SIMILARITY SEARCHING FOR ON-LINE HANDWRITTEN DOCUMENTS

Sascha Schimke

Otto-von-Guericke University Magdeburg,
Germany, School of Computer Science
sascha.schimke@ovgu.de

Claus Vielhauer

Brandenburg University of Applied Sciences,
Department of Intformatics & Media

Otto-von-Guericke University Magdeburg,
Germany, School of Computer Science
claus.vielhauer@
{fh-brandenburg.de;ovgu.de}

ABSTRACT

With the increasing pervasion of computers, the handwriting seemed to forfeit its position as the primary way of permanent expression of humans ideas; typed texts appeared as the new and better solution. However, with the today's rise of modern pen based computer devices (e.g. TabletPC), we may see a renaissance of the traditional handwriting in the digital world. More and more electronic documents will be written with pens directly on the screen. One of the benefits of digital documents in comparison to paper, is the convenience of automated document management, including retrieval. For example, full text search in large amounts of sheets of papers is a time-consuming task in the analog world, while modern search engines demonstrate everyday the simplicity of the same task in the future digital world. The continuous breakthrough of digital handwriting will require search possibilities as they exist for typed documents. This paper discusses problems related to search in digital handwriting data and describes a novel approach to solve this searching problem.

KEYWORDS

database searching – document retrieval – on-line handwriting

1. INTRODUCTION

The rising spread of computers with pen sensitive screen surfaces or special digital pen devices will lead to a growing amount of digital handwritten documents. As compared to keyboard typing, using a pen is often less disturbing for example when taking meeting notes or may be more convenient for rapid drawing of sketches or diagrams. Until now, a main problem of digital handwritten documents is difficult management because of the lack of appropriate retrieval mechanisms. A straightforward idea of course would be to use textual recognition to translate the written inputs in digital text documents, so that usual text indexing and searching algorithms can be utilized. However, this approach may be problematic because of the existing error rates of today's handwriting recognition systems, especially for persons with unclear writing style or those with slightly supported non-Latin scripts. The problem is even worse, if no textual content at all can be recognized, namely for drawn or sketched documents. This paper addresses this problem by using a novel shape based search approach for finding words, phrases or symbols without any textual recognition.

With respect to data acquisition, different classes of devices for digital handwriting can be distinguished. The most popular class of devices for the moment are small computers (with or without a keyboard) equipped with pen sensitive screens – personal digital assistants (PDA) along with some actual mobile phones with PDA features, Tablet PCs or, recently, so called ultra mobile PCs (UMPC) are examples. The second class is formed by large black- or whiteboards with sensors for capturing the pen tip position while writing (e.g. Xerox Liveboard [1] or mimio Xi [2]). The third important class is composed of pen-and-paper combinations, where an autonomous system captures and stores the written and drawn contents (e.g. IBM CrossPad [3], Logitech ioPen [4] or Pegasus PC Notes Taker [5]). All these devices are able to acquire data about the pen movement *during the writing process* as sequences of pen-tip coordinates and sometimes also pressure information. This sampling scheme is the so called *on-line* approach, in contrast to *off-line* handwriting acquisition, where the handwritten content is optically scanned *after the writing process*. The on-line approach is considered as more appropriate for recognition than the off-line one, due to the availability of more process-related data, like stroke direction and ordering and timing information [6][7].

The main contributions of this paper are a) the presentation of a new algorithm for searching within handwritten data, b) the discussion of different features types and their respective parameters and c) an evaluation of the new approach using a test database and a comparison of the results with those, which were achived with related systems.

This paper is structured as follows: Section 2 gives a short overview about related work as published in the literature and describes the algorithmic basics of our system. In section 3 details about our algorithm are described and Section 4 explains the aspect of determining appropriate features for our system. Our test setting and first experimental results are presented in section 5.

2. PRELIMINARY REFLECTIONS

2.1. Related Work

As mentioned in the introduction, the most obvious approach for searching handwritten documents is the *textual recognition* and then searching in the resulting set of character data. This approach is described inter alia in [8][9]. The advantage of this approach is the high searching performance by using powerful full-text indexing strategies. On the other hand, the main disadvantages are the relative high error rates of the actual handwriting recognizers (when used with unclear writing styles) and the problem of recognizing non-text contents like sketches or diagrams.

Another approach for searching in handwritten texts is the *word spotting*, which works on the basis of shape comparison of query words and the words of the documents instead of recognizing the words [10][11][12]. Using this approach, it is possible to search for pen based shapes without any interpretation of the content. The disadvantage of the published systems is the need for a good word-wise segmentation of the handwritten documents during a preprocessing step. If the segmentation fails, the retrieval will fail, too. The published systems in the literature differ in the way of performing the comparison between words or other pen shapes. For example in [11] and [12] the handwritten inputs are transformed into sequences of states, which describe the path of the pen-tip while writing, i.e. the shape of the input. The word comparison then is realized by sequence matching, e.g. using dynamic time warping.

Our approach is related to the latter one in the sense that we extract shape describing sequences out of on-line handwriting signals, but we match them without previous word segmentation. Due to the influence of individual writing style on the shape of handwriting inputs, we consider only intra personal searching.

2.2. String Algorithms

The algorithmic basis of our handwriting search system is the comparison of sequences over a finite alphabet, i.e. strings. The most widely used comparison technique for strings is a dissimilarity measure called *edit-distance*. The background of this measure is the idea that two strings are similar, if only a few operations are needed to transform one string into the other. In the original edit-distance (also called Levenshtein distance [13]), the supported operations are *insertion*, *deletion* and *substitution* of individual characters. The *Hamming distance* for strings can be interpreted as a variant of the edit-distance, where only the substitution operation of characters is allowed. Another variant is the *Damerau-Levenshtein* distance, which allows insertion, deletion, replacement and additional character transposition operations [14]. The classical solution for obtaining the edit-distance of two strings, i.e. the minimal needed number of operations, is to use dynamic programming techniques with a asymptotic time complexity of $O(m \times n)$, where m and n are the lengths of the strings. The classical edit-distance can be used for example for correction of spelling errors in word processing and has been successfully adapted for comparison of handwritten signatures for biometric applications [15].

Another variant of the edit-distance is the *local similarity*, where for two strings the longest substrings are computed, having minimal edit-distance. This technique is used in bioinformatics for finding similarities between genomic data [16].

A further variant is the *approximate string searching* which computes the edit-distance between one string and all substrings of a second string. That way it is possible to find all *similar* appearances of a short string within a longer string [17]. We have adopted the later approach and extended it towards our search system for on-line handwritten documents. Section 3 describes in detail this searching algorithm and necessary adaptations for our purpose of handwriting retrieval.

3. APPROXIMATE STRING SEARCHING

The approximate string searching problem, i.e. the fuzzy search for all appearances of a short string q (query) within a long string d (document) can be realized by filling a matrix D of the size $(m+1) \times (n+1)$ with m and n being the string length of q and d, respectively (see equation 1) [18].

$$D(i,j) = \begin{cases} 0 & \text{if } i = 0, \\ D(i-1,0)+1 & \text{if } i > 0, \\ & \text{and } j = 0, \\ \min \left\{ \begin{array}{l} D(i,j-1)+1 \\ D(i-1,j)+1 \\ D(i-1,j-1)+\delta(i,j) \end{array} \right\} & \text{else.} \end{cases} \tag{1}$$

Where the function $\delta(i,j)$ indicates the cost for substitution of the i^{th} character of query string q by the j^{th} character of document string d:

$$\delta(i,j) = \left\{ \begin{array}{ll} 0 & \text{if } q[i] = d[j], \\ 1 & \text{else.} \end{array} \right. \tag{2}$$

To reduce the $O(m \times n)$ memory complexity, it is possible to calculate matrix D column-wise and hold only the two actual columns. After calculation, the matrix row $D(m, 0 \ldots n)$ contains the edit distances between the query string q and a substring of document d, which ends at the position j of d. To find those positions j, which indicate a match of the query string q, the matrix element $D(m,j)$ has to be smaller than a threshold τ: $D(m,j) < \tau$. Like in other classification systems, if τ is chosen too small, the missing rate increases and so the recall rate decreases. On the other hand, by choosing τ too large, the mismatch rate increases and so the precision declines. In theory, $D(m,j)$ can not be greater than m, but practical experiments with random strings q and d (uniformly distributed randomness) over a finite alphabet A show, that $D(m,j)$ in average is smaller (see figure 1).

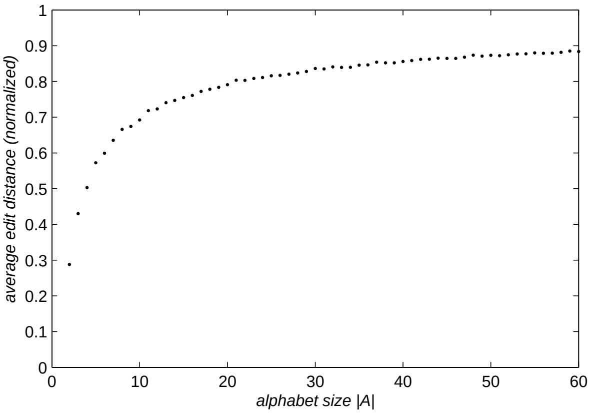

Figure 1: *Normalized edit distance as a function of alphabet size $|A|$ while searching for a short random string (100 characters) within a long random string (10,000 characters) – averaged results of 100 tests.*

As can be seen in figure 1, the averaged normalized edit distance $(D(m,j)/m)$ tends to become greater, the longer the alphabet is. If the threshold τ for the maximal allowed edit distance is chosen greater than this averaged value for the respective alphabet size, in practice the mismatch rate rises rapidly.

4. FEATURES FOR RETRIEVAL

To be able to use the approximate string searching algorithm for searching in handwritten documents, it is necessary to extract string-like feature data from the handwriting signals (i.e. time discrete signals of pen tip position x_t, y_t, and binary pen pressure p_t). In context of the work presented in this paper, we studied four different types of these string features.

4.1. Freeman Grid Codes

The first feature type bases on coding of line drawings, first presented by H. Freeman in [19]. The idea is to superimpose the handwriting shape (words, phrases, symbols, ...) with a regular grid (see figure 2-a and to quantize the sampling points with respect to the respective grid nodes. So the original handwriting shape can be expressed as a word over an alphabet, which consists of eight directions (two horizontal, two vertical and four diagonal, if square grids are used). It is possible to consider gaps between pen strokes, but in our experiments, the influence of a gap-coding was not measurable.

It is obvious, that the direction coding tends to be more precise by using a fine grid size but on the other hand, a very fine grid size implies a very long resulting string feature. This feature type has been successfully utilized for the recognition of gesture shape in the domain of pen based graphical user interface design [20].

Besides using the square grids, it is possible to use grid on the base of other geometric shapes, if these shapes are able to be packed compact. Thus, regular triangles or hexagons are alternative solutions. The advantage of grids consisting of these shapes is that all neighbor nodes of a grid node have the same distance, while in square grids the diagonal distance is higher than the horizontal and the vertical. A disadvantage is the higher computational complexity of a grid quantization using triangular or hexagonal base shapes.

4.2. Direction based Codes

A general problem of Freeman grid codes is the strict limitation of the possible directions to eight for square grids (six for triangular and three for hexagonal grids). To overcome this problem, it is possible to use the direction of the pen stroke between two consecutive sampling points as the basis for a string like feature. Because of the varying distance of the raw sampling points as a result of varying writing speed, an equidistant re-sampling (i.e. regularly spaced in arc length) of the original handwriting signals x_t and y_t is necessary, as illustrated in figure 2-b. The re-sampling can be performed using cubic spline interpolation [21].

For two consecutive re-sampled points (x_t, y_t) and (x_{t+1}, y_{t+1}), the direction α_t is calculated as follows (see figure 2-c:

$$\alpha_t = \arctan \frac{y_{t+1} - y_t}{x_{t+1} - x_t} \quad (3)$$

To obtain a string of symbols over a finite alphabet, the resulting direction α_t has to be quantized. The most simple way is to divide the complete possible range of α_t $(0 \leq \alpha_t < 2\pi)$ into s equal parts, which leads to an alphabet of the size s. We may expect an increase in coding precision with increasing s.

Another parameter, besides the degree of direction quantization, is the arc length of two consecutive points after the re-sampling. Similar to the Freeman grid coding, here the length of the resulting string is directly dependant on this arc length.

4.3. Curvature based Codes

Additionally to the local direction of pen strokes, the local curvature of the handwriting can be the base of a shape describing string coding. There are two simple methods for the estimation of this local curvature; a direction-based and a circular-based approach. Both approaches are explained briefly, but only the former one will be used in evaluation. The most straightforward way to estimate the curvature of a pen stroke is to use the difference of two consecutive local directions, i.e. the curvature κ_t could be defined as the difference of α_t and α_{t+1}. The actual curvature string coding is done in the same way as the directional string coding by quantizing the value of κ_t in s steps.

Besides this curvature estimation based on direction differences, it is possible to use the radius of the circle, which is described by three consecutive sampling points. The curvature is the higher the smaller is this radius and vice versa. In this paper we use the direction difference approach to estimate the local curvature of a pen stroke.

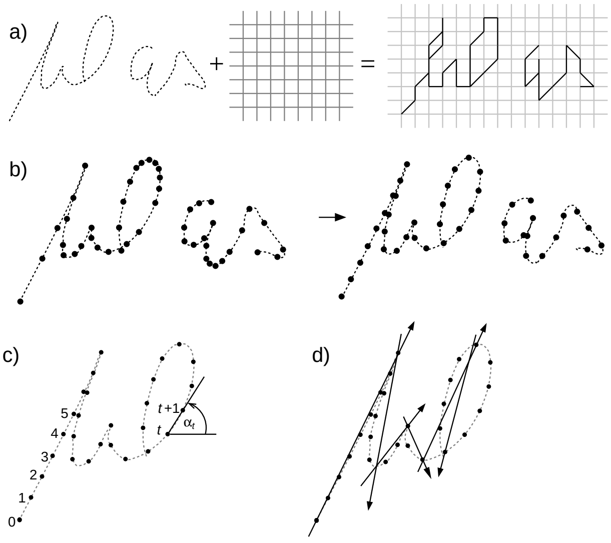

Figure 2: *Extraction of string features from handwriting signals and preprocessing. a) grid-based direction features according to Freeman [19], b) equidistant re-sampling of handwriting signals, c) direction of secant between consecutive sampling points after equidistant re-sampling, d) word slant as direction of consecutive Y-minima and -maxima.*

4.4. Slant based Codes

A fourth method for extracting string like features from handwritten inputs bases on the slant of pen strokes. The idea is to measure the angle σ_t of the connection between a vertical minimum and the consecutive vertical maximum or a vertical maximum and the consecutive vertical minimum, respectively [22]. Figure 2-d shows some examples for these directions. A great advantage of this approach over those described in sections 4.1 to 4.3 is the very small length of the resulting strings. As discussed in section 3 the asymptotic time complexity for performing the approximate string searching is quadratic with the size of the inputs ($O(m \times n)$), so the time performance benefits from smaller strings. Parallel to the direction and the curvature based codes, for slant based coding the directions σ_t need to be quantized in s steps.

A disadvantage of this slant based approach could be the insufficient expressive power to describe pen handwriting shapes and out of it a low resultant retrieval performance.

Note that these four string features constitute an initial set for our studies. Our suggested string matching concept can be adapted to additional features in the future.

5. TEST SESSTING AND TEST RESULTS

5.1. Dataset and Test Environment

For performance evaluation of our system we collected our own test database[1] in a period of the last twelve months based on the following procedure: a number of persons were asked to write at least one page of text. The only requirement for the texts was that they contain some words or phrases more than once. This is necessary to be able to search for a word and find other matches. We decided to collect our own database, as we did not find suitable test data in the community at the time of our test. The only other data sets we found were collections of off-line data, i.e. scanned images of handwritten texts, and the UNIPEN database [23], which does not contain texts but only characters and symbols.

For collecting handwriting data sets, we used the ioPen device of Logitech [4]. This autonomous pen device allows writing on special paper, which is printed with a fine dot pattern. An optical sensor in the pen allows deriving the position within the paper area from the dot pattern. The temporal resolution (i.e. the sampling rate) of the ioPen device is varying and reaches up to 50 Hz. The real spatial resolution is unknown to the authors, but the device driver resulted in effective values of 710 ppi (*points per inch*), i.e. one unit is about 0.036 mm. The pressure is given in 127 different levels, but for the purpose of our handwriting retrieval, only the fact of pressure/no-pressure (pen down/pen up) is regarded. We acquired 59 documents (a document corresponds to one A4 sheet of paper) consisting of 10,990 words and 65 sketches or icons from twelve persons. The document languages are English and German (see figure 3 for an example of a short handwritten text and a collection of symbols and a sketch).

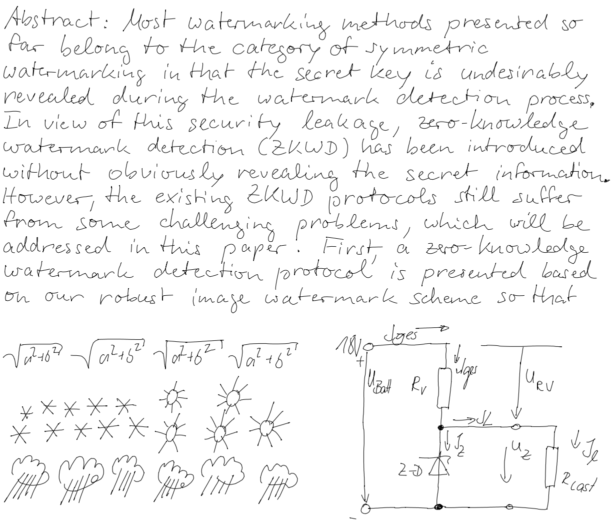
Abstract: Most watermarking methods presented so
far belong to the category of symmetric
watermarking in that the secret key is undesirably
revealed during the watermark detection process.
In view of this security leakage, zero-knowledge
watermark detection (ZKWD) has been introduced
without obviously revealing the secret information.
However, the existing ZKWD protocols still suffer
from some challenging problems, which will be
addressed in this paper. First a zero-knowledge
watermark detection protocol is presented based
on our robust image watermark scheme so that

Figure 3: *Excerpt of a handwritten text and a collection of hand-drawn symbols and a sketch.*

From these 59 documents an amount of 271 query words, phrases and symbols and their respective matches (ground truth), altogether 332 occurances were selected and tagged manually. Using these query words, for a set of given parameter setting (see section 4) the number of correct matches, mismatches and missed instances of the search queries were measured. Because of the user dependant nature of the system, the searching only involves those documents of the actual person instead of all documents. Using the mentioned measures *number of correct matches*, *mismatches* and *missing matches* we calculate the common retrieval measures *precision*, *recall rate* [24] and F_1*-Measure* [25]:

$$\begin{aligned} precision &= \frac{matches}{matches + mismatches} \\ recall &= \frac{matches}{matches + missings} \\ F_1 &= \frac{2 \times precision \times recall}{precision + recall} \end{aligned} \tag{4}$$

The evaluation tests were performed using a Pentium M 1.6 GHz and 512 MB RAM with MS Windows XP SP2 Tablet Edition running JDK 1.5.0 and a MySQL 5.0 database.

5.2. Test Goals and Test Results

Our main goal is to evaluate the retrieval performance of our new approach. We measure the performance by using the widely used scores *precision*, *recall rate* and F_1*-Measure* for a number of parameter settings (i.e. grid width for grid based coding, arc length and degree of angle quantization for direction and curvature based coding and degree of angle quantization for slant based coding) for all four feature types (see section 4). Our second goal is to estimate the average time complexity per document for all feature types and different parameters.

Figure 4 and table 1 show the retrieval results in form of graphs and in numerical form, respectively. As can be seen, the retrieval performance is the best when using Freeman grid features. In our best case (using a small grid with size being 6 units) the F_1-Measure for this feature class is 0.815. The smaller the grid size, the better the recognition performance. Surprisingly the direction based coding without a grid shows a worse performance than the grid based one. In our best case (using an arc length of 6 units between re-sampled points and 12 directions) it resulted in a F_1-Measure of 0.71. The F_1-Measure for curvature features using direction difference is with 0.323 very low (arc length of 20 units and 16 directions). A similar value could be achieved for slant based features; here the F_1-Measure is 0.312. Analyzing the average searching time show, that the slant based features are by far the best with only about 15 ms per document in average. The reason for that is the $O(m \times n)$ time complexity of the approximate string searching algorithm: the smaller the input strings, the faster is the calculation of matrix D (see section 3). Since for the slant based approach only a limited number of features are taken, namely the directions between consecutive Y-minima/-maxima, the resulting feature strings are relatively short, compared to the three remaining approaches, where points from the complete pen tip trajectory are used to derive features.

In addition to the results of our searching system, figure 4 and table 1 also show the results of similar approaches from the literature. In the best case, the retrieval performance of our system is better than the values of Lopresti et al. [12]. However the system of Jain et al. [11], reported to yield 93.2% precision and 90% recall rate, achived a higher recognition performance than our system.

Because of different databases and slightly different goals, all these results are not well comparable. Both Lopresti et al. and Jain et al. perform word segmentation, while we tried to avoid such preprocessing. Furthermore in [11], the authors used different individual thresholds, which enhance the performance, while we used in our first evaluation a common threshold for all

[1] The public part of our database is available under http://wwwiti.cs.uni-magdeburg.de/~sschimke/hwdb/

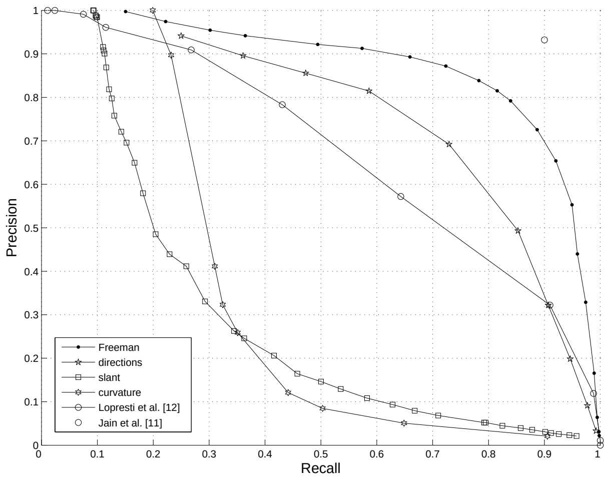

Figure 4: *ROC (receiver operation characteristic) curve, showing the precision and recall rates for four feature classes (see section 4) using their best parameter setting. Additionally results from the literature are plotted [11][12].*

documents and all persons. Automatic optimization of individual thresholds and parameter settings could potentially raise the performance results of our system.

6. CONCLUSION AND FUTURE WORK

In our paper we present a new approach for searching within handwritten documents without textual recognition. We utilize the approximate string searching technique, known from the domain of bioinformatics, where it is typically used for finding pieces of gene sequences. We discuss four different feature types for converting handwriting signals into strings, to be able to use the string searching algorithm. While evaluating our system with an own database of handwritten documents we achieved results of precision and recall rate of each 81.5%. So we were able to show the general capability of our new system.

For the future work it is planned to evaluate the system with a larger amount of documents to get more significant results. Furthermore, from the algorithmic point of view, we plan to test different fusion strategies, to combine the results of our different features, to potentially achive a higher recognition performance.

7. ACKNOWLEDGEMENTS

This work has been supported by the EU Network of Excellence SIMILAR (FP6-507609). The contents of this publication are the sole responsibility of the authors and can in no way be taken to reflect the views of the European Union.

We want to thank all people, who provided us with valuable insights or supported us with our database, especially the advisors, colleagues and students of the Advanced Multimedia and Security Lab (AMSL) of the University of Magdeburg.

8. REFERENCES

Feature Class	Prec.	Rec.	F_1	$\frac{ms}{doc}$
Freeman (size = 6)	**81.5%**	**81.5%**	**0.815**	**1555**
Freeman (size = 8)	82.3%	78.9%	0.806	1607
Freeman (size = 10)	78.3%	78.8%	0.785	572
Freeman (size = 12)	77.1%	73.9%	0.755	451
Freeman (size = 16)	73.8%	71.6%	0.727	284
Direction (length = 6)	**69.2%**	**72.9%**	**0.710**	**1933**
Direction (len. = 10)	64.6%	71.1%	0.677	739
Direction (len. = 12)	65.2%	67.2%	0.662	537
Curvature (len. = 15)	31.2%	30.0%	0.306	287
Curvature (len. = 20)	**32.3%**	**32.4%**	**0.323**	**176**
Curvature (len. = 25)	30.8%	33.3%	0.320	91
Slant (directions = 3)	18.3%	15.3%	0,167	15
Slant (direct. = 4)	24.5%	24.9%	0.247	15
Slant (direct. = 7)	26.5%	24.1%	0.252	15
Slant (direct. = 11)	**33.1%**	**29.3%**	**0.312**	**14**
Slant (direct. = 14)	27.6%	29.1%	0.283	13
Lopresti et al. [12]	64.3%	57.2%	0.605	n/a
Jain et al. [11]	93.2%	90.0%	0.916	n/a

Table 1: *Precision, recall rate, F_1-measure and average time per document for four classes of features using different parameter settings*

[1] S. Elrod, R. Bruce, R. Gold, D. Goldberg, F. Halasz, W. Janssen, D. Lee, K. McCall, E. Pedersen, K. Pier, J. Tang, and B. Welch, “Liveboard: A Large Interactive Display Supporting Group Meetings, Presentations and Remote Collaborations”, in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 599–607, 1992. 49

[2] Virtual Ink Corp., “mimio Capture - Specifications”. http://www.mimio.com/products/capture/. 49

[3] IBM Research, “Pen Technologies”. http://www.research.ibm.com/electricInk/. 49

[4] Logitech Inc., “Logitech io Digital Pen”. http://www.logitech.com/. 49, 52

[5] Pegasus Technologies Ltd., “Pegasus - Digital Pens”. http://www.pegatech.com/. 49

[6] R. Plamondon and S. N. Srihari, “On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 63–84, 2000. 49

[7] C. C. Tappert, C. Y. Suen, and T. Wakahara, “The State of the Art in On-Line Handwriting Recognition”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 8, pp. 787–808, 1990. 49

[8] H. Oda, A. Kitadai, M. Onuma, and M. Nakagawa, “A Search Method for On-Line Handwritten Text Employing Writing-Box-Free Handwriting Recognition”, in *Proceedings of the Ninth International Workshop on Frontiers in Handwriting Recognition*, pp. 545–550, 2004. 49

[9] M. P. Perrone, G. F. Russell, and A. Ziq, “Machine learning in a multimedia document retrieval framework”, *IBM Systems Journal*, vol. 41, no. 3, pp. 494–503, 2002. 49

[10] D. Frohlich and R. Hull, “The Usability of Scribble Matching”, in *Proceedings of ACM CHI 96 Conference on Human Factors in Computing Systems*, pp. 189–190, 1996. 50

[11] A. K. Jain and A. M. Namboodiri, “Indexing and Retrieval of On-line Handwritten Documents”, in *International Conference on Document Analysis and Recognition*, pp. 655–659, 2003. 50, 52, 53

[12] D. P. Lopresti and A. Tomkins, “On the Searchability of Electronic Ink”, in *Proceedings of International Workshop on Frontiers in Handwriting Recognition*, pp. 156–165, 1994. 50, 52, 53

[13] V. I. Levenshtein, “Binary codes capable of correcting deletions, insertions and reversals”, *Soviet Physics Doklady*, vol. 10, no. 8, pp. 707–710, 1966. 50

[14] G. Navarro, “A guided tour to approximate string matching”, *ACM Computing Surveys*, vol. 33, no. 1, pp. 31–88, 2001. 50

[15] S. Schimke, C. Vielhauer, and J. Dittmann, “Using Adapted Levenshtein Distance for On-Line Signature Authentication”, in *International Conference on Pattern Recognition*, vol. 2, pp. 931–934, 2004. 50

[16] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, “Basic Local Alignment Search Tool”, *Journal of Molecular Biology*, vol. Volume, no. 215, pp. 403–410, 1990. 50

[17] D. Gusfield, *Algorithms on Strings, Trees, and Sequences*. Cambridge University Press, 1997. 50

[18] P. H. Sellers, “The Theory and Computation of Evolutionary Distances: Pattern Recognition”, *Journal of Algorithms*, vol. 1, pp. 359–373, Dec. 1980. 50

[19] H. Freeman, “Computer Processing of Line-Drawing Images”, *ACM Computing Surveys*, vol. 6, no. 1, pp. 57–97, 1974. 51

[20] A. Coyette, S. Schimke, J. Vanderdonckt, and C. Vielhauer, “Trainable Sketch Recognizer for Graphical User Interface Design”, in *Proceedings of INTRERACT 2007 – International Conference on Human-Computer Interaction*, Sept. 2007. 51

[21] C. de Boor, *A Practical Guide to Splines*. New York: Springer Verlag, 1978. 51

[22] L. Denoue and P. Chiu, “Ink Completion”, in *Graphics Interface 2005 – Posters and Demos*, 2005. 51

[23] I. Guyon, L. Schomaker, R. Plamondon, M. Liberman, , and S. Janet, “UNIPEN project of on-line data exchange and benchmarks”, in *International Conference on Pattern Recognition*, pp. 29–33, 1994. 52

[24] J. Makhoul, F. Kubala, R. Schwartz, and R. Weischedel, “Performance measures for information extraction”, in *Proceedings of DARPA Broadcast News Workshop*, pp. 249–252, 1999. 52

[25] Y. Yang and X. Liu, “A re-examination of text categorization methods”, in *Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval*, pp. 42–49, 1999. 52