Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 324645, 12 pages
http://dx.doi.org/10.1155/2014/324645

Research Article

An Immune Clonal Selection Algorithm for Synthetic Signature Generation

Mofei Song and Zhengxing Sun

State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China

Correspondence should be addressed to Zhengxing Sun; szx@nju.edu.cn

Received 14 March 2014; Accepted 18 May 2014; Published 2 June 2014

Academic Editor: Erik Cuevas

The collection of signature data for system development and evaluation generally requires significant time and effort. To overcome this problem, this paper proposes a detector generation based clonal selection algorithm for synthetic signature set generation. The goal of synthetic signature generation is to improve the performance of signature verification by providing more training samples. Our method uses the clonal selection algorithm to maintain the diversity of the overall set and avoid sparse feature distribution. The algorithm firstly generates detectors with a segmented r-continuous bits matching rule and P-receptor editing strategy to provide a more wider search space. Then the clonal selection algorithm is used to expand and optimize the overall signature set. We demonstrate the effectiveness of our clonal selection algorithm, and the experiments show that adding the synthetic training samples can improve the performance of signature verification.

1. Introduction

Handwriting signature recognition is an effective identity authentication method by using signature data, since every person's signature is different, and especially the dynamic characteristic is difficult to imitate. Recently, lots of signature verification methods have been proposed [1–3], and the main goal is to improve the identification effect by investigating the effective classification feature and algorithm.

An important challenge is that most existing approaches require sufficient signature samples to guarantee the effect. First, the performance evaluation of these systems needs to provide a large number of test samples [4]. More importantly, most of the classifier algorithms' (such as neural networks, hidden Markov model) performance generally depends on the amount of training data, and training a stable and efficient classifier needs providing a sufficient number of samples [5]. Although some commercial signature databases have been established, the sharing and distribution of these data are very difficult due to some legal issues [6]. Besides, the number of signature databases that can be shared is fairly limited. A direct solution is to collect signature samples by oneself. However, the database collection is time consuming and expensive, since users are unwilling to submit their privacy data due to potential security problems. In addition, the boring repeated submission process will affect the quality of signature samples.

To overcome the database collection problem, some synthetic signature generation methods have been presented. By these methods, some synthetic signatures can be automatically created by synthesizing real signatures [4–7]. According to the sample synthesis strategies, existing methods can be divided into three categories: duplicated samples, combination of different samples, and synthetic individuals. The duplicate-based method [7–10] generates a new sample through different transformation, and it is suitable for producing different signatures corresponding to the same person. The combination-based method [11] creates a new sample by combining person's handwritten letters or units from different samples. In the synthetic-based method [6], some kind of a priori knowledge (such as stroke placement distribution, length feature) is used to create a new sample, and this method can create new individuals' signature. In summary, existing methods focus on novel sample deformation technology, which can make the new sample simulate the characteristics of the real data. Meanwhile, the accuracy of signature verification is improved by adding new training data.

In fact, adding synthetic training samples does not always improve the performance of the classifier. On one hand, adding the synthetic samples can increase the diversity of

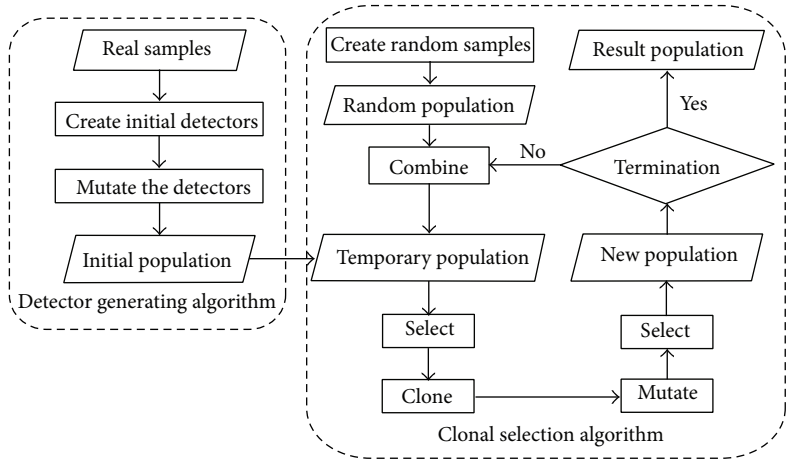

Figure 1: The process of our method.

the training set, which is helpful to optimize the decision parameters in order to improve the identity rate. On the other hand, unnatural deformation will produce large bias away from real samples, which can lead to a deterioration of the accuracy. Besides, the feature distribution of the whole sample set also affects the classifier's performance. Sparse or uneven distribution will make the classifier unstable [12]. Therefore, the diversity and effectiveness of the overall training sample set have an important impact on the performance of signature verification.

Accordingly, the artificial immune system (AIS) is introduced as a means for creating synthetic signatures. Our goal is to use the clonal selection algorithm (CSA) for expanding the signature set from an initial set, composed of a small amount of data. The result sample set can be used as the training data to improve the verification performance. Our method expands the population of signature data in each generation rather than creating a synthetic sample successively. Throughout the process, we focus on the quality of the population more than that of the individuals. By utilizing the advantages of AIS in the self-recognition capabilities and the diversity manipulation mechanisms, the diversity and effectiveness of the overall training sample set can be guaranteed. To investigate whether AIS improves the quality of the synthetic sample set, the duplicated-based method is selected as our synthesis strategy and a new clonal selection algorithm is proposed by introducing a novel detector generation algorithm. The detector generation algorithm uses the segmented r-continuous bits matching rule and P-receptor editing strategy to create the initial population of CSA. The experiment shows the effectiveness of the method.

2. Algorithm Overview

This paper uses the clonal selection algorithm to expand the signature data set. On the basis of ensuring the effectiveness of each new sample, this method focuses on the diversity and effectiveness of the overall set. Standard clonal selection algorithm generates the initial population randomly [13]. Due to lack of the guidance of the input samples, the structures of the antibodies in the initial population and the antigens will be quite dissimilar, which affects the convergence efficiency of the algorithm. Therefore, the detector generation algorithm is introduced into the clonal selection process to present a novel clonal selection algorithm.

The process is shown as in Figure 1. Our process flow can be divided into two steps: detector generation and clonal selection. In detector generation, an iterative enumeration algorithm is firstly used to generate some new samples that can be matched with the input sample by segmented r-continuous bits matching rule, which means that the new sample has several successive stroke sections that are similar to some sections of the input sample. Then P-receptor editing strategy is used to create some new samples that have P different strokes from the input sample. At last, the two sets are combined as the detector set, and the purpose is to get some individuals that have some differences with the input sample in advance. Therefore, the clonal selection algorithm will search the samples in a wider range to avoid losing the opportunity to generate other useful individuals. In the clonal selection process, the initial population is iteratively updated by cloning-mutation-selection operation. And more effective samples can be obtained by hypermutation and the diversity of the overall set is maintained.

Before introducing the algorithm, we firstly introduce our individual representation method that is used in our algorithm. In this paper, every signature is defined as an immune cell. The input samples are defined as antigens, and the synthetic samples are treated as antibodies. A signature sample TS consists of several sequences of strokes, and every stroke s is a sampling point sequence, which consists of the points sampled between a pair of pen-down and pen-up operations. The horizontal coordinate x, the vertical coordinate y, the pressure p_r, and the time t are recorded for each sampling point sp. Because different persons have different writing habits, it is difficult to use a fixed-length sequence to define all the signatures. Even when the same person writes the same signature at different times, the number of strokes is not fixed.

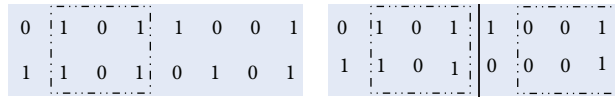

Figure 2: The segmented r-continuous bits matching rule: (a) 3-continuous bits matched but not segmented 3-continuous bits matched; (b) segmented 3-continuous bits matched.

So every signature TS is defined as a variable-length stroke sequence as follows:

$$\mathrm{TS} = s_0, s_1, \ldots, s_N, \tag{1}$$

where N is the number of the strokes.

3. Detector Generation Algorithm

Detector generation algorithm is the key of this algorithm. Detector generation algorithm [14] is widely used in negative selection algorithm to generate the candidate data. Here, it is used to create the initial population. The initial population of the standard CSA is generally generated randomly, and the structure of the random individual is much different from that of the antigen. Theoretically, it requires many times of iterations to get the population that can identify antigens. The efficiency problem caused by the random initial population can be improved by creating the antigen-guided initial population. For example, the mature detector or existing memory cells created by self-tolerance can be used as the initial population [15]. However, the diversity of the initial population created by this method is not as wide as that of the random initial population. As a result, it may make the algorithm fall into a local convergence to lose the opportunity to learn other effective structures. To solve it, a new detector generation algorithm is presented by combining the segmented r-continuous bits matching rule and P-receptor editing strategy. The detectors that are generated by segmented r-continuous bits matching rule capture the global structure of the antigen, and the detectors that are generated by P-receptor editing strategy make the initial population have a wider search space.

3.1. The Segmented r-Continuous Bits Matching Rule. The r-continuous bits matching rule is used to compute the matching degree of two strings, which is widely used in artificial immune system [16, 17]. The value r determines the matching degree. Since the r-continuous bits matching rule only measures the partial sequence, it is difficult to make the generated detector maintain the global structure of the antigen by the rule directly. Therefore, we introduce the idea of sequence segmentation. First, given two signatures, the two stroke sequences are divided into n ($n > 1$) segments simultaneously. Then a single stroke is defined as a matching bit, and if every pair of corresponding segments between the signatures is r-continuous bits matched, the two signatures are segmented r-continuous bits matched. Compared with previous r-continuous bits matching rule, the segmented r-continuous bits matching rule improves the global matching degree of two signatures.

Figure 2 shows the segmented r-continuous bits matching rule indicated by string. The two strings in Figure 2(a) are 3-continuous bits matched but not segmented 3-continuous bits matched, because the second string only has one segment that is 3-continuous bits matched. And the strings in Figure 2(b) are segmented 3-continuous bits matched, since the two strings have 2 segments that are 3-continuous bits matched. The vertical line is the segment line, while the bits in the rectangles are the same. From the simple example of Figure 2, we can see that the generated detector that is only r-continuous bits matched cannot control the last 4 bits, because the front 4 bits have satisfied the requirement. According to the segmented r-continuous bits rule, the global structure controllability is improved.

There are two important parameters in the segmented r-continuous bits matching rule: bits length r and segment number n. The bits length r shows the local matching degree, and the segment number n shows the global matching degree. There is considerable variability in the stroke number of different signatures; even the signatures given by the same person often have different stroke number in different acquisition sessions. Therefore, if fixed segment number and bits length are used to control the matching degree between the detector and antigen, it is difficult to capture the important structure information while adapting to different sketching habits and acquisition sessions. To solve it, the two parameters are determined according to the stroke number N of the signature as follows:

$$\begin{aligned} n &= \lfloor N \div 10 \rfloor + 2, \\ r &= \lfloor N \div n \rfloor - 1. \end{aligned} \tag{2}$$

Given a signature sample TS, a mutated sample is indicated by TS_χ, where the symbol χ is a subset of $\{1, 2, \ldots, N\}$. And if the ith stroke in the sample TS is mutated in TS_χ, i is an element of χ. The possible set χ is exponentially large, that is, 2^N. However, not every χ can make the TS and TS_χ segmented r-continuous bits matched. It is infeasible to check every set χ successively to find the desirable cases; besides, it is unnecessary to search all the cases. Our goal is to generate the samples that can be segmented r-continuous bits matched with the antigen, while introducing more mutation to improve the diversity of the initial population. Accordingly, an iterative enumeration method is proposed to add the valid bit to the set χ progressively while maintaining the segmented r-continuous bits matching rule, until the set χ cannot be expanded. The whole process is shown in Algorithm 1.

A constraint-based enumeration method is proposed to update the set χ. Given any set χ (t is the maximum in

(1) **Input**: number of matched continuous bits r; number of segments n
(2) **Output**: the result set list Ψ, which is a set of integral sets.
(3) Set the initial integral set $\chi = \emptyset$;
(4) Add the set χ into the set Ψ;
(5) Set the segment flag T of set χ as 0;
(6) **repeat**
(7) **for all** $\chi \in \Psi$ **do**
(8) Update the set χ and get the candidate set Ψ'; (see Algorithm 2)
(9) Remove χ from Ψ;
(10) **if** Update succeed **then**
(11) Add all the elements of the set Ψ' into Ψ;
(12) **end if**
(13) **end for**
(14) **until** Ψ is not changed

Algorithm 1: The segmented r-continuous bits matched detector generation.

the set χ), the method uses a segment flag T to indicate the segment number of the subsequence $s_0, s_1, \ldots, s_t$ of sample TS_χ; that is, the front t bits of samples TS and TS_χ are segmented r-continuous bits matched, and the largest segment number is T. According to the current set χ, 4 bit positions are defined: A, B, C, and D. Here, A means the next possible mutated bit position; B is the smallest value that makes the front B bits of samples TS and TS_χ segmented r-continuous bits matched (segment number is $T + 1$); C is the largest value that makes the last $(N-C)$ bits of samples TS and TS_χ segmented r-continuous bits matched (segment number is $(n-T)$); D is the largest value that makes the last $(N-D)$ bits of samples TS and TS_χ segmented r-continuous bits matched (segment number is $(n - T - 1)$). The four positions are computed according to the maximum t and the segment flag T as follows:

$$\begin{aligned} A &= t + 1, \\ B &= t + 1 + r, \\ C &= N - (n - T) \times r, \\ D &= N - (n - T - 1) \times r. \end{aligned} \tag{3}$$

Then the bits from A to D are selected successively to update the set χ, and the detail of the process is described as in Algorithm 2. By Algorithm 2, the possible set χ is generated to make TS and TS_χ segmented r-continuous bits matched. The corresponding positions of the strokes in the set χ are then mutated by adding the random noise to create a mutated sample TS_χ. The generated samples are the first part of the detector set.

3.2. The P-Receptor Editing Strategy. Figure 3 shows the hierarchical structure of receptor editing. This type of mutation can edit the stroke in any position without any r-continuous bits matching requirement. The p-receptor editors can be created from $(p - 1)$-receptor editors. According to the property, the generation efficiency of the p-receptor editors can be improved. The detail of this process is described as in Algorithm 3.

In Algorithm 3, the parameter p' shows the degree of discrimination between the antigen TS and the new sample

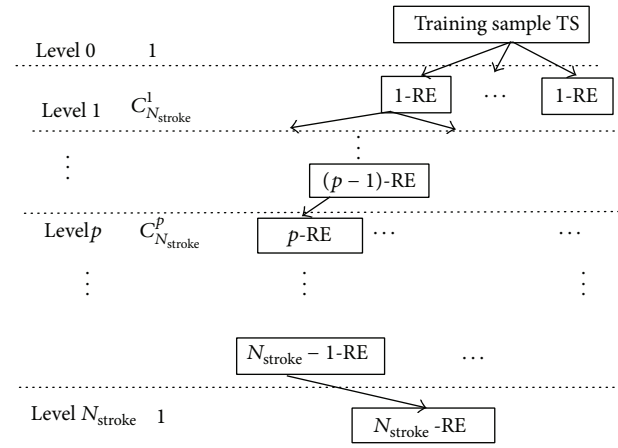

Figure 3: Hierarchical structure of receptor editing (RE) on sample TS.

TS_χ. Because the p'-receptor editors can be created from $(p' - 1)$-receptor editors, the process is the same as the mutation in the clonal selection process. As a result, the parameter p' is set as a small value for generating the initial population (in this paper $p' = 2$). The P-receptor editors are the second part of the detector set, which is used as the initial population Pop_0 of the clonal selection algorithm.

4. Clonal Selection Algorithm

After getting the initial population, the clonal selection algorithm is used to remove the invalid samples and generate more valid samples. In this section, we first give some basic operators in our CSA, such as affinity operator, mutation operator, and density operator. Then the process and stop criterion of our CSA are described.

Affinity Operator. The affinity measures the degree of matching between the antigen and antibody. Because different signatures often have different number of strokes, it is difficult to compute the matching cost directly. So the stroke segmentation algorithm based on dynamic programming (DP)

(1) **Input**: the current set χ;
(2) **Output**: the candidate set Ψ';
(3) Compute the 4 positions A, B, C and D by (3);
(4) **if** $B > C$ **then**
(5) **for** $p = A \rightarrow C$ and $B \rightarrow D$ **do**
(6) Create the new candidate integral set $\chi' = \chi \cup \{p\}$ and push it into the set Ψ';
(7) **end for**
(8) **else**
(9) **for** $p = A \rightarrow D$ **do**
(10) Create the new candidate integral set $\chi' = \chi \cup \{p\}$ and push it into the set Ψ';
(11) **end for**
(12) **end if**

Algorithm 2: Updating the integral set.

(1) **Input**: the initial sample TS $= s_1, s_2, \ldots, s_N$; the number p';
(2) **Output**: the result sample set Φ;
(3) Add the sample $\text{TS}_\emptyset$ into the set Φ;
(4) **for** $i = 1 \rightarrow p'$ **do**
(5) **for all** $\text{TS}_\chi \in \Phi$ **do**
(6) t = the maximum in the χ;
(7) **for** $j = t + 1 \rightarrow N$ **do**
(8) Create the new set $\chi' = \chi \cup \{j\}$;
(9) Mutate the jth stroke of TS_χ;
(10) Create the new sample $\text{TS}_{\chi'}$ and add it into Φ;
(11) **end for**
(12) Remove the sample TS_χ from the Φ;
(13) **end for**
(14) **end for**

Algorithm 3: The P-receptor detector generation.

[18] is firstly used to make the two signatures have the same number of strokes and establish a bijective mapping between the two stroke sequences. During the DP-based segmentation process, the set of temporal ordered candidate segment points are firstly extracted according to the curvature feature, and then the segmentation of two signatures is treated as an optimization problem, which maximizes the matching degree between the two signatures by selecting the segment points from the ordered candidate segment points. For a selected segment point, optimal segmentation contains the optimal segmentation of the input stroke(s) up to this point. Accordingly, the dynamic programming is used to search the segment points recursively through a retroactive formula in order to achieve the optimization. Figure 4 shows the DP-based segmentation result. The sample in Figure 4(a) has 17 strokes, and the sample in Figure 4(b) has 10 strokes. The x-axis coordinate curves of the two samples are shown in Figure 4(c), and the horizontal and vertical axis are the time stamp and x-axis coordinate of the sample point, respectively. The endpoints are also shown by the circles. By the DP-based segmentation, each sample is divided into 17 strokes. And the vertical lines in Figure 4(d) are the segmentation lines.

Then, Mahalanobis distance is used to compare the feature between the corresponding strokes of the two segmented samples. And the affinity between the two signatures is computed by

$$\text{Aff}\left(\text{TS}_1, \text{TS}_2\right) = \sum_{i=1}^{n'} e^{-(\text{distance}_i)^2}, \tag{4}$$

where n' is the segment number after the DP-based segmentation, and distance_i is the Mahalanobis distance between the ith segments of the samples TS_1 and TS_2. The feature vector that is used to compute Mahalanobis distance includes 2 geometric features and 4 dynamic features, as shown in Table 1. Given N_g antigens Ag, the affinity of the antibody Ab is the maximum value of the affinity between the antibody and the N_g antigens, which is computed by

$$\text{Aff}\left(\text{Ab}\right) = \max \text{Aff}\left(\text{Ab}, \text{Ag}_i\right), \quad 1 \le i \le N_g. \tag{5}$$

Mutation Operator. The mutation operator firstly selects some strokes randomly from the signature to mutate the individual. Then each selected stroke is distorted by adding random noise to the horizontal coordinate x, the vertical coordinate y, and the pressure p_r of the sampling points as follows:

$$\widehat{x} = x + U_1\left(-\omega, \omega\right) \times \frac{\left(\max_x - \min_x\right)}{\text{Aff}\left(\text{Ab}\right)},$$

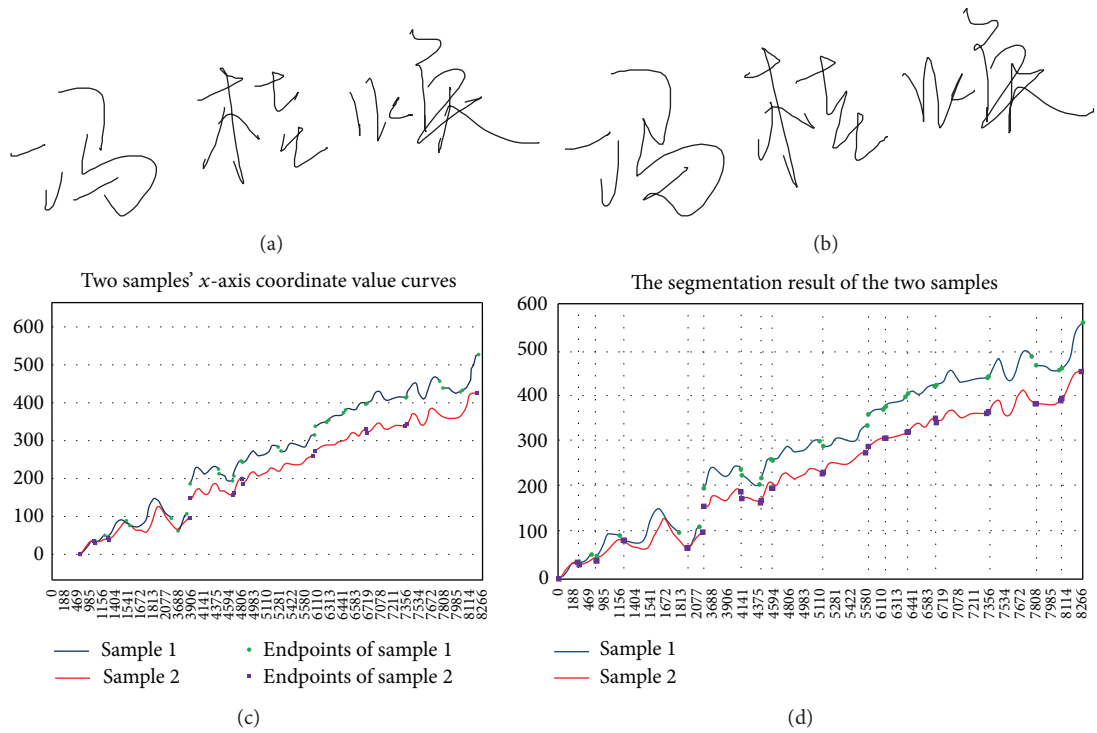

FIGURE 4: The segmentation result. (a) The first sample; (b) the second sample and its x-axis coordinate curves; (c) two samples' x-axis coordinate curves; (d) the segmentation results of the two samples.

$$\widehat{y} = y + U_2(-\omega, \omega) \times \frac{(\max_y - \min_y)}{\mathrm{Aff}(\mathrm{Ab})},$$
$$\widehat{p_r} = p_r + U_3(-\omega, \omega) \times \frac{(\max_p - \min_p)}{\mathrm{Aff}(\mathrm{Ab})}, \tag{6}$$

where $U_1(-\omega, \omega)$, $U_2(-\omega, \omega)$, and $U_3(-\omega, \omega)$ are uniform random number from $-\omega$ to ω (in this paper, $\omega = 0.05$), $\max_x$, $\min_x$, $\max_y$, $\min_y$, $\max_p$, and $\min_p$ are the maximum and minimum values of the x-axis coordinate, y-axis coordinate, and pressure of the antibody Ab, respectively. Figure 5 shows the new sample that is created by mutating the sample in Figure 4(a).

Density Operator. Density manipulation is an important characteristic in CSA to maintain the diversity of the sample set. The density of the antibody Ab is computed by

$$\mathrm{Density}(\mathrm{Ab}) = \frac{\sum_{i=1}^{M} \mathrm{Aff}(\mathrm{Ab}, \mathrm{Ab}_i)}{M}, \tag{7}$$

where M is the size of the current sample set.

After defining the above three operators, the clonal selection algorithm is used to expand the initial population Pop_0 while improving the diversity and distribution of the population. The clonal selection algorithm is described as in Algorithm 4.

The algorithm is terminated when the sample set meets the following requirements: (1) the size of the population exceeds a threshold η_1; (2) the dispersion is below a threshold η_2; (3) the change of the dispersion is below a threshold η_3. The dispersion D measures the sparsity of the sample distribution, which is computed by

$$D = \sum_{i=1}^{M} (1 - \mathrm{Aff}(\mathrm{Ab}_i)). \tag{8}$$

TABLE 1: The stroke feature.

Type	Symbol	Description
Geometric	mean_x	Mean value of the sampling points' horizontal coordinates
	mean_y	Mean value of the sampling points' vertical coordinates
Dynamic	mean_p	Mean value of the sampling points' pressure
	mean_v	Mean value of the sampling points' velocity
	mean_θ	Mean value of the sampling points' tangent direction
	str_dr	Duration time of the stroke

5. Evaluation

We have implemented the proposed algorithm and used several signatures to show the efficiency of our method. The experiment environment is Intel Core i5-2400 3.10 Ghz with 8 G of memory, based on a single threaded C++ implementation.

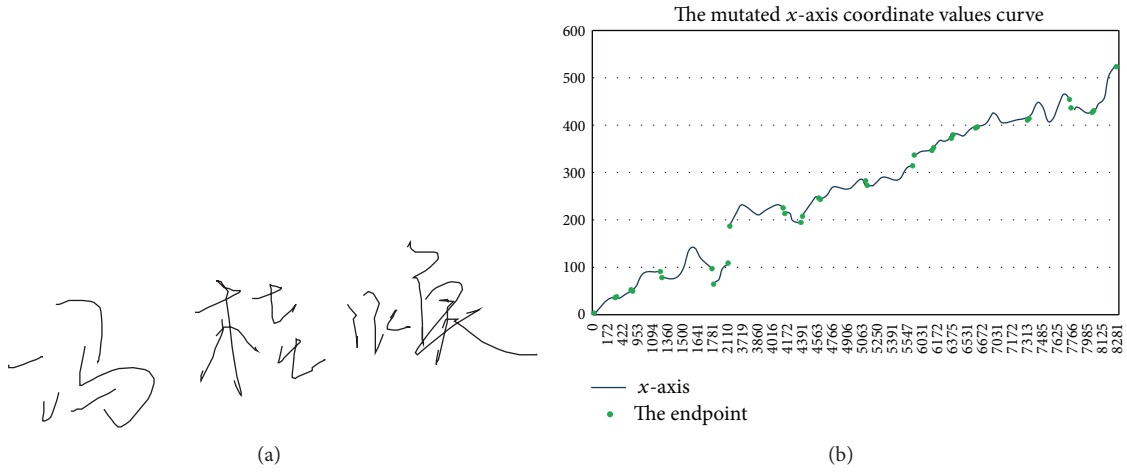

Figure 5: The mutated individual. (a) The mutated sample; (b) the mutated x-axis coordinate curve.

(1) **Input**: the initial population Pop_0;
(2) **Output**: the result population Pop_R;
(3) $\text{Pop}_R = \text{Pop}_0$;
(4) **repeat**
(5) Add some random individuals to Pop_R;
(6) Compute the affinity of every individual of Pop_R;
(7) Select the N_1 highest affinity individuals and generate a temporary population Pop_1;
(8) Compute the density of every individual of Pop_1;
(9) Clone every individual Ab of Pop_1 to generate a clone population Pop_2, and the clone number is computed by $N_2 \times (\text{Aff}(\text{Ab})/\text{Density}(\text{Ab}))/\sum_{i=1}^{M}(\text{Aff}(\text{Ab}_i)/\text{Density}(\text{Ab}_i))$;
(10) Mutate the clone population Pop_2 to a degree inversely proportional to their affinity to produce a mature population Pop_3;
(11) Compute the affinity of the individuals of Pop_3 and select the N_3 highest affinity individuals to generate the new population Pop_R
(12) **until** Termination

Algorithm 4: The clonal selection algorithm.

5.1. Data Collection. We collect some signature data for our experiment. Thirteen students are invited to give their signatures. Every student is first asked to write his signature for 50 times. Then, five other students are asked to forge his/her signature for 10 times for every student. So there are 50 genuine samples and 50 forged samples for every person. During the collection process, we record the horizontal and vertical coordinate, the pressure, and the time stamp of the sample points. The pen-up and pen-down events are also captured and the sample points between a pair of pen-up and pen-down events constitute a stroke. Then the genuine samples are divided into 10 groups and each group has five samples. Besides, we also include a public benchmark provided by the First International Signature Verification Competition (SVC2004) [19]. This corpus consists of 40 sets of signatures. Each set contains 20 genuine signatures from one contributor and 20 skilled forgeries from five other contributors, and the 20 genuine signatures are divided into 4 groups.

Fierrez's method is used as our verification system [20], which uses the hidden Markov models (HMM). The similarity of the signature is computed by using 10 left-to-right HMM states and mixtures of 8 Gaussians per state. We compute the equal error rate (EER) for performance comparison. Figure 6 shows the relation between the size of training set and EER for person 1. When the size of the training set increases (x-axis) before three groups, the EER (y-axis) decreases significantly. Our experiment focuses on investigating whether our algorithm can improve the performance when the size of the training set is small. So each time only one group (five samples) is used as the input training set. The other signatures are used as the test samples to evaluate the performance.

5.2. Parameter Settings. N_1, N_2, and N_3 are the three parameters that determine the size of the temporary, clone, and new population. We have ascertained experimentally that higher numbers of N_1, N_2, and N_3 will achieve better results. However, in order to deal with the tradeoff of computational time and accuracy, we set $N_1 = 30$, $N_2 = 1000$, and $N_3 = 200$. The mutation range ω is set as 0.05. The termination criterion includes three parameters: the size of the population η_1 is set as 200; the dispersion threshold η_2 is set as 0.75; the variation threshold of the dispersion η_3 is set as 0.01.

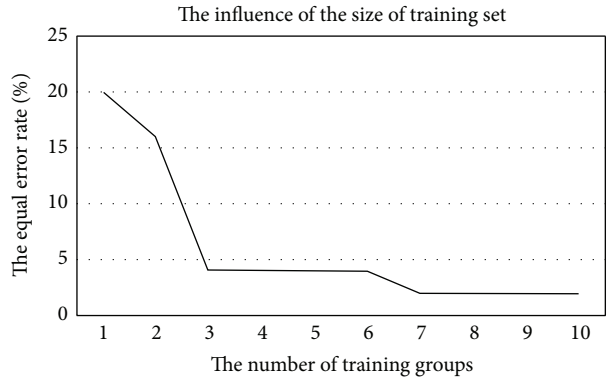

Figure 6: The relation between the size of training set and EER.

Table 2: EER statistics in % of repeating our CSA for 10 times by the same input (20 subjects).

User ID	Average	SD	Maximum	User ID	Average	SD	Maximum
1	13.50	3.37	20.00	2	14.50	1.58	15.00
3	30.50	1.58	35.00	4	14.50	1.58	15.00
5	22.50	2.64	25.00	6	19.50	1.58	20.00
7	9.50	1.58	10.00	8	26.00	3.16	30.00
9	14.00	2.11	15.00	10	5.00	1.58	10.00
11	17.00	3.50	20.00	12	7.50	1.67	10.00
13	14.00	2.11	15.00	14	29.50	1.58	30.00
15	7.50	0.00	7.50	16	7.50	1.67	10.00
17	13.00	2.58	15.00	18	9.50	1.58	10.00
19	13.50	2.41	15.00	20	0.50	1.58	5.00

SD denotes standard deviation.

Table 3: Average EERs in % of different training samples.

User ID	1	2	3	4	5	6	7	8	9	10	11	12	13
Real	14.8	3.6	5.8	15.8	1.8	30.2	10.4	13.4	7.2	18.4	13.6	**1.8**	8.4
Synthetic	**6.3**	**2.6**	**3.2**	**15.0**	**0.8**	**25.0**	**4.6**	**12.0**	**3.6**	**14.2**	**11.6**	5.8	**6.0**

5.3. Statistical Analysis. We select 20 sets of signatures from the SVC2004 database to show the statistical analysis result of the proposed CSA. The input consists of one group randomly selected from each set of signatures and the proposed CSA is executed for 10 times to generate 10 sets of synthetic signatures from the same input. Then the corresponding hidden Markov models are trained by the synthetic signatures and evaluated for signature verification. A summary of evaluation results is given in Table 2, which shows the average, standard deviation, and maximum values of the EERs. From the statistical analysis result, we can see that though the CSA is a random algorithm, the evaluation performance of the generated synthetic signatures is stable. The reason is that our detector generation algorithm creates the initial population under the guidance of the input samples and many invalid samples will not be searched due to the high quality of the initial population.

5.4. Comparison between Real and Synthetic Training Samples. We compare the performance by using different training samples to train the corresponding HMMs. The proposed CSA is firstly used to create some synthetic signatures in order to expand the initial group. Then the initial group and the expanded group (five genuine signatures) are used as the training set for the HMM-based verification system, respectively. The cross validation method is used to compare the performance, and each time one group is used for training the HMM-based recognition system. Then the other nine groups and all the forged samples are used for estimating the corresponding EER. The comparison result of our own database is shown in Table 3, which records the average EER for every invited contributor. The bolded values show better case. From the table, we can see that the performance is improved by our method. Except person 12, the EERs of the other twelve persons are decreased. Among the twelve persons, the EERs of persons 1, 5, 7, and 9 have fallen by more than 50%. Figure 7 shows the relation between false acceptance rate (FAR) and false rejection rate (FRR) of persons 1, 7, and 9. From the FAR-FRR curve, we can see that the performance is improved significantly by using the expanded sample set as the training set.

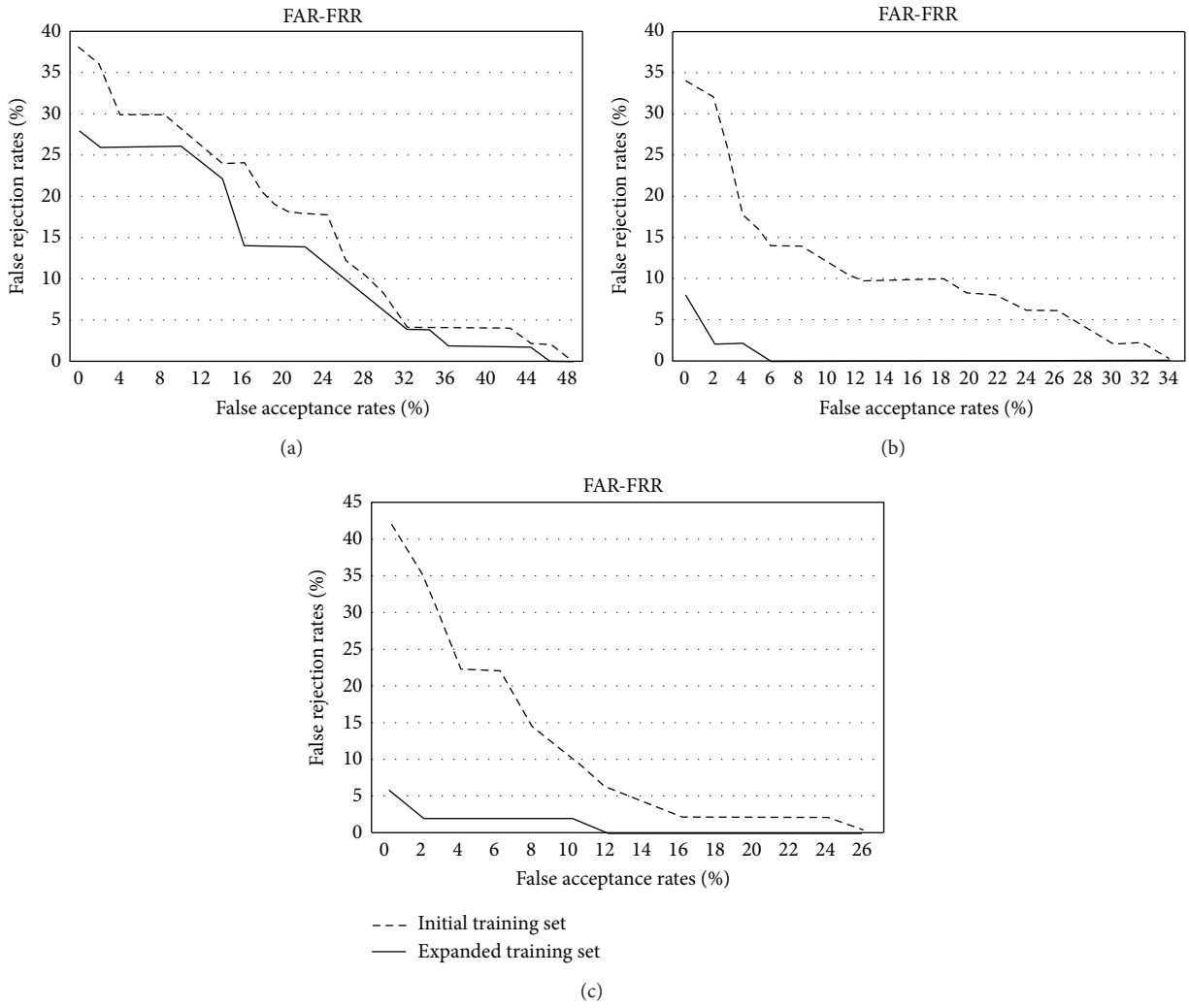

Figure 7: Training samples experiment: the relation between false acceptance rate (FAR) and false rejection (FFR) for persons 1, 7, and 9.

5.5. Comparison between Galbally's Method and Ours. We use the SVC2004 database and our collection to compare the quality of our generated synthetic signature set with Galbally's method [7]. The cross validation method is used to compute the average EERs for each set of signatures, and the same process is described in Section 5.4. Galbally's method and ours use one group as the input and create the corresponding synthetic signatures each time, respectively. Then the average EER is computed to compare the performance. Table 4 shows the comparison using SVC2004 database, while Table 5 shows the comparison using our own collection. The bolded values show better results. From the 40 sets of signatures in Table 4, there are 27 sets showing that the corresponding average EERs of our method are lower than that of Galbally's method, while there are 9 sets showing that Galbally's method performs better than ours. From the 13 sets of signatures in Table 5, there are 8 sets showing that the average EERs of our method are lower than that of Galbally's method, while there are 4 sets showing that Galbally's method performs better than ours. A summary of evaluation results is given in Table 6, which shows average, standard deviation, and maximum values of all the average EERs by SVC2004 and our own database. The results show that introducing our CSA method to optimize the whole signature improves the verification performance (12.7% and 20.8% improvement in the SVC2004 and our collection database, resp.). Besides, the verification performance of ours is more stable for different sets of signatures (the standard deviation of ours is lower than that of Galbally's method).

5.6. Comparison between Our CSA and Other CSAs. Then we compare the performance of different CSAs for synthetic sample generation. We compare three algorithms: the standard CSA (CSA1), the CSA by using the antigen as the initial population (CSA2), and our CSA (CSA3). The main differences are the generation of the initial population. CSA1 generates the initial population randomly. And CSA2 uses the antigen group as the initial population. CSA3 uses the segmented r-continuous bits matched detectors and the P-receptor editors as the initial population. We randomly select one group as the antigen group and then use the three algorithms to expand the sample set for each person.

Table 4: Average EERs (%) comparison between Galbally's method and ours by SVC2004.

User ID	1	2	3	4	5	6	7	8	9	10
Galbally's	11.25	11.25	22.50	**18.75**	**7.50**	**15.63**	**22.50**	13.75	28.75	21.25
Ours	**5.63**	**7.50**	**17.50**	21.25	12.50	25.00	31.25	**12.50**	**15.00**	**20.00**
User ID	11	12	13	14	15	16	17	18	19	20
Galbally's	**1.25**	11.25	21.25	**12.50**	11.25	16.25	45.00	25.00	5.00	0.00
Ours	10.00	**6.25**	**20.00**	20.00	11.25	**15.00**	**22.50**	25.00	**3.75**	0.00
User ID	21	22	23	24	25	26	27	28	29	30
Galbally's	5.63	3.75	10.00	3.13	20.00	11.25	10.00	3.75	6.88	8.75
Ours	**0.63**	**0.00**	**1.88**	**1.25**	**8.13**	**6.88**	**8.13**	**2.50**	**6.25**	**0.63**
User ID	31	32	33	34	35	36	37	38	39	40
Galbally's	13.75	28.75	9.38	2.50	**13.75**	**15.00**	**11.25**	11.88	1.88	0.00
Ours	**10.00**	**25.00**	**5.00**	**0.00**	20.00	26.25	12.50	**11.25**	**0.00**	0.00

Table 5: Average EERs (%) comparison between Galbally's method and ours by our own collection.

User ID	1	2	3	4	5	6	7	8	9	10	11	12	13
Galbally's	16.0	**1.6**	4.5	**14.6**	1.7	25.6	**3.4**	**7.6**	3.6	25.6	16.6	12.6	6.5
Ours	**6.3**	2.6	**3.2**	15.0	**0.8**	**25.0**	4.6	12.0	3.6	**14.2**	**11.6**	**5.8**	**6.0**

Table 6: EER statistics in % for the comparison between Galbally's method and ours.

Database	Galbally's			Ours		
	Average	SD	Maximum	Average	SD	Maximum
SVC2004	12.83	9.22	45.00	11.20	8.99	31.25
Our collection	10.76	8.46	25.60	8.52	6.76	25.00

SD denotes standard deviation.

Table 7: Comparison of the average running time(s).

User ID	1	2	3	4	5	6	7	8	9	10	11	12	13
CSA1	2608	1292	1779	934	115	3043	1131	1212	**47**	2171	3005	1386	1322
CSA2	2491	1151	**629**	751	**98**	2637	629	1222	75	2054	2712	1478	919
CSA3	**791**	**102**	729	**215**	200	**525**	**286**	**110**	89	**226**	**686**	**584**	**183**

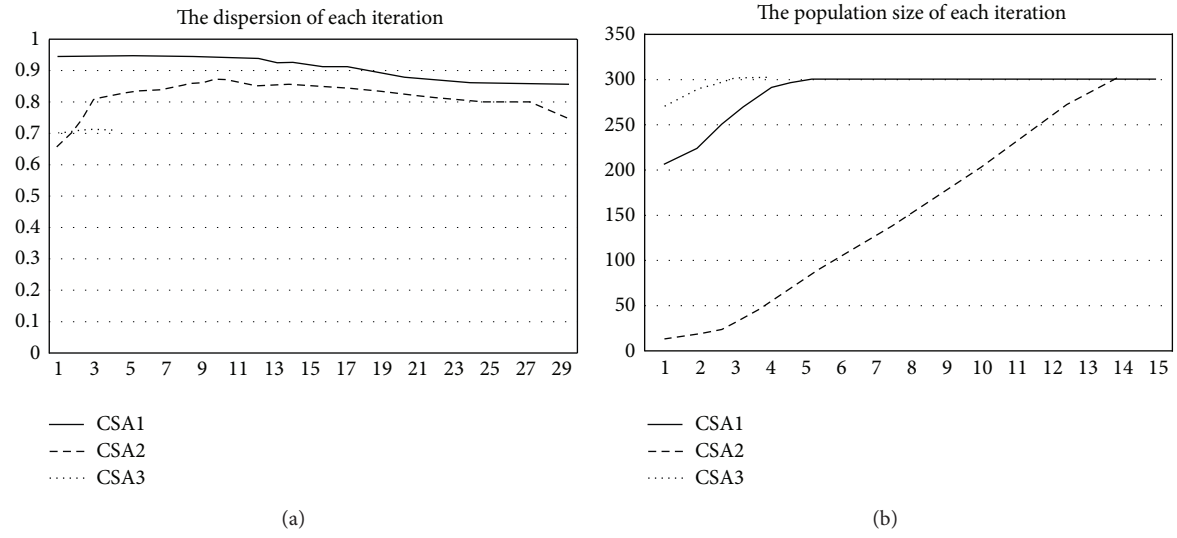

Figure 8: The convergence analysis. (a) The dispersion; (b) the population size.

Table 8: Average EERs in % of different CSA algorithms.

User ID	1	2	3	4	5	6	7	8	9	10	11	12	13
CSA1	16	6	10	18	6	44	10	18	12	10	20	8	10
CSA2	18	6	10	18	6	52	10	12	12	10	20	8	14
CSA3	**8**	**4**	**6**	**12**	**4**	**26**	**8**	**8**	**2**	**2**	20	**4**	**6**
Real	24	8	16	18	6	34	16	**8**	12	10	20	**4**	8

We compare the running time of the three algorithms, as shown in Table 7. The bolded values show the fastest algorithm. The input consists of one group that is selected randomly from each set of our own collection. Each CSA is executed for 10 times, and the average running time is then computed. From Table 7, we can see that our algorithm is significantly faster than the other two algorithms in most cases. The main reason is that our initial population is created by the proposed detector generation algorithm. Figure 8 shows the corresponding convergence analysis results. The dispersion of the initial population in CSA1 is rather high, while the size of the initial population in CSA2 is very small. So it needs sufficient number of iteration to achieve the convergence. And our algorithm uses the segmented r-continuous bits matching rule and P-receptor editing strategy to create the initial population, and both the size and dispersion are optimized preliminarily. The iteration number of CSA3 is 4 in this case, which is significantly smaller than that of CSA1 and CSA2.

We also use the expanded sample sets that are created by the three algorithms to train HMMs for signature verification, respectively. The average EERs are shown in Table 8. From the table, we can see that the EERs of our algorithm are better than that of the CSA1 and CSA2. In some cases, the EERs of CSA1 and CSA2 are larger than that of verification using the initial real samples as the training set. The experiment shows that the generation of better initial population is important to improve both the efficiency and effectiveness of the method.

6. Conclusion

In this paper, we present a novel clonal selection algorithm for synthetic sample generation. Our method focuses on the overall set rather than creating a sample successively in order to improve the signature verification performance by expanding the initial signature set. The proposed clonal selection algorithm keeps the diversity of the population while maintaining the feature distribution nonsparse. To improve the efficiency and effectiveness of the standard CSA, the detector generation algorithm is introduced by combining the segmented r-continuous bits matching rule and the P-receptor editing strategy to create the initial population for clonal selection process. The experiment shows the efficiency and effectiveness of the method. By using the synthetic samples as the training samples, the performance of the signature verification system is improved. The future work is to extend our method to other types of synthetic signature generation methods, such as the combination-based or the synthetic-individual method.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is supported by the National High Technology Research and Development Program of China (2007AA01Z334), National Natural Science Foundation of China (61321491, 61272219, 61021062, and 61100110), Program for New Century Excellent Talents in the University of China (NCET-04-04605), Project (no. ZZKT2013A12) supported by Key Projects Innovation Fund of State Key Laboratory, Natural Science Foundation of Jiangsu Province (BK2010375), Key Technology R & D Program of Jiangsu Province (BY2012190, BE2010072, BE2011058, and BY2013072-04), Project (no. CXLX12_0054) supported by the Graduate Training Innovative Projects Foundation of Jiangsu Province, and the program B for Outstanding Ph.D. Candidate of Nanjing University.

References

[1] K. R. Radhika, G. N. Sekhar, and M. K. Venkatesha, "Pattern recognition techniques in on-line hand written signature verification: a survey," in *Proceedings of the International Conference on Multimedia Computing and Systems (ICMCS '09)*, pp. 216–221, April 2009.

[2] K. Barkoula, G. Economou, and S. Fotopoulos, "Online signature verification based on signatures turning angle representation using longest common subsequence matching," *International Journal on Document Analysis and Recognition*, vol. 16, no. 3, pp. 261–272, 2013.

[3] M. Parodi and J. C. Gómez, "Legendre polynomials based feature extraction for online signature verification: consistency analysis of feature combinations," *Pattern Recognition*, vol. 47, no. 1, pp. 128–140, 2014.

[4] M. A. Ferrer, M. Diaz-Cabrera, and A. Morales, "Synthetic offline signature image generation," in *Proceedings of the International Conference on Biometrics (ICB '13)*, pp. 1–7, 2013.

[5] C. Rabasse, R. M. Guest, and M. C. Fairhurst, "A new method for the synthesis of signature data with natural variability," *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics*, vol. 38, no. 3, pp. 691–699, 2008.

[6] J. Galbally, R. Plamondon, J. Fierrez, and J. Ortega-Garcia, "Synthetic on-line signature generation. Part I: methodology and algorithms," *Pattern Recognition*, vol. 45, no. 7, pp. 2610–2621, 2012.

[7] J. Galbally, J. Fierrez, M. Martinez-Diaz, and J. Ortega-Garcia, "Improving the enrollment in dynamic signature verification with synthetic samples," in *Proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR '09)*, pp. 1295–1299, July 2009.

[8] M. Djioua, C. O'Reilly, and R. Plamondon, "An interactive trajectory synthesizer to study outlier patterns in handwriting recognition and signature verification," in *Proceedings of the 18th International Conference on Pattern Recognition (ICPR '06)*, vol. 1, pp. 1124–1127, August 2006.

[9] C. Oliveira, C. A. Kaestner, F. Bortolozzi, and R. Sabourin, "Generation of signatures by deformations," in *Advances in Document Image Analysis*, N. A. Murshed and F. Bortolozzi, Eds., vol. 1339 of *Lecture Notes in Computer Science*, pp. 283–298, Springer, Berlin, Germany, 1997.

[10] C. Rabasse, R. M. Guest, and M. C. Fairhurst, "A method for the synthesis of dynamic biometric signature data," in *Proceedings of the 9th International Conference on Document Analysis and Recognition (ICDAR '07)*, vol. 1, pp. 168–172, September 2007.

[11] L. Ballard, D. Lopresti, and F. Monrose, "Forgery quality and its implications for behavioral biometric security," *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics*, vol. 37, no. 5, pp. 1107–1118, 2007.

[12] B. Fang and Y. Y. Tang, "Improved class statistics estimation for sparse data problems in offline signature verification," *IEEE Transactions on Systems, Man and Cybernetics C: Applications and Reviews*, vol. 35, no. 3, pp. 276–286, 2005.

[13] L. N. de Castro and F. J. von Zuben, "Learning and optimization using the clonal selection principle," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 3, pp. 239–251, 2002.

[14] S. Forrest, L. Allen, A. S. Perelson, and R. Cherukuri, "Self-nonself discrimination in a computer," in *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pp. 202–212, May 1994.

[15] T. Li, "An immunity based network security risk estimation," *Science in China F: Information Sciences*, vol. 48, no. 5, pp. 557–578, 2005.

[16] U. Aickelin, J. Greensmith, and J. Twycross, "Immune system approaches to intrusion detection: a review," in *Artificial Immune Systems*, G. Nicosia, V. Cutello, P. J. Bentley, and J. Timmis, Eds., vol. 3239 of *Lecture Notes in Computer Science*, pp. 316–329, Springer, Berlin, Germany, 2004.

[17] W. Zheng, D. Qi, K. Xu, and H. Han, "A rapid r-continuous bits matching algorithm for large-scale immunocomputing," in *Proceedings of the International Conference on Computer Science and Software Engineering (CSSE '08)*, vol. 1, pp. 431–434, December 2008.

[18] Z. X. Sun, B. Yuan, and J. Yin, "Online composite sketchy shape recognition using dynamic programming," in *Graphics Recognition. Ten Years Review and Future Perspectives*, W. Liu and J. Llads, Eds., vol. 3926 of *Lecture Notes in Computer Science*, pp. 255–266, Springer, Berlin, Germany, 2006.

[19] D. Y. Yeung, H. Chang, Y. Xiong et al., "Svc2004: first international signature verification competition," in *Biometric Authentication*, D. Zhang and A. K. Jain, Eds., vol. 3072 of *Lecture Notes in Computer Science*, pp. 16–22, Springer, Berlin, Germany, 2004.

[20] J. Fierrez, J. Ortega-Garcia, D. Ramos, and J. Gonzalez-Rodriguez, "HMM-based on-line signature verification: feature extraction and signature modeling," *Pattern Recognition Letters*, vol. 28, no. 16, pp. 2325–2334, 2007.