A Self-Correcting 100-Font Classifier

HENRY S. BAIRD

GEORGE NAGY

AT&T Bell Laboratories
600 Mountain Avenue, Room 2C-322
Murray Hill, NJ 07974-0636

ECSE Department
Rensselaer Polytechnic Institute
Troy, NY 12180-3590

Abstract

We have developed a practical scheme to take advantage of local typeface homogeneity to improve the accuracy of a character classifier. Given a polyfont classifier which is capable of recognizing any of 100 typefaces moderately well, our method allows it to specialize itself automatically to the single — but otherwise unknown — typeface it is reading. Essentially, the classifier retrains itself after examining some of the images, guided at first by the preset classification boundaries of the given classifier, and later by the behavior of the retrained classifier. Experimental trials on 6.4M pseudo-randomly distorted images show that the method improves on 95 of the 100 typefaces. It reduces the error rate by a factor of 2.5, averaged over 100 typefaces, when applied to an alphabet of 80 ASCII characters printed at ten point and digitized at 300 pixels/inch. This self-correcting method complements, and does not hinder, other methods for improving OCR accuracy, such as linguistic contextual analysis.

Keywords: character recognition, polyfont, self-correcting, learning

1. Introduction

We propose a practical scheme to take advantage of local typeface homogeneity to improve accuracy in character recognition. We are motivated by the ease with which human readers adapt to new typefaces, even in the absence of linguistic or other contextual cues. We observe that, although among a large set of typefaces several distinct characters may have similar shapes, within the same typeface the shapes of distinct characters will differ significantly.

Single-font text may be easier to read
automatically than multiple-font text.

Figure 1: Examples of text printed in single and multiple typefaces. Most modern polyfont OCR algorithms will read the top line no better than the bottom line.

Figure 1 exhibits two lines of text, one printed in a single typeface, and the other printed in many typefaces. A typical modern polyfont classifier can be expected to perform no better on one than on the other: this

insensitivity to typeface can be considered harmful if it prevents the classifier from adapting to single-font text and so achieving higher accuracy.

Our method requires a high-performance polyfont† classifier, and prior knowledge that the text is in a single — but not necessarily known — typeface. The text may occur at one or more unknown type sizes. Given these preconditions, our method acts fully automatically and without further cues. Good but imperfect polyfont classifiers are now not uncommon, and single-typeface problems are frequently occurring special cases. Using the method does not preclude any resort to language context, nor does it materially restrict the use of better features, metrics, and speed optimizations — thus it can freely combine with other future improvements. It does not involve typeface recognition or require a set of typeface-specific classifiers.

In terms of statistical pattern recognition, our method falls in the realm of unsupervised or imperfectly supervised parameter estimation for mixture distributions. In spite of considerable interest in this topic in the sixties ([CC64], [PH66], [Spr66], [Sta68], [Yak70]), to the best of our knowledge unsupervised classification has not been applied to OCR since the series of experiments reported in [NS66] and [NT68]. Recently however, Mandler describes a computer-assisted method of classifier design where 90% of the work is done by an ancillary polyfont classifier [Man91]. Some current OCR devices display single examples of pattern classes that could not be identified confidently: after the user provides labels, all the unknown patterns of each class are relabeled. Similar schemes were demonstrated on prototype text readers in [AKM71] and [WCW82].

Early attempts at theoretical analysis of an iterative self-corrective method ([HA67], [NT68]) showed that under certain assumptions, the *mean* of the relevant class-conditional probability distributions could be accurately estimated. However, the extremely restrictive assumptions that were postulated were never realized in practice. We will therefore rely on large-scale empirical trials to demonstrate the robustness of the estimators even under adverse initial conditions. In addition to demonstrating the feasibility of obtaining essentially single-font recognition results in a polyfont setting, our experiments show the value of a flexible OCR research environment with automated facilities for classifier design, data management, and performance monitoring.

Section 2 gives the engineering context of our experiments. In Section 3 the self-correction method is described, and in Section 4 it is illustrated on a small-scale problem. The principal results of this paper are given in Section 5, which reports on a large-scale experimental trial. Section 6 discusses a more realistic variation of this experiment. Section 7 summarizes the results and discusses future work.

2. The Engineering Context

The general requirements for applying our self-correcting method are as follows. There should exist a polyfont classifier (the ''given classifier'') capable of distinguishing among N character (symbol) classes $\{c_i\}_{i=1,N}$ across a large number of typefaces with ''reasonably good'' accuracy. In addition, there should exist a classifier technology that is trainable on sample images labeled with classes.

In our environment, the given classifier is Bayesian. An unknown symbol image I is transformed into an M-dimensional *binary feature vector* $\mathbf{x} \in \{0,1\}^M$ which is normalized to be nominally invariant to type size and location (by the method of [Bai88b]). The classifier computes the *a posteriori* probabilities $\{P(c_i|\mathbf{x})\}_{i=1,N}$ that $\mathbf{x}$ belongs to each class c_i, and reports c_{best} as the top choice, where $best = argmax_i\{P(c_i|\mathbf{x})\}$. Each $P(c_i|\mathbf{x})$ is computed by reference to statistics stored in *class prototypes* T_i, each of which includes the conditional feature probabilities $\{P(x_j|c_i)\}_{i=j,\ldots,M}$ that feature j is set to 1 among samples from class c_i. The computation is carried out as described in [DH73], under the assumptions that the features are class-conditionally independent and

† We use the terms *font* and *typeface* interchangeably, as is usual among computer typesetters, to mean a typeface design that is distinguishable by shape. This differs from the conventions of letterpress typographers, for whom each different type size creates a distinct font. Also, we consider Times Roman and *Times Italic* to be distinct typefaces even though they belong conventionally to the same ''typeface family.'' Within a typeface family, moderate variations in weight (light/bold) and width (condensed/expanded), are represented in our trials by deformations due to the image defect model. Thus, our ''100 fonts'' represent approximately 50 complete body-text typeface families.

that the classes are equally likely. The conditional feature probabilities are actually estimates of the means of Bernoulli distributions.

In addition, sensitivity to type size and location (height above baseline), which is essential for case distinctions in the Latin alphabet, is achieved by the method of [Bai88a]: briefly, this is accomplished by reference to per-class first-order statistics (mean and variance) of size and location, also stored in the class prototypes. To allow this to be tested, each image used in the experiments is labeled with its true type size and baseline location.

The given classifier is thus completely specified by the set of prototypes $\mathbf{T} = \{T_i\}_{i=1,\ldots,N}$. The statistics stored in them have been estimated during an off-line training phase from sample images labeled with true classes. During self-correction, we will use this same training procedure to estimate new prototypes from relabeled training samples.

The accuracy of the classifier depends of course on many details, including the method for feature extraction, the size and quality of the training sample database, and the correctness of their class labels. In this study we will assume that feature-extraction (and thus M) is fixed, but that the class prototypes may be retrained on the fly given new or differently-labeled training samples. The runtime require for (re)training is asymptotically linear in the number of training samples.

The throughput of the classifier need not be linear in the number of classes, since there are sublinear-time algorithms for evaluating Bayes classifiers; but, for reasons of space, we will not elaborate on this here. Also, we will not discuss complications resulting from missegmented (merged or fragmented) symbols.

In the trials discussed below, the given classifier is similar to the one described in [Bai91], and has been trained on at least 100 typefaces commonly used to print bodytext in 20th C. American publications. The number of classes $N = 80$, comprising these printable-ASCII‡ symbols:

A-Z a-z 0-9 ., -:; *‘ ’ &$!?%/()[]

The number of binary-valued features $M = 512$: thus the vectors $\mathbf{x}$ are 512-bit strings. Test results reported in [Bai91] shows good performance at type sizes above 9 point at a spatial sampling rate of 300 pixels/inch (ppi).

This paper and [Bai91] both use large pseudo-randomly generated test sets, using a model of distortions [Bai92] caused by printing and imaging. Although it has not yet been established to what extent this model is complete and representative, our experience suggests that the resulting data sets are as challenging to present-day OCR algorithms as the majority of naturally occurring printed text. We generated 6.4M images, more than in any collection of real images of machine-print known to us. Certainly, these more *uniformly* represent all possible combinations of typefaces, symbols, and type sizes, than *ad hoc* collections. All the trials were conducted on images at a spatial sampling rate of 300 pixels/inch.

3. A Method for Self-Correction

We will now describe the self-correction method. At the outset, we are given a classifier specified by its set of class prototypes $\mathbf{T}^0$. Also, we are given a set of isolated symbol images $\mathbf{I} = \{I_k\}_{k=1,\ldots,K}$ whose class labels are unknown. The method proceeds in three stages:

Stage 1 Read the entire sample set $\mathbf{I}$, classifying each symbol image I_k using the given classifier $\mathbf{T}^0$. Let c_k^0 be the top-choice class reported for image I_k.

Stage 2 For each class c_i, estimate a new class prototype T_i^1 using the *retraining set* of images $\mathbf{I}_i^1 = \{I_k : c_k^0 = c_i\}$; that is, retrain assuming that the given classifier's top-choice classes are the true classes. Call the resulting set of prototypes $\mathbf{T}^1$.

Stage 3 Reclassify each image in $\mathbf{I}$ using $\mathbf{T}^1$.

The assumption used in Stage 2 does not, of course, always hold true, since the given classifier is not perfect. Thus the method generally retrains on imperfectly labeled samples: such ‘‘learning from poisoned data’’ is

‡ We exclude the ASCII symbols @ # + = < > ^ ~ { } _ | since artwork for them was not provided in all 100 typefaces.

generally agreed to be a risky practice.

If it happens that $\mathbf{I}_i^1$ is empty, then of course class c_i is not retrained and $T_i^1 = T_i^0$ as before. Retraining can be iterated: for each $n \geq 1$, use the top-choice class labels reported by classifier $\mathbf{T}^n$ to train the next classifier $\mathbf{T}^{n+1}$, in the obvious analogous manner.

4. An Illustrative Trial

Let us illustrate the method on a small problem. We choose six symbols { 0, O, Q, D, G, C } that are often confused, and a typeface, Avant Garde Book Oblique, on which the method behaves in a way typical of most. In this typeface, the symbols appear as follows:

0 O Q D G C

For each of these symbols, we generated 200 distorted images, at a nominal type size of 10 point; a few of these are shown (magnified) in Figure 2.

Figure 2: Fifteen pseudo-randomly distorted images for each of the six symbols used in the trial, illustrating the degradations introduced by the image defect model on 10 point text at 300 ppi.

The top-choice confusion matrix of the given polyfont classifier on this data is shown in Figure 3.

		top-choice						
		0	O	Q	D	G	C	
	0	96	104	0	0	0	0	52.0
t	O	0	200	0	0	0	0	0.0
r	Q	0	1	199	0	0	0	0.5
u	D	0	50	4	146	0	0	27.0
e	G	0	0	0	0	197	3	1.5
	C	0	0	0	0	2	198	1.0
		0.0	43.7	2.0	0.0	1.0	1.5	13.67

Figure 3: The top-choice confusion matrix of the given polyfont classifier on classes { 0, O, Q, D, G, C }, printed in Avant Garde Book Oblique at 10 point and 300 ppi, 200 pseudo-randomly distorted images each. Error rates are shown as percentages.

The confusion matrix is read as follows. Each (row,column) entry (i, j) gives the number of images of true class c_i which were classified as top-choice class c_j. Thus, of 200 images of '0' (numeric zero), 96 were correctly classified, 104 were incorrectly classified as 'O' (alphabetic oh), none as 'Q', and so forth. At the far right of each row, the overall error rate for that class is shown (in percent): thus, 52% of the images of '0' were misclassified. At the bottom of each column, the percent error rate for that top-choice class is shown: thus, 0.0% of images classified as '0' were in fact misclassified, but 43.7% of top-choice 'O's were misclassified. At the extreme bottom-right of the matrix, the overall error rate is shown: 13.67% of the 1200 images were misclassified.

After self-correction, the new classifier (now specialized to a single typeface) exhibits the confusion matrix shown in Figure 4.

		top-choice						
		0	O	Q	D	G	C	
	0	178	22	0	0	0	0	11.0
t	O	0	200	0	0	0	0	0.0
r	Q	0	0	200	0	0	0	0.0
u	D	0	26	0	174	0	0	13.0
e	G	0	0	0	0	200	0	0.0
	C	0	0	0	0	0	200	0.0
		0.0	19.4	0.0	0.0	0.0	0.0	4.00

Figure 4: The top-choice confusion matrix of the self-corrected single-font classifier on classes { 0, O, Q, D, G, C }, printed in Avant Garde Book Oblique at 10 point and 300 ppi, 200 pseudo-randomly distorted images each. Error rates are shown as percentages.

Note that the overall error rate has fallen to 4%, a reduction by a factor of ×3.5. Throughout this study, we report improvement in terms of error-reduction factors in order to compare results across typefaces. An error-reduction factor of ×1.0 means no improvement in accuracy; a factor of ×2.0 means that 50% of the errors are corrected; ×3.5 means that 71% of errors are corrected.

The effect on the top-choice 'O' images may seem remarkable. Even though almost 44% of the data were poisoned, retraining repelled over half of them.

If the method is iterated once again, the result is as shown in Figure 5.

		top-choice						
		0	O	Q	D	G	C	
	0	196	3	0	1	0	0	2.0
t	O	0	200	0	0	0	0	0.0
r	Q	0	0	200	0	0	0	0.0
u	D	2	16	0	182	0	0	9.0
e	G	0	0	0	0	200	0	0.0
	C	0	0	0	0	0	200	0.0
		1.0	8.7	0.0	0.5	0.0	0.0	1.83

Figure 5: The top-choice confusion matrix after two iterations of self-correction, on classes { 0, O, Q, D, G, C }, printed in Avant Garde Book Oblique at 10 point and 300 ppi, 200 pseudo-randomly distorted images each. Error rates are shown as percentages.

The error rate has fallen again, to 1.83%, a further reduction of ×2.2, for a total reduction of ×7.5 compared to the given classifier. Further iterations of the method on this data yield only small improvements: after five iterations, the error rate drops to 1.33%, for a total reduction of ×10.3.

An upper bound on the improvement possible through retraining can be estimated by the following computation: retrain on the images using their *true* class labels, and then test on the same data. In the circumstances of this trial, this yields an error rate of 0.67%, for an improvement by ×20. Although this is a biased and highly artificial statistic, it suggests that there may be room for further improvement in the method.

When the same experiment is run on each of the 100 typefaces used in [Bai91] (see Appendix A), we observe that 94 them enjoyed some improvement after one iteration, with an average† reduction in the error rate by ×3.4 (so Avant Garde Book Oblique is a typical example). Iterating five times yielded a better overall average

† When computing these averages, improvement factors greater than 25 were truncated to 25, to avoid exaggerations due to a few extraordinary factors (some are greater than 100!).

error reduction (×4.6), but at a cost: only 81 of the typefaces improved overall. The estimated upper bound on improvement is ×11, about twice as good as what was achieved. On some typefaces the error rate *increases* through all iterations, and on others the error rate decreases at first but increases in later iterations.

This six-symbol example has illustrated the principal features of the method. The next sections will report the results of larger and more realistic trials.

5. A 100-Font, Large-Alphabet Experiment

Next, we investigate the success rate of the method using a larger trial, on the 80 ASCII symbols listed in Section 2, and at a wider range of type sizes: 6, 10, 12, and 16 point. Six point type (at 300 ppi) is below the threshold of reliably good performance using modern commercial OCR machines. Ten and twelve point are representative of much book text and typewritten material. Sixteen point is larger than most body-text in books and magazines. As before, 200 images were generated for each combination of typeface, symbol, and type size: in total, 6.4M images.

The improvement, averaged over all 100 typefaces, is shown in Figure 6 for each type size, as a function of the number of iterations of the method.

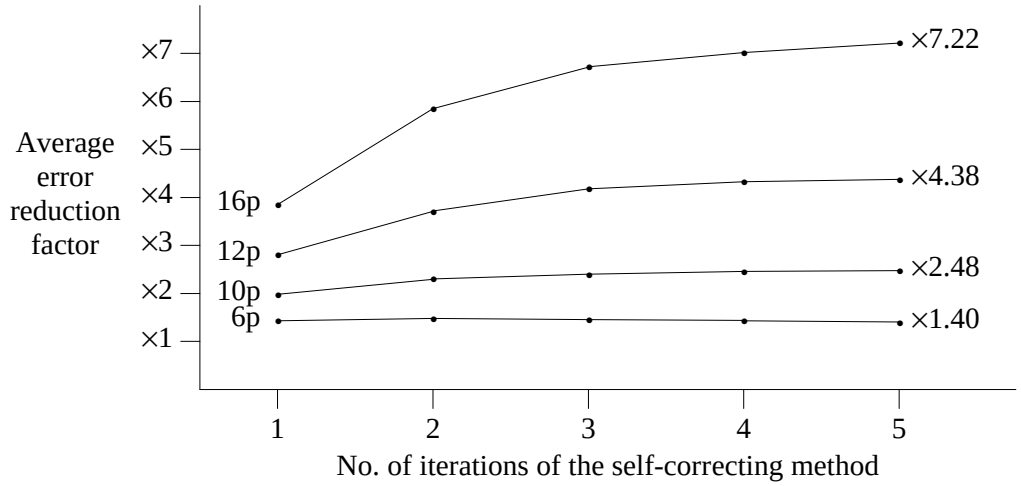

Figure 6: Reduction in error rate (multiplicative factor), averaged over 100 typefaces, for each type size 6, 10, 12, and 16 point, as a function of the number of iterations of the self-correcting method. The error-reduction factors after five iterations are printed on the right. A factor of ×1.0 means no improvement in accuracy. (On an alphabet of 80 ASCII characters, 200 pseudo-randomly distorted images for each symbol/typeface/size.)

Some overall improvement occurs at all four type sizes. The improvement at six point of ×1.4 means that nearly 30% of the errors have been corrected, which in many applications is significant. At ten point, ×2.5 means that 60% of the errors have been corrected. Greater improvements occurred at larger sizes. At all four sizes, the estimated upper bound on improvement was about two to three times greater than what was achieved.

Iteration clearly helps the average improvement, with most of the advantage occurring in the first three iterations. However, iteration often has a subtle cost: generally, the more iterations, the larger the number of typefaces that do not improve, or even worsen. For example, at ten point, after one iteration all 100 of the typefaces improve, but after 2 iterations only 99 improve; after 3 iterations, 97; and, after 5 iterations, 93. Therefore it may be judicious to limit the number of iterations, for example in applications where the cost of deteriorating on some typeface is higher than the reward for improving on average.

Across all four type sizes, the best improvement is remarkably high, while the worst is not very bad (after five iterations):

size	best	worst
16p	×141.5	×0.8
12p	×33.8	×0.9
10p	×10.9	×0.8
6p	×3.7	×1.0

Is it possible to predict which typefaces will improve, without actually running the algorithm? More specifically, does the outcome of the method correlate well with any easily measured statistical properties of typefaces? We investigated four statistics derivable from the top-choice confusion matrix resulting from running the given polyfont classifier on test images for each typeface, as follows.

1. *Overall error, averaged over all classes.*
2. *Maximum error among true classes.*
3. *Maximum error among top-choice classes.*
4. *Maximum ''worst/good ratio'' among top-choice classes.* The ''worst/good ratio'' for class c_i is computed by examining images whose top-choice label is c_i, and counting the images in the most frequently misclassified class (''worst''); then divide this number by the number of images that are correctly classified (''good''). A ratio greater than 1.0 means that, for some top-choice class, correctly labeled images are outnumbered by mislabeled images for some class. In cases like this, where mislabeled images dominate correctly labeled images, we might expect retraining to fail.

For all 100 typefaces, using the 10 point data, we plotted the improvement factor after one iteration as a function of each statistic. These plots revealed no clear correlations, and in particular no even approximately monotonic functional relationships. We conclude that these four statistics are not useful for predicting the improvement possible under the method. The feasibility of predicting a typeface's potential for improvement under this method remains an open question.

The data from this large trial, on 6.4M images of 80 symbols in 100 typefaces, comprise the principal results reported in this paper. We believe they are promising, but we recognize that the experimental design is unrealistic in several respects. For example, we cannot expect to have a minimum of 200 images for every class before retraining. In the next Section, we relax this constraint.

6. Sensitivity to the Number of Samples

We will now examine the effect of varying the number of image samples per class available for retraining. For this purpose, we reduced the scale of the experiment, using only the 10 point data and executing only one iteration. Also, we selected ten typefaces‡, each representing one of the 10%iles of improvement from the previous trial: the average improvement among these ten (×2.05) is close to the average among all 100 (×1.98). The input data was the same as in the earlier trial. The algorithms were modified to enforce an upper bound U on the number of images used when retraining a class: that is, if in Stage 2 $|\mathbf{I}_i^n| > U$, then we remove from $\mathbf{I}_i^n$ all but the first U members before retraining. Testing was carried out on all the input data, as before.

Figure 7 shows the average improvement after one iteration of retraining, as a function of U (''all'' means that all available data was used in retraining: this is the default behavior tested in the previous trial.) No average improvement is seen until $U \geq 3$, but fully half of the possible improvement is realized by $U = 10$. For $U \geq 25$, 100% of the typefaces improve. Little further improvement occurs for $U \geq 50$.

‡ Adobe Corona Roman, Adobe Corona-Italic, Adobe Excelsior Roman, Adobe Helvetica Italic, Adobe Times-BoldItalic, ITC Bookman Light Roman, Linotype Memphis Medium, Linotype Sabon Italic, and Linotype Trade Gothic.

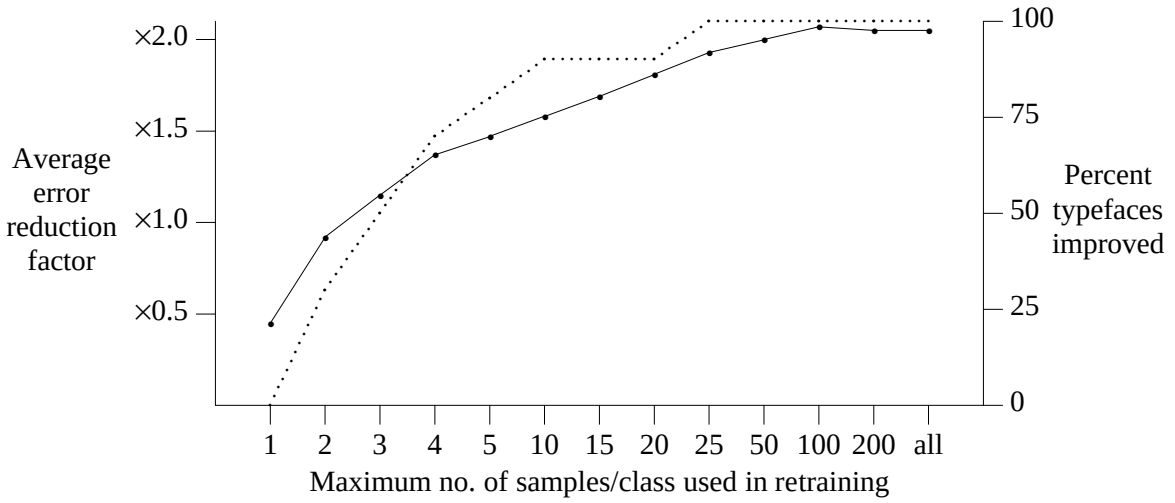

Figure 7: Reduction in error (left scale, solid plot), averaged over 10 typefaces, for 10 point type, as a function of the maximum number of sample images per class used in retraining. Also shown: the percent of typefaces that are improved (right scale, dotted plot). (On an alphabet of 80 ASCII characters, 200 pseudo-randomly distorted images for each symbol/typeface/size.)

Thus the self-correcting method appears to perform reliably when the number of samples per class available for retraining is at least 20. This minimum frequency requirement may not be unrealistically high: within the text of this article (excluding Figures and equations), among the 80 ASCII characters used in the trial, 62 of them (77%) occurred at least 20 times, and 22 (27%) at least 20 times per page, on average.

7. Discussion

We have described a way to improve classification by energetically exploiting a weak constraint on problems: in this case, the constraint is that the text is printed in a single typeface. Since the system must be told this fact, the method does not operate purely automatically. However, single-font text is a frequently occurring special case, often in large batches, so users need not fear having to intervene manually for every page. Also, the user need not possess the arcane skill of identifying typefaces by name.

Error reduction by factors of 2.5 or more can be significant in many applications, and so it is potentially important to have shown that this is possible on text images as small as 10 point (at 300 ppi). The unusually large scale of the experiments, involving 80 ASCII characters in 100 typefaces, should be reassuring.

This has been an exercise in combining technologies, in which the accuracy of an existing OCR system is amplified by a simple self-correcting scheme. It does not attempt the complex task of ‘‘typeface recognition,’’ nor does it require access to a library of pre-constructed typeface-specific classifiers. For a simple scheme of our sort to succeed, it seems to us that the given classifier must already perform reasonably well, above some minimum threshold. Although we do not know how to quantify this threshold, we have at least shown that one such classifier already exists.

The simplicity of the scheme has several advantages. It complements, and does not interfere with, other methods for improving OCR accuracy, such as linguistic contextual analysis. Also, it is easy, in an engineering sense, to add speed optimizations to the given classifier, such as preclassifiers, without changing the self-correction mechanism. If designed conservatively, the preclassifiers can run unchanged in combination with the retrained classifiers.

The large improvements from 12 point to 16 point type sizes may be an indication that the method may work not merely on good-quality images, but more generally on images whose degradations may be severe but are consistent. Also, it may save CPU time to restrict retraining to those symbols which are error-prone and likely to improve.

The upper bound on possible improvement that we have estimated is typically two or three times better than what has been achieved. This suggests that there may be room for further improvement. One idea is to prune the retraining sets in a better way, perhaps through clustering or ranking by confidence scores (based on $P(c_i|\mathbf{x})$) provided by the classifier. Another interesting idea is to enforce lower bounds on the size of the retraining sets.

Future work must of course include experiments on naturally occurring text images. In preparation for this, the sensitivity of the method to missegmented character images must be assessed.

8. Acknowledgement

We are grateful for helpful comments by Tin Kam Ho, David Ittner, and Jill Nagy.

9. References

[AKM71] R. N. Ascher, G. M. Koppelman, M. J. Miller, G. Nagy, and G. L. Shelton, ''An Interactive System for Reading Unformatted Printed Text,'' *IEEE Trans. Computers*, C-20, 12, pp. 1527-1543, December 1971.

[BF91] H. S. Baird and R. Fossey, ''A 100-Font Classifier,'' *Proc., 1st Int'l Conf. on Document Analysis and Recognition (ICDAR'91)*, St.-Malo, France, 30 September - 2 October, 1991.

[Bai88a] H. S. Baird, ''Feature Identification for Hybrid Structural/Statistical Pattern Classification,'' *Computer Vision, Graphics, and Image Processing*, Vol. 42, No. 3, pp. 318-333, June 1988.

[Bai88b] H. S. Baird, ''Global-to-Local Layout Analysis,'' *Proc., IAPR Workshop on Syntactic and Structural Pattern Recognition*, Pont-à-Mousson, France, 12-14 September, 1988.

[Bai92] H. S. Baird, ''Document Image Defect Models,'' in H. S. Baird, H. Bunke, and K. Yamamoto (Eds.), *Structured Document Image Analysis*, Springer-Verlag: New York, 1992.

[CC64] D. B. and P. W. Cooper, ''Non-supervised Adaptive Signal Detection and Pattern Recognition,'' *Information and Control*, 7, 3, pp. 416-444, September 1964.

[DH73] Duda, R. O., and Hart, P. E., *Pattern Classification and Scene Analysis*, Sects. 2.9-2.10, Wiley (New York, 1973).

[HA67] Y. Ho and A. K. Agrawala, ''On the Self-Correcting Learning Scheme of Nagy and Shelton,'' *Proc. IEEE (Letters)* Vol. 55, 10, pp. 1764-1765, October 1967.

[Man91] E. Mandler, ''AdlatiX—A Computer-Trainable OCR-System,'' *Proc., 1st Int'l Conf. on Document Analysis and Recognition*, Saint-Malo, France, pp. 341-349, September 30-October 2, 1991.

[NS66] G. Nagy and G. L. Shelton, ''Self-corrective Character Recognition System,'' *IEEE Trans. Information Theory*, IT-12, No. 2, pp. 215-222, April 1966.

[NT68] G. Nagy and N. Tuong, ''On a Theoretical Pattern Recognition Model of Ho and Agrawala,'' *Proc. IEEE*, 56, 6, pp. 1108-1109, June 1968.

[Nag68] G. Nagy, ''The Application of Nonsupervised Learning to Character Recognition,'' pp. 391-398, *Pattern Recognition* (L. N. Kanal, Ed.), Thompson Book Company (Washington, 1968).

[PH66] E. A. Patrick and J. C. Hancock, ''Nonsupervised sequential classification and Recognition of Patterns,'' *IEEE Trans. Information Theory*, IT-12, 3, pp. 362-372, July 1966.

[Spr66] J. Spragins, ''Learning Without a Teacher,'' *IEEE Trans. Information Theory*, IT-12, 2, pp. 223-230. April 1966.

[Sta68] D. F. Stanat, ''Unsupervised Learning of Mixtures of Probability Functions,'' pp. 357-390, *Pattern Recognition* (L. N. Kanal, Ed.), Thompson Book Company (Washington, 1968).

[WCW82] K. Y. Wong, R. G. Casey, F. M. Wahl, ''Document Analysis System,'' *IBM J. Res. & Dev.* 26, 6, 1982.

[Yak70] S. J. Yakowitz, ''Unsupervised Learning and the Identification of Finite Mixtures,'' *IEEE Trans. Information Theory*, IT-16, pp. 258-263, May 1970.

Appendix A. The 100 Fonts Used in the Trials

The rationale for selecting these fonts is given in [BF91]. They are all trademarks of Linotype AG, unless shown otherwise in square brackets.

Aster Roman

Aster Italic

Avant Garde Book Roman (ITC)
Avant Garde Book Oblique (ITC)
Bembo Roman
Bembo Italic
Bodoni Roman
Bodoni Italic
Bookman Light Roman [ITC]
Bookman Light Italic [ITC]
Breughel Roman
Breughel Italic
Caledonia Roman
Caledonia Italic
Caslon Old Face # 2 Roman
Caslon Old Face # 2 Italic
Cheltenham Roman
Cheltenham Italic
Clearface Regular Roman [ITC]
Clearface Regular Italic [ITC]
Cloister Roman
Cloister Italic
Corona Roman [Adobe]
Corona Italic [Adobe]
Courier 10 Roman [Bitstream]
Courier Twelve [Monotype]
Eurostile Roman
Eurostile Italic
Excelsior Roman [Adobe]
Excelsior Italic [Adobe]
Frutiger #55 Roman
Frutiger #56 Italic
Futura Book Roman
Futura Book Italic
Galliard Roman [ITC]
Galliard Italic [ITC]
Garamond # 3 Roman
Garamond # 3 Italic
Gill Sans Roman
Gill Sans Italic
Goudy Old Style Roman
Goudy Old Style Italic
Helvetica Roman
Helvetica Italic
Ionic Roman [Monotype]
Ionic Italic [Monotype]
Janson Text Roman [Adobe]
Janson Text Italic [Adobe]
Leamington Roman
Leamington Italic
Letter Gothic Roman [Adobe]
Letter Gothic Slanted [Adobe]
Lucida Roman [Adobe]
Lucida Italic [Adobe]
Melior Roman
Melior Italic
Memphis Medium Roman
Memphis Medium Italic
Meridien Roman
Meridien Italic
New Baskerville Roman [ITC]
New Baskerville Italic [ITC]
New Century Schoolbook Roman
New Century Schoolbook Italic
Optima Roman
Optima Italic
Palatino Roman
Palatino Italic
Plantin Light Roman
Plantin Light Italic
Prestige Elite Roman
Prestige Elite Italic
Print Out Roman
Rockwell Light Roman
Rockwell Light Italic
Sabon Roman
Sabon Italic
Serifa Roman
Serifa Italic
Souvenir Medium Roman [ITC]
Souvenir Medium Italic [ITC]
Spartan Book Roman
Spartan Book Italic
Textype Roman
Textype Italic
Times Roman
Times Italic
Trade Gothic Roman
Trump Mediaeval Roman
Trump Mediaeval Italic
Typewriter Elite [Monotype]
Typewriter Pica [Bitstream]
Univers #55 Roman
Univers #56 Italic
Walbaum Roman
Walbaum Italic
Weiss Roman
Weiss Italic
Zapf Book Light Roman [ITC]
Zapf Book Light Italic [ITC]