To appear in —
"International Journal of Pattern Recognitn and Artificial Intelligence"

Signature Verification using a "Siamese" Time Delay Neural Network

Jane Bromley James W Bentz* Léon Bottou Isabelle Guyon

Yann LeCun Cliff Moore* Eduard Säckinger Roopak Shah *

July 1992

Abstract

This paper describes the development of an algorithm for verification of signatures written on a touch-sensitive pad.

The signature verification algorithm is based on an artificial neural network. The novel network presented here, called a "siamese" time delay neural network, consists of two identical networks joined at their output. During training the network learns to measure the similarity between pairs of signatures. When used for verification, only one half of the Siamese network is evaluated. The output of this half network is the feature vector for the input signature. Verification consists of comparing this feature vector with a stored feature vector for the signer. Signatures closer than a chosen threshold to this stored representation are accepted, all other signatures are rejected as forgeries.

System performance is illustrated with experiments performed in the laboratory.

Keywords — dynamic signature verification, artificial neural networks, time delay neural network, touch-sensitive pads

*The authors are with AT&T Bell Laboratories, Crawford Corner Road, Holmdel, NJ 07733 and *NCR Corporation E&M Atlanta, 2651 Satellite Boulevard, Duluth, GA 30136.

1 Introduction

NCR, a division of AT&T, has been developing signature capture devices. These are touch-sensitive pads which provide positional coordinates as a function of time for a pen moving over the unit's writing surface. NCR plans to use these devices to automate credit-card transactions. The device will be sited at the point of sale and used, instead of credit-card slips and carbon copies, to electronically record signatures. Information from the transaction, such as the amount of the purchase and the person's signature, can be kept in an electronic file.

When a signature is electronically captured it could also be automatically verified, therefore, as part of this automation process NCR also wishes to provide signature verification. Credit card companies have large losses due to fraud. (The credit card companies, Mastercard and Visa combined, lost $450 million dollars in 1991, 7.7% of their pretax profits. Their major losses come from late/non-payment and bankruptcies, which totalled $7.92 billion [1]). Signature verification can provide a method to check on the identity of the signer and hence help to reduce these losses.

The aim of the project reported here was to make a signature verification system based on the NCR 5990 Signature Capture Device and to use 80 bytes or less for signature feature storage in order that the features can be stored on the magnetic strip of a credit-card. Verification using a digitizer such as the 5990, which generates spatial coordinates as a function of time, is known as dynamic verification.

1.1 NCR Requirements

Signature verification works by comparing a sample signature signed on a 5990 Signature Capture Device with a model of that person's signature. This model may be stored in a local file, on a credit card or SMART card, or in a central database.

NCR specified a number of requirements that the signature verification algorithm must fulfill:

- 99.5% of genuine signatures must be accepted while detecting 80% of forged signatures.
- The algorithm should be tunable so that the ratio of the percentage of genuines accepted to forgeries

detected can be raised or lowered.

- The model of a person's signature should be about 80 bytes in size.
- The model should become a more accurate representation of a person's signature with each successful use of the card.

Further requirements with respect to speed and cost of the verification system were made. They are not addressed in this paper.

2 Data

All signature data was collected using 5990 Signature Capture Devices. This device consists of an LCD with a transparent digitizer. As a guide for signing, a 1 inch by 3 inches box is displayed on the LCD. However all data captured both inside and outside this box, from first pen down to last pen up, is returned by the device. The 5990 provides the trajectory of the signature in Cartesian coordinates as a function of time. It also uses a pen pressure measurement to report whether the pen is touching the writing screen or is in the air, hence information is provided, not only about the motion of the pen on the touch-sensitive screen, but also for the pen in the air (within a certain proximity of the pad). I.e: both the trajectory of the pen on the pad and of the pen above the pad is recorded. Forgers usually copy the shape of a signature. Using such a touch sensitive pad for signature entry means that a forger must also copy dynamic information and the trajectory of the pen in the air. Neither of these are easily available to a forger and it is hoped that capturing such information from signatures will make the task of a forger much harder. Strangio [2], Herbst and Liu [3] have reported that pen up trajectory is hard to imitate, but also less repeatable for the signer. The spatial resolution of signatures from the 5990 is about 300 dots per inch, the time resolution 200 samples per second and the pad's surface is 5.5 inches by 3.5 inches. Performance was also measured using the same data treated to have a lower resolution of 100 dots per inch. This had essentially no effect on the results.

2.1 Data Sets

We wished to collect as many genuine signatures from each person as possible without driving our donors away from boredom. The initial data collection was made by Bell Labs and NCR researchers at cafeterias in Bell Labs, Holmdel, N. J. and NCR, Cambridge, Ohio and each genuine author was asked to sign 10 times. This data contained 1115 genuine signatures collected from 120 authors and also 521 forgeries by 62 people of 43 of these genuine signatures. Later data was collected from students at Wright State University. During the first collection, it was remarked that it took a few signatures before people felt comfortable signing on the pad. Since most people had signed 10 signatures without complaint, we decided to ask for 20 signatures from each genuine author. 47 students signed 897 genuine signatures and 507 forgeries by 29 people of 18 of these signatures were also collected.

When producing forgeries, the signer was shown an example of the genuine signature on a computer screen. The amount of effort made in producing forgeries varied. Some people practiced or signed the signature of people they knew, others made little effort. Hence, the quality of the forgeries varied from undetectable to obviously different. Skilled forgeries are the most difficult to detect, but in real life a range of forgeries occur from skilled ones to the signatures of the forger themselves. Our database consists of reasonably accurate forgeries, however by using genuine signatures from people other than the original signer we can also measure performance for these so-called zero-effort forgeries (also known as random forgeries).

Except at Bell Labs., the data collection was not closely monitored so it was no surprise when the data was found to be quite noisy. It was cleaned up according to the following rules:—

- Genuine signatures must have between 80% and 120% of the strokes of the first signature signed and, if readable, be of the same name as that typed into the data collection system. (The majority of the signatures were donated by residents of North America, and, typical for such signatures, were readable.) This was to remove signatures for which only some part of the signature was present or where people had signed another name e.g. Mickey Mouse.

- Forgeries must be an attempt to copy the genuine signature. This was to remove examples where people had signed completely different names. They must also have 80% to 120% of the strokes of the signature.

- A person must have signed at least 6 genuine signatures or forgeries.

After cleaning, the first data set consisted of 982 genuine signatures for 102 people and 370 forgeries of 34 different signatures by 50 people. The corresponding figures for the second data set were 788 genuine signatures from 43 people and 418 forgeries of 14 signatures by 24 different people.

3 Neural Network Architectures

Very good performance for dynamic signature verification, well past the requirements set out for this project, can be achieved if no constraints are placed on the verification algorithm. Plamondon and Lorette [4] provide a review of current achievements in automatic signature verification. However, once limitations are set, the task becomes much harder. Particularly restricting is the requirement that the model of a person's signature fit within 80 bytes. A person typically takes around 4 seconds to sign their name. For a signature written on the 5990 this gives 800 sets of x,y and pen up-down (pud) points. In order for a person's signature to be represented in 80 bytes a large compression must be made, while retaining features of the signature that allow it to be discriminated from forgeries. This need to compress a signature and the observation that with practice, humans can learn to discriminate forgeries from genuine signatures, led us to consider the use of artificial neural network techniques. Neural networks can learn from examples and they can also be used to compress data. The verification process decides on the claimed identity of an author by a comparison process. This work is based on a neural network architecture, called a siamese network, which first extracts features from two signatures and then evaluates the distance between these two sets of features. A similar architecture was independently proposed for fingerprint identification[5].

The network has two input fields to compare the two patterns and one output whose state value corresponds to a measure of the distance between the two patterns. Two separate sub-networks, based on Time Delay Neural Networks (TDNN)[6] [7] act, one on each input pattern, to extract features, then the cosine of the angle between the two feature vectors is calculated and this represents the distance value. (The cosine distance measure was used rather than the euclidean distance. Using the euclidean distance and requiring 2 genuine signatures to have a small distance could lead to the trivial solution of zero size feature vectors.)

The sub-networks are constrained to be identical, they are multi-layer and feed-forward with several layers of feature extraction before the distance measure. TDNN's use processing units having inputs that are delayed in time. In other words, a unit receives inputs from just a portion of the signature trajectory. In a multi-layer TDNN, successive layers of units can extract features spread over a wider range of time.

The connections obey the following rules:

- processing units have fields of view which are limited in the time direction,
- the outputs of different units, which are members of the same field of view, are connected to one feature vector in the next layer,
- the same set of units is repeated along the time axis: the connections between a field of view and a feature vector in the next layer and the connection between the next field of view and the next input vector share the same weights (this operation can be viewed as a nonlinear convolution over the input field).

This last rule is important when implementing the network – although it may have many thousands of connections this constrains the number of independent weights and hence the memory requirements. With suitable averaging or sub-sampling the time extent of the network can be reduced.

A number of 2 and 3 layer network architectures were investigated in a pilot study. The network described in Fig. 1 gave the best results in this pilot study. Architecture 2 (Fig 2) was developed later to meet the requirement that the model of a person's signature fit in 80 bytes.

Architecture 1 has 5,347 units, 210,272 connections, but due to the convolutional architecture, only

3,004 independent weights. Architecture 2 has 6,883 unit, 194,216 connections and 654 independent weights. To achieve compression in the time dimension, architecture 1 uses a sub-sampling step of 3, while architecture 2 uses averaging.

When the network is used in the forward direction, the x,y coordinates, pen up/down and a number of other precomputed features (see next section for a discussion of input features) in 200 time steps, are placed on the input to the network. Each processing unit sums its weighted inputs and passes this value through a transfer function to the units of the next layer. The transfer function used is a sigmoid function,

$$f(x) = tanh(2x/3)/tanh2/3$$

where $f(x)$ passes through the origin, $f(1) = 1$ and $f(-1) = -1$. These values were chosen so that the transfer function range was somewhat larger than the target range of $+1.0$ and -1.0.

All the weights of the units are learnt during the training process which uses a modified version of backpropagation[8]. To train the network, two patterns are applied one to each input and a desired value for the output is used to backpropagate the error. The desired output is for a small angle between the outputs of the two subnetworks ($f1$ and $f2$) when two genuine signatures are presented and a large angle if one of the signatures is a forgery. For the cosine distance used here

$$(f1.f2)/(|f1||f2|)$$

the desired outputs were 1.0 for a genuine pair of signatures and -0.9 or -1.0 for the second case. There is no clear choice for the desired values. Using 0.0 for the desired output of a genuine:forgery pair gave essentially the same result. The choice of desired outputs should be important for optimal network performance, but it was not a limiting factor on performance in this case and hence, no effort was made to optimize it.

4 Signature Preprocessing

The 5990 provides position and pen up or down as a function of time for signatures entered on the digitizer. These signals are sampled every 5ms and most signatures are between 300 and 1500 points in length. Signature verification performance using function-based methods have been found to lead to higher performance while parameter-based methods make a lower requirement on memory space for signature storage (see Lorette and Plamondon for comments [9]). We chose to use these complete signals, with the preprocessing described below, as input to the network, rather than a parameter-based method, and allow the network to compress the information. We also investigated verification based on eight global features and achieved similar performance figures to those reported here, but the signature model was larger than 80 bytes. We believe that it is more robust to provide the network with low level features and to allow it to learn higher order features during the training process, rather than making heuristic decisions e.g. such as segmentation into balistic strokes. We have had success with this method previously [7] as have other authors [10]. $x(t)$ and $y(t)$ are originally in absolute position coordinates. By calculating the linear estimates for the x and y trajectories as a function of time and subtracting this from the original x and y values, they are converted to a form which is invariant to the position and slope of the signature. Then, dividing by the y standard deviation provides some size normalization (a person may sign their signature in a variety of sizes, this method would normalize them). The next preprocessing step is to resample, using linear interpolation, all signatures to be the same length of 200 points. This must be done in order to train the neural network which requires a fixed input size. Next, further features are computed for input to the network and all input values are scaled so that the majority fall between $+1$ and -1. Ten different features were calculated, but a subset of eight were used as input to the network (see also Fig. 3) :

feature 0 pen up $= -1$; pen down $= +1$, pud

feature 1 x position, as a difference from the linear estimate for $x(t)$, normalized using the standard deviation of y

feature 2 y position, as a difference from the linear estimate for $y(t)$, normalized using the standard deviation of y

feature 3 speed at each point, spd

feature 4 centripetal acceleration, acc-c

feature 5 tangential acceleration, acc-t

feature 6 the direction cosine of the tangent to the trajectory at each point, cosθ

feature 7 the direction sine of the tangent to the trajectory at each point, sinθ

feature 8 cosine of the local curvature of the trajectory at each point, cosϕ

feature 9 sine of the local curvature of the trajectory at each point, sinϕ

In contrast to the features chosen for character recognition with a neural network[7], where we wanted to eliminate writer specific information, the features such as speed and acceleration were chosen to carry information that aids the discrimination between genuine signatures and forgeries. At the same time we still needed to have some information about shape to prevent a forger from breaking the system by just imitating the rhythm of a signature, so positional, directional amd curvature features were also used. The resampling of the signatures was such as to preserve the regular spacing in time between points. This method will penalize forgers who do not write at the correct speed.

5 Training the Neural Networks

Up to 1,636 of the signatures were used for training, the remaining were used for testing (see the next section). During training, part of the data set is used for learning the weights of the units, another part, called the validation set, is used after a training iteration to test the network's performance. A measure of the performance of the network at accepting genuine pairs and rejecting forgeries is found by calculating the percentage of genuine signature pairs for which the output (the cosine distance between

Network	Input Features	Best Performance on:	
		Training Set	Validation Set
1	pud acc-c acc-t spd $\cos\theta$ $\sin\theta$ $\cos\phi$ $\sin\phi$	GA 97.0% FR 65.3%, 26 passes through set	GA 90.3% FR 74.8%, 6 passes through set
2	same as 1, but pen up trajectory removed	GA 97.8% FR 60.0%, 11 passes through set	GA 93.2% FR 75.2%, 2 passes through set
3	same as 2, but first and last points of pen up trajectory included	GA 99.3% FR 82.6%, 100 passes through set	GA 93.2% FR 74.6%, 2 passes through set
4	x y pud spd $\cos\theta$ $\sin\theta$ $\cos\phi$ $\sin\phi$	GA 99.8% FR 88.8%, 100 passes through set	GA 91.7% FR 74.2%, 32 passes through set

Table 1: Summary of the first round of training. Note: the aim of removing all pen up points for Network 2 was to investigate whether the pen up trajectories were too variable to be helpful in verification.

them) was greater than 0 (genuines accepted, GA) and the percentage of genuine:forgery pairs for which the output was less than 0 (forgeries rejected, FR). During training the network goes through several iterations. The network used for the testing reported in the next section was the one with the highest GA rate on the validation set. A well-known problem with backpropagation networks is over-fitting of the data; after much training very good performance is achieved on the training set, but to the detriment of the network's generalization ability. By measuring the network's performance on an independent data set during the training, the version of the network with the best generalization performance can be found.

5.1 First Round of Training

The first round of training was conducted before any cleaning of the data, using part of the data set collected at Bell Labs and NCR. The data set was small and noisy. It contained 91 genuine signers with an average of 9 examples of each person's signature. There were about 340 forgeries for about 34 different signers. The training set consisted of 5,003 signature pairs; 50% genuine:genuine pairs (g:g), 40% genuine:forgery pairs (g:f) and 10% genuine:zero-effort pairs (g:0). The zero-effort data was made by pairing genuine signatures of two different people. The validation set consisted of 1,566 signature pairs in the same proportions as the training set. See Table 1 for a summary of the experiments. Training takes a few days on a SPARC 1+.

Network	Input Features	Best Performance on: Training Set	Validation Set
5	same as network 4	GA 98.2% FR 81.7%, 42 passes through set	GA 99.4% FR 80.5%, 42 passes through set
6	same as 5, except architecture 2 was used	GA 98.6% FR 81.5%, 69 passes through set	GA 99.6% FR 80.1%, 44 passes through set.
7	same as 6, except the amount of forgery data was increased by presenting the g:f and g:0 pairs 4 times in every iteration	GA 96.1% FR 81.0%, 20 passes through set	GA 98.1% FR 79.5%, 18 passes through set

Table 2: Summary of the second round of training Notes: For Network 5 the training simulation crashed after the 42nd iteration and was not restarted. Performance may have improved if training had continued past this point.

All these networks show the effect of over-fitting the data. They have learnt the training data by heart, but their performance on the validation set is poor. This indicates that the training set is too small. With more data the performance on the validation set will approach that on the training set.

5.2 Second Round of Training

This round of training used a larger and cleaner training set (the cleaned-up data collected at Bell Labs and NCR). It contained 982 genuine signatures from 108 signers, also 402 forgeries of about 40 of these signers. The training set consisted of 7,701 signature pairs; 50% genuine:genuine pairs (g:g), 40% genuine:forgery pairs (g:f) and 10% genuine:zero-effort pairs (g:0). The validation set consisted of 960 signature pairs in the same proportions as the training set. See Table 2 for a summary of the experiments.

6 Testing

Two rounds of testing were made. For the first round only a small test set was available. By the time of the second round more data had been collected and a larger test set was available. During testing, signatures to be verified are compared to one single model for the person's signature.

The output values from one half of the siamese network comprise the signature feature vector for the input signature. We assume that the feature vectors for each person's signatures form a multi ~~uni~~variate normal density i.e. that they are continuously-valued, mildly corrupted versions of a single prototype vector. For simplicity, we assume that the features are statistically independent, and that each feature has the same variance. During testing, a model is constructed of each person's signature by calculating the mean feature vector and the variance. (See pp22-27 of "Pattern Classification and Scene Analysis" by Duda and Hart[11] for a fuller description of this.) In the results presented here, the last six signatures signed by each person were used to make the model signature, but any number of signatures could have been used to make the model. In fact the more that are used the better the model will be. With that in mind, the following results should be viewed as a worst case value for performance.

The likelihood that a test signature is genuine, p-yes, is found by evaluating the normal density function. The likelihood of a test signature being a forgery, p-no, is assumed to be a constant value. An estimate for this value was found by averaging the p-yes values for all forgeries. Then the probability that a test signature is genuine is p-yes/(p-yes + p-no). All results are plotted showing how the percentage of genuine signatures accepted and forgeries detected vary as the threshold probability is varied. It takes 2 to 3 minutes running on a SPARC2 Sun workstation to preprocess 6 signatures, collect the 6 outputs from the half network and build the gaussian model.

6.1 First Round

This test set consisted of 63 genuine signatures and 63 forgeries for 18 different people. There were about 4 genuine test signatures for each of the 18 people, and 10 forgeries for 6 of these people.

Fig. 4 shows the results for Network 1 tested with the standard data and with the data simulated to have low resolution. The results are essentially identical showing that the verification algorithm developed for the 5990 data can be easily ported to a device with lower spatial resolution.

Fig. 5 shows the results for Networks 2 and 3, trained with little or no pen up points. For comparison the result for Network 1 on the standard test set is also shown. The graph shows that there may be some advantage to using just the first and last points as, at least in some part of the range, Network 3 has the best results. However, with such a small test set, this difference may be hardly significant.

Fig. 6 shows the results for Network 4 and for comparison the results for Network 1. This graph shows that using x and y as input features, rather than acc-c and acc-t, has improved performance in some part of the range. No study was made to find-out whether the performance improvement came from using x and y or from leaving out acc-c and acc-t. Plamondon and Parizeau [12] have shown that acceleration is not as reliable as other functions.

6.2 Second Round

This test set consisted of 532 genuine and 424 forgeries for 43 different people. There were about 12 genuine test signatures for each person, and 30 forgeries for 14 of the people.

Fig. 6 shows the results for Network 4 on this large test set, compared with performance on the smaller test set, closed circles. This result is on a much larger test set and must thus be considered a more accurate measure of the performance of Network 4.

Fig. 7 shows the results for Networks 5, 6 and 7. Network 5 has the best performance, closely followed by that of Network 6. Hence reducing by half the size of the model has had little effect on performance. Increasing the proportion of forgeries in the training data has reduced the performance of Network 7.

6.3 Zero-Effort Forgeries

This test again used the larger test set. It consisted of 532 genuine signatures, as before, and 430 zero-effort forgeries. For each person, the first genuine signature of the following ten people in the test set was used to make the ten zero-effort forgeries.

Fig 8 shows the results for Network 5 on genuine and zero-effort forgeries. For comparison, the results of testing the same network on genuine signatures and skilled forgeries is also plotted. Surprisingly, Network 5 is better at detecting skilled than zero-effort forgeries. Removing people with very variable signatures from the test, improved the performance on zero-effort forgeries.

6.4 The 80 Byte Constraint

The requirement to represent a model of a signature in 80 bytes means that the signature feature vector must be encodable in 80 bytes. Architecture 2 was specifically designed with this requirement in mind. Its signature feature vector has 76 dimensions. When testing Network 6, which was based on this architecture, 50% of the outputs were found (surprisingly) to be redundant and the signature could be represented by a 38 dimensional vector with no loss of performance. One explanation for this redundancy is that, by reducing the dimension of the output (by not using some outputs), it is easier for the neural network to satisfy the constraint that genuine and forgery vectors have a cosine distance of -1 (equivalent to the outputs from 2 such signatures pointing in opposite directions).

These results were gathered on a Sun SPARC2 workstation where the 38 values were each represented with 4 bytes. A test was made representing each value in one byte. This had no detrimental effect on the performance. Using one byte per value allows the signature feature vector to be coded in 38 bytes, which is well within the size constraint. It may be possible to represent a signature feature vector with even less resolution, but this was not investigated. For a model to be updatable (a requirement of this project), the total of all the squares for each component of the signature feature vectors must also be available. This is another 38 dimensional vector. From these two vectors the variance can be calculated and a test signature verified. These two vectors can be stored in 80 bytes.

6.5 Comments on the Testing

Fig. 9 shows some typical examples of genuine signatures scored with high or low probability of being genuine. It was observed that when a genuine signature was rejected it was very often one of the first written (signatures 8 and 11 of Fig. 9). (The test set happened to be made up from the first signatures signed by each person on the pad.) Signing on the 5990's glass screen has a different "feel" than signing on paper and from our results and people's observations it seems that a person needs some practice at writing on this screen before they can sign consistently. Other authors have also noticed that a short period of practice was needed to adjust to new writing apparatus [3]. Testing Network 4 with the larger test set, using a threshold of 0.5 led to 80% of forgeries being detected (as required by the NCR requirements), but only 95.5% of genuines being accepted (24 were rejected). However, if all first and second genuine signatures are removed from the test set the result becomes 97.0% (13 rejected). This confirms the observation that the first few signatures signed on the pad are not typical of the person's signature. The protocol for use of credit cards with this verification algorithm has not been decided. These results suggest a few guidelines for the model building process; the first few signatures should be discarded and at least 6 signatures should be used to build the model (When this verification algorithm was used in a demonstration system we found that the more signatures given, the more accurate the model and the harder the person was to forge).

Many genuine signatures are low scoring because of uncharacteristic pen up trajectories. In the test of Network 4, once the first and second signatures are excluded 9 of the remaining 13 genuines with low scores had pen up trajectories that differed from the person's typical signature. However, leaving out pen up trajectories does not seem to be the solution to this problem. Network 2 with no pen up information led to lowered performance, Network 3 with very few pen up points gave hardly significant performance improvement over Network 1 which used both pen up and pen down points. The conclusion is that pen up points are useful for some signature verifications and not others. What is required is a method to measure distance between a model and a test signature which allows for flexible matching along the pen trajectory. By training a variable length input network and then using an elastic matching method for measuring distance, we hope for improvement. This has been shown to lead to improvements on speech recognition using TDNN's[13]. The neural network based verification algorithm described here allows a person's signature to be stored in a very small space, however, it does not achieve the required rejection/accepting accuracy and a distance measure based on time warping offers a solution to this deficit. Another solution to improving performance, if SMART cards were used instead of credit cards and the 80 byte signature storage constraint relaxed, would be to combine the method reported here with our verification system based on global features. This system had comparable performance but different strengths.

Due to a quirk in functioning of the digitizer, some parts of a signature that should normally have been pen down, were actually reported as pen up. For the networks which kept both pen up and pen down data, there was some tolerance to this and these genuine signatures could still be correctly verified.

Some people seemed unable to sign consistently and would suddenly miss out letters or add new strokes to their signature. Any verification method has limits; personal identification number (PIN) based systems are limited by people forgetting their PIN, signature verification is limited by how consistently people sign, fingerprint verification by the quality of the fingerprint taken. From this trial is seems that, for some people, signing is not a natural and automatically repeatable process. This inconsistency can be compensated for by using many signatures to build a model. However a large variability leads to a large variance for the signature model, which in turn leads to more forgeries being accepted.

Some forgeries were very good. In a test in which two unskilled humans first studied genuine signatures as they would appear on paper and then tried to verify a mixture of forgeries and genuines, the humans actually performed worse than the neural networks. This suggests that this verification algorithm would be better able to spot forgers than a shop cashier and could help in reducing credit card fraud.

Network 5 was found to be better at detecting skilled forgeries than zero-effort forgeries. There are two explanations for this; the network was trained on very few zero-effort forgeries (5 times less zero-effort forgeries than skilled forgeries) and hence never learnt to reliably detect them; secondly, because most forgers are trying to replicate the shape of a signature rather than its temporal characteristics, skilled forgeries tend to have temporal properties uncharacteristic of any signature whereas zero-effort signatures are genuine signatures for someone and hence cannot be detected by an acharacteristic temporal pattern. This also suggests that more training on zero-effort forgeries is required in order for them to be detected as reliably as skilled forgeries. During training we can control the proportions of the three types of signature pairs (g:g, g:0, g:f) however, we have no data on the proportions in the real world or the relative cost to credit card companies of these two types of forgeries. Given our small amount of data we chose to use all available genuine signature pairs and an equivalent amount of genuine:forgery pairs and made no attempt (except network 7) to examine the effect of varying this.

Architecture 1 has two defects; (i) it performs a compression in the time extent of signatures by using a step size of three for the two convolutional layers and (ii) its output of 16 by 19 units is probably too large to be represented in 80 bytes. Architecture 2 was designed to address these two defects. It does not subsample during the convolutional layers, but instead an averaging layer is introduced after each of these convolutional layers to effect the required compression in time. This method should be more robust to noise in the data. The output size has been reduced to 4 by 19. Performance of Network 6, which is based on the second architecture, was only slightly worse than the best performance of any network using the first architecture. Less effort was spent on fine-tuning this architecture and its training.

The signature data was treated to have a lower resolution of about 100 dots per inch. Test results were unaffected showing that this algorithm should work for signatures captured on touch-sensitive pads with lower resolution.

The results in this paper should be seen as an estimate of possible performance in the real world. The data collected for this task is not a good representation of data from the real world. People could choose whether to sign their own signature or try to forge another from the database and thus it does not reflect the ratio of skilled forgeries to genuine signatures that would be expected from a working system. Also, the proportion of forgeries in the real world that may be of the type zero-effort is unknown. There was no incentive for people to sign consistently. In fact, many people deliberately tried to break the system. This led to good forgeries, but also to genuine signers signing inconsistently. As far as was known, no occupational forger signed in our data sets, so we could expect to get better forgeries in a real life system, however, we would also expect more consistency in genuine signatures. All this contributes to making predictions about real-world performance difficult. This question can only be answered by running a field trial. However, results presented here can only be compared with other methods which use identical test sets.

Another idea which has not been investigated due to lack of time is the increase in accuracy that could be obtained by requiring some or all people to sign twice when being verified. This will certainly improve performance, but at the cost of slowing down the verification process.

7 Conclusions

This paper describes an algorithm for signature verification. It is tunable, a model of a person's signature can easily fit in 80 bytes and the model can be updated and become more accurate with each successful use of the credit card (surely an incentive for people to use their credit card as frequently as possible). Other beneficial aspects of this verification algorithm are that it is more resistant to forgeries for people who sign their signature consistently, the algorithm is independent of the general direction of signing and is insensitive to changes in size and slope.

As a result of this project, a demonstration system incorporating the neural network signature verification algorithm was developed. It has been used in demonstrations at Bell Laboratories where it worked equally well for American, European and Chinese signatures. This has been shown to commercial customers. We hope that a field trial can be run in order to test this technology in the real world.

8 Acknowledgments

All the neural network training and testing was carried out using SN2.6, a neural network simulator package developed by Neuristique. We would like to thank Bernhard Boser, John Denker, Donnie Henderson, Vic Nalwa and the members of the Interactive Systems department at AT&T Bell Laboratories, and Sam Wagner at NCR Corporation, for their help and encouragement. Finally, we thank all the people who took time to donate signatures for this project.

References

[1] Michael Quint. The credit card balance sheet. *New York Times*, A1:p1, November 19th 1991.

[2] C. E. Strangio. Numerical comparison of similarly structured data perturbed by random variations, as found in handwritten signatures. Technical Report Dept. of Elect. Eng., MIT, 1976.

[3] N. M. Herbst and C. N. Liu. Automatic signature verification based on accelerometry. *IBM J. Res. Develop.*, Vol. 21:245–253, 1977.

[4] Réjean Plamondon and Guy Lorette. Automatic signature verification and writer identification — the state of the art. *Pattern Recognition*, Vol. 22:107–131, 1989.

[5] P. Baldi and Y. Chauvin. Neural networks for fingerprint recognition. *To appear in Neural Computation*, 1993.

[6] K. J. Lang and G. E. Hinton. A time delay neural network architecture for speech recognition. Technical Report CMU-cs-88-152, Carnegie-Mellon University, Pittsburgh, PA, 1988.

[7] I. Guyon, P. Albrecht, Y. LeCun, J. S. Denker, and W. Hubbard. A time delay neural network character recognizer for a touch terminal. *Pattern Recognition*, 1990.

[8] Y. LeCun. Generalization and network design strategies. Technical Report CRG-TR-89-4, University of Toronto Connectionist Research Group, 1989.

[9] Guy Lorette and Réjean Plamondon. Dynamic approaches to handwritten signature verification. In R. Plamondon and C. G. Leedham, editors, *Computer processing of handwriting.* World Scientific, 1990.

[10] I. Yoshimura and M. Yoshimura. On-line signature verification incorporating the direction of pen movement – an experimental examination of the effectiveness. In S. Impedova and J. C. Simon, editors, *From pixels to features III: frontiers in Handwriting recognition.* Elsevier, 1992.

[11] R. Duda and P. Hart. *Pattern Classification and Scene Analysis.* John Wiley and Sons, Inc., 1973.

[12] Réjean Plamondon and M. Parizeau. Signature verification from position, velocity and acceleration signals: a comparitive study. *Proc. 9th Int. Conf. on Patt. Recog.*, pages 26–265, 1988.

[13] Xavier Driancourt, Léon Bottou, and Patrick Gallinari. Learning vector quantization, multi layer perceptron and dynamic programming: Comparison and cooperation. *International Joint Conference on Neural Networks*, Vol. 2:815–819, 1991.

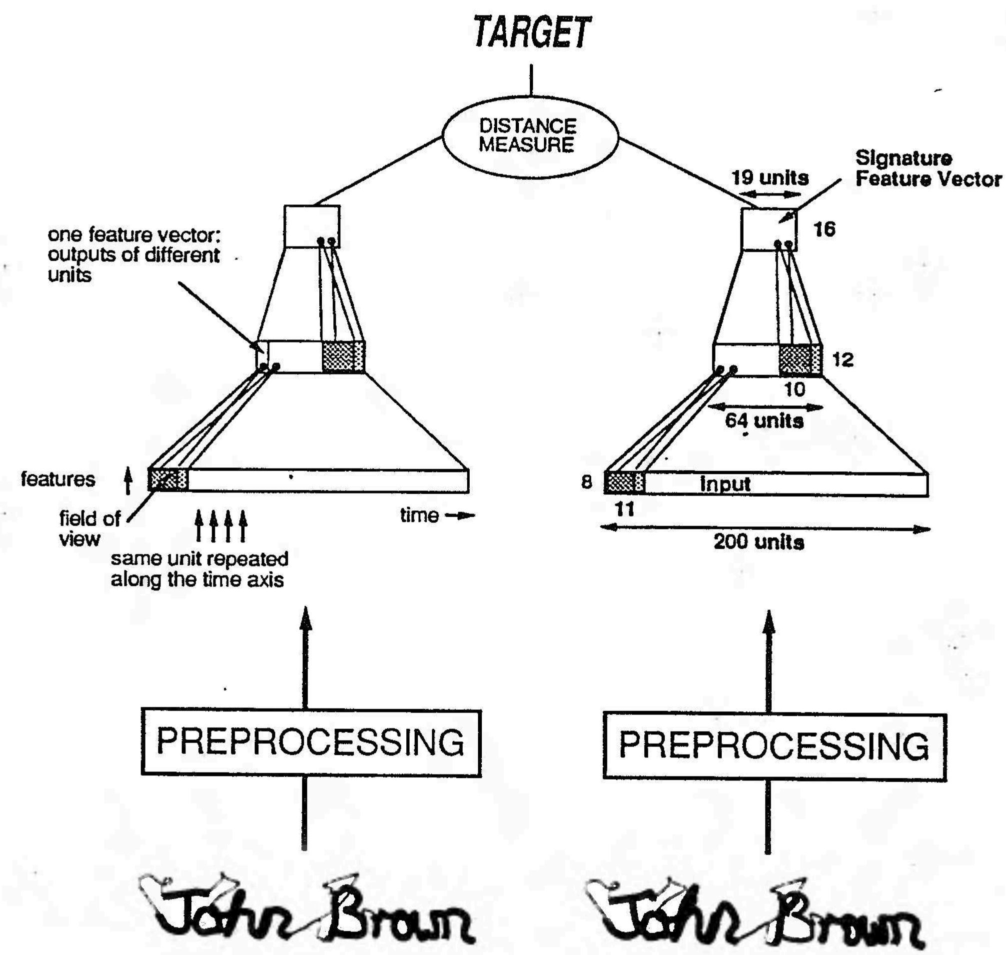

Figure 1: Architecture 1 consists of two identical time delay neural networks. Each network has an input of 8 by 200 units, first layer of 12 by 64 units with receptive fields for each unit being 8 by 11 and a second layer of 16 by 19 units with receptive fields 12 by 10. Compression in the time direction is effected by sub-sampling during the convolution.

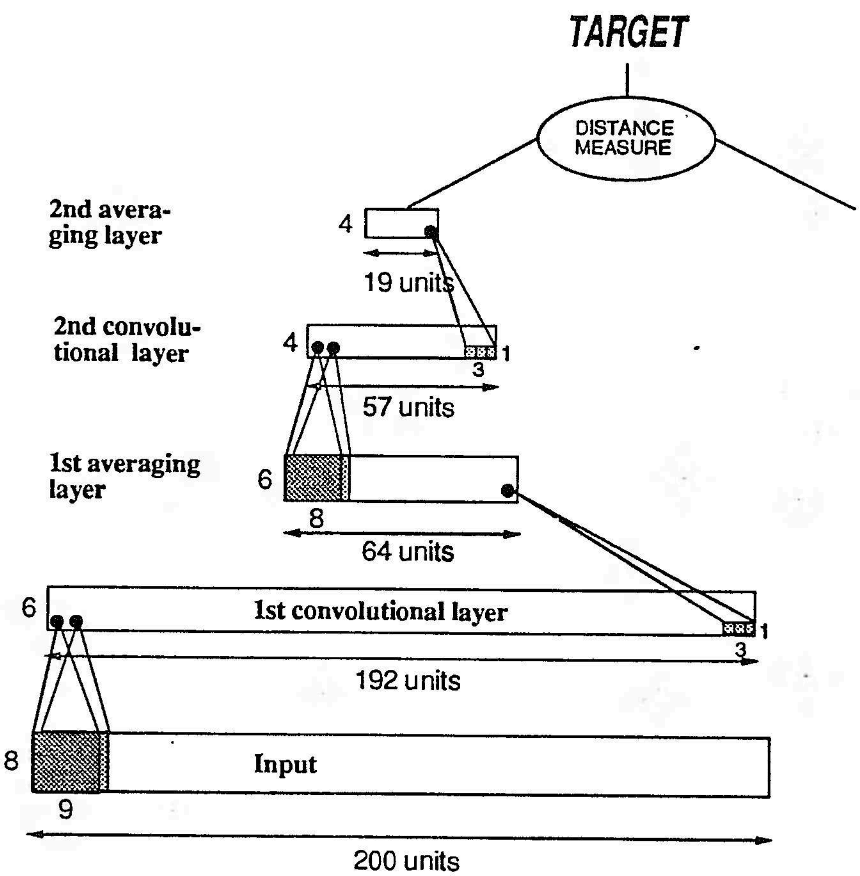

Figure 2: Architecture 2 consists of two identical time delay neural networks, but only one of them is drawn here. The input is 8 by 200 units, the first convolutional layer is 6 by 192 units with each unit's receptive field covering 8 by 9 units of the layer below. The first averaging layer is 6 by 64 units, the second convolution layer is 4 by 57 with 6 by 8 receptive fields and the second averaging layer is 4 by 19. Compression in the time direction is effected by the two averaging layers. Each unit in these layers averages the outputs of 3 neurones in the layer below

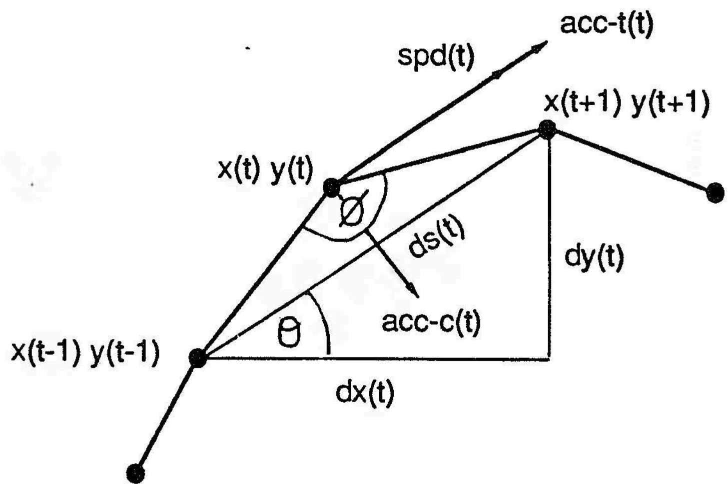

Figure 3: A point on a typical signature trajectory showing the features computed during the preprocessing.

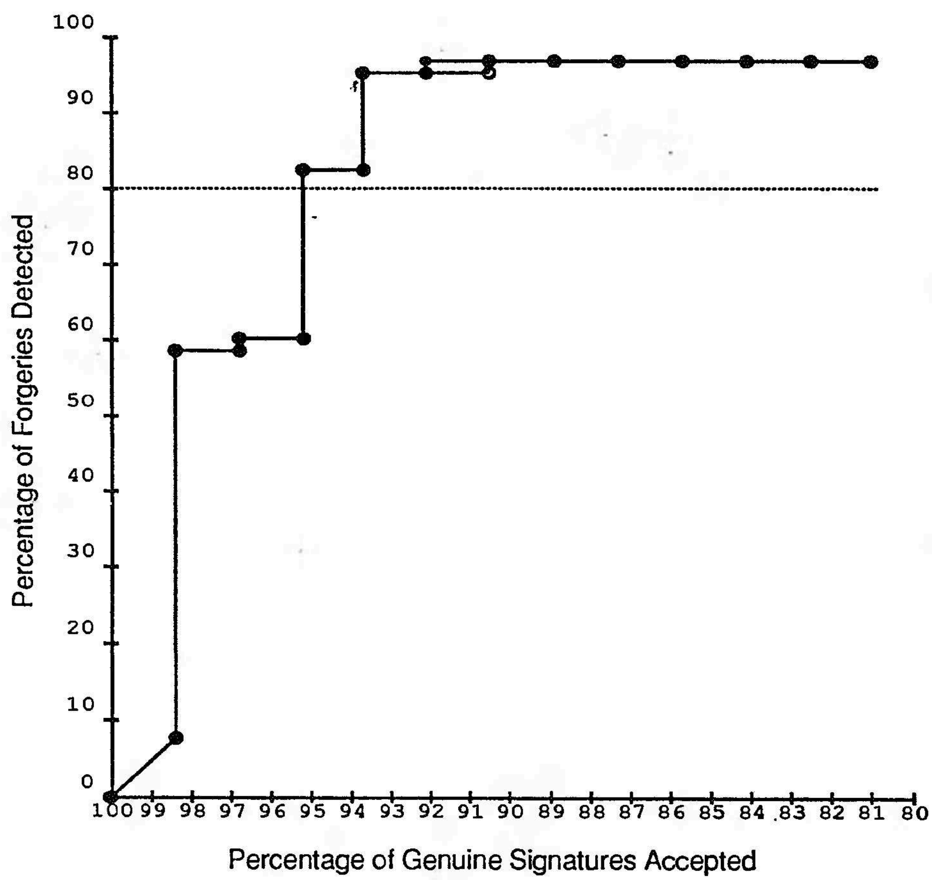

Figure 4: Results for Network 1 on the test set (closed circles) and with data simulated to come from the 5991 Signature Capture Device that has 3-fold less resolution in x and y than the standard data (open-circles).

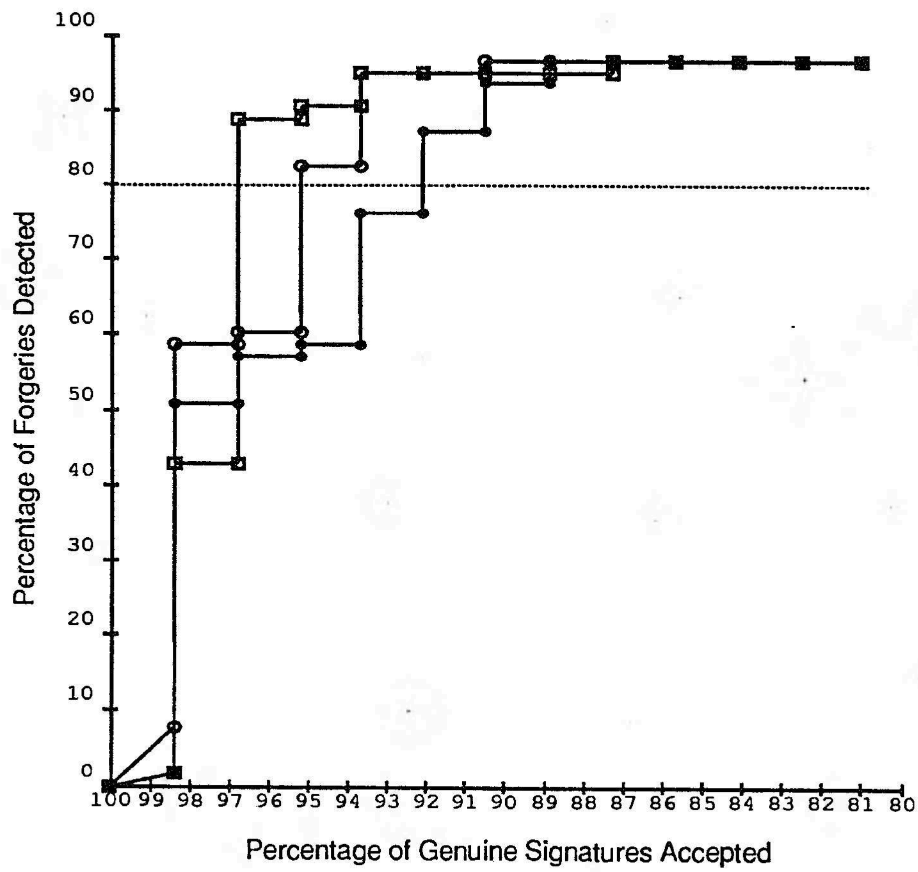

Figure 5: Results for Network 2 trained and tested on just pen down signature trajectories (closed circles) and Network 3 trained and tested on data with all but first and last pen up trajectories removed (open squares) and, for comparison, Network 1 on the standard test set (open circles).

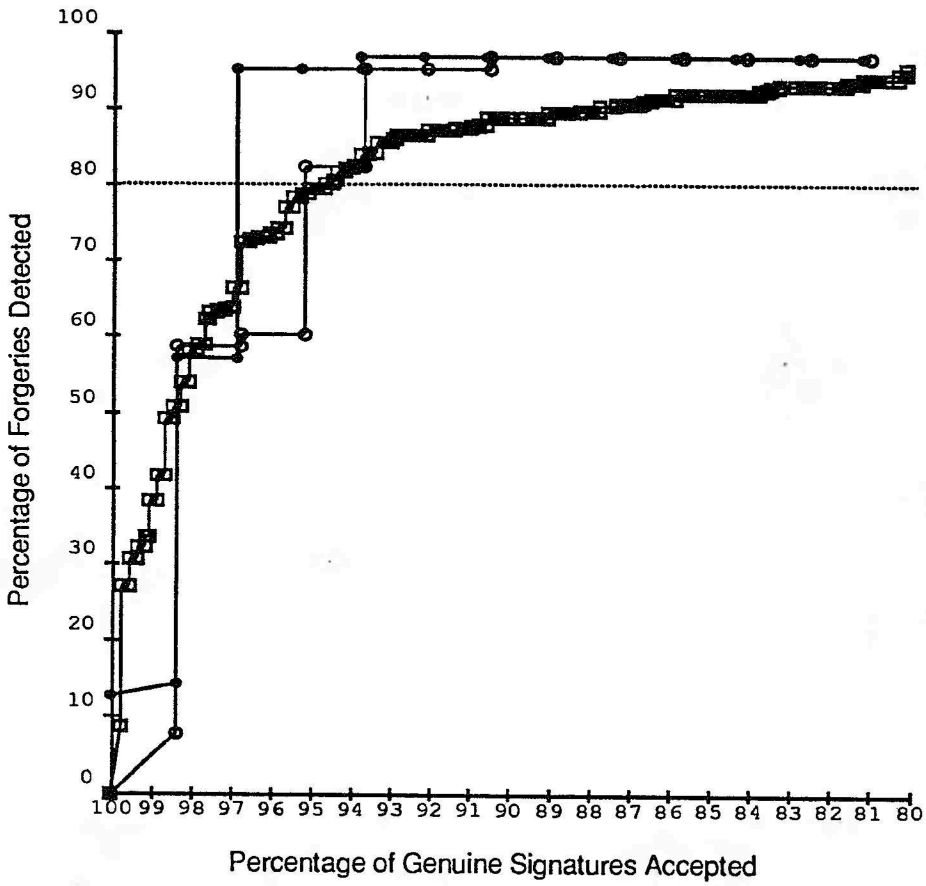

Figure 6: Results for Network 1 on the standard test set (open circles) and Network 4 (closed circles) for which x and y were substituted as input features instead of acc-c and acc-t. Network 4 was also tested on the larger test set (open squares).

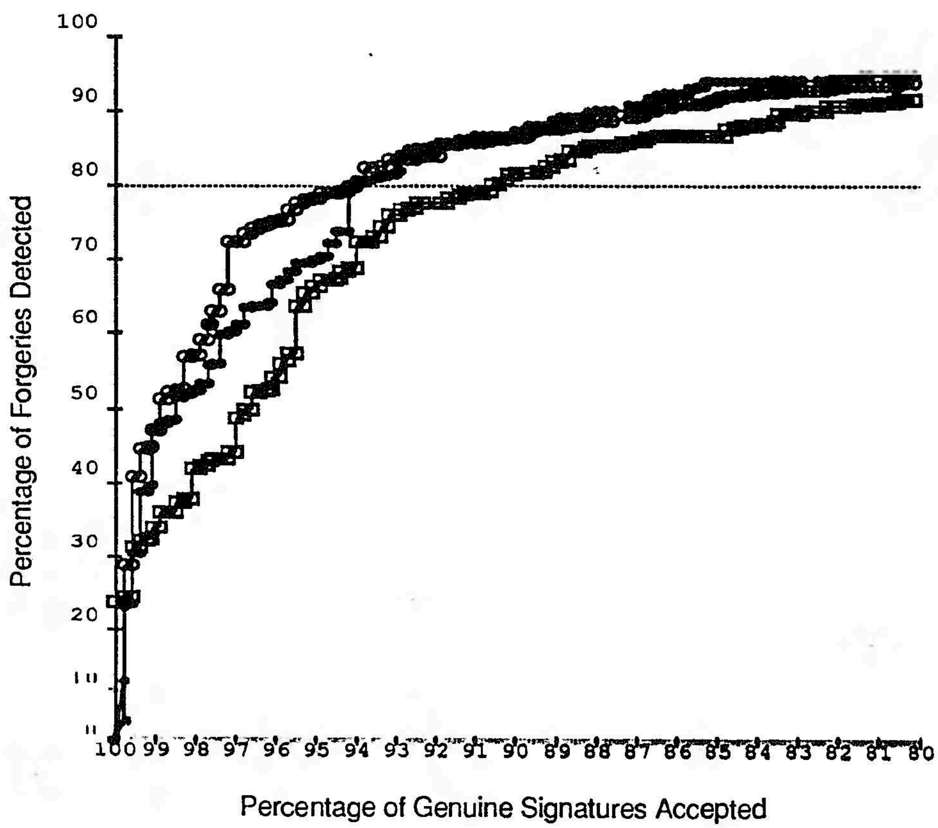

Figure 7: Results for Networks 5 (open circles), 6 (closed circles) and 7 (open squares). The training of Network 5 was essentially the same as for Network 4 except that more data was used in training and it had been cleaned of noise. Networks 6 and 7 were based on architecture 2. The signature feature vector from this architecture is just 4 by 19 in size. Network 7 was trained on four times as many forgery signature pairs as Network 6.

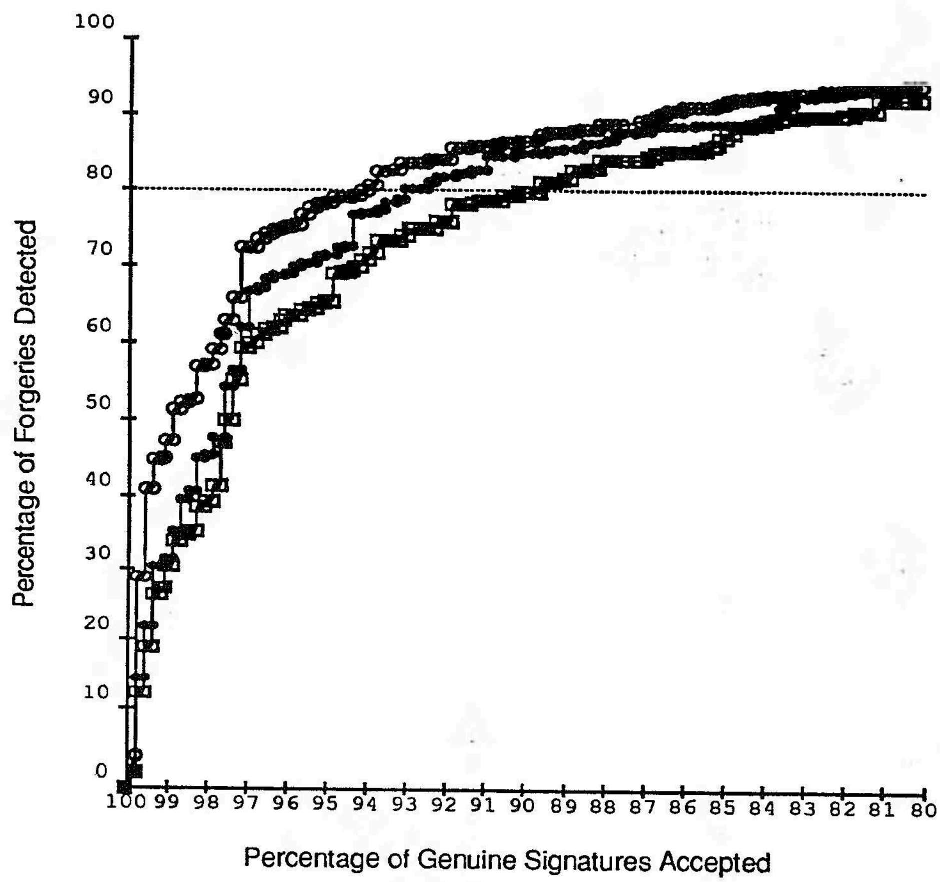

Figure 8: Results for Network 5 tested on genuine and zero-effort forgeries (open squares), the same test except all people with very variable signatures (variance greater than 0.08) were removed from the test set (closed circles) and for comparison the results when testing with genuines and skilled forgeries (open circle).

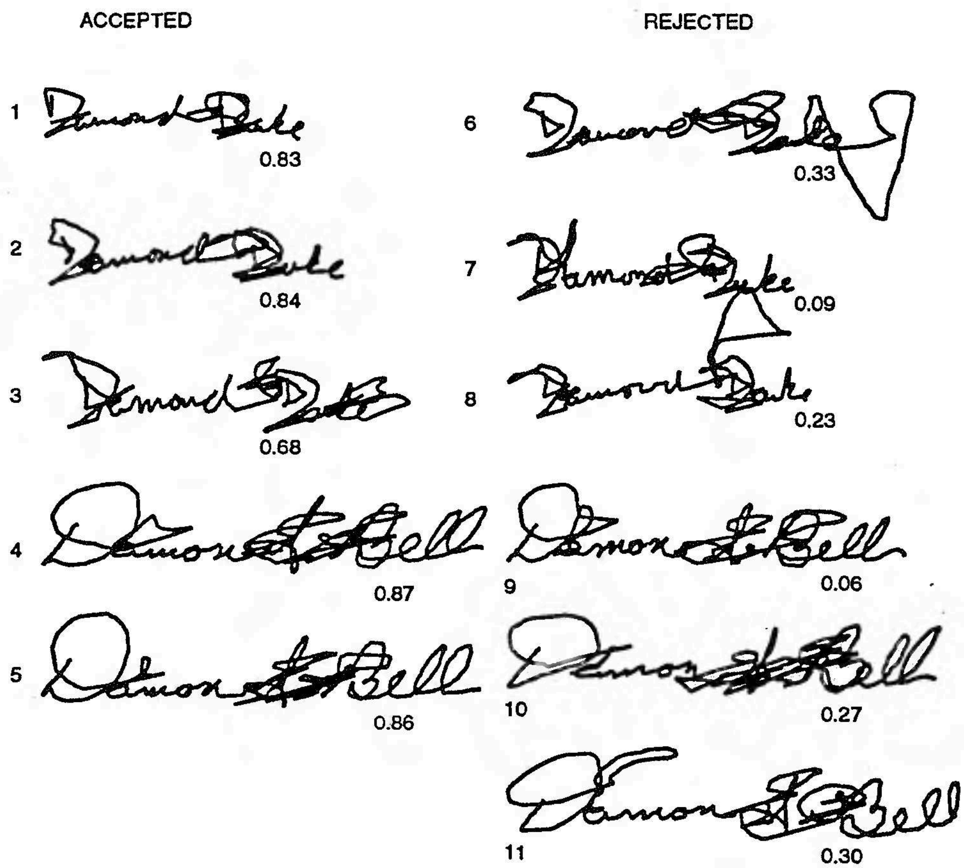

Figure 9: Typical examples of two peoples' genuine signatures. 1,2,3,6,7 and 8 were signed by one person, 4,5,9,10 and 11 by another. These signatures were used in a test of Network 4. Left hand column is signatures with high probability of being genuine, right hand column is signatures with low probability of being genuine. The probability value is shown at the bottom right corner of each signature. Both pen up and pen down trajectories are shown.