

CS473 Practicum in Artificial Intelligence, Fall 2003.

# **Signature Based Authentication**

Final Report

Sam Krasnik, Ron Hose. (smk72,rh228)

Cornell University

## **1. Introduction**

Currently, the viable authentication mechanisms in use are password or PIN based. A user who wishes to be authenticated enters a short string or a number which is supposedly easy to remember, but is easy to forget and the authenticating machine checks the hash of the password against a stored hash in a database. If a user were to forget his or her password or it was stolen by someone wishing to steal the user's identities, the identity of the user becomes endangered and easy to compromise.

A common solution to the problem is to implement a biometric authentication mechanism such as retina scans or fingerprint based authentication. The advantage of this mechanism is that a retina or a fingerprint cannot be forgotten and it is unique to each user. While such a system is extremely difficult to compromise, The cost of implementation restricts its deployment. In addition, if such biometric data was in fact compromised, the result would be disastrous for the user-- he or she cannot change fingerprints or retinas.

To solve both the problems of password based authentication (relatively easy to compromise, easy to forget) and biometric authentication (expensive, intrusive, and disastrous if compromised), we attempt to solve the problem of user authentication via Handwritten Signature Verification (HSV). The hardware necessary for HSV is inexpensive and already widely deployed in (e.g. credit card machines in stores, tablet PCs)

The problem consists of verifying a user's identity based on her signature by requiring the user to enter a signature and comparing the signature to a set of previously entered signatures. The system should have a very low false acceptance rate and a pessimistic false rejection rate. Our basic approach to solving the HSV problem uses on-line back-propagation. Both local and global features of the input signatures are used to train an Artificial Neural Network (ANN) to authenticate the user.

Some related work in the field uses only off-line data, while others also use on-line data. Some use only global or only local features, while others use a combination. The work presented is one of the most general approaches to the problem.

Our results indicate that the back-propagation algorithm was able to converge on more than 90% of user data and achieved a pessimistic False Rejection Rate of around 30% and a low False Acceptance Rate of 2%. We conclude that the solution presented is successful in solving the Handwritten Signature Authentication Problem.

## **2. Problem Definition and Algorithm**

### **2.1 Task Definition**

Initial input to the system consists of a set of valid signatures by the user in question. The system uses the user's signatures to learn to verify future signatures by the user. The initial output consists of a network that is saved to use for future authentication.

Input to the system consists of a (user signature, login identity) pair. The login identity is entered manually or chosen from a list of users on the system. The user signature is entered using a pen or a mouse device, and is recorded as a set of points on a grid, along with temporal data recording the pen's movement during the signature. Output produced by the system is in the form of a Boolean answer signifying whether or not the user has been authenticated based on his signature and login identity.

## 2.2 Algorithm Definition

### Training:

A Multi-layer layer neural network trained with back propagation with momentum and bias was used to classify the authentication attempt of a user as being valid or invalid. Input to the network is a user signature codified as a real numbers. The output of the ANN is 2 real numbers in the range 0.0 to 1.0 with a threshold function applied to determine the authentication status of the user. The neural network is initially trained against a set of n valid signatures given by the user when he or she is first introduced to the system as well as a set of m invalid signatures. Before each signature is inputted to the ANN for training, a Gaussian noise is applied to the data to increase the robustness of the trained network.

The back-propagation algorithm works as follows:

1. The input data is presented to the network as a set of real numbers. The input is propagated to the output as standard feed forward network network. At each node, the output of the node is the sum of the inputs to that node multiplied by a weight for that input and passed through a biased sigmoid squashing function. The squashing function is computed as follows:

$$\sigma(x, \eta) = \frac{1}{[1 + \exp(-x + \eta)]}$$

The inputs to a non-input node are the outputs of the previous layer of nodes.

2. After the output is computed, the errors at each node are calculated as follows:

For output nodes:

$$\beta_i = (\text{Desired}_i - \text{Output}_i) * \sigma'(\text{Output}_i)$$

For hidden nodes:

$$\beta_{ij} = \beta_i * W_{ij} * \sigma'(\text{Output}_{ij})$$

3. The weights in the network are updated as follows:

For output nodes:

$$\Delta W_i = \sum_j \alpha * \beta_i * \text{Output}_{ij} + \mu * \Delta \text{OldOutput}_{ij}$$

For hidden nodes:

$$\Delta W_{ij} = \sum_k \alpha * \beta_{ij} * \text{Input}_k + \mu * \Delta \text{OldOutput}_{ij}$$

4. Finally, the biases for the output and hidden nodes are updated as follows:

For output nodes:

$$\Delta \eta_i = \sum_j \alpha * \beta_{ij}$$

For hidden nodes:

$$\Delta \eta_i = \sum_k \alpha * \beta_{ik}$$

The constants involved are:

$\alpha$  – the learning rate, set to 0.3

$\mu$  – the momentum rate, set to 0.25

$\tau$  – the minimum tolerance, set to 0.03

$\phi$  – the Gaussian noise applied, set to 0.1

$\rho$  – the maximum number of training rounds, set to 500

The topology of the network is as follows:

85 Input Nodes

40 Hidden Nodes

2 Output Nodes

The input to the network consists of a set of features extracted from the user's signature. The features used were a combination of both global features and features specific to the particular signature. The global features consist of:

1. The total X arc length
2. The total Y arc length
3. The number of sign changes in the X velocity
4. The number of sign changes in the Y velocity
5. The number of disjoint strokes in the entire signature

The local features consists of the X(t) and Y(t) derivatives, optionally passed through a Fast Fourier Transform, in which case the coefficients were used as input. This temporal input consists of 40 inputs for each of the X(t) and Y(t) derivatives.

To train the network, 5-10 user signatures are used as positive examples and 1-2 signatures from 10-15 other users are used as negative examples. If the MSE of the network reaches the minimum tolerance,  $\tau$ , then the training stops and it is said to converge. If after  $\rho$  steps, the network has not converged, it is discarded and must be retrained with either a larger  $\rho$  or different data.

## **Authentication:**

Authentication of a user proceeds as follows:

A user login and signature are entered; The signature is codified with the features described and fed forward through the network that has been trained for that user. The output of network consists of two real numbers, one to indicate how “good/positive” the signature was, and one to indicate how “bad/negative” the signature was. If both values exceed an acceptance threshold, the user is deemed authenticated. The thresholds used were 0.9 for the positive output and 0,1 for the negative output.

## **3. Experimental (or Theoretical) Evaluation**

### **3.1 Methodology**

Signature data was collected from 30 participants. Each participants were asked to sign their name ten times as consistently as possible. Most participants have not used a tablet PC before, which may have effected the consistency of their signature. Some participants expressed concerns about submitting their signature, and were allowed to sign a word instead. Some of the samples collected are in non-roman scripts (Hebrew, Thai).

To test the performance of different network configurations, a network was trained on 6 randomly selected signatures from each user. 15 randomly selected signatures from 15 other unique participants were used as negative examples. The network was then tested on 3 of the user's signatures that were not used in the training phase. The process was repeated 5 times for each user (each time selecting a different subset), and results were gathered.

The above experiment was repeated with fast Fourier transformations and Derivatives. Both were tested with and without noise generation. Other features of the network were tested by experimentation but were not carried to completion since performance trends such as convergence and prediction accuracy were easy to identify empirically.

### **3.2 Results**

What follows is a table summarizing the results obtained by training the network on all 30 users. In the table, the Derivs test used derivatives for local features. The With Global test utilizes the additional global features. The Non Skewed test uses only one negative example per user, while the others use all of a users. The With Noise test applies a noise to the input before training. The FFT test passes the input through an FFT before presenting it to the network.

The positive error is the error of the two nodes when tested against positive examples. The negative error is the error of the two nodes when tested against negative examples. The FRR is the false rejection rate-- the rate of rejection of the user's signatures. The

FAR is the false acceptance rate-- the rate of acceptance of other user's signatures. While it would be advantageous to also provide FAR data for skilled forgeries, this data was not available since we were not able to obtain a sufficient amount of such forgeries, so the values for FAR in the table can be considered to be those for zero-knowledge forgeries, i.e. those which were made with no knowledge of the user's signature or login identity.

<i>Error\Features</i>	<i>Derivs</i>		<i>With Global</i>		<i>Non-Skewed</i>		<i>With Noise</i>		<i>FFT</i>	
<b>Positive Error</b>	0.310	0.317	0.298	0.305	0.303	0.311	0.226	0.237	0.2430	0.2640
<b>Negative Error</b>	0.317	0.311	0.223	0.210	0.299	0.287	0.316	0.308	0.2440	0.2440
<b>FRR</b>	0.00%		40.20%		29.23%		25.00%		26.53%	
<b>FAR</b>	5.78%		1.83%		3.44%		4.04%		2.09%	
<b>% Converging</b>	0.00%		83.10%		93.10%		90.00%		64.60%	

For all tests in the above table, the accept threshold was set to 0.9 and 0.1 for the two output nodes, respectively. Convergence was judged based on the standard value of  $\rho$ .

As can be seen from the table, the best FRR was achieved using non-skewed data with noise; the best FAR was achieved using non-skewed data without noise, with FFT close behind. What is interesting to note is that the Negative Error for these two tests were very similar. Finally, it is important to note that using global features significantly reduced the Negative Error and more importantly significantly reduced the FRR in the last three tests.

### 3.3 Discussion

From the results presented in the previous section, we conclude that the solution succeeded in solving the HSV problem. The FAR was indeed low and the FRR was pessimistic in authenticating users. Arguably, the FAR is not low enough, because in high-security environment, accepting 1 forgery in 100 is not sufficiently secure. However, in a password based system, if an onlooker were to see what the user typed in, the FAR would be 100%; the onlooker simply needs to enter the same password. In the case of HSV, the onlooker would have no knowledge of the signature the user uses to authenticate, even if he were to look, since a secure implementation of the interface would not show what the user is writing, so 1 in 100 is sufficient.

The first improvement due to the use of global features can be explained by the fact that these global features eliminate very different signatures from even approaching acceptance, which yields a much lower FAR.

The use of non-skewed data, while not significantly different in positive error and

negative error not significantly different from the the Derivs test, does differ significantly in the FRR. The negative error is higher than the Global test, but this can be understood since fewer negative examples were used. However, by balancing the amount of positive and negative examples, the network is no longer as noisy with respect to the output on positive examples, so the FRR decreases.

Adding Noise further reduced the FRR, since the network is now able to accept a more general class of signatures.

Finally, transforming the input with an FFT decreased both the FRR and FAR resulting in the lowest overall error. However, its convergence rate is fairly low, much lower than the rest of the methods. This particular behavior could be due to the fact that frequency data is difficult to differentiate in common roman script without more complex features.

The results presented in the previous section are much more pessimistic in the FRR compared to other methods. While this may be seen as unwanted behavior, in a secure environment, a low FAR is much more important than a low FRR. This ability to control the FRR via two thresholds is an advantage of our system compared to others.

A disadvantage of the presented system is the lack of more complex features such as more accurate time and pressure data. Unfortunately, the time data in the example database only provides information about the sequence of events, as opposed to their exact time, which does not allow the use of such potentially important features such as pen up/down time ratio. Also, due to the limitations of the testing hardware, we were not able to utilize pressure data as originally proposed, and in the literature, pressure data is considered important in distinguishing forgeries from authentic signatures.

#### **4. Related Work**

Other work in the field consists of both on-line and off-line HSV systems. Since the system presented is on-line, the features that are used are mostly temporal, so it does not have much in common with off-line systems. Other on-line systems utilize a larger variety of features, especially those that have access to data such as pressure and tilt. Also, instead of transforming the input with FFT, some systems use various Wavelet transforms.

The problem itself is common to all the systems, except that in many other systems, forgery data was readily available to train against. In our system, forgeries were only available for testing. The greatest difference between our system and other systems in the field is the ability to change the acceptance threshold and hence directly change the inherent security of the system. Other systems had much more symmetric FAR/FRR values, while our system is able to vary in security depending on the situation.

#### **5. Future Work**

A major obstacle in improving prediction quality is the lack of quality negative examples. In our system, we use other user's signatures as negative data, however these

may not produce the required fidelity for a secure authentication method. In an ideal world, we would like to train against signature forgeries to lower false acceptance rate, while keeping false rejection rate low.

Modern Tablet interfaces provide additional data such as pressure and tilt that can be used for distinguishing a users signature style which can forgeries more difficult. These features were not supported by our hardware. Furthermore, the time data collected only provides sequence information and not the actual time of the points, which does not allow for more complicated temporal features.

Increasing the number of positive examples used to train the network may also improve prediction. We have observed that skewing the distribution of pos/neg examples to include significantly more negative examples has sharply reduces FAR, at the cost of an increase FRR, since the MSE is dominated by the negative examples. It is possible that adding yet more positive example may help lower FRR without greatly affecting FAR.

Another approach for improving classification is better selection of signature data used to represent the initial network. Since data is collected from the user, the system can prompt the user for more signatures than necessary, and attempt to find the best subset of these signatures to provide secure authentication. If none of the subsets provides a secure network (e.g. the user may have entered an 'easy' signature), the software should prompt the user to provide a better signature. This is analogous to password selection systems which verify the quality of the password before allowing it.

## **6. Conclusion**

We have introduced a system for handwriting signature verification that can be implemented even on hardware with that collects a limited amount of features. We have found that a combination of global and local features yields the best error rates. We have also found that adding noise to the input before training makes the resulting system more robust. Transforming the input before training yields much lower error, but is more sensitive. Most importantly, we have presented system can vary in security depending on the situation.

Uses for such a system range from securing a credit card transaction at the point of sale to user authentication on tablet PCs. While our system does is not 100% resilient to forgery, it offers significant additional protection, especially if used in conjunction with other authentication methods. The results of our experiments indicate that many more features and tweaks can be exploited before we reach the theoretical bounds of prediction quality.

We hope that this system will help future research in creating variable security HSV systems as well as systems which can select feature sets which are optimal for a specific user.

## **7. References:**

- [1] K. Gurney. "An introduction to neural networks", UCL Press, 1997.  
<http://www.shef.ac.uk/psychology/gurney/notes/>
- [2] R. Abbas. "Backpropagation Networks Prototype For Off-Line Signature Verification ", MIT 1994  
<http://citeseer.nj.nec.com/161981.html>
- [3] S. Dennis, D. McAuley. "Neural Networks By Example: An Introduction to Neural Networks Using the BrainWave Simulator", Univ. of Queensland.  
<http://www2.psy.uq.edu.au/~brainwav/Manual/BackProp.html>
- [4] C. Stergiou, D.Siganos. "Neural Networks". Imperial College, London.  
[http://www.doc.ic.ac.uk/~nd/surprise\\_96/journal/vol4/cs11/report.html](http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html)
- [5] D. A. Mighell, "Backpropagation and its application to handwritten signature verification", Advances in Neural Information Processing Systems I, pp. 340-347, 1989.
- [6] Gupta, J and McCabe, A. "A Review of Dynamic Handwritten Signature Verification", James Cook University, Australia (1997).  
<http://citeseer.nj.nec.com/gupta97review.html>
- [7] T. Ohishi, Y. Komiya, T. Matsumoto. "On-Line Signature Verification Using Pen-d Position, Pen-Pressure and Pen-Inclination", Trajectories , Waseda University  
<http://www.computer.org/proceedings/icpr/0750/Volume%204/07504547abs.htm>
- [8] Y. Sato and K. Kogure, "On-line signature verification based on shape, motion and handwriting pressure", Proc. 6th Int. Conf. on Pattern Recognition, Vol. 2, pp 823-826, Munich, 1982.