Integrating Neural Network With Genetic Algorithms For The Classification Plant Disease

Dr.Alia Karim Abdul Hassan* & Sarah Sadoon Jasim*

Received on: 22/6/2008
Accepted on:7/5/2009

Abstract

In this work Artificial Neural Network (ANN) is used as a classifier capable of recognizing the most important features of the plant disease, with minimum error value. Genetic algorithm has been used to minimize error values of the ANN classifier. Error value of ANN classifier is defined as more than (%5). This ratio is a threshold (cut-of-value) to determine if GA is executed or not after the ANN classifier execution. Genetic algorithm execution results in either optimal solution (%100) recognition or suggests a modified parameter to the ANN classifier (specifically learning rate and number of neurons).The result obtained from integrating neural network with genetic algorithm for classification plant diseases indicates that the classifier recognizes most of input pattern with accuracy (96%). Integrating neural network with genetic algorithm for classification plant diseases implemented using Visual Basic version 6 programming.

Keywords: Artificial Neural Network, Genetic algorithm, Genetic algorithm

تكامل الشبكات العصبية مع الخوارزميات الجينية لتصنيف امراض النبات

الخلاصة

في هذا العمل استخدمت الشبكات العصبية الاصطناعية (ANN classifier) كمصنف قادر على تمييز اغلب الملامح المهمة في امراض النبات مع اقل قيمة خطأ . فأستخدمت الخوارزميات الجينية لتقليل قيم الخطأ للشبكات العصبية. قيمة الخطأ المعرفة في هذا المصنف هي اكثر من 5% بذلك سيتم معالجتها في الخوارزميات الجينية. فهذه القيمة تعتبر(threshold) لتحديد فيما اذا الخوارزميات الجينية سيتم تنفيذها او لا بعد تنفيذ الشبكات العصبية المقترحة. ان نتائج تنفيذ الخوارزميات الجينية هي أما حل امثل أي تمييز (100%) او ستقترح تعديل المعاملات المستخدمة في الشبكات العصبية الاصطناعية المقترحة خصوصا معامل التعلم (Learning rate) وعدد الخلايا العصبية في الطبقة المخفية (No. of neurons in the hidden layer). في هذا البحث تم استخدام الشبكات متعددة الطبقات (Mlti-Layer-Percetron) التي هي نوع من الشبكات العصبية مع خوارزمية الانتشار الخلفي للخطأ (Back Propagation algorithm) لتدريب الشبكة لتصنيف أمراض النبات. النتيجة المحصلة من تكامل الشبكات العصبية مع الخوارزميات الجينية لتصنيف أمراض النبات يشير إلى أن الشبكات العصبية المقترحة قد ميزت معظم رموز الأمراض مع دقة (96%).

* Computer Science Department, University of Technology / Baghdad

https://doi.org/10.30684/etj.28.4.5
2412-0758/University of Technology-Iraq, Baghdad, Iraq

Introduction

The theory of neural network is the methodological technique used in this study Neural networks derive their name from the manner in which they simulate the brain activity at a very elementary level of operations. Neural networks have become a popular tool for solving complex problems in diverse domains over the past ten years. In principle, neural networks can solve any function, and they can do anything that digital computer can do [1]. Neural networks are simple, confidence can be attached to the results of neural networks , and can be implemented in software or hardware, and trained. A variety of neural networks architectures are currently in vogue, but still the Multi-Layered Perceptron (MLP) trained under back-propagation algorithm is the most widely spread neural [2].

Effective disease control depends primarily on early accurate identification and the casual agents, in most cases, it's too late to control a disease However, and timely control measures can prevent the disease spreading to other plants. In agriculture mass production, it is needed to discover the beginning of plant diseases batches early to be ready for appropriate timing control to reduce the damage, production costs and, increase the income. So it needs a classifier capable of recognizing the diseases[3,4].

GAs is general purpose search algorithms which use principles inspired by natural genetics to evolve solutions to problems. The basic idea is to maintain a population of chromosomes, which represents candidate solutions to the concrete problem being solved that evolves over time through a process of competition and controlled variation. Each chromosome in the population has an associated fitness to determine (select) which chromosomes are used to form new ones in the competition process. The new ones are created using genetic operators such as crossover and mutation. GAs has had a great measure of success in search and optimization problems. The reason for a great part of this success is their ability to exploit the information accumulated [5].

In this work ANN is used as a classifier capable of recognizing the most important features of the plant disease, with minimum error value. GA has been used to minimize error values of the ANN classifier.

Artificial Neural Networks

(ANNs) models have been studied for many years in the hope of achieving human like performance. Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyze. This expert can then be used to provide projections given new situations of interest and answer "what if" questions [6].

The Perceptron and Multi Layer Perceptrons (MLPs)[7]

The perceptron is the simplest form of a neural network, since a single layer perceptron operates as a single neuron, multilayer perceptron are an expansion of the perceptron idea and can be used to solve much more difficult problems. They consist of an input layer, one or more hidden layers of an output layer. The hidden layers give the network its power and allow for it to extract extra features from the input. One of the most popular methods used in training a multilayer perceptron is the (error) *back-propagation algorithm,* which includes two passes through each layer of the network: a forward pass and a backward pass.

A Simple Genetic Algorithm (SGA)[8,9,10]

Genetic algorithms were developed by John Holland in 1960s. GAs are procedures for solving problems by using principles inspired by natural population genetics. GAs use procedures to maintain a population of knowledge structures that represent candidate solutions, and then let that population evolve over time through competition (survival of the fittest) and controlled variation (recombination and mutation). Given a clearly defined problem to be solved and a bit string representation for candidate solutions, a simple GA works as follows:

Step 1: Start with a randomly generated population of (n_bit) chromosomes (candidate solutions to problem), such that (n=9).

Step 2: Calculate the fitness (f(x)) of each chromosome (x) in the population.

Step 3: Repeat the following steps until (n) off springs have been created:

a) Select a pair of parent chromosomes from the current population, the probability of selection being an increasing function of fitness. Selection is done "with replacement", meaning that the same chromosome can be selected more than once to become a parent.

b) With probability Pc (the "crossover probability" or "crossover rate"), crossover the pair at a randomly chosen point (chosen with uniform probability) to from two offspring .If no crossover takes place, from two offspring that are exact copies of their respective parents. (Note that here the crossover rate is defined to be the probability that two parents will crossover in a single point. There are also "multi-point crossover" versions of the GA in which the crossover rate for a pair of parents is the number of points at which a crossover takes place).

c) Mutate the two offspring at each locus with probability Pm (the mutation probability or mutation rate), and place the resulting chromosomes in the new population.

Step 4: Replace the current population with the new population.

Step 5: Go to step 2.

The Proposed Classifier

The ANN Classifier

A MLP is used to design a network to classify plant disease. An ANN trained under back propagation algorithm will be constructed for performing classification for plant disease. Table (1) illustrates pattern code for 12 plant disease which is used in this research. Each code is constructed from 9 bit each bit represent a specified disease symposium so a data base of 2^9 (512) disease can be recognized.

MLP Classifier Structure

In this work a data base of 512 elements will be defined so the proposed ANN will have nine neurons. Perceptron of three layers had been designed to pattern recognition process. One hidden layer is used in learning process. The number of hidden layers is one for decreasing the complexity of artificial neural network from time and space, on the other side fully connected neural network is defined and the number of neurons of input layer is defined according to the size of input pattern to be recognized and the number of neurons of the out put layer is defined according to the size of a class related to each group of patterns. Fig (1) shows the proposed MLP neural network for classification of plant diseases neural network structure is: Input layer consists of 9 neuron, Hidden layer consists of 7 neuron, Output layer consists of 3 neuron. The number of neurons of any layer can affect the number of connections to the related layer. As the number of connection increases, this can be time and cost consuming, because the process of each connection needs one multiplication process (w_{ij} * input i), the weight matrix (input layer) = 9*7 = 63 connection. If the number of hidden layer increase (for example = 8) then the weights matrix w= 9*8 = 72 (full connection). For example, if this process takes five millisecond then for 72 connections needs (5*72) mill second (this for time) and the same criteria can be applied to space. This base can be applied for all layers then as result.

.Activation Function

Activation function used in the MLP model proposed is called sigmoid function .

. Training the Network

The network training process is the most important part of the system. It takes all the feature vectors of the pattern and trains the network on them by using the back propagation training algorithm. This algorithm is suitable for all the types of recognition and gives a good result. This algorithm is used for learning process, since the error can be propagating from the output layer to the previous layer, this error can help the neural network to learn a pattern and recognize it later. The error is decreased, this means that the learn network success to learn that pattern and can recognize it later or can not be recognized. The error values are related to weights modification process. The ANN is designed for this purpose; the recognition process is executed with an error resulting from recognition process. In this work

the proposed ANN is initially trained with a set of basic pattern codes which are represented with (0 and 1). The first neuron in the hidden layer which takes a sum of all its inputs multiplied by the corresponding weights, i.e. for the proposed ANN training process will be described as follows according to Equation (1):

$$Sum = input1 \times w1 + input2 \times w2 + \ldots.. + input_n \quad \ldots\ldots (1)$$

When all the outputs have been calculated for the hidden layer the outputs from the hidden layer are used as the inputs for the output layer. Altering the weights in a MLP, the weights to the output layer are changed first followed by the weights to the hidden layer from the input layer. This is known as the back propagation training rule. The weights are changed using the following formula.

$$Weight_n = weight_n + (gain \times error \times output_n) \quad \ldots.. (2)$$

where n is the current neuron in the network, weight n is the going out of the current neuron, gain is a number between 0 and 1 that determines how fast the network will converge (i.e., learning rate is 0.2 which is determined by practice. Since if less than this value will slower ANN work may be entered to a local minimum, learning rate value more than 0.2 may speedup ANN work but may lose some important solutions) , output n is the output of the current neuron (calculated using the sigmoid function).

Calculating error is different for the output layer than it is for the hidden layers. For the output layer error is calculated using equation(3), where the subscripts define the current layer and target is the value the output should be.

$$Error_{out} = spread \times output_{out} \times (1 - output_{out}) \times (target_{out} - output_{out}) \ldots. (3)$$

Once error output has been calculated, the delta for the next layer up can be calculated. In the hidden layer error hidden is calculated by equation (4).

$$Error_{hidden} = spread \times output_{hidden} \times (1 - output_{hidden}) \times (target_{below} - output_{above}) \quad \ldots. (4)$$

When the network is being set up every neuron is given a small random value called weight and is represented by arrow. Using a set of random weights is obviously not going to produce the correct output straight away so they have to be changed somehow if the output is correct the error is changed a little but if the output is incorrect the error is changed a lot. If this is done enough times the network will have learnt what the operator wants it to do when altering the weights in a MLP, the weight of output layer is changed first followed by the weights of the hidden layer. As a result of the training process the network requirements consist of changing threshold = 0.5, learning rate = 0.2, number of inputs = 9, number of outputs = 3, and number of hidden layers= 1

Evaluating ANN Classifier results

After running ANN classifier the results will be evaluated according to a defined threshold. In this work the threshold value is defined to be (%5). According this value the output of the

ANN classifier if less than this value is defined as recognized, else not recognized. Algorithm (1) describes the MLP classifier construction, training, and evaluating results.

Algorithm 1 The ANN classifier

Input: Pattern value.

Output: Pattern code and error value.

Step 1:

1.1 Building ANN: The network consists of input layer (nine neurons), hidden layer (seven neurons), and output layer (three neurons), and they are fully connected to each stage, noting the length of iput pattern must be nine digit.

1.2 Initializing weights: initialize weight value must be stable in first layer and, must be set to one in the second layer which must be learned (weight values maybe changed).

Step 2: Training ANN using back propagation algorithm:

2.1 ANN is initially trained with a set of basic pattern codes.

2.2 Train first neurons.

For each neuron in the network do

-Computing equation (1).

function(binary sigmoid function)

2.3 Go to step2 to change weight according to equation (2).

2.4 Calculate error:

2.4.1. for output layer using equation (3).

2.4.2. for hidden layer using equation (4).

Step 3: Run ANN classifier on the present data base.

Step 4: Evaluate ANN classifier results according to a cut-of-value= (%5). If (result >= %5, not recognized & result <= %5, recognized).

Step 5: End.

Example1: ANN Classifier

Algorithm (1) is applied to train a network with 12 pattern (disease) to produce the weights array for the (input-hidden) layer and (hidden-output) layer. Table (2) shows only two cases ar recognized at the sixth row of the table with error value (%1) and the fourteenth row with error value (%2).

Integrating Neural Network with Genetic Algorithm

In order to get an optimal classifier capable of recognizing the most important features of the plant disease, with minimum error value, the concept of genetic algorithm has been introduced. In next section the way in which the genetic algorithm is used. Genetic algorithm is used to enhance the ability of artificial neural network to recognize a given pattern. The error results of proposed ANN classifier can be reduced by using GA, so the output of ANN classifier is defined with large error value if the result error is more than (%5). This ratio is a threshold (cut-of-value) to determine if GA is executed or not after the ANN algorithm. This value is chosen because if the error weight of the given pattern is very small (rarely given pattern), the change of the digit in the pattern may created a very different pattern which may be closed to another pattern in the same class or different class. As a result this small threshold (%5) may not be reached. the unrecognized pattern in the ANN classifier is entered to GA, in this stage it executes processes that will be given the new pattern code which a recognized or else not

recognized, so modify the parameters in the NN according to Table (2).

Data Encoding

Since perceptron has been used to implement the proposed technique, the binary values are used to represent the input values (pattern want to recognize), and the same for the output values. As a result the input to the GA is binary, (i.e. the chromosome is implemented using binary values, i.e. the same data type which is used in perceptron in the previous stage). The corresponding output values are binary too, Table (3) represents the input code to the GA and corresponding output code.

Fitness Operation

XOR – Logic operation has been used in genetic routine in this work to recognize string and put it in a related class. XOR operation has attractive effect on the population where the disease that has less different digits with the original input this disease. It will be the parent to create the new generation by it. XOR function has been used for the following reasons:-

It is a logic function this means it accepts binary values 0 or 1 to work since perceptron works with binary values only (0.1) and, GA is binary. XOR is easy function to define the difference between any two given binary strings or patterns. And it is used in GA to define fitness of pattern how it can be closed to another pattern in a given population. XOR logic function as a fitness operation works by sum the position number for each position with value equal to 1 in each output chromosome starting from left to right, for example(see figure 2).

Selection – Mating Pool

In this step, each chromosome (i.e. proposed ANN classifier), is ordered according to its fitness value, and selects the less value of fitness operation. In this work using the roulette-proportional selection. Its uses this operation because string that has large values, probability of distributed chromosome is high.

Crossover Operation

One point crossover is a simple and often used for GAs which operates on binary strings, crossover is the exchange of genes between the chromosomes of the two parents. In the simplest case we can realize this process by cutting two strings at a randomly chosen position and swapping the two tails. Table (4) illustrates this operation. Its uses this operation because one point crossover is faster and efficiency from the rest.

Mutation Operation

Mutation is an important part of genetic search as it helps to prevent the population from stagnating at any local optima, and flip bit mutation operator is used for this work. See Table (5) which illustrates this operation. Its uses this operation because the best and it takes a single candidate and randomly changes some aspect of it.

GA Execution

GA execution is made when the output of the ANN classifier is with higher error values (not recognized).

These diseases are fed to GA. GA execution result will be either:

1: Recognize, recognize the disease with minimum error ratio use algorithm 2 ,Or

2: Modify parameter of ANN classifier. In this case suggests modifying ANN parameter (learning rate and number of neurons), using algorithm(3). Algorithm (4) describes the general steps for Integrating NN with GA for classification of plant diseases.

Algorithm 2 GA for Classification
Input: Patterns with error value >= (%5).
Output: Either pattern with error value <= (%5), or pattern with learning rate, and number of neurons.
Step1: For j = 1 to 12 (number of training set)
-Randomly selected two patterns
-Recombine them (crossover) so as to option one offspring
-Perform mutation on the offspring
Step2: Evaluate results of GA operations
If output pattern error result <= (%5) Go to 3
Else call algorithem3 to modify (number of neurons and learning rate), then call Algorithm (1) ANN classifier.
Step3: Next j.
Step4: End.
Algorithm3 Modification ANN parameter
Input: Not recognizing pattern from ANN classifier and GA error ratio.
Output: Modified parameter.
Step1: Get difference between the error ratio from ANN and error ratio GA.
Step2: Apply (MSE) to the difference result.
Step3: find relationship between (No. of neurons, and learning rate) according MSE result.
Step4: return
Algorithm4 Integrating NN with GA
Input: Given a training set.
Output: Recognize input pattern or not.
Step1: Randomly initialize a population of classifier neural network
Step2: While pattern is not recognized Do
- Train patterns by MLP network on training set
-Evaluate the network results with threshold (%5)
If (the result error ratio >=%5) Do
Call algorithm (2).
For i = 1 to randomly training set Do
Randomly select two patterns codes
Crossover them so as to produce offspring
Perform mutation on the offspring
Step3: Update variables of ANN classifier
Step4: End.

Example 2: Integrating NN with GA classifier

In Table (2) not recognized diseases with (error ratio >=%5), are fed to GA. For example, the pattern code in the first row in Table (2) is recognized with error ratio (%41), in this case the pattern code is entered in GA to decrease the error ratio. Note

the pattern code recognizes and minimizes the error rate to zero. At seventh row of the same table the pattern recognized with (%51), and in this case the code is entered GA that is not recognized (%100). Table (6) lists results of execution GA(algorithm 2) on not recognized by ANN classifier. From Table (6) for the not recognized input patterns execute algorithm (3) to modify parameters (see Tables (7,8). So the error value decreases by small ratio, by retraining the network in new parameters (number of neurons is (9), learning rate is (0.6), and bais is (0)), as it is used in Table (7) on all results. Table (6) illustrates the results after modifying the ANN parameters (learning rate and number of neurons in the hidden layer according to Table (7).

Example 3:

The proposed classifier applied to another set of pattern diseases the results are showed in Table (10). As see in the Table the proposed classifier can capable all of recognizing pattern of diseases with minimum error value ((exactly (0.0)).

Example 4:

The proposed classifier applied to another set of pattern diseases the results are showed in Tables (11, 12, 13). As see in the Tables (11,13) the proposed classifier can capable recognizing all of pattern of diseases with minimum error value ((exactly (0.0)). Table (11) shows the proposed ANN classifier can capable of recogning some pattern of diseases with minimum error value ((exactly (0.0)).

From Table (11) the result from ANN classifier indicate that the ANN classifier could recognize 5 from 10 input patterns, and in Table (11) the ANN classifier can not capable of recognizing, so the patterns from rows (6 to 10) are fed to GA. The pattern in row 6 entered to GA for decreasing the error ratio and observes the pattern code is recognized and minimized the error ratio to the zero. In Table (11) the neural network is recognized the pattern code in the row 7 about (%74), and in this case the code entered to GA that is not recognized (%100). Table (12) list results of execution GA on selected results of ANN classifier output. So the error value decrease by small ratio, by retrained the network in new parameters (number of neurons is (14), learning rate is (0.9), and bais is (1)), so as it's using the Table (7) on all results. Table (13) illustrated the results after modified the ANN parameters (learning rate and number of neurons in the hidden layer according to Table (7).

Proposed Classifier Accuracy

Table (14) illustrates the accuracy of the proposed classifier. Notes in example (1,2) the proposed classifier is recognized 13 pattern from 15 pattern, example 3 the proposed classifier is recognized 15 pattern from 15 pattern, and example 4 is recognized 15 pattern from 15 pattern. From Table (14) illustrates the average accuracy of the proposed classifier that it is near (96%).

Conclusions

In work the feed-forward neural networks trained under back-

propagation algorithm (MLP) to classification. Now will summarize the results obtained form the work:

1)Due to the huge number of parameters in MLP, determining the optimal parameters and architecture of neural network depending on the trial-and- error approach is tedious task, time consuming, and often leads to bad results in case of large training sets. In this regard, a neural network-based classifier has been integrated Genetic algorithms searching technique to modify the neural networks parameters(no. neurons and learning rates) to recognize the plant diseases.

2)The results obtained from integrating ANN with GA revealed a great deal of improvement than those of trail-and-error approach and gives accurate results, the classifier recognizes most of input pattern with accuracy (96%).

3)A feed forward MLP network is used for plant disease classification, and is trained under back propagation algorithm. GA is a good tool to enhance ANN classifier capabilities for recognizing diseases. GA output either optimal solution to disease

4)Mean square error (MSE) was used as tool to modify ANN classifier parameters.

References

[1]E. devalue, and P. Naim, "Neural networks" Translated by A. Rawsthorne, University of Manchester, February 1995.

[2]Nurnberger, D. Nauk, R. Kruse, "Neuro-Fuzzy control based on the NEFCON-model: recent developments", Soft Computing PP-168,182, Springer-Verlag, 1999.

[3]Hansen, "State of the Art of Neural Network in Meteorology", 1997:mid-term paper for a Neural Network course at the Technical University of Nova Scotia.

[4]E. Rumelhart, B. Windrow, and M. A. Lehr, "The basic ideas in neural networks ", Communications of the Association for computing machinery, Vol.37 No.3, March 1994, PP.87-92.

[5]P. Tsongas, SE. Pap ad akis, D. Manolakis, "Plant leaves classification based on morphological features and a fuzzy surface selection technique", 2005

[6]F. Herrera, and L. Magdalena "Genetic Fuzzy system: A Tutorial", Tetra Mountains Mathematical publications Vol.13, 1997, 93-121. R. Messier, B. Rican (Eds.) Fuzzy structures current trends. Lecture Notes of the Tutorial: Genetic fuzzy systems. Seventh IFSA world congress (IFSA 97), page, June 1997.

[7]G. Edirisinghe, D. Edirisinghe, et al. "Genetic Algorithms and Neural Networks for Astrophysical Applications", American Astronomical Society Meeting 202, #02.02; Bulletin of the American Astronomical Society, Vol. 35, p.701,2003.

[8]G. A. Rovithakis, I. Chalkiadakis, and M. E. Zervakis, Member, IEEE, " High-order neural network structure selection for function approximation applications using genetic algorithms", IEEE transaction on System, Man, and Cybernetics Part B: Cybernetics Vol-34, No.1, February 2004.

[9]V. Bevilacqua, G. Mastronardi, et al. "Genetic Algorithms and Artificial

Neural Networks in Microarray Data Analysis: a Distributed Approach", Engineering Letters, 13:3, EL_13_3_14 Advance online publication: 4 November 2006.

[10]Fiszelew, P. Britos, Ochoa, et al. "Finding Optimal Neural Network Architecture Using Genetic Algorithms", Software & Knowledge Engineering Center. Buenos Aires Institute of Technology. Intelligent Systems Laboratory. School of Engineering. University of Buenos Aires, 2007.

Table (1) Symbol of disease and codes (Training set)

Symbol of disease	Pattern code
1	010010111
2	101111100
3	011010010
4	010110011
5	101000101
6	110100100
7	100110110
8	100001110
9	001010000
10	010101010
11	000111000
12	011001000

Table (2) Results of proposed ANN classifier execution

Number	Input pattern code	Result (error value in NN)
1	101010111	%41
2	011010101	%55
3	101010100	%76
4	111001010	%54
5	011110011	%35

6	011101101	%1
7	000000100	%51
8	111001101	%79
9	101100000	%58
10	111111110	%64
11	001001110	%49
12	011111110	%74
13	010111000	%56
14	000100010	%2
15	000101110	%17

Table (3): Data Encoding

Symbol of disease	Input code	Output code
1	010010111	001
2	101111100	001
3	011010010	001
4	010110011	001
5	101000101	010
6	110100100	010
7	100110110	010
8	100001110	010
9	001010000	100
10	010101010	100
11	000111000	100
12	011001000	100

Table (4) one point crossover operation

Parent	Child	Random crossover point	XOR logic function
010101010	010000101	4	15
101000101	101101010	4	17

Table (5) Flip bit mutation operation

Parent	Child	Random crossover point & mutation	XOR logic function
010101010	010000100	4	7
101000101	101101010	4	17

Table 6 GA execution (algorithm2)

P attern in NN	Error value	Pattern in GA	Error value
101010111	41%	000110111	0.0%
101010100	76%	100010000	Not recognized
111001010	54%	100101010	3%
011110011	35%	101110011	Not recognized
000000100	51%	01100110	Not recognized
111001101	79%	010101001	0.0%
101100000	58%	010100100	Not recognized
111111110	64%	011001110	60%
001001110	49%	010001110	63%
01111110	74%	001111100	0.0%
010111000	56%	010011000	0.0%
000101110	17%	1011010110	0.0%

Table 7 Relationship between Learning rates, No. of neurons, and the distance

No. of neurons	Learning rate	The distance
17	0.15	5 – 10
15	0.13	11 – 20
14	0.9	21 – 30
13	0.7	31 – 40
9	0.6	41 – 50
8	0.4	51 – 60
7	0.2	61 – 100

Table 8 Results of NN & *GA*

NN	GA	MSE	No. of neurons	Learning rate	Bias
51%	Not recognized (100%)	49	9	0.6	0
76%	Not recognized(100%)	24	14	0.9	0 or 1
35%	Not recognized(100%)	65	7	0.2	0
49%	63	14	15	0.13	0

Table (9) The Final results after modification

Error value in NN	Error value in GA
76%	0.0%
35%	79%
51%	50%
58%	0.0%
64%	0.0%
49%	0.0%

Table (10) Results of ANN classifier (Recognized (%100))

Number	Input pattern code	Result (error value in NN)
1	010101011	0.0
2	010010101	0.0
3	010010011	0.0
4	010000011	0.0
5	110000011	0.0
6	011001001	0.0
7	011001010	0.0
8	011001011	0.0
9	111001011	0.0
10	111011011	0.0
11	111111011	0.0
12	111111111	0.0
13	111111101	0.0
14	111011101	0.0
15	111110101	0.0

Table (11) Results of ANN classifier (Recognized (%100))

Number	Input pattern code	Result (error value in NN)
1	011111101	0.0
2	001111111	0.0
3	001111110	0.0
4	000100011	0.0

5	000101111	0.0
6	110101111	%61
7	010010110	%74
8	001111101	%76
9	000100110	%23
10	101101111	%73
11	010001110	0.0
12	010001010	0.0
13	010001011	0.0
14	010001111	0.0
15	110010111	0.0

Table (12) Results of NN & GA

NN	GA	MSE	No. of neurons	Learning rate	Bias
74%	Not recognized (100%)	26	14	0.9	1
76%	17%	59	8	0.4	1
23%	Not recognized (100%)	77	7	0.2	0

Table (13) The Final results after modification

Error value in NN	Error value in GA
74%	0.0%
76%	0.0%
23%	0.0%

Table (14) Accuracy of System

Input pattern	15	15	15
Recognized	13	15	15
Percentage ratio	87%	100%	100%

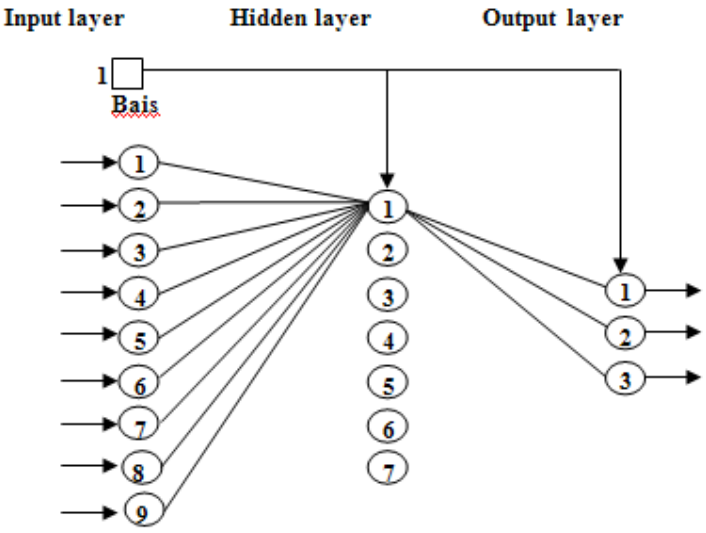

Figure (1): MLP proposed for classification plant of diseases

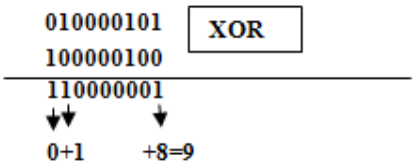

Figure (2)