaf (owning more than one class) is split into an interior node and two leaves. This proceeds recursively until all leaves own training data all of a single class.

A mixed leaf owns two nonempty sets c_1 and c_2 of training samples. We compute the sample mean m_i of each class c_i ($m_i \in \mathbf{R}^{48\times48}$). A line is drawn from m_1 to m_2. The family of hyperplanes $\{h_1, h_2, ..., \}$ perpendicular to this line, parameterized by their distances to m_1 ($d(m_1, h_j)$), are then examined. The parameter is quantized by fixed increments of $0.05 \times ||m_2 - m_1||$. For each hyperplane h in this family, the error

$$\begin{aligned} e_h &= |\{x | x \in c_1 \wedge d(x,h) > 0\} \cup \\ &\quad \{x | x \in c_2 \wedge d(x,h) \leq 0\}| \end{aligned}$$

is calculated, and the h with minimum error e_h is chosen.

Figure 6 shows the tree constructed using training set 15. A new tree was built, from scratch, for each training set. Table 3 summarizes the results of classification for the test set using these trees.

For small training sets, the number of errors is much larger than for nearest–neighbors. However, accuracy improves quickly. Comparing the results in line 15 of Tables 3 and 2, we can see that the accuracy is comparable to that of nearest–neighbor matching using normalized Hamming distance. Because of insufficient computational resources, we have not yet grown bigger trees. Nevertheless, it is clear that the increase in training samples helps improve the generalization power of the trees.

As before, more 'e's than 'c's are misclassified. The figures in the last few lines of Table 3 agree closely with the apparent limit in Table 2. Yet only 32 of the 86 errors, shown in Figure 7, are identical to those in Figure 4 (see Conclusions).

5 Perfect Metrics

In [10] we have described a method for constructing "perfect metrics" for character classification. The metric $d(x, c) \geq 0$ measures the dissimilarity of an image x to a (description of a) class c (note: it does not compare two images). We call such a metric "perfect" on a set if, for all images x in the set, and all classes c, $d(x, c) = 0$ iff x is in class c. In practice, we can construct metrics which are perfect on the training set provided that there is no ambiguity. Ambiguity arises when there is an x such that $d(x, c) = 0$ for more than one class.

The metrics could be less than perfect on test sets. We say that a metric is perfect on a set S with probability p if p equals the fraction of images $x \in S$ for which $d(x, c) = 0$ iff x is in class c for all c. To achieve a high probability of perfection on the test set our experience suggests that it is necessary (but not sufficient) that the number of training samples is much greater than the number of distinct values taken on by any feature.

The metric is represented as a "distribution map" where the occurrence in the training set of each value of each feature is explicitly recorded. These classifiers tend to be more prone to ambiguities than to errors. In an extreme example of this, if all values of all features occur for all classes, then all images are ambiguous.

Ambiguities will be completely eliminated if the following condition is satisfied: for each pair of classes c_i and c_j, there exists at least one feature whose distribution maps for c_i and c_j do not overlap. Such a feature set may be difficult or impossible to find, either manually or automatically. A more easily satisfied, but less efficient, condition is: for each pair of classes c_i and c_j, and for each x in c_i, there exists at least one feature in which x has a value that does not occur in the distribution map of c_j. We say that such a feature is an *i, j–discriminating feature for x*.

We experimented with the following heuristic to discover discriminating features for classes 'e' and 'c'. Quantized distances to hyperplanes — as in the above discussion for decision trees — were used as features. At the outset, all

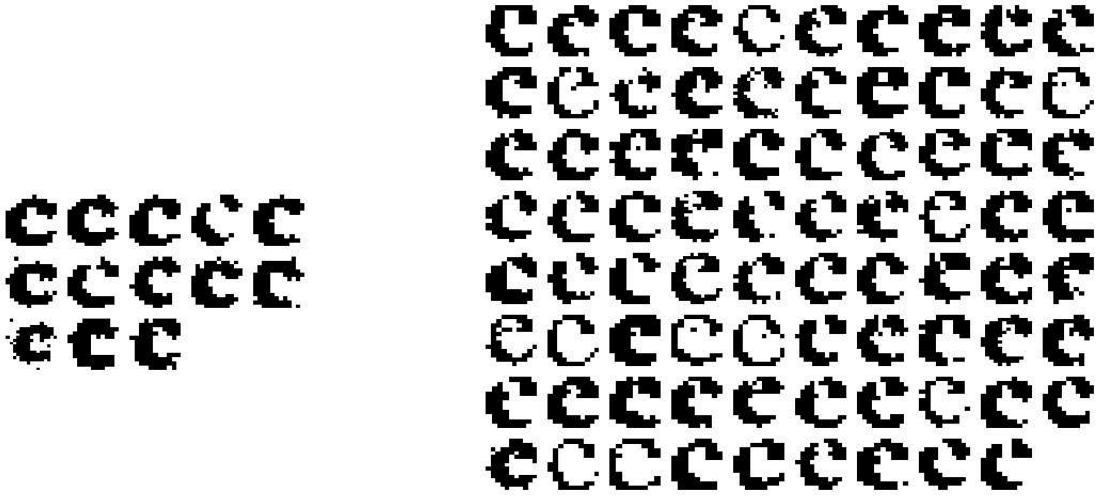

Figure 4: Images misclassified using 500,000 prototypes and the normalized Hamming distance (out of 100,000 tested, shown normalized). On the left: samples of ‘c’s; right: samples of ‘e’s.

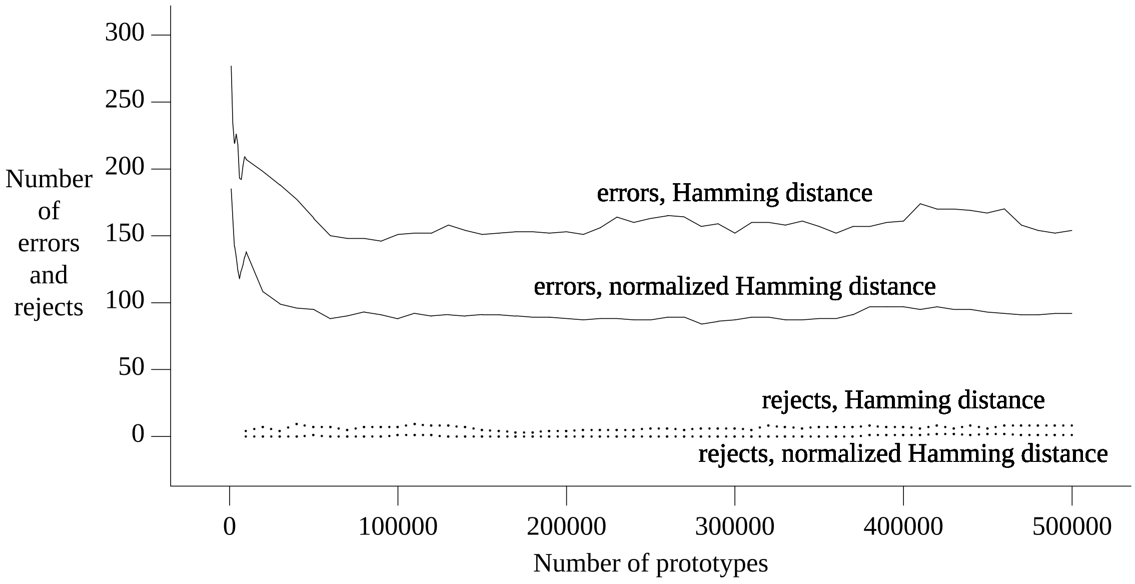

Figure 5: Nearest–neighbor matching: number of errors and rejects versus number of prototypes.

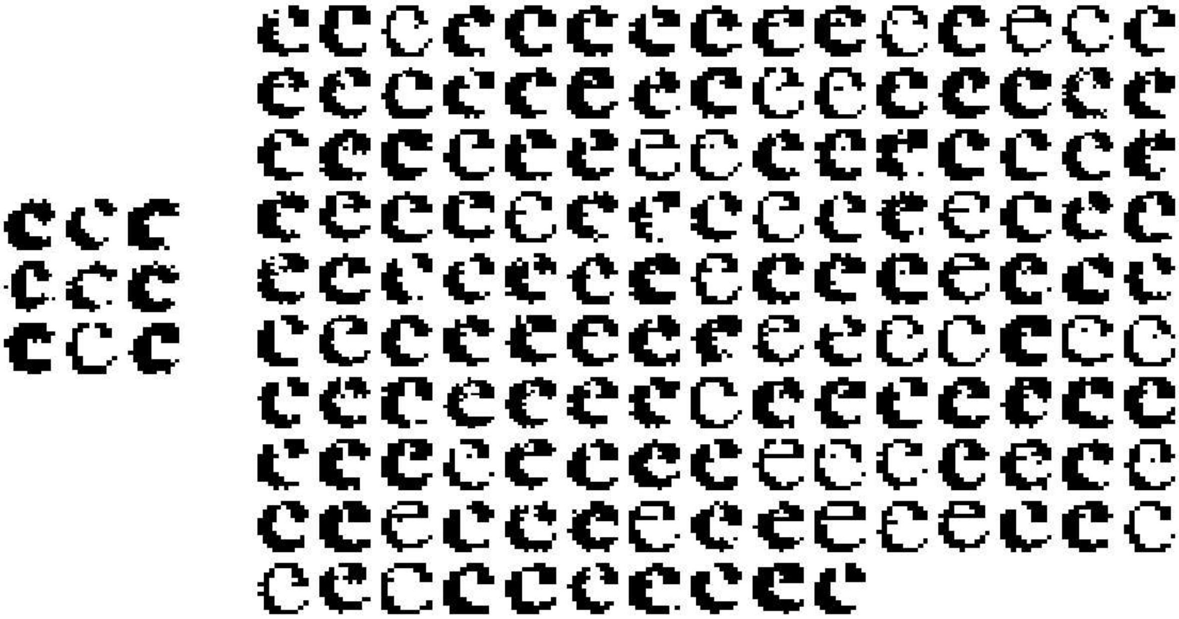

Figure 2: Images misclassified using 500,000 prototypes and Hamming distance (out of 100,000 tested, shown normalized). On the left: samples of ‘c’s; right: samples of ‘e’s.

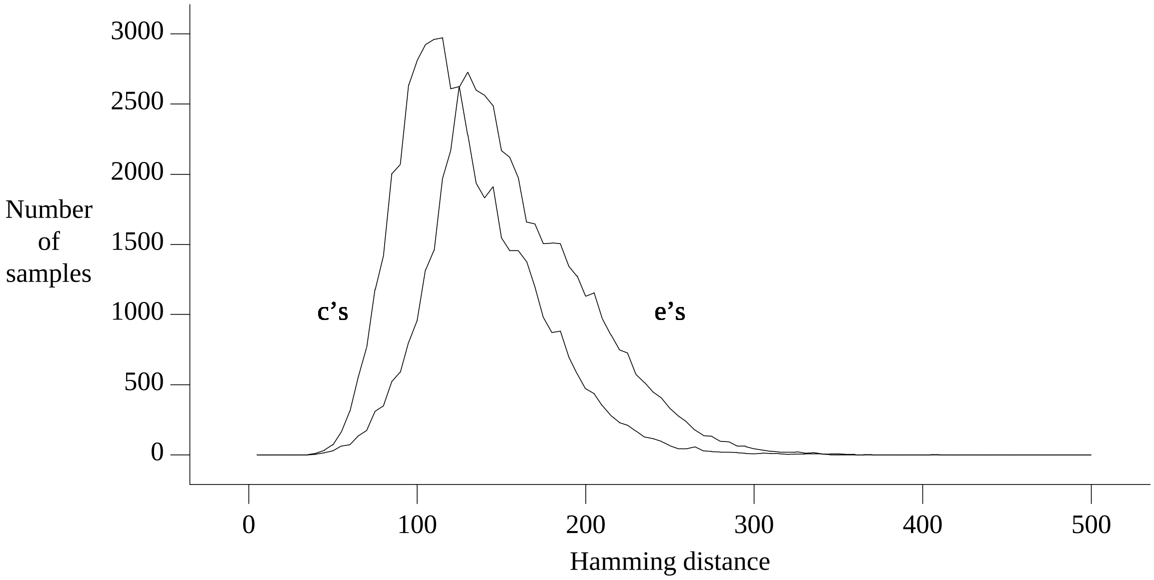

Figure 3: Distribution of Hamming distance between a test sample and its nearest neighbor in the training set.

verify this, the distance between each test sample and its nearest neighbor in the training set was recorded. The distribution of the distances for each class is shown in Figure 3. Indeed, this shows that 'e's are more widely scattered, under Hamming distance, than 'c's.

No clear downward trend is visible after 90,000 prototypes. This might suggest that 99.8% is an upper bound on accuracy possible using this training set. However, results from the second test show that this is not the case. In the second test, the same features were used but the metric was changed to the normalized Hamming distance (H.d. divided by the number of black pixels in the prototype). The results are shown in Table 2 and Figure 4 respectively. The number of errors is compared to that of the first test in Figure 5.

Table 2: Results of nearest–neighbor matching on an independent test set using normalized Hamming distance.

training set	# prototypes (,000)	truth = c 50,000		truth = e 50,000		Total 100,000	
		# err	# rej	# err	# rej	# err	# rej
1	1	90	0	95	2	185	2
2	2	78	1	85	1	163	2
3	3	60	2	82	1	142	3
4	4	53	2	81	2	134	4
5	5	44	2	80	0	124	2
6	6	47	1	71	0	118	1
7	7	48	0	76	0	124	0
8	8	45	0	83	0	128	0
9	9	47	0	87	0	134	0
10	10	48	0	90	0	138	0
11	20	24	0	84	0	108	0
12	30	23	0	76	0	99	0
13	40	19	0	77	0	96	0
14	50	19	0	76	1	95	1
15	60	21	0	67	0	88	0
16	70	20	0	70	0	90	0
17	80	20	0	73	0	93	0
18	90	20	0	71	0	91	0
19	100	19	0	69	1	88	1
20	110	23	0	69	1	92	1
21	120	22	0	68	1	90	1
22	130	21	0	70	0	91	0
23	140	20	0	70	0	90	0
24	150	18	0	73	0	91	0
25	200	17	0	71	0	88	0
26	250	16	0	71	0	87	0
27	300	15	0	72	0	87	0
28	350	16	0	72	0	88	0
29	400	16	0	81	1	97	1
30	450	15	0	78	2	93	2
31	500	13	0	79	1	92	1

There are, on average, only 3/5 as many errors as in the first test. There are also many fewer rejects. Otherwise, roughly similar behavior is observed: the number of errors quickly decreases up to about 60,000 prototypes, and thereafter shows no clear downward trend. So, an accuracy of 99.9% seems to be the best achievable with 500,000 prototypes under normalized Hamming distance. It is of course possible that, given even larger numbers of prototypes, some improvement would be possible.

The change of metric had a greater effect on accuracy than an order–of–magnitude enlargement of the training set. We feel that adding more prototypes, chosen from the given distribution, is unlikely to overwhelm the advantage of using the better metric.

However, we have conjectured that we might be able to improve accuracy substantially, even for an inferior metric, by adding prototypes chosen from a special distribution: where the errors are concentrated. In an attempt to locate these concentrations, we analyzed the distribution of the defect–model parameters associated with errors.

It turns out that errors are strongly correlated with `size`: 93% of the errors in both tests involve 6 point images, and the rest involve 8 point images. The other model parameters, examined independently, do not correlate strongly with errors. We have not tested for correlations of errors with pairs, triples, etc of parameters, since the data is sparse.

This suggests future experiments in which we enrich the set of prototypes with samples of low `size`. Another way to use the errors, in the absence of marked correlations, would be to generate additional prototypes by slightly perturbing the parameter vectors associated with error cases.

4 Decision Trees

The second type of classifier we examine is decision trees. Their distinctive advantage is speed, and their weakness is rapid error accumulation with depth. We have conjectured that the use of unusually large training sets may circumvent this problem.

Many heuristics for decision–tree design have been proposed [4] [11] [12] [16] [17] [19]. We will focus here on accuracy (generalization of discrimination to the test set), rather than on space or time characteristics (although, as we been applied so far only to a simpler problem [3]. We hope that, in the future, this problem may be similarly characterized.

3 Nearest–Neighbor Matching

Nearest–neighbor matching is appealing for several reasons: the method is relatively simple to implement; and there exists a theorem that, under certain conditions on the class–conditional distributions, its asymptotic error rate is bounded above by twice the Bayes risk [5]. The proof depends on the fact that, as the number of prototypes increases, the nearest neighbor of a sample chosen from the class distributions is, in the limit, identical to the sample itself. In practice, given finite sets of prototypes, this limiting condition may not apply: Fukunaga [7] has shown that in this case there can be substantial bias in the estimate of the nearest–neighbor error.

The present state of the art does not offer a systematic procedure for choosing and quantizing features that is guaranteed to support the best possible discrimination. This uncertainty has encouraged engineers to use many and/or finely–quantized features. The resulting space of distinct representations is large, decreasing the probability that a sample will be identical to any prototype from a finite set. In spite of these difficulties, nearest–neighbor classification is widely considered one of the most reliable methods for achieving the highest possible accuracy on hard problems [15].

The most serious practical drawbacks of nearest–neighbor classification are the potentially exorbitant time and space requirements of naive implementations. Most prior work focuses on pruning the prototypes [8] [9] [14] [15] and speed–optimizing the search [6]. Still, their accuracy is always bounded above by the results of brute–force matching to the entire training set. Given an unbounded training set, we are now in a position to conduct large-scale trials to examine the asymptotic effects of the number of training samples on the classification accuracy.

We experimented first with perhaps the simplest image metric: Hamming distance between normalized 48×48 bilevel images (that is, simply the number of pixels that differ). Table 1 summarizes the results of the test. The training set referred to in each entry is a subset of that in the next entry. To distinguish between errors and ambiguities, we count a sample as a reject whenever its minimum distances to both classes are the same. Figure 2 shows the images of the errors found using the largest training set.

Table 1: Results of nearest–neighbor matching on an independent test set using Hamming distance.

training set	# prototypes (,000)	truth = c 50,000		truth = e 50,000		Total 100,000	
		# err	# rej	# err	# rej	# err	# rej
1	1	17	0	260	7	277	7
2	2	13	0	221	5	234	5
3	3	12	0	207	3	219	3
4	4	7	0	219	3	226	3
5	5	10	0	208	2	218	2
6	6	10	0	183	1	193	1
7	7	11	1	181	3	192	4
8	8	9	0	193	4	202	4
9	9	13	0	196	4	209	4
10	10	13	0	194	4	207	4
11	20	11	1	187	6	198	7
12	30	12	1	176	3	188	4
13	40	12	1	165	8	177	9
14	50	11	1	152	6	163	7
15	60	10	0	140	7	150	7
16	70	9	0	139	5	148	5
17	80	9	0	139	7	148	7
18	90	8	0	138	7	146	7
19	100	10	0	141	7	151	7
20	110	13	0	139	9	152	9
21	120	13	0	139	8	152	8
22	130	12	0	146	8	158	8
23	140	10	0	144	7	154	7
24	150	8	0	143	5	151	5
25	200	6	0	147	4	153	4
26	250	6	2	157	4	163	6
27	300	11	2	141	4	152	6
28	350	11	0	146	7	157	7
29	400	11	1	150	6	161	7
30	450	9	1	158	5	167	6
31	500	9	1	145	7	154	8

Accuracy generally improves, but slowly and not monotonically. The number of rejects is small, especially for the 'c's. The most rapid improvement occurred when the number of training prototypes was small. Beyond 90,000 prototypes (training set 18) no clear improvement was observed.

Errors are not symmetric: substantially more 'e's are misclassified as 'c's than 'c's misclassified as 'e's. One possible explanation is that 'c's are in "tighter clusters" so that their nearest neighbors are more likely to be 'c's; and that 'e's are more "scattered," so that some of them lie closer to 'c's than to other 'e's. To effects of classification methodology. Since all three classifiers will be trained using data from the same stochastic source, and in the same quantity, we can compare their ability to learn and generalize.

The three types of classifiers we consider are: nearest neighbors, decision trees, and perfect metrics [10]. These classifiers are similar in that, during training, their "capacity" can grow indefinitely: that is, their VC–dimension [18] can increase as they are exposed to more training samples. In other respects they are quite different.

2 Experimental Design

The recognition problem is to distinguish images of the symbols 'c' and 'e' in the Adobe [1] Times Roman typeface (Figure 1), under a model of image defects described below. These two letters were selected for the trial because:

1. commercial OCR machines often confuse them ([13] ranks 'e'→'c' and 'c'→'e' as the 2nd and 14th most common mistakes);
2. they are easily distinguishable when no noise is present (unlike '1' and 'l', which are nearly identical in Times Roman); and
3. they have identical height, width, and height above baseline (and thus cannot be easily distinguished by size and location).

Thus the problem is of practical interest, difficult but not hopeless, and not easily resolved by geometric context.

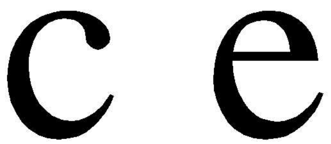

Figure 1: Ideal images of 'c' and 'e' in the Adobe Times Roman typeface.

The Times Roman typeface was selected because it is often used in American technical documents [1]. We assume that the two classes occur with equal probability, and that their images are isolated (that is, there are no spatial proximity effects); these assumptions are not realistic (in English prose, 'e' and 'c' occur approximately in the ratio 3.7:1), but they simplify the design of the experiment, so that it will be easier to replicate.

The model of image defects specifies a distribution on the parameters that control the distortion of an ideal character image, called the *prototype image.* The model is a single–stage parametric model of per–symbol and per–pixel defects ([2] gives details). A brief description of the parameters and their values in this experiment is given in the Appendix. Note that we generated the training and testing sets from slightly different distributions on the `size` parameter (nominal text size in units of points): the training data is distributed uniformly among sizes {5,7,9,11,13}, and testing data among {6,8,10,12,14}. This is a safeguard against the danger of generating long subsequences of images that are identical in both the training and testing sets. Although this problem is statistically unlikely to occur in any case, using different `size` distributions perturbs the pseudorandom number sequence and makes it even less likely.

Using this model, we generated a training set of 500,000 samples (250,000 'c's and 250,000 'e's), and a testing set of 100,000 samples (50,000 'c's and 50,000 'e's). Images were normalized to 48×48 images by first centering, and then linearly scaling (in X and Y separately) so that width and height just fit. The resulting 48×48 images are still bilevel. For the remainder of this paper, all images we refer to or show in Figures have been normalized.

The great variety of images generated may be appreciated by considering that within the testing set only 26 images of 'c' and 4 images of 'e' were repeated. Further, none of the 'c's were identical to any 'e's. We have not examined the training set exhaustively for ambiguities, since the computation would be excessive.

Ideally, error rates we report here should be compared to the Bayes risk of the problem, which is the lowest error achievable by any classifier. A computationally feasible method to estimate Bayes risk has been reported, but has

[1] Artwork defining the ideal images of these is available from Adobe Systems, Inc., 1585 Charleston Road, P.O. Box 7900, Mountain View, CA 94039.

Asymptotic Accuracy of Two–Class Discrimination

Tin Kam Ho and Henry S. Baird

AT&T Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974, USA

Abstract

Poor quality — e.g. sparse or unrepresentative — training data is widely suspected to be one cause of disappointing accuracy of isolated–character classification in modern OCR machines. We conjecture that, for many trainable classification techniques, it is in fact the dominant factor affecting accuracy. To test this, we have carried out a study of the asymptotic accuracy of three dissimilar classifiers on a difficult two–character recognition problem. We state this problem precisely in terms of high–quality prototype images and an explicit model of the distribution of image defects. So stated, the problem can be represented as a stochastic source of an indefinitely long sequence of simulated images labeled with ground truth. Using this sequence, we were able to train all three classifiers to high and statistically indistinguishable asymptotic accuracies (99.9%). This result suggests that the quality of training data was the dominant factor affecting accuracy. The speed of convergence during training, as well as time/space trade–offs during recognition, differed among the classifiers.

1 Introduction

Automatically trainable classifiers sometimes yield a disappointingly low accuracy on isolated–character recognition problems. It is often unclear whether this is due to flaws in the classification methodology (*e.g.* poorly chosen features), or inadequacies in the training sets (*e.g.* too few samples), or both. Given this uncertainty, and the expense of acquiring large and representative training sets, most OCR research in the last few decades has focused on novel methods for classification. If, however, we believed that the quality of training sets, rather than classification methodology, was the determining factor in achieving higher accuracy, then we might choose to devote more effort to improving the quality of training sets.

We have investigated this question through large–scale empirical studies of the asymptotic accuracy of three types of statistical classifiers, applied to the same problem. For this purpose we chose a two–class isolated–character recognition problem that often troubles modern OCR machines. We depart from previous studies by stating this problem *precisely*, in terms of high–quality prototype images and a parameterized model of image defects [2]. The model specifies a distribution on parameters governing a distortion algorithm that approximates the physics of printing and image acquisition. By pseudo–random sampling from the distribution, training and testing sets can be generated. Thus there are no limits, other than those imposed by our computing environment, on the size of these data sets. And, since the training and test sets are both selected at random from the same distribution, the training set is representative by construction. In this way we control what we feel are the two most important aspects of the "quality" of image data sets: their size and representativeness.

This experimental design is intended to isolate the effects of training set quality from the