International Journal of Computer Science & Communication (ISSN: 0973-7391) Vol. 3, No. 2, January-June 2012, pp. 247-251

A Genetic Approach to improve the Performance Index by Using Artificial Neural Networks

Kanaka Durga Returi[1] and Dr. Radhika, Y[2]

[1,2]*Department of Computer Science and Engineering, GITAM University, Vishakhapatnam, AP, INDIA*

[1]E-mail: returi_1210@yahoo.com, [2]E-mail: yradhikacse@gmail.com

Abstract— The selection of connection weights in Artificial Neural Network (ANN) is a key issue. To initialize the network weights before training, usually weights randomization methods are used. The main aim of weights randomization techniques is to avoid sigmoid saturation problems that cause slow training. There are different weights randomization methods such as Manual, Automatic, optimized for uniform distribution of networks. In this study, a new hybrid model of Neural Networks together with Genetic Algorithm (GA) is used to initialize and optimize the connection weights of ANN so as to improve the performance of the Artificial Neural Network for the mass transfer function. This study reveals that the mass transfer enhancement in tube with entry region coil-disc assembly as an insert promoter by using ANN with Genetic Algorithm (GA) reported a high degree of accuracy.

Keywords— Artificial Neural Network; Genetic Algorithm; Training; Mean Square Error; Weights; Performance.

1. INTRODUCTION

In the recent years two popular research areas such as Neural Networks (NN) and Genetic Algorithms (GA) have good attention. These two are computational abstractions of biological information processing systems and both have captured the thoughts of the researchers all over the world. So many difficulties in life are solved through some kind of searching method. Although, gradient descent techniques have been used effectively to train feed forward neural network connection weights, researchers have experimented with developing a best possible set of connection weights with biologically inspired genetic algorithms.

Several researchers did research on the combination of Neural Networks and Genetic Algorithms. Some of them are: Simon Haykin [1] presented the detailed analysis of back-propagation learning and multi-layer perceptrons in neural networks. Stjepan Oreski et al [2] predicted the borrower's ability to repay the loan from bank on time with genetic algorithm and artificial neural networks approach. J. Rezaeian et al [3] presented a new nonlinear programming model in a dynamic environment based on the GA and ANN. Mokhatab Rafiei, F et al [4] suggested a model to predict financial health of companies based on artificial neural networks (ANN) and Genetic Algorithm (GA). Adel, H, M and Raed, A, Z [5] presented a lot of modification on a machine learning method inspired by the human immune system called artificial immune system with the aid of Genetic Algorithm. Daniel Rivero et al [6] described a new technique that uses Genetic Programming in order to automatically develop simple ANNs, with a low number of neurons and connections. Adel Mellit et al [7] developed an artificial neural network based genetic algorithm model for generating the sizing curve of stand-alone photovoltaic systems. Kyoung-jae Kim and Ingoo Han [8] proposed genetic algorithms approach to feature discretization and the determination of connection weights for artificial neural networks to predict the stock price index. Taeksoo Shin and Ingoo Han [9] proposed an integrated thresholding design of the optimal or near-optimal wavelet transformation by GA to represent a significant signal most suitable in ANN models. Hyun, J, K and Kyung, S, S [10] investigated the effectiveness of a hybrid approach based on the ANNs for time series properties, such as the adaptive time delay and the time delay neural networks with the genetic algorithms. Massimiliano Versace et al [11] evaluated the performance of a heterogeneous mixture of neural network algorithms for predicting the exchange-traded fund. DIA. R.J. Kuo [12] proposed fuzzy neural network with initial weights generated by genetic algorithm for the sake of learning fuzzy IF–THEN rules for promotion obtained from marketing experts. Leonard, J and Kramer, M, A [13] discussed the application of artificial neural networks to chemical engineering problems. Pollard, J, F et al [14] described a series of tests in which a back propagation neural network was used for process identification.

2. ARTIFICIAL NEURAL NETWORK

ANN is a collection of mathematical models that emulate the real neural structure of the brain. The brain has evolved many efficient ways to store and process information that we attempt to model through artificial neural networks. In general, ANN is made up of individual interconnected simple processing elements called neurons, arranged in a layered structure to form a network that is capable of performing massively parallel computation. ANN is a network of many simple processors called units, linked to certain neighbours with varying coefficients of connectivity (called weights) that represent the strength of these connections. The basic unit of ANNs, called an artificial neuron, simulates the basic functions of natural neurons: it receives inputs, processes them by simple combination and threshold operations and outputs a final result.

2.1 The Neuron

Artificial neural networks had their start relatively recently in the 1940's. The basic processing unit of a neural network is the neuron. At the highest level a neuron receives a series of inputs and depending upon the strength of the input and the connection determines whether the neuron will fire or not. The inputs are multiplied by their synaptic connection and summed. This sum is then used as input for a transfer function which calculates the output of the neuron. This function is represented in Equation 1. The basic conceptual framework for a single neuron is show in Figure 1.

$$r^{Out} = g\left(\sum_{i} w_i \, r_i^{in}\right) \qquad (1)$$

'w' represents the weight of the synaptic connection between the input and the neuron, 'r' represents the input value. 'g' represents the transfer function of the neuron. 'i' represents the number of inputs.

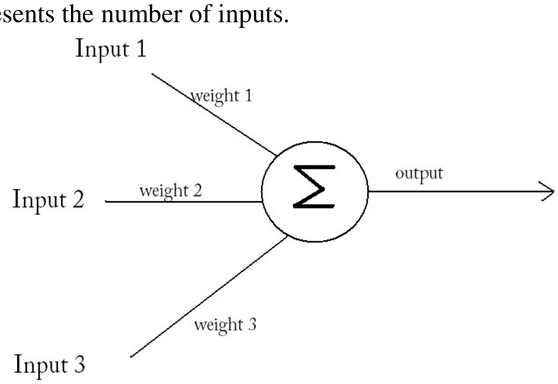
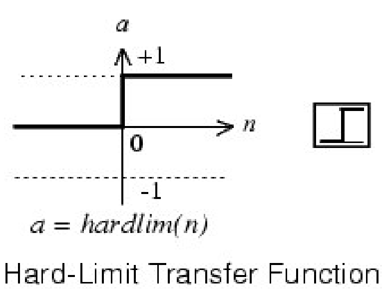


Figure 1: A Single Neuron

Each neuron utilizes a transfer function that determines a neurons response to the sum of its inputs. Some commonly used transfer/activation functions are shown in Figure 1.



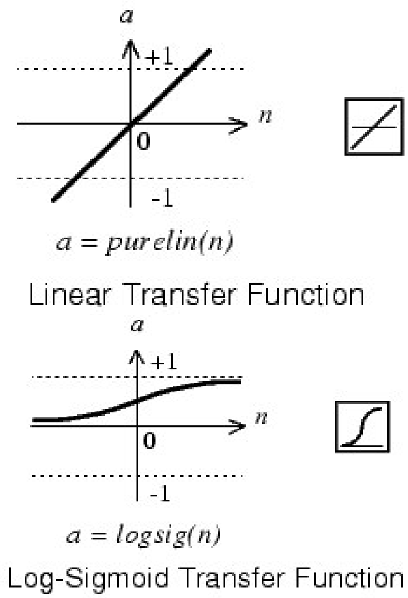


Figure 2: Sample Transfer/ Activation functions.

These different transfer functions result in different neuron output. For example a hard limit function will only propagate a 1 or a 0, revealing little information to how accurate the neuron is but resulting in very clear propagation of signal. Whereas a continuous transfer function is much more precise with outputs but can potentially propagate irrelevant signals. The connections between neurons represent a weighted connection called a synapse. Like the single neuron seen the inputs are multiplied by the synaptic weight and passed through a transfer function. A typical ANN consists of an input layer, k hidden layers, and an output layer. This network topology is determined by the user and is based on the type and complexity of the problem space.

2.2 Training

Training is an iterative process that seeks to modify the network through numerous presentations of data. There are many different methods to train neural networks, the two main distinctions are unsupervised and supervised learning. An unsupervised neural network only uses the input data to adjust its synaptic weights. Supervised learning however relies on a set of training data with known target values. In other words, the training data consists of a set of input patterns and output values. The goal of training is to optimize a function that will map the inputs to the outputs that can be used to correctly approximate unseen inputs.

Constructing an ANN using a supervised learning methodology requires the initialization of a network with random synaptic weights between neurons. At this point an input signal presented to the network would result in no meaningful output. To derive a meaningful output the network synapses must be adjusted. The method to adjust the many weights of the network requires a calculation of error of the network for an input pattern at each epoch. An epoch represents an iteration of measuring the output error and updating the synaptic weights in response. The error of the network for a given input pattern is described as the difference between the network output and the desired output value. A standard error measure, such as mean squared error is often used to describe this and shown in equation 2.

$$Error = \left| \frac{TargetValue - Network\ Output}{Network\ Output} \right| \qquad (2)$$

While this error value shows how far the network output is from a desired value, it does not reveal anything about how to correct the network to more closely match the desired value. To minimize the error value, the error is used in a learning function to update the synaptic connections of the network. A simple synaptic learning function is shown in Equation 3.

$$W_{new} = W_{old} + \alpha(Error)\, x\, Input \qquad (3)$$

W_{new} represents the new weight of the input neuron, W_{old} the old weight of the neuron, alpha the learning rate, Target Value-output is the error function and the input is the input that goes through the synapse A learning rate is often used to control how quickly the weights are updated. If a large value is used the weights of the network will oscillate wildly if set too low it will take more epochs to adjust the weights.

Several learning paradigms exist to train networks; one of the most commonly used is back propagation. Back propagation is accomplished through a process where error correction flows through the network in a reverse direction. Back propagation attempts to minimize the amount of error by modifying the synapses with the strongest connections. The weights of these synapses are then modified to a greater degree than their weaker counterparts using a method similar to Equation 3.

2.3 Testing

After training has been completed, usually signalled by a lack of further decrease in the error or after a set number of epochs, the weights of the network are set and testing of new samples begins. During testing the testing data is presented to the network to obtain a measure of performance. This performance is measured by a similar method that is used to determine the error of the network during training.

2.4 Cross Validation

An important part in evaluation of any classifier is the use of cross validation. However with microarray data this is difficult, as typically there are a statistically small number of samples, making over fitting of the model a real possibility. Some basic cross validation techniques include, leave one out, hold out and k-fold.

Leave one out cross validation is one of the most extreme cross validation methods. In this case, 1 sample is used for testing and n-1 samples are used for training. This is repeated n times, until all individuals have taken a turn as the testing sample. The n-fold cross validation method is a scaled down version of the leave one out method, where the dataset is divided into n divisions.

While supervised artificial neural networks are powerful tools by themselves they can only sort what is presented to them. The typical structure of microarray data has too many features and not enough replicates. To alleviate this problem, a Genetic Algorithm (GA) can be used to evaluate combinations of genes.

3. GENETIC ALGORITHMS

Genetic algorithms comprise a search algorithm that guides its search based on a model of evolution by which organisms continually improve over generations, through, selection, crossover and mutation. Evolution derives its power by evaluating many possible solutions at once, and propagating the fittest.

In a genetic algorithm, a population of organisms is represented by a population of short strings called chromosomes. Each chromosome represents a different portion of the possible search space that is to be evaluated. A search space is all of the possible combinations or values of features for a given problem. One could evaluate every possible solution for a problem, known as a brute force approach, but as the number of parameters increases the search space grows in dimensionality, resulting in problem spaces that are too large to search with exhaustive methods. Genetic algorithms have been shown to be a robust search method for problems with extremely large search spaces. Within a search space there are often many local minima/maxima and one global minima/maxima. Local minima/maxima are solutions that are close but are not the best solution. Minima or maxima are substituted depending on the direction that function is being optimized. Many optimization algorithms go to great lengths to avoid or escape local solutions. The global solutions represent the best possible solution and are often difficult to find.

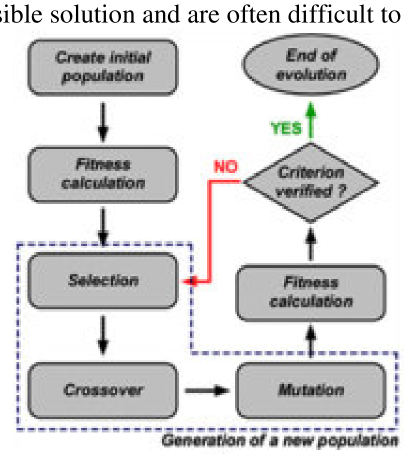


Figure 3: Simple Genetic Algorithm

3.1 Representation

The choice of representation in a genetic algorithm is of utmost importance; this process involves mapping parameters into a string that the algorithm can manipulate. Many Genetic algorithms utilize a binary representation of the data. In a feature selection problem this would consist of a string the length of the feature set, where each character was a binary value that represented the presence (1) or absence (0) of a feature. Representation is also important due to linkage. Linkage is the probability that genes will be inherited together. This probability is based directly upon how far apart the genes are located from each other. Two genes that are close together are less likely to separate than two genes that are located at either end of the chromosome. This is because there are more possible crossover events that could separate the two genes when the genes are farther apart, than if the genes were adjacent to each other.

3.2 Fitness Function

The fitness function represents the problem that the genetic algorithm seeks to solve. At each generation the performance of the individuals within the population must be measured against this function. The performance of an individual with the fitness function is used to determine which organisms are allowed to reproduce.

3.3 Selection

Selection is the method by which GAs determine which chromosomes should propagate. The scores from the fitness function are evaluated by the selection function. Individuals reproduce proportionately to their fitness. There are several types of selection methods including ranked and tournament among others. There is no correct answer for which selection method to use. The method implemented here is that of a roulette wheel. To implement a roulette wheel the fitness of the entire population must be evaluated at each generation. Next the probability of selection for each individual is calculated by dividing the individual's fitness by the sum of the population's fitness.

$$\Pr obability\,(i) = \frac{f_t}{\sum_0^i f_t} \qquad (4)$$

Individuals are then ranked in descending order and a vector is constructed of accumulating probabilities.

$$SelectionVector = \sum_{i=1}^{n} \Pr obability\,(t) + \Pr obability\,(t-1) \qquad (5)$$

4. METHODOLOGY

A genetic algorithm that optimizes inputs for a neural network was constructed in MATLAB 7.0. This system evaluated the classification ability of feature combinations using an artificial feed forward neural network. Each set of features was used to train a neural network and then the classification ability of those features was evaluated. High scoring features were preserved by the GA while low scoring features and feature combinations were discarded. A fixed feature size was used because the neural network requires a fixed number of input features.

In most supervised learning studies, the dataset is divided into a training set and a testing set. The experimental apparatus and procedure are described in detailed in Vaka Murali Mohan et al [15] to study the potential benefits of the mass transfer enhancement in tube with entry region coil-disc assembly as an insert promoter. The original data division of 225 training and 100 testing samples was changed in this study to increase the number of samples that could be included within the training dataset. By increasing the number of samples in the training set, the ANN is able to perform at a higher level due to a more robust training set. A uniformly distributed initial weight set is produced within the range of [-1, 1]. The network parameters such as learning rate and momentum term are set in all attempts as 0.1 and 0.3, respectively.

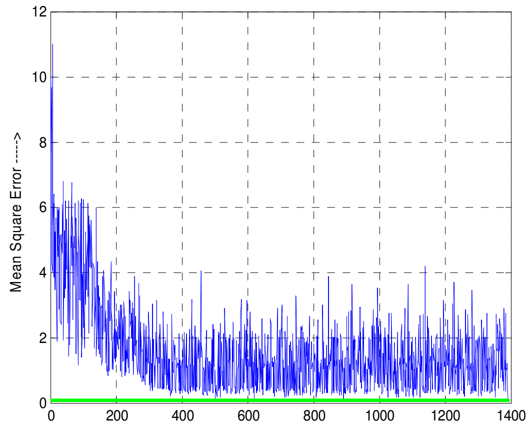
5. RESULTS AND DISCUSSION

In neural network each neuron used a tangent sigmoid transfer function as seen in Equation 6.

$$g(x) = \frac{1}{1+e^{-x}} \qquad (6)$$

At every generation, each chromosome was decoded into its values and 225 samples were used to train a neural network using the MATLAB adapt function. The adapt function uses incremental training, resulting in an update of weights after presentation of each input value. Each network was initialized to the same weight values to decrease fluctuations in performance caused by randomized starting weights. Each network was trained for 1400 epochs, after which classification ability was evaluated on the 100 testing samples. For each network this process was repeated 20 times with different samples of training and testing classes.

The main aim of training a neural network is to update the weights; with these weights a set of inputs produces the desired outputs. It calculates the appropriate weight updates, Mean-Squared-Error (MSE) and learning rate to accelerate error convergence. The artificial neural network performance for the mass transfer function is shown in figure 4. The figure shows the MSE=0.6246%, Standard Deviation=1.2819%, R2 = 0.99. It gives the good and considerable performance of that mass transfer function. This study reveals that the mass transfer enhancement in tube with entry region coil-disc assembly as an insert promoter by using ANN with Genetic Algorithm (GA) reported a high degree of accuracy.



No. of Epochs ---------->

Figure 4: Training graph

6. CONCLUSION

In this study, a new hybrid model of Neural Networks and Genetic Algorithm (GA) to initialize and optimize the connection weights of ANN so as to improve the performance of the Artificial Neural Network performance for the mass transfer function. The artificial neural network performance for the mass transfer function shows the MSE=0.6246%, Standard Deviation=1.2819%, $R^2 = 0.99$. It gives the good and considerable performance of that mass transfer function. This study reveals that the mass transfer enhancement in tube with entry region coil-disc assembly as an insert promoter by using ANN with Genetic Algorithm (GA) reported a high degree of accuracy.

7. REFERENCES

[1]. Simon Haykin, "Neural Networks: A Comprehensive Foundation" 2nd edition, Prentice Hall PTR Upper Saddle River, NJ, USA, 1998.

[2]. Stjepan Oreski., Dijana Oreski., Goran Oreski "Hybrid system with genetic algorithm and artificial neural networks and its application to retail credit risk assessment" Expert Systems with Applications, Volume 39, Issue 16, 15 November 2012, pp 12605–12617

[3]. J. Rezaeian., N. Javadian., R. Tavakkoli-Moghaddam and F. Jolai "A hybrid approach based on the genetic algorithm and neural network to design an incremental cellular manufacturing system" Applied Soft Computing Volume 11, Issue 6, September 2011, pp 4195–4202

[4]. F. Mokhatab Rafiei., S.M. Manzari and S. Bostanian "Financial health prediction models using artificial neural networks, genetic algorithm and multivariate discriminant analysis: Iranian evidence" Expert Systems with Applications, Volume 38, Issue 8, 2011, pp 10210–10217

[5]. Adel Hamdan Mohammad and Raed Abu Zitar "Application of genetic optimized artificial immune system and neural networks in spam detection" Applied Soft Computing, Volume 11, Issue 4, June 2011, pp 3827–3845

[6]. Daniel Rivero., Julian Dorado., Juan Rabuñal., Alejandro Pazos "Generation and simplification of Artificial Neural Networks by means of Genetic Programming" Neurocomputing, Volume 73, Issues 16–18, October 2010, pp 3200–3223

[7]. Adel Mellit., Soteris A. Kalogirou and Mahmoud Drif "Application of neural networks and genetic algorithms for sizing of photovoltaic systems" Renewable Energy Volume 35, Issue 12, December 2010, pp 2881–2893.

[8]. Kyoung-jae Kim and Ingoo Han "Genetic algorithms approach to feature discretization in ANN for the prediction of stock price index" Expert Systems with Applications Volume 19, Issue 2, Aug 2000, pp 125–132

[9]. Taeksoo Shin and Ingoo Han "Optimal signal multi-resolution by genetic algorithms to support artificial neural networks for exchange-rate forecasting" Expert Systems with Applications Volume 18, Issue 4, 2000, pp 257–269

[10]. Hyun-jung Kim and Kyung-shik Shin "A hybrid approach based on neural networks and genetic algorithms for detecting temporal patterns in stock markets" Applied Soft Computing Volume 7, Issue 2, Mar 2007, pp 569–576

[11]. Massimiliano Versace., Rushi Bhatt., Oliver Hinds., Mark Shiffer "Predicting the exchange traded fund DIA with a combination of genetic algorithms and neural networks" Expert Systems with Applications Volume 27, Issue 3, October 2004, pp 417–425

[12]. R.J. Kuo "A sales forecasting system based on fuzzy neural network with initial weights generated by genetic algorithm" European Journal of Operational Research Volume 129, Issue 3, 16 March 2001, pp 496–517

[13]. Leonard, J and Kramer, M. A, "Improvement of the back propagation algorithm for training neural networks" Computers & Chemical Engineering, 14 (1990) 337-341.

[14]. Pollard, J.F., Broussard, M.R., Garrison, D.B and San, K.Y, "Process identification using neural networks" Computers & Chemical Engineering, 16 (1992) 253-270.

[15]. Vaka Murali Mohan., P. R. Prasad., V. Sujatha & S. Sarveswarao, "Flow pattern and model development for coaxially placed entry region coil-disc assembly as turbulence promoter in circular conduits" Expt. Thermal and Fluid Science, Vol.32, 2008, pp 1748-1753.