Handwritten Signature Authentication using Artificial Neural Networks

Milton Roberto Heinen and Fernando Santos Osório, *Member, IEEE*

Abstract— **The main goal of this paper is to describe our research and implementation of a handwritten signature authentication system based on artificial neural networks. In this system the authentication process occurs in the following way: firstly, the users' signatures are read using a pen tablet device and then stored; after that some adjustments in position and scale are accomplished; representative signature features are extracted; the input space dimensionality is reduced using principal component analysis; and finally, the users' signatures are classified as authentic or not, through the use of a neural network. Several experiments were accomplished using a 2440 real signatures database, and the obtained results were very satisfactory.**

I. Introduction

Nowadays, one of main problems in computer security systems is the users' authentication, that is, how to assure that the one is trying to access a system is really the legitimate user. In many information systems, the users' authentication is assured through the use of alphanumeric passwords, which need to be memorized by the users and maintained safe from other people. Even if this is the most used users' authentication method, password authentication has some important security vulnerabilities, mainly because passwords can be easily stolen by someone else.

In order to overcome this password related security problem, biometric authentication techniques based on physiological characteristics, like fingerprint identification, iris and retina recognition were also adopted to guarantee users' authenticity[1]. These techniques are much more secure and effective than passwords, but there were some disadvantages, mainly because they are very intrusive[2], [3]. For example, fingerprint identification have a negative connotation, because this procedure is related to criminal investigations[4].

Beyond the human physiological biometric features, we can also adopt behavioral biometric characteristics for users' authentication, as for example handwritten signatures[3], [5]. Behavioral biometric methods have several advantages related to other techniques. The first advantage is directly related to the security level, because in opposition to passwords, even if someone knows the user signature, usually it is not possible to reproduce easily this signature. The reproduction of a signature is more difficult as, besides the final signature shape, we include behavioral parameters related to user pen movements. The other advantage come from the user's comfort, because users have the habit of use handwritten signatures in business transactions, and they feel secure regarding this authentication method. The degree of intrusion presented in signature authentication systems is very low, and the necessary hardware has a low cost too (usually costing from U$30 to U$100).

Although it has the described advantages, the handwritten signatures authentication is a problem difficult to solve. This is due to the large variability that exists between signatures of different people and even in the signatures of a single person[6]. This variability leads to the use of solutions based on machine learning techniques, like artificial neural networks, to achieve better results compared to other solutions[7].

In this work we developed a handwritten signature authentication methodology and its implementation in a system prototype[8], [9], [10], [11]. This paper is structured as follows: the Section II present some concepts associated to signature authentication; the Section III describes some machine learning techniques, like artificial neural networks and principal component analysis; the Section IV describes the proposed signatures authentication system and the developed prototype; the Section V shows the experiments results and the Section VI presents some final remarks and future research perspectives.

II. Handwritten Signature Authentication

A signature authentication system is a software responsible for validating signatures, indicating if a signature is authentic or not. When a signature is not authentic, probably it can be categorized in one of these three different forgeries[12], [8]:

- Random forgeries: these forgeries are accomplished by people that ignore the design of the original signatures, or don't have the ability to correctly reproduce it;
- Traced forgeries: these forgeries are accomplished following the original signature outline available in a printed form, resulting in a forgery with a shape quite close to the original signature shape;
- Skilled forgeries: these forgeries are accomplished by expert people that possesses the ability to reproduce signatures in a very satisfactory way[13].

The signatures data acquisition and authentication can be implemented in two different ways: on-line and offline[12]. The offline authentication method uses previously drawn signatures in paper sheets, which are digitalized later using a scanner. After that, they are treated by the signature authentication system. The on-line signature authentication systems use a special hardware device to directly input the signature drawn to the system, like a Pen Tablet (Figure 1).

Milton Roberto Heinen and Fernando Santos Osório are with the Applied Computing (PIPCA), Universidade do Vale do Rio dos Sinos (UNISINOS), São Leopoldo, CEP 93022-000, Brazil (email: mheinen@turing.unisinos.br, fosorio@unisinos.br).

Fig. 1. Example of a digitalizing tablet

The on-line signature authentication has several advantages related to the offline authentication[8], [11]:

- Captures more information: besides the visual signature features, it is also possible to obtain temporal and dynamical signature information (user behavioral information);
- Captures better information: a scanner digitalized signature can have a high level of noise (image artifacts and distortions), what hinders the authentication process. Pen tablet devices can provide better signature data.
- Popularization of tablet based input devices (e.g. palmtops, handhelds and tablets), simplify data acquisition.

In this work, we chose to use on-line signatures' authentication due to these main advantages, and also because we consider this method more effective and well adapted to electronic transactions.

III. Machine Learning

According to Mitchell[14], a program is capable to learn when its performance is improved with the experience in a certain task. So, to define a machine learning problem, we should identify three fundamental characteristics: the task to be learned, the performance measure and the experience source. It is also necessary that the knowledge to be learned through the experience, called objective function, must be well defined. In the case of signatures authentication, the knowledge that must be learned is a signature database composed of previously well classified (as authentic or not) signatures. This kind of application problem adopts supervised machine learning methods. Therefore, the task to be learned is the signature classification, the experience source is the signatures database and the performance measure is the evaluation of the correct answers rate in the overall signature authentication task. Nowadays, several machine learning techniques can be used for signatures' authentication, as for example, induction of decision trees, fuzzy inference systems, genetic algorithms[15] and artificial neural networks[16], [17].

A. Artificial Neural Networks

Through the use of an abstract and simplified model of human neurons, is possible to develop a neural simulator capable to classify, to generalize and to learn how to classify and approximate functions. In this work we are particularly interested in the artificial neural networks (ANN) classification properties. One of the most used neural learning models is the so called multi-layer perceptron (MLP) with back-propagation learning algorithm[16]. Some improved versions of the original back-propagation algorithm were developed in the few past years, and the Resilient Propagation (RPROP) algorithm[18] become an interesting choice among them. The RPROP algorithm performs a direct adaptation of the weight step (learning rate) based on local gradient information. To achieve this, each weight has its individual update value Δ_{ij}, which solely determines the size of the weight update. This adaptive update-value evolves during the learning process based on its local sight of the error function E, according to the following learning-rule[18]:

$$\Delta_{ij}^{(t)} = \begin{cases} \eta^{+} * \Delta_{ij}^{(t-1)} \ , & \text{if } \frac{\partial E}{\partial w_{ij}}^{(t-1)} * \frac{\partial E}{\partial w_{ij}}^{(t)} > 0 \\ \eta^{-} * \Delta_{ij}^{(t-1)} \ , & \text{if } \frac{\partial E}{\partial w_{ij}}^{(t-1)} * \frac{\partial E}{\partial w_{ij}}^{(t)} < 0 \\ \quad \Delta_{ij}^{(t-1)} \ , & \text{else} \end{cases} \qquad (1)$$

where $0 < \eta^{-} < 1 < \eta^{+}$, $\frac{\partial E}{\partial w_{ij}}$ is the partial derivative of the error function for the weight w_{ij}, and $\Delta_{ij}^{(t-1)}$ is the last weight update.

In order to learn a specific task, it is necessary a database containing training examples with input patterns and expected answers (patterns and their corresponding classes). This learning database is presented to the artificial neural network, which can learn how to answer in a similar way to the database examples, classifying the patterns. It can also be used a second database for learning validation (evaluation of generalization level), that is only used to evaluate the ANN performance (it is not used to adjust the ANN parameters and weights). This second database is different from the one used in the learning task[19]. This type of learning is known as supervised learning with cross-validation[17].

Through an iterative process, learning database examples are presented to the artificial neural network, adapting the neural network connection weights. These weights simulate the reinforcement and inhibition of synaptic connections present in real neurons. The neural learning occurs from this weights adaptation. The weights' optimizations are responsible to do the neural network learn how to answer correctly to the input data, accordingly to the examples contained in the learning database.

B. Principal Component Analysis

Principal component analysis (PCA) [20] is an essential technique in data compression and feature extraction and selection. Methods for input space dimensionality reduction, as the PCA, are used to discard those linear combinations of input variables which have small variances and to preserve only those that have large variances. Even if all linear combinations are maintained, when the variances are as non uniform as possible, variable-length coding schemes allow a very efficient coding and decoding[21].

Assume that x is an n-dimensional input data vector that is zero mean centered. The purpose of the PCA is to find those $p\,(p \le n)$ linear combinations $w_1^T x, w_2^T x, \cdots, w_p^T x$ of the elements of x that maximizes

$$E\{(w_i^T x)^2\}, \;\; i = 1, \cdots, p \tag{2}$$

under the constraints

$$w_i^T w_j = \delta_{ij}, \;\; j < i. \tag{3}$$

The solution for the vectors $w_1, \cdots, w_p$ are the p dominant eigenvectors of the data covariance matrix

$$C = E\{xx^T\}. \tag{4}$$

These are the p orthogonal unit vectors $c_1, \cdots, c_p$ given by

$$Cc_i = \lambda_i c_i \tag{5}$$

where $\lambda_1, \cdots, \lambda_p$ are the p largest eigenvalues of matrix C in descending order of magnitude. The first linear combination $c_1^T x$ is called the first principal component, the second linear combination $c_2^T x$ is called second principal component, and so on[21].

In a signature authentication system, each input variable is a feature extracted from the signature, and a signature description can be composed by several features. Thus, we can use the PCA to reduce the input space dimensionality, discarding those linear combinations that have small variances, and retaining only those terms that have large variances[17].

IV. Proposed System

The main goal of this paper is to propose a methodology for on-line handwritten signature authentication based on artificial neural networks. To validate our approach, a practical system was proposed and implemented in a prototype[1]. In the proposed system, the signature authentication is accomplished in the following way:

- The signatures are collected and stored in a database;
- Some position and scale adjustments are accomplished over the signatures;
- Relevant features used in the authentication process are extracted from the signatures;
- The input space dimensionality is reduced using PCA;
- The signatures' authentication is accomplished using neural networks.

A. Signature Acquisition

In order to validate the proposed system, it was necessary the creation of a signatures database. Thus, it was developed a module of the system called Data Acquisition Module. This module is responsible for read the signature data from the tablet and to save these data. This module generates a signature description file, where each signature is composed by a sequence of pen coordinates (x, y), state (1 or 0: drawing or pen lifted up) and time stamp for each captured point (in milliseconds). The pen tablet we used in the experiments with our system was a SuperPen WP4030, manufactured by UC-Logic[2]. The Figure 2 shows an example of a typical signature (the captured points were highlighted).

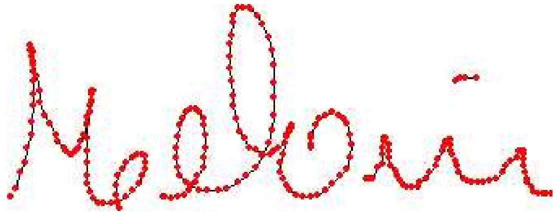


Fig. 2. Example of a typical signature

The signatures' database used in our experiments was collected during one year, and each user contributed with several signatures picked up in different moments, so typical variations that happen along the time in a signature are also present in the database. The composition of the 2440 signatures database was the following:

- 1800 authentic signatures, accomplished by 60 different users (30 signatures per user);
- 400 handwriting images, representing 40 drawing models (10 patterns per model);
- 120 traced forgeries;
- 120 skilled forgeries;

In our initial experiments it was determined that about 10 signatures per user would be enough for an adequate neural network learning. Since the learning validation process needs more different signatures (cross-validation), we decided to collect up to 30 signatures per user, so we were able to learn and to test properly the authentication task.

The handwriting images represent various drawings, that don't have a direct relationship with the user's name[22]. The traced forgeries were accomplished following the original signature outline available in a printed form, and the skilled forgeries were accomplished by expert people that practiced for some time until to be able to reproduce the authentic signatures in a very satisfactory way.

Once the signatures database was created, position and size adjustments were accomplished over these signatures, aiming to become the authentication process more robust. The position adjustments minimize the variations in the signature position over a virtual grid. These variations are common among different signatures from a single person, since the signature position can be displaced according to its starting point in the grid. The implemented algorithms allow to adjust signatures related to the upper left corner or to the signature center of gravity. The scale adjustments allow to resize the signatures to a standard size, so the differences of scale among the signatures of users are minimized;

B. Feature Extraction

After the signatures database was created and signature adjustments were done, the extraction of features was accomplished. Some techniques of signature features extraction used in our system prototype were found in the literature[23], [24], [25], and many other feature extraction techniques were created or specifically adapted by us, where a detailed description of them can be found in [26], [8], [9], [10], [11]. After the study and implementation of the signature features

[1] NeuralSignX – http://www.inf.unisinos.br/~osorio/neuralsignx/

[2] UC-Logic SuperPen – http://www.superpen.com/

extraction methods, a new study was accomplished in order to verify the relevance of each feature. The main signature features we selected to use in this work are (the numbers between parentheses represents the number of neural entries for each feature):

Signature elapsed time: the signature time duration from the start until the end of the signature drawing (1 entry);

Quantity of pen lifts: the number of times that the pen leave the tablet during the signature drawing (1 entry);

Total signature length: the total distance covered (path length) by the pen during the signature drawing (1 entry);

Medium and maximum pen velocity: the overall medium speed and the maximum speed of pen movements (2 entries);

Number of pen direction changes: the quantity of times the pen changed of direction related to the x and y axis. We consider that a direction change occurred when an increasing coordinate value began to decrease, and vice versa (2 entries);

Cardinal points measure: the number of pseudo-vectors (simplified structural shape) that are pointing to each cardinal point section (N, S, E, W, NE, NW, SE, SW) [9], [10]. The Figure 3 shows the eight cardinal point sections defined and some signature pseudo-vectors used to quantify this measure (8 entries);

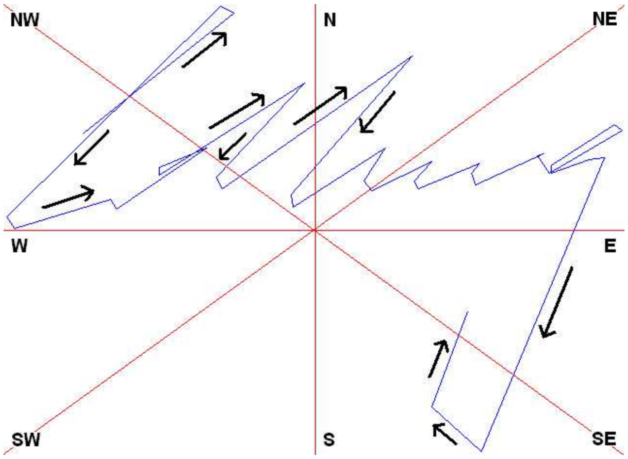


Fig. 3. Cardinal point sections and pseudo-vectors

Pseudo-vectors total length: the total length of all pseudo-vectors pointing to the same cardinal point (8 entries);

Signature density grid: the signature is divided into several cells, and for each cell the density (number of points falling into this cell) is calculated (adapted from [7]). For the cell i, the density D_i is calculated through the equation:

$$D_i = \frac{np_i - np_{min}}{np_{max} - np_{min}}, \tag{6}$$

where np_i is the number of points falling into the cell i, np_{min} is the number of points of the least dense cell, and np_{max} is the number of points of the most dense cell. The Figure 4 demonstrates the use of this feature, in which high density cells in the grid are more dark than others. (48 entries);

Vertical and horizontal line intersections: the bitmap containing the signature is intersected by virtual lines that cut the signature in fixed intervals, and for each virtual line we count how many times this line intersects the signature drawing[26]. The Figure 5 shows the vertical and horizontal lines used by this feature (26 entries);

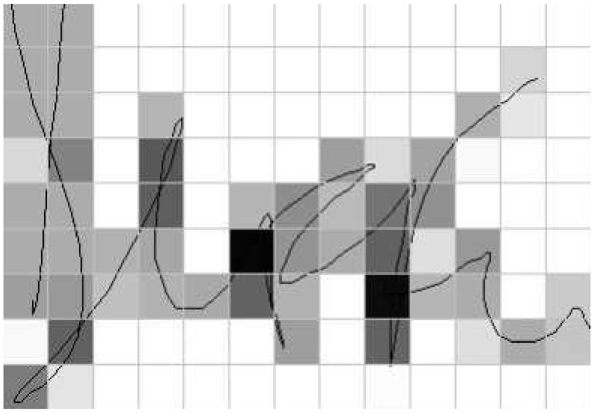


Fig. 4. Signature density grid

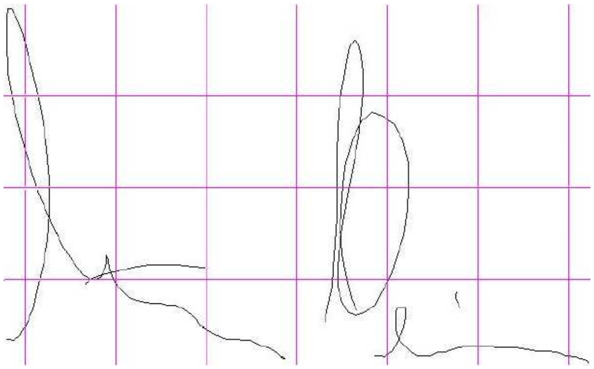


Fig. 5. Vertical and horizontal line intersections

Sequential signature sampling: the whole signature trajectory is represented only by a small number (predefined) of sampled points. The Figure 6 shows the sequential signature sampling. (16 entries);

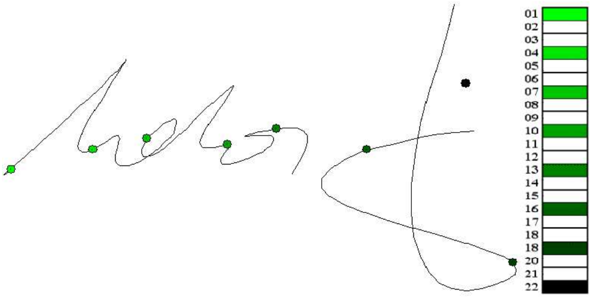


Fig. 6. Sequential signature sampling

Symmetry: the symmetry level of the signature is measured in relation to the x and y axis[26] (2 entries).

C. Input Space Reduction

To reduce the total number of neural network inputs, and to avoid the problems that usually happen when this number is very high[17], the features *Cardinal points measure (Cardinal)*, *Pseudo-vectors total length (Vectors)*, *Vertical and horizontal line intersections (Intersec)*, *Sequential signature sampling (Sequent)* and *Signature density grid (Density)*, which result in several ANN inputs, were submitted to PCA. The PCA was used to extract only the first principal components of each feature, and thus to reduce the neural network input space, with the minimum loss of information as possible.

The Table I shows the cumulative variance of the first six components for the features mentioned above. The first column (Feature) shows the short name of the features, the columns C1 to C6 show the cumulative variance (percentage) for the first six PCA components, the column NI shows the original number of inputs for each feature and the column NC shows the number of factors selected to be used in the signature authentication process. We adopted the selection of PCA components that explain at least 60% of the variance for each feature in the database. The cumulative variance for the first twelve components of the *Signature density grid* feature is 62.37%.

TABLE I
CUMULATIVE VARIANCE OF THE FIRST SIX COMPONENTS

Feature	C1	C2	C3	C4	C5	C6	NI	NC
Density	19.8	27.6	32.8	37.1	41.4	45.2	48	12
Intersec	29.5	44.5	52.0	57.3	61.4	65.2	26	5
Sequent	21.5	32.5	42.6	51.7	59.1	65.8	16	6
Cardinal	56.8	68.9	77.5	85.7	91.0	95.3	8	3
Vectors	25.6	44.9	59.9	71.7	80.7	89.4	8	3

Using the PCA, the input space dimensionality was reduced from 117 to 40 inputs. The only disadvantage of input space reduction is that a small part of the total available information is lost. However, the ANN generalization rate increased with this input space dimensionality reduction, which justifies some information loss.

D. Signatures Authentication

After the feature extraction and the dimensionality reduction processing, the neural network is used to authenticate the signatures. The neural network receives the features values and uses them to classify the signatures as authentic or not. For the signatures classification, the artificial neural network model used was a MLP with back-propagation. It was also used for weights optimization a learning algorithm with a better performance than the traditional back-propagation, the algorithm RPROP[18]. This algorithm was selected because it is more efficient than back-propagation, making the learning process much faster. The neural network simulator adopted was the Stuttgart Neural Network Simulator - SNNS[3], it is a free software, and a quite complete neural network simulator that have several additional tools that allow us to create scripts and execute learning and simulation tasks in batch mode. The SNNS facilities also simplify the analysis of the obtained results and creation of graphic plots.

The main ANN parameters we configured to use with the SNNS simulator are showed in Table II. Because the RPROP automatically adjust the values for the Learning rate during the simulation, the default value (0.1) was retained for this parameter. Several preliminary experiments are accomplished, and it was verified that no neurons are necessary in the hidden layer (the patterns are linearly separable). The experiments were accomplished during 1000 epochs each. The neural network we used in our experiments had only 1 output, The value 1.0 obtained in the output represents an authentic signature of a specific user, and the value 0.0 represents a non authentic signature of this user. The Score threshold was fixed in 0.4.

TABLE II
NEURAL NETWORK PARAMETERS

Parameter	**Value**
ANN model	MLP - back-propagation
Learning algorithm	Resilient propagation
Number of inputs	117 or 40
Number of outputs	1
Number of hidden neurons	0
Activation function	Sigmoid
Learning rate	0.1
Maximum of generations	1000
Sigmoid prime offset	0.1
Weight decay	0 or 1.0×10^{-38}
Score threshold	0.4

In order to make the experiments statistically valid, the simulations were accomplished using a ten-fold cross-validation method. For each individual fold we also repeated 10 times the simulation using exactly the same data and parameters, with only different initializations of the weights. The folds division were obtained ensuring the proportion between examples of the 0 and 1 class in each fold.

V. RESULTS

This section describes the main results obtained in the experiments accomplished using our system prototype. Initially we present the experiments accomplished using the complete input space dimensionality (117 entries), and after we present the experiments accomplished using a reduced input space dimensionality (40 entries), and we conclude this section presenting a comparison between the results of the different experiments.

A. Full Features Experiments

In the first set of experiments, we used the whole set of features available in the system. The neural network used in the experiments had 117 neurons in the input layer and one neuron in the output layer. The neural network parameters used in the simulations are showed in Table II (in these experiments the weight decay was set to 0.0). The signature database was composed of 2440 examples always divided into 10 different folds (10-fold cross-validation).

The Table III shows the results obtained in the first experiments for ten different users in the generalization test database. For each user, it was used a 10-fold cross-validation

[3] SNNS – http://www-ra.informatik.uni-tuebingen.de/SNNS/

method, and it was calculated the mean MSE error (mean square error), the mean and standard deviation values of the following measures: correct authentication rate (HIT), false positive rate (FPR) and false negative rate (FNR). The first column (U) shows the user identification number. All the values presented in the Table III are expressed in percentages, excluding the MSE error. The Table IV shows the results obtained in the same experiments, but showing the results for the learning database (instead of the test database results), which contains 2196 patterns of the class 0 and 27 patterns of the class 1. The amount of time necessary to simulate the Table III experiments was 2.57 hours in a typical computer[4].

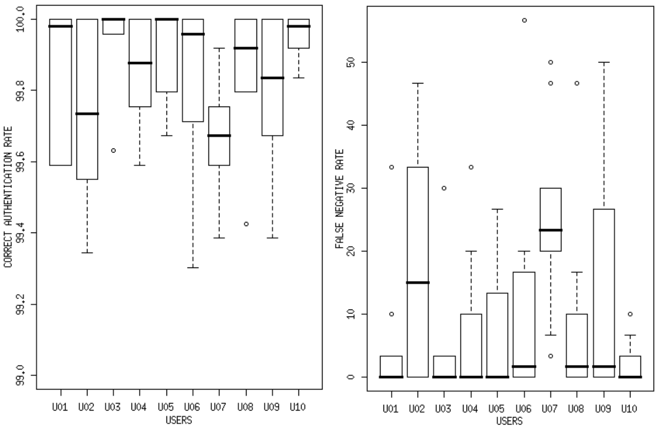
TABLE III
SIMULATION RESULTS USING THE COMPLETE SET OF INPUT FEATURES (GENERALIZATION TEST DATABASE)

	MSE	HIT		FPR		FNR	
U	μ	μ	σ	μ	σ	μ	σ
01	2.06e-02	97.94	4.17	2.02	4.21	4.67	10.56
02	2.63e-03	99.72	0.24	0.06	0.09	18.00	17.86
03	7.36e-04	99.92	0.15	0.00	0.00	6.33	12.52
04	1.43e-03	99.85	0.16	0.07	0.09	6.67	11.44
05	8.11e-04	99.91	0.14	0.00	0.01	6.67	11.33
06	1.91e-03	99.80	0.26	0.07	0.17	10.33	17.88
07	3.19e-03	99.67	0.18	0.02	0.04	25.33	14.92
08	1.12e-03	99.87	0.18	0.02	0.05	8.67	14.59
09	1.94e-03	99.79	0.23	0.07	0.09	11.67	18.21
10	4.15e-04	99.95	0.06	0.02	0.02	2.33	3.53

TABLE IV
SIMULATION RESULTS IN THE LEARNING DATABASE

	MSE	HIT		FPR		FNR	
U	μ	μ	σ	μ	σ	μ	σ
01	2.00e-02	98.00	4.18	2.01	4.23	1.19	1.58
02	1.81e-03	99.82	0.04	0.00	0.00	14.70	2.79
03	3.48e-04	99.96	0.04	0.00	0.00	2.85	3.21
04	3.96e-04	99.96	0.05	0.01	0.02	2.52	3.00
05	2.09e-04	99.98	0.03	0.00	0.00	1.63	1.91
06	6.53e-04	99.93	0.06	0.02	0.02	3.96	3.78
07	1.53e-03	99.85	0.09	0.00	0.00	12.48	7.48
08	2.82e-04	99.97	0.06	0.00	0.00	2.41	4.99
09	3.72e-04	99.96	0.08	0.01	0.03	2.30	4.67
10	3.18e-04	99.97	0.06	0.00	0.00	2.59	4.59

False positive (FP) authentications occur when a false signature is classified as authentic, and false negative (FN) authentications occur when an original signature is classified as not authentic. The Figure 7(a) shows a *boxplot* graph of the learning rate (HIT) showed in Table III.



(a) HIT

(b) FNR

Fig. 7. Boxplot graph for the experiments using the complete feature set

Through the results showed in Table III, we notice that the correct authentication rate (HIT) is very high. These results are somewhat expected, since for each fold containing 244 examples, 241 patterns are of the class 0 (non authentic) and only 3 are examples of the class 1 (authentic signatures). If the neural network always answers with 0 for all the patterns in validation set, we can achieve a correct authentication rate of 98.77%. So, we expect the HIT rate to be at least superior to this value, in order to really consider that the learning was successful. In our case, a more reliable measure used to analyze the results is the false positive rate (FPR) and false negative rate (FNR), that are calculated through the following equations:

$$FPR = \frac{N_{FP}}{N_{cl0}} \times 100 \qquad (7)$$

$$FNR = \frac{N_{FN}}{N_{cl1}} \times 100 \qquad (8)$$

where N_{FP} is the number of false positives, N_{FN} is the number of false negatives, N_{Cl0} is the number of patterns of the class 0 and P_{Cl1} is the number of patterns of the class 1. The Figure 7(b) shows a *boxplot* graph of the false negative rate (FNR) for the Table III experiments. Except for the user 01, for all other users the FPR was very small (less than 0.1%), but the FNR was high, among 2.33% and 25.33%. In an on-line signature authentication system, the FPR must be as low as possible, even if it increases the FNR a little bit. This is because we are able to request to the user to repeat the signature, and this is less harmful than to accept a false signature. A FNR greater then 20% is also not acceptable, because in this case there will be too many authentic signatures rejected.

According to Haykin[17], when a neural network have too many free parameters (connection weights), and if the learning database doesn't have a sufficient number of examples, an overfitting problem will then occur. When this problem occurs, even a cross-validation method can't avoid it. This happens when the learning and validation databases are very small and consequently don't contain sufficient data variance.

In our case, we have a reduced set of pattern for the class 1 (just 30 signatures per user), and a large number of free parameters in the network (118 weights), thus the neural network tends just to decorate the training dataset patterns. In order to avoid this problem, we have the following possibilities[17]:

[4] Processor AMD Athlon XP 1.54GHz, 512MB of RAM Memory

- Reduce the input space dimensionality using PCA;
- Force some of the synaptic weights to take values close to zero using a weight decay method;
- Reduce the network complexity using some network pruning technique;

Thus, we chose to use input space dimensionality reduction with PCA and also to use a small weight decay term. We verified through several experiments that if the weight decay term goes larger than 1.0×10^{-36}, the neural network always answers with 0 for all patterns, (weight decay seems to be stronger than the error adjustment), and thus we adopted a weight decay term equal to 1.0×10^{-38} in the following experiments.

B. Principal Components Experiments

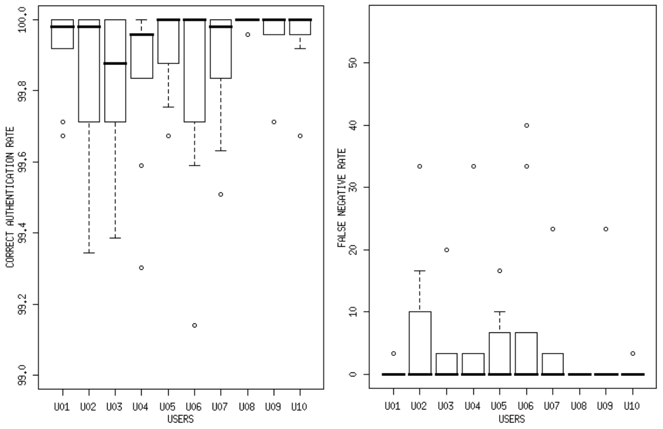
The Table V shows the results obtained in the generalization test database for the same users described previously, but using input space dimensionality reduction and a weight decay term equal to 1.0×10^{-38}. The neural network used in these experiments had 40 neurons in the input layer and one neuron in the output layer. The neural network used the same parameters as indicated in the earlier experiment, except for weight decay term. The amount of time necessary to simulate the Table V experiments was 43.33 minutes in the same typical computer. Each individual run (1000 epochs) takes about 2.6 seconds.

TABLE V
SIMULATIONS USING INPUT SPACE DIMENSIONALITY REDUCTION

	MSE	HIT		FPR		FNR	
U	μ	μ	σ	μ	σ	μ	σ
01	7.68e-04	99.92	0.12	0.07	0.13	0.33	1.05
02	1.63e-03	99.82	0.25	0.10	0.21	6.33	11.05
03	1.47e-03	99.83	0.19	0.14	0.18	3.00	6.18
04	1.37e-03	99.85	0.23	0.06	0.09	7.33	13.77
05	7.52e-04	99.92	0.12	0.04	0.08	3.33	5.88
06	1.47e-03	99.84	0.29	0.06	0.13	8.00	15.33
07	1.04e-03	99.88	0.18	0.06	0.08	5.00	9.72
08	4.09e-05	100.0	0.01	0.00	0.01	0.00	0.00
09	5.33e-04	99.94	0.12	0.03	0.09	2.33	7.38
10	4.21e-04	99.95	0.10	0.04	0.10	0.67	1.41

We observed in the Table V that the FNR was significantly reduced, and the general mean was 3.63% (mean of all rows of the FPR μ column). The Figure 8(a) shows a *boxplot* graph of the learning rate (HIT) and the Figure 8(b) shows a *boxplot* graph of the false negative rate (FNR) for the Table V experiments.

Comparing the results of these two experiments described in tables III and V, and observing the *boxplot* graphics of Figures 7 and 8, we can clearly notice that a better performance was achieved using PCA. Besides that, the correct authentication rate (HIT) in this second experiment had a smaller variability, what demonstrates that it was less sensitive to small variations in the dataset. Analyzing the false negative rate (FNR) of the two experiments, we noted that the input space dimensionality reduction and a little weight decay term are capable to improve the generalization rate of the system.



(a) HIT

(b) FNR

Fig. 8. Boxplot graph for the experiments using a reduced input space

The Table VI shows in details the results obtained in the test generalization database for each fold simulation of the user 03 (detailing the results of the same user presented in Table V). In this table, each line represents the mean value for the experiments accomplished for each fold, and the columns represents the HIT, FPR and FNR rate as in the Table V columns. The worst result was obtained in the fold 07, that had 20% of false negative rate (FNR).

TABLE VI
DETAILED RESULTS FOR ALL FOLDS OF THE USER 03

	MSE	HIT		FPR		FNR	
F	μ	μ	σ	μ	σ	μ	σ
01	0.00e+00	100.0	0.00	0.00	0.00	0.00	0.00
02	5.53e-03	99.39	0.29	0.58	0.29	3.33	10.54
03	0.00e+00	100.0	0.00	0.00	0.00	0.00	0.00
04	8.56e-04	99.92	0.17	0.04	0.13	3.33	10.54
05	6.17e-04	99.88	0.20	0.12	0.20	0.00	0.00
06	2.05e-03	99.80	0.22	0.21	0.22	0.00	0.00
07	2.03e-03	99.71	0.28	0.04	0.13	20.00	17.21
08	2.41e-03	99.71	0.28	0.29	0.28	0.00	0.00
09	1.21e-03	99.88	0.20	0.08	0.17	3.33	10.54
10	1.20e-07	100.0	0.00	0.00	0.00	0.00	0.00

The Figure 9 shows the evolution of the MSE error in the training dataset (doted lines) and the validation dataset (solid lines) during training in one of the accomplished experiments. In this experiment, we can notice that the best generalization cycle was about 400 cycles, and after that the MSE error for the training database continues to decrease, but in the the MSE error for the validation database begins to increase. This behavior is because the neural network begins to specialize in specific characteristics of the training dataset, that are not present in the validation dataset. In all experiments, the learning was simulated during 1000 cycles, and the synaptic weights of the best generalization cycle were saved, and later they were used in the final generalization test.

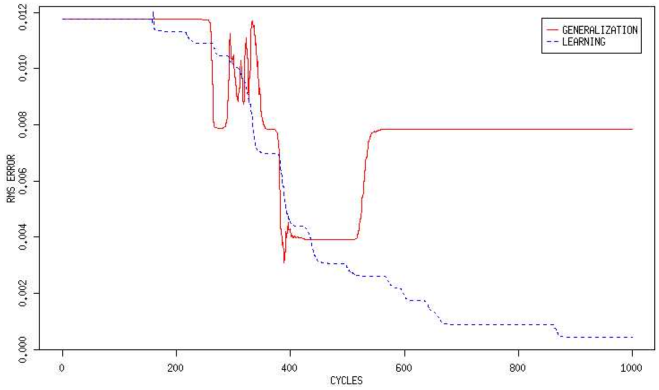


Fig. 9. Progress of the MSE error

VI. Conclusions

The main goal of this work was the study, research and development of a signature authentication system that should be able to authenticate users based on handwritten signatures. In order to reach this objective an extensive study of several topics related to pattern recognition, signature authentication, machine learning and artificial neural networks was accomplished. Our approach to deal with this problem was described and we implemented an on-line signature authentication system prototype, whose implementation use principal component analysis to reduce the input space dimensionality and artificial neural networks for the authentication task. The system prototype is complete and operational, and it was used to validate our approach and to evaluate the authentication system performance.

The results obtained in the simulations using our system prototype showed that this kind of on-line signature authentication system is not only viable, but it is also a very good solution to improve authentication security in information systems. The obtained results also proved that the neural networks are very suitable to be used in signature authentication tasks. The signature features we proposed and selected were very effective allowing to obtain a proper signature classification system. The selected input features and the input space dimensionality reduction allowed the neural network to obtain high learning rates and a very good generalization level, resulting in low incorrect signature authentication rates. This high accuracy presented by the system is fundamental to provide a really secure signature authentication system.

Our future research work is being directed to improve the learning task, using Radial-Basis Function Networks (RBF) and Support Vector Machines[17]. Besides that, our future research work is also being directed to hybrid authentication systems. The hybrid systems, through the combination of different techniques (e.g. signatures, fingerprints, eye scanning, passwords, face recognition and voice), can improve authentication systems performance to a virtually unbreakable level of security.

References

[1] Z. Riha and V. Matyas, “Biometric authentication systems,” FI MU Report Series, Technical Report RS-2000-08, Nov. 2000.

[2] G. Gupta and A. Mccabe, “A review of dynamic handwritten signature verification,” James Cook Univ., Townsville, Australia, Technical Report, Nov. 1997.

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

[4] A. K. Jain, F. D. Griess, and S. D. Connell, “On-line signature verification,” *Pattern Recognition*, Jan. 2002.

[5] D. Y. Yeung, H. Chang, Y. Xiong, S. George, R. Kashi, T. Matsumoto, and G. Rigoll, “Svc2004: First int. signature verification competition,” in *Proc. 1st Int. Conf. Biometric Authentication (ICBA)*, ser. LNCS, vol. 3072. Hong Kong, China: Springer, July 2004, pp. 16–22.

[6] K. Huang and H. Yan, “Off-line signature verification based on geometric feature extraction and neural network classification,” *Pattern Recognition, V30, N1*, pp. 9–17, 1997.

[7] H. Baltzakis and N. Papamarkos, “A new signature verification technique based on a two stage neural network classifier,” *Engineering applications of Artificial intelligence 14*, pp. 95–103, Sept. 2000.

[8] M. R. Heinen, “Autenticação on-line de assinaturas utilizando redes neurais,” Final Undergraduate Dissertation, Universidade do Vale do Rio dos Sinos (UNISINOS), São Leopoldo, RS, Brazil, 2002.

[9] M. R. Heinen and F. S. Osório, “Biometria comportamental: Pesquisa e desenvolvimento de um sistema de autenticação de usuários utilizando assinaturas manuscritas,” *Infocomp: Revista de Ciência da Computação*, vol. 3, no. 2, pp. 32–37, Nov. 2004.

[10] ——, “NeuralSignX: Sistema neural para a autenticação de assinaturas,” *Hífen*, vol. 28, no. 54, pp. 103–108, Dec. 2004.

[11] ——, “Autenticação de assinaturas utilizando algoritmos de aprendizado de maquina,” in *Anais do V ENIA*, São Leopoldo, RS, Brazil, July 2005.

[12] R. Abbas, “Backpropagation networks prototype for off-line signature verification,” Minor Thesis, RMIT – Department of Computer Science, Melbourne, Australia, Mar. 1994.

[13] A. M. Namboodiri, S. Saini, X. Lu, and A. K. Jain, “Skilled forgery detection in on-line signatures: A multimodal approach,” in *Proc. 1st Int. Conf. Biometric Authentication (ICBA)*, ser. LNCS, vol. 3072. Hong Kong, China: Springer, July 2004, pp. 505–511.

[14] T. Mitchell, *Machine Learning*. New York: McGrall-Hill, 1997.

[15] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[16] D. Rumelhart, G. Hinton, and R. Williams, *Learning Internal Representations by Error Propagation*. Cambridge, MA: MIT Press, 1986.

[17] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 1999.

[18] M. Riedmiller and H. Braun, “A direct adaptive method for faster backpropagation learning: The RPROP algorithm,” in *Proc. IEEE Int. Conf. Neural Networks (ICNN)*, San Francisco, CA, Mar. 1993, pp. 586–591.

[19] F. S. Osório, “Inss: Un système hybride neuro-symbolique pour l’apprentissage automatique constructif,” Doctoral Thesis, INPG/IMAG, Grenoble, France, 1998.

[20] R. Cattell, *Factor Analysis*. New York: Harper Books, 1952.

[21] E. Oja, “Principal components, minor components and linear neural networks,” *Neural Networks*, vol. 5, pp. 927–935, Mar. 1992.

[22] K. K. Lau, P. C. Yuen, and Y. Y. Tang, “Directed connection measurement for evaluating reconstructed stroke sequence in handwriting images,” *Pattern Recognition*, vol. 38, no. 3, pp. 323–339, Mar. 2005.

[23] H. Lei and V. Govindaraju, “A comparative study on the consistency of features in on-line signature verification,” *Pattern Recognition Letters*, vol. 26, no. 15, pp. 2483–2489, Nov. 2005.

[24] K. Yu, Y. Wang, and T. Tan, “Writer identification using dynamic features,” in *Proc. 1st Int. Conf. Biometric Authentication (ICBA)*, ser. LNCS, vol. 3072. Hong Kong, China: Springer, July 2004, pp. 512–518.

[25] M. Wirotius, J. Y. Ramel, and N. Vincent, “New features for authentication by on-line handwritten signatures,” in *Proc. 1st Int. Conf. Biometric Authentication (ICBA)*, ser. LNCS, vol. 3072. Hong Kong, China: Springer, July 2004, pp. 577–584.

[26] F. S. Osório, “Um estudo sobre reconhecimento visual de caracteres através de redes neurais,” Master’s Thesis, Univeridade Federal do Rio Grande do Sul (UFRGS), Porto Alegre, RS, Brazil, 1991.