Expert Systems with Applications 38 (2011) 5940–5946

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Iris recognition using artificial neural networks

Fadi N. Sibai [*], Hafsa I. Hosani, Raja M. Naqbi, Salima Dhanhani, Shaikha Shehhi

Faculty of Information Technology, P.O. Box 17551, UAE University, Al Ain, United Arab Emirates

ARTICLE INFO

Keywords:
Iris recognition
Feedforward neural networks
Backpropagation training algorithm
Image data partitioning techniques

ABSTRACT

Biometrics recognition is one of the leading identity recognition means in the world today. Iris recognition is very effective for person identification due to the iris' unique features and the protection of the iris from the environment and aging. This paper presents a simple methodology for pre-processing iris images and the design and training of a feedforward artificial neural network for iris recognition. Three different iris image data partitioning techniques and two data codings are proposed and explored. BrainMaker simulations reveal that recognition accuracies as high as 93.33% can be reached despite our testing of similar irises of the same color. We also experiment with various number of hidden layers, number of neurons in each hidden layer, input format (binary vs. analog), percent of data used for training vs testing, and with the addition of noise. Our recognition system achieves high accuracy despite using simple data pre-processing and a simple neural network.

1. Introduction

Biometric techniques (Delac & Grgic, 2004) such as recognizing one's fingerprints, face, iris, and voice greatly help in person identification, authentication, and authorization. Iris recognition is better than other recognition methods such as fingerprint and face recognition. This is because each person's iris is unique – even for twins – and hardly changes while other technologies based on fingerprint or voice recognition can be compromised as fingerprints may get damaged by burning or by taking some antibiotics and voices can be altered by colds. In fact, the eyelid, cornea and aqueous humor protect the iris. Furthermore, the iris is relatively immune to aging, and the wearing of contact lenses or glasses. Therefore, iris recognition is a biometric technique which can be trusted for producing accurate and correct results. As irises differ in size, shape, color, and patterns, they offer high confidence for recognizing a person's identity by mathematical analysis. Airports such as the Dubai International airport have been using iris recognition technology to scan travelers. Other secure offices such as banks and government offices could also be protected with this technology. Typically, iris recognition software based on advanced mathematical techniques such as Wavelets and real-time hardware with high resolution camera(s) is employed. In this work we take a different approach aiming at simplifying the iris recognition system and reducing its cost. Our approach is characterized by: (i) the use of artificial neural networks; and (ii) simple mathematical analysis and preprocessing of the iris data based on a very basic partitioning of the iris image, and summing the RGB values in each partition.

The paper presents a methodology for preprocessing iris images and getting them ready for artificial neural network processing. The paper also presents the design and training of an artificial neural network for recognizing irises.

The paper is organized as follows. Section 2 reviews related and background work. In Section 3, the preprocessing steps are detailed. Biometric recognition is defined along with its importance with focusing on irises. Section 4 presents the three image data partitioning techniques. Section 5 describes and discusses our simulation experiments. The simulation results are presented in Section 6. Finally, Section 7 draws some conclusions.

2. Related and background work

2.1. Iris recognition

Over the years it has been known that the iris patterns have a wonderful and rich structure and are full of complex textures in contrast to other physiological characteristics. The iris is the colored circular part of the eye between the pupil and sclera. Bertillon (1885) reported to be the first to have suggested the use of the iris feature and color for person identification. When security is highly desired, iris recognition is a preferred method of identification because the iris patterns are hardly affected by the environment and can hardly be lost. The iris texture is unique from one person to another. Some of the iris features include furrows, ridges, arching ligaments, zigzag collarets, and crypts. Based on the above reasons, for over a decade, England, the US, Japan, and Germany applied iris

* Corresponding author. Tel.: +971 3 713 5589; fax: +971 3 7672018.
E-mail address: fadi.sibai@uaeu.ac.ae (F.N. Sibai).

recognition in ATM environments as a verification tool for accessing bank accounts. This reduced the need for customers to enter a PIN or passwords. Moreover, airports have begun to use iris scanning for identification for employees as they are moving through secure areas. Other applications include monitoring prison transfers and releases, as well as projects designed to authenticate online purchasing, banking, voting and stock trading.

We next review relevant background work. Daugman (2002) used 2D Gabor filters and phase coding to generate a 2048 binary feature code for the iris. Wildes (1994) used the Hough transform to locate the iris and a Laplacian pyramid with four resolution levels to generate the iris code. Boles and Boashash (1998) built a 1D representation of the grey level signature of the iris and applied to it zero-crossing of the dyadic wavelet transform to generate the iris representation. Ma, Wang, and Tan (2002) used a bank of Gabor filters to capture the iris profile. Based on a 2D Haar wavelet, Lim, Lee, Byeon, & Kim (2001) extracted high frequency information to generate an 87 binary code and employed an LVQ neural network for classification. Based on phase of Daubechies wavelet, Poursaberi (2005) generated a binary code representation of the iris and used a minimum Euclidian distance for matching. This iris recognition system consists of six major steps as follows: (1) preprocessing including image capturing, (2) image filtering and enhancement, (3) image iris localization and iris normalization, (4) iris de-noising and enhancement, (5) iris feature extraction, and (6) iris feature classification. Their suggested algorithm was tested on the CHUK iris database and an acceptable success rate was obtained. Binary coding in feature extraction stage also caused the matching process more quickly and easily. Another novel iris feature extraction technique with 1D circular profile of the iris and intelligent classifier based on probabilistic neural network and particle swarm optimization produced good recognition results (Chen and Chu, 2009).

Xu, Zhang, and Ma (2008) employed an intersecting cortical model (ICM) neural network to generate the iris codes and the Hamming distance between the compared iris codes. This ICM neural network is a simplified model of pulse-coupled neural network (PCNN), which has excellent performance for image segmentation, so the coding process is fast enough. Hassanien, Abraham, and Grosan (2009) presented a new iris recognition method based on Independent Component Analysis ICA. Hassanien et al. (2009) represented iris patterns with ICA coefficients, determined the centre of each class by competitive learning mechanism and finally recognized the pattern based on Euclidean distances. This method is insensitive to variant illumination and noise caused by eyelids and eyelashes, and even for blurred iris image, it can be recognized well. Murakami, Takano, and Nakamura (2003) used a rotation spreading neural network (R-SAN net) for iris recognition. This type of neural network is suitable for recognizing the orientation of the object regardless of its shape. In the recognition experiments, the rotation spreading neural network was able to simultaneously recognize the orientation of iris images of learned persons. Another work on iris recognition using neural networks includes (Broussard, Kennell, Ives, & Rakvic, 2008).

2.2. Other biometric techniques

A biometric system is essentially a pattern recognition system that operates by acquiring physiological and/or behavioral characteristic data from a person, extracting some features from the acquired data, and comparing these features against a recorded feature set in the database (Jain, Ross, & Prabhakar, 2004) for the purpose of determining or confirming the person's identity. Biometric applications include computer systems security, secure electronic banking, mobile phones, credit cards, secure access to buildings, health and social services. By using biometrics a person could be identified based on her/his physiological and/or identity rather than her/his possession (card, token, key) or her/his knowledge (e.g. password, PIN).

Desirable characteristics of a biometric recognition system include: (i) *universality*: the feature should apply to every person or special alternative tests should be administered to those who do not apply, e.g. blind or persons without fingerprints, (ii) *uniqueness*: the system should extract and compare a feature unique to each person, (iii) *longevity*: the feature should not vary with time, (iv) *collectability*: the feature must be easily collectible, (v) *accuracy*: the system should deliver accurate recognition, and (vi) *tampering*: the technique should be hard to tamper.

There is no ideal biometric that fits all needs. All biometric systems have advantages and disadvantages. Retinal scans target the capillary pattern in the retina but require close physical contact with a scanning device. Iris recognition does not require such close physical contact but iris recognition systems are expensive and require a large amount of computer storage. Fingerprint scans have criminal overtones and cannot be taken on persons with burnt fingers or those with fingerprints erased by antibiotic use. Hand scans require small amount of computer storage and again do not always work (e.g. hand injury). Facial imaging loses its accuracy under different facial hair patterns. Voice recognition also requires a large amount of computer storage and loses its accuracy when voices change due to illness or with the presence of background noise. Signature recognition is not effective when signatures are forged. For the above reasons, iris recognition is a preferred biometric technique.

2.3. Artificial neural networks

Artificial neural networks model biological neural networks in the brain and have proven their effectiveness in a number of applications such as classification and categorization, prediction, pattern recognition and control. An artificial neural network consists of an interconnected group of artificial neurons. Such a network performs computation and manipulates information based on the connectionist approach in a similar but simpler fashion than the brain would perform. Many types of artificial neural networks (Fausett, 1994) exist including feedforward neural networks, radial basis function (RBF) networks, Kohonen self-organizing networks, recurrent networks, stochastic neural networks, modular neural networks, dynamic neural networks, cascading neural networks, and neuro-fuzzy networks. Multi-Layer Perceptron (Haykin, 1998; Rumelhart & McClelland, 1996) (MLP) is perhaps the most popular, where neurons in a feedforward type network perform a biased weighted averaging of their inputs and this sum is then subjected to a transfer function, in order to limit the output value.

As depicted in Fig. 1, a neuron is made of a cell body bordered by a membrane, inside of which is a nucleus, across which incoming electric (or chemical) signals composed of polarized ions arrive via neuron inputs known as dendrites. The neuron output or

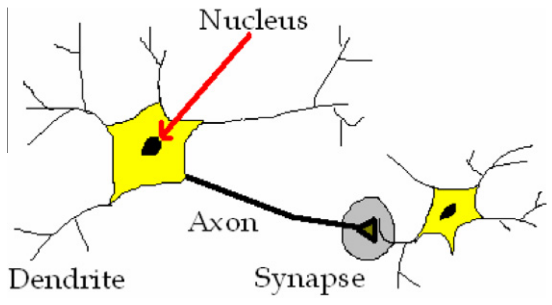

Fig. 1. Structure of a neuron network in the brain.

outgoing signal travels over a connector or terminal known as axon – where the neurotransmitters reside – and which connect to other neuron dendrites via synapses. Ten thousand neuron types are known to exist in the brain of different shapes and terminal densities. A neuron may have thousands of dendrites but only one axon. A neuron output-axon – connects to another neuron input-dendrite – in an area called a synapse where the axons terminate. When enough positive ions gather inside the neuron's membrane the neuron fires, i.e. a large electric signal is generated and travels out over the axon to reach the axon terminal. At electric synapses, the output is the electrical signal transmitted over the axon while at chemical synapses, the output is a neurotransmitter. Neurons are either sensory motor or inter-neuron. The first type conveys sensory information. The second type conveys motor information. The third type conveys information between neurons.

An artificial neuron models a real neuron as depicted in Fig. 2. First, electric signals from other neurons are multiplied by weights (represented by the rectangles in Fig. 2) and then are input into the artificial neuron. The weighted signal values are them summed by an adder ("Σ" in Fig. 2) and the sum is subjected to a transfer function ("T" in Fig. 2) which is one of: (i) linear, where the output is proportional to the weighted sum of inputs; (ii) threshold, where the output is one of two values based on whether the weighted sum is greater or smaller than the threshold value; and (iii) sigmoid, a non-linear function which most closely mimics real neurons. Artificial neural networks are composed of several artificial neurons as a real neuron network is composed of many real neurons. Artificial neural networks come in different forms and shapes.

Artificial neural network organizations are either feedforward, or recurrent. Feedforward networks do not have cycles and signals flow from input to output. Recurrent neural networks have loops, i.e. links to neurons in the same or previous layers. The MLP neural network organization, shown in Fig. 3, is an example of feedforward network. Hebbian networks are example of recurrent networks. Recurrent networks because of their feedback cycles go through dynamic state changes until the network stabilizes. Neural networks can be either fixed or adaptive. Fixed networks have unchanging weights usually set at the start and need not change as the network is expected to keep operating in the same way. Adaptive neural networks as those which allow their weights to change (i.e. allow the network to adapt and learn). Adaptive neural networks can therefore learn and their learning usually falls under two large classes: supervised or unsupervised. Supervised learning involves a supervisor who tells the network how to learn, i.e. what the network output should be for each input combination. In the supervised learning, the inputs and corresponding outputs are known, so the neural network learns by applying a "cost" function to generate the desired outputs for each input combination. Popular cost functions include the mean squared error function, and random functions. The mean squared error function attempts to minimize the error between the actual output value computed by the network and the desired output value. In unsupervised

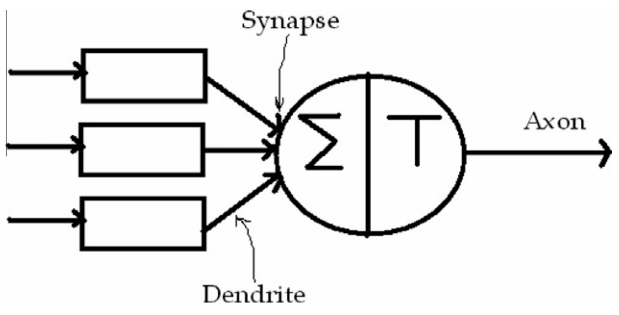

Fig. 2. Artificial neuron model.

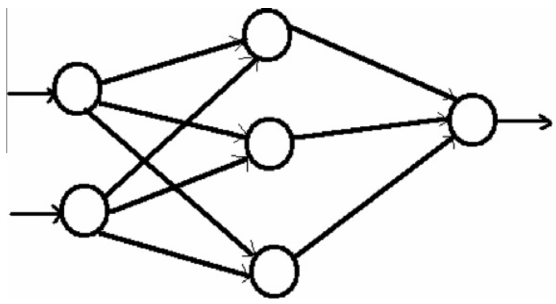

Fig. 3. MLP neural network.

learning, the network is not told of what the generated output should be for each input combination but the neural network learns by itself by self-organizing the data and identifying the data's characteristics and properties. Yet another class of learning is reinforcement learning. Reinforcement learning differs from the supervised learning problem in that correct data inputs and matching output are never presented to the network. In addition, suboptimal actions are not explicitly corrected. Instead, in reinforcement learning, agents interact with the environment and supply the data to the network which attempts to formulate a policy for agent actions which optimizes some cost function. Evolutionary algorithms and simulated annealing are examples of other types of neural network training methods.

The MLP is an example of feedforward artificial neural network with multiple layers and where each neuron output in one layer feeds as input to the neurons in the next layer as shown in Fig. 3.

We chose our artificial neural network for iris recognition of the feedforward type due to its simplicity and its suitability for this application.

We also employ the backpropagation algorithm for supervised training of our network, a well known and widely used algorithm. The training algorithm minimizes the error between the obtained output and the required target output by finding the lowest point or minimum in the error surface. Starting with initial weights and thresholds, the training algorithms look for the global minimum of the error surface. Usually the slope of the error surface at the present location and guides the next move down. It is the goal of the training algorithm to reach a low point on the surface which happens to be the local minimum, and in some unlucky situations, the algorithm stops at a local minimum. In the backpropagation algorithm, training moves the state along the steepest descent, reducing in each training epoch the error. The size of the move during each epoch is a function of the learning rate and the gradient (slope). At the end of each epoch, the weights are adjusted as a function of the error and the error surface gradient. The derivatives of the weights are computed to determine how the error changes as a result of increasing or decreasing the weights. The number of epochs taken to conduct training is determined by the error reaching a satisfactory value, or when the number of epochs reaches a predetermined number. Alternatively, training ends when the error stops shrinking.

Other training algorithms exist such as conjugate gradient descent – where after minimizing along one direction on the error surface, next moves assume that the second derivative along the first direction is held at zero – and Quasi-Newton training (Bishop, 1995).

3. Preprocessing

When confronting the task of gathering iris images for training and testing the neural network, we decided to select iris images of the same color (brown) in order to create more difficult situations for our recognition system to detect and achieve higher recognition accuracy. Up close, the irises that have been collected are different in their patterns and shapes although from a further distance, the irises look similar to each other. We collected and pre-processed 20 brown-colored iris images of different persons (from same side) from the iris database (Chek image database). The iris images were between 400 kB and 500 kB and were not ready for processing but had to be pre-processed. For instance the white sclera and black pupil are visible in all the images. Additionally, the relevant content of the binary iris image (with irrelevant headers, ...) was not ready to be fed to an artificial neural network for processing. Thus four pre-processing steps were taken:

1. Edit each image with Adobe Photoshop to convert it from png to jpeg (JPEG Specifications) format needed by the program in step 2. JPEG is a common format used for graphic images and has high resolution as it can support millions of colors. In addition, it is the most common format for storing and transmitting photographic images on the World Wide Web.
2. On each image, run a Java program to extract the 100×100 RGB values (Lin, xxxx) from the iris image's jpg file.
3. For each image, store 100×100 RGB pixel values in a Microsoft Excel spreadsheet. Next, in the spreadsheet, RGB values of sclera pixels close to 0 were replaced by 0. RGB values of pupil pixels close to FFFFFF were also replaced by 0. The replacement of RGB values of sclera and pupil pixels by 0 was done to avoid processing these non-iris areas. RGB values in row/column strips and blocks (see next section) were then summed.
4. The Excel spreadsheet content was converted to a .dat file input into the NetMaker (IEEE Expert Staff, 1992) application which generated the input files for the BrainMaker artificial neural network simulator (IEEE Expert Staff, 1992).

The first three steps are data pre-processing steps while the fourth step is a BrainMaker pre-processing step. Many signal processing and filtering techniques have been described in the literature to extract the iris or enhance the iris image. In our work we manually manipulated the images as our focus was on the design of the artificial neural network for iris recognition. However the first three pre-processing steps required manipulating the iris image can be simply automated. We used a published Java program (Lin, xxxx) to generate a file with the RGB values of all pixels. Because that program expected a jpeg image for input, we opened all 20 images with Photoshop, cut a slight portion from the top and bottom of the image to remove unwanted pixels, and saved them as jpg files. The Java program converted each pixel content into a 6 hex digit RGB value given by

$$\text{rgbValue} = \text{src.getRGB}(j,k) + 0\text{xFFFFFF} + 1 \tag{1}$$

White pixels had high RGB values (FFFFFF) while black ones had low RGB values (000000).

4. Iris image data partitioning

In this section, we describe how we converted a matrix of RGB values into data ready for neural network processing. After completion of the pre-processing step 2 of Section 3, we obtain a matrix of RGB values each with six hexadecimal numbers "R1R0G1G0B1B0" where R1R0 represent the hexadecimal red value, G1G0 represent the hexadecimal green value, and B1B0 represent the hexadecimal blue value. For simplicity, we treat the 6 hexadecimal digits as one number in the order generated by the Java program thus giving more weight to the red component of the pixel's RGB value. In theory, it is fair to treat red, green and blue colors with equal weights but as our results with concatenating the RGB numbers into one number were satisfactory, we did not experiment with uniform color weights.

As it is desired to reduce the cost of the artificial neural network, and as BrainMaker limits the number of neurons per layer, the iris image's RGB matrix had to be partitioned to reduce as much possible the number of values fed as input to the neural network. For that purpose, we considered three simple data partitioning techniques pictured in Fig. 4.

1. Horizontal strip partitioning (rows).
2. Vertical strip partitioning (columns).
3. Block partitioning.

In horizontal strip (row) partitioning, we divided the RGB matrix into r (=10, 25) horizontal strips and summed all the RGB values of all pixels falling in one horizontal strip. When r was set to 10 (25), and as the image contains 100×100 pixels, each horizontal strip contained the RGB values of 100/10 (25) = 10 (4) rows of 100 pixels each, and a 1000 (400) RGB values were summed into one number representing that horizontal strip.

Similarly, in vertical strip (column) partitioning, we divided the RGB matrix into c (=10, 25) vertical strips and summed all the RGB values of all pixels falling in one vertical strip. When c was set to 10 (25), and as the image contains 100×100 pixels, each vertical strip contained the RGB values of 100/10 (25) = 10 (4) columns of 100 pixels each, and 1000 (400) RGB values were summed into one number representing that vertical strip.

In block partitioning, the image was divided into b (=16, 25) square blocks. When b was set to 16 (25), and as the image contains 100×100 pixels, each block consists of 25×25 (20×20) pixels whose RGB values were summed into one number per block.

Each horizontal/vertical strip or block was thus represented by one number which was fed as one input into the artificial neural network for identifying a match/no match of the presented iris image.

Our data partitioning techniques are characterized by simplicity and fast processing (summing), compared to more complicated techniques based on wavelets, and are key to reducing the system cost.

5. Experiments

As the NetMaker accepts DAT files as inputs, these DAT files containing the sums of RGB values in the various strips and blocks previously discussed were prepared for that purpose. Neural network training and testing experiments were conducted for two different data encodings: binary and analog. In binary coding, each sum is converted into 6 bits for both horizontal and vertical strip partitionings and 4 bits for block partitioning. These numbers are governed by the maximum number of neurons per layer (=64) acceptable by the BrainMaker simulator. For that reason, for

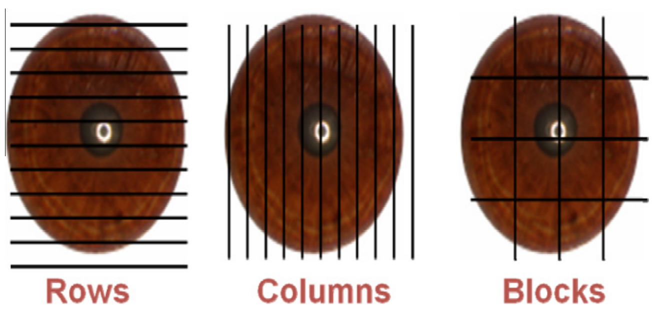

Fig. 4. Data partitioning techniques: horizontal (left), vertical (middle), block (right).

horizontal or vertical partitioning, we limited the number of neurons in the input layer of the simulated artificial neural networks to 60 analog inputs (one RGB sum per neural network input) and 60 binary inputs (10 RGB sums of 6 bits each). As the RGB sums are very large integer numbers we divided these numbers by a large constant (1 billion) to convert them into real numbers under 10.0. In analog coding, each of these real numbers was fed to a neurals network input. In binary coding, the real number was truncated and converted into the closest whole number under 10 and represented by a 6-bit binary number when the image is divided into horizontal or vertical strips. For instance, if the RGB sum of a horizontal strip, vertical strip or block, is 6311370815, it is then divided by 1000000000 to yield 6.311370815 which is then approximated to 6 which is then represented by 000110.

All in all, 6 DAT files were prepared: (1) horizontal strip (row) partitioning scenario with binary values, (2) horizontal strip (row) scenario with analog values, (3) vertical strip (column) with binary values, (4) vertical strip (column) scenario with analog values, (5) block partitioning scenario with binary values, and (6) block partitioning scenario with analog values. Each DAT file contains the RGB sum information for 20 iris images and each image information is mapped to a unique value that is used to distinguish the irises.

Iris recognition experiments using neural network BrainMaker software were performed then in two steps: training and testing. The DAT files were provided for the NetMaker application which prepared the input files required by the BainMaker simulator and which provided the capability to set (i) *tr,* the percentage of images for training the network and (ii) *te,* the percentage of images to test the network, where $tr + te = 100\%$. We later experimented with these percentages, setting (*tr*, *te*) to (25%, 75%), (75%, 25%), and (50%, 50%) and discovering that with a total of 20 iris images, (*tr*, *te*) = (25%, 75%) – i.e. 5 images for training and 15 images for testing – resulted in the best artificial neural network accuracy.

Three BrainMaker files were created by NetMaker: the definition file, the training fact file, and the testing fact file. The first two files are used to train the neural network while the third one is used in the testing phase, where images are fed into the neural network and the network determines if they match or do not match the iris image which the network was trained to recognize. After generating the files required by BrainMaker, the BrainMaker application was started to train the network first and then to test it. In the training phase, 5 iris images were used; one was presented to the artificial neural network as the image to "memorize' while the remaining 4 images were presented as false iris images. In the testing phase, 15 iris images including the true image which the network was trained to "memorize" and 14 false images were presented to the neural network which indicated with an output of 1 if the input image matches the true memorized iris image, and with an output of 0 that there is no match.

This process was repeated for various neural network topologies based on the same feedforward neural network of Fig. 5 with a number of neurons in the first layer matching the number of RGB sums (in analog values) or 6 times the number of RGB sums (in binary coding, where a RGB sum is converted into 6 bits, 1 bit per neural network input). Fig. 5 displays a 3-layer network where the 30 neurons in the input layer are drawn over 2 rows due to space limitations. The various network topologies varied the number of neurons in input layer, number of neurons in the hidden layer, number of hidden layer, and whether noise is added or not. The training and testing process was repeated for the various horizontal strips (with 10 or 25 horizontal strips), vertical strips (with 10 or 25 vertical strips), and block partitioning (with 16 or 25 blocks).

After trials and errors of the percentage of iris images used for training and for testing, we report in the subsequent section the results with 5 iris images used for training and 15 images used for testing as these portions generated the best iris recognition accuracy results by the feedforward artificial neural network.

6. Results and analysis

As there are three main data partitioning scenarios: horizontal strip (rows), vertical strip (columns), and blocks, and each scenario is repeated with different partition sizes, the accuracy of the iris recognition by the artificial neural network is expected to differ across scenarios. Table 1 shows the iris recognition accuracy under

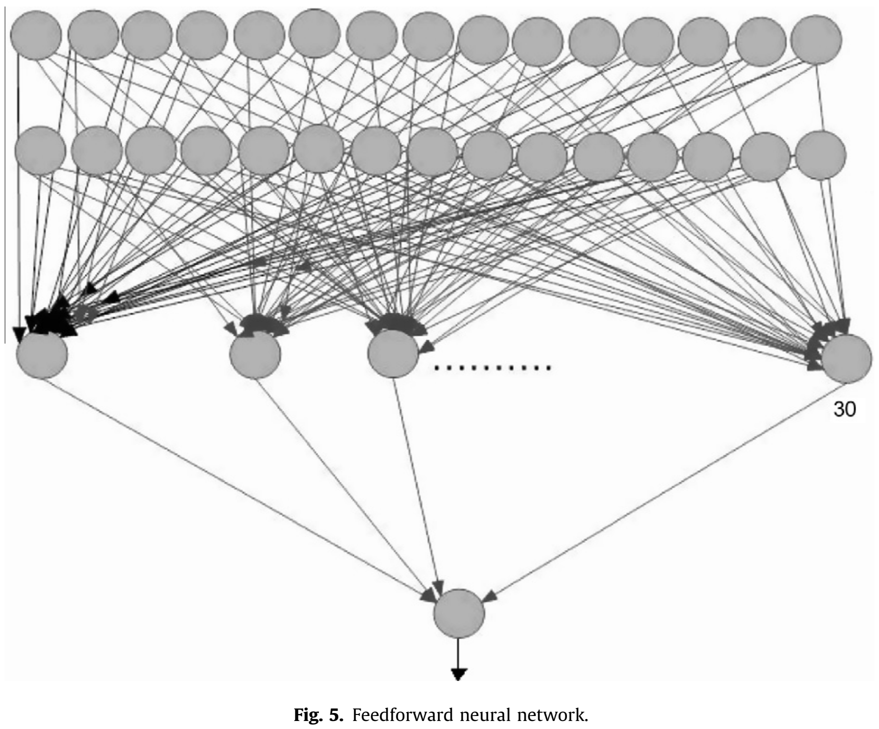

Fig. 5. Feedforward neural network.

Table 1
Experiments results with 10 horizontal or vertical strips, or 16 blocks.

Experiment description	Incorrect detection	Accuracy (%)
(Rows) Analog input: with one hidden layer (10 neurons)	5/15	66.67
(Columns) Analog input: with one hidden layer (10 neurons)	5/15	66.67
(Blocks) Analog input: with one hidden layer (10 neurons)	2/15	86.67
(Blocks) Analog input: with one hidden layer (50 neurons)	1/15	93.33
(Blocks) Analog input: with two hidden layers (10 neuron each)	2/15	86.67
(Blocks) Analog input: with two hidden layers (5 and 10 neurons)	2/15	86.67
(Rows) Binary input: with one hidden layer (50 neurons) and without noise	10/15	33.33
(Columns) Binary input: with one hidden layer (50 neurons)	10/15	33.33
(Blocks) Binary input: with one hidden layer (64 neurons)	10/15	33.33

Table 2
Experiments results with 25 horizontal or vertical strips or blocks.

Experiment description	Incorrect detection	Accuracy (%)
(Rows) Analog input: with one hidden layer (25 neurons)	6/15	60
(Columns) Analog input: with one hidden layer (25 neurons)	6/15	60
(Blocks) Analog input: with one hidden layer (25 neurons)	4/15	73.33
(Blocks) Analog input: with one hidden layer (25 neurons)	6/15	60
(Blocks) Analog input: with two hidden layers (25 neuron each)	7/15	53.33
(Blocks) Analog input: with two hidden layers (25 and 50 neurons)	7/15	53.33

each experiment when the images are divided into 10 horizontal strips, 10 vertical strips, or 16 blocks. Table 2 shows the accuracy results when the images are divided into 25 horizontal strips, 25 vertical strips, or 25 blocks. After comparing the results in the two tables, it is clear that the results obtained by dividing the images into fewer partitions are better than the results obtained after dividing the images into a large number of partitions. In the binary coding case, this can be explained as follows. As our version of the BrainMaker simulator limits the number of neurons per layer to 64, with 10 horizontal or vertical strips per image, each RGB sum for a horizontal or vertical strip can be coded into 6 bits while still meeting the 64 neurons per layer limit. Six bits are sufficient to represent the final sum (after division by 1 billion) which is an integer between 0 and 10. However when the number of horizontal or vertical strips is increased to 25, now the number of bits used to represent each RGB sum is 2 ($2 \times 25 = 50$ inputs < 64 input limit). With only 2 bits, an integer between 0 and 10 cannot be precisely represented and a large representation error is introduced. Therefore in the binary coding, the error is introduced when the final RGB sum is concatenated and its fraction is discarded, and also when not enough bits are available to represent the final sum. In the analog case, it is clear that with 10 horizontal or vertical strips or 16 blocks, fewer neurons are needed per network layer compared to the 25 strip or block case. In the former case, the total number of neurons in the network is $10 + 10 + 1 = 21$ ($16 + 16 + 1 = 33$ for blocks) assuming one hidden layer, while in the latter case, this number climbs to $25 + 25 + 1 = 51$ neurons. As each neuron in the input layer of the feedforward neural network interconnects to all neurons in the hidden layer, 10 partitions appear to better capture the features of the iris than the 25 partitions case where iris features may be diluted – and therefore lost – among the higher number of partitions of the latter case.

In Table 1, comparing the accuracy of horizontal strip partitioning versus vertical strip partitioning versus block partitioning, block partitioning is the clear winner. There is no accuracy advantage between horizontal and vertical partitioning as the experiments were conducted on square (100×100 pixel) images. So no extra iris feature information was present in either type of strip partitioning. However with block partitioning, the detailed iris features were better preserved than with either strip partitioning methods.

The best accuracy (93.33%) was obtained with 10 block partitioning with 10 neurons in the input layer and 50 neurons in the hidden layer and 1 hidden layer only. When the number of neurons in the hidden layer was reduced to 10 neurons to match the number of neurons in the input layer, the accuracy dropped to 86.66%. This accuracy result was obtained with 1 or 2 hidden layers of 10 neurons each, or with the first hidden layer containing 5 neurons and the second hidden layer containing 10 neurons. Also increasing the number of hidden layers from 1 to 2 reduced the accuracy (not shown in Tables 1 and 2). When binary coding is used and due to the above stated reasons (discarded fraction and insufficient number of bits), the accuracy drops significantly.

It is clear that the accuracy decreases when noise is added to the network as we experimented when we used 10 images for training and the other 10 for testing. When the number of neurons in the hidden layer is increased from 50 to 64, the accuracy stays stable/constant at 93.33% with one incorrectly detected image.

Our 93.33% accuracy result exceeds the results obtained by (Daugman, 1993) (54.44%), (Wildes, 1997) (86.49%), (Masek) (83.92%), (Liam et al., 2002) (64.64%), and are close to those obtained by Xu et al. (2008) (98.42%) despite our simple approach to iris image partitioning, our simple neural network type, and our testing with similar color and near-like irises to harden the recognition process.

Another advantage of our approach is that our neural network directly issues a match or no match output while in others' work, a neural network computes an iris code which must later be subjected to a Hammer distance computation to indicate a match.

7. Conclusion

Iris recognition is a trusted biometric technology used in secure places. The iris is generally protected from the environment and from aging. In addition, even twins have different irises. In this paper, we addressed the problem of iris recognition using a simple feedforward artificial neural network trained with the Backpropagation algorithm. We described a pre-processing method to prepare the neural network inputs from the iris images in jpeg format using an application for retrieving the RGB values from the jpeg image. We chose 20 100×100 pixel images of brown colored irises. After slightly trimming the top and bottom of the iris image, we retrieved the RGB values for all pixels in the image using that Java application, stored these RGB values in an Excel spreadsheet, and then blanked the areas of the spreadsheet corresponding to sclera and pupil areas of the image. We then considered three data partitioning techniques: horizontal strip, vertical strip, and blocks. These data partitions were then represented by the sum of the RGB values of the pixels falling in these areas. These pre-processing steps can be easily automated for repetitive experiments and operation. We prepared BrainMaker input files using NetMaker and then simulated various feedforward neural networks using BrainMaker. We varied the number of hidden layers, number of neurons per hidden layer, percentage of images used in training versus testing, and explored analog versus binary coding of the neural network input data. Results of simulation experiments revealed that 10 horizontal or vertical partitions (or 16 blocks) resulted in a better iris recognition accuracy than 25 horizontal or vertical partitions (or blocks). Block partitioning outperformed the horizontal or vertical strip partitioning methods. One hidden network layer was best. We were able to achieve 93.33% accuracy despite the fact that we used similar brown colored irises for training and testing. Using dissimilar irises of various colors is expected to enhance our artificial neural network's accuracy even further.

One idea for future work is to evaluate processing both left and right irises (rather than just one). With only one neural network, this serial process will however increase the time needed for person identification. If the recognition system is redesigned to simultaneously accept a pair of irises instead of just one by employing two neural networks, one for each eye, this will increase the system cost. Either way, further refinements of the proposed methodology presented may be possible without resorting to pairs of irises.

References

Bertillon, A. (1885). *La Couleur de l'Iris*. France: RevueScientifique

Bishop, C. (1995). *Neural networks for pattern recognition*. UK: Oxford University Press.

Boles, W., & Boashash, B. (1998). A human identification technique using images of the Iris and wavelet transform. *IEEE Transactions on Signal Processing, 46*(4), 1185–1188.

Broussard, R., Kennell, L., Ives, R., & Rakvic, R. (2008). An artificial neural network based matching metric for iris identification. In J. Astola, K. Egiazarian, & E. Dougherty (Eds.), *Image processing: Algorithms and systems VI. Proceedings of SPIE* (Vol. 6812, pp. 68120S–68120S-11).

Chen, C., & Chu, C. (2009). High performance iris recognition based on 1-D circular feature extraction and PSO–PNN classifier. *Expert Systems with Applications, 36*(7), 10351–10356.

Chek Image Database. Iris images retrieved from <http://www.phoenix.inf.upol.cz/iris/download/>.

Delac, K., & Grgic, M. (2004). A survey of biometric recognition methods. In *46th International symposium electronics in marine, ELMAR-2004, Zadar, Croatia*.

Daugman, J. (1993). High confidence visual recognition of persons by a test of statistical independence. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 15*(1), 1148–1161.

Daugman, J. (2002). How iris recognition works. In *Proceedings of international conference on image processing* (Vol. 1).

Fausett, L. (1994). *Fundamentals of neural networks*. New York: Prentice Hall.

Hassanien, A. E., Abraham, A., & Grosan, C. (2009). Spiking neural network and wavelets for hiding iris data in digital images. *Soft Computing* **13**(4), 401–416.

Haykin, S. (1998). *Neural networks: A comprehensive foundation* (2nd ed.). NY: Prentice Hall.

IEEE Expert Staff (1992). Brainmaker professional. *IEEE Expert: Intelligent Systems and Their Applications, 7*(2), 70–71.

JPEG Specifications. <http://www.jpeg.org/committee.html>.

Jain, A., Ross, A., & Prabhakar, S. (2004). An introduction to biometric recognition. *IEEE Transactions on Circuits and Systems for Video Technology, Special Issue on Image- and Video-Based Biometrics, 14*(1).

Liam, L. et al. (2002). Iris recognition using self-organizing neural network. *Student Conference on Research and Development*, 169–172.

Lim, S., Lee, K., Byeon, O., & Kim, T. (2001). Efficient Iris recognition through improvement of feature vector and classifier. *ETRI Journal, 23*(2), 1–2.

Lin, E. (xxxx). Program for retrieving the RGB values of a jpeg image's pixels. <http://www.ericlin2.tripod.com/getrgb/getrgbt.html>.

Ma, L., Wang, Y., & Tan, T. (2002). Iris recognition using circular symmetric filters. In *Proceedings of 16th international conference pattern recognition* (Vol. II, pp. 414–417).

Masek, L. (xxxx). Recognition of Human Iris Patterns for Biometric Identification. <http://www.csse.uwa.edu.au/opk/studentprojects/labor>.

Murakami, M., Takano, H., & Nakamura, K. (2003). Real-time Iris recognition by a rotation spreading neural network. In *Annual SICE conference* (Vol. 1, pp. 283–289).

Poursaberi, A., & Araabi, B. (2005). A novel iris recognition system using morphological edge detector and wavelet phase features. *ICGST-GVIP Journal, 5*(6).

Rumelhart, D., & McClelland, J. (1996). *Parallel distributed processing: Explorations in the microstructure of cognition*. Cambridge: MIT Press.

Wildes, R. et al. (1994). A system for automated iris recognition. In *Proceedings of 2nd IEEE workshop on applications of computer vision* (pp. 121–128).

Wildes, R. (1997). Iris recognition: An emerging biometric technology. *Proceedings of IEEE, 85*(9), 1348–1363.

Xu, G., Zhang, Z., & Ma, Y. (2008). An efficient Iris recognition system based on intersecting cortical model neural network neural network. *International Journal of Cognitive Informatics and Natural Intelligence, 2*(3).