CROSS-DOMAIN SEARCHING USING HANDWRITTEN QUERIES[a]

D. LOPRESTI, G. WILFONG
Bell Labs, Lucent Technologies Inc.,
600 Mountain Avenue,
Murray Hill, NJ 07974, USA
E-mail: [dpl,gtw]@research.bell-labs.com

In this paper, we show how cross-domain approximate string matching can be applied to searching a database of scanned typeset documents using handwritten queries without requiring the correction of recognition errors. We present preliminary experimental results that suggest this approach can significantly improve retrieval effectiveness.

1 Introduction

As progress continues to be made in on-line and off-line handwriting recognition, an increasing number of applications will allow for the storage and retrieval of handwritten documents. It also seems likely that the current emphasis on forcing the user to correct errors as they arise will be supplanted by systems that capture writing unobtrusively, performing recognition in the background for the purposes of later retrieval. This same observation applies across other media as well: typeset text, voice recordings, images, etc.

A number of researchers have begun examining the problem of retrieval in the presence of recognition errors. For example, Taghva, *et al.*, observed that traditional vector-space techniques from the field of information retrieval are for the most part unaffected by errors due to optical character recognition (OCR) when the input is relatively clean.[1] Keyword-based searching, however, is much more susceptible to noise, especially when the input is degraded. One approach to this problem, based on combining techniques from approximate string matching and fuzzy logic, is described by Lopresti and Zhou.[2]

This earlier work generally assumes that the query is error-free. Consider the case, however, where both the query and the documents in the database may contain recognition errors. This scenario might arise, for example, when search terms written on the screen of a personal digital assistant (PDA) are used to query a database of faxes that have been OCR'ed. While it is known how to apply the error model for a single recognition process to improve retrieval, the question of combining models for different processes remains open.

[a]Presented at the *Seventh International Workshop on Frontiers in Handwriting Recognition*, Amsterdam, The Netherlands, September 2000.

Recently, we introduced a formalism known as *cross-domain approximate string matching* capable of modeling such scenarios.[3] In this paper, we apply this paradigm to the problem of searching a database of OCR'ed documents using handwritten queries, both of which contain uncorrected recognition errors. We present preliminary experimental results that demonstrate how this approach can increase the effectiveness of retrieval.

2 Cross-Domain Approximate String Matching

Approximate string matching is a widely-studied technique with important applications ranging from speech recognition to molecular biology.[4] A key principle is the concept of string edit distance, a measure for quantifying the similarity between two strings as well as for understanding the precise ways in which related strings may differ. In its most popular formulation, three basic operations are permitted: the deletion, insertion, and substitution of individual symbols. Each of these operations is assigned a cost, and the edit distance between two strings is then defined as the cost of the least expensive sequence of operations that transforms one string into the other.

As indicated earlier, we are interested in the problem where two strings to be compared both result from processes that potentially induce errors, but where the error models are not necessarily the same. For example, in querying via handwriting a database that was created from scanned documents, we must contend with on-line handwriting recognition errors as well as a completely different class of errors arising from OCR, as illustrated in Fig. 1.

This is the *cross-domain approximate string matching* problem. A formal model and an algorithm are presented elsewhere;[3] here we summarize the most relevant results. A domain is a means for producing strings of symbols over a given alphabet. The manner in which strings are produced suggests a means for judging the similarity between any two strings. In the domain of scanned pages that have been OCR'ed, symbols with similar shapes may be mistaken for one another, whereas a handwriting recognizer will likely make a different set of errors more dependent on confusing pen stroke sequences. The error model defines the costs for deleting, inserting, or substituting one symbol for another, *e.g.*, via a confusion matrix. This in turn induces a string-to-string distance measure within a single domain.

The cross-domain approximate string matching problem is defined by three domains $\mathcal{D}_i$, $1 \leq i \leq 3$, each with its own, possibly distinct, alphabet Σ_i, cost function $ecost_i$, and corresponding distance measure $edist_i$. In our case, $\mathcal{D}_1$ is the domain of OCR'ed documents, $\mathcal{D}_2$ is the domain of handwritten queries, and $\mathcal{D}_3$ is a third, unifying domain where strings originating in the first two can

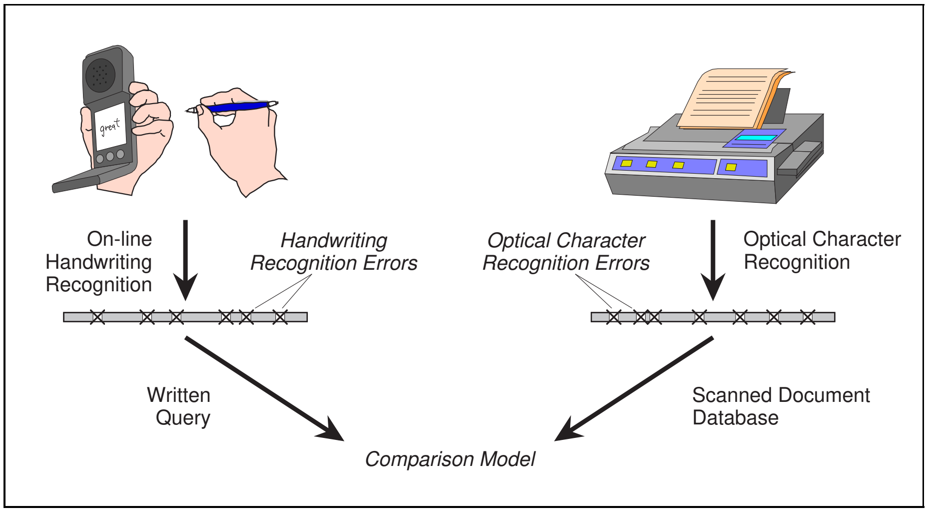

Figure 1: An example of cross-domain retrieval.

be mapped and compared. In the most general specification of the problem, the mappings between domains (called "transcriptions") also incur a cost and induce a distance measure, $tdist_i$. Here, however, we can assume that these mappings are the identity mapping, with cost zero. The formal statement of the optimization problem is then:

$$xdist_{1,2,3}(A,B) = \min_{\substack{A' \in \Sigma_1^*,\, B' \in \Sigma_2^* \\ A'',\, B'' \in \Sigma_3^*}} \left[\begin{array}{c} edist_1(A,A') \;+\; tdist_{1,3}(A',A'') \;+ \\ \quad edist_2(B,B') \;+\; tdist_{2,3}(B',B'') \;+ \\ edist_3(A'',B'') \end{array} \right] \tag{1}$$

Eq. (1) states that the cross-domain edit distance between two strings A and B is the optimal way to: (1) edit A into some other string A' in domain $\mathcal{D}_1$, (2) map A' to A'' in domain $\mathcal{D}_3$, (3) edit B into some other string B' in domain $\mathcal{D}_2$, (4) map B' to B'' in domain $\mathcal{D}_3$, and (5) compare A'' and B'' in domain $\mathcal{D}_3$. The computation is perhaps more easily visualized by considering Fig. 2.

3 Algorithm

The optimization problem of Eq. (1) can be solved using dynamic programming. Let $A = a_1 a_2 \ldots a_m$ be a string from Σ_1^* and $B = b_1 b_2 \ldots b_n$ be a string from Σ_2^*.[b] We wish to determine $xdist_{1,2,3}(A,B)$.

Consider first the case that A and B each correspond to at most a single symbol in their respective target strings. Then the distance between them should include the cost of editing A into zero or one symbols A', the cost of editing B into zero or one symbols B', the costs of transcribing A' into A'' and B' into B'' (assumed to be zero in this application, with $A'' = A'$ and $B'' = B'$), and finally the edit cost between A'' and B''. Thus we define:

$$xcost_{1,2,3}(A,B) = \min_{\substack{A' \in \Sigma_1 \cup \{\varepsilon\},\ B' \in \Sigma_2 \cup \{\varepsilon\} \\ A''(=A'),\ B''(=B') \in \Sigma_3 \cup \{\varepsilon\}}} \left[\begin{array}{c} edist_1(A,A') + \\ edist_2(B,B') + \\ ecost_3(A'',B'') \end{array} \right] \quad (2)$$

Note the similarity between the formulations of Eqs. (2) and (1).

For any string $S = s_1 s_2 \ldots s_t$, let $S_{i,j}$ denote the substring $s_i s_{i+1} \ldots s_j$. We can determine $xdist_{1,2,3}(A,B)$, the cross-domain edit distance between A and B, by computing $xdist_{1,2,3}(A_{1,i}, B_{1,j})$ recursively:

$$xdist_{1,2,3}(A_{1,i}, B_{1,j}) = \min \left\{ \begin{array}{l} \displaystyle\min_{1 \le k \le i} \left[\begin{array}{c} xdist_{1,2,3}(A_{1,i-k}, B_{1,j}) + \\ xcost_{1,2,3}(A_{i-k+1,i}, \varepsilon) \end{array} \right], \\ \displaystyle\min_{1 \le k' \le j} \left[\begin{array}{c} xdist_{1,2,3}(A_{1,i}, B_{1,j-k'}) + \\ xcost_{1,2,3}(\varepsilon, B_{j-k'+1,j})] \end{array} \right], \\ \displaystyle\min_{1 \le k \le i,\ 1 \le k' \le j} \left[\begin{array}{c} xdist_{1,2,3}(A_{1,i-k}, B_{1,j-k'}) + \\ xcost_{1,2,3}(A_{i-k+1,i}, B_{j-k'+1,j}) \end{array} \right] \end{array} \right. \quad (3)$$

where we define the base case to be $xdist_{1,2,3}(\varepsilon,\varepsilon) = xcost_{1,2,3}(\varepsilon,\varepsilon)$. The three minimization terms reflect the costs of deleting, inserting, and substituting substrings in A, respectively. That is, we test the hypothesis that the left-hand-sides of the last operation are of length k in $A_{1,i}$ and k' in $B_{1,j}$. The cost of this hypothesis is the sum of the costs of performing the operation and the earlier-computed cross-domain edit distance between $A_{1,i-k}$ and $B_{1,j-k'}$. Then $xdist_{1,2,3}(A_{1,i}, B_{1,j})$ is the minimum over all such hypotheses.

Following the analysis for the general algorithm,[3] it can be shown that with $O(m^2 n^2 |\Sigma|^2)$ preprocessing time, we can compute $xdist_{1,2,3}(A,B)$ in $O(m^2 n^2)$ additional time. One possible heuristic for speeding up the computation is to limit the range of k and k' in Eq. (3) by a constant. The resulting complexity becomes $O(mn|\Sigma|^2)$ for preprocessing, with $O(mn)$ time needed to then compute $xdist_{1,2,3}(A,B)$.

[b] We write Σ_i^* to mean a string of length zero or more symbols from Σ_i, and use ε to denote the null string.

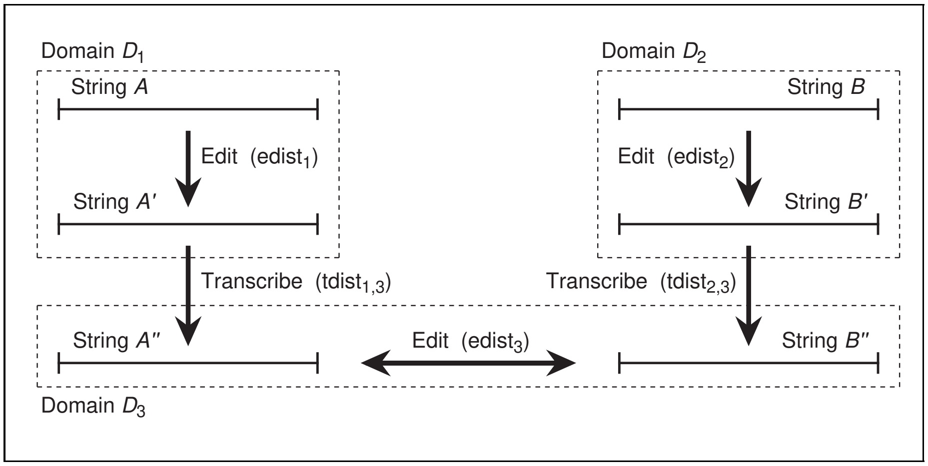

Figure 2: The cross-domain approximate string matching problem.

4 Experimental Results

To examine the effectiveness of cross-domain approximate string matching, we designed a simple experiment involving a small database of OCR'ed documents queried by handwritten search terms. Since this work is still in its early stages, these results must be considered preliminary. A more comprehensive evaluation is necessary, but is complicated by the substantial effort needed to collect sufficiently large, independent corpora of OCR'ed documents and handwritten queries for building confusion matrices and for running tests, and by the computational complexity of the algorithm.

4.1 Handwriting Recognition System

The handwritten character recognizer used in these experiments is an on-line, writer-dependent system that requires the user to provide at least one labeled sample of each character to construct a dictionary.[5] Given an unknown input character, the recognizer finds the entry in the dictionary that best matches the input based on a least-squares-type measure that is independent of scale, rotation, and translation. The recognizer does, however, rely on consistency in the ordering of pen strokes, and in the writing direction within each stroke.

Each character in the dictionary is represented by a sequence of (x, y) pairs corresponding to the motion of the pen tip as the user writes. The representation also includes information about which data points represent the last point before a pen lift. The sequence of (x, y) pairs is modified from the original data to normalize for translational differences. Since a least-squares-type distance measure is to be used, it would seem natural to subtract from each data point the centroid of the set, since this gives the best match in the least-squares sense. However, this is not robust because variations in writing speed could bias the centroid. One obvious solution would be to re-sample uniformly based on arc length along each stroke of the symbol. Instead, the centroid of the piecewise-linear curve obtained by interpolating linearly between consecutive data points is used, as this will be invariant under any re-sampling along such a piecewise-linear curve.

Suppose (a_i, b_i) for $1 \leq i \leq n$ is the representation of some character α in the dictionary (as modified for translational differences as indicated above), and (x_i, y_i) for $1 \leq i \leq n$ is the representation (normalized for translation) of some unknown character β to be recognized. Let $A = (a_1, b_1, \ldots, a_n, b_n)$ and $X = (x_1, y_1, \ldots, x_n, y_n)$. Define $A^R = (-b_1, a_1, -b_2, a_2, \ldots, -b_n, a_n)$, where A^R can be thought of as the symbol α rotated counterclockwise by $\pi/2$ radians. A measure of similarity $S(\alpha, \beta)$ between α and β that is invariant with respect to rotation and scaling is defined by $[(X \cdot A)^2 + (X \cdot A^R)^2]/[(X \cdot X)(A \cdot A)]$. It can be shown that this measure computes the best correlation between X and A for any rotation or scaling of X.

Thus, recognition involves computing $S(\alpha, \beta)$ for each α in the dictionary, and the entry with the highest score is chosen as the most likely identity of β.

4.2 Experiments

The database consisted of 100 professionally written news articles collected from Usenet. Each article was truncated to a maximum of 40 text lines, and the first 10 were typeset in 10-point Courier and printed one-per-page. These pages were then photocopied on a Xerox 5352C copier with the exposure set to the lightest possible setting. The copied pages were then scanned on a UMAX Astra 1200S scanner at a resolution of 300 dpi and OCR'ed using a system developed by Baird and described in detail elsewhere.[6]

An error classification procedure based on string matching was then run to build a confusion matrix for the OCR'ed typeset text.[7] Some of the more common errors we encountered are listed in Table 1. As indicated in Table 2, the OCR accuracies for these pages ranged from a low of 74.9% to a high of 91.9%. This data was used to inject synthetic noise into the remaining 90 pages in the test database.[c]

[c]This step was used as an expedient until we are able to print, copy, scan, and OCR all of the pages in question.

Table 1: Some representative OCR errors (10-point Courier, light photocopies).

Pattern	Frequency	Pattern	Frequency
6 → &	0.43	p → y	0.06
2 → ?	0.25	u → v	0.06
E → 8	0.17	r → z	0.04
, →	0.12	f → t	0.03
t → c	0.06	h → b	0.02

Table 2: Retrieval performance for clean queries (average precision at 100% recall).

		Unique	Average Precision		
	OCR	Non-Stop		Edit Distance	
Document	Accuracy	Terms	Exact	Simp	CM
15233	91.9%	86	0.43	0.75	0.81
15234	89.8%	158	0.37	0.63	0.70
15242	88.8%	153	0.36	0.59	0.66
15243	84.2%	191	0.33	0.62	0.66
15256	78.3%	75	0.30	0.58	0.62
15261	82.5%	52	0.21	0.61	0.66
15264	82.0%	67	0.25	0.55	0.65
15265	88.9%	127	0.41	0.65	0.71
15266	86.8%	92	0.40	0.62	0.67
15267	74.9%	119	0.22	0.48	0.57
Average	84.8%	112	0.34	0.61	0.67

A completely different set of 10 pages was similarly typeset in 10-point Courier, printed, scanned, and OCR'ed. The error analysis in this case was used to build a confusion matrix for setting the edit costs for string matching.

As an initial test, we ran each unique non-stopword from each of the first 10 database documents (the ones that had been actually OCR'ed) as a query and measured the average precision at 100% recall for: (1) exact matching (Exact), (2) simple edit distance with constant costs (Simp), and (3) edit distance with costs based on the confusion matrix (CM). These results are reported in Table 2. Clearly, edit distance provides a significant advantage over exact matching when there is noise in the database. Specializing edit distance to take into account expected error behavior via a confusion matrix provides an additional degree of improvement.

For our cross-domain retrieval experiment, each author created a personal dictionary for the handwritten character classifier described in the previous

Table 3: Some representative handwriting recognition errors.

Writer 1		Writer 2	
Pattern	Frequency	Pattern	Frequency
`Y` $\rightarrow$ `y`	0.80	`w` $\rightarrow$ `W`	0.90
`S` $\rightarrow$ `s`	0.70	`R` $\rightarrow$ `B`	0.80
`1` $\rightarrow$ `'`	0.60	`z` $\rightarrow$ `2`	0.30
`L` $\rightarrow$ `(`	0.30	`n` $\rightarrow$ `u`	0.20
`+` $\rightarrow$ `X`	0.20	`a` $\rightarrow$ `9`	0.10

section. Pre-segmented handwriting was captured off of a Wacom ArtZ II digitizing tablet attached to an SGI O2 workstation using *inkedit*, a graphical tool we have developed to support such research. We then wrote the alphabet 10 times to gather data to build a confusion matrix. Because the current system works on isolated characters, there were no deletion or insertion errors, but the algorithm is powerful enough to handle these and we plan to examine such cases in the future.[d] Table 3 lists some common errors for the two writers.

It is important to note the fundamental differences between the two domains (*i.e.*, Tables 1 and 3). Errors such as `E` $\rightarrow$ `8` that arise in OCR because two characters look similar will never occur in handwriting recognition because the stroke counts are so different (three or four for `E` vs. one or two for `8`). Likewise, common errors in the latter such as `a` $\rightarrow$ `9` are a function of the way `a`'s and `9`'s are written, the only distinction being the length of the "tail."

Next we analyzed the OCR results to identify a set of 10 keyword queries, one based on each of the first 10 documents in the database (but possibly with hits to other documents as well). In particular, we chose search terms that corresponded to non-stopwords that had suffered real OCR errors. These were written by the two authors and recognized using the handwritten character classifier. We then ran our cross-domain approximate string matching algorithm (XD), along with exact matching (Exact) and simple edit distance with constant costs (Simp) for comparison purposes. The results of this experiment are presented in Table 4.

As can be seen, cross-domain approximate string matching has the potential to greatly increase precision, despite considerable damage to the queries. An example of an optimal alignment (or "trace") found using this technique is shown in Fig. 3. This power comes at a cost, however. The time needed to perform a single keyword-document comparison averages 20 seconds on an SGI O2 workstation with a 200 MHz MIPS R5000 CPU and 64 MB main memory.

[d]The OCR'ed pages did, of course, exhibit numerous deletion and insertion errors.

Table 4: Cross-domain approximate string matching results (precision at 100% recall).

	Writer 1				Writer 2			
		Precision				Precision		
Query	Recognized	Exact	Simp	XD	Recognized	Exact	Simp	XD
practices	praCtiCes	0.07	0.11	0.25	Pr9Cti[es	0.07	0.08	0.25
great	gneat	0.09	0.09	0.17	gre9t	0.09	0.11	0.26
1994	1a94	0.25	0.37	0.96	I9a4	0.25	0.57	1.00
result	reS9lt	0.16	0.16	0.34	re5uIt	0.16	0.15	0.20
accused	acCused	0.04	0.05	0.14	a(cuSed	0.04	0.10	0.18
shares	shareS	0.22	0.39	0.84	9hares	0.22	0.22	0.84
October	OCtObOr	0.09	0.10	0.20	0Ct0ber	0.09	0.09	0.16
expense	eXpenSe	0.08	0.08	0.18	eXPenSe	0.08	0.09	0.13
rates	rateS	0.10	0.12	0.35	rateS	0.10	0.12	0.35
fewer	feW0r	0.06	0.10	0.14	feoer	0.06	0.06	0.13

5 Conclusions and Future Research

In this paper, we have described how cross-domain approximate string matching can be applied to searching databases of OCR'ed documents using handwritten queries without requiring the correction of recognition errors. The paradigm is a general one, assuming only that error models for the recognition processes are known and not, for example, that the text is limited to a specific lexicon or language model. While we presented an experimental evaluation for one particular scenario, it should be clear that many other interesting combinations are possible (*e.g.*, one writer querying documents written by another).

Still, the results reported here are only preliminary; more work is necessary. This includes examining much larger datasets, both for the database and for the queries. Since the accuracy of the error models is a key issue, it would be interesting to examine the effects of different fonts and document degradations (*e.g.*, faxing). It would also be useful to incorporate character segmentation into the handwriting recognizer so that it is possible to study the impact of segmentation errors on the handwritten queries. Finally, techniques must be found to speed up the computation.

References

1. K. Taghva, J. Borsack, A. Condit, and P. Inaparthy. Effects of OCR errors on short documents. In *Annual Report of UNLV Information Science Research Institute*, pages 99–105, Las Vegas, NV, 1995.

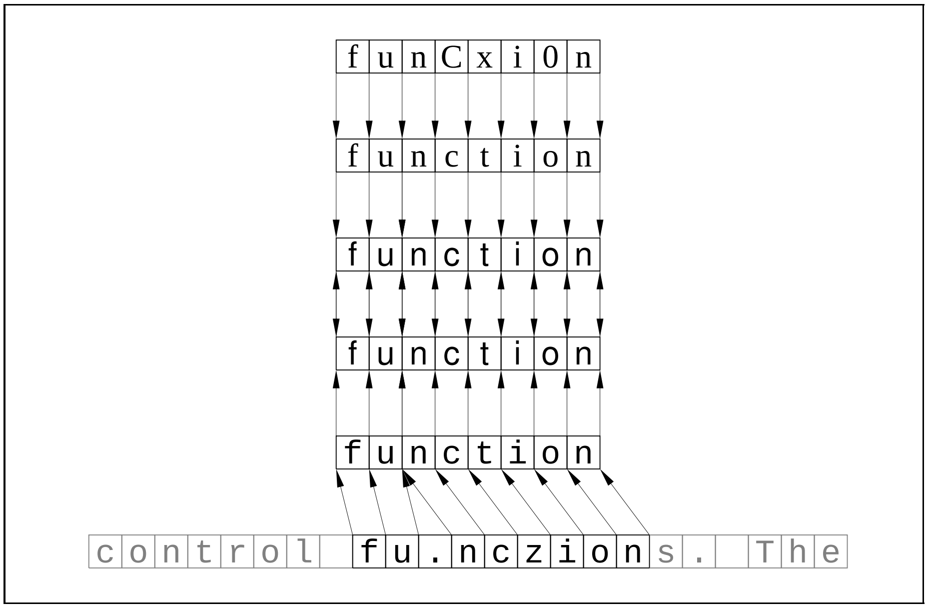

Figure 3: Example of a trace found using cross-domain approximate string matching.

2. D. Lopresti and J. Zhou. Retrieval strategies for noisy text. In *Proceedings of the Fifth Annual Symposium on Document Analysis and Information Retrieval*, pages 255–269, Las Vegas, NV, Apr. 1996.
3. D. Lopresti and G. Wilfong. Cross-domain approximate string matching. In *Proceedings of the Sixth International Symposium on String Processing and Information Retrieval*, pages 120–127, Cancún, Mexico, Sept. 1999.
4. D. Sankoff and J.B. Kruskal, editors. *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison.* Addison-Wesley, Reading, MA, 1983.
5. G. Wilfong, F. Sinden, and L. Ruedisueli. On-line recognition of handwritten symbols. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):935–940, 1996.
6. H.S. Baird. Anatomy of a versatile page reader. *Proceedings of the IEEE*, 80(7):1059–1065, 1992.
7. J. Esakov, D.P. Lopresti, J.S. Sandberg, and J. Zhou. Issues in automatic OCR error classification. In *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval*, pages 401–412, Las Vegas, NV, Apr. 1994.