2011 International Conference on Document Analysis and Recognition

On-line Signature Verification Using Segment-to-segment Graph Matching

Kaiyue Wang, Yunhong Wang and Zhaoxiang Zhang
School of Computer Science and Engineering, Beihang University
Beijing, China
wangkaiyue@ss.buaa.edu.cn, yhwang@buaa.edu.cn, zxzhang@buaa.edu.cn

***Abstract*—This paper proposes a novel approach of on-line signature verification. Firstly, on-line signatures are partitioned into a series of segments, which are then represented by graphs. Four segmentation methods are taken into account. Secondly, graph matching techniques are adopted to compute edit distance between corresponding graphs, which measures the similarity of them. Finally, having been able to compare two signatures, limited genuine signatures are used to train user-dependent classifiers for each user. Experiments are conducted to validate the effectiveness of the proposed method and promising results are achieved.**

***Keywords*-on-line signature, signature verification, signature segmentation, graph matching, biometrics**

I. Introduction

Among all means of identity verification nowadays, signature verification is one of the most accepted methods. The formation influenced by physical condition, family environment, education and many other factors, signatures differ from person to person. Thus as such a unique behavioral biometric feature which is difficult to imitate, it is suitable for identity verification. This paper concerns on-line signatures, which are collected by devices such as digital tablets or electronic pens, instead of traditional signatures signed on paper. Each sample of on-line signature is composed of a series of points, each of which contains features like pen position, pen-tip pressure, moving direction, and speed. Since these features are accurately recorded by acquisition devices, on-line signatures are assumed to be more reliable for identity verification than traditional signatures.

Quite an amount of pattern recognition methods have already been applied to on-line signature verification, such as Dynamic Time Warping (DTW), Hidden Markov Model (HMM), and Support Vector Machine (SVM), etc [1]. Some methods involving graph representation of signature and graph matching had been applied to off-line signature verification, while there had been seldom researches on on-line signature verification using graph matching as far as we know. In this paper, we propose a novel approach of on-line signature verification using graph representation and segment-to-segment graph matching. Signatures are first partitioned into smaller segments, each of which is represented by a graph, then corresponding segments are compared using graph matching methods.

II. Graph Representation and Matching

A. Graph Representation of On-line Signatures

A graph, composed of nodes and edges, is usually denoted by $G = (V, E)$, where $V = \{v_1, v_2, ..., v_M\}$ is the set of nodes and $E = \{e_{ij} = (v_i, v_j)\} \subseteq V \times V$ the set of edges. For a weighted graph, there exists a weighting function $w(\cdot)$ that gives each node and edge a real nonnegative value. In the case of undirected graph, which is concerned in our work, $w(e_{ij}) = w(e_{ji})$. A n-node graph is usually represented by a $n \times n$ adjacency matrix $A_G = \{a_{ij}\}$, a_{ij} defined as

$$a_{ij} = \begin{cases} w(e_{ij}) = w(v_i, v_j), i \neq j \\ w(v_i), i = j \end{cases} \quad (1)$$

Obviously, A_G is symmetric if the graph is undirected.

Suppose S is an on-line signature or a segment of a signature, containing N sample points. To represent S with graph, firstly, sample points are regarded as nodes, i.e. $S = \{v_1, v_2, ..., v_N\}$. While each sample point may contain a variety of directly extracted features, e.g. pen position and pressure, depending on the data acquisition device, we only take into account three most commonly used features: x-position, y-position and pressure at sample points. As a result, x_i, y_i and p_i respectively denote x-position, y-position and pressure at v_i.

Based on x_i, y_i and p_i, more complicated features are computed and regarded as the weights of nodes or edges. According to the definition of node and edge, weights of nodes should be local features that describe properties of sample points, while weights of edges describe relationship between nodes.

In our work, two sets of features are respectively chosen to generate two types of graph for each segment. Type I involves pressure and angle:

$$A_{G1} = \begin{pmatrix} p(v_1) & a(v_1, v_2) & \cdots & a(v_1, v_N) \\ a(v_2, v_1) & p(v_2) & & a(v_2, v_N) \\ \vdots & & \ddots & \vdots \\ a(v_N, v_1) & a(v_N, v_2) & \cdots & p(v_N) \end{pmatrix} \quad (2)$$

where $p(v_i) = p_i$ is the pressure at sample point v_i and $a(v_i, v_j) = arctan[(y_i - y_j)/(x_i - x_j)]$ is the angle between vector $\overrightarrow{v_i v_j}$ and x-axis. Type II involves curvature and

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.165

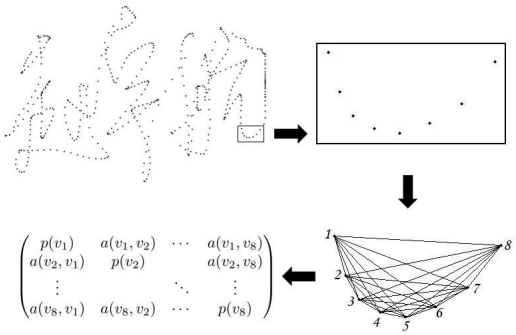

Figure 1. Framework of graph representation

distance:

$$A_{G2} = \begin{pmatrix} c(v_1) & d(v_1,v_2) & \cdots & d(v_1,v_N) \\ d(v_2,v_1) & c(v_2) & & d(v_2,v_N) \\ \vdots & & \ddots & \vdots \\ d(v_N,v_1) & d(v_N,v_2) & \cdots & c(v_N) \end{pmatrix} \quad (3)$$

where $c(v_i) = |x'_i y''_i - x''_i y'_i|/(x'^2_i + y'^2_i)^{3/2}$ is the curvature at sample point v_i and $d(v_i, v_j) = [(y_i - y_j)^2 + (x_i - x_j)^2]^{1/2}$ is the distance between v_i and v_j. Fig.1 shows the process of representing a segment of a signature in the form of A_{G1}.

So far, a signature can be partitioned into several segments, each of which can be represented by two graphs in the form of A_{G1} and A_{G2}. The comparison between two signatures is now converted into computing the distance between graphs, which will be introduced below.

B. Graph Matching Techniques

Graph matching methods can be roughly partitioned into two categories: exact matching and inexact matching. Due to the unavoidable variation among signatures signed by the same person, inexact matching turns out to be more suitable for the task of signature verification.

In the graph matching phase, we employ the concept of graph edit distance. In this sense, a graph can be transformed to another one through a finite sequence of graph edit operations, which can be defined differently in different circumstances. Summing up the costs of all edit operations obtains the total cost of the whole transforming process. The minimum total cost of all possible sequences is the graph edit distance between the graphs compared. Two main issues can be drawn from the description of graph edit distance above: definition of cost functions and the search of minimum-cost edit operation sequence.

1) Cost Functions: Edit operations are usually classified into four categories: deleting a node, inserting a node, substituting a node and substituting an edge. However, for simplification, we generalize all these operations into the same type: the change of weights.

To further specify the cost function we adopted, for the features of pressure, curvature and distance, the cost is the absolute difference of weights before and after change, i.e.

$$c(r^s \to r^t) = \left| w(r^s) - w(r^t) \right| \quad (4)$$

where $w(\cdot)$ can be substituted by $p(\cdot)$,$c(\cdot)$ or $d(\cdot)$, r is a node or an edge according to specific weighting function and r^s is the node or edge before change while r^t is after.

As for the feature of angle, the following cost function is used to measure the difference between angles:

$$c(r^s \to r^t) = \begin{cases} |a(r^s) - a(r^t)|, when\, |a(r^s) - a(r^t)| < \pi \\ 2\pi - |a(r^s) - a(r^t)|, otherwise \end{cases} \quad (5)$$

2) Computing Edit Distance: The minimum-cost edit path can be retrieved by a tree search in the space of all possible edit paths. Although the optimal edit path can always be found using this class of algorithms, the computational complexity is exponential. For efficiency, we adopted a sub-optimal algorithm proposed in [2]. The process of graph matching is seen as an assignment problem, i.e. assigning nodes of graph g_1 to nodes of graph g_2, so that the overall edit costs are minimal. One of the popular method to deal with assignment problem is Munkres' algorithm [3].

Suppose graph g_1 has n nodes, graph g_2 has m nodes, $v_i \in g_1$ and $u_j \in g_2$, the cost matrix of nodes is constructed as a $n \times m$ matrix $C = \{c_{ij}\}_{n \times m}$, in which $c_{ij} = c(v_i \to u_j)$. Each entry of C indicates the cost of changing v_i in g_1 to u_j in g_2. Munkres' algorithm finds a set of entries of C, the sum of which is minimum while any two of which are neither in the same row nor the same column.

However, if Munkre's algorithm is directly applied to C, no edge information will be taken into account. To exploit edge information, c_{ij} is redefined as

$$c_{ij} = c(v_i \to v_j) + min\{\sum c(e_{vi} \to e_{uj})\} \quad (6)$$

The last term of (6) implies the minimum-cost edge assignment between edges connected to v_i in g_1 and edges connected to u_j in g_2. The Munkre's algorithm is recursively computed to obtain $min\{\sum c(e_{vi} \to e_{uj})\}$. Details of the algorithm can be found in [2]. We now address the distance between graph g_1 and graph g_2 as $d(g_1, g_2)$.

III. Verification with Graph Representation

A. Preprocessing

As mentioned before, an on-line signature sample is composed of a sequence of sample points. Since the numbers of sample points of different signatures may vary drastically, we resampled each signature down to 100 sample points, the number of which is neither too small to retain enough information, nor too large to provide efficiency; also, signatures are normalized to avoid the impact of the variation of position and scale. Segmenting points of each signature are found respectively using methods that will be described in Section IV-A and two types of graphs of each segment are formed, therefore each signature is represented by graphs twice the number of segments.

B. Training

Since skilled forgeries are difficult to obtain in practice, we use only 5 genuine signatures of each user to train user-dependent classifiers, the number of which is also used in many related researches. Given the training set of 5 genuine signatures of a user, we denote them as s_i, $i = 1, 2, 3, 4, 5$.

The number of segments of different signatures may vary using certain segmentation methods, so when a comparison of two signatures occurs, one of them should be regarded as the reference signature so that the other one can be aligned to establish correspondence between segments, process of which is described in Section IV-A. Function $D(\cdot, \cdot)$ indicates the distance between two signatures with the first one the reference signature. Specifically, it is the sum of edit distances between each pair of corresponding segments of two signature:

$$D(s_i, s_j) = \sum_{k=1}^{m} d(g_i^k, g_j^k) \tag{7}$$

In this equation, m is the number of graphs of s_i, and g_i^k means the kth graphs of signature s_i while g_j^k is the correspondence of g_i^k in signature s_j. Each training signature regarded as the reference signature once, distances between reference and other four signatures are computed, and the one with the smallest average distance is selected as the template:

$$t = \underset{i}{argmin}\{(\sum_{j} D(s_i, s_j))/5\} \tag{8}$$

So s_t is chosen as the template.

In addition, for each graph, the maximum, average and minimum distance from the template, as the reference signature, to the other 4 signatures are computed:

$$\begin{aligned} max^k &= \underset{g_i^k \neq s_t^k}{max}\{d(s_t^k, g_i^k)\} \\ avg^k &= (\textstyle\sum_{g_i^k \neq s_t^k} d(s_t^k, g_i^k))/4 \\ min^k &= \underset{g_i^k \neq s_t^k}{min}\{d(s_t^k, g_i^k)\} \end{aligned} \tag{9}$$

C. Verification

Given a test signature s, we adopt the following scoring strategy to verify its authenticity. First, for each of the graphs, compute the distance between the template and the corresponding graph of the test signature:

$$d^k = d(s_t^k, g_s^k) \tag{10}$$

Then, it is compare to max^k, avg^k and min^k and scored:

$$score(g_s^k) = \begin{cases} t_1, d^k > max^k \\ t_2, avg^k < d^k \leq max^k \\ t_3, min^k < d^k \leq avg^k \\ t_4, d^k \leq min^k \end{cases} \tag{11}$$

Having obtained scores for all graphs, sum them up to get the final score:

$$score(s) = \sum_k score(g_s^k) \tag{12}$$

The larger the score is, the more likely it is genuine. Thus, given a certain threshold, if the score is lager than it, the signature is assumed to be genuine; otherwise, forgery. As for $\{t_1, t_2, t_3, t_4\}$, although it turns out that $t_1 = 0, t_2 = 1, t_3 = 2, t_4 = 3$ might be a choice, our preliminary experiments show that with a punishment and bonus respectively given to the cases of $d^k > max^k$ and $d^k \leq min^k$, the performance is slightly better. So $t_1 = -1, t_2 = 1, t_3 = 2, t_4 = 4$ would be used in the following experiments.

IV. Experiments

A. Segmenting Signatures

To reduce computational complexity, we introduce the process of signature segmentation, so that the matching is carried out between smaller parts of signatures, rather than between graphs at the scale of the whole signatures. Meanwhile, this avoids some unnecessary comparisons, for example, the comparison between the first points of one signature and the last ones of another, for they almost definitely do not match.

There is a variety of techniques for on-line signature segmentation, for example, by pen-tip pressure, by perceptually important points, and by local extrema [1]. We adopt three of the representative methods of segmentation.

1) Method 1: Equally Segmenting: This is quite intuitive. Signatures are equally partitioned into smaller parts, which all contain the same number of sample points.

2) Method 2: Segmenting by Strokes: Although it is simple to directly use strokes to segment signatures, for some signatures which contain very few strokes, it cannot divide them into parts small enough to fit our graph matching approach. So a signature is first segmented at pen-down points, and if the number of segments is no larger than a preset parameter r, the longest segment is equally partitioned into two parts. Repeat this check until the number of segments is satisfying.

3) Method 3 and 4:Segmenting by Local Extrema: In our work, since verification phase is largely dependent on graph matching, segmentation is merely a part of preprocessing and is not determinant on verification result. Therefore, we only use local maximum and local minimum to partition signatures, which is satisfying serving our purpose. We denote the method using maxima and minima of y-coordinates as Method 3, while that of x-coordinates as Method 4.

Take Method 3 for example, suppose (x_i, y_i) denotes the coordinate of the i-th sample point of a signature:

1. It is considered as a local maximum, if $y_i \geq y_{i \pm j}$ holds for all j, in which $j = 1, 2, ..., 5$ while here 5 indicates the

Figure 2. Examples of Different Segmentation Methods

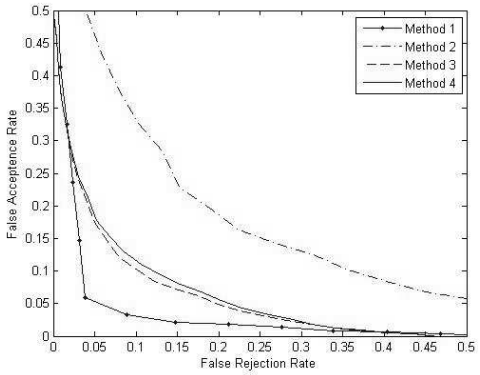

Figure 3. ROC Curves of Different Segmentation Method

range of points which a maximum dominates. Definition of local minimum is likewise. Both maximum and minimum are regarded as extremum candidates.

2. To avoid the impact from cases of 'platform', i.e. where y-coordinates of consecutive points are equal, and flux caused by instability of signing process, if an extremum candidate which satisfies the conditions above is no more than 5 points after a determined extremum, it is ruled out. Otherwise, it is accepted as an extremum.

The procedure of Method 4 is likewise. However, in this way, some of the actual extrema may be ignored, but this does not impair the successive procedure, for the purpose of segmentation in our work is trying to divide signatures into smaller parts under a uniform principle, rather than actually find all extrema.

Figure 2 shows examples for each of the methods described above, with the top-left sub-figure Method 1, the top-right Method 2 and the lower row Method 3 and 4. Black dots indicate the segmenting points, some of which may not be on the strokes of signatures because sample points where the pen is up are also included, but not displayed.

4) Alignment of Segments: There have been quite a lot of articles trying to address this problem, most of which involves merging and dividing segments, or in other words, insertion and deletion of segmenting points. For Method 2 to 4, every comparison is carried out between a reference signature and a signature to be compared to the reference, during the process of which the number of segments of the reference does not change while that of the other signature is adjusted to match the reference, by insertion or deletion of points.

Since using Method 1, the measurement of distance is symmetric, i.e. $D(s_i, s_j) = D(s_j, s_i)$, during the verification phase for each segment, we substitute the distance between the test signature and the template with the average distance between the test signature and all training signatures. Specifically, (10) is substituted by $d^k = \sum_{i=1}^{5} d(g_i^k, g_s^k)/5$ In this way, we take advantage of the symmetry of Method 1 to better measure the distance between the test signature and the training set.

Table I
PERFORMANCES OF DIFFERENT SEGMENTATION METHODS

	Average EER	Min EER	Max EER	σ_{EER}
Method 1	5.80%	0.00%	40.00%	11.14%
Method 2	19.33%	3.33%	50.00%	11.75%
Method 3	10.18%	0.00%	43.33%	9.95%
Method 4	10.84%	0.00%	46.66%	10.20%

B. Comparison among Segmentation Methods

All four segmentation methods introduced in Section IV-A are respectively implemented on the visual subcorpus of SUSIG on-line signature database [4], which contains 20 genuine signatures and 10 skilled forgeries of 94 users. Five genuine signatures are randomly selected to establish the training set. The rest of the genuine signatures and all skilled forgeries are used as the test signatures for each user. We adopt equal error rate (EER) as the main criteria of performance. Results are shown in Figure 3 and Table I.

Intuitively, Method 2 to 4 should present better results than Method 1 since they exploit characteristics of each signature rather than treat them equally. However, the truth is just the opposite. Performance of Method 2 is bad, probably because the number of strokes of signatures from the same person vary so drastically sometimes, that the alignment of segments is inaccurate. As for Method 3 and 4, the problem maybe the asymmetry of distance between two signatures, since when the numbers of extrema are not the same, $D(s_i, s_j) \neq D(s_j, s_i)$. Even though we restricted for every comparison that there should be a reference, this asymmetry impairs discrimination of signatures. As a result, the performance of Method 3 and 4 is not ideal.

Figure 4 displays the distribution of segments of Method 1 over the the whole testing set, showing proportions of segments fall into each category of $\{t_1, t_2, t_3, t_4\}$. Segments of genuine signatures concentrate in t_2 and t_3 while majority

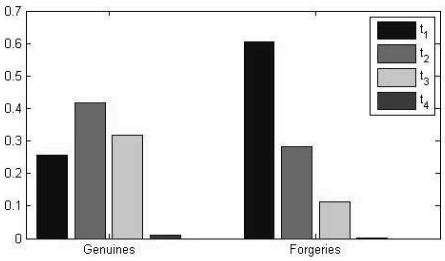

Figure 4. Distribution of Segments

Table II
COMPARISON WITH OTHER METHODS ON SUSIG

	Single System	Fusion with DTW
Method 1	5.80%	2.46%
FD [5]	6.20%	3.03%

Table III
COMPARISON WITH OTHER METHODS ON SVC2004

Methods	Skilled Forgeries		Random Forgeries	
	EER	σ_{EER}	EER	σ_{EER}
SVM [8]	6.84%	10.18%	1.11%	3.46%
DTW [6]	6.96%	11.76%	3.47%	4.23%
Method 1	7.02%	8.73%	1.57%	4.13%

of segments of forgeries fall in t_1, thus providing discrimination between genuine signatures and forgeries.

C. Comparison with Other Method

Method 1 is compared to the approach proposed in [5], which adopted Fourier Descriptors (FD) to describe signatures and was applied to SUSIG-Visual database. We have also combined our method with a modified implementation of the DTW method in [6], as is done in [5]. Results in Table II show that our method performs better whether combine with DTW or not. However, it is not clear in [5] how the DTW method was modified, so we implemented it according to our understanding, thus the fusion results might not be conclusive.

We have also implemented our method on SVC2004 on-line signature database. This database was designed and established for the First International Signature Verification Competition [7]. It has 1600 signatures in total from 40 users, each has 20 genuine signatures and 20 forged signatures. Since this database provides the information of altitude, which we assume to be more consistent than pressure, so all the pressure information in our method is substituted by altitude. Table III shows the results, comparing to other two methods applied on this database. Adopting the same protocol as in the Competition, the performance of our method is close to the best approaches on this database.

V. CONCLUSION

In this paper, we proposed a new method of on-line signature verification using segment-to-segment graph matching. Four types of segmenting strategies and two types of graph representation were introduced, and with a sub-optimal graph matching algorithm to compute distance between graphs, experiments are conducted. Results show that one of the four segmentation methods gives good result, which competes with the state-of-art methods. Further studies are needed to solve the problems for other segmentation methods, which failed to give expected results.

ACKNOWLEDGEMENT

This work is funded by the National Basic Research Program of China (No. 2010CB327902), the National Natural Science Foundation of China (No. 60873158, No. 61005016, No. 61061130560) and the Fundamental Research Funds for the Central Universities.

REFERENCES

[1] D. Impedovo and G. Pirlo, "Automatic signature verification: the state of the art," *IEEE Trainsactions on System, Man and Cybernetics - Part C: Applications and Reviews*, vol. 38, no. 5, pp. 609–635, 2008.

[2] K. Riesen, M. Neuhaus, and H. Bunke, "Bipartite graph matching for computing the edit distance of graphs," in *Proceedings of 6th International Workshop on Graph Based Representations in Pattern Recognition*. springer, 2007, vol. 4538, pp. 1–12.

[3] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[4] A. Kholmatov and B. Yanikoglu, "SUSIG: An on-line signature database, associated protocols and benchmark results," *Pattern Analysis and Applications*, 2008. [Online]. Available: http://biometrics.sabanciuniv.edu/SUSIG

[5] B. Yanikoglu and A. Kholmatov, "Online signature verification using fourier descriptors," *Eurasip Journal on Advances in Signal Processing*, vol. 2009, pp. 1–14, 2009.

[6] A. Kholmatov and B. Yanikoglu, "Identity authentication using improved online signature verification method," *Pattern Recognition Letters*, vol. 26, no. 15, pp. 2400–2408, 2005.

[7] D. yan Yeung, H. Chang, Y. Xiong, S. George, R. Kashi, T. Matsumoto, and G. Rigoll, "Svc2004: First international signature verification competition," in *In Proceedings of the International Conference on Biometric Authentication (ICBA), Hong Kong*. Springer, 2004, pp. 16–22.

[8] C. Gruber, T. Gruber, and S. Krinninger, "Online signature verification with support vector machines based on lcss kernel functions," *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, vol. 40, pp. 1088–1100, 2010.