Using Adapted Levenshtein Distance for On-Line Signature Authentication

Sascha Schimke, Claus Vielhauer, Jana Dittmann
Otto-von-Guericke University Magdeburg, D-39106 Magdeburg, Germany
{sascha.schimke, claus.vielhauer, jana.dittmann}@iti.cs.uni-magdeburg.de

Abstract

In this paper a new method for on-line signature authentication will be presented, which is based on a event-string modelling of features derived from pen-position and pressure signals of digitizer tablets. A distance measure well known from textual pattern recognition, the Levenshtein Distance, is used for comparison of signatures and classification is carried out applying a nearest neighbor classifier. Results from a test set of 1376 signatures from 41 persons are presented, which have been conducted for four different feature sets. The results are rather encouraging, with correct identification rates of 96% at zero false classifications.

1. Introduction

One of the key problems in biometrics is the intra-class variability. Here, like in other fuzzy processes in the analogous world, we are confronted with the fact that nearly no two structures are completely the same although they are of identical origins. To address this problem, algorithms for *inexact matching* may be applied in order to get a value of the similarity of two structures. Such structures can e.g. be text strings, biological patterns like DNA sequences or features of some biometric modality.

One possible approach to solve this inexact matching problem known from textual pattern recognition is the *edit distance*, which defines a way to measure the difference or distance between two strings by transforming one string into the other, applying a series of edit operations on individual characters. For every character in the first string, a sequence of the operations *insert*, *delete* and *replace* can be performed to transform that string into the other. The edit distance between two strings is then defined as is the minimum number of edit operations needed to transform the first string into the second. Since the scientist V. I. Levenshtein introduced that distance [3], it is commonly known as *Levenshtein distance*.

Instead of just counting the number of operations, scores can be used. For example, it is possible to weight the cost for deletion of a character higher than replacing it with another character. The scoring either can be used with respect to operations or, in addition to that, with respect to individual character values to which operations are performed. Replacing of a character `i` by `j`, for example, could have a smaller score than replacing `s` by `t`. To model this operation weighting, vectors for operations *insert* and *delete* and a matrix for operation *replace* have been introduced in the area of bioinformatics. Here, PAM and BLOSUM are examples for such scoring matrices in bioinformatics[1], where the alphabet in DNA sequences comparison consists of amino acids instead of text characters.

The edit distance D between two strings S_1 and S_2 can be calculated by a dynamic programming algorithm and can be formally described by the following recursion:

$$\left.\begin{aligned} D(i,j) &:= \min[D(i-1,j)+w_d, \\ &\qquad D(i,j-1)+w_i, \\ &\qquad D(i-1,j-1)+w_r] \\ D(i,0) &:= D(i-1,0)+w_d \\ D(0,j) &:= D(0,j-1)+w_i \end{aligned}\right\} \forall i,j>0$$
$$D(0,0) := 0$$

where i and j are lengths of strings S_1 and S_2 respectively and w_i, w_d, w_r are the scores of operations *insert*, *delete* and *replace*. The resulting value D thus becomes smaller with increasing similarity of the two strings S_1 and S_2. For our initial evaluation of the Levenshtein Distance in respect to online signature authentication, we have used an simplified weighting. The weight score has been set to 1 for each operation, except for replacing of a character by itself, which is scored 0.

2 Adaptation to Online Signatures

In order to use Levenshtein distance for on-line signature authentication – in other words: for comparing of two sets of on-line signature sample data – it is necessary to represent signature data as strings or sequences of codes. In our case the raw data of writing signal is a sequence of discrete pen information tuples – pen tip position on writing pad and pressure. From that data a pen movement can be interpolated and some additional signals can be derivated, e.g. velocity and acceleration. Fig. 1 shows a 2-dimensional writing sample and fig. 2 shows raw and some velocity signals of the sample.

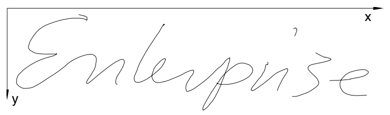

Figure 1. Writing Sample

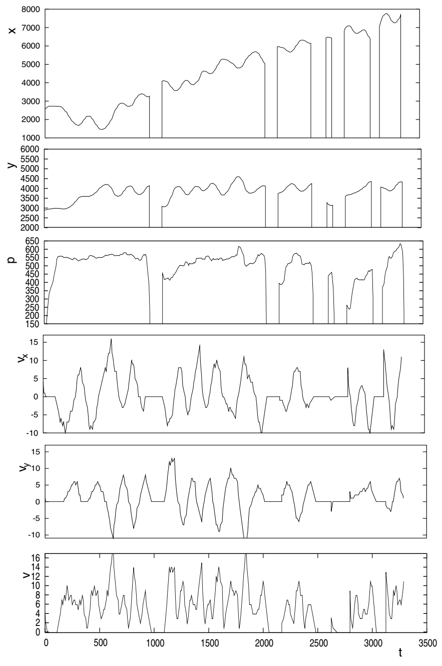

Figure 2. Raw and derivative signals

The basic idea of our approach is based on *events* and can be described as follows: The function curves of movement – e.g. $x(t)$, $y(t)$ and $p(t)$ – have a series of local extrema (minima, maxima). Every occurrence of extrema can be interpreted as an *event*. In addition to extrema there are other kinds of events. One of them is the gap between two writing segments. (Note we define a segment as the signal at times between a pen-down and the subsequently following pen-up.) Dependent on hardware sampling rate and writing velocity some signatures have segments with too few position data to gain reasonable velocity information. In some cases however, if short segments have more than one piece of position data, we additionally can get the stroke direction (e.g. from left to right). In order to achieve a string-like representation of the online handwriting samples, we analyze the sample signals, extract the feature events ε listed in the following table and arrange them in temporal order of their occurrences, leading to *event strings*; examples for such event strings are presented in fig. 3

By coding the event features with string codes as denoted in column "S-Code" in tab. 1, we can yield string representations as illustrated in fig. 3. In this coding, we model local

E-Code	S-Code	Description
$\varepsilon_1 \dots \varepsilon_6$	$xXyYpP$	x-min, x-max, y-min, y-max, p-min, p-max
$\varepsilon_7 \dots \varepsilon_{12}$	$v_xV_xv_yV_yvV$	v_x-min, v_x-max, v_y-min, v_y-max, v-min, v-max
$\varepsilon_{13} \dots \varepsilon_{14}$	gp	gap, point
$\varepsilon_{15} \dots \varepsilon_{22}$		short events; directions: $\uparrow$, $\nearrow$, $\rightarrow$, $\searrow$, $\downarrow$, $\swarrow$, $\leftarrow$, $\nwarrow$

Table 1. Possible event types

maxima with capital letters – X, Y, P (pressure), V, V_x, V_y – and local minima with lower case letters – x, y, p, v, v_x, x_y – and the gap event with g. For our further, more general discussion, we will synonymously make use of enumerated annotation of the event types ($\varepsilon_1 \dots \varepsilon_{22}$) as can be seen in column "E-Code" of tab. 1.

Fou our example in fig. 1 there are six segments, each of which is followed by a gap.

3rd segment: $(Xp)v_xV_yxYPyVpv(XV_xv_y)PpPpPg$
4th segment: $XPyg$
5th segment: $pXPxv_ypPpv_yXvPpPg$
6th segment: $(XP)v_ypyxPYg$

Figure 3. Events string for segments 3 to 6

A problem arising from this coding is, that due to the sampling at a discrete sampling rate, events may occur pseudo-simultaneously. This effect is modelled by multiple symbols in parenthesis, denoting events occuring synchronously. With these event sequences, we have adequate input data for the determination of edit distances of online writing samples e.g. signatures, if we find a way to handle synchronous events. One possible way to handle simultaneous events is, to treat them as just one *combination event*, requiring the definition of scores for edit operations on those combination events.

Let $E = \{\varepsilon_1 \dots \varepsilon_n\}$ be the set of n possible events and be $\varepsilon^{com} \in E^*$ a combination event consisting of two or more synchronous events out of E. In order to handle the simultaneity of event occurence, we define the scores of operations of insertion and deletion of ε^{com} as the sum of the cost of the respective individual operations:

$$w_{ins,\varepsilon^{com}} := \sum_{\varepsilon_i \in \varepsilon^{com}} w_{ins,\varepsilon_i} \qquad w_{del,\varepsilon^{com}} := \sum_{\varepsilon_i \in \varepsilon^{com}} w_{del,\varepsilon_i}$$

Although, depending on the degrees of freedom, the weight determination for replacement of combination events can be modelled in very complex manners, in our first evaluation, we define simplified replacement weights as the minimum operation score of a member event, as per the folling equation.

$$w_{rep,\varepsilon_1^{com},\varepsilon_2^{com}} := \min[w_{rep,\varepsilon_i,\varepsilon_j}] \quad i,j \in \varepsilon_1^{com}, \varepsilon_2^{com}$$

Due to its nature, the average edit distance is linearly dependent to the string lengths of both strings. For example, two different signatures of a person with a long name have a larger distance than two different signatures of a person with a shorter name. Therfore, for application in a biometric authentication system, normalization is required. In our approach, normalization is performed with respect to the distance of the event string length. The original Levenshtein distance will be adapted by introducing a divisor $i + j$ (sum of string lengths). To make allowance of the problem that now someone could attack the system by entering a very large writing sample with a very long event string, we also include a compensation factor R to the original distance. R is the relation of the larger length to the smaller one. The resulting adapted Levenshtein distance $\mathcal{D}$ is defined as

$$\mathcal{D}(i,j) := \frac{D(i,j)R(i,j)}{i+j} \qquad R(i,j) := \begin{cases} \frac{i}{j} & \text{if } i > j \\ \frac{j}{i} & \text{else} \end{cases}$$

The adopted Levenshtein distance has been integrated in an evaluation system for online handwriting authentication, as outlined in the following figure. Test results of our initial evaluation will be presented in the following section and have been obtained with the weight parameter setting fixed to the values described in the first section.

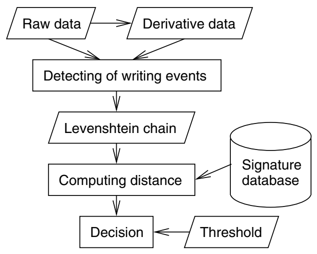

Figure 4. Model of Authentication Process

3. Test Results

The test environment consists of verification and identification tests. Basis of the tests is a database of 1376 writing samples from 41 persons. Writing samples are structured based on a classification presented in [6] into *authentic*, *random forgery*, *blind forgery*, *low-force forgery* and *brute-force forgery* samples. The samples were taken from three different digitizer tablets (Wacom USB, Cintiq 15, Intuos 2) and for the initial tests, only signature semantics have been taken into account.

The organization of our test set can be described as follows: Let n be the number of authentic samples of a person. We choose randomly m out of them build the reference sample set, whereas the remaining are verification or identification trials. $m := \lceil n/2 \rceil$, but if $m > 10$ then $m := 10$. In total, we yield 157 enrollment samples, 748 authentic, 64 blind forged, 217 low-force forged and 190 brute-force forged samples for our database.

In our verification tests, each acquired sample from the test set gets compared to all reference samples of a person. The verification is positive if the smallest distance of the acquired sample to the reference samples is smaller than a threshold (nearest neighbor classifier). Fig. 5 shows the FRR and FAR (*false rejection/acceptance rate*) results as function of the threshold for our verification tests. To examine the influence of different event types, the tests have been conducted separatly for four different event type sets. The first set (a) consists of all available event types, whereas the second set (b) consists of short and position events. The event types in third set (c) are short and velocity events. In fourth set (d) there are only position events. Fig. 5 shows resuls using event type sets (a) to (d).

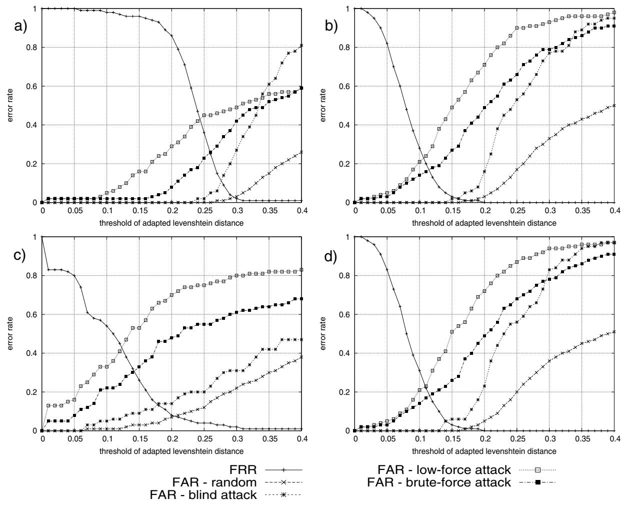

Figure 5. Verification test results

Using the same test environment, identification tests have been performed. The test has been organized as follows: Having an acquired writing sample, the enrollment sample with the least distance has been determined. If both samples, enrollment and acquired, are from the same originator we classify the identification as correct. Also, identification is classified correctly, if a forgery against a person is identified as that person. If another person than the originator of the acquired sample is identified we define this occurance as an incorrect identification. The same applies, if more than one persons have enrollment samples with the same least distance. Table 2 shows the identification rates for authentic identification and for forgery trials. Identification rates are shown for four different event type sets as described before.

	auth	blind	low	brute
correct (a)	96.4%	20.3%	40.1%	22.1%
correct (b)	98.8%	21.9%	65.9%	41.6%
correct (c)	53.5%	7.8%	36.4%	22.6%
correct (d)	98.4%	21.9%	61.8%	37.9%

Table 2. Correct identification rates

By introducing a threshold for maximal acceptable distance for a classification, we get another category for results additionally to correct and incorrect identifications – the category of non-identification. Nobody is identified, if the least distance is greater than the threshold. Fig. 6 shows the correct and incorrect identification rates for authentic and forged writing samples with respect to such a maximum identification distance threshold.

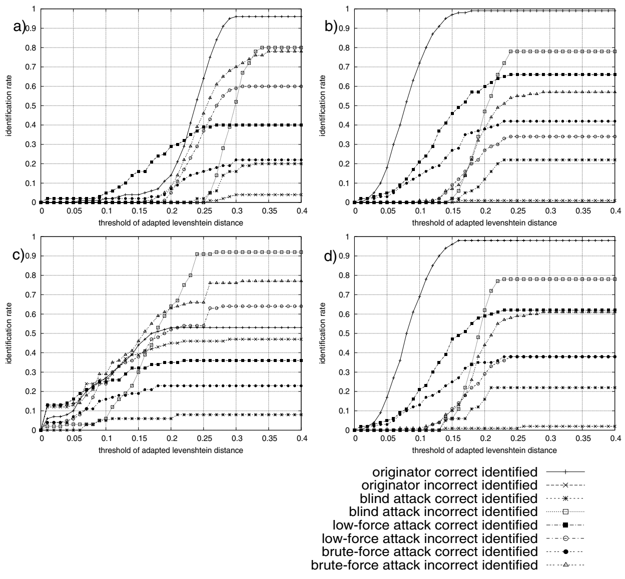

Figure 6. Identification test results

It can be seen, that the EER (*equal error rate*) for verification tests is rather low for event type sets b and d (only using position feature events) as compared to the other two sets. At first glance, this appearance appears inconsistent with observations published earlier, like [4], where pressure and velocity features have been identified as significant for on-line signature authentication. In our scenario however, one explanation for negative influence of pressure and velocity based feature events could be high-frequency noise in those signals (see fig. 2). In summary, it can be stated that we have achieved a correct identification rate of $>96\%$ and incorrect rate of 0% at a operating point for a threshold of 0.15, which is a quite encouraging result for our initial evaluation.

4 Conclusion and Future Work

We have presented a new approach for online signature authentication, which is based on event-string modelling, determination of the Levenshtein distance and a nearest-neighbor classifier. Tests performed on a reasonably large database have shown good results for our method, both for verification and identification problems. By choosing adequate event feature sets, equal-error-rates (EER) below 2% could be achieved in verification tests for non-intensional forgeries and in our identification evaluation we could determine an operating point leading to a correct identification rate of 96% at zero-false identifications. Since our approach is invariant against time shift between two compared writing samples, it eliminiates the necessity for a time warp process, which is required in other systems which use functional interpretation of writing signals. Therefore, the method can be implemented in a computationally efficient way.

However, the test results have shown an increase of the False-Acceptances of roughly a magnitude, once the system is exposed to intentional forgeries. This obervation confirms test results of earlier work in the area of evaluation of handwriting based biometrics [6].

Our future work will include additional tests of this method with writing samples of additional other semantic classes than signatures, as suggested in [5].

Furthermore, optimization of weight vectors and weight matrix and the composition of the weights for combination events, as well as elimination of the noise factor in pressure and second-order based features (e.g. velocities), using a frequency filter should allow improvements to the accuracy of our approach.

Acknowledgment

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

References

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

[2] F. Leclerc and R. Plamondon. Automatic signature verification: The state of the art – 1989-1993. *Internation Journal of Pattern Recognition and Artificial Intelligence*, 8(3):643–660, 1994.

[3] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics*, 10:707–710, 1966.

[4] R. Plamondon and G. Lorette. Automatic signature verification and writer identification – the state of the art. *Pattern Recognition*, 22(2):107–131, 1989.

[5] C. Vielhauer. Handschriftliche Authentifikation für digitale Wasserzeichenverfahren. In M. Schumacher and R. Steinmetz, editors, *Sicherheit in Netzen und Medienströmen*, pages 134–148. Springer Verlag, September 2000.

[6] F. Zöbisch and C. Vielhauer. A test tool to support brut-force online and offline signature forgery tests on mobile devices. *Proceedings of IEEE ICME 2003*, pages 60–64, 2003.