

Fingerprint Identification System Using Neural Networks

Hamsa A. Abdullah

hamssa_kareem@yahoo.com

Information Engineering College/ Nahrain University / Baghdad/Iraq

Abstract

The use of fingerprint in biometric identification has been the most widely used authentication system. The uniqueness of the fingerprint for every human provides us with all we need for faultless identification. However, during the fingerprint scanning process, the image generated by the scanner may be slightly different during each scan. This paper puts the implementation of Artificial Neural Networks to provide an efficient matching algorithm for fingerprint authentication. Using the Back-Propagation technique, the algorithm works to match twelve fingerprint parameters and relate them to a unique number provided for each authorized user. Upon matching, the algorithm returns the best match for the given fingerprint parameters.

Key Words— Back propagation, Bio-metric, Neural network, Authentication

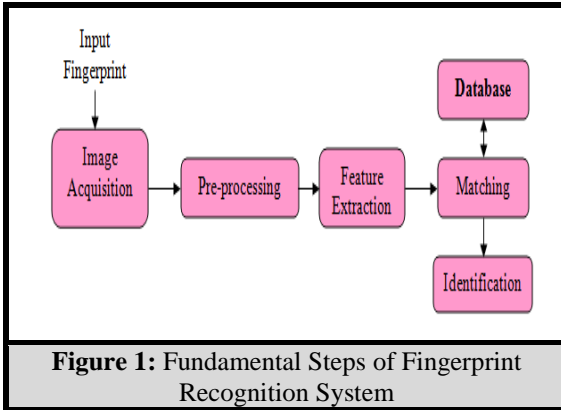
1. Introduction

This paper deals with the implementation of a pattern matching algorithm to match and authenticate fingerprints. The algorithm uses Neural Network concepts to perform effective pattern matching to identify fingerprints. The concept relies on image processing techniques to locate 12 fingerprint features using fingerprint analysis by moment. Fingerprint processing must return the match of each of these features. Though any pair of corresponding feature values may be same for two individuals, the combination of all 12 values may never be the same. Hence: an effective and unique identification parameter is found for each individual.

A fingerprint is a graphical pattern of ridges and valleys on the surface of a human finger. Due to the uniqueness and permanence of fingerprints, they are among the most reliable human characteristics that can be used for personal identification [1]. The performance of an automatic fingerprint identification system relies heavily on the fingerprint image quality which can be affected by several factors such as the presence of scars,

variations of the pressure between the finger and acquisition sensor, worn artifacts, and the environmental conditions during the acquisition process. For constructing an effective fingerprint identification system, an effective enhancement algorithm is necessary.

Directional behavior is an obvious characteristic in a fingerprint image caused by the continuous flow of fingerprint ridges whose orientations are slowly changing in the fingerprint pattern. Ridges and valleys in a local neighborhood form a sinusoidal-shaped wave, which has a well-defined frequency and orientation. The ridge orientations as well as the ridge period are maintained almost constant inside a small local area. The basic idea for enhancing a fingerprint is to first detect the orientations of the ridges, and then a smoothing operation is performed along the ridge directions to remove the noise. Ridge orientations can be estimated directly from a gray-scale image using gradient methods [2, 3] and others require a binary fingerprint image [4]. With the orientation information available, the smoothing operation is usually done by adaptive filters [5] attuned to the corresponding orientation and ridge period in local regions. Hong et al. [6] use Gabor filters as Sobel filters to remove the noise and retain true ridge and valley structures, and highpass filter that is used to enhance the image. The classical stages of enhancement and ridge detection are binarization, enhancement, feature extraction, training and matching. The basic fundamental steps of these systems (see Fig. (1)) are image acquisition, pre-processing segmentation, enhancement and feature extraction, matching along with classification through databases. Authentication or verification systems authenticate the person's identity by comparing the own biometric template(s) stored in database (One-to-One comparison). An identification system is recognized by an individual by searching the entire templates in database for match (One-to-Many Comparison) [7].



2. Fingerprint Image

A fingerprint is the feature pattern of one finger. Each person has his own fingerprints with the permanent uniqueness [8]. Due to the uniqueness and permanence of fingerprints, they are among the most reliable human characteristics that can be used for personal identification.

2.1 Image Acquisition

A number of methods are used to acquire fingerprints. Among them, the inked impression method remains the most popular one. Inkless fingerprint scanners are also present eliminating the intermediate digitization process. Fingerprint quality is very important since it affects directly the minutiae extraction algorithm. The size of the scanned fingerprints that are used in this research is 188x240 pixels. The images are taken in this size in order to ease the computational burden.

2.2 Fingerprint Image Processing

Processing of fingerprint image is necessary to: (i) improve the clarity of ridge structures of fingerprint images (ii) maintain their integrity, (iii) avoid introduction of spurious structures or artifacts, and (iv) retain the connectivity of the ridges while maintaining separation between ridges. Fingerprint image processing operations are image enhancement, image normalization and image binarization [9].

3. Fingerprint Image Enhancement

The local ridge orientation is an intrinsic property of the fingerprint images. By viewing a fingerprint image as an oriented texture, a number of methods have been proposed to estimate the orientation field of fingerprint images. The previous enhancement algorithm is either local orientation field filter-based or Gabor filter-based. The orientation field

filtering techniques usually assume the local ridge orientation could be reliably estimated and be taken advantage to enhance the fingerprint image. The ridge structures in poor-quality fingerprint images are not always well-defined and, hence, the orientation information could not be correctly detected, which greatly restricts the applicability of these techniques. The Gabor filter based technique could obtain a reliable orientation estimate even for corrupted images. It is unsuitable for an on-line fingerprint recognition system such as AFIS because the algorithm is computationally expensive [10].

3.1 Image edge Detection

Edges are one of the most important features in an image as they are strong indicators of object boundaries. Sharp edges help us observe features and objects. Edge shape and position can be seen as an effective representation of an object. One could image extracting a polygonal representation of the boundary of both objects. In practice this requires a very controlled environment like the one shown with few objects and good contrast between the objects and the background. There are three basic types of detection: points, lines and edges. For point detection using high pass filter with center 8 as following [11, 12]:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

For line detection the mask bellow can be used:

-1	-1	-1	-1	-1	2	-1	2	-1	2	-1	-1
2	2	2	-1	2	-1	-1	2	-1	-1	2	-1
-1	-1	-1	2	-1	-1	-1	2	-1	-1	-1	2
Horizontal			+45			Vertical			-45		

For edge detection the techniques that are used Gradient operator, Sobel operators which are shown in the masks below:

$$h_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad h_y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Based on convolution operations – compute weighted averages over a 3x3 neighborhood with the equation bellow:

$g_x(x, y) = h_x \times f(x, y)$	1
$g_y(x, y) = h_y \times f(x, y)$	2

The two gradients g_x and g_y computed at each pixel are regarded as the x and y components of a gradient vector, which has gradient magnitude and direction given by:

$$g = \sqrt{g_x^2 + g_y^2} \quad \phi = \tan^{-1} \left[\frac{g_y}{g_x} \right] \quad 3$$

Where the orientation θ is measured relative to x axis. Gradient magnitude is sometimes approximated by:

$$g = |g_x| + |g_y| \quad 4$$

4. Fingerprint Profile Normalization Algorithm

An ideally sensed or scanned fingerprint image has clear and distinct ridges and valleys. For example, an ideal fingerprint image could be a rolled ink impression on a fingerprint card. The finger skin profile made of the ridges is evenly pressed on the flat paper card, which leaves the ink impression of ridges as continuous flow of foreground passes, and in between the ridges are the valleys as background with white color. But in reality the fingerprint scanning devices are far from this ideal set up. Even the National Institute of Standards and Technology (NIST) fingerprint images scanned from the inked fingerprint cards are not perfect. Although the noise introduced in the fingerprint images during the scanning is a problem in fingerprint minutiae extraction, the uneven distribution of the pixel levels in the fingerprint images is another common problem for detection of ridge orientation. Figure 2 shows an example image of uneven ridge profile.

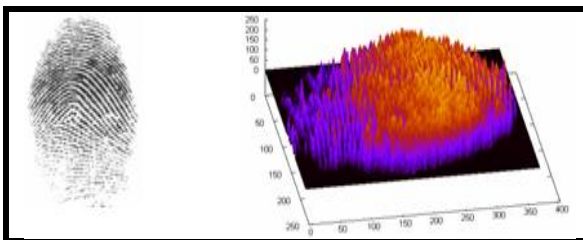


Figure 2: Example fingerprint image of uneven ridge profile.

There are algorithms for image normalization. To decrease the dynamic range of the gray scale between ridges and valleys of a fingerprint image, Hong et al [13]. Proposed the following algorithm. Let $I(i, j)$ denote the gray-level value at pixel (i, j) , M and VAR denote the estimated mean and

variance of I , respectively, and $G(i, j)$ denote the normalized gray-level value at pixel (i, j) . The normalized image is defined as follows:

$$G(i, j) = \begin{cases} M_0 + \sqrt{\frac{VAR_0 (I(i, j) - M)^2}{VAR}} & \text{if } I(i, j) > M \\ M_0 - \sqrt{\frac{VAR_0 (I(i, j) - M)^2}{VAR}} & \text{otherwise} \end{cases} \quad 5$$

Where M_0 and VAR_0 are the desired mean and variance values of fingerprint image, respectively. This algorithm is a pixel-wise operation, and it does not change the clarity of the ridge and valley structures. Although the main purpose of the normalization is to reduce the variations in gray-level values along ridges and valleys, the algorithm adjusts the pixel levels using only the statistical parameters M and VAR .

Since the fingerprint identification systems rely on information extracted from ridges, in this research the normalization algorithm based on the approximation of the finger skin profile to highlight the ridges used.

Let us treat a fingerprint image as a three-dimensional object whose positional coordinates are in the x - y plane and the pixel gray values are in the z plane. Ideally, with even pressure on a flat surface such as a fingerprint card or a flat top fingerprint scanner, the finger skin profile is represented by ridges pixels with dark colors, and the ridge pixel colors are within a narrow range. The finger skin profile is a curved surface going through the tips of the ridges. In the case of real fingerprint images with uneven finger skin (ridge) profile, by assuming the skin profile is a continuous surface due to the elasticity of the human skin. To normalize a fingerprint image according to the skin profile, we first approximate the profile surface using selected ridge pixels. We accomplish this using a locally adaptive binarization algorithm. The objective is to retain most of the foreground ridge pixels. Direct application of a binarization algorithm on a fingerprint image generally may not extract the ridges well, and can often result in broken ridge lines. But the binary image generated by the locally adaptive algorithm gives locations of ridge pixels that are enough to represent the finger skin profile, and the skin profile is approximated based on these pixels.

4.1 Fingerprint Binarazation

Binarazation is the process of converting the gray-scale image into a binary image is called binarazation. Zeros and ones forms binary image. Global threshold algorithm is used for performing binarazation process. Looking at each pixel on the fingerprint image and deciding whether it should be converted to black (0) or white (255) i.e., to 0 or 1 from gray level to black and white image each pixel value is compared with threshold level then pixel value is set to Zero; otherwise it is set to 255 [14]. Figure 3 shows a binary image from the binarization algorithm and the selected ridge pixels, and the rest of the pixels are filled in white.

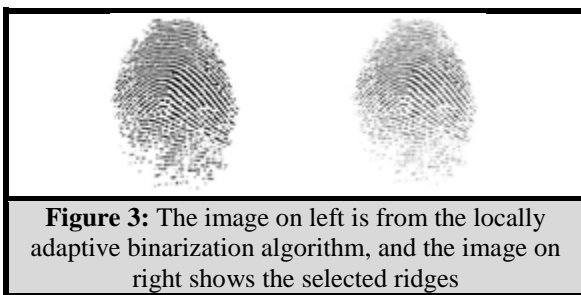


Figure 3: The image on left is from the locally adaptive binarization algorithm, and the image on right shows the selected ridges

5. Fingerprint Feature Extraction With Moments

Moment invariants are important shape descriptors in computer vision. There are two types of shape descriptors: contour-based shape descriptors and region-based shape descriptors. Regular moment invariants are one of the most popular and widely used contour-based shape descriptors is a set of derived by Hu [15]. A computer vision system recognizing objects in captured images is established using Geometric Moment (GM). GM was derived from the theory of algebraic invariant. GM technique is chosen to extract image features since the features generated are Rotation Scale Translation (RST)-invariant. GM was successfully applied in aircraft identification, texture classification and radar images to optical images matching. Two-dimensional moments of a digitally sampled $M \times N$ image that has gray function $f(x, y)$, $(x = 0, \dots, M - 1)$ and $(y = 0, \dots, N - 1)$ is given as:

$$m_{pq} = \sum_{x=0}^M \sum_{y=0}^N x^p y^q f(x, y) \quad 6$$

It should be noted that eq. (6) can assume very large values, especially for high order moments (large p, q).

6. Overview Of Neural Network

A *neural network* is a computational structure inspired by the study of biological neural processing. There are many different types of neural networks, from relatively simple to very complex, just as there are many theories on how biological neural processing works. A layered feed-forward neural network has *layers*, or subgroups of processing elements. A layer of processing elements makes independent computations on data that it receives and passes the results to another layer. The next layer may in turn make its independent computations and pass on the results to yet another layer. Finally, a subgroup of one or more processing elements determines the output from the network. Each processing element makes its computation based upon a weighted sum of its inputs. The first layer is the *input layer* and the last the *output layer*. The layers that are placed between the first and the last layers are the *hidden layers*. The processing elements are seen as units that are similar to the neurons in a human brain, and hence, they are referred to as *cells*, *neuromimes*, or *artificial neurons*. A *threshold function* is sometimes used to qualify the output of a neuron in the output layer. Synapses between neurons are referred to as *connections*, which are represented by edges of a directed graph in which the nodes are the artificial neurons. Nets consist of small units called *cells*, and these are connected to each other in such a way that they can pass signals to each other [16].

The weights used on the connections between different layers have much significance in the working of the neural network and the characterization of a network. The following actions are possible in a neural network:

1. Start with one set of weights and run the network. (No Training)
2. Start with one set of weights, run the network, and modify some or all the weights, and run the network again with the new set of weights. Repeat this process until some predetermined goal is met. (Training)

The connections have certain *strengths* or *weights*. The net starts off with these connection strengths set randomly. The network is exposed to various inputs and the strengths adjust them according to some mathematical plan. This is what we call *training* and after it, the network can recognize input patterns or, at least, do something sensible - whatever it has been trained to do. The information is therefore *stored in the strengths of the connections*, just as it is in the human brain.

6.1 Neural Network Architecture

Neural Network Architecture defines its structure including number of hidden layers, number of hidden nodes, number of output nodes and activation function [17].

- i. Number of hidden layers: The hidden layer(s) provide the network with its ability to generalize. In theory, a neural network with one hidden layer with sufficient number of hidden neurons is capable of approximating any continuous function. In practice, neural network with one and occasionally two hidden layers are widely used and have to perform very well.
- ii. Number of hidden nodes: There is no magic formula for selecting the optimum number of hidden neurons. However, some thumb rules are available for calculating number of hidden neurons. A rough approximation can be obtained by the Kolmogorov theorem. For a three layer network with n input, the hidden layer would have $2n+1$ neurons [18]. In this paper the numbers of input layer that used are 12 so the numbers of hidden neurons are 25.
- iii. Number of output nodes: Neural network with multiple outputs, especially if these outputs are widely spaced, will produce inferior results as compared to network with a single output.
- iv. Activation function: Activation functions are mathematical formulae that determine the output of a processing node. Each unit takes its net input and applies activation to it. Non liner functions have been used as activation functions such as logistic, tanh. The purpose of the transfer function is to prevent output from reaching very large value which can 'paralyze' neural networks and thereby inhibit training. Transfer functions such as sigmoid are commonly used because they are nonlinear and continuously differentiable which are desirable for network learning [17].

7. Fingerprint Classifier

Classification is the final stage of any image-processing system where each unknown pattern is assigned to a category. The degree of difficulty of the classification problem depends on the variability in feature values for objects in the same category, relative to the difference between feature values for objects in different categories. In this study MLP (Multilayer Perceptron) classifiers used as pattern classifiers.

MLP has been termed a universal approximate, and can provide an optimal solution to an arbitrary

classification problem. It implements linear discriminates, but in a space where the inputs have been mapped nonlinearly. The key power provided by such networks is that they admit fairly simple algorithms where the form of the nonlinearity can be learned from the training data. The models are thus extremely powerful, have nice theoretical properties, and apply well to a vast array of real-world applications. Figure 4 shows a simple three-layer MLP neural network, consisting of an input layer, hidden layer and an output layer, interconnected by modifiable weights represented by links between layers. One of the most popular methods for training such multilayer networks is based on gradient descent in error-the backpropagation algorithm a natural extension of the LMS (Least Mean Square) algorithm. Guided by an analysis of networks and their function we can make informed choices of the scaling of input values and initial weights, desired output values, and more. Network architecture or topology plays an important role for neural net classification, and the optimal topology will depend on the problem at hand [19].

Extraction of appropriate features is one of the most important tasks for a recognition system. A MLP of three layers is trained to detect the features in the fingerprint image of size 188x240. The first layer of the network has 12 neurons associated with the components of the input vector. The hidden layer has 25 neurons and the output layer has 6 neurons. Figure 4 shows Three-layer MLP structure. The networking will be trained using the backpropagation algorithm. State the number of epochs needed for convergence as well as the training time for this method. Once the network is trained, the next step is to input the prototype fingerprint images to extract the feature. An essential issue in the field of pattern analysis is the recognition of objects. The idea of using moments in shape recognition gained prominence derived a set of invariants using algebraic invariants.

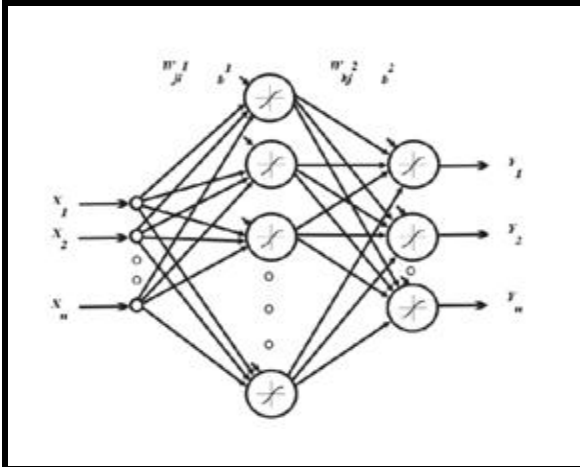


Figure 4: Three-layer MLP structure.

8. Training

Since the output(s) may not be what is expected, the weights may need to be altered. Some rule then needs to be used to determine how to alter the weights. There should also be a criterion to specify when the process of successive modification of weights ceases. This process of changing the weights, or rather, updating the weights, is called *training*. A network in which learning is employed is said to be subjected to *training*. Training is an external process or regimen. Learning is the desired process that takes place internal to the network. The training steps are as follows:

1. The first input (training) pattern is presented to the network.
2. The connections are adjusted a tiny amount to improve the network's chances of recognizing that pattern if it sees it again.
3. The second pattern is presented, and step 2 repeated.
4. The same thing happens for all the training patterns.
5. The whole process is rerun with all the training patterns for hundreds (thousands) of times.

9. Back-Propagation

Back Propagation is the standard way of training neural networks. It works basically like this:

The input pattern on which the network is to be trained is presented at the input layer of the net and the net is run normally to see what output it actually does produce. The actual output is compared to the desired output for that input pattern. The differences between actual and desired form an *error pattern* [16].

Now the errors have to be calculated for the output layer. We use the following formula and apply it

for every node of *i* in the output layer (*i.e.* go through all the nodes in the output layer):

$$E_i = \frac{1}{2} \sum_i (y_i - d_i)^2 \quad 7$$

In this case E_i is the error for node *i* in the output layer, y_i is the activation for node *i* in the output layer and d_i is the desired output for the node [17]. The error pattern is used to adjust the weights on the output layer so that the error pattern would be reduced the next time if the same pattern were presented at the inputs. Then the weights of the hidden layer are adjusted similarly, this time comparing what they actually produce with the neurons in the output layer to form an error pattern for the hidden layer. Finally, if there is an input layer (as opposed to simply a set of points where the input is passed into the network), then the weights for that layer are trained similarly. Indeed, the error can be propagated as far back as you like, over as many layers as you like [16].

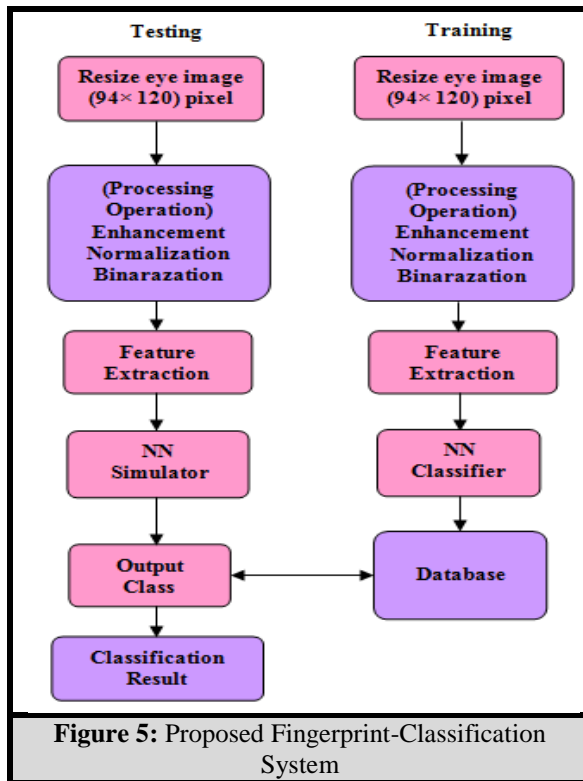
In Back propagation, the input signal moves forward through the network and the errors trickle backwards (they "propagate backwards", hence the name), followed by the next signal. The network is trained by presenting each input pattern in turn at the inputs and propagating forwards and backward, followed by the next input pattern. Then this cycle is repeated several times.

The reason for not putting input pattern 0 at the inputs and propagating many times over before moving to pattern 1 and doing that many times, is that the network would develop weights that would only reflect the pattern on which it was being trained at the time. By the time it had undergone 1000 cycles of training on pattern 1. The number of cycle's 1000 chosen in order to reach to suitable error.

10. Implementation

The entire source code has been written in Matlab. This work deals with recognition fingerprint using low resolution images through Neural Networks. In the proposed technique (see Fig. 5), image resizing is applied to a unified image size for all fingerprint samples. In order to accomplish fingerprint recognition task, a single neural network is incorporated. Implementation is divided into two phases. The pre-processing phase and the neural processing phase. In Pre-processing phase time effective preprocessing is performed in order to make image data best fit for neural network input. The features extracted by implement the method of moments feature extraction with matlab by computes, the results of this part are extract 12

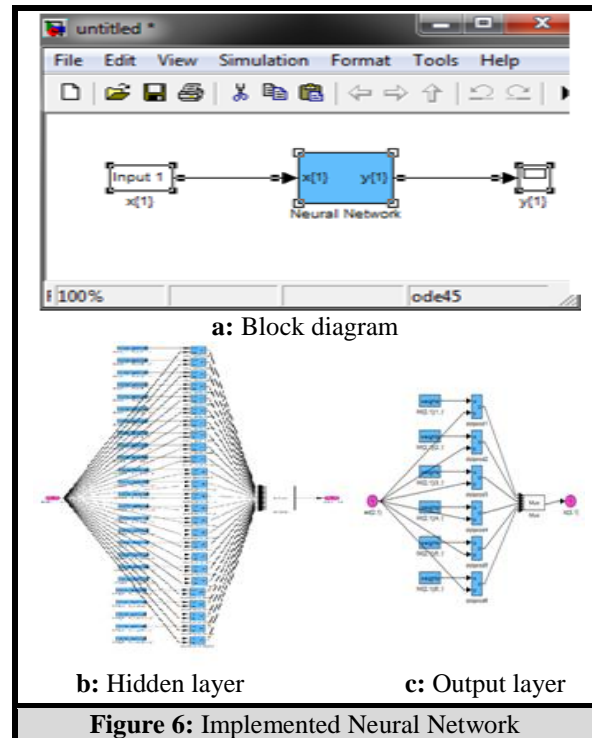
features from each fingerprint. The system trained 50×12 symbol as input for database using the proposed technique and compared their performance.



Appendix C illustrates the implementation program of first phase. Neural network further got two steps training and testing, in training neural network learns through examples, training set of images is presented as input, each layer output is calculated through transfer functions acting as summation junction, and it uses back propagation algorithm to accomplish recognition training part to achieve desired results. The diagram below shows an illustration of the network that has been implemented.

Initially the program needs to train the net. This is done using the back-propagation algorithm previously explained. The implementation of the algorithm is as follows. There are **12 input parameters** and **6 output parameters** as shown in Fig 6. The number of output parameters 6 taken according to number of training patterns $50 \approx (2^6)$, for each pattern there is sequence of six bit to represent it. There is also only one hidden layer. This layer has **25 neurons**. During training, the input pattern and the corresponding output target pattern are required by the net. This is fed into the program by using matrices.

The input patterns are stored in a 2D matrix of order 50×12 where **50** stands for the number of training patterns. And **12** stands for the number of features that extracted from each training pattern. The output patterns are stored similarly, in a 50×6 matrix, **50** again being the number of training pattern.



The input training pattern matrix and the corresponding output training pattern matrix have a one-to-one correspondence, where the 1st output pattern is the target pattern for the corresponding 1st input pattern in the input matrix. The input and the target output patterns are fed to the program and the net through a training file. The first row represents the first training set. The 1st 12 values represent the 12 input parameters (each parameter is separated from the other by a single 'space'). The output pattern follows its corresponding input, separated from the input by a single 'tab'.

11. Experimental Result

The algorithm is implemented using 2.1 GHz Pentium 4 machine with Windows 7 and MATLAB 8.0 as the development tool. Two set of images are required, one for the training of the neural network and another set of images upon which testing is done. Each image is of size 188×240 pixels. In this research, the testing subjects reach to 100 images taken from 100

different peoples. These 100 images are divided into 50 known images (previously trained one) and 50 images (newly untrained one). Table (1) shows the recognition phase of the 100 testing fingerprint samples. Neural Network is trained upon some set of images, and tested upon unseen images. The results are analyzed to calculate the recognition rate of the system as shown in table (2). A recognition rate of 100% is obtained for this system. This recognition rate value is perfectly suitable for fingerprint recognition systems. In proposed technique neural network uses back propagation algorithm for error computation and new weights calculation for each neuron link. The network undergoes process of training, continuously in an iterative manner it calculates output from each layer, extracting the mean square error and propagating it backwards if it is not approaching targets. Due to this backward error propagation, error-signal for each neuron is calculated. Which in fact is used for neuron weight updating. If its approaching targets then training is considered done. The process of training these images is shown in Fig. 7, in which training curve is approaching its goal through readjustment of weights and biases.

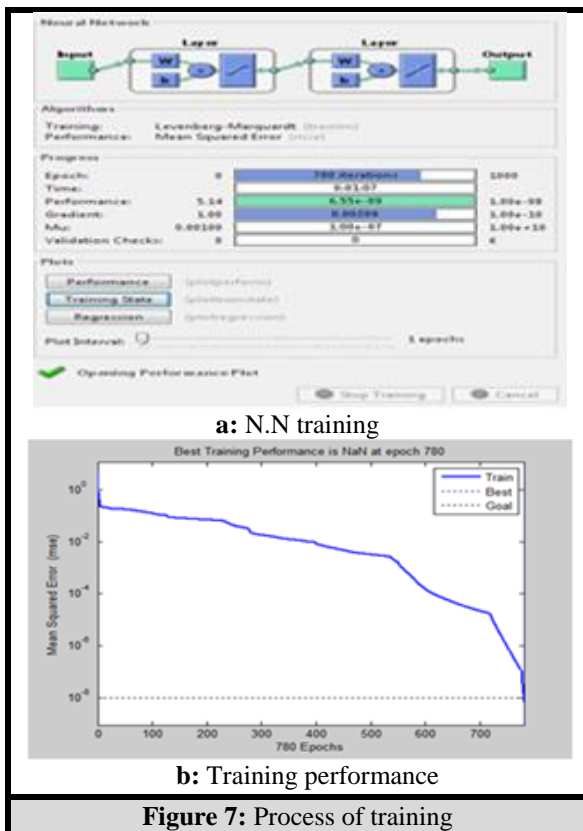
The response of the Neural Network is dependent upon weights, biases and activation functions. The activation functions used in the feed-forward back propagation neural network are Tangent sigmoid (tansig) used in hidden layer and purelin used in output layer. These functions acts as summation junction and calculates the output from the inputs presented. After the training phase is completed, the identification process must be implemented in order to evaluate the proposed system. The evaluation process is accomplished by testing the system with known and newly fingerprints images. New images for testing are applied to the trained neural network along with already trained images for calculating the percentage of accuracy and error. A set of (100) fingerprint samples are used to test the proposed system. Appendix A & B show fingerprint samples of training and testing respectively.

Finger-print Samples	Actual Class	Calculated Class	Finger-print Samples	Actual Class	Calculated Class
1	000001	000001	51	New	None
2	000010	000010	52	New	None
3	000011	000011	53	New	None
4	000100	000100	54	New	None
5	000101	000101	55	New	None
6	000110	000110	56	New	None
7	000111	000111	57	New	None
8	001000	001000	58	New	None
9	001001	001001	59	New	None
10	001010	001010	60	New	None
11	001011	001011	61	New	None
12	001100	001100	62	New	None
13	001101	001101	63	New	None
14	001110	001110	64	New	None
15	001111	001111	65	New	None
16	010000	010000	66	New	None
17	010001	010001	67	New	None
18	010010	010010	68	New	None
19	010011	010011	69	New	None
20	010100	010100	70	New	None
21	010101	010101	71	New	None
22	010110	010110	72	New	None
23	010111	010111	73	New	None
24	011000	011000	74	New	None
25	011001	011001	75	New	None
26	011010	011010	76	New	None
27	011011	011011	77	New	None
28	011100	011100	78	New	None
29	011101	011101	79	New	None
30	011110	011110	80	New	None
31	011111	011111	81	New	None
32	100000	100000	82	New	None
33	100001	100001	83	New	None
34	100010	100010	84	New	None
35	100011	100011	85	New	None
36	100100	100100	86	New	None
37	100101	100101	87	New	None
38	100110	100110	88	New	None
39	100111	100111	89	New	None
40	101000	101000	90	New	None
41	101001	101001	91	New	None
42	101010	101010	92	New	None
43	101011	101011	93	New	None
44	101100	101100	94	New	None
45	101101	101101	95	New	None
46	101110	101110	96	New	None
47	101111	101111	97	New	None
48	110000	110000	98	New	None
49	110001	110001	99	New	None
50	110010	110010	100	New	None

Type	No. of Samples	Recognition Rate %
Previously Trained Samples	50	100
New Samples	50	100
Overall Recognition Rate %		100

12. Literature Survey

Many researchers studied the field of fingerprint recognition, few of them focused on moment feature extraction. In 2008 B. Gour, T. Bandopadhyaya, and S. Sharma presented



Fingerprint Feature Extraction Using Midpoint ridge Contour method and Neural Network. This paper examines the minutia extraction approach by using the midpoint ridge contours. The result of this method takes less time and do not detect any false minutiae [9]. In 2009 A. Askarunisa, Sankaranarayanan. K, Sundaram. R and Sathick .M. Batcha presented Finger Print Authentication Using Neural Networks. This paper puts forth the implementation of Artificial Neural Networks to provide an efficient matching algorithm for fingerprint authentication. Using the Back-Propagation technique, the algorithm works to

match six fingerprint parameters and relate them to a unique number provided for each authorized user. Upon matching, the algorithm returns the best match for the given finger print parameters [16]. In 2010 A. Chatterjee, S. Mandal, G. M. Atiqur Rahaman, and A. M. Arif presented Fingerprint Identification and Verification System by Minutiae Extraction Using Artificial Neural Network. This paper introduced a new method for fingerprint identification technology by minutiae feature extraction using back-propagation algorithm. For an input image, the local ridge orientation is estimated and the region of interest is located. For fingerprint recognition, the verification part of the system identifies the fingerprint based training performance of the network [8].

This paper puts the implementation of Artificial Neural Networks to provide an efficient matching algorithm for fingerprint authentication. Using the Back-Propagation technique, the algorithm works to match twelve fingerprint parameters and relate them to a unique number provided for each authorized user.

13. Conclusion










Biometrics-based methods for personal authentication assume that the biometric characteristics used for the verification of an individual's identity are unique from person to person. However the uniqueness of fingerprints from one individual to another has been thoroughly verified. This algorithm aims to capitalize on this uniqueness and improve the efficiency, in terms of matching accuracy, of fingerprint identification and authentication. In this paper back-propagation N.N has been trained as a fingerprints classifier to identify fingerprints with time effective preprocessing, which greatly increases the performance of the network. The recognition rate of fingerprints depends on the quality of fingerprints and effectiveness of preprocessing system.

References

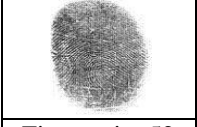








- [1] Jain and E. A., "An Introduction to Biometric Recognition", IEEE Tran. On Circuits and Systems for Video Technology, Vol.14, No.1, pp: 4-20, 2004.
- [2] P. Baldi and Y. Chauvin, "Neural Networks for Fingerprint Recognition", neural computation, 5, pp.485-501, 1993.
- [3] V. Srinivasan , and N. Murthy, "Detection of Singularity Points in Fingerprint Images", Pattern recognition, 20, pp.513-523., 1992.

- [4] D. Maio and D. Maltoni, "Direct Gray Scale Minutiae Detection in Fingerprints", IEEE Tran on Pattern Analysis and Machine Intelligence, Vol.19, No. 1, 1997.
- [5] S. Al, "Fuzzy Control for Fingerprint Feature Selection", Proc. ACCV Vol. 3, pp: 767-771, 1995.
- [6] X. Xia, and L. O’Gorman, "Innovations in Fingerprint Capture Devices", Journal of Pattern Recognition, Pergamon Press, Vol. 36, No. 2, pp. 361-370, 2002.
- [7] D. Maltoni, Dmaio, A.K. Jain, S. prabhakar, "Hand book of Fingerprint Recognition", 2003.
- [8] A. Chatterjee, S. Mandal, G. M. Atiqur Rahaman, and A. M. Arif, "Fingerprint Identification and Verification System by Minutiae Extraction Using Artificial Neural Network", ISSN 2078-5828 (PRINT), ISSN 2218-5224 (ONLINE), VOLUME 01, ISSUE 01, MANUSCRIPT CODE: 100703, 2010.
- [9] B. Gour, T. Bandopadhyaya, and S. Sharma, "Fingerprint Feature Extraction Using Midpoint ridge Contour method and Neural Network", IJCSNS International Journal of Computer Science and Network Security, VOL.8, No.7, 2008.
- [10] Y. Zhang, Y. Jiao, J. Li, and X. Niu, "A Fingerprint Enhancement Algorithm using a Federated Filter", Information Countermeasure Technique Institute, Harbin Institute of Technology, China, 2005.
- [11] M. Thurley, and T. Andersson, "Edges & Edge Detection", Industrial Image Processing, 2009.
- [12] H. Rhody, "Image Enhancement and Spatial Filtering II", Chester F. Carlson Center for Imaging Science Rochester Institute of Technology, 2005.
- [13] Z. Shi ,and V. Govindaraju, "Fingerprint Image Enhancement Based on Skin Profile Approximation", Center of Excellence for Document Analysis and Recognition (CEDAR) State University of New York at Buffalo, Amherst, USA zshi@cedar.buffalo.edu, 2006.
- [14] A.Ida, V. Kavitha, and K. Easwarakumar, "Data Integrity in Fingerprint Images", International Journal of Soft Computing, Vol.2, No. 2, pp:331-334, 2007.
- [15] M. Rizon, H. Yazid, P. Saad, A. Shakaff, A. Saad,M. Mamat, S. Yaacob, H. Desa and M. Karthigayan, "Object Detection using Geometric Invariant Moment", American Journal of Applied Sciences, Vol. 2, No. 6, pp: 1876-1878, 2006.
- [16] A. Askarunisa, Sankaranarayanan. K, Sundaram. R and Sathick .M. Batcha, "Finger Print Authentication Using Neural Networks", MASAUM Journal of Computing, Vo. 1,No. 2, 2009.
- [17] G. Jha, "Artificial Neural Networks", Indian Agricultural Research Institute, PUSA, New Delhi-110 012, 2005.
- [18] J.P. Hayes, "Computer Architecture & Organization", Mc,Grew-Hill compon, 1998.
- [19] M. Mercimek, K. Gulez, and T. Mumcu, "Real object recognition using moment invariants", Sadhana Vol. 30, pp: 765-775. © Printed in India, 2005.
- [20]

Appendix A: Fingerprint training samples

		
Fingerprint 1	Fingerprint 5	Fingerprint 11
		
Fingerprint 18	Fingerprint 22	Fingerprint 29
		
Fingerprint 35	Fingerprint 38	Fingerprint 46

Appendix B: Fingerprint testing samples

		
Fingerprint 52	Fingerprint 58	Fingerprint 65
		
Fingerprint 68	Fingerprint 71	Fingerprint 76
		
Fingerprint 82	Fingerprint 87	Fingerprint 94

Appendix C: Pre-processing & Feature Extraction Program

```
function input2=Insertimage(a)
M=mean(mean(a)); V=var(var(a));
for i=2:x-1
    for j=2:y-1; p=1;
        for m=i-1:i+1; q=1;
            for n=j-1:j+1
d(p,q)=-1*a(m,n); q=q+1;
end p=p+1; end
d(3,3)=-8*d(3,3);
m(r)=d(3,3); r=r+1;
c(i,j)=255-sum(sum(d));
if c(i,j)>M
g(i,j)=M0+(((V0*(c(i,j)-
M)^2))/V)^0.5);
else g(i,j)=M0-
(((V0*(c(i,j)-M)^2))/V)^0.5);
end end end
for i=1:x
    for j=1:y
```

```
if g(i,j)<t
N(i,j)=0;
else N(i,j)=255;
end end end
[M1,M2,M3,M4,M5,M6,M7,M8,M9,M10,M11,M12]=extmom2(N);
input1 =
[M1,M2,M3,M4,M5,M6,M7,M8,M9,M10,M11,M12];
function
[M1,M2,M3,M4,M5,M6,M7,M8,M9,M10,M11,M12]=extmom2(im)
for g=1:p for h=1:q
for i=1:x for j=1:y
mom(i,j)=(i^(g-1))*(j^(h-1))*G(i,j); end end
MomentGray(g,h)=sum(sum(mom)); end
End
MomentGray=reshape(MomentGray,1,p*q);
[M1,M2,M3,M4,M5,M6,M7,M8,M9,M10,M11,M12]= MomentGray;
```

نظام التعرف على بصمة الاصبع باستخدام الشبكات العصبية

همسة عبدالكريم عبدالله

كلية هندسة المعلومات/ جامعة النهدين/ بغداد/ العراق

الخلاصة

إن استخدام بصمة الاصبع للتعرف على الشخص أصبحت تدخل في جميع الميادين التي تستخدم أنظمة الحماية. الانفراد ببصمة الاصبع لكل شخص يعطينا ما نحتاجه للتعرف على الأشخاص. خلال إدخال بصمة الاصبع عبر جهاز المسح قد تختلف البصمة اثناء كل مسحة. إن هذا البحث يضع تمثيل الشبكة العصبية الصناعية لتجهيز طريقة مثلى لتطابق بصمة الاصبع. تم استخدام طريقة الانتشار الخلفي (Back-Propagation) كطريقة أساسية للعمل على إيجاد التطابق لكل بصمة تدخل للنظام, وذلك عن طريق استخراج اثني عشر عنصر لكل بصمة واعتبارها مدخلات للنظام للتعرف على بصمة الإصبع. ونتيجة للتطابق, يرجع النظام البصمة التي تم التطابق معها من خلال تطابق العناصر الخاصة بكل بصمة