5.1 Introduction to neural network

The term neural network is related to biology. But similar concept can be implemented with computer science also. With this chapter, initially it is required to have introduction to neural network concept. The definition of neural network can be seen as:

“A neural network is an interconnected assembly of simple processing elements, units or nodes, whose functionality is loosely based on the animal neuron. The processing ability of the network is stored in the interunit connection strengths, or weights, obtained by a process of adaptation to, or learning from, a set of training patterns [1].”

Human brain consist about 100 billion neurons. These neurons communicate with electrical signals, which are impulses or spikes in the voltage of the cell wall. The interneuron connections are mediated by electrochemical junctions called synapses, which are located on branches of the cell referred to as dendrites. Each neuron typically receives many thousands of connections from other neurons and is therefore constantly receiving a multitude of incoming signals, which eventually reach the cell body. They are integrated with some mechanism and if resulting signals exceeds some threshold values, neuron will fire or will generate voltage impulse as response. This can be transmitted to other neurons via branching fiber called axon.

The artificial equivalents of biological neurons are the nodes or units. Synapses are modeled with a single number or weight, so that each input can be multiplied by weight before sending to equivalent cell body. Weighted signals are summed together with simple arithmetic addition for supplying node activation. The activation is compared with threshold. If it exceeds threshold, then unit produces high output value (‘1’), otherwise output will be ‘0’.

The network of such kind of artificial neurons can be referred as artificial neural network (ANN).
5.1.1 Types of neural network structures

There are different kinds of network structures, having different computational properties. In major categorization, feed forward and recurrent networks can be considered. In feed forward networks, links are unidirectional and no cycles will be there. In recurrent networks, links can create arbitrary topologies [2].

A *feed-forward network* computes a function of the input values that depends on the weight settings; it has no internal state other than the weights themselves. Such networks can implement adaptive versions of simple reflex agents or they can function as components of more complex agents. The figure 5.1 shows example of feed forward network.

![Two layered feed forward network](image)

Figure 5.1: Two layered feed forward network

The human brain can be considered as an example of *recurrent network*. Here the activation is fed back to the units which caused it. The recurrent networks have internal state stored in activation levels of the units. Here the computation will be less orderly than feed forward network. They can become unstable and learning will be more difficult. Hopfield networks can be considered as best example of recurrent network.

The network shown in figure 5.1 is very simple feed forward network. On the left side are the input units. The activation value of each of these units is determined by the environment. At the right-hand side end of the network are four output units. In between, the nodes labeled H₃ and H₄ have no direct connection to the outside world. These are called hidden units, because they cannot be directly observed by noting the input/output behavior of the network.
The perceptrons can be considered as another type of network where hidden units will not be there. The learning can be very simple with this kind of network. Networks with one or more than one hidden units are called multilayer networks. Perceptrons are the example of layered feed forward network studied first in 1950s. Now a day, single layer feed forward networks are better known as perceptrons.

![Perceptron Network](image1)

![Single Perceptron](image2)

Figure 5.2 Perceptrons

Most neural network learning algorithms, including the perceptron learning method, follow the current-best-hypothesis (CBH) scheme. Here the hypothesis is a network, defined by the current values of the weights. The initial network has randomly assigned weights, usually from the range [-0.5,0.5]. The network is then updated to try to make it consistent with the examples. This is done by making small adjustments in the weights to reduce the difference between the observed and predicted values. The main difference from the logical algorithms is the need to repeat the update phase several times for each example in order to achieve convergence.
5.2 Neural network back propagation

Similar to single layer feed forward network called perceptrons, learning can be applied to multilayer feed forward networks. The popular learning method with multilayer feed forward network is called back propagation.

This back propagation was introduced by Bryson and Ho in 1969. But it was not popular till 1980s.

The major issue with multilayer feed forward network is to identify no. of hidden units for learning. Learning in such a network proceeds the same way as for perceptrons: example inputs are presented to the network, and if the network computes an output vector that matches the target, nothing is done. If there is an error, a difference between the output and target, then the weights are adjusted to reduce this error. In perceptrons, this is easy, because there is only one weight between each input and the output. But in multilayer networks, there are many weights connecting each input to an output, and each of these weights contributes to more than one output. Back propagation approach is sensible to divide the contribution of each weight.

Similar to perceptron learning algorithm, try to minimize the error between each target output and the output actually computed by the network. With the output layer, the weight update rule is similar to the rule for the perceptron. There are two differences: the activation of the hidden unit is used instead of the input value; and the rule contains a term for the gradient of the activation function.
5.3 Applications of neural network approach in biometrics

Now a day, neural network concept is used in every area of information technology to improve the performance of the system. Here are some applications of neural network approach in fingerprint and face recognition.

5.3.1 Applications of neural network approach in fingerprint recognition

There are several reasons to prove that neural network approaches are well suited for fingerprint problems. First, fingerprints form a very specific class of patterns with very peculiar flavor and statistical characteristics. Second, neural networks can avoid some of the drawbacks inherent to other more conventional approaches. Third, neural networks are robust, adaptive, and trainable from examples [28]. This is particularly important as fingerprint images can include several different sources of deformation and noise ranging from the fingers and their positioning on the collection device. [13]

The neural network concept was first implemented by Leung, Engeler, and Frank (1990) [3]. They introduced a neural network-based approach where a multi-layer perceptron analyzes the output of a rank of Gabor filters applied to the gray-scale image. The image is first transformed into the frequency domain where the filtering takes place; the resultant magnitude and phase signals constitute the input to a neural network composed of six sub-networks, each of which is responsible for detecting minutiae at a specific orientation; a final classifier is employed to combine the intermediate responses.

Maio and Maltoni [4] used a shared-weights neural network to verify the minutiae detected by their gray-scale algorithm [5]. The minutiae neighborhoods in the original gray-scale image are normalized, with respect to their angle and the local ridge frequency, before passing them to a neural network classifier, which classifies them as termination, bifurcation, and non-minutia.

Typical three-layer neural network architecture has been adopted, where a partial weight sharing allows the termination/bifurcation duality to be exploited (Figure 5.4). In fact, the weight sharing requires the same type of processing to be performed by the first layer of neurons both on the positive and the negative neighborhoods. This network has more degrees of freedom with respect to a three-layer (26-10-2) perceptron trained both on the
positive and the negative versions of the same neighborhood, and used twice for each classification.

![Diagram of neural network architecture](image)

Figure 5.4: The neural network architecture to classify gray-scale minutiae neighborhoods into termination, bifurcation, and non-minutiae (Maio and Maltoni).

Most of the existing fingerprint classification methods can be coarsely assigned to one of these categories: rule-based, syntactic, structural, statistical, neural network-based and multi-classifier approaches.

In 1990s the neural network based fingerprint classification method became successful with compare to traditional methods. Most of the proposed neural network approaches are based on multilayer perceptrons and use the elements of the orientation image as input features ((Hughes and Green) [6], Kamijo, Mieno, and Kojima [7], and Pal and Mitra [8]). Kamijo [9] presents an interesting pyramidal architecture constituted of several multilayer perceptrons, Jain, Prabhakar, and Hong [10] train 10 feed forward neural networks to distinguish between each possible pair of classes. Some researchers proposed the use of self-
organizing neural networks. In Moscinska and Tyma [11], a Kohonen map is trained to find delta points, and a rule-based approach is applied for the final classification; in Halici and Ongun [12] a multilayer self-organizing map provides the classification. Now let us see some work carried after 1990s. J. Urias, D. Hidalgo, P. Melin, O. Castillo proposed a new method for response integration in modular neural networks using type-2 fuzzy logic [14]. Biometric authentication is used to achieve person recognition. Biometric characteristics like face, fingerprint, and voice are used. A modular neural network of three modules is used. Each module is a local expert on person recognition based on each of the biometric features. The response integration approach of the modular neural network has the objective of integrating the responses of the modules to enhance the recognition rate of the individual modules. The results of a type-2 fuzzy logic approach for response integration has shown higher performance over type-1 fuzzy logic approaches.

Wang, H.; Min, L.Q. & Liu, J. proposed the cellular neural/nonlinear network (CNN) as a powerful tool for fingerprint feature extraction. They presented two theorems for designing two kinds of CNN templates. These two theorems provided the template parameter inequalities to determine parameter intervals for implementing the corresponding functions [15].

Sitalakshmi Venkataraman proposed a Grid-based neural network framework for adopting multimodal biometrics with the view of overcoming the barriers of performance, privacy and risk issues that are associated with shared heterogeneous multimodal data centre. The framework combines the concept of Grid services for reliable brokering and privacy policy management of shared biometric resources along with a momentum back propagation ANN (MBPANN) model of machine learning for efficient multimodal fusion and authentication schemes. [16]

C. Park, M. Ki, J. Namkung, and J.K. Paik suggested that introducing momentum backpropagation ANN improves the accuracy. [17]

B. Jayaraman, C. Puttamadappa, E. Anbalagan, E Mohan and Srinivasrao madane suggested back propagation algorithm to obtain higher accuracy in fingerprint recognition. They implemented ANN after minutiae filtering step and back propagated in to the network, until the desired performance of the authentication mechanism. [18]
Fawaz Alsaade suggested resilient back propagation training algorithm to combine information from separate modalities to provide complementary data. His experimental investigations involved the recognition mode of verification in mixed quality data conditions. He deployed this at score level and found that system error can be reduced considerably [19].

5.3.1.1 Effect on Performance
Mehran Yazdi, and Kazem Gheysari performed experiments of classification of fingerprint with feed forward neural network approach. They used two databases and they trained the neural network. They were able to achieve 99.02% accuracy in classification with four classes. Anil Jain, Salil Prabhakar and Lin Hong trained a multilayer feed forward network using a quick propagation training algorithm. The neural network was having 20 neurons in one hidden layer, 192 neurons at input layer and 5 output neurons for 5 classes. They were able to achieve 86.4% accuracy for five class problem. For four class problem they achieved the accuracy of 92.1% [21].

5.3.2 Applications of neural network approach in face recognition
In chapter 3, the researcher has mentioned various methods and techniques of face recognition. Along with various methods of face recognition, it is possible to use the concept of neural network approach for recognition of face. Here are few cases where neural network approach has been used. P. Latha et al. [22] designed a face recognition system with PCA using neural network approach and experimented Yale University database.

<table>
<thead>
<tr>
<th>No. of images</th>
<th>Acceptance ratio (%)</th>
<th>Execution time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PCA</td>
<td>PCA with BPNN</td>
</tr>
<tr>
<td>40</td>
<td>92.4</td>
<td>96.5</td>
</tr>
<tr>
<td>60</td>
<td>90.6</td>
<td>94.3</td>
</tr>
<tr>
<td>120</td>
<td>87.9</td>
<td>92.8</td>
</tr>
<tr>
<td>160</td>
<td>85.7</td>
<td>90.2</td>
</tr>
<tr>
<td>200</td>
<td>83.5</td>
<td>87.1</td>
</tr>
</tbody>
</table>

Table 5.1: Comparison of acceptance ratio and execution time for Yale database images
They experimented with 40, 60, 120, 160 and 200 images database and derived the results shown in table 5.1. They concluded that when BPNN technique is combined with PCA, non linear face images can be recognized easily. This method had the acceptance ratio of more than 90 % and execution time of only few seconds.

In a study that involved morphological feature extraction from hand-shape images, a pattern spectrum was used as input to recognition systems. In this study, neural networks and support vector machine (SVM) techniques were employed for identification. The verification case was analyzed through Euclidean distance classifier, obtaining the False Acceptance Rate (FAR) and False Rejection Rate (FRR) of the system for a number of K-fold cross validation experiments. The identification rate was found to be as high as 98.15%. The results indicated that the pattern spectrum represents a good alternative of feature extraction for biometric applications and also form pertinent input sequences for neural networks [23].

Attempt has been made to investigate the effects of facial artifacts on the recognition rate of Eigen face based neural networks. It has been found that Eigen faces coupled with Euclidean distance can be successfully used to recognize the human face in almost real-time, as facial artifacts can cause the features that characterize a face to be distorted [24]. Efforts were directed to identify problematic facial artifacts that can cause false identification or no identification. The focus of the research was the investigation of common facial artifacts on the performance of recognition and the proposition of modification to existing databases to improve the positive rate of identification. A professional graphic artist was used to modify the images used in the experiments. Single and multiple Eigen face based neural network are used as classifiers [25].

Nonlinear classification for face detection can be performed using neural network approach. With the neural methods [26], a classifier may be trained directly using preprocessed and normalized face and nonface training subwindows. Rowley et al. [26] use the preprocessed $20 \times 20$ subwindow as an input to a neural network. The network has retinal connections to its input layer and two levels of mapping. The first level maps blocks of pixels to the hidden units. There are 4 blocks of $10 \times 10$ pixels, 16 blocks of $5 \times 5$ pixels, and 6 overlapping horizontal stripes of $20 \times 5$ pixels. Each block is input to a fully connected neural network and mapped to the hidden units. The 26 hidden units are
then mapped to the final single-valued output unit and a final decision is made to classify the $20 \times 20$ subwindow into face or nonface. Several copies of the same networks can be trained and their outputs combined by arbitration (ANDing) [26].

5.3.3 Applications of neural network approach in multibiometrics

The concept of neural network approach has also been used with multibiometrics and multimodal biometrics. Here are few cases where neural network approach has been implemented with either multibiometrics or multimodal biometrics.

Sitalakshmi Venkataraman has proposed grid based neural network framework for multimodal biometrics [27]. Typically, a Grid consists of certain basic and advanced functions that are inherent features of Grid information services. They are listed below and could be leveraged for processing multimodal biometric transactions:

- Discovery and Brokering
- Monitoring
- Policy controlling
- Security
- Resource Management

The proposed Grid-based framework used a momentum back propagation method of artificial neural network (MBPANN) for incorporating multimodal biometric fusion schemes within the requested biometric information services. It provided the flexibility at the client services layer for both users and business transactions to choose the suitable biometric modalities that are compatible with the user preferred and transaction-specific risk levels that are assigned for different business applications. An overview of the framework is depicted in Figure 5.5, which shows the four main layers and their major components.
Grid-based neural network framework that uses momentum back propagation for multimodal biometric fusion and authentication is capable of taking advantage of Grid services for seamless integration and information sharing among large, heterogeneous and distributed multimodal data centers. Further, by combining with the MBP-ANN model, the proposed framework caters to real-time performance and accuracy of complex fusion schemes, user-centric privacy policy schemes and adaptable risk schemes that have recovery mechanisms for cancelling and re-issuing multimodal biometric identities.

Karbhari V. Kale et al. proposed multimodal biometric system using neural network approach for finger knuckle and nail recognition [29]. They selected multilayer perceptron with backpropagation learning algorithm. They collected dorsal hand images...
of 100 objects. For each person they collected 6 images – 3 of right hand and 3 of left hand. So, total 600 dorsal hand images were collected for database. They conducted extensive experiments for evaluation of effectiveness and robustness. The performance of system proposed by them was 97%.

5.4 Building multimodal biometric system using face and fingerprint recognition using neural network approach with weight adjustment

In chapter-4, the researcher has mentioned multimodal biometric system with face and fingerprint recognition. The performance achieved with this system was 93.33 % at optimum weights. But the problem with that system was about weight assignment of face samples. If weight of face sample is more, the performance of the system will be less. The weights could not be adjusted automatically. The researcher has built a model in which weight adjustment will be done automatically. And performance of the system can be improved with this approach. The basic model of the system is shown below: The model of proposed basic multimodal biometric system is shown in figure 5.6.

![Diagram of multimodal biometric system](image)

Figure 5.6: Basic model of multimodal biometric system with neural network approach

The system model operation starts with capturing face and fingerprint sample with the process described in chapter 3. The captured face and fingerprint samples will be compared with respective master databases. The Euclidean distances and minutiae scores of each comparison will be stored in the files. Weights will be entered for face and
fingerprint scores. These weights will be multiplied with respective matching scores and summation of face and fingerprint scores will be done. Maximum match score will be identified and person index for respective maximum matching score will be searched.

Figure 5.7: Multimodal biometric system model with neural network approach
This person index will be matched with person index entered in Edit control by the user. If both indexes are matched, then successful identification will be there. Otherwise weights will be adjusted with neural network and match scores will be recalculated. The weight adjustment process will continue until face and fingerprint weights reaches min and max values or person is identified successfully. Detailing is shown in figure 5.7.

5.4.1 Flowchart representation of multimodal biometric system using neural network approach

![Flowchart of Multimodal Biometric System](image)

Figure 5.8: Flow chart representation of Multimodal biometric system
The flowchart representation of the proposed system is shown in figure 5.8.

5.4.2 Neural network architecture for multimodal biometric system

Another aspect to represent here is the use of concept of neural network approach. The researcher has used the concept of feed forward neural network for dynamic adjustment of the weights. The structure of the basic neural network can be represented in figure 5.9.

![Neural network structure for multimodal biometric system using face and fingerprint recognition](image)

Here, \( F_1 \) and \( Fp_1 \) are the face matching and fingerprint matching scores respectively. \( W1 \) and \( W2 \) suggested weights entered by the user. \( W1 \) represent weight of face match score and \( W2 \) represent weight of fingerprint match score. \( H_{11} \) and \( H_{12} \) are the hidden layer values, which are derived by multiplying \( W1 \) and \( W1 \) with \( F_1 \) and \( Fp_1 \) respectively. The hidden layer 2 calculate sum of the values of \( H_{11} \) and \( H_{12} \). The summation value is then
sent to output layer. These operations are carried for the values of $F_2$ and $F_{p2}$ and up to $F_{30}$ and $F_{p30}$. So, at output layer, there will be 30 values starting from $S_1$ to $S_{30}$. All these values will be compared and maximum score will be identified and respective person index will be shown as $O_1$. Suppose the $O_1$ is not matching $O_p$, then values of $W_1$ and $W_2$ will be adjusted and again the process will be done at hidden layer 1 and hidden layer 2. This weight adjustment process will continue until system identifies the correct matching of $O_1$ and $O_p$ or values of weights $W_1$ and $W_2$ reaches at their maximum and minimum value i.e. 0.9 and 0.1 respectively. The GUI representation of the system working is shown in section 5.4.3.

5.4.3 GUI representation of multimodal biometric system using neural network approach

The researcher has developed multimodal biometric system GUI which provides facility of unimodal biometric system and multimodal biometric system also. The unimodal face and fingerprint biometric system working is shown in chapter 3. The multimodal biometric system without use of weight adjustment in neural network approach is used in chapter 4. Here is the representation of multimodal biometric system using neural network approach with weight adjustment.

*Step-1: Initially, run multimodal_ide.m file to load multimodal IDE.*

The user has to run multimodal_ide.m file from the directory. On executing this file, system will load multimodal IDE.

*Step-2: Select option ‘Multimodal with backpropagation’ from menu options*

On the IDE, user will find two menu options on the top : Unimodal and Multimodal. The user has to select Multimodal menu option and in that, selection multimodal with backpropagation option.
Figure 5.10: Load multimodal IDE

Figure 5.11: Selection of ‘Multimodal with backpropagation’ option from menu
The user has to select first ‘Multimodal; menu and in that menu, select ‘Multimodal with backpropagation’ option.

**Step-3:** System displays respective components on GUI representation.

On selection of the ‘Multimodal with backpropagation’ option, system will hide components which are not required in this process. The components which will be visible are axes controls and edit controls. Only four buttons will be visible in controls panel. The user has to enter face and fingerprint weights and person index for verification.

**Step-4:** Press ‘Load face image’ button for selection of sample face image.

The next step is to load sample face and fingerprint images. First, the user has to load face image in axes control1. The dialog box will open showing directories. Select directory from which sample face image is to be loaded.

![Figure 5.12: System displaying necessary components on GUI](image)
Step-5: Select sample face image from directory and open it in axes control1.

The user has to select sample face image with .pgm file format and open that image on GUI with axes control1.

Step-6: Press ‘Load fingerprint image’ button and select sample fingerprint image from directory.

After selection of sample face image, next step is to select sample fingerprint image. For that, press ‘Load fingerprint image’ from GUI and select sample fingerprint image with file type of ‘.jpg’ from directory and image will be loaded on axes control2. Figure 5.15 shows this process.

Step-7: System will show both sample images in respective axes controls.

After selection of both the sample images, system will show sample face image on axes control1 and sample fingerprint image on axes control2 as shown in figure 5.16.
Figure 5.14: Selection of sample face image with .pgm format from directory

Figure 5.15: Selection of sample fingerprint image
Step-8: Enter face and fingerprint weight scores (values) in edit controls.

The next step is to enter face and fingerprint weight scores or values in edit controls. The sum total of both values must be 1.0. So, range can be from 0.0 to 1.0 for both weights. The user can also start from 0.9 for face and 0.1 for fingerprint ideally. The user has to enter the value of person index in the range of 101 to 130 for his/her verification. Then press the button ‘Match face and fp with BPN’.

Step-9: System displays maximum match score value and person index for multimodal identity

After pressing match button, the system will start verification process for the user. Here the process starts with comparison of sample with master database faces and Euclidean distances are calculated. Similarly, sample fingerprint is compared with db.mat fingerprints and minutiae scores are generated. Both the scores are stored in finalfacescores1 and finalfpscores1 txt files. Then scores are multiplied with respective weights and updated scores will again stored in array. Next step is to make summation of multiplied face scores and fingerprint scores of respective persons. The next step is to
identify maximum score and getting the person index. This person index is then compared with person index entered in edit control on IDE. If person indexes are match, then it will be considered as success, otherwise weights will be adjusted with neural network. Face weight is decreased by 0.1 and fingerprint weight is increased by 0.1. The process will continue until finding successful verification or scores reaches maximum of minimum values.

**Step-10: Press ‘Clear all’ button to clear content on GUI.**

At last, the user has to press ‘Clear all’ button to clear the contents from axes controls and edit controls from GUI. Then he/she can proceed for next verification process.

Figure 5.17: Enter the value of face and fingerprint weights and person index
Figure 5.18: System will display highest match score and multimodal identity on GUI

Figure 5.19: Press ‘Clear all’ button to clear all content on axes and edit controls
5.5 Experimentation results

The performance measurement of the model has been done carefully. The researcher has prepared 5 different sets, each set containing 30 face samples and 30 fingerprint samples (30 + 30 = total 60 samples) selected arbitrarily from test database. So, total 150 face images and 150 fingerprint images are used in testing performance of the system. The test images are compared with master database. The results are shown here in the table 5.2.

Here is the comparison of the performance of multimodal biometric system and multimodal biometric system with neural network approach with 5 sets of samples. Master face database contains 60 images and master fingerprint database contains 30 images. So, out of total 300 samples, rest 240 face images will be considered as test face database and 270 fingerprint images will be considered as test fingerprint database. From these, set 1 will be created by selecting 30 face images and 30 fingerprint images. Set 2 will be created using rest face 210 images and 240 fingerprint images. The same process will continue for set 3, set 4 and set 5.

<table>
<thead>
<tr>
<th>Method</th>
<th>FW-FPW</th>
<th>Set 1(%)</th>
<th>Set 2(%)</th>
<th>Set 3(%)</th>
<th>Set 4(%)</th>
<th>Set 5(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multimodal Biometric system with static weights</td>
<td>0.9 - 0.1</td>
<td>76.67</td>
<td>80.00</td>
<td>83.33</td>
<td>80.00</td>
<td>80.00</td>
</tr>
<tr>
<td></td>
<td>0.8 - 0.2</td>
<td>86.67</td>
<td>83.33</td>
<td>83.33</td>
<td>86.67</td>
<td>90.00</td>
</tr>
<tr>
<td></td>
<td>0.7 - 0.3</td>
<td>90.00</td>
<td>83.33</td>
<td>86.67</td>
<td>90.00</td>
<td>90.00</td>
</tr>
<tr>
<td></td>
<td>0.6 - 0.4</td>
<td>96.67</td>
<td>83.33</td>
<td>93.33</td>
<td>90.00</td>
<td>90.00</td>
</tr>
<tr>
<td></td>
<td>0.5 - 0.5</td>
<td>96.67</td>
<td>86.67</td>
<td>96.67</td>
<td>93.33</td>
<td>90.00</td>
</tr>
<tr>
<td></td>
<td>0.4 - 0.6</td>
<td>96.67</td>
<td>93.33</td>
<td>96.67</td>
<td>96.67</td>
<td>93.33</td>
</tr>
<tr>
<td></td>
<td>0.3 - 0.7</td>
<td>96.67</td>
<td>90.00</td>
<td>100.00</td>
<td>96.67</td>
<td>93.33</td>
</tr>
<tr>
<td></td>
<td>0.2 - 0.8</td>
<td>100.00</td>
<td>90.00</td>
<td>100.00</td>
<td>96.67</td>
<td>96.67</td>
</tr>
<tr>
<td></td>
<td>0.1 - 0.9</td>
<td>100.00</td>
<td>93.33</td>
<td>100.00</td>
<td>96.67</td>
<td>96.67</td>
</tr>
<tr>
<td>Multimodal Biometric system Neural network approach</td>
<td>100.00</td>
<td>96.67</td>
<td>100.00</td>
<td>100.00</td>
<td>96.67</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.2: Performance of multimodal biometric system with neural network approach

The Matlab code implementation of the system is shown here:

%multimodal_backpropagation.m
% Open files with face score and fingerprint score
fid1=fopen('finalfacescore1.txt','r');
fid2=fopen('finalfpscore1.txt','r');
fid3=fopen('weightdatabase1.txt','wt+');
facescore = [];
fpscore = [];
fcount=1;
count=100;
tscore=0.0;
success=0;
failure=0;

faceweight=0.9;
fpweight=0.1;

%fetch face score and fingerprint score in array
facescore=fscanf(fid1,'%f',900);
fpscore=fscanf(fid2,'%f',900);

% first loop for 30 persons
for i=1:30
    maxscore=0;
    person=count+i;
totalscore = [];
index=0;
tempface = [];
tempfp = [];
faceweight=0.9;
fpweight=0.1;

% second loop for each person's score comparison
for j=1:30
    tempface = [tempface facescore(fcount)];
    tempfp = [tempfp fpscore(fcount)];
fcount=fcount+1;
end

k=1;
% loop will go for nine times for nine different score combinations. If
% person matches with index than success else failure and loop will
% stop on finding success.
while (k<=9)
    %call identyperson function to get matching index
    [index]=identyperson(person,faceweight,fpweight,tempface,tempfp);
    if(person==index)
        fprintf(['in success 
']);
success=success+1;
strl='success';
        fprintf(fid3,'%d %t %f %t %f %t %s
', ,person,faceweight,fpweight,strl);
brea
    else
        fprintf(['in failure']);
failure=failure+1;
end
end
% On failure in matching person with index weights are adjusted.  
% fpweight is increased by 0.1 and faceweight decreased by 0.1

[fp,fw]=adjustweight(faceweight,fpweight);  
k=k+1;  
faceweight=fp;  
fpweight=fw;  
if(k==10)  
strl='failure';  
fprintf(fid3,'%d \t %f \t %f \t %s \n', person,  
faceweight, fpweight,str1);  
fprintf(['Failure to identify \n']);  
end  
end
end

% Calculate success rate
successrate=(success/30)*100;  
fprintf(['Successful identification : ' num2str(success) '\n']);  
str2='successrate=';  

% Write success rate in file  
fprintf(fid3,'%s \t %f \n',str2,successrate);

fclose(fid1);  
fclose(fid2);  
fclose(fid3);  
-----------------------------------------------------------------------
% identifyperson.m

function [index]=identifyperson(person,faceweight,fpweight,tempface,tempfp)  
maxscore=0;  
totalscore = [];  
for j=1:30  
tscore=(tempface(j)*faceweight) + (tempfp(j)*fpweight);  
totalscore = [totalscore tscore];  
if (j==1)  
maxscore=totalscore(j);  
index=1;  
else  
if(maxscore<totalscore(j))  
maxscore=totalscore(j);  
index=j;  
end  
end
end

% Add 100 to index having max score of face and fingerprint
index=index+100;

-----------------------------------------------------------------------
% adjustweight.m
function [fw,fp]=adjustweight(faceweight,fpweight)

% increase fpweight and decrease faceweight

faceweight=faceweight-0.1;
fpweight=fpweight+0.1;

% return faceweight and fpweight
fw=faceweight;
fp=fpweight;

%multimodal_weight.m
fid1=fopen('finalfacescore5.txt','r');
fid2=fopen('finalfpscore5.txt','r');
fid3=fopen('multimodalweightresults5.txt','a');
facescore = [];
fpscore = [];
fcount=1;
count=100;
tscore=0.0;
success=0;
failure=0;

faceweight=0.1;
fpweight=0.9;

tempface = [];
tempfp = [];

facescore=fscanf(fid1,'%f',900);
fpscore=fscanf(fid2,'%f',900);

for i=1:30
    maxscore=0;
    person=count+i;
    totalscore = [];
    index=0;

    for j=1:30
        tempface = [tempface facescore(fcount)];
        tempfp = [tempfp fpscore(fcount)];
        tscore=(facescore(fcount)*faceweight) +
(fpscore(fcount)*fpweight);
        totalscore = [totalscore tscore];
        fcount=fcount+1;
        if (j==1)
            maxscore=totalscore(j);
            index=1;
        else
            if(maxscore<totalscore(j))
                maxscore=totalscore(j);
                index=j;
            end
        end
    end
end
end
index=index+100;
if(person==index)
    success=success+1;
else
    failure=failure+1;
end
%    pause(2);
end
successrate=(success/30)*100;
fprintf(['Successful identification : ' num2str(success) '\n']);
fprintf(['Failed identification : ' num2str(failure) '\n']);
fprintf(fid3,'%f \t %f \t %d \t %d %f\n',faceweight,fpweight,success,failure,successrate);
fclose(fid1);
fclose(fid2);
fclose(fid3);

The code contains basically four Matlab .m files. Two functions namely – adjustweight.m and identifyperson.m have been defined for ease of programming. Matlab file multimodal_weight.m has been created for evaluation of system with face and fingerprint weights. Matlab file multimodal_backpropagation.m is created for evaluation of the system performance with backpropagation. The graphical representation of the performance is shown in figure 5.20.

Figure 5.20: Performance of multimodal biometric system with neural network approach
5.6 Conclusion

The performance comparison of two approaches – multimodal biometric system and multimodal biometric system with neural network approach is shown in table 5.2. From the results, it is clear that system performance with neural network approach with face and fingerprint recognition is improved over the multimodal biometric system using face and fingerprint recognition without weight adjustment shown in table 4.9. It can be observed that in set 1,3 and 5, it was possible to achieve 100% GAR, while in set 2 and 4, it was possible to achieve 96.67% GAR. Over all GAR was achieved up to 98.67%, which is improvement over the experimentation results of system model explained in chapter 4.

References


