Indexing and Retrieval of On-line Handwritten Documents

Anil K. Jain and Anoop M. Namboodiri
Department of Computer Science and Engineering
Michigan State University
East Lansing, MI 48824
{jain,anoop}@cse.msu.edu

Abstract

Recent advances in on-line data capturing technologies and its widespread deployment in devices like PDAs and notebook PCs is creating large amounts of handwritten data that need to be archived and retrieved efficiently. Word-spotting, which is based on a direct comparison of a handwritten keyword to words in the document, is commonly used for indexing and retrieval. We propose a string matching-based method for word-spotting in on-line documents. The retrieval algorithm achieves a precision of 92.3% *at a recall rate of* 90% *on a database of* $6,672$ *words written by* 10 *different writers. Indexing experiments show an accuracy of* 87.5% *using a database of* $3,872$ *on-line words.*

Keywords: Online Document, Word Spotting, Indexing, Document Retrieval.

1. Introduction

The increase in popularity of computing devices which accept handwritten input such as PDAs, Tablet PC [2], etc. has resulted in an increasing demand for algorithms that can be used to efficiently store and retrieve such data. The most efficient method for storage of handwritten text would be to convert the text into ASCII code using a text recognizer and store the resulting text document. This method enables efficient storage and retrieval of documents based on keywords. However, the accuracy of text recognizers is highly dependent on the individual writing styles. Text recognizers also need user intervention for reliable translation. Moreover, handwritten data often contain diagrams and sketches and so it is desirable to store the raw data (*digital ink*) as well. The limitations of recognition algorithms and the expressive power of handwriting over text have led researchers to explore the idea of using digital ink as a basic data type. Another reason for requiring a recognition-free solution is the multitude of languages and symbols that an application using pen-based input needs to handle.

An alternate approach to retrieval of handwritten data is to use a search technique which can find a document in a database using a handwritten keyword as a query. The process of comparing handwritten or spoken words in a document, without explicit recognition is referred to as 'word-spotting' [10].

Handwritten data, which is captured (digitized) at the time of writing, encodes the dynamic information in the strokes [1] and is referred to as on-line handwriting (see section 1.1). Recent advances in the processing of on-line handwritten data includes algorithms for *(i)* segmentation of handwritten text into lines, words and sub-strokes [13, 8], *(ii)* character and word recognition [15, 12, 5], *(iii)* analysis of on-line document structure [6], and *(iv)* indexing and retrieval of on-line handwritten documents.

Indexing is the problem of dividing the words in a database into equivalence classes, each consisting of instances of a specific word. Indexing algorithms use a similarity measure for pairwise comparison of words to form clusters of words. Representative instances from each cluster/class are then used to create a word index for the database. In contrast, the problem of word-spotting deals with retrieval of documents by comparing a keyword with individual words in the documents in the database (see figure 1(a)). Although the indexing process may seem functionally similar to retrieval, there are many aspects of indexing applications which make it different from retrieval [10]. However, the critical factor in solving both the indexing and retrieval problems is the choice of a distance measure between words, which should have a small value (dissimilarity) for the instances of the same word and a large value for different words.

Several approaches to word-spotting in documents have been reported in the literature. However, these studies are restricted to off-line handwritten or printed documents and audio documents for the most part. Kuo and Agazzi [7]

[1] A stroke is defined as the locus of the tip of the pen from pen-down to the next pen-up position.

Proceedings of the Seventh International Conference on Document Analysis and Recognition (ICDAR'03)
0-7695-1960-1/03 $17.00 © 2003 IEEE

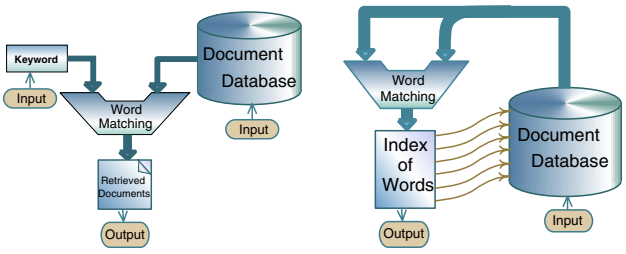


Figure 1. Indexing and retrieval problems. Schematic diagram of (a) a word-spotting based retrieval system; (b) an indexing system.

used a Pseudo 2-D HMM model to spot keywords in poorly printed documents. They achieved 96% accuracy on a synthetically generated dataset with words of different font sizes. Curtins [4] employs multi-font character templates to match individual characters to spot keywords in noisy printed text and attains a recall rate of 90% and precision rate of 97% on a set of 380 document images. O'Neill et al. [11] reported 90% recall and more than 95% precision in spotting printed words using moment features of the word pixels. The test set contained two words in 13 font sizes with 20% salt-and-pepper noise.

In the case of handwritten documents, word-spotting systems attain considerably lower recall rates due to the variability in handwritten words. Manmatha et al. [10] used an Euclidean distance map-based algorithm to index handwritten text using word image templates. Kolcz et al. [3] used the profile of the words to match handwritten words. Their method achieves a recall rate of 45% with no false alarms. Singer and Tishby [14] modeled on-line handwriting as modulated cycloidal motions of the pen tip to do recognition. The authors reported successful spotting of parts of a word in a small database of 30 words using dynamic time warping. Lopresti and Tomkins [9] used an edit distance measure to match a sequence of feature vectors extracted from segments of strokes in on-line handwritten data. The method achieves a recall rate of 95% with a precision of 4% *to* 7% on databases containing $2,000$ and $4,000$ words, respectively, by two writers.

1.1. Online Documents

There are a few important aspects of on-line documents that enable us to process them in a fundamentally different way than off-line documents. The most important characteristic of on-line documents is that they capture the temporal sequence of strokes while writing the document. This allows us to analyze the individual strokes and use the additional temporal information for matching the keyword to the words in a document.

In the case of on-line documents, segmentation of foreground from the background is a relatively simple task as the captured data, i.e. the (x, y) coordinates of the locus of the stylus, define the characters and any other point on the page belongs to the background. We extract both the spatial and temporal information from the strokes in a word for matching purposes. Unfortunately, the temporal information also leads to large intra-class variations of strokes in each class (word). Figure 2 shows six samples of the word 'computer', written by six writers. The strokes in each word are connected together in the writing order. Note the differences in writing styles of the characters *m*, *p*, *t* and *r*. The intra-class (word) variability in on-line documents arises from a variety of sources such as *(i)* writing styles of the users, *(ii)* writing surface of the data capturing device, *(iii)* correction or over-writing of words, which are recorded as additional strokes, and *(iv)* the writing speed.

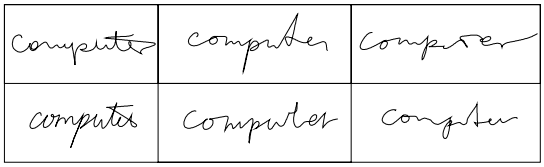


Figure 2. Intra-class (word) variability of On-line Handwriting.

The retrieval and indexing systems based on word-spotting is independent of the language/script of the document. Further, the keyword need not even be a legal text as long as the 'writing' to be spotted is properly segmented from the pages in the database.

2. Data Collection and Pre-processing

The data used in this paper was collected using the *CrossPad* [1]. The *CrossPad* has a pen and paper interface along with the ability to digitally capture the (x, y) position of the pen tip using an *RF* transmitter embedded in the pen. The pen position is sampled at a constant rate of 132 samples per second and the device has a resolution of 254 dpi along the x and y axes [2]. The data was collected on a ruled paper with an inter-line distance of 8.75 mm with text written on alternate lines. We must point out that the actual device for data collection is not important for our study as

[2] Users have reported that the actual resolution is only about half of this value.

long as it can generate a temporal sequence of x and y positions of the pen tip.

Our word-spotting system operates on a database containing a number of handwritten pages. To carry out the matching, we need to segment a page into individual lines and words. Ratzlaff et al. [13] attempted the problem of text line identification in on-line documents. To identify the individual lines, the inter-line distance, d, is computed from the y-axis projection of the strokes in the page. To reliably measure the distance between successive lines from the projection, we compute the autocorrelation of the y-axis projection, whose adjacent peaks are at a distance d apart. Lines are identified by finding valleys in the projection. To avoid local minima, we choose only those points as valleys that have the smallest magnitude within a window of width d.

The text is divided into lines by collecting all the strokes which fall in between two successive line boundaries. The temporal information from stroke order is used to disambiguate strokes which fall across line boundaries and to correctly group small strokes which may fall into an adjacent line. The segmentation of a line into words is done using an x-axis projection of the text in the line. The strokes which fall between two valleys in the projection are collected and labeled as a word. The minimum width of a valley in the projection for word segmentation was experimentally determined as 30 pixels for the resolution of the digitizing device used ($254\ dpi$). Details of the line and word detection algorithm may be found in [6].

During pre-processing, the individual strokes in a word are joined together to form one single stroke. The strokes are then resampled to make the sampled points equidistant. This helps to reduce the intra-class variations in the words due to different writing speeds and to avoid anomalous cases such as having a large number of samples at the same position when the user holds the pen down at a point. The strokes are then smoothed using a lowpass (Gaussian) filter to reduce noise introduced during data capture. The individual strokes are again resampled to make the points equidistant. During the lowpass filtering and resampling operations, the *critical points* in a stroke are retained. A critical point is defined as a point in the stroke where the x or y direction of the stroke changes its sign, in addition to the extreme points (pen-up or pen-down) of the stroke. Figures 3(a)-(d) show an example of preprocessing the on-line character *a* (consisting of a single stroke). Figures 3(e) and (f) show a multi-stroke word before and after pre-processing.

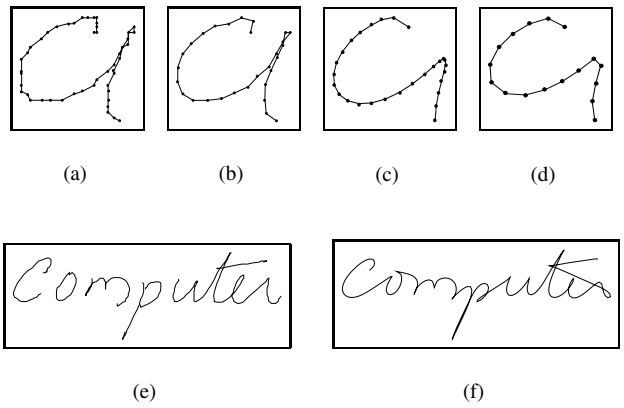


Figure 3. Pre-processing. (a) An input stroke, the dots represent the sampling points. (b) after equidistant resampling. (c) lowpass filtering. (d) result after second resampling. (e) input word, and (f) word after preprocessing. Note that the delayed stroke (in t) is joined to the end of the last stroke.

3. Word Matching

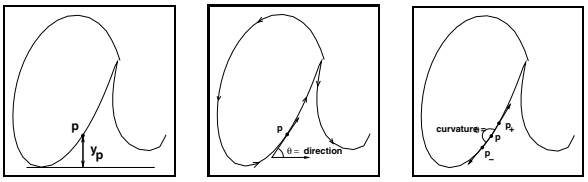
The method used for word matching is based on a string matching technique, called *dynamic time warping*. Dynamic time warping (DTW) is a powerful method for comparing two sequences of data points. To employ DTW, we first preprocess the keyword that is provided by the user as described in section 2. The processed keyword is used as a template string for matching purposes. The words in the documents to be searched are extracted using the techniques mentioned in the previous section. The following three features are computed at each sample point in the word, resulting in a sequence of feature vectors (see figure 4):

1. The height (y) of the sample point: This is the distance of the sample point from the base of the word. The base of the word is defined as the horizontal line passing through the lowest sample point in the word.

2. The stroke direction at p: To stroke direction at p is defined as the positive or counter clock-wise angle between horizontal axis and the tangent of stroke at p. In practice, the tangent is approximated with the line connecting the two neighboring points of p. The writing order is important in the computation of this feature.

3. The curvature of the stroke at point p: We use the angle subtended by the lines joining p and its neighbors at p as a measure of the curvature of the stroke.

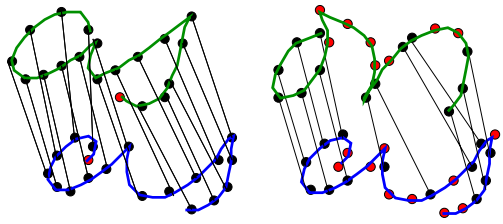
Each word in a document is compared with the keyword. The word to be compared is first scaled so that it is of the same size (height) as the keyword, and translated so that both the words have the same centroid. The DTW technique then aligns the feature vector sequence from a database



(a) (b) (c)

Figure 4. Computing word features. (a) height (y_p) at point p, (b) the direction feature, and (c) computation of curvature at p.

word to that of the keyword using a dynamic programming-based algorithm. The algorithm computes a distance score for matching points by finding the Euclidean distance between corresponding feature vectors and penalizes missing or spurious points in the word being tested. The optimal alignment between the two words is computed, with each feature weighted differently. The weights are learned during the training phase of the algorithm. Figure 5 illustrates the correspondence between two words in the database. Note that our algorithm does not match the spatial coordinates, but matches feature vectors computed at the corresponding points. The distance score computed is scaled based on the length of the keyword. Since longer strings tend to have larger values of distances with the keyword, we normalize the distance measure by dividing it with the sequence length.



Distance Score = 34.6 Distance Score = 111.7

(a) (b)

Figure 5. Correspondences between (a) two instances of the word *as*, and (b) between the words *on* and *as*.

The string matching algorithm uses a number of parameters including the weights for each of the features and the penalties for missing (or spurious) points in the sequence. These parameters were determined from an independent training set collected from a single user. The training set consisted of 5 occurrences of 56 different words and were written in a single session. The penalty for missing and spurious points were determined to be 200 and the weights for the features were, $height = 3.0$, $direction = 3.0$ and $curvature = 1.0$.

4. Experimental Results

The test set consisted of a passage of 30 pages (410 lines and 3,872 words) collected from a single writer and a set of 280 keywords (5 instances each of 56 words) collected from 10 writers (a total of 2,800 words). The database was collected over a period of two weeks in over five sessions. To measure the performance of the algorithm, the words in the database were manually labelled (ground truth) after detection of lines and words (a total of 6,672 words). Figure 6 show examples of pages from our on-line database.

(a) (b)

(c) (d)

Figure 6. Sample pages from the database. (a)-(c) samples of keywords from 3 different writers. (d) sample page from the passage.

Every keyword in the database was compared with every other word written by the same writer and the results were used to measure the performance of the algorithm. For the passage, the set of keywords written by the same writer was used for document retrieval.

One of the common measures of performance of retrieval algorithms is the precision vs. recall (*PR*) curve. Precision refers to the percentage (or proportion) of correct retrievals out of the total number of words retrieved. Recall denotes

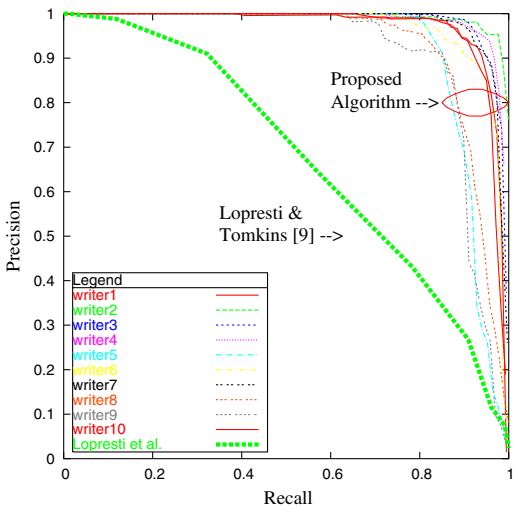


Figure 7. The precision vs. recall curves.

the fraction of the number of correct retrievals to the total number of instances of the keyword in the database. Figure 7 shows the *PR* curves for ten different users who contributed to the database. The results reported by Lopresti and Tomkins [9] is also plotted along with the curves for comparison. This is the only work, to our knowledge, which has reported the *PR* curve for on-line documents. The accuracy of our algorithm was 92.3% at a recall rate of 90% averaged over the whole database ($6,672$ words). This is significantly better than the results in [9].

The indexing experiment was carried out on the passage of 30 pages. We discard short words (consisting of 3 characters or less) to avoid articles, prepositions etc. The word matching algorithm computes the distance between every pair of words; word pairs with distance less than a threshold (determined by 90% recall rate) were joined together to form word classes. The label (ground truth) of every class is set to the word which is in majority in the class. Each label occurs only once in the index. Every word instance which is not classified under its label is counted as an error. Our algorithm achieved an accuracy of 87.5% with $1,461$ words (after removing short words, the original database of $3,872$ words was reduced to $1,461$ words) being clustered into 943 word classes. Most of the errors were due to grouping/clustering of similar words (e.g., *foot* and *feet*).

The algorithm takes approximately 9 $msecs$, on an average, for a word comparison on a machine with an Intel P-III 650 Mhz CPU and 192 MB of RAM.

5. Conclusions and Future Work

We have proposed a string matching based word-spotting algorithm using features computed from the strokes of individual words. The algorithm achieves a precision of 93.2% at a recall rate of 90% averaged over 10 writers. Indexing experiments show an accuracy of 87.5% using a database of $3,872$ words. These accuracies are quite good considering the large intra-class variability encountered in on-line handwritten words. We are currently extending our algorithm to match non-text regions such as line drawings. Computation of the feature vectors can be adapted to handle devices of different resolutions. The algorithm may also be optimized to improve the run time performance.

References

[1] IBM Pen Technologies. http://www.research.ibm.com/handwriting/.

[2] Pen Computing Magazine: PenWindows. http://www.pencomputing.com/frames/tablet_pc.html.

[3] Word-spotting studied in cursive writing using query-by-example search approach. *Electronic Imaging Optical Engineering Reports*, 8(2), December 1998. http://www.spie.org/web/oer/december/dec98/eitg.html.

[4] J. De Curtins. Comparison of OCR versus word shape recognition for key word spotting. In *Proc. of the SDIUT*, pages 205–213, Annapolis, 1997.

[5] S. Jaeger, S. Manke, J. Reichert, and A. Waibel. Online handwriting recognition: The NPen++ recognizer. *Int. J. on Document Analysis and Recognition*, 3(3):169–180, 2001.

[6] A. K. Jain, A. M. Namboodiri, and J. Subrahmonia. Structure in on-line documents. In *Proc. of the* 6^{th} *ICDAR*, pages 844–848, Seattle, September 2001.

[7] S. S. Kuo and O. E. Agazzi. Keyword spotting in poorly printed documents using pseudo 2-d hidden markov models. *IEEE Trans. on PAMI*, 16(8):842–848, August 1994.

[8] X. Li, M. Parizeau, and R. Plamondon. Segmentation and reconstruction of on-line handwritten scripts. *Pattern Recognition*, 31(6):675–684, June 1998.

[9] D. Lopresti and A. Tomkins. On the searchability of electronic ink. In *Proc. of the* 4^{th} *IWFHR*, pages 156–165, Taipei, December 1994.

[10] R. Manmatha, C. Han, and E. Riseman. Word spotting: A new approach to indexing handwriting. In *Proc. of the IEEE CVPR Conference*, pages 631–637, San Francisco, June 1996.

[11] J. C. O'Neill, A. O. Hero, and W. J. Williams. Word spotting via spatial point processes. In *Proc. of the IEEE ICIP*, pages 215–219, Sept. 1996.

[12] R. Plamondon and S. N. Srihari. On-line and off-line handwriting recognition: A comprehensive survey. *IEEE Trans. on PAMI*, 22(1):63–84, 2000.

[13] E. H. Ratzlaff. Inter-line distance estimation and text line extraction for unconstrained online handwriting. In *Proc. of the* 7^{th} *IWFHR*, Nijmegen, Netherlands, September 2000.

[14] Y. Singer and N. Tishby. Dynamical encoding of cursive handwriting. *Biological Cybernetics*, 71(3):227–237, 1994.

[15] T. Wakahara, H. Murase, and K. Odaka. On-line handwriting recognition. *Proceedings of the IEEE*, 80(7):1181–1194, July 1992.