Neural Network-based Handwritten Signature Verification

Alan McCabe, Jarrod Trevathan and Wayne Read
School of Mathematics, Physics and Information Technology, James Cook University, Australia
Email: alan@mymait.com, {jarrod.trevathan, wayne.read}@jcu.edu.au

Abstract— **Handwritten signatures are considered as the most natural method of authenticating a person's identity (compared to other biometric and cryptographic forms of authentication). The learning process inherent in Neural Networks (NN) can be applied to the process of verifying handwritten signatures that are electronically captured via a stylus. This paper presents a method for verifying handwritten signatures by using a NN architecture. Various static (e.g., height, slant, etc.) and dynamic (e.g., velocity, pen tip pressure, etc.) signature features are extracted and used to train the NN. Several Network topologies are tested and their accuracy is compared. The resulting system performs reasonably well with an overall error rate of 3.3% being reported for the best case.**

I. Introduction

Biometric security is a computerised method of verifying a person's identity based on his/her body and/or physical attributes. Various forms of biometric security exist including fingerprinting, iris recognition [10], speech recognition [17], heart sound recognition [7], and keystroke recognition [12]. However, dispite the novelty and perceived security of the aforemention techniques, the longest standing and most natural method for verifying one's identity is through the use of a handwritten signature. Handwritten Signature Verification (HSV) is an automated method of verifying a signature by capturing features about a signature's shape (i.e., static features) and the charactertics of how the person signs his/her name in real-time (i.e., dynamic features). HSV is more generally accepted by the public and is less intrusive than other biometric authentication techniques.

Neural networks (NNs) have been a fundamental part of computerised pattern recognition tasks for more than half a century, and continue to be used in a very broad range of problem domains. The two main reasons for their widespread usage are: 1) power (the sophisticated techniques used in NNs allow a capability of modeling quite complex functions); and 2) ease of use (as NNs learn by example it is only necessary for a user to gather a highly representative data set and then invoke training algorithms to learn the underlying structure of the data).

The HSV process parallels this learning mechanism. There are many ways to structure the NN training, but a very simple approach is to firstly extract a feature set representing the signature (details like length, height, duration, etc.), with several samples from different signers. The second step is for the NN to learn the relationship between a signature and its class (either "genuine" or "forgery"). Once this relationship has been learned, the network can be presented with test signatures that can be classified as belonging to a particular signer. NNs therefore are highly suited to modeling *global* aspects of handwritten signatures.

Concentrated efforts at applying NNs to HSV have been undertaken for over a decade with varying degrees of success (e.g., see [9], [16]). The main attractions include:

1) *Expressiveness*: NNs are an attribute-based representation and are well-suited for continuous inputs and outputs. The class of multi-layer networks as a whole can represent any desired function of a set of attributes, and signatures can be readily modeled as a function of a set of attributes.
2) *Ability to generalise*: NNs are an excellent generalization tool (under normal conditions) and are a useful means of coping with the diversity and variations inherent in handwritten signatures.
3) *Sensitivity to noise*: NNs are designed to simply find the best fit through the input points within the constraints of the network topology (using nonlinear regression). As a result, NNs are very tolerant of noise in the input data.
4) *Graceful degradation*: NNs tend to display graceful degradation rather than a sharp drop-off in performance as conditions worsen.
5) *Execution speed*: The NN training phase can take a large amount of time. In HSV this training is a one-off cost undertaken off-line (i.e., rarely performed while a user waits for verification results).

This paper presents a method for HSV by using a NN architecture. Various static (e.g., height, slant, etc.) and dynamic (e.g., velocity, pen tip pressure, etc.) signature features are extracted and used to train the NN. Several Network topologies are tested and their accuracy is compared. The resulting system performs reasonably well with an overall error rate of 3.3% being reported for the best case.

This paper is organised as follows: Section II describes the methodology regarding how signatures are captured and the feature extraction process. Section III details the experimentation performed as part of the NN HSV sys-

This paper is derived from "Handwritten Signature Verification Using Complementary Statistical Models," by A. McCabe, which served as a PhD Dissertation at James Cook University, November 2003. © 2003.

tem development. Section IV provides some concluding remarks.

II. METHODOLOGY

This section describes the methodology behind the system development. It discusses the pre-processing performed, the signature database, and the NN features.

A. Pre-processing

This study required minimal signature pre-processing. Other areas of handwriting analysis require large amounts of pre-processing such as slant correction, rotation correction and size normalisation to reduce variations in the handwriting [2]. However, in HSV most of the subtle nuances of the writing such as size and slant are indicative of the signer's natural style, removal of which would deny the HSV system of useful information. Additionally, the use of high quality tablet hardware to capture signatures prevents most of the noise that might be introduced through processes such as scanning.

The only pre-processing performed is rotation normalisation (necessary as the orientation of the tablet and resulting signatures is not always consistent). This procedure involves extracting the baseline points from the signature (i.e., the bottoms of all non-descender characters). Linear regression is used to best fit a straight line through the baseline points. The signature is translated to the origin and is rotated using the following formulae:

$$\begin{aligned} x' &= x \cdot cos(\theta) - y \cdot sin(\theta) \\ y' &= x \cdot sin(\theta) + y \cdot cos(\theta) \end{aligned}$$

where θ is the inverse tan of the gradient of the line found during linear regression and x' and y' are the new x and y coordinates.

B. Signature Database

A signature database was created in order to facilitate the HSV experimentation. One of the most unfortunate aspects of current HSV research is the lack of standard databases. This is a major inconvenience for researchers who need to spend a large amount of time capturing their own data. Furthermore, using different databases makes it almost impossible to perform meaningful comparisons between different HSV systems.

The following describes the criteria for an effective signature database:

- *Large sample size*: In HSV most researchers tend to work with dataset of over one thousand signatures. The more signatures used and the more diverse the dataset, the more reliable the obtained error rates are.
- *Diversity of samples*: The system must simulate a realistic environment using signers of different age groups, sexes, nationalities, backgrounds and left- and right-handedness.
- *No arbitrary exclusion*: It is unacceptable to remove "undesirable" or "inappropriate" signers, or those that are not likely to work well with the proposed system (e.g., [3], [8]). It is necessary to include *all* captured signatures in the database.
- *Inclusion of skilled forgeries*: It is more difficult to obtain skilled forgeries than genuine signatures, but without the inclusion of skilled forgeries, quoting false rejection rates is far less meaningful.

Fig. 1. *The sampled coordinates captured from the handwritten word "farley" (Left). Interpolation of the sampled coordinates produces the off-line, or static, image of the word (Right).*

All samples used in the project were captured using a XGT Serial Digitizing tablet [1]. The XGT consists of an opaque tablet and a cordless non-inking pressure-sensitive pen. The XGT has a 152×203 millimetre (6×8 inch) effective writing area and captures samples at the rate of 205 points per second. The resolution is 1,000 points per centimetre (2,540 points per inch) at an accuracy of 0.0127 centimetres (0.005 inches). In addition, the tablet captures pen-tip pressure as one of 256 levels measured through the pressure sensitive tip of the cordless pen. The pressure and position values are translated into coordinates on the serial bus. The hardware interface is a Serial EIA Standard RS-232C port connected to a laptop computer running custom-written driver software.

The values in the output stream produced by the digitiser are equidistant in time and consist of tuples (x, y, p) that contain the following data:

- $x(t)$, the x-coordinate sampled at time t;
- $y(t)$, the y-coordinate sampled at time t;
- $p(t)$, the axial pen force at time t.

The sampled coordinates of a typical handwritten word can be plotted and displayed (see Figure 1).

The XGT tablet samples pen-tip position even when the pen is not in contact with the writing surface, as long as it is within 2.5 centimetres (1 inch) of the tablet. The pen's path while in a "pen-up" state is undetectable by forgers examining off-line signature copies, providing a useful source of extra information for a HSV system.

The experimental database was captured using the above tablet with the signer being seated in a comfortable position with good lighting. The signers were orally prompted to provide their signature sample in their own time. The signers generally provided five samples in one sitting, with this operation being repeated on two or three separate occasions (resulting in between ten and fifteen genuine signatures per person). Forgeries were obtained by allowing the forger to view a static image of the written password as well as being provided with basic

[1] http://www.mutoh.com/

TABLE I

A summary of the signature database used in experimentation.

Property	Value
Number of signers	111
Number of genuine signatures	2,779
Number of forgeries	1,110
Digitiser	Kurta XGT
Sample rate	205 pps
Resolution	1,000 ppcm (2,540 ppi)
Error	+/- 0.0127 cm (0.005 inches)
Pressure levels	256

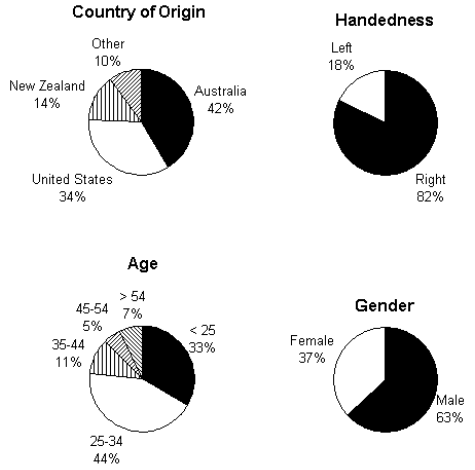

Fig. 2. *Contributors to the database grouped according to (a) nationalities, (b) handedness (left or right), (c) age and (d) gender.*

information on the signature dynamics in the form of velocity and pressure profiles. Forgers were then allowed to practice their forgeries and to attempt the forgery in a similar environment to that in which the original signature was performed. Forgers were not professional or trained forgers, but they were aware of the nature of the verification system, and were aware of the nature of the writing they were attempting to forge.

Attempts were made to obtain a reasonably realistic database population. The set of genuine signatures contained writers of several different nationalities and backgrounds as well as different age groups, genders and left- and right-handedness (see Figure 2). In total there are 111 signers in the database contributing a total of 2,779 genuine signatures. In most experiments, five signatures are used to build each reference, leaving 2,779 - (111×5) = 2,224 genuine signatures for testing. Ten skilled forgeries were captured for each signer, providing a total of 1,110 forgeries in the database. No captured signatures were excluded from use in the database (with the exception of a small number in which a device failure occurred and no signature data was obtained). The database itself is quite large and diverse compared with others in the literature. (See Table I.)

C. Extracted Features

The features extracted from signatures or handwriting play a vital role in the success of any feature-based HSV system. They are the most important aspect, exceeding the choice of model or comparison means. If a poorly constructed feature set is used with little insight into the writer's natural style, then no amount of modeling or analysis is going to result in a successful system. Further, it is necessary to have multiple, meaningful features in the input vector to guarantee useful learning by the NN [6].

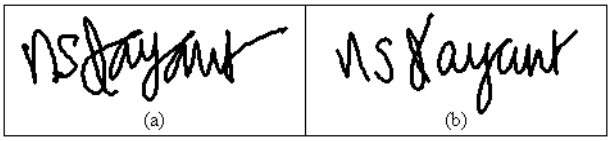

Fig. 3. *This illustrates the difficulty that a potential forger has in trying to identify the pen-down ratio. (a) is a genuine signature and (b) is an attempted forgery based on the forger having seen an off-line version of the signature (both taken from the signature database used in this project). The pen-down ratio for the genuine signature is 0.992 and is 0.879 for the forgery (forgeries were typically found to have much lower pen-down ratios, presumably because of the extra attention to detail).*

The initial decisions as to which features to incorporate, in order to maximise the accuracy, involved a combination of studying other publications in the area (what other researchers have found useful or useless) and intuitively considering which other features might be most applicable. The intuitive approach was based on study of the handwriting process, forensic analysis of handwriting by humans and examination of features that are most useful to humans in deciding whether a particular handwriting sample is produced by some author.

The properties of "useful" features must satisfy the following three requirements [19]:

1) The writer must be able to write in a standard, consistent way (i.e., not unnaturally fast or slow in order to produce a particular feature);
2) The writer must be somewhat separable from other writers based on the feature;
3) The features must be environment invariant (remain consistent irrespective of what is being written).

The third point is more relevant to the process of writer identification than HSV, as a person's signature is most often a fixed text. It is relevant to HSV, however, in the sense that the features should remain stable irrespective of the environment in which the signature is being performed (e.g., the pen's weight, the pen tip's friction, etc.).

What follows now is a description of each of the features that are extracted from a given signature, as well as their significance and method of calculation. Each of these features acts as a single input to the NN.

Signature Duration: The time taken to perform a signature is perhaps the single most discriminatory feature in HSV. A study reported in [18] found that 59% of forgeries can be rejected on the basis of the signature duration being more than 20% different from the mean.

Pen-Down Ratio: This is the ratio of the pen-down time to the total writing time. This feature does not undergo a large amount of variation when signing, irrespective of the writer's mood or emotions. In addition, it is very difficult to forge as there is no way of determining the pen-down ratio from an off-line writing copy (see Figure 3). Calculation is performed by removing leading and trailing zeroes from the captured data, then taking the ratio of the number of non-zero points to the total number of points.

Fig. 4. *Horizontal length of a typical handwritten word. This sample's horizontal length is 1,345 pixels.*

Horizontal Length: This is the horizontal distance measured between the two most extreme points in the x direction (often simply the distance between the first point captured and the last point captured). Any fragments such as 't' crossings or 'i' dottings are excluded (such fragments far less stable and individual traits such as extravagant 't' crossings can cause high variability with this feature). The horizontal length tends to remain stable with a practiced word and particularly with a signature, irrespective of the presence of a bounding box, horizontal line or even with no line present. (See Figure 4.)

Aspect Ratio: This is the ratio of the writing length to the writing height. It remains invariant to scaling. If the user signs in a different size, the height and length will be altered proportionally to retain the aspect ratio.

Number of "pen-ups": This indicates the number of times the pen is lifted while signing after the first contact with the tablet and excluding the final pen-lift. This is highly stable and almost never changes in an established signature. This can be a difficult feature for a forger to discern from an off-line copy of the signature.

Cursivity: This is a number normalised to between zero and one that represents the degree to which a writer produces isolated handprint characters or fully-connected cursive script. The higher the cursivity value, the more connected the word is. A value of one means that there were no pen-ups over the entire word and value closer to zero means that the writing was mostly printed rather than cursive. In [14] the original formula for cursivity is:

$$C_n = \frac{1}{n}\sum_{w=1}^{n}\frac{N_{l,w} - N_{pd,w} + 1}{N_{l,w}}$$

where:

- C_n is the cursivity index;
- n is the number of words without letters 'i' or 'j';
- $N_{l,w}$ is the number of letters in a "non-ij" word;
- $N_{pd,w}$ is the number of pen-down streams in this word.

However, the drawback with this approach is that it is necessary to have a priori knowledge of how many letters are in the word being written. While this is possible is some situations it is not a valid assumption with signatures and would defeat the purpose of making this authentication system entirely automated. The calculation of cursivity employed in this paper is done through the use of strokes instead of letters. Strokes are objective and easily calculable from the signature body itself. The formula for cursivity calculation here then is:

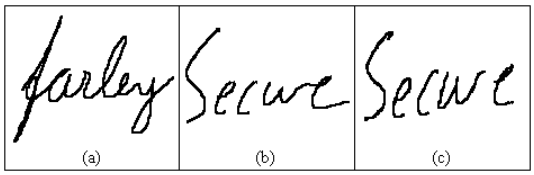

Fig. 5. *Cursivity varies widely between different authors but remains similar for different samples produced by the same author. Parts (a) and (b) contain words written by different authors with very different cursivity values of 16.0 and 3.6 respectively. Part (c) is a sample of the same word as (b), by the same author, and has a very similar cursivity value of 3.8.*

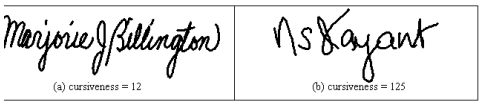

Fig. 6. *Cursiveness varies between authors and is a feature that is highly indicative of natural handwriting style. (a) shows a signature with a seemingly high cursiveness, but the actual value for this is 12 which is significantly lower than the signature in (b) at 125. These signatures show how visual inspection can be deceptive in estimating cursiveness.*

$$Cursivity = \frac{number\ of\ strokes}{number\ of\ pen-downs}$$

A high value indicates a lean towards a more cursive style and a low value implies a more handprinted style. The average level of cursivity is something that remains close to constant for an individual across a large body of handwriting [14]. The same can be said of the same word or small phrase being written by an individual, which is why this feature was considered for use. (See Figure 5.)

Cursiveness: This is a different measure of whether a particular signature is more cursive or more handprinted. This feature remains close to constant across different productions of the same handwritten word by the same author. Cursiveness is a personal handwriting aspect and can be very difficult for a forger to discern due to the dependence of writing style and pause characteristics.

Cursiveness is the ratio of the horizontal length of the handwriting to the number of pen-downs. The cursiveness value goes up when there are less pen-downs and goes down when there are more pen-downs. (See Figure 6.)

Top Heaviness: This is a measure of the proportion of the signature that lies above the vertical midpoint (i.e., the ratio of point density at the top half of the signature versus the density at the bottom half). It is measuring the concentration of the handwriting about its midpoint or conversely, how widely spread the handwriting is. The only real issue in the calculation of top heaviness is to decide which measure of central tendency is most indicative of the true vertical midpoint. It would be pointless to use the median in this situation as, by definition, half of the points would lie above the median and half below. It is therefore a question as to which of the other two standard central measures (mean or mode) is more appropriate. Both of these options were investigated (see Figure 7) (mean is calculated in the standard way, while mode is found by creating a horizontal frequency histogram of the points in the sample and selecting the peak in that histogram). Although the mean is more likely to be affected by outliers in the sample, upon investigation

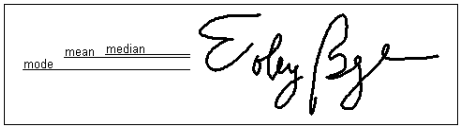

Fig. 7. *Different measures of central tendency can give different midpoints for calculating top-heaviness. The three horizontal lines illustrate the location of the midpoint using the different central tendency measures.*

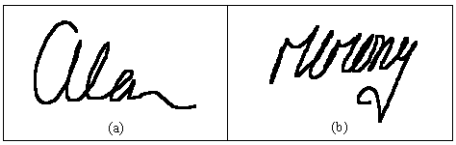

Fig. 8. *Different handwriting samples can result in quite different curvature values. For example, (a) shows a sample in which the writing is quite flat and not well-formed, resulting in a curvature value of 3.96. Conversely (b) shows a sample with a much more pronounced forming of the handwritten characters resulting in the higher value of 5.22.*

with several different signatures from different users, it was found that the large number of points in the data minimised this effect and no measure is regularly more visually central than the other two. As such, both values are used to generate separate measures of top heaviness.

Horizontal Dispersion: This is the same as top heaviness but with respect to the horizontal spread of the handwriting rather than the vertical spread. Calculation is done in a similar fashion.

Curvature: This is a measure of how "flat" or how "curved" the handwriting is. A high curvature value means that the writing is more dramatically curved, which is associated with more thorough or exaggerated completion of handwritten characters (see Figure 8).

Curvature, is slightly susceptible to change depending on the writer's mood or demeanor. If a user is trying to write quickly then they are more likely to write with a lower curvature. However, this feature is still produced with sufficient consistency to be of use when the text is strongly fixed. If the writer's mood remains stable this can be a highly effective feature in identifying the author as it is indicative of the writer's natural style.

Curvature is calculated as the ratio of the signature path length to the word length. The path length is the sum of distances between each consecutive point in the sample so is generally quite large, of the order of 10,000 pixels. The word length is the physical, or Euclidean, distance between the captured writing's first and last point.

Average Curvature per Stroke: This is a feature based on the curvature value described above, except that the curvature value is calculated for each individual stroke in the handwriting sample, then averaged. The difference between this feature and the global curvature value is that by examining the curvature of the individual strokes, it is possible to obtain a more insightful measure of the depth of the local curves in the handwriting. (See Figure 9.)

Number of Strokes: This feature is indicative of how many segments or states the handwriting goes through during the signature's production. This feature remains quite stable over a user's various samples as even with the natural variations in a user's signature, the segmentation

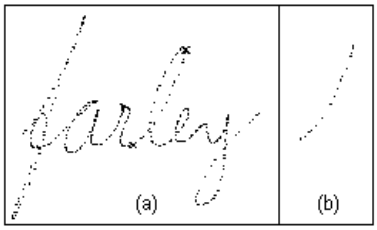

Fig. 9. *Calculating the average curvature per stroke. (a) shows the entire handwritten word and (b) shows an isolated view of one of the extracted strokes.*

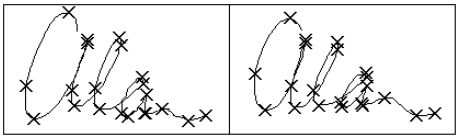

Fig. 10. *Two signatures sections produced by the same author illustrating the consistency in the number of strokes. The crosses on the handwriting represent the stroke boundaries. Both of these samples have 21 strokes and as can be seen the segmentation is quite consistent.*

remains quite similar. Furthermore, this feature is non-trivial for a forger to reproduce as the segmentation is based purely on the pen-tip velocity, which is not visible with just a written version of the word. (See Figure 10.)

Mean Ascender Height: This is the mean height of "ascenders" in the handwriting. Ascenders are letters such as 'd', 'f' and 't' in which a part of the letter extends above the main body of the sample [1]. Formal detection of ascenders in the body of a signature involves computing the mean of the data, as well as points at one quarter and three quarters of the maximum height. The ascender's peaks are the local maxima in the y direction that are above the three quarter mark. The distance between a local maximum and the y mean is found and this distance is taken as the height of that ascender. The mean height for all ascenders is used as the value for this feature. (See Figure 11.)

Mean Descender Depth: Descenders are the opposite of ascenders. They are letters such as 'g', 'y' and 'j' that typically contain parts extending below the main body of the sample (it is possible for an individual letter to be an ascender and a descender – the letter 'f' is sometimes written in this way). Finding the descender extremities is done in a similar fashion to ascenders and uses the same frequency histogram. The descender extremities are the local minima in the y direction that fall below the lower quarter of the sample. The depth value for each extremity is measured as the distance between the local minimum and the y mean expressed as a positive integer. The depth values for all descenders are averaged to give the value

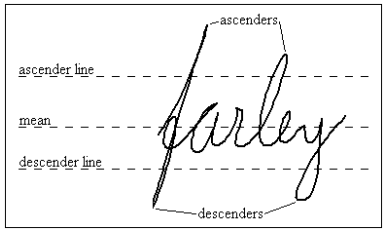

Fig. 11. *Handwriting sample illustrating the ascenders, descenders, mean vertical displacement, ascender height and descender depth.*

Fig. 12. *The maximum height of a signature or handwritten word. The vertical line to the right of the writing sample is the maximum height, and in this case is calculated as 1,005 pixels.*

for this feature. (See Figure 11.)

Maximum Height: This is the distance between the lowest point in a word (the lowest descender's depth) and the highest point in a word (the highest ascender's height). This calculation ignores 'i' dottings and 't' crossings or other such artifacts occurring in the handwriting. Also removed from consideration is the final trailing stroke in a signature – in examination of the trailing strokes in different signatures produced by the same signer, this stroke's height was found to be by far the most variable. The maximum height feature using the remaining captured points reflects, to some extent, the "flair" with which the author writes and the maximum distance typically traversed by the pen tip. This feature remains reasonably stable across several written samples. (See Figure 12.)

Maximum Velocity: Calculation is performed in terms of component velocities v_x and v_y, calculated as the first derivative of the x and y streams:

$$v = \sqrt{v_x^2 + v_y^2}$$

While maximum velocity is subject to some variability, it is a valuable feature because it is unable to identified by a potential forger in an off-line copy of the signature.

Average Velocity: This measures how fast the pen-tip is travelling across the tablet's surface. This is calculated as the mean of all individual velocity values (there is one velocity value for each pair of consecutive points).

Standard Deviation of the Velocity: This is calculated as the standard deviation of all the individual velocity values in the writing sample. It is a measure of the variation of the velocity values characteristic to the signer.

Average Absolute Acceleration: This is the absolute value of the acceleration and deceleration measurements. It is computed as the data stream's second derivative (or the derivative of the velocity values calculated earlier). The average absolute acceleration captures the mean rate of velocity change in both positive and negative directions. The third derivative of the data stream (the derivative of the acceleration) is often referred to as "jerk". Jerk was examined as a potential feature, but did not prove to be either repeatable or individual.

Standard Deviation of the Absolute Acceleration: This measures the dispersion of the absolute acceleration values. It captures the consistency (or lack thereof) with which a user's handwriting accelerates.

Maximum Acceleration: While this feature is less stable than some others, the purely dynamic nature and difficulty in forging still make it a useful characteristic.

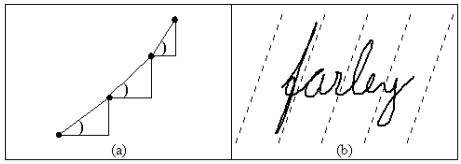

Fig. 13. *The gradient of the line between each pair of consecutive points is determined (part (a)), and the mean of those values found – this mean is the handwriting slant. Part (b) illustrates the computed slant value, drawn as a series of dotted lines laid over the handwriting sample.*

Fig. 14. *"Long strokes" extracted from a typical handwriting sample. The long stroke is represented as bolded handwriting with the remainder of the handwriting appearing as a broken line in the background.*

Maximum Deceleration: This is the rate which the pen-tip's velocity decreases as it approaches a stroke's end.

Handwriting Slant Using All Points: Slant calculation bears much importance in handwriting analysis. It is not trivial and several different approaches were considered. The first approach involved using all captured writing points. The points are spatially resampled and the angle (expressed as a gradient) between each pair of consecutive points in the signature is calculated, giving several gradient values (see Figure 13). The slant is given by the mean of these gradient values. 'i' dottings, 't' crossings and other such artifacts are removed from consideration (and in other discussed slant calculation methods).

Handwriting Slant Using "Long Stroke" End-points: This technique for calculating handwriting slant is based on the extraction of a "long stroke". The definition of a long stroke is two-fold: firstly, the series of points between each vertical minimum and following vertical or horizontal maximum (whichever is encountered first) are extracted; secondly, the long stroke is retained if and only if the stroke path length is greater than some pre-defined threshold (experimentation was performed to find the threshold producing the most visually accurate slant and the final value was set at thirteen). Once the long stroke start and end points have been identified, the stroke's gradient is given by the gradient between just those two points. The mean of all such gradients then gives the handwriting slant. (See Figure 14.)

Handwriting Slant Using All Points of "Long Strokes": This approach also uses long strokes and a technique very similar to that used to obtain the "Handwriting Slant Using All Points". After the long strokes are extracted and the constituent points spatially resampled, the gradient between each pair of consecutive points within the long stroke is calculated. The mean of these gradient values gives the handwriting slant. In experimentation, this slant calculation method was found to be far less stable than

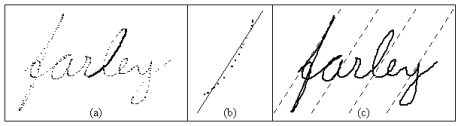

Fig. 15. *Handwriting slant calculation through regression of "long strokes". (a) shows one of the long strokes extracted from a typical handwriting sample. (b) shows a close-up view of that same stroke with the straight line being the line-of-best-fit using linear regression. This line's gradient is taken as the handwriting slant. (c) shows the same handwriting sample used in (a) and is overlaid with a series of straight lines parallel to the calculated slant using the regression of long strokes.*

the method using regression (described below) and was removed from consideration in the NN.

Handwriting Slant Through Regression of "Long Strokes": This approach is slightly different from slant calculation using long stroke end-points. The extraction of long strokes here is again done in the same fashion as described previously. The difference comes in the actual calculation of the slant where linear regression is performed using all of the points in the long stroke. The mean of this value taken over all long strokes in the signature is used as the final slant. (See Figure 15.)

Handwriting Slant Using Cai and Liu Technique [2]: This approach involves calculating the centroid c_i for each row i in the signature within the vertical inter-quartile range (i.e., ignoring ascenders and descenders that may skew the slant value if they are at one end of the signature), and obtaining h row centroid points, where h is the height of this range in pixels. Linear regression is used to best fit a straight line through the centroid points. The slant is the slope of the straight line, given by:

$$Cai - Liu\ slant = ctan^{-1}\left(\frac{S_{xy}}{S_{yy}}\right)$$

where:

- $S_{xy} = \sum_{i=0}^{h-1} G(i)(c_i - \overline{x})(i - \overline{y})$;
- $S_{yy} = \sum_{i=0}^{h-1} G(i)(i - \overline{y})^2$;
- $G(i)$ is the weight associated with the i^{th} row (the number of handwriting strokes crossed by a horizontal projection at i);
- $(\overline{x}, \overline{y})$ is the centroid of the signature;
- $ctan$ is the complex circular tangent.

Handwriting Slant Based on Vertical Overlap: This measures the average number of handwriting strokes crossed by vertical projections through the handwritten sample. This method does not attempt to calculate a value for the gradient of the handwriting. The relationship between vertical overlap and slant is based on the fact that a more pronounced slant will result in a higher value for vertical overlap. In experimentation this feature was found to be highly stable and is difficult to forge. Calculation is performed by making a series of vertical projections along the entire sample's length. For each vertical projection the number of strokes crossed is determined and averaged across all vertical projections. (See Figure 16.)

Stroke Concavity: This measures how close the average stroke is to being a straight line. A stroke of high concavity does not closely follow the imaginary line drawn

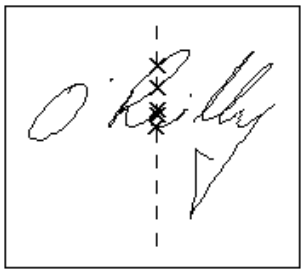

Fig. 16. *The dotted line represents a single vertical projection, one of many used in the calculation of vertical overlap. The crosses are the points of intersection between the vertical projection and the handwriting stream (there are five in this instance). The average number of intersections is then a measure of the degree of handwriting slant (the higher the slant the higher the number of in*

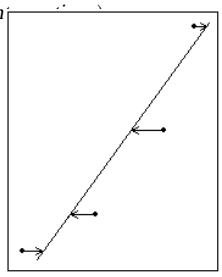

Fig. 17. *Stroke concavity is depicted in this figure, showing a close-up of a stroke segment with a line-of-best-fit drawn through four points. The concavity is then found by taking the square root of the sum of squares of the minimum distance from each point in the stroke to the line-of-best-fit.*

from the stroke start-point to the end-point. Calculation is performed using linear regression on the points in the stroke to obtain the line-of-best-fit. This measures how well the points in the stroke "fit" or approximate that line. Next, the following formula is applied to each stroke:

$$Stroke\ Concavity = \sqrt{\sum_{i=1}^{n}(s_i - r_i)^2}$$

where:

- n is the number of points in the stroke;
- s_i is the i^{th} point in the stroke;
- r_i is the coordinate along the line-of-best-fit that is the least distance from s_i.

The stroke concavity is taken as the mean of the individual concavity values for each stroke in the sample. (See Figure 17.)

Horizontal Velocity: This is the average velocity over the x direction. It measures how fast the signature moves horizontally and is related to pen-tip velocity, cursivity, horizontal length and acceleration. It is impossible for a potential forger to discern the horizontal velocity from an off-line copy of the writing. This feature is calculated as the ratio of horizontal distance to the duration in which the sample's body was produced (artifacts and the duration associated with their production are removed from the calculation). (See Figure 18.)

Mean Pen-Tip Pressure: This measures the amount of vertical pressure being applied by the pen to the top of the tablet. This is an option available on almost all current tablet and stylus hardware and is typically measured by an accurate sensor in the pen's tip. Other verification

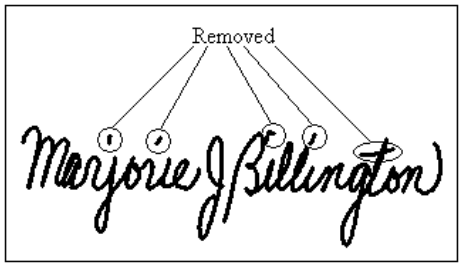

Fig. 18. *Depending on the feature used, it may be necessary to remove certain pixels from calculation. Typically, fragmented information such as the dotting of 'i's and the crossing of 't's are removed.*

software makes use of more complicated characteristics such as the breakdown into a horizontal and vertical component of the pen-tip pressure or the breakdown of the angle at which the pen is held. Because of the fact that angular pressure breakdowns are unavailable in much of the hardware, this verification system has been restricted to the assumption of a single $pressure$ profile. Additionally, if pressure values are unavailable, all pen-down occurrences have pressure set to one and pen-up occurrences set to zero.

Pressure, like most features used in this system, is very difficult for a forger to discern from an off-line copy of the handwriting. Although pen-tip pressure is less stable than other dynamic features such as velocity, it is included because of the difficulty that a potential forger has in accurately simulating the pressure profile. Mean pen-tip pressure used in this feature is simply the average of all non-zero values in the pressure profile.

Standard Deviation of Pen-Tip Pressure: This gauges how much a writer typically varies his/her pen-tip pressure during signing. This is calculated as the standard deviation of the non-zero values in the pressure profile.

Maximum Pen-Tip Pressure: This is the highest value in the pressure profile. This can not be extracted from an off-line copy of the writing and, while not as repeatable as velocity, is still useful in combination with other features.

Minimum Pen-Tip Pressure: This is the lowest non-zero value in the pressure profile. This feature is not able to be extracted from off-line copies of the writing and previous researchers have found this feature to be quite useful [4].

Degree of Parallelism: This refers to the extent to which slant remains consistent throughout the entire sample. This is a feature intrinsic to a writer's natural handwriting and is a characteristic naturally produced without conscious thought. This feature's main problem is that users tend to write with higher parallelism if they are forcing themselves to write slower and more deliberately.

Calculation is based on the handwriting slant. Long strokes are extracted and the standard deviation of the slant of the long strokes is obtained (a higher value indicates a lower slant consistency). (See Figure 19.)

Baseline Consistency: The baseline of a single handwritten word is the line-of-best-fit drawn through the bottom of all non-descender characters. The baseline is analogous to the position of the line when a user is signing on a ruled or dotted line. This is another very personal feature and is particularly representative of a

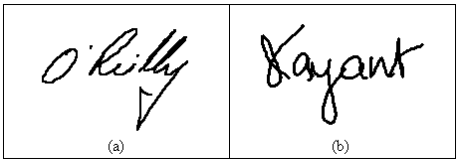

Fig. 19. *Different degrees of parallelism. (a) and (b) are two sections of signatures taken from different authors with different values for parallelism. Part (a) has a parallelism value of 0.10, whereas (b) has 0.34.*

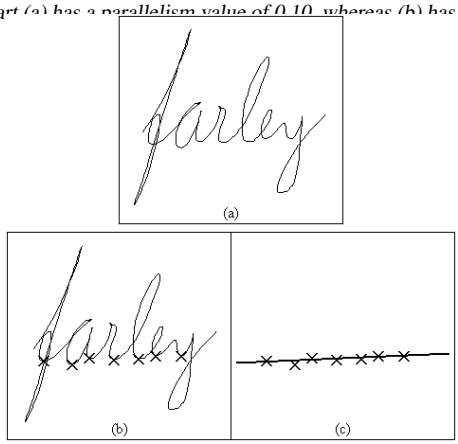

Fig. 20. *The various stages in the calculation of baseline consistency. (a) shows the original handwritten word, (b) shows the extracted minima for non-descender characters and (c) shows the line-of-best-fit calculated for these points using linear regression. The baseline consistency is then the square root of the sum of the squares of the distances between the extracted minima and the line. The baseline consistency of this handwriting sample is 25.2.*

signer's natural tendency when no ruled line is present as a guide. Some writers are naturally more irregular or "sloppy" when forming their baseline than others.

Calculation is done by extracting the set of minima from all non-descender characters (i.e., y-minima that fall *below* the mean of the y data and *above* the lower quarter). Linear regression is performed using these points and the line-of-best-fit is found (See Figure 20). The baseline consistency is given by the formula:

$$Consistency = \sqrt{\sum_{i=1}^{n}(b_i - r_i)^2}$$

where:

- n is the total number of baseline minima extracted;
- b_i is the i^{th} point in the set of baseline minima;
- r_i is the coordinate along the line-of-best-fit corresponding to the x value of b_i.

Ascender-line Consistency: The ascender-line is the line-of-best-fit drawn through the upper extremity of ascender characters such as 't', 'b' and 'd' and ignoring fragments such as 't' crossings and 'i' dottings (as initially used in [1]). Given the ascender-extremity points, ascender-line consistency calculation is done is the same fashion as the baseline consistency calculation.

Circularity: This feature tries to capture how "round" or "distended" the handwritten characters are. It is measured as the ratio of the *area* to the horizontal length. This feature is one that is quite difficult for a forger to judge so proves useful in preventing false acceptances.

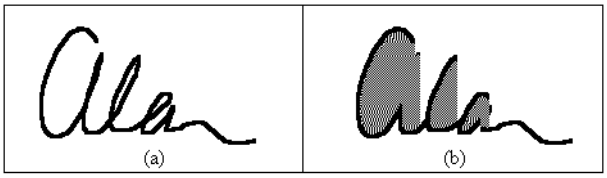

Fig. 21. *The area of a signature. (a) shows the original sample and (b) shows the calculated area. In (b) the black lines represent the vertical extremities (the maximum and minimum intersections with vertical projections) and the shading shows the area of the signature segment. The area of signatures with multiple components is found by summing each of the independently calculated component areas.*

Circularity is a computationally expensive feature as it requires several iterations through the x and y profiles. The first step in the calculation is to extract the various components (areas of connected handwriting) within the sample. Calculation of circularity is then done separately on each of these components as described below.

For each x value in the component, a vertical projection is taken and the position of each of the points of intersection with the handwriting is found. One of the main difficulties in this calculation (and also with the vertical overlap feature) is in determining exactly when this intersection occurs. The problem is that in an on-line handwriting system there may not be an *actual* intersection between the projection and one of the recorded points (it is less likely that a direct intersection will take place). It is therefore necessary to perform an iteration through the recorded points in the current component and deem that an intersection has occurred if and only if there are two consecutive points for which:

$$(x[i] < x_p)\ and\ (x_p < x[i+1])$$

where:

- x_p is the x value of the vertical projection (the line with the equation $x = p$);
- $x[i]$ is the i^{th} point in the x data stream;
- $x[i+1]$ is the $(i+1)^{th}$ point in the x data stream.

The next step is to determine the height, (i.e., the y value, of the intersection). This is found as follows:

$$y_{int} = \left(\left(\frac{x_p - x[i]}{x[i+1] - x[i]}\right) \times (y[i+1] + y[i])\right) + y[i]$$

where y_{int} is the y coordinate of the point of intersection between the handwriting and the projection. For each projection, the heights of the various intersections are found and the difference in height between the uppermost intersection and the lowermost intersection is found and added to the cumulative total for area. If there is only one intersection (i.e., a joining or trailing stroke) then one is added to the area value for the component (the area of a single horizontal line is defined to be one). Once projections are made for all x values in the component the area of that component is known. The ratio of the summed area to the summed horizontal length across all components is calculated and gives circularity. (See Figure 21.)

Area: This is the actual area of the handwritten word. Calculation is performed as a part of the *circularity* calculation and is shown in Figure 21 (the value is retained from the circularity calculations).

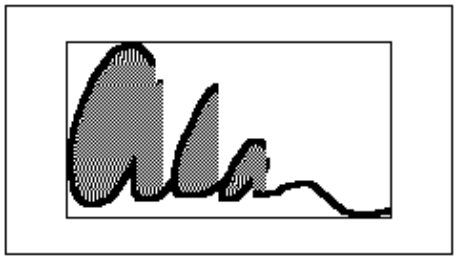

Fig. 22. *"Middle-heaviness". The bounding box is shown, and all shaded pixels are points interior to (or part of) the sample. The area of the shaded pixels is divided by the bounding box area to give middle-heaviness.*

Fig. 23. *Component Physical Spacing. When there is only one component, a default value of zero is returned.*

Middle-Heaviness: This is the percentage of the handwritten samples bounding box that is interior to the signature itself. It measures the concentration of the handwriting around the midpoint as opposed to featuring high ascenders and low descenders. Calculation is undertaken by finding the signature's area (as performed previously) and dividing this value by the area of the bounding box. The bounding box is a rectangle drawn around the sample using the two extremities in each of the x and y data streams (with artifacts removed). (See Figure 22.)

Component Physical Spacing: The average spacing between the components is again indicative of a writer's natural style and is very stable across multiple instances of the signature. Calculation involves taking the Euclidean distance between the last point sampled in a component and the first point sampled in the following component (if any). This value is calculated for each pen-up instance and averaged to obtain the final feature value (see Figure 23).

Component Time Spacing: This is the average duration of a pen-up instance in a signature (often referred to as pen-up time). It is slightly less stable than physical component spacing, but it is impossible for a forger to copy this feature from an off-line signature image.

III. Experimental Setup

This section details the experimentation performed as part of the NN HSV system development. It is split into three parts: a discussion of the benchmark linear network results, the multi-layer perceptron development (including the structure and network parameters resulting in the most robust and accurate system) and finally the different training approaches (including the classification error rates and timing results using different training algorithms and different compositions of the training set). In all experiments, five genuine signatures are used to train the network unless otherwise stated. The following performance metrics are used throughout this section:

- *FAR:* false acceptance rate expressed as a percentage.

- *FRR:* false rejection rate expressed as a percentage.
- *OER:* overall error rate (FAR + FRR).

A. Linear Network Development

Linear networks are NNs with no hidden layers or nodes. Despite limitations, these networks provide a useful benchmark against which to measure more complex techniques. It is common to find that problems perceived to be quite difficult are handled well by linear approaches. To this end a linear network was developed and trained to verify signatures. Training was done using the pseudo-inverse technique, five genuine signatures and various combinations of negative examples. (See Figure 24(a).)

This network's performance is quite reasonable, with the lowest OER achieved being 6.1% (3.4% FAR and 2.7% FRR) using a set of ten other user's genuine signatures as negative examples. These performance rates imply that the signatures used in this study are somewhat linearly separable via the extracted features. The network's stability is quite poor in relation to that displayed by networks with hidden layers.

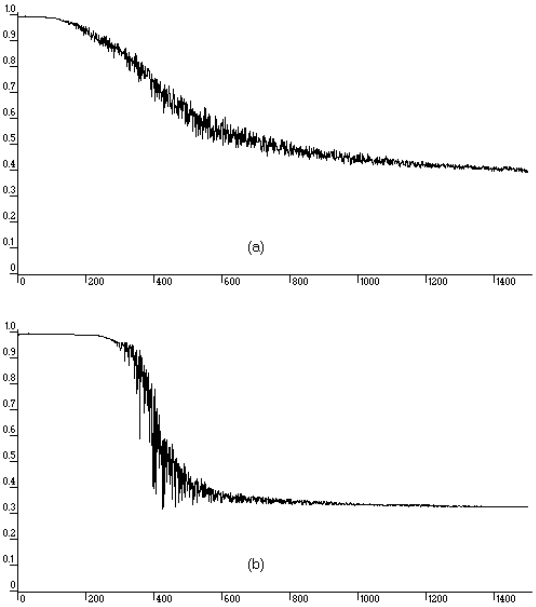

Fig. 24. *Convergence of training and verification errors. (a) In a linear network. (b) In a multi-layer perceptron with a single hidden layer.*

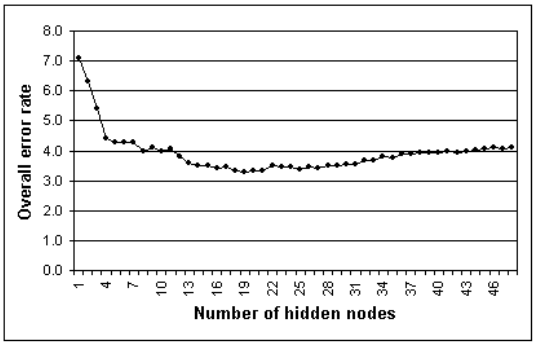

Fig. 25. *Error rates resulting from varying the number of hidden nodes in a MLP with one hidden layer.*

B. Multi-Layer Perceptron Development

Most of the experimentation in the NN development involved MLPs as this structure is well suited to the parametric HSV problem. The structural experimentation is constrained to fully-connected multi-layer feed-forward networks. That is, the networks have a distinctly layered structure, all nodes in layer i have connections to all nodes in layer $i+1$ and no connections to previous layers.

The input layer always consists of n nodes where n is the number of features in the set and the output layer consists of a single node that calculates the weighted sum of the connections coming into it. The network's final output is a confidence value indicating the likelihood that the test signature was performed by the same person that provided the reference signatures used in training. The confidence value is compared to a threshold and the test signature is verified if the confidence exceeds this threshold or rejected otherwise.

Training via the back-propagation algorithm, the MLPs are able to fit genuine signatures much better than linear networks. The result is a lower OER and a more pronounced convergence than was achieved with the linear network (in terms of training the network, see Figure 24) and a clearer class separation between the two classes.

The following discusses the different network parameters and architectural issues explored during experimentation, followed by an examination of MLPs with two hidden layers and the different training scenarios.

- *Number of Nodes in the Hidden Layer*: This is the most influential adjustable parameter within the model constraints. Architecture determination can be treated as an optimization problem exploring various possible designs looking for the most suitable structure. In a MLP with one hidden layer, once the training features are decided upon, the optimization problem reduces to a decision over the number of units in the hidden layer. This is a linear search of a noisy function (each time a network is trained a slightly different error rate results). In the NN HSV system, the search involved imposing a minimum of two and a maximum of 120 (roughly three times the number of input units) on the number of hidden units and experimenting exhaustively within this range. Values between fifteen and thirty proved to be the most successful (in terms of OER) with nineteen hidden nodes used in any further NN experimentation. Figure 25 plots the number of hidden nodes versus the resulting OER.
- *Learning Rates*: The NN learning rate controls the speed with which the network learns. A higher learning rate causes the algorithm to converge faster but may introduce instabilities, especially if the data is noisy. Experimentation involved exhaustively searching all learning rates from 0.05 to 0.95 with increments of 0.5. Small changes in the learning rate did not have a significant influence on the final error rates and a value of 0.6 produced consistently acceptable results. Degradation occurred when the learning rate moved closer to zero or one.
- *Momentum*: Small changes in the momentum value

TABLE II
The training performance of three network structures.

Structure	Number of Epochs	Relative Time
Linear Network	1350	7.2%
MLP with 1 Hidden Layer	1000	60.1%
MLP with 2 Hidden Layer	1200	100.0%

did not have a large impact on system accuracy. A value of 0.3 was used in the final system.

- *Activation Functions*: Sigmoidal activation functions were used in each of the NN nodes as they are generally more appropriate for HSV applications.

The final OER using the above structure and network parameters and the most successful structure was 3.3% (2.0% FAR and 1.3% FRR), which represents a clear improvement over linear techniques.

MLP With Two Hidden Layers: Experiments were conducted using MLPs with two hidden layers to better fit the signatures. The RMS error for the genuine signatures was lower than for the single-hidden-layer MLP as the extra hidden layer enabled the underlying function to be more closely approximated. Experiments were conducted using various combinations of unit numbers in the two hidden layers. It was found that the results are best when the number of units is equal to around half the number of input units. The momentum value and the activation functions are the same as for the MLP with one hidden layer. The learning rate differed slightly and a setting of 0.3 resulted in the most stable network using the same training set contents (training sets are described below).

This more complex model has a detrimental effect on the network performance (in terms of classification error and processing speed). The network execution (classifying a test signature) is slightly slower and the training procedure takes on average double the time to converge. Table II presents an example of the time spent training the three network structures. The "number of epochs" is a measure of how many iterations each of the structures requires for convergence. This is less relevant than the overall time requirements (e.g., a training epoch for a linear network is very different to a training epoch for a multi-layer perceptron with two hidden layers). The relative time is the average time taken to converge, expressed as a percentage of the time taken by the two-layer MLP. The network's overall classification accuracy deteriorated slightly as a result of the initial hidden layer. The network is able to fit the genuine signatures in the training set very closely, but over fitting seemed to occur as the FRR was increased. The final OER using two hidden layers is 3.8% (1.3% FAR and 2.5% FRR).

C. Training Approaches

Experimentation included different training algorithms, different training sets and the use of forgeries (both skilled and zero-effort) in the training set. The experimentation and error rates apply to a MLP structure with one hidden layer unless otherwise specified.

The first training aspect under consideration is the algorithms to perform the actual weight adjustments. The performance of each of the three implemented training algorithms is discussed below.

TABLE III
The classification accuracy of the three implemented neural network training algorithms.

Training Algorithm	FAR	FRR	OER
Back-propagation	2.0%	1.3%	3.3%
Conjugate Gradient Descent	1.9%	2.4%	4.3%
Levenberg-Marquardt	2.4%	1.6%	4.0%

TABLE IV
The convergence speed of the three implemented NN training algorithms.

Training Algorithm	Number of Epochs	Relative Time
Back-propagation	1000	4.6%
Conjugate Gradient Descent	147	0.8%
Levenberg-Marquardt	1280	100.0%

- *Back-propagation*: This is the most widely used algorithm in NN training due to its efficiency, simplicity and performance. Back-propagation has been used successfully in many different environments, and results in the highest classification accuracy compared to the other implemented algorithms. The performance details using back-propagation are presented in Table III (error rates) and Table IV (training time).
- *Conjugate Gradient Descent*: This is the major alternative to back-propagation but is not as widely used in HSV. The conjugate gradient descent algorithm converged much faster during training than back-propagation. Unfortunately, due to the fact that this training algorithm is suited to more complex networks (with several hundred weights), the resulting error rate was somewhat worse than networks trained via back-propagation. (See Tables III and IV.)
- *Levenberg-Marquardt*: The classification accuracy of the Levenberg-Marquardt training algorithm was slightly better than that obtained using conjugate gradient descent but not as good as back-propagation. However, the training process was exceedingly slow, which may be a problem in a large, dynamic database. (See Tables III and IV.)

The back-propagation algorithm seems to be superior in this HSV environment. Although it didn't converge as quickly as conjugate gradient descent, the classification accuracy was superior to the other two approaches and training error rates continue to improve after several hundred epochs. The actual "wall clock time" required for training to complete using back-propagation is typically between thirty seconds and two minutes. This training time is not prohibitively slow as it is generally a one-off cost and is not performed while the user waits.

As the training is unsupervised, a *stopping condition* is needed to ensure that training time is finite and that the resulting network is not under- or over-trained. Experiments were conducted using heuristic approaches with different limitations and slightly different rules. The most successful method (and the method used in all training scenarios) is a two-fold rule of stopping if the RMS error reduces to below 0.05 or if this error fails to improve over a span of ten epochs. The first half of the condition

ensures the training stops when error rate is low enough to be considered effective, and in a finite number of epochs. On some training runs, the RMS error continued to reduce slowly for hundreds of thousands of epochs, so this element of the stopping condition protects against this occurrence. The second half of the condition is necessary as sometimes the RMS error reached a point beyond which it did not improve, irrespective of further time spent training. Ten epochs was found to be a large enough period of time to wait for at least *some* improvement. Note that if there is a deterioration of the error rate after a minimum has been reached, training will eventually stop due to the "n epochs without improvement" condition. Rather than use the now slightly deteriorated network, the attributes (weights) of the network revert to those which produced the lowest error rate.

One other point to note is that no limit is placed on the number of epochs spent training – in experimentation, the above stopping condition caused the training of the network to cease in a reasonable amount of time (a maximum of a number of minutes) in every instance. In practice it may be necessary for some (extremely large) maximum value to be placed on the number of epochs to prevent an exceedingly long training phase for a network that is regularly slightly improved. This was not found to be the case in any of the experiments conducted, but this is by no means proof that the scenario will never happen.

The above stopping condition was used in all of the training scenarios described below unless otherwise specified. What follows now is a discussion of the different scenarios with respect to the makeup of the training set.

Training One Network with the Entire Database

This scenario involved training on the signature database as a whole, using forty-one input units (one for each feature), various different experiments with the size of the hidden layer and a single output unit (attempting to identify the signature's author). The target output value was the identifier for the signer who provided the signature. That is, a single network was trained to identify the most closely matching user in the database for a given signature. This strategy was not greatly successful (it makes the assumption that the entire database of signatures is separable using a single set of weights) and the network displayed a large amount of confusion, resulting in an OER of over 47%. Additionally, this approach is impractical in that the training takes a very large amount of time and re-training is necessary each time the database population changes.

Training One Network with the Entire Database and Multiple Outputs

This scenario works as above, except the number of output nodes is equal to the number of users. When the network is presented with a test signature, the output node with the highest activation value is deemed the "winning" node and the corresponding user deemed to the be the test signature's author. This is a multi-class classification problem and is often referred to as *one-of-N* classification. Slight improvements are achieved using this approach

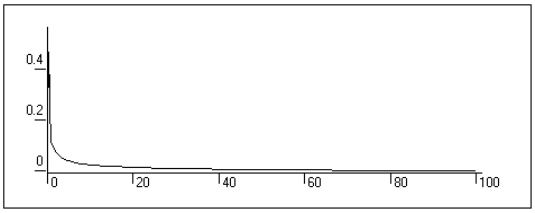

Fig. 26. *Convergence of error rate using back-propagation in a typical MLP with no negative examples.*

(an OER of just under 45%), but the accuracy is still significantly inadequate for use in HSV. In addition, the problem remains of re-training the network when a new user registers. The conclusion drawn from this is that both of the first two scenarios are impractical for HSV.

Training Individual Networks for Each Signer

This scenario involved creating and training a single network for each individual, training using a randomly selected set of five genuine signatures for each user, with the remaining genuine signatures used in the verification set. This approach is very simple, fast and easily achievable (no forgeries are required). The problem is that the lack of negative examples in the form of forgeries (either skilled or zero-effort) makes the training process a somewhat haphazard approach as it is difficult to know with any certainty when to stop training (the most successful stopping condition was found to be simply to place a limit on the number of epochs). Error convergence occurs very quickly (see Figure 26) but the typical under-fitting causes problems when the network is tested using forgeries.

Classification of previously unseen genuine signatures was performed well (i.e., there were very few false rejections), but classification of forgeries was quite poor (false acceptances were commonplace). However, this is expected, as it is difficult for a network to recognise forgeries if it has not seen any before. The OER using this approach is 42.5%, made up largely from false acceptances (31.2% FAR and 11.3% FRR).

Training Individual Networks for Each Signer using Negative Examples

This training approach produced the lowest OER. Again a separate network was trained for each individual signer in the database. However, to improve the resulting system's accuracy, negative examples (in the form of forgeries) are introduced into the training set (five reference signatures per signer are still used as the positive examples). Without the use of negative examples it is difficult to obtain effective threshold values on the NN output, and the network is less likely to recognise a forgery. Ideally the negative examples would be skilled forgeries, but these are rarely available in practical situations so an alternative must be found. The alternatives explored with respect to negative examples are presented below. Table V presents the classification accuracy for all approaches.

1) The brute force approach is to use the entire set of reference signatures from other users in the database. This necessitates a change in the approach to training as genuine signature misclassifications can become insignificant in the makeup of the

overall RMS error (as there are many more forgeries than genuine signatures). The modification involves applying a similar concept to the more powerful "loss matrix" and specifically associating a more severe penalty with misclassifying genuine signatures. The results obtained using this approach were quite good, but this is impractical due the increased training time (training time increases linearly with respect to the number of users in the database) and the need to re-train the system every time a new user is added to the database. An OER of 3.2% is obtained using this approach.

2) To reduce the training time and the necessity to re-train, experiments were conducted using reference signatures from other randomly selected users as forgeries. Sets of ten, fifteen, twenty, twenty-five and thirty users were considered with the best results occurring with set sizes of twenty and twenty-five. When a new user is added to the database their negative training examples are extracted from the users that are already present. Existing networks are not re-trained when a new user is introduced. The OER, averaged over ten applications of the random approach, is 7.6%.
3) The random approach relieves the impracticalities of the brute force approach, but the resulting error rate is too high. Many of the selected signatures bear little or no resemblance to the reference signature they are trying to forge. As a result the negative examples are much less effective. In an attempt to create more effective negative examples, random noise (in the form of spatial resampling and geometric translation) is applied to the reference signatures. The amount of noise was varied during experimentation, but no combination made this approach successful. The most successful classification rates are obtained when noise is applied to 25% of the sampled points. However, the network still exhibits some confusion in this case. An OER of 16.1% is the result.
4) The final (and most successful) attempt at selection of appropriate signatures to use as negative examples is handled using a basic, very fast similarity measure from [11]. This is based on string edit distance. The nearest one hundred signatures according to this similarity measure are included as negative examples (the hundred signatures typically came from around twenty-five different signers). Due to the large number of forgeries a misclassification penalty (doubling the RMS error) is applied to false rejections. As the entire database needs to be searched when adding a new user, the training process takes longer. Fortunately the total cost to the training process is not great and effectively lengthens training by around thirty seconds per hundred users in the database (on Sun SparcStation 20) to find the similar signatures. This approach is realistic as the training time is not excessive, and re-training of existing networks is not required when enrolling new users. This approach's performance rivals the brute force approach at 3.3% OER.
5) For interest sake and comparative purposes, training was done using three skilled forgeries. Use of skilled forgeries in the training set results in a large improvement in error rates. The OER using three skilled forgeries in training is 2.3%.

TABLE V
The HSV system's performance using different approaches to obtaining negative examples.

Approach	**FAR**	**FRR**	**OER**
1. All other reference signatures	2.4%	1.8%	3.2%
2. Random reference signatures	5.2%	2.4%	7.6%
3. Introduced noise	7.3%	8.8%	16.1%
4. Similarity measure	1.1%	2.2%	3.3%
5. Skilled forgeries	0.7%	1.6%	2.3%

TABLE VI
The HSV system's performance using different MLP structures.

Structure	**FAR**	**FRR**	**OER**
One MLP, single output unit	19.3%	27.9%	47.2%
One MLP, one output unit per signer	25.4%	19.5%	44.9%
One MLP per signer, no forgeries	31.2%	11.3%	42.5%
One MLP per signer, with forgeries	1.1%	2.2%	3.3%
One MLP per signer, two hidden layers, with forgeries	1.2%	2.5%	3.7%
One linear network per signer, best case	3.4%	2.7%	6.1%

Table V summarises the results using the above approaches for creating negative training examples. Approach five (using skilled forgeries) is the most successful, followed by using the entire set of genuine signatures from the database. While these approaches are both impractical, the performance of approach four (using other signatures that are "close" according to a similarity measure) still performs very well and is does not suffer from practicality issues. This approach is used to train the final version of the NN based HSV system.

In summary, the final (most successful) version of the NN based HSV system uses a single MLP with one hidden layer to model each user's signature and is trained using five genuine signatures and one hundred zero-effort forgeries (selected using a similarity measure). Table VI presents a summary of results using various MLP structures. Overall results deteriorated slightly when using two hidden layers in the MLP, which indicates that perhaps over-fitting is occurring as the training results do not generalise as well to the unseen signatures. Another aspect that can be seen in the summary is that the performance using a single MLP trained to *identify* any signer within the database results in quite a high OER. The poor results indicate that a single, global weight vector is insufficient for class separation in HSV systems with a large number of users. What is surprising is the low error rate obtained when a linear network is used. This suggests that the signatures in this database (through the extracted features) are somewhat linearly separable.

Figure 27 shows the HSV system's performance using different threshold values. Included in the figure are the FAR, FRR and OER. Not included is the *equal error rate*,

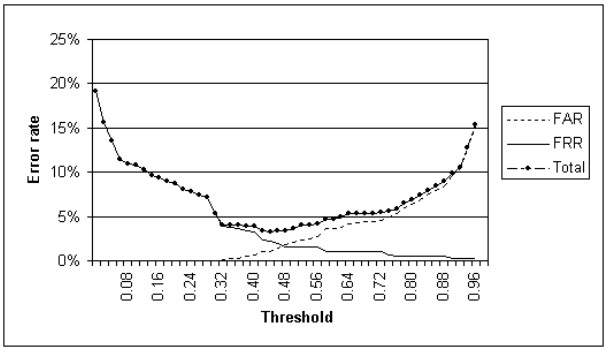

Fig. 27. *The performance of the optimal network structure using different threshold values.*

which is the point at which the FAR and FRR are of equal value (i.e., when the lines on the graph cross) – this value being 1.9%, occurring at a threshold of 0.51 (using one MLP per signer, trained with zero-effort forgeries). The lowest OER was 3.3% (1.1% FAR and 2.2% FRR) occurring at a threshold of 0.42.

IV. Conclusions

This paper presents a method for verifying handwritten signatures by using a NN architecture. Various static (e.g., height, slant, etc.) and dynamic (e.g., velocity, pen tip pressure, etc.) signature features are extracted and used to train the NN. Several Network topologies are tested and their accuracy is compared. The most successful version of the NN based HSV system uses a single MLP with one hidden layer to model each user's signature. It is trained using five genuine signatures and one hundred zero-effort forgeries. Using this approach, a 3.3% OER is reported for the best case.

Future work entails creating a NN-Markov Model hybrid system for HSV. Furthermore, it would be useful to apply the approach outlined in Woo and Dlay [15]. When a NN is employed, its structural complexity must be regulated by a progressive approach that is verified by various error rates. [15] suggests automating this process by constructing a penalty term in the cost function that ensures that the weights in the NN are optimised such that they follow a sparse distribution.

References

[1] Y. Bengio, Y. Le Cun, C. Nohl and C. Burges. *LeRec: A NN/HMM Hybrid for On-Line Handwriting Recognition.* Neural Computation, Vol. 7, No. 5, 1995.

[2] J. Cai and Z-Q. Liu. *Integration of Structural and Statistical Information for Unconstrained Handwritten Numeral Recognition.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 21(3), pp 263–270, 1999.

[3] H.D. Crane and J.S. Ostrem. *Automatic Signature Verification Using a Three-axis Force-Sensitive Pen.* IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-13, No. 3, pp 329–337, 1983.

[4] J.G.A. Dolfing, E.H.L. Aarts and J.J.G.M. van Oosterhout. *On-Line Signature Verification with Hidden Markov Models.* Proceedings of the 14th International Conference on Pattern Recognition, Brisbane, Australia, pp 1309–1312, 1998.

[5] L. Fausett. *Fundamentals of Neural Networks.* Prentice Hall Publishing, New York, 1994.

[6] P. Gallinari, S. Thiria, F. Badran and F. Fogelman-Soulie. *On the Relations between Discriminant Analysis and Multilayer Perceptrons.* Neural Networks, Vol. 4, pp 349–360, 1991.

[7] K. Phua, J. Chen, T. Huy Dat and L. Shue. *Heart Sound as a Biometric.* Pattern Recognition, Vol. 41, Issue 3, pp 906–919, 2008.

[8] L.L. Lee. *On-Line Systems for Human Signature Verification.* Ph.D. Thesis, Cornell University, 1992.

[9] D.Z. Lejtman. *On-line Handwritten Signature Verification Using Wavelets and Back-propagation Neural Networks.* Proceedings of the Sixth International Conference on Document Analysis and Recognition, pp 992, 2001.

[10] L. Ma, T. Tan, Y. Wand and D. Zhang. *Personal Identification Based on Iris Texture Analysis.* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 25, No. 12, pp 1519–1533, 2003.

[11] A. McCabe and J. Trevathan. *Implementation and Analysis of a Handwritten Signature Verification Technique.* In Proceedings of the 2nd International Conference on Security and Cryptography (SECRYPT), pp 48–58, 2007.

[12] S.K. Modi and S.J. Elliott. *Keystroke Dynamics Verification Using a Spontaneously Generated Password.* In Proceedings of the 40th IEEE International Carnahan Conference on Security Technology, Lexington Kentucky, 2006.

[13] R. Plamondon. *The Design of an On-Line Signature Verification System: From Theory to Practice.* Series in Machine Perception and Artificial Intelligence, Vol. 13, pp 155–172, 1993.

[14] L. Vuurpijl and L. Schomaker. *Coarse Writing-Style Clustering Based on Simple Stroke-Related Features.* Progress in Handwriting Recognition, Colchester, 1996.

[15] W.L. Woo and S.S. Dlay. *Regularised Nonlinear Blind Signal Separation using Sparsely Connected Network.* IEEE Proceedings on Vision, Image and Signal Processing, Vol. 152, No. 1, pp 61-73, 2005.

[16] X. Xiao and G. Leedham. *Signature Verification by Neural Networks with Selective Attention.* Journal Applied Intelligence, Vol. 11, No. 2, pp 213-223, 1999.

[17] J. Zhang, W.L. Woo and S.S. Dlay. *Blind Source Separation of Post-Nonlinear Convolutive Mixture.* IEEE Transactions on Audio, Speech and Language Processing, Vol. 15, No. 8, pp 2311-2330, 2007.

[18] K.P. Zimmermann and M.J. Varady. *Handwriter Identification from One-bit Quantized Pressure Patterns.* Pattern Recognition, Vol. 18, No. 1, pp 63-72, 1985.

[19] E. Zois and V. Anastassopoulos. *Methods for Writer Identification.* Proceedings of the Third IEEE International Conference on Electronics, Circuits and Systems (ICECS), 1996.