Indexing Handwriting Using Word Matching

R. Manmatha, Chengfeng Han, E. M. Riseman and W. B. Croft, *
Center for Intelligent Information Retrieval,
Computer Science Department,
University of Massachusetts, Amherst, MA-01003.
manmatha@cs.umass.edu

Abstract

There are many historical manuscripts written in a single hand which it would be useful to index. Examples include the W. B. DuBois collection at the University of Massachusetts and the early Presidential libraries at the Library of Congress. The standard technique for indexing documents is to scan them in, convert them to machine readable form (ASCII) using Optical Character Recognition (OCR) and then index them using a text retrieval engine. However, OCR does not work well on handwriting. Here an alternative scheme is proposed for indexing such texts. Each page of the document is segmented into words. The images of the words are then matched against each other to create equivalence classes (each equivalence classes contains multiple instances of the same word). The user then provides ASCII equivalents for say the top 2000 equivalence classes.

The current paper deals with the matching aspects of this process. Due to variations in even a single person's handwriting, it is expected that the matching will be the most difficult step in the whole process. A matching technique based on Euclidean distance mapping is discussed. Experiments are shown demonstrating the feasibility of the approach.

1 Introduction

Text has always been the primary source of information in conventional, and more recently digital libraries. If the text is in machine readable form (ASCII), it can be indexed using standard text retrieval engines. However, much of the text in both historical and even

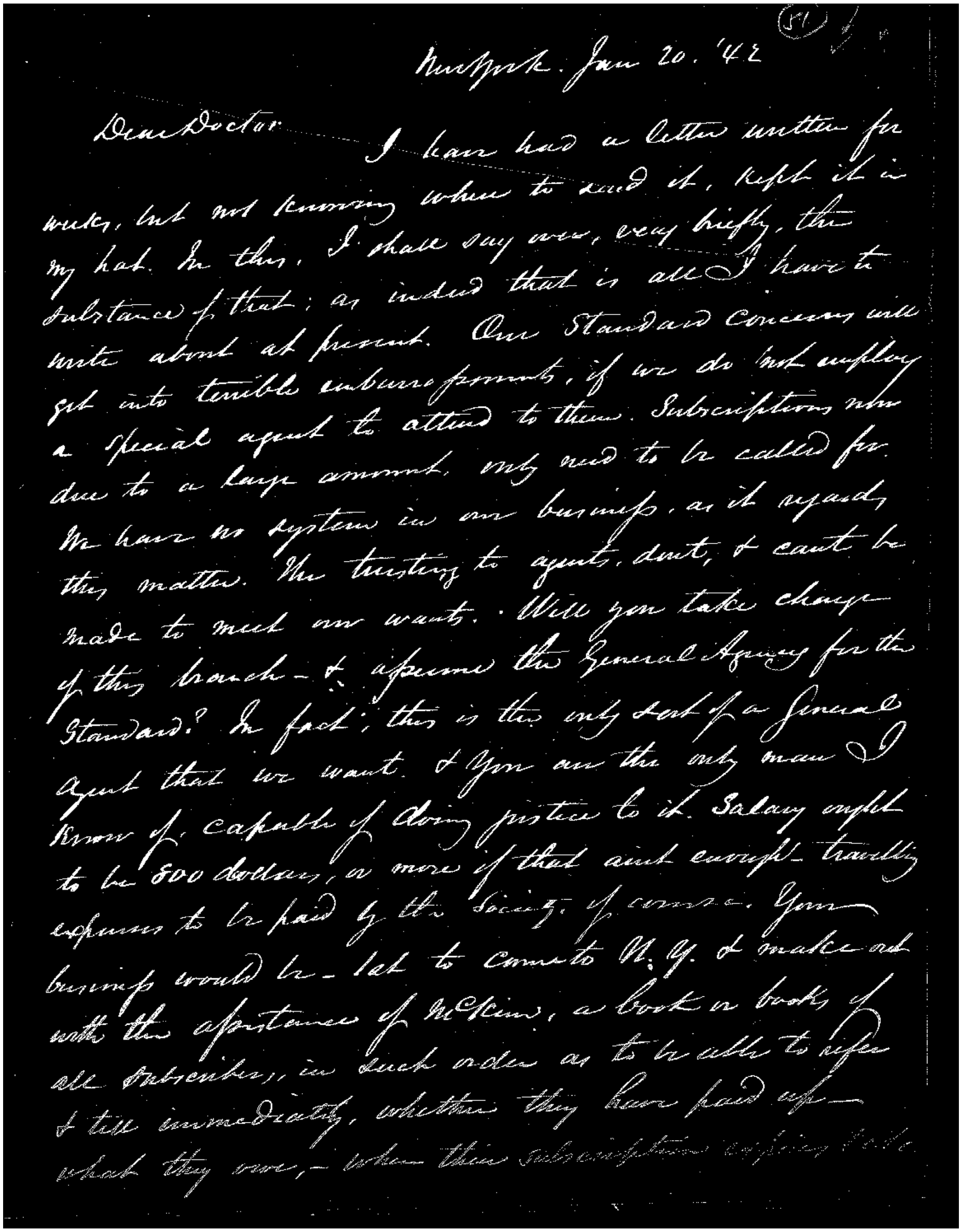
*This research was supported by the NSF Center for Intelligent Information Retrieval and by ARPA grant number N66001-94-D-6054.

current collections is contained in paper documents. One solution is to use Optical Character Recognition (OCR) to convert scanned paper documents into ASCII. Existing OCR technology works well with standard machine printed fonts against clean backgrounds. It works poorly if the originals are of poor quality or if the text is handwritten. We propose an alternative solution for indexing handwritten text when a large corpus of texts written by a single person exists.

In the context of digital libraries, the problem being addressed in this paper is primarily related to the indexing of historical manuscripts. These manuscripts are largely written in a single hand and most of them are unpublished. For example, the collected works of well-known people like W. E. B. Du Bois, the African American civil rights leader, and Margaret Sanger, a pioneer in birth control are mostly unpublished and are stored at archives at the University of Massachusetts and Smith College respectively. Both left a substantial amount of their work and correspondence written in their own hand. and it is unlikely that all of this material will ever be published. Such manuscripts are, however, valuable resources for scholars as well as others who wish to consult the original manuscripts. It would, therefore, be useful to index them to allow rapid access to relevant texts. Since conventional OCR and text retrieval engines cannot be used, this paper proposes an alternative strategy for indexing such documents.

The indexing scheme proposed here also simplifies reading documents where the handwriting is hard to read. A scanned page from the correspondence and journals of Erasmus Darwin Hudson (1809-1880) - an anti-slavery organizer and pioneer orthopaedic surgeon - is shown in Figure 1. The authors are still unable to decipher some of the words on this page.

Since the document is written by a single person, the assumption is that the variation in the word images will be small. The proposed solution will match the actual word images against each other to create equivalence classes. Each equivalence class will consist of multiple



New York. Jan 20. '42

Dear Doctor — I have had a letter written for weeks, but not knowing where to send it, kept it in my hat. In this, I shall say over, very briefly, the substance of that; as indeed that is all I have to write about at present. Our Standard Concerns will get into terrible embarrassments, if we do not employ a special agent to attend to them. Subscriptions now due to a large amount, only need to be called for. We have no system in our business, as it regards this matter. The trusting to agents, don't, & can't be made to meet our wants. Will you take charge of this branch — & assume the General Agency for the Standard? In fact, this is the only sort of a General agent that we want. & You are the only man I know of, capable of doing justice to it. Salary ought to be 800 dollars, or more if that aint enough — travelling expenses to be paid by the Society, of course. Your business would be — 1st to come to N. Y. & make out with the assistance of McKim, a book or books of all Subscribers, in such order as to be able to refer & tell immediately, whether they have paid up — what they owe, — when their subscription expires &c.

Figure 1: Manuscript from the Collected Papers of the Hudson Family

instances of the same word. Each word will have a link to the page it came from. The number of words in each equivalence class will be tabulated. Those classes with the largest numbers of words will probably be stopwords, i.e. conjunctions such as "and" or articles such as "the". Classes containing stopwords are eliminated (since they are not very useful for indexing). A list is made of the remaining classes. This list is ordered occuring to the number of words contained in them. The user provides ASCII equivalents for a representative word in each of the top m (say m = 2000) classes. The words in these classes can now be indexed. This technique will be called "wordspotting" as it is analogous to "wordspotting" in speech processing [5].

The proposed solution completely avoids machine recognition of handwritten words as this is a difficult task [8]. Robustness is achieved compared to OCR systems for two reasons:

1. Matching is based on entire words. This is in contrast to conventional OCR systems which essentially recognize characters rather than words.

2. Recognition is avoided. Instead a human is placed in the loop when ASCII equivalents of the words must be provided.

The present paper deals with the first part of the problem where the scanned document is segmented into word images and the word images are matched against each other. A future paper will deal with the rest of the system. The matching phase of the problem is expected to be the most difficult part of the problem. This is because unlike machine fonts, there is some variation in even a single person's handwriting. This variation is difficult to model. Figure (2) shows two examples of the word "Lloyd" written by the same person. The last image is produced by XOR'ing these two images. The white areas in the XOR image indicate where the two versions of "Lloyd" differ. This result is not unusual. In fact, the differences are sometimes even larger.

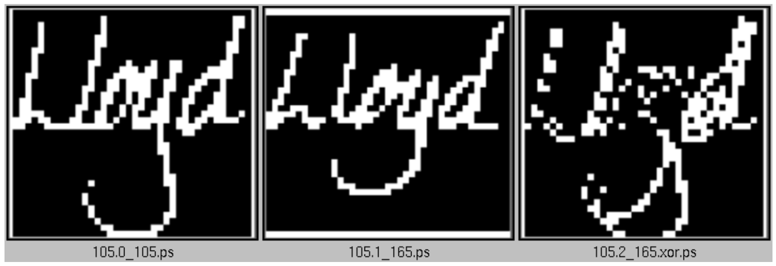


Figure 2: Two Examples of the Word "Lloyd" and the XOR image

In this paper, a matching technique based on Euclidean distance mapping [2] is discussed and preliminary results are given.

2 Prior Work

The traditional approach to indexing documents involves first converting them to ASCII and then using a text based retrieval engine [10, 9]. Scanned documents can be converted into ASCII by first segmenting a page into words and then running them through an OCR [1]. The OCR segments the words further into characters and then attempts to recognize the characters using statistical pattern classification [1]. This approach has been highly successful with standard machine fonts against clean backgrounds. It has had much more limited success when handwriting is used. Primarily, this is because character segmentation is much more difficult in the presence of handwriting and also because of the wide variability in handwriting (not only is there variability between writers, but a given person's writing also varies).

An approach similar to ours has been used to recognize words in documents which use machine fonts [6]. The word images are compared against each other and divided into equivalence classes. The words within an equivalence class - all of which are presumably identical - are used to construct a noise-free version of the word. This word is then recognized using an OCR. Recognition rates are much higher than when the OCR is used directly [6].

Machine fonts have a number of advantages over handwriting. Multiple instances of a given word printed in the same font are identical except for noise. This situation does not hold for handwriting. Multiple instances of the same word on the same page by the same writer show variations. The variations are many - these include scaling of the words with respect to each other, small changes in orientation, and changes in the lengths of descenders and ascenders. The first two pictures in Figure 2 are two identical words from the same document, written by the same writer. It may thus be necessary to account for these variations.

3 Outline of Algorithm

1. A scanned greylevel image of the document is obtained.

2. The image is first reduced by half by gaussian filtering and subsampling.

3. The reduced image is then binarized by thresholding the image (note the thresholding is done in such a way that the characters are white and the background black).

4. . The binary image is now segmented into words. this is done by a process of smoothing and thresholding described later.

5. A given word image (i.e. the image of a word) is used as a template and matched against all the other word images. This is repeated for every word in the document. The matching is done in two phases. First, the number of words to be matched is pruned using the areas and aspect ratios of the word images. Next, the actual matching is done by comparing the minimum distance of the Xor'ed images. The matching divides the word images into equivalence classes - each class presumably containing other instances of the same word.

6. Indexing is done as follows. For each equivalence class, the number of elements in it is counted. The top n equivalence classes are then determined from this list. The equivalence classes with the highest number of words (elements) are likely to be stop-words (i.e. conjunctions like 'and' , articles like 'the', and prepositions like 'of') and are therefore eliminated from further consideration. Let us assume that of the top n, m are left after the stopwords have been eliminated. The user then displays one member of each of these m equivalence classes and assigns their ASCII interpretation. These m words can now be indexed anywhere they appear in the document.

We now discuss these techniques in detail.

3.1 Word Segmentation

The technique to segment words is simple. It assumes that a binary image of each page is available and further assumes that the words are white against a dark background (if it is otherwise in the original image, the image can be inverted). Since the spacing between adjacent characters in a word is smaller than the spacing between adjacent words, a new image is constructed using a smoothing and thresholding operation. If two white pixels are separated by less than a certain distance k, the intermediate pixels are made white. This is done in the horizontal direction k_{horiz}. In the case of handwriting, this procedure also needs to be performed in the diagonal direction k_{diag} - mainly to prevent descenders from breaking up. Note that each of these window operations may be viewed as a smoothing and thresholding operation or as a morphological closure operation. Connected components are now recovered from this image. A minimum bounding rectangle is now constructed using the connected components. The minimum bounding rectangles essentially give a segmentation of the page into words. Figure 3 shows an example. Certain errors do occur; for example, the dot over the i is segmented as a separate word. This is ignored by requiring that word images have a minimum size. Other errors in segmentation may also occur because the writer left a large gap between parts of a word in one instance but did not do so when writing the word again.

A number of algorithms exist in the literature for segmenting words from binary images and essentially any of them can be used [11, 3, 4].

4 Determination of Equivalence Classes

The matching is done in a number of phases. First, the number of possible words that need to be matched is pruned by using the areas and aspect ratios of the words. Since the entire document is written by the same hand, it is expected that variations in size will be small. Thus the pruning can be done on the basis of the area of the word images and the aspect ratios of the word images.

4.1 Pruning

It is assumed that

$$\frac{1}{\alpha} \leq \frac{A_{word}}{A_{template}} \leq \alpha \tag{1}$$

where $A_{template}$ is the area of the template and A_{word} is the area of the word to be matched. A typical value of α used in the experiments is 1.2. A similar filtering step is performed using aspect ratios (ie. the width/height ratio). It is assumed that

$$\beta \leq \frac{Aspect_{word}}{Aspect_{template}} \leq \beta \tag{2}$$

. The value of β used in the experiments is 1.4. In both the above equations, the exact factors are not important but it should not be so large so that valid words are omitted, nor so small so that too many words are passed onto the matching phase.

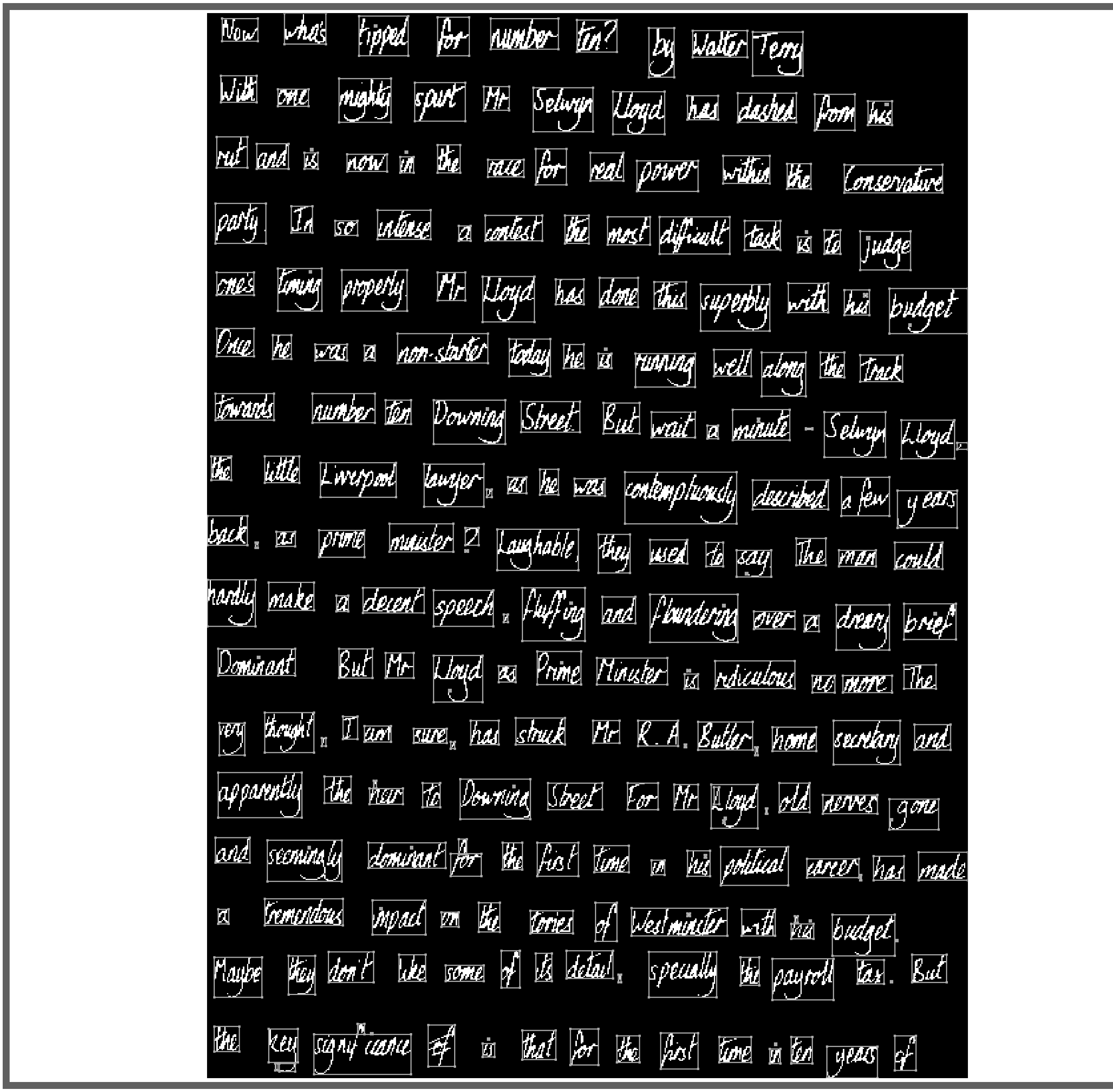


Figure 3: Segmentation of Page

4.2 Matching

The template is then matched against the word of each image in the pruned list. The matching function must satisfy two criteria:

1. It must produce a low match error for words which are similar to the template.

2. It must produce a high match error for words which are dissimilar.

The matching algorithm assumes that no distortions have occured except for relative translation and is fast. This algorithm usually ranks the matched words in the correct order (i.e. valid words are first followed by invalid words). and returns the lowest errors for words which are similar to the template.

The technique used is similar to that used by [6] to match machine generated fonts. Consider two images to be matched. There are three steps in the matching:

1. First the images are roughly aligned. In the vertical direction, this is done by aligning the baselines of the two images. The baseline is computed as follows. The difference in the number of white pixels between adjacent scan lines is computed. The point at which the difference is maximum is declared to be the baseline. The baseline computation is performed for both images, and the images then shifted so that they are aligned.

   In the horizontal direction, the images are aligned by making their left hand sides coincide.

   The alignment is, therefore, expected to be accurate in the vertical direction and not as good in the horizontal direction. This is borne out in practice.

2. Next the XOR image is computed. This is done by XOR'ing corresponding pixels. An example of two images and the corresponding XOR image is shown Figure 2. A match error E_{XOR} may be computed by finding the number of white pixels in the XOR image.

However, the XOR image match error is in general not accurate enough for matching. Notice that XOR images may consist of either isolated pixels or pixels in a blob. The error measure computed above gives equal weight to both. However, an isolated pixel in the XOR image may be due to noise while a blob may be due to a major mismatch. Therefore, blobs should be given more weight. This can be done by using an Euclidean distance mapping.

3. An Euclidean distance mapping [2] is computed from the XOR image by assigning to each white pixel in the image, its minimum distance to a black pixel. Thus a white pixel inside a blob will get a larger distance than an isolated white pixel. An error measure E_{ED} can now be computed by adding up the distance measures for each pixel.

4. Although the approximate translation has been computed using step (1), this may not be accurate and may need to be fine-tuned. Thus steps (2) and (3) are repeated while sampling the translation space in both x and y. A minimum error measure E_{EDmin} is computed over all the translation samples.

5 Experiments

Experiments were performed on a handwritten page obtained from the DIMUND document server on the internet. This page can be obtained from http://documents.cfar.umd.edu/resources/database/ and was scanned by Andrew Senior (this page will be referred to as the Senior document). The handwriting on this page is fairly neat. The page was segmented into words with $k_{horiz} = 9$ and $k_{diag} = 3$ and the resulting output is shown in Figure (3). The algorithm was then run on the segmented words. In the following figures, the first word shown is the template. After the template, the other words are ranked according to the match error E_{ED}. The area threshold α was chosen to be 1.2 and the aspect ratio threshold β was chosen as

1.4. The translation values were sampled to within ± 4 pixels in the X direction and ± 1 pixel in the y direction. Experimentally, this gave the best results.

In Figure (4), the template is the word "Lloyd". The figure shows that the four other instances of "Lloyd" present in the document are ranked before any of the other words. As Table (1) shows, the match errors for other instances of "Lloyd" is less than that for any other word. In the table, the first column is the Token number (this is needed for identification purposes), the second column is a transcription of the word, the third column shows the area in pixels, the fourth gives the match error and the last two columns specify the translation in the x and y directions respectively. Note the significant change in area of the words.

Figure 4: Rankings for template "Lloyd" (the rankings are ordered from left to right and from top to bottom).

Figure (5) and Table (2) display the results when the template "the" is used. In this case, there are a few instances where other words are ranked ahead of two instances of "the". Token numbers 191,33 and 161 are ranked ahead. Note that two of these 191 and 161 are actually instances of "he" which is fairly close to the correct word. In general it is expected that small words will have the largest errors. However, most small words are stopwords and

Token Number	Word	Area	E_{EDmin}	Xshift	Yshift
105	Lloyd	1360	0.000	0	0
70	Lloyd	1224	0.174	0	0
165	Lloyd	1230	0.175	-2	0
197	Lloyd	1400	0.194	4	0
239	Lloyd	1320	0.197	-3	0
21	Maybe	1147	0.199	-1	0
180	along	1156	0.200	1	0
215	party	1209	0.202	1	0
245	spurt	1170	0.205	-1	0
121	dreary	1435	0.206	3	0

Table 1: Rankings and Match Errors for template "Lloyd".

are not useful for indexing. Therefore, the errors are not necessarily serious.

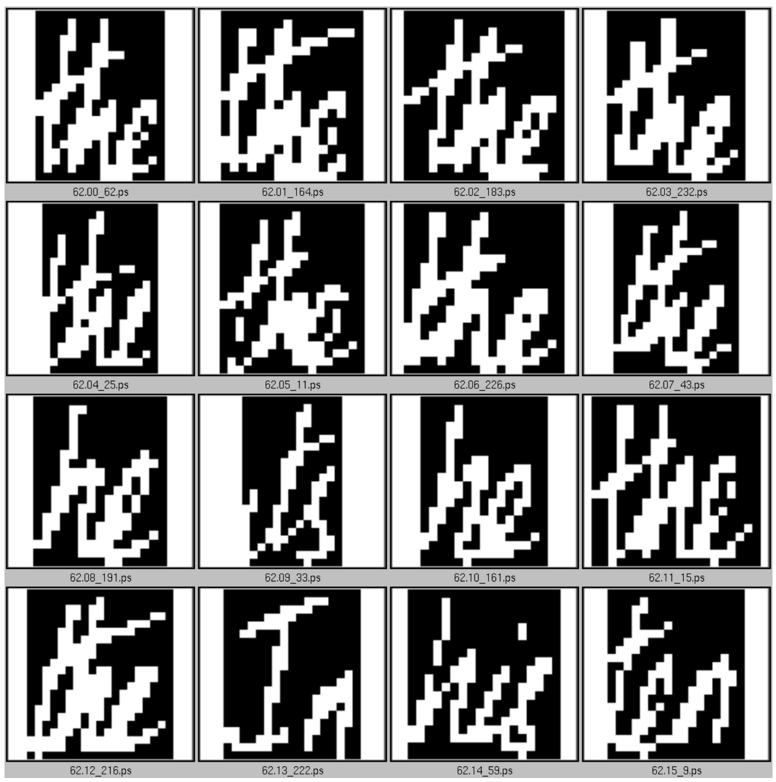


Figure 5: Rankings for template "the" (the rankings are ordered from left to right and from top to bottom).

In English, the first letter in a word is capitalized when the word begins a sentence and not otherwise (unless it is a proper noun). Thus it is desirable that the technique be relatively insensitive to this capitalization. Figure (6) and Table (3) shows an example of

Token Number	Word	Area	E_{EDmin}	Xshift	Yshift
62	the	336	0.000	0	0
164	the	304	0.143	-3	0
183	the	380	0.149	-1	0
232	the	396	0.170	1	0
25	the	330	0.193	0	0
11	the	378	0.223	0	0
226	the	380	0.256	0	0
43	the	391	0.259	0	0
191	he	285	0.265	2	0
33	its	286	0.265	2	0
161	he	300	0.271	1	0
15	the	400	0.280	-1	0
216	the	418	0.289	0	0
59	In	357	0.312	3	0
222	his	360	0.315	0	0
9	ten	357	0.333	1	0

Table 2: Rankings and Match Errors for template "the".

this. The word "minister" is the highest ranked word obtained for the template "Minister" despite the fact that "minister" begins with a lower case letter while "Minister" starts with an uppercase letter.

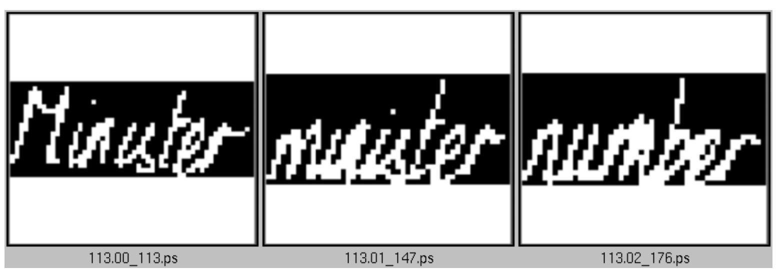


Figure 6: Rankings for template "Minister" (the rankings are ordered from left to right and from top to bottom).

Token Number	Word	Area	E_{EDmin}	Xshift	Yshift
113	Minister	1134	0.000	0	0
147	minister	1078	0.210	-1	0
176	number	1104	0.285	2	0

Table 3: Rankings and Match Errors for template "Minister".

Comment The overall performance of the technique on the Senior document is good. This is especially remarkable considering that there is some variation in the words and this variation is not modelled by the algorithm.

The performance of the method is expected to correlate with the quality of the handwriting. This was verified by running experiments on a page from the Hudson collection (Figure 1). The handwriting in the Hudson collection is difficult to read even for humans looking at grey-level images at 300 dpi (legend has it that Erasmus Hudson was berated by his wife for his bad handwriting). The writing shows wide variations in size - for example, the area of the word “to” varies by as much as 100% ! However, this large a variation is not expected to occur and is not seen when the words are larger. Since humans have difficulty reading this material, we do not expect that the method will perform very well on this document. However, it clearly shows where the method breaks.

For segmentation into words, $k_{horiz} = 8$ and $k_{diag} = 2$ were used. Figure (7) and Table (4) show the results of matching the template “to” (The question mark implies that the word cannot be deciphered). Among the top 14 words ranked are 8 examples of “to”. There are 13 instances of “to” in the document - many of the ones not found show considerable variation from the template. Notice in particular the second picture from the right in the bottom row. This word is “to” although it appears to be more like “its”.

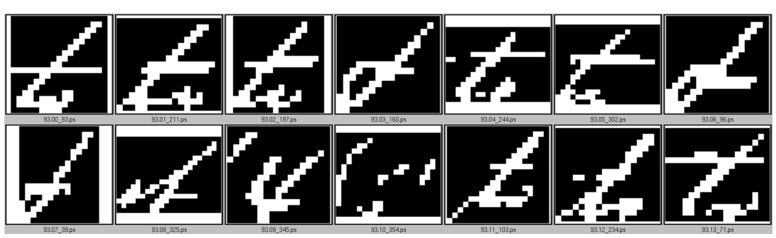


Figure 7: Rankings for template “to” (the rankings are ordered from left to right and from top to bottom).

Figure (8) and Table (5) show the results of matching the template “they”. There is only one other instance of “they” in the document and it is ranked correctly.

Token Number	Word	Area	E_{EDmin}	Xshift	Yshift
93	to	289	0.000	0	0
211	to	255	0.121	1	0
197	to	272	0.135	-3	0
160	of	240	0.159	0	0
244	to	315	0.170	-1	0
302	to	336	0.176	1	0
96	of	272	0.201	-1	0
39	0f	238	0.215	0	0
325	if	352	0.225	-2	0
345	’4	272	0.249	-4	0
354	?	320	0.249	3	0
103	to	342	0.263	2	0
234	to	288	0.280	1	0
71	to	288	0.280	-1	0

Table 4: Rankings and Match Errors for template “to”.

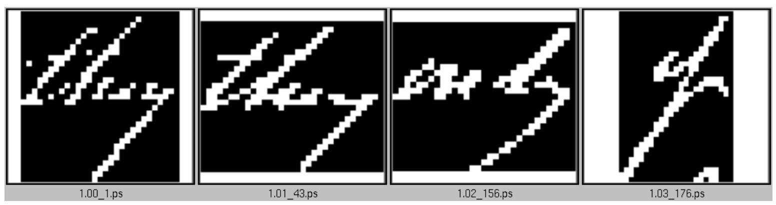


Figure 8: Rankings for template “they” (the rankings are ordered from left to right and from top to bottom).

Token Number	Word	Area	E_{EDmin}	Xshift	Yshift
1	they	899	0.000	0	0
43	they	891	0.145	3	0
156	only	775	0.182	0	0
176	if	782	0.192	0	0

Table 5: Rankings and Match Errors for template “they”.

Finally the word “Standard” is matched. Figure (9) and Table (6) show the results of this matching. The performance is not very good. The reason is that the words are written differently. In the template, there is a gap between the “t” and the “a”. However, in the second example of “Standard” there is no gap. This implies that a technique which models some kind of distortion may be needed.

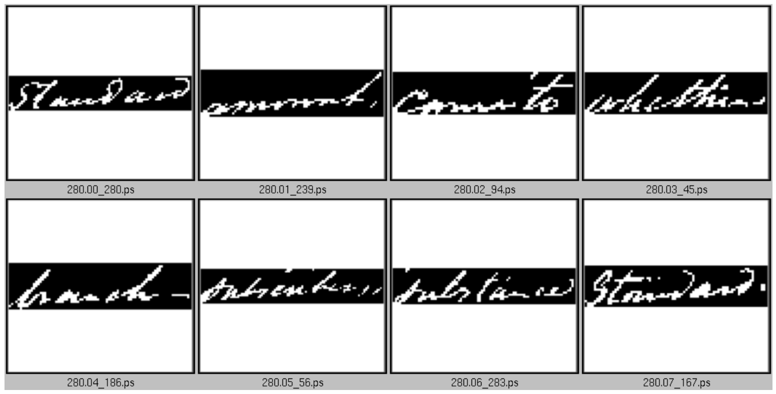


Figure 9: Rankings for template "Standard" (the rankings are ordered from left to right).

Token Number	Word	Area	E_{EDmin}	Xshift	Yshift
280	Standard	1530	0.000	0	0
239	comment	1722	0.203	-4	0
94	come to	1241	0.212	1	0
45	whether	1258	0.212	1	0
186	branch	1743	0.218	0	0
56	subscribes	1900	0.228	-4	0
283	substances	1479	0.231	1	0
167	Standard	1440	0.231	1	0

Table 6: Rankings and Match Errors for template "Standard".

Comment The Hudson document is a good example of the weaknesses of the technique. However, considering the poor quality of the handwriting (difficult to read even for humans), the method performs reasonably well. We are currently investigating techniques which perform much better on both the Senior and Hudson documents [7].

6 Conclusion

The work clearly shows that the idea of indexing a corpus of written words in a single hand is feasible. Even a simple matching algorithm which accounts for only translational shifts is able to match most of the words when the handwriting is good. The performance degrades with the quality of the handwriting, although surprisingly it does reasonably well on even poor quality handwriting. Future investigations will focus on more powerful matching techniques which compensate for some of the distortions (variation) between words.

7 Acknowledgements

We would like to thank Gail Giroux and the Umass Library for the scanned page from the Hudson collection. We also wish to thank Bob Heller and Jonathan Lim for systems help.

References

[1] M. Bokser. Omnidocument technologies. *Proceedings IEEE*, 80(7):1066–1078, 1992.

[2] Per-Erik Danielsson. Euclidean distance mapping. *Computer Graphics and Image Processing*, 14:227–248, 1980.

[3] L. Fletcher and R. Kasturi. A robust algorithm for text string separation from mixed text/graphics images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10:910–918, 1988.

[4] G. J. F. Jones, J. T. Foote, K. Sparck Jones, and S. J. Young. Video mail retrieval: The effect of word spotting accuracy on precision. In *International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 309–316, 1995.

[5] Siamak Khoubyari and Jonathan J.Hull. Keyword location in noise document image. In *Second Annual Symposium on Document Analysis and Information Retrieval, UNLV, Las Vegas*, pages 217–231, 1993.

[6] R. Manmatha, Chengfeng Han, and E. M. Riseman. Word spotting: A new approach to indexing handwriting. Technical Report CS-UM-95-105, Computer Science Dept, University of Massachusetts at Amherst, MA, 1995.

[7] S. Mori, C. Y. Suen, and K. Yamamoto. Historical review of ocr research and development. *Proceedings of the IEEE*, 80(7):1029–1058, 1992.

[8] G. Salton. *Automatic Text Processing*. Addison-Wesley, 1988.

[9] H.R. Turtle and W.B. Croft. A comparison of text retrieval models. *Computer Journal*, 1992.

[10] F. Wahl, K. Wong, and R. Casey. Block segmentation and text extraction in mixed text/image documents. *Computer Vision Graphics and Image Processing*, 20:375–390, 1982.

[11] D. Wang and S. N. Srihari. Classification of newspaper image blocks using texture analysis. *Computer Vision Graphics and Image Processing*, 47:327–352, 1989.