Journal of Computer Science 9 (11): 1435-1442, 2013
ISSN: 1549-3636

doi:10.3844/jcssp.2013.1435.1442 Published Online 9 (11) 2013 (http://www.thescipub.com/jcs.toc)

GENETIC ALGORITHM AND NEURAL NETWORK FOR OPTICAL CHARACTER RECOGNITION

Hendy Yeremia, Niko Adrianus Yuwono, Pius Raymond and Widodo Budiharto

Department of IT, School of Computer Science, Bina Nusantara University, Jakarta-Indonesia

Received 2013-01-15, Revised 2013-04-16; Accepted 2013-09-19

ABSTRACT

Computer system has been able to recognize writing as human brain does. The method mostly used for character recognition is the backpropagation network. Backpropagation network has been known for its accuracy because it allows itself to learn and improving itself thus it can achieve higher accuracy. On the other hand, backpropagation was less to be used because of its time length needed to train the network to achieve the best result possible. In this study, backpropagation network algorithm is combined with genetic algorithm to achieve both accuracy and training swiftness for recognizing alphabets. Genetic algorithm is used to define the best initial values for the network's architecture and synapses' weight thus within a shorter period of time, the network could achieve the best accuracy. The optimized backpropagation network has better accuracy and less training time than the standard backpropagation network. The accuracy in recognizing character differ by 10, 77%, with a success rate of 90, 77% for the optimized backpropagation and 80% accuracy for the standard backpropagation network. The training time needed for backpropagation learning phase improved significantly from 03 h, 14 min and 40 sec, a standard backpropagation training time, to 02 h 18 min and 1 sec for the optimized backpropagation network.

Keywords: Backpropagation Network, Genetic Algorithm, Optical Character Recognition, Optimized Artificial Neural Network

1. INTRODUCTION

Human brain consists of 10^{11} sets of interconnected neurons to facilitate our reading, breathing, motion and thinking. In term of learning, human brain is superior to a microprocessor. Because of that fact, backpropagation network tries to adapt the ability of human brain to learn by experience (Pinjare and Kumar, 2012).

Backropagation is probably the most common method for training forward-feed neural networks. A forward pass using an input pattern propagates through the network and produces an actual output. The backward pass uses the desired outputs corresponding to the input pattern and updates the weights according to the error signal. There are hundreds of papers covering the subject of backward propagation. Unfortunately, many of them tend to exhibit a vast stockpile of equations and complicated partial derivatives with undefined variables to explain a concept that is really quite simple. Quite often, a pseudocode algorithm or an example with pictures is the most effiecient method to convey information.The most popular method used in optical character recognizing is nevertheless backpropagation network. This method weakness is the required time to achieve the best result for recognizing alphabets tends to be long. Backpropagation itself could do the preprocessing phase for alphabet recognition less complex than genetic algorithm (Negnevitsky, 2005).

Genetic algorithm would be used to optimize what a standard backpropagation network lacks, architecture and initial weights. This algorithm is often used to find an optimal solution in complex problems (Matic, 2010) by

Corresponding Author: Hendy Yeremia, Department of IT, School of Computer Science, Bina Nusantara University, Jakarta-Indonesia

adapting the law of natural selection and natural genetics called survival of the fittest (Malhotra *et al*., 2011).

To achieve both better accuracy and less training time needed, genetic algorithm is being used to optimize the backpropagation network. Genetic algorithm focused on exploring the best architecture possibilities and the best weight initial values to be then inputted for the backpropagation network's structure.

efficient and quick learning, the weight optimizationof Back Propagation neural network has been carried outusing Genetic Algorithm. Genetic Algorithms (GA) areadaptive search and optimization techniques, mimicking theprinciples of natural evolution. Genetic Algorithms havebeen proposed as one of the potential candidates foroptimization of weight parameters of neural network.Conventionally, Standard Back Propagation network performing gradient descent learning algorithms haveencountered difficulties of getting struck in local minimaproblem. Whereas Genetic Algorithm does not guarantee aglobal minimum solution, however it can locate theneighborhood of optimum solution much quicker thanconventional strategies and provide encouraging results (Patra *et al*., 2012).

This paper presents a new approach in optimizing backpropagation neural network training for optical character recognition. Genetic algorithm will be used to determine which architecture to be used and to define the initial weights for the network.

1.1. Genetic Algorithm

Genetic algorithm is an algorithm for optimization and machine learning based loosely on several features of biological evolution. It can reduce bad components in programming and replace it with a better one. These components are called genes, as in biology, in genetic algorithm. Genetic algorithms are a class of parallel adaptive search algorithms based on the mechanics of natural selection and natural genetic system. It can find the near global optimal solution in a large solution space quickly. It has been used extensively in many application areas, such as image processing, pattern recognition, feature selection and machine learning (Majida *et al*., 2010).

On a population of chromosomes (individuals) in genetic algorithm, each chromosome has its own genes. From that population, a selection process will occur to find a chromosome with the best fitness/survival rate to be crossover-ed to produce a new chromosome with better fitness. The new chromosome will then replace the chromosome with the lowest survival rate. This process will be iterated until the desired error rate is achieved.

2. MATERIALS AND METHODS

Artificial neural network is an information processing system that resembles human biological neural system. This system processes information after the synapses' weight has been adjusted. The adjustment process is based from the previously collected training data. In the other word, neural network learns from examples and if it is trained well, neural networks will be able to produce a good result for new problems that does not exist in the training data. An artificial neural network consists of a collection of processing elements that are highly interconnected and transform a set of inputs to a set of desired outputs. The result of the transformation is determined by the characteristics of the elements and the weights associated with the interconnections among them. By modifying the connections between the nodes the network is able to adapt to the desired outputs (Gregoire, 2012):

- Knowledge is gained from learning processes
- The inter-neuron strength or synapses are used to store the knowledge

2.1. Neural Network Architecture

Based on its architecture, neural network model is separated into three types of network.

2.2. Single Layer Network

In this network, input neurons are directly connected to the outputs. Signal flows in one direction, from input layer to output layer **Fig. 1**.

This model also includes: ADALINE, Hopfield, Perceptron and LVQ.

2.3. Multi Layer Network

This is a developed single layer network. In this network model, besides the input and output layers, another type of layer exists. It is often called the hidden layer. A multiple layer network could have multiple hidden layers in it. This model includes: Backpropagation Network and MADALINE **Fig. 2**.

2.4. Recurrent Network

This network model is similar to both single layer network and multi layer network model. It is only differentiated by output nodes that signaled the input nodes, or usually called feedback loop. In this model, signals go two ways, forward and backward. This model includes: Hopfield Net, Jordan Network and Elmal Network.

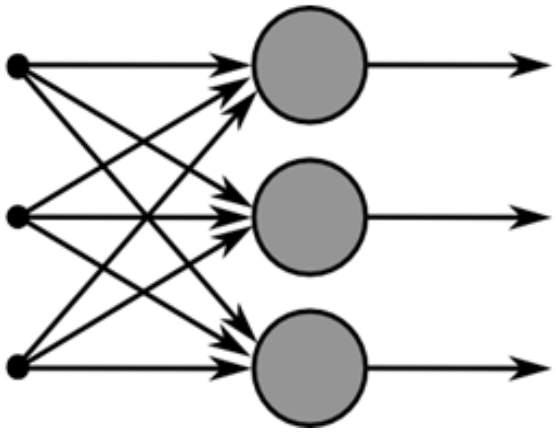

Fig. 1. Single layer network

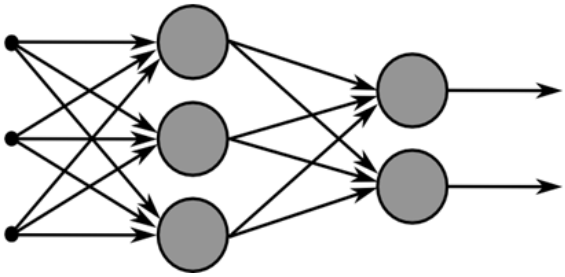

Fig. 2. Multi layer network

2.5. Activation Function

The most important factor in neural network is the activation function or threshold function or transfer function. Activation function is used to limit the neuron output. Two most common activation functions are: sigmoid binary function and sigmoid bipolar function (Karlik and Olgac, 2010).

2.6. Sigmoid Binary Function

This function is useful in neural network with backpropagation training model because it is easy to distinguish and reduce the capacity needed **Fig. 3** Equation (1):

$$g(x) = \frac{1}{1 + e^{-x}} \quad (1)$$

2.7. Sigmoid Binary Function

This activation function is similar to sigmoid binary function. This function works well for application that produces value between interval of [-1; 1] as shown in **Fig. 4**.

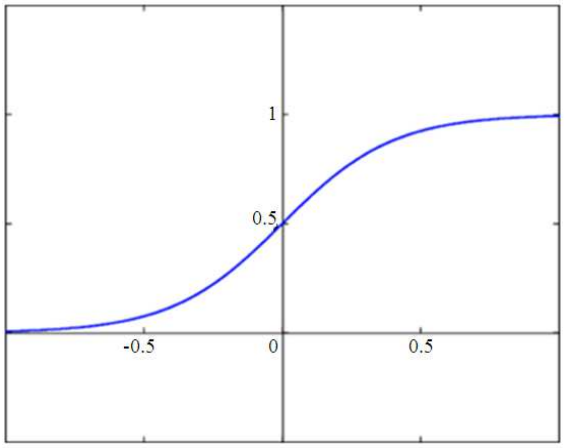

Fig. 3. Sigmoid binary function graph

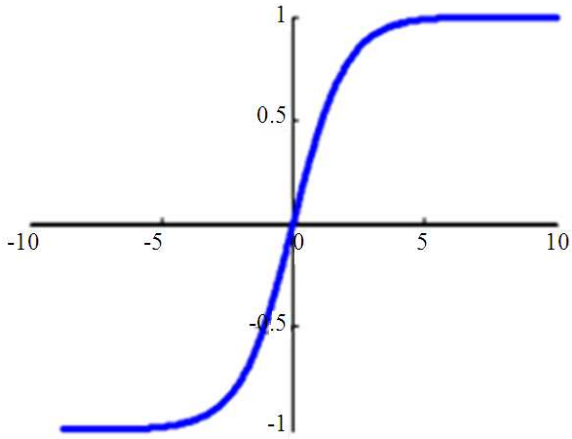

Fig. 4. Sigmoid bipolar function graph

Figure 5 show the model of backpropagation network that consist of input values, input layer, hidden layer, output layer and output values.

2.8. Backpropagation Network

Backpropagation network is a multi layer neural network. Backpropagation is more like the learning/training algorithm rather than the network itself.

Backpropagation use supervised training algorithm for multi layer network, therefore the input and target output has been prepared for the training process. A data error in output layer is counted using network output and target output. The data errors then back propagated to the hidden layer, resulting in weight change for the synapses heading to the hidden layer (Pinjare and Kumar, 2012).

Basically, backpropagation network consists of two phases, feedforward phase and backward phase.

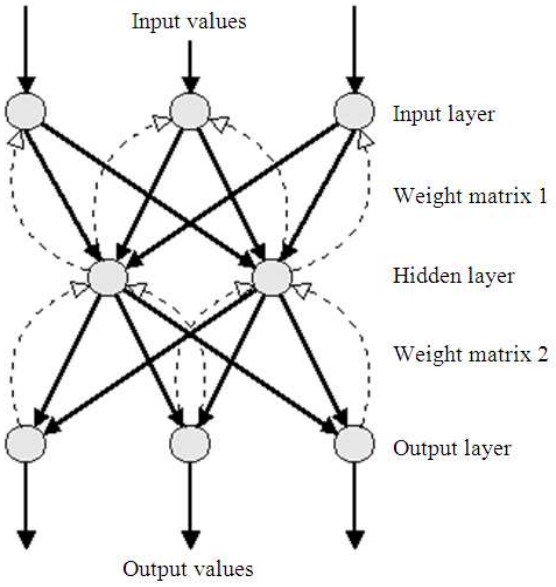

Fig. 5. Backpropagation network

2.9. Feed forward Phase

In this phase, there are no cycles. Information only goes one way, from the input layer to the hidden layer (if there is any) and then to the output layer.

According to Sutojo *et al*. (2011), the algorithm of feed forward phase is:

- Define network architecture, consist of input layer (x_i; i = 1, 2, 3... n), hidden layer (z_j; j = 1, 2, 3, ..., p) and output layer (y_k; k = 1, 2, 3, ..., m)
- Initialized weight value from input to hidden layer (v_{ij}; i = 0, 1, 2, 3, ..., n; j = 1, 2, 3, ..., p) and hidden to output layer (w_{jk}; j = 0, 1, 2, 3, ..., p; k = 1, 2, 3, ..., m) with a relatively small random value, such as interval -0.5 to 0.5
- Each node in input layer (x_i; i = 1, 2, 3, ..., n) receives x_i signals and forwarding the signals to every nodes in hidden layer
- Each node in hidden layer (z_j; j = 1, 2, 3, ..., p) sum up the input nodes' weights Equation (2):

$$z\_in_j = v_{oj} \sum_{i=1}^{n} x_i v_{ij} \quad (2)$$

and use the activation function to measure the output signal Equation (3):

$$z_j = f(z - in_j) \quad (3)$$

and forwarding those output signals to every node to output layer.

Each node in output layer (y_k; k = 1, 2, 3, ..., m) sum up the input nodes' weights Equation (4):

$$y\_in_k = w_{0k} + \sum_{j=1}^{p} z_j w_{jk} \quad (4)$$

and use the activation function to measure the output signal Equation (5):

$$y_k = f(y\_in_k) \quad (5)$$

2.10. Backward Phase

In the backward phase, weight readjusments are to be done. Backward phase algorithm (Rahajaan, 2011).

Calculate factor δ output unit based on errors in every output unit y_k (k = 1, 2, 3, …, m) Equation (6):

$$\delta_k = (t_k - y_k) f'(y_{ink}) = (t_k - y_k)\, y_k (1 - y_k) \quad (6)$$

Where:

t_k = Target output

δ_k = Error unit that will be used for readjustment on the lower layer:

Calculate W_{ki} weight adjustment with learning rate α Equation (7):

$$\Delta w_{kj} = a\delta_k z_j \; k = 1, 2\ldots m; \; j = 0, 1\ldots p \quad (7)$$

Calculate factor δ output unit based on errors in every hidden unit z_j (j = 1, 2, …, p) Equation (8):

$$\delta_j = \delta\_in_j f'(z\_in_j) = \delta\_in_j z_j (1 - z_j) \quad (8)$$

Calculate weight adjustment value for v_{ji} Equation (9):

$$\Delta v_{ji} = a\delta_j x_i \; j = 1, 2,\ldots p; \; i = 1, 2\ldots n \quad (9)$$

Calculate every weight readjustments Equation (10 and 11):

$$w_{kj}(new) = w_{kj}(old) + \Delta w_{kj}$$
$$k = 1, 2, \ldots, m; j = 0, 1, \ldots, p \quad (10)$$

$$v_{ji}(new) = v_{ji}(old) + \Delta v_{ji}$$
$$j = 1, 2, \ldots, p; i = 1,2, \ldots, n \quad (11)$$

Parameter α is learning rate that controlled the speed of iterations. Its value is between 0 and 1. The bigger its value is, the lesser iterations will occur. But, if α is assigned with a value too big, it could ruin the previously correct pattern and slow down the learning process.

2.11. Genetic Algorithm Optimized Neural Network

Genetic algorithm runs twice in this system, first is to determine the network architecture and sec, to determine the weight for the network synapses. Basic processes in genetic algorithm, such as population initialization, fitness calculation, selection, crossover and mutation, executed as much as the number of desired generations. In this optical character recognition system, the number of generations has been set to 10 before hand to find the optimal architecture and optimal weight. Each generation consist of five training data sets. Each set consists of 26 alphabetical characters, from 'A' to 'Z'.

Population Initialization is the first step in genetic algorithm process. This starts with initializing random numbers as much as the multiplication of chromosome numbers and chromosome base for each chromosome. For the architecture, each chromosome consists of 9 chromosome bases and there are 50 chromosomes as the initial population. Base for the architecture is a binary number, 0 or 1.

Each chromosome represents the number of nodes in the neural network's hidden layer. With the length of 9 bases per chromosome, the architecture of hidden layer could consist of 270 to 405 nodes. Outside those boundaries, there will be a chromosome re-initialization based on Panchal *et al*. (2011) that optimum nodes in hidden layer range from 2/3 to 3/3 of the sum of input and output nodes.

Fitness calculation needs to be done before we enter the selection phase. Fitness is a comparison value to determine which chromosome should be eliminated and replaced.

Selection concept in this system is a Tournament selection. k random candidates will be selected (in this system, k = 8) and paired with each other, leaving 4 pairs. Then a tournament selection will be executed and candidate with higher fitness value will survive. We do this until it is only a single candidate left with the biggest fitness value (Sivaraj and Ravichandran, 2011). This is held to find 10% or 5 pairs of parent to be crossover-ed.

Crossover will be used to create new chromosomes from the 5 pairs of parent. Every crossover will have a random crossover point according to the chromosome length. If the new chromosomes' fitness value is bigger than the previous ones, the worse chromosomes will be replaced by the new chromosomes.

Mutation is executed on the new chromosomes (offspring). Mutation rate is decided by calculating the offspring fitness value first. A random number of 0 and 1 will be used to decide whether the offspring will be mutated or not.

Pseudocode for the system:

```
BEGIN
Initialize Population
Loop for i generation
Read training data set
Feed forward
Count fitness for each individual
Selection
Crossover
Mutation
If child's fitness greater than the worst individual's fitness
    Replace the individual with child
End If
End Loop
END
```

3. RESULTS

The implementation of alphabetical character recognition as seen in **Fig. 6**. Is done by firstly taking the character image through a web-camera. Then the captured image will be preprocessed and be gray-scaled. To achieve a better recognition, a bounding box needs to be implemented to get the exact size of the character. After the exact size obtained, the no-white-space image will be resized to 20×20 pixels resolution. This resolution decrease the training time needed quite significantly because the smaller the resolution is, less nodes are there in the neural networks, which means less synapses and less weight to be readjusted.

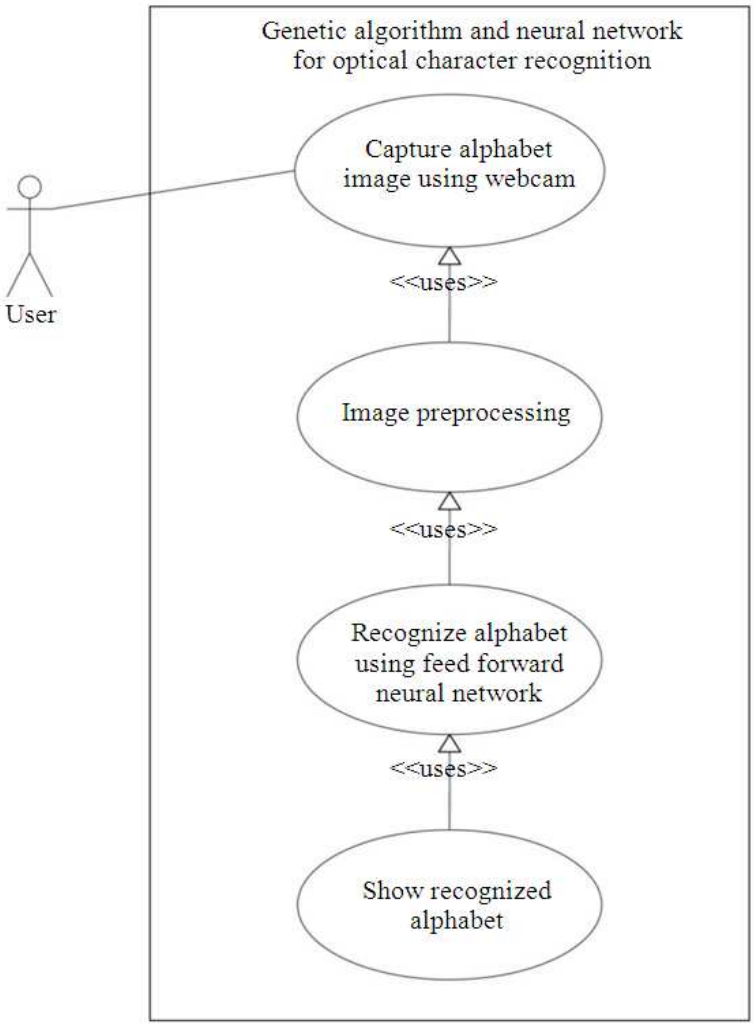

Fig. 6. Use-case diagram of application system

4. DISCUSSION

Experiments have been performed to test the proposed method. Microsoft Visual C++2010 Express is the software tool that were used for alphabetical character recognition. Recognized image will be given a bounding box to assure that the features and white-spaces around the character are similar. The experiments were performed on 10 sets of alphabetical characters.

Table 1 Gives the results of recognition accuracy by standard Backpropagation Network (BPN) and by optimized Genetic Algorithm Neural Network (GABPN) performed on 20×20 pixels alphabetical characters.

The backpropagation optimized with genetic algorithm showed significant improvement, especially for time and accuracy. By using genetic algorithm, the best architecture for neural network is obtained. That architecture is implemented in both standard backpropagation and genetic algorithm backpropagation to compare each other whether if genetic algorithm could really make improvements on the standard backpropagation.

After spending 3 h, 14 min and 40 sec, the standard backpropagation has reached its optimum solution for recognizing the characters. As the **Table 1** above shows us, genetic algorithm optimized neural network still recognize better. Also, it requires much lesser time than the standard backpropagation needs to achieve the best solution. Genetic algorithm researched for the best architecture to be used and weights to be initialized for the neural network.

Table 1. Recognition accuracy comparison

Alphabetical characte	Accuracy	
	GABPN(%)	BPN(%)
A	100	100
B	100	100
C	100	80
D	100	80
E	80	60
F	80	60
G	100	100
H	100	60
I	80	100
J	100	40
K	80	80
L	100	100
M	60	60
N	60	80
O	100	80
P	100	100
Q	100	80
R	100	100
S	100	80
T	80	80
U	100	40
V	80	80
W	80	80
X	80	80
Y	100	80
Z	100	100

Finding the architecture spent 39 min and 52 sec, finding the best weight spent 39 min and 55 sec. Backpropagation training using the optimized architecture and optimized weight itself runs for 58 min and 14 sec before the desired error is achieved. In total, the genetic algorithm optimized neural networks spent 2 h, 18 min and 1 sec to finish all the jobs where the standard neural networks needs 1 hour more to achieve the same result.

Figure 7 show the method to capture character in front of webcam.

Besides it's swiftness, the accuracy of recognition alsi improves. With a shorter amount of time for training, genetic algorithm optimized neural networks got more accurate recognition than the standard backpropagation.

Overall, GABPN is also superior from BP in term of accuracy. From the total of 10 character sets used to test the system result, GABPN has an accuracy rate of 90.77% while BP's 80%.

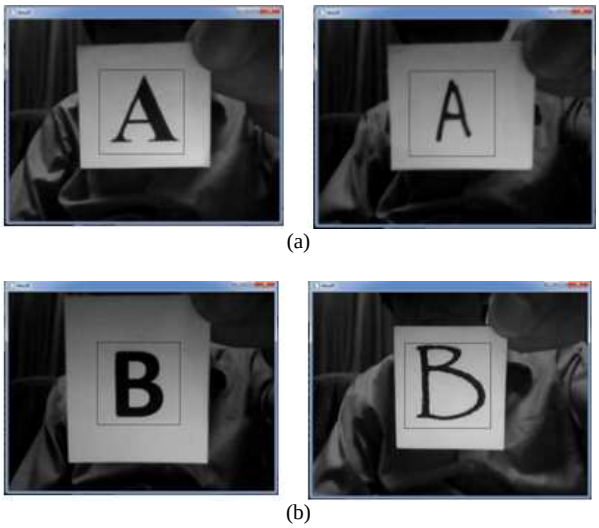

Fig. 7. Webcam capture the character A (a) and B (b)

5. CONCLUSION

The backpopagation is studied and successfully optimized with genetic algorithm. Optical character recognition application can be built with combination of genetic algorithm and neural network to achieve more accurate application and shorter learning time. This can help any similar recognizing application to accurately recognize with less error.

For further development, in developing the genetic algorithm capabilities toward its potentiality, we suggest to try the Lamarckian theory in genetic algorithm mutation on deciding the best optimum weight to achieve better survival rate to even further reducing the time needed for training the neural network. Another future scope are to develop an application that can recognize a whole word at a time, or even a whole sentence at a time, also to be able to recognize characters outside of alphabets.

6. REFERENCES

Gregoire, B.O., 2012. Neural Networks: Tricks of the Trade. 2nd Edn., Springer London, Limited, ISBN-10: 364235288X, pp: 781.

Karlik, B. and A.V. Olgac, 2010. Performance analysis of various activation functions in generalized MLP architectures of neural networks. Int. J. Artif. Intell. Exp. Syst., 1: 111-122.

Majida A., A.N. Ismail and Z.M. Hazi, 2010. Pattern recognition using genetic algorithm. Int. J. Comput. Electr. Eng., 2: 1793-8163.

Malhotra, R., N. Singh and Y. Singh, 2011. Genetic algorithms: Concepts, design for optimization of process controllers. Comput. Inform. Sci., 4: 39-54.

Matic, D., 2010. A genetic algorithm for composing music. Yugoslav J. Operat. Res., 20: 157-177. DOI: 10.2298/YJOR1001157M

Negnevitsky, M., 2005 Artificial Intelligence: A Guide to Intelligent Systems. 1st Edn., Addison-Wesley, New York, ISBN-10: 0321204662, pp: 415.

Panchal, G., A. Ganatra, Y.P. Kosta and D. Panchal, 2011. Behaviour analysis of multilayer perceptrons with multiple hidden. Int. J. Comput. Theory Eng., 3: 332-337.

Patra, S.R., R. Jehadeesan, S. Rajeswari, S.A.V.S. Murty and M.S. Baba, 2012. Development of genetic algorithm based neural network model for parameter estimation of fast breeder reactor subsystem. Int. J. Soft Comput. Eng., 2: 87-90.

Pinjare, S.L. and M.A. Kumar, 2012. Implementation of neural network back propagation training algorithm on FPGA. Int. J. Comput. Applic., 52: 1-1.

Rahajaan, J.A., 2011. Pengembangan wahana pencitraan bawah air guna identifikasi dan kuantifikasi terumbu karang dengan metode jaringan syaraf tiruan. MSc Thesis, Bogor Agriculture University.

Sivaraj, R. and T. Ravichandran, 2011. A review of selection methods in genetic algorithm. Int. J. Eng. Sci. Tech., 3: 3792-3797.

Sutojo, T., E. Mulyanto, and V. Suhartono, 2011. Kecerdasan buatan. ANDI, Yogyakarta.