ISSN: 2321-7782 (Online)
Volume 3, Issue 6, June 2015
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

Signature Recognition and Verification using ANN

Ketki S. Muzumdar[1]
Computer Science & Engineering,
Prof. Ram Meghe Institute of Technology & Research,
Badnera, Amravati, India

Vrushali S. Sakharkar[1]
Computer Science & Engineering,
Prof. Ram Meghe Institute of Technology & Research,
Badnera, Amravati, India

Abstract: ***Signatures are composed of special characters and flourishes and therefore most of the time they can be unreadable. Also interpersonal variations and the differences make it necessary to analyze them as complete images and not as letters and words put together. As signatures are the primary mechanism both for authentication and authorization in legal transactions, the need for research in efficient automated solutions for signature recognition and verification has increased in recent years. Various methods have already been introduced in this field and application of Statistical models is one of them in this regard. Many signatures can be unreadable. They are a kind of artistic handwriting objects. However, a signature can be handled as an image, and hence, it can be recognized using computer vision and artificial neural network techniques.***

Keywords: signature recognition; image processing; ANN

I. INTRODUCTION

Pattern recognition can be done both in normal computers and neural networks. Computers use conventional arithmetic algorithms to detect whether the given pattern matches an existing one. It is a straightforward method. It will say either yes or no. It does not tolerate noisy patterns. On the other hand, neural networks can tolerate noise and, if trained properly, will respond correctly for unknown patterns. Neural networks may not perform miracles, but if constructed with the proper architecture and trained correctly with good data, they will give amazing results, not only in pattern recognition but also in other scientific and commercial applications[12].

For person's identity various forms of biometric securities exists, such as finger print recognition, iris recognition, speech recognition, heart sound recognition and key stroke recognition. But the simplest method for verifying the person's identity is through the use of handwritten signature. The signature represents a personal style of human handwriting. Various projects have been carried out in the past on signature verification with varying degrees of success. Azriel Rosenfeld has done static signature work. In the Dynamic signature field a number of groups have taken several approaches. By using artificial neural network it is easier to implement a system to recognize and verify the signature. In neural network we have to train the signature by using various algorithms such as multilayer perceptron, feed forward algorithm or back propagation algorithm, etc so that the system gives the result efficiently[5].

The aim of this work is handwritten signature identification and verification. The work deals with recognition of number of sample signatures using artificial Neural Network. First of all, the user is required to register five signatures for storage and training of the neural network. In case the user has already registered, he/she is prompted to select his/her identity as contained in the database. The next step will be the preprocessing of the signatures received. The third stage is to carry out features extraction procedure from the given signature information which is later used for training Neural Network. The fourth stage deals with the process of comparing the signatures submitted with reference signatures stored in database for verification using Neural Network method. The final step is the storage of the processed signature in the database and the performance evaluation to verify the user signature.

The first part is the registration procedure in which the user is required to provide the handwritten input signature. The information obtained shall provide the essential raw data to the computer database. User is prompted to sign his/her signature five times for the initial data acquisition phase and training using neural network.

The second part is the preprocessing of the signatures involving the use of suitable algorithms. Next part involves the actual neural network training and identifying process of the input signature patterns that will yield suitable weights and biases. Storing of the processed information in a database immediately follows this. Finally, test signature pattern from the user shall be verified (also after being preprocessed) using the proposed verification algorithms with the application of neural network method taking into account some tolerance bands and threshold values. If the test pattern matches those in database with respect to a number of parameters, it shall be accepted otherwise it will be outright rejected[3].

II. System Overview

First of all, the user is required to register five signatures for storage and training of the neural network. In case the user has already registered, he/she is prompted to select his/her identity as contained in the database. The next step will be the preprocessing of the signatures received. The third stage is to carry out features extraction procedure from the given signature information which is later used for training Neural Network. The fourth stage deals with the process of comparing the signatures submitted with reference signatures stored in database for verification using Neural Network method. The final step is the storage of the processed signature in the database and the performance evaluation to verify the user signature.

The first part is the registration procedure in which the user is required to provide the handwritten input signature. The information obtained shall provide the essential raw data to the computer database. User is prompted to sign his/her signature five times for the initial data acquisition phase and training using neural network. The second phase is the preprocessing of the signatures involving the use of suitable algorithms. Next part involves the actual neural network training and identifying process of the input signature patterns that will yield suitable weights and biases. Storing of the processed information in a database immediately follows this. Finally, test signature pattern from the user shall be verified (also after being preprocessed) using the proposed verification algorithms with the application of neural network method taking into account some tolerance bands and threshold values. If the test pattern matches those in database with respect to a number of parameters (entropy, standard deviation, energy, average) it shall be accepted otherwise it will be outright /rejected[7].

The handwritten signature of the user is scanned and stored in JPEG format. Digital signal corresponding to scanned signature is a two dimensional signal. DFT is computed and 64 DFT coefficients are computed using Fast Fourier transform algorithm. For this, FFT command in Matlab is used. Here 64 DFT coefficients are used as parameters to train the neural network. In recognition mode parameters of the test signature are compared with the parameters in the database to find minimum distance. This comparison is done using neuro solution[5].

III. Image Processing

A. What is an image?

An image is an array, or a matrix, of square pixels (picture elements) arranged in columns and rows.

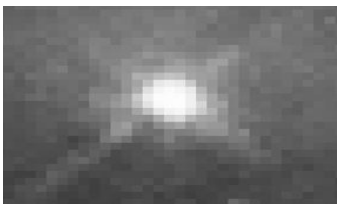


Figure 3.1: an array or a matrix of pixels arranged in columns and rows

In a (8-bit) grayscale image each picture element has an assigned intensity that ranges from 0 to 255. A grey scale image is what people normally call a black and white image, but the name emphasizes that such an image will also include many shades of grey [9].

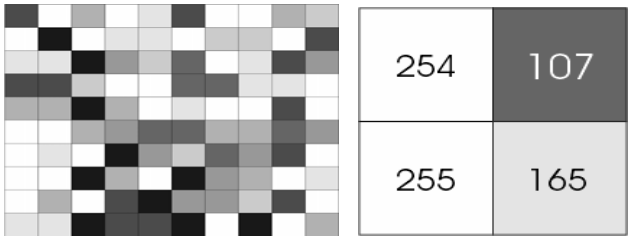


Figure 3.2 : Each pixel has a value from 0 (black) to 255 (white). The possible range of the pixel values depend on the color depth of the image, here 8 bit = 256 tones or grayscales

A normal grayscale image has 8 bit colour depth = 256 grayscales. A "true colour" image has 24 bit colour depth = 8 x 8 x 8 bits = 256 x 256 x 256 colours = ~16 million colours.

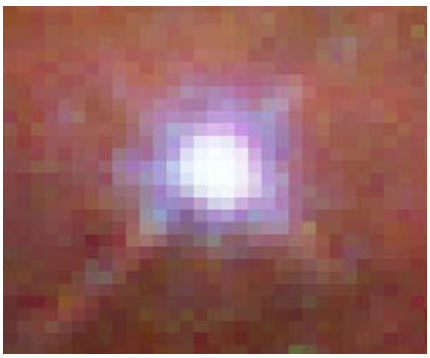


Figure 3.3 : A true-color image assembled from three grayscale images colored red, green and blue

Some grayscale images have more grayscales, for instance 16 bit = 65536 grayscales. In principle three grayscale images can be combined to form an image with 281,474,976,710,656 grayscales. There are two general groups of 'images': vector graphics (or line art) and bitmaps (pixel-based or 'images'). Some of the most common file formats are:

» GIF — an 8-bit (256 color), non-destructively compressed bitmap format. Mostly used for web. Has several sub-standards one of which is the animated GIF.
» JPEG — a very efficient (i.e. much information per byte) destructively compressed 24 bit (16 million colors) bitmap format. Widely used, especially for web and Internet (bandwidth-limited).
» TIFF — the standard 24 bit publication bitmap format. Compresses non-destructively with, for instance, Lempel-Ziv-Welch (LZW) compression.
» PS — Postscript, a standard vector format. Has numerous sub-standards and can be difficult to transport across platforms and operating systems.
» PSD – a dedicated Photoshop format that keeps all the information in an image including all the layers.

B. Image Processing

Image processing is any form of signal processing for which the input is an image, such as a photograph or video frame; the output of image processing may be either an image or a set of characteristics or parameters related to the image. Most image-processing techniques involve treating the image as a two-dimensional signal and applying standard signal-processing techniques to it.

Image processing usually refers to digital image processing, but optical and analog image processing also are possible. This article is about general techniques that apply to all of them. The acquisition of images (producing the input image in the first place) is referred to as imaging. Closely related to image processing are computer graphics and computer vision. In computer

graphics, images are manually made from physical models of objects, environments, and lighting, instead of being acquired (via imaging devices such as cameras) from natural scenes, as in most animated movies. Computer vision, on the other hand, is often considered high-level image processing out of which a machine/computer/software intends to decipher the physical contents of an image or a sequence of images (e.g., videos or 3D full-body magnetic resonance scans)[9].

In modern sciences and technologies, images also gain much broader scopes due to the ever growing importance of scientific visualization (of often large-scale complex scientific/experimental data). Examples include microarray data in genetic research, or real-time multi-asset portfolio trading in finance [9].

C. Digital Image Processing

Digital image processing is the use of computer algorithms to perform image processing on digital images. As a subcategory or field of digital signal processing, digital image processing has many advantages over analog image processing. It allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the build-up of noise and signal distortion during processing. Since images are defined over two dimensions (perhaps more) digital image processing may be modeled in the form of multidimensional systems [10].

D. Database Used

The handwritten signature of the user is scanned and stored in JPEG format. The samples of the signature from twenty persons are digitized through a signal conditional element that is already built-into the system. Later, this information will undergo or pass through a number of processes for future person needs to be acquired first before it is preprocessed. Sample signatures are shown below.

Figure 3.4: Samples of signatures

E. The Back Propagation Algorithm

The overall idea behind back propagation is to make a large change to a particular weight, w, if the change leads to a large reduction in the errors observed at the output nodes. For each sample input combination, you consider each output's desired value,0,and the influence of a particular weight, w, on the error,d-0. A big change to w makes sense if that change can reduce a large output error and if the size of that reduction is substantial. On the other hand , if a change to w does not reduce any large output error substantially, little should done to that weight. Most of the computation needed to compute the change to particular weight is also needed to compute the changes to weights that are closer to output nodes. Consider the net shown in figure 3.7.1 Once see how to compute a change for a typical weight, $wi \rightarrow j$, between a node in layer i and a layer in node j, you see taht the required computation involve computation needed for the weights, , $wj \rightarrow i$, Between nodes in layer j and nodes layer k [11].

First node that a change in change in the input to node j result in a change in output at node j that depends on the slope is steepest, a change in the input has the maximum effect on the output. Accordingly, you arrange for the change in $wi \rightarrow j$, to

depend on the slope of threshold function. Where the slope is steepest, a change in the input has the maximum effect on the output.

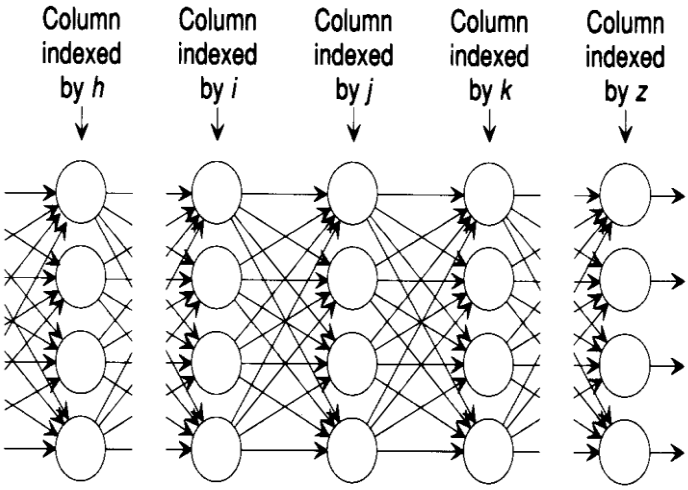


Figure 3.5: a trainable neural network

The slope of the squashed S function is given by a particular simple formula,0(1-0). Thus ,the use of squashed S function as the threshold function leads to the following conclusion about changes to $wi \rightarrow j$:

» Let the change in $wi \rightarrow j$ be proportional $oj(1 - oj)$.

Next ,the change in the input to node j, given in the weight , $wi \rightarrow j$, depends on the output of node i. Again, on the ground that change should be liberal only where it can do a lot of good, you arrange for $wi \rightarrow j$, substantially only if the output of node i is high:

» Let the change in the $\rightarrow j$, be proportional to oi to $oj(1 - oj)$, and to a factor that captures how beneficial it is to change output of node j? To make it easier to write things down ,let us agree that greek letter β stands for the benefit obtained by changing the output value of a node. Thus the change to $wi \rightarrow j$ should be proportional to $oi * oj(1 - oj) * Bj$.

Just how beneficial is it to change the output of node j? Imagine first that node j is connected to just one node in next layer namely ,node k.then the reasoning just completed can be reapplied with a slight modification:

» Because change should be liberal only where it can do substantial good,the change to oj

» For the same reason, the change to oj should be proportional to , $wj \rightarrow k$,the weight on the link connecting node j to node k.

The overall benefit obtained by changing oj must be sum of individual effects,each of which includes a weight , $wj \rightarrow k$, a slope $ok(1 - ok)$,and the factor βk . thus the benefit that you obtained by changing the output of node j is summarized as follows:

$$\beta j = \sum_{k} wj \rightarrow k\, ok(1 - ok)\beta k.$$

At this point, recall that the change to $wi \rightarrow j$ is proportional to βj. Evidently, the weight change in any layer of weights depend on a benefit calculation, βj ,that depends, in turn, on benefit calculations, βk, needed to deal with weights closer to the output nodes. To finish the analysis, you need to answer only one remain in question concerning the benefit you obtained by changing the value of an output node. this value depends, of course, on how wrong the output nodes value of an output node.

this value depends, of course ,on hoe wrong the output nodes value happens to be. if the difference between the dz, the desired output at node z, and oz,

The actual output at the same node, is small, then the change in the output at node z should be in proportion to the difference ,$dz - oz$. recasting the conclusion in the framework of benefit, we have the following:

$$\beta z = dz - oz.$$

Finally ,weight changes should depend on the rate parameter ,r, that should be as large as possible to encourage rapid learning, but not so large as to cause changes to the output values that considerably overshoot the desired values:

» Let the change in $wi \rightarrow j$ be proportional to a rate parameter, r, determined experimentally.

Combining all the equation, you have the following back-propagation formulas:

$$\Delta wi \rightarrow j = roi(1 - oj)\beta j,$$

$$\beta j = \sum_k wj \rightarrow k\, ok(1 - ok)\beta k \quad \text{for nodes in hidden layer.}$$

$$Bz = dz - oz \quad \text{for nodes in the output layer.}$$

Once you have worked out the appropriate change in the weights for one input combination, you face an important choice. Some neural set enthusiasts make changes after considering each sample input. Other add up the changes suggested by individual sample inputs and make actual changes only after all the sample inputs are considered[11].

» ***The Algorithm***

The algorithm used for this system is as follow :

1. Start
2. Randomly select the initial values of weight vectors $w^m ij$ for i=1,2,3,…., n and m=2 (number of layers).
3. Initialize all the weights $w^m ij$ were initialized to random number and given as $w^m ij(0)$

$$w^m ij \leftarrow w^m ij(0)$$

4. Calculate output of system

$$a^m ij = (w^1 ij)\ TxXk$$

$$Yi = (w^2 ij)\ Txa^m ij$$

5. Calculate error : Ej = Dj-Yj
6. Calculate derivative of network output of each layer.

$$F'(n) = \frac{d}{dn} \frac{1}{1+\exp^{-n}}$$

$$= (1-a^m ij)(a^m ij)$$

7. Back propagation of error by sensitivities at each layer.

IV. SYSTEM DESIGN

The architecture of our system is shown in figure 5.1.1. In which we firstly provide the signature then the image is cropped as required, then the feature extraction is applied on that signatures. After that the NN training is performed and stores them into the database. After that for verification we have to check the threshold of the signatures depending on that the I/O target is generated and result comes out in the form of success or failure.

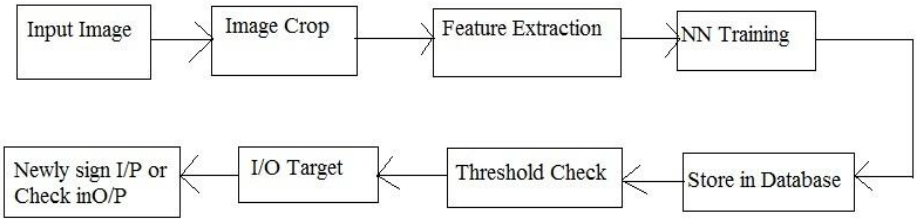


Figure 4.1: Architectural Model

V. SYSTEM IMPLEMENTATION

In implementation process, we implement the system in which perform various operations on signature such as noise reduction, background elimination and border clearing of signature, feature extraction, converting color image to gray scale image and the last one is digitization. Then train this signature using the back propagation algorithm.

Step1 : Firstly we perform noise reduction and background elimination on signature which looks like as shown in figure 5.1.

Figure 5.1: noise reduction and background elimination of signature

Step 2: Then we extract the features and convert the signature into grayscale, since we don't require the color signature. It will looks like as shown in figure 5.2.

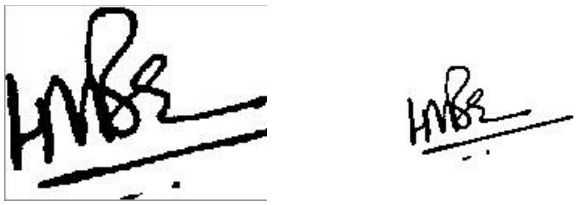


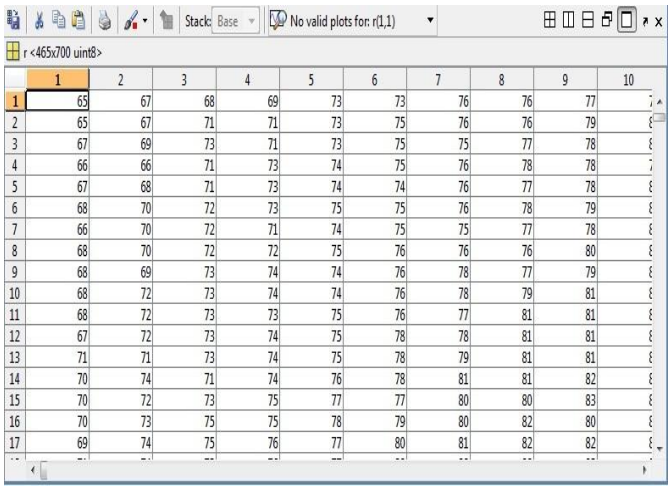
Figure 5.2: feature extraction and grayscale conversion

Step 3 : After that we have to perform the digitization on that signatures as shown in figure 5.3

5.18647933247972E-0001
1.72351384466328E-0001
1.32449449923652E-0001
2.13629276301316E-0001
1.45538602648840E-0002
8.86884674185101E-0002
5.61213979201577E-0003

Figure 5.3 : Sample digitized values of signature

Step 4 : After that the last one is the training of the signatures and store the result into the database in the form of vector as shown in figure 5.4



	1	2	3	4	5	6	7	8	9	10
1	65	67	68	69	73	73	76	76	77	
2	65	67	71	71	73	75	76	76	79	
3	67	69	73	71	73	75	75	77	78	
4	66	66	71	73	74	75	76	78	78	
5	67	68	71	73	74	74	76	77	78	
6	68	70	72	73	75	75	76	78	79	
7	66	70	72	71	74	75	75	77	78	
8	68	70	72	72	75	76	76	76	80	
9	68	69	73	74	74	76	78	77	79	
10	68	72	73	74	74	76	78	79	81	
11	68	72	73	73	75	76	77	81	81	
12	67	72	73	74	75	78	78	81	81	
13	71	71	73	74	75	78	79	81	81	
14	70	74	71	74	76	78	81	81	82	
15	70	72	73	75	77	77	80	80	83	
16	70	73	75	75	78	79	80	82	80	
17	69	74	75	76	77	80	81	82	82	

Figure 5.4: signature database

VI. RESULTS

During training, the progress is constantly updated in the training window. Of most interest are the performance, the magnitude of the gradient of performance and the number of validation checks. The magnitude of the gradient and the number of validation checks are used to terminate the training. The gradient will become very small as the training reaches a minimum of the performance. If the magnitude of the gradient is less than 1e-5, the training will stop.

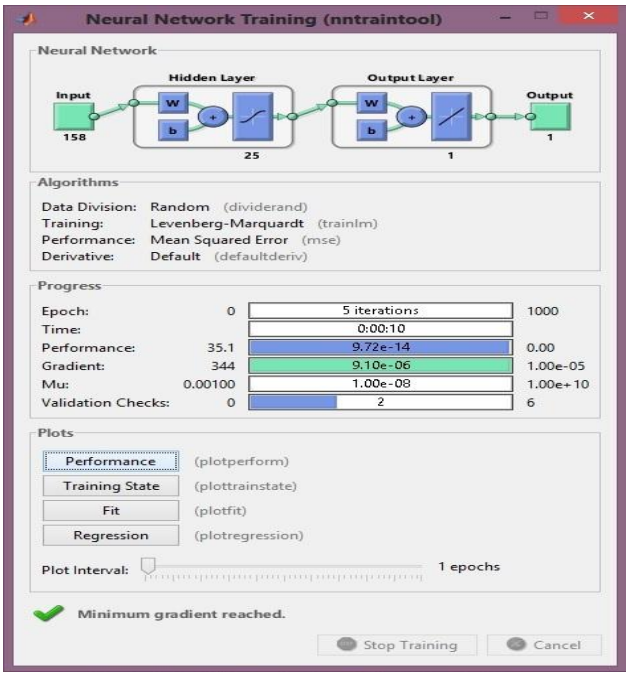


Figure 6.1: Neural Network Training

The number of validation checks represents the number of successive iterations that the validation performance fails to decrease. If this number reaches 6 (the default value), the training will stop. In this run, you can see that the training did stop because of the number of validation checks. (Note that your results may be different than those shown in the following figure, because of the random setting of the initial weights and biases.)

The following figure 6.2 shows the Neural Network Training Performance graph of signatures recognition and verification system. In which we have shown the training of signatures, Validation of signature, Testing of signatures, etc.

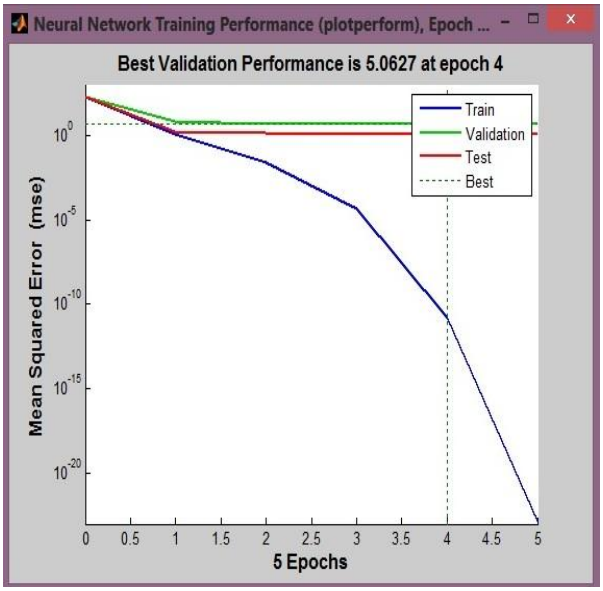


Figure 6.2: Neural Network Training Performance

In this we have shown the graph of 5 Epochs VS Mean Square Error, In which we train each signature 5 times to reduce the error rate for accuracy of result and as shown in graph the training line is decreases as is goes on training the signature up to 5 times and at the end error rate is zero. At the start point the validation, test line is variable but as the training increases the validation and the test line becomes constant.

Once you're finished training, then you run against your testing set and verify that the accuracy is sufficient.

» **Training Set**: this data set is used to adjust the weights on the neural network.

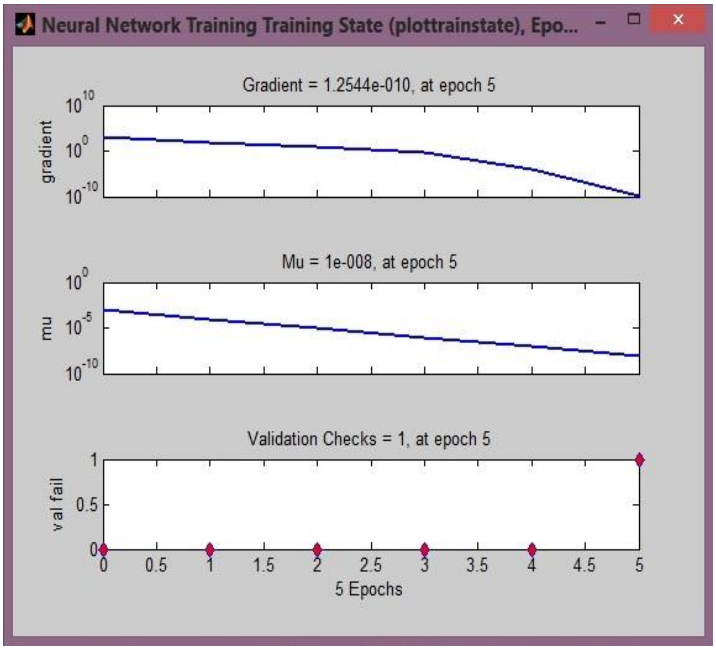


Figure 6.3: Neural Network Training State

» **Validation Set**: this data set is used to minimize over fitting. You're not adjusting the weights of the network with this data set, you're just verifying that any increase in accuracy over the training data set actually yields an increase in accuracy over a data set that has not been shown to the network before, or at least the network hasn't trained on it (i.e. validation data set). If the accuracy over the training data set increases, but the accuracy over then validation data set stays the same or decreases, then you're over fitting your neural network and you should stop training.

» **Testing Set**: this data set is used only for testing the final solution in order to confirm the actual predictive power of the network.

The following figure 6.4 shows the Neural Network Training Regression which is having different graphs for training of signature, validation of signature, attesting of signature, etc.

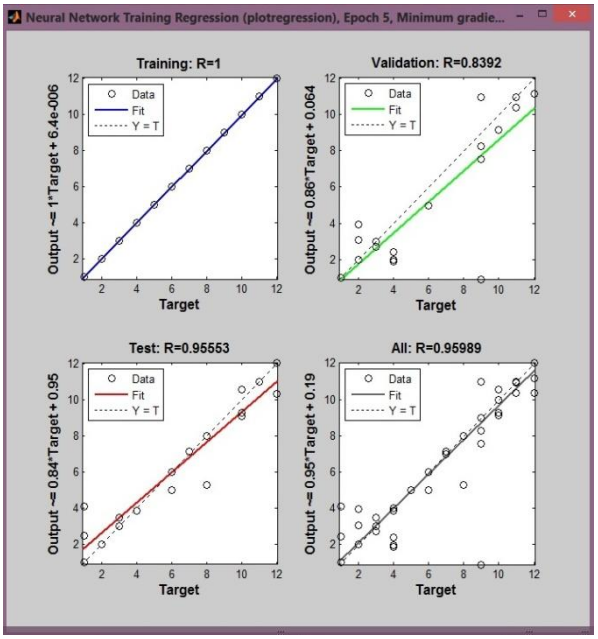


Figure 6.4: Neural Network Training Regression

The three plots represent the training, validation, and testing data. The dashed line in each plot represents the perfect result – outputs = targets. The solid line represents the best fit linear regression line between outputs and targets. The R value is an indication of the relationship between the outputs and targets. If R = 1, this indicates that there is an exact linear relationship between outputs and targets. If R is close to zero, then there is no linear relationship between outputs and targets. For this example, the training data indicates a good fit. The validation and test results also show R values that greater than 0.8.

VII. CONCLUSION

In this, we presented off-line signature recognition and verification system which is based on image processing, some global properties and ANN. Both systems used a four-step process. In the first step, the signature is separated from its image background. Second step digitization of the original signature. Some global properties which are used as input features for the NN are obtained in the third step. Two separate ANN are used; one for signature recognition and another for verification. Our recognition system exhibited a 100% success rate by identifying correctly all of the 5 signatures that it was trained for. However, it exhibited poor performance when it was presented with signatures that it was not trained for earlier. We did not consider this as a "high risk" case, because recognition step is always followed by the verification step and these kinds of false positives can be easily caught by the verification system.

Artificial Neural Network is adaptive in nature and it implements any system near to the biological neural network. It uses the training of signatures and threshold.

References

1. Ranjan Jana , Rituparna Saha , Debaleena Datta "Offline Signature Verification using Euclidian Distance"International Journal of Computer Science and Information Technologies, Vol. 5 (1) ,pages 707-710,2014.
2. Poornima G Patil, Ravindra S Hegadi "Offline Handwritten Signatures Classification Using Wavelet Packets and Level Similarity Based Scoring" International Journal of Engineering and Technology (IJET),Vol 5,pages 421-426, Feb-Mar 2013.
3. Yan Wu, Hui Geng, Xiao-yue Bian " A New Method of Signature Verification Based on Biomimetic Pattern Recognition Theory" I.J. Engineering and Manufacturing, pages7-13, 2011.
4. 2011 Explainthatstuff Neural Network (Neural Network) [online].available: http://www.explainthatstuff.com/introduction-to-neural-networks.html
5. Alan McCabe, Jarrod Trevathan and Wayne Read, school of mathematics, physics and Information Technology, James cook University, Australia."Neural Network-based handwritten signature verification," Journal of computers, vol.3, 8 August 2008.
6. A. Piyush Shanker, A.N. Rajagopalan "Off-line signature verification using DTW" Pattern Recognition Letters 28, pages 1407–1414,2007.

7. S.Armand, M. Blumenstein, and V. Muthukkumarasamy. "Off-linesignature verification using the enhanced modified direction feature and neural-based classification". International Joint Conference on Neural Networks, pages 684-691, 2006.
8. Yuan Y., T., Ernest C.M.L., "New method for feature extraction based on fractal behavior." Pergamon Pattern Recognition 35, pp.1071-1081, 2002.
9. Awcock, G.J., and Thomas, R., "Applied image processing", McGraw Hill, Inc., 1996.
10. Lim, J.S., "Two-Dimensional and Image Processing",Prentice-Hall, 1990.
11. Patrick Henry Winston, "Artificial Intelligence," 3rd edition ,USA, Addision –Wesley,1993.
12. OZ, C. Ercal, F. and Demir, Z. "Signature Recognition and Verification with ANN" .
13. Artificial Neural Networks Technology (Training in ANN) [online]. available: http://www.psych.utoronto.ca/users/reingold/courses/ai/cache/neural3.html
14. Kiyoshi Kawaguchi 2000-06-17 Multilayer Network [online].available:

    http://www.ece.utep.edu/research/webfuzzy/docs/kkthesis/kkthesishtml/node16.html.
15. Daniel Shiftman THE NATURE OF CODE (Chapter 10. Neural Networks) [online book].available: http://natureofcode.com/book/chapter-10-neural-networks/.
16. Kiyoshi Kawaguchi (2000-06-17) Biological Neural Networks[online]. available:

    http://osp.mans.edu.eg/rehan/ann/2_2%20Biological%20Neural%20Networks.htm

