Information Processing Letters 116 (2016) 22–25

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

A solution representation of genetic algorithm for neural network weights and structure

Najmeh Sadat Jaddi *, Salwani Abdullah, Abdul Razak Hamdan

Data Mining and Optimization Research Group (DMO), Center for Artificial Intelligence Technology, Faculty of Information Science and Technology, National University of Malaysia, Malaysia

ARTICLE INFO

Article history:
Received 20 July 2014
Received in revised form 17 July 2015
Accepted 6 August 2015
Available online 10 August 2015
Communicated by S.M. Yiu

Keywords:
Approximation algorithm
Artificial neural network training
Optimization of weights and structure
Genetic algorithm
Time series prediction

ABSTRACT

This paper presents a new solution representation for genetic algorithm to optimize the neural network model. During the optimization process, the weights, biases and structure of the neural network are considered for altering. The quality of the model is examined by a cost function that deliberates over both minimization of error and complexity of the neural network model. The performance of the proposed method is investigated by applying it on two time series prediction problems. The results show promising results when we compare it with other methods in the literature.

1. Introduction

Over the past years, the demand of Artificial Neural Network (ANN) training in many areas has been growing [1,4,5,14]. The main reason is the nonlinearity of the artificial neural network. Selection of the proper weights, number of layers and nodes in each layer is the most challenging issue in ANN models. The number of layers and nodes affect the complexity of ANN model and therefore increasing the difficulty for the training process. In this case an economical ANN is required, because a very small network may not be able to characterize the real state due to its limited potential, while for a huge network besides making its process complex, it may provide noise in the training data and therefore fail to present its superior capability [7].

Sexton et al. [12] applied tabu search for ANN training. Later on, Sexton et al. [13] used simulated annealing and genetic algorithm (GA) for the same problem. Hill climbing algorithm was used for training the neural network in [2]. A hybrid Taguchi-genetic algorithm was employed in [6]. The most common learning algorithm is the back-propagation; however, it leads to noisy fitness evaluation which is the main disadvantage of this technique [1]. In recent years researchers were interested to produce ideas of merging ANN with other search algorithms for superior performance neural networks [6,8–10]. Ludermir et al. [7] applied hybridization of simulated annealing and tabu search to optimize the weights and connection of the neural network. Subsequently, they extended their studies by applying the hybridization of the simulated annealing, tabu search and genetic algorithm [15].

In this paper, a genetic algorithm based dynamic neural network (GADNN) is proposed to preside over both the performance and the complexity of the neural network for training process. This method provides the opportunity of checking different weights, biases, number of hidden layers, number of nodes and selected inputs during the search process. Therefore, it has the chance to find an effective model with less prediction error and less complexity.

* Corresponding author.
E-mail address: najmehjaddi@gmail.com (N.S. Jaddi).

http://dx.doi.org/10.1016/j.ipl.2015.08.001
0020-0190/

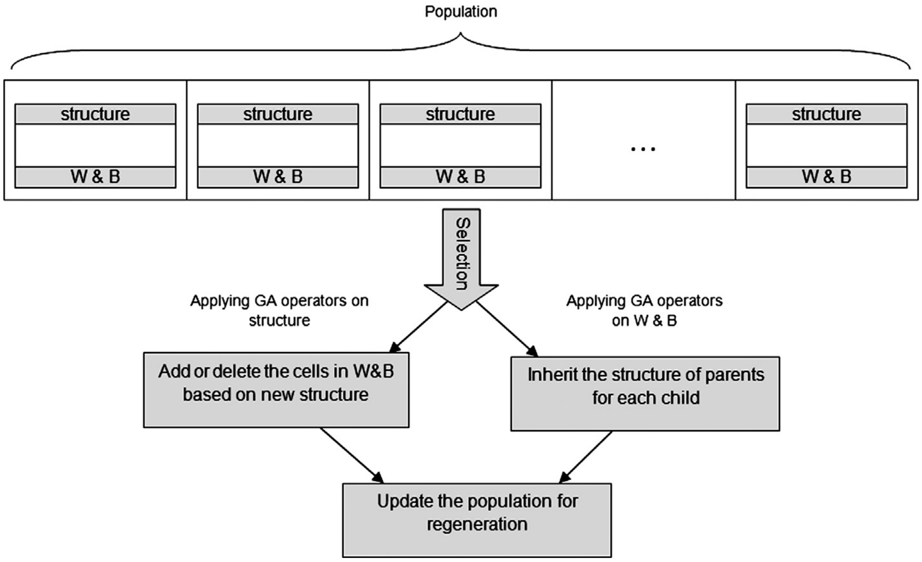

Fig. 1. Schematic of the GADNN procedure.

2. GADNN

2.1. Solution representation and construction

In this work, each solution is represented by two one-dimensional vectors. The first vector is for structure of the solution and the second vector is for weights and biases in the neural network. Each cell in the structure solution contains 0 or 1, whilst each cell in the weights and biases solutions, W&B, contains real number between range of $[-1, +1]$. In the structure solution, three cells are considered for the number of hidden layers and another three cells are used for the number of nodes in each hidden layer. Three numbers in structure solution are binary representation of number of the hidden layers and number of nodes in each hidden layer. The inputs for the neural network is chosen based on those employed in the literature, in order to have a fair comparison with other approaches in the literature [3,8,9].

For the weights and biases solution, W&B, the length of the vector is based on the number of weights plus the number of biases in the network. The number of weights and biases is calculated based on the number of hidden layers and the number of nodes in each hidden layer involved in the structure solution.

2.2. Fitness function

The fitness function is employed to evaluate the quality of the solution in successive iterations. With the use of the fitness function, a solution will be selected that optimizes an objective function. In this work, the fitness function, $f(s)$ of solution, is calculated by the average of the error, ε and the percentage of the number of weights (connections) in the network which is adapted from [15] and is given by equation (1). In the prediction problem, the error ε, is based on the mean squared error (MSE) or root mean squared error (RMSE) percentage expressed by equation (2),

$$f(s) = \frac{1}{2}(\varepsilon + p) \quad \text{where } p = \frac{100}{n}\sum_{i=1}^{n} w_i \tag{1}$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \qquad RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{2}$$

where N is the total number of instances, and y_i and $\hat{y}_i$ are the actual value and the model output respectively.

2.3. GADNN mechanism

The initial solution in the initial population is produced by random generation of structure solution, and then the W&B solution is randomly generated based on the structure of each individual. Population size is set to 100 as well as the number of generations which is set to 100. A single-point crossover is applied in this work. The crossover rate is 0.6 and the mutation rate is 0.4. In GADNN an additional probability is employed in order to balance the application of the crossover and mutation on the structure solution or W&B solution which is set to 0.5. The schematic procedure of GADNN and the pseudocode is shown in Fig. 1 and Fig. 2 respectively.

The crossover and mutation are applied with the probabilities of 50% on the structure of the parents, and there are 50% chances that the crossover and mutation would be applied on W&B during the iterations. If the crossover and mutation are to be applied on the structure of parents, then the weights and biases cells in the W&B will be randomly added or deleted to fit the structure of the solution. However, if the crossover and mutation are to be applied on W&B, then the structure of the child will be inherited from its parents. This mechanism allows the GADNN to have an extensive variety of solutions with the different structures and values of weights. The iterations will be stopped when the number of iterations exceeds the number of generations.