Particle Swarm Optimisation for Evolving Artificial Neural Network

Chunkai Zhang, Huihe Shao
Institute of Automation, Shanghai Jiao Tong University
Shanghai, 200030, P. R. China

Yu Li
Institute of LSI, Shanghai Jiao Tong University
Shanghai, 200030, P. R. China

Abstract

The information processing capability of artificial neural networks (ANNs) is closely related to its architecture and weights. The paper describes a new evolutionary system for evolving artificial neural networks, which is based on the particle swarm optimisation (PSO) algorithm. Both the architecture and the weights of ANN's are adaptively adjusted according to the quality of the neural network. This process is repeated until the best ANNs is accepted or the maximum number of generations has been reached. And a strategy of evolving added nodes and a partial training algorithm are used to maintain a close behavioural link between the parents and their offspring. This system has been tested on two real problems in the medical domain. The results show that ANN's evolved by PSONN have good accuracy and generalisation ability.

1 Introduction

There have been many attempts to design artificial neural networks' architectures automatically, such as various constructive and pruning algorithms [1, 2]. However, "Such structural hill climbing methods are susceptible to becoming trapped at structure local optima, and the result depends on initial network architectures." [3].

Design of a near optimal ANNs architecture can be formulated as a search problem in the architecture space where each point represents an architecture. Miller et al. indicated that evolutionary algorithms are better candidates for searching the architecture space than those constructive and pruning algorithms mentioned above [4]. GAs was used to evolve ANNs, but the evolution of ANNs architectures often suffers from the permutation problem [5]. This problem not only makes the evolution inefficient, but also makes crossover operators more difficult to produce highly fit offspring.

This paper proposes an evolutionary system for evolving feed-forward ANNs called PSONN, which combines the architectural evolution with weight learning. And a strategy of evolving added nodes and a partial training algorithm are used to maintain the behavioural link between a parent and its offspring to prevent destruction of the behaviour already learned by the parent Its purpose is to produce very compact ANNs with good generalization ability.

The rest of this paper is organised as follows. Section 2 describes the PSO algorithm and the PSONN system, and the motivations on how to evolve the entire ANNs. Section 3 presents experimental results on PSONN in the medical domain. The paper is concluded in Section 4.

2 PSONN system

2.1 Particle swarm optimisation (PSO)

In PSONN system, the learning algorithm is the PSO algorithm. PSO is a population based optimisation algorithm that is motivated from the simulation of social behaviour [6]. Each individual in PSO flies in the search space with a velocity that is dynamically adjusted according to its own flying experience and its companions' flying experience. Compared with other evolutionary algorithms, such as GAs, PSO algorithm possesses some attractive properties such as memory and constructive cooperation between individuals, so it has more chance to "fly" into the better solution areas more quickly and discover reasonable quality solution much faster. In this paper we propose an improved PSO algorithm, which is as follows:

1) Initialise positions *Pesentx* and associated velocity *v* of all individuals (potential solutions) in the population randomly in the *D* dimension space.
2) Evaluate the fitness value of all individuals.
3) Compare the *PBEST[]* of every individual with its current fitness value. If the current fitness value is better, assign the current fitness value to *PBEST[]* and assign the current coordinates to *PBESTx[][d]* coordinates. Here *PBEST[]* represents the best fitness value of the nth individual, *PBESTx[][d]* represents the dth component of an individual.
4) Determine the current best fitness value in the entire population and its coordinates. If the current best fitness value is better than the *GBEST*, then assign the current best fitness value to *GBEST* and assign the current coordinates to *GBESTx[d]*.
5) Change velocities and positions using the following rules:

0-7803-6583-6/00/$10.00 © 2000 IEEE

$$v[][d] = W * v[][d] + C1 * rand * (PBESTx[][d] - Pesentx[][d]) + C2 * rand * (GBESTx[d] - Pesentx[][d]) \quad (1)$$

$$Pesentx[][d] = Pesentx[][d] + v[][d]$$

$$W = W_{\infty} + (W_0 - W_{\infty})[1 - t/K]$$

where $C1 = C2 = 2.0$, t and K are the number of current iterations and total generation. The balance between the global and local search is adjusted through the parameter $W \in (W_0, W_{\infty})$.

6) Repeat step 2) - 6) until a stop criterion is satisfied or a predefined number of iterations is completed.

Because there are not selection and crossover operator in PSO, each individual in an original population has a corresponding partner in a new population. It can avoid the premature convergence and stagnation in GAs to some extent.

2.2 PSONN system

In PSONN all the data sets are partitioned into three sets: a training set, a validation set, and a testing set. Such use of the validation set improves the generalisation ability of evolved ANNs and introduces little overhead in computation time.

The major steps of PSONN can be described as follows:

1) Generate an initial population of M networks. The number of hidden nodes and the initial connection density for each network are uniformly generated at random within certain ranges. The random initial weights are uniformly distributed inside a small range.
2) Using the PT algorithm to train each network's nodes on the training set (this process can evaluate the quality of a given network architecture).
   a. Choose a network as a parent network, then randomly generate *N-1* initial networks as a population where each network's initial weights uniformly generated at random within certain ranges, but its architectures are the same as the parent's architecture, and then the parent network is added into the population.
   b. Employ the PSO algorithm to evolve this population for a certain number of epochs. The number of epochs, K, is specified by the user. Here the encoding scheme is that each individual is to parameterise a whole group of g nodes in ANNs, this means that every component of each individual represents a connection weight.
   c. The best network that survived will join the network architecture evolution.
3) All survived networks form a new population. Rank the networks in the population according to their error values on the validation set.
4) If the best network found is accepted or the maximum number of generations has been reached, go to step 7). Otherwise continue.
5) Employ the PSO algorithm to evolve the network's architecture. Here each individual in PSO represents the number of the hidden nodes and the connection density of a network. When the network architecture of an individual changes, there are two situations:
   a. If some hidden nodes need to be added to a network, the PT algorithm only evolves the new added nodes to explain as much of the remaining output variance as possible. It can decrease the computation time than evolving the entire network and prevent from destroying the behaviour already learned by the parent
   b. If some hidden nodes need to be deleted from a network, the nodes will be deleted in the reverse order in which they were originally added to the network or uniformly at random. Then the PT algorithm evolves the weights of the entire network.
6) Go to Step 3).
7) Train the best ANN further using the PT algorithm on the combined training and validation set until it "converges".

In PSONN, evolving ANNs' architectures and weight learning are alternated. This process can avoid a moving target problem resulted from the simultaneous evolution of both architectures and weights [7], and maintain the behavioural link between a parent and its offspring [11]. And the network architectures are adaptively evolved by PSO algorithm, starting from the parent's weights instead of randomly initialised weights, so this can preferably solve the problem of the noisy fitness evaluation that can mislead the evolution [3].

3 Experimental studies

In order to evaluate the ability of PSONN in evolving ANNs, it was applied to two real problems in the medical domain, i.e., breast cancer and heart disease. All data sets were obtained from the UCI machine learning benchmark repository. These medical diagnosis problems have the following characteristics [12]:

- The input attributes used are similar to those a human expert would use in order to solve the same problem.
- The outputs represent either the classification of a number of understandable classes or the prediction of a set of understandable quantities.
- Examples are expensive to get. This has the consequence that the training sets are not very large.

The purpose of the breast cancer data set is to classify a tumour as either benign or malignant based on cell

descriptions gathered by microscopic examination. It contains 9 attributes and 699 examples of which 485 are benign examples and 241 are malignant examples. The purpose of the heart disease data set is to predict the presence or absence of heart disease given the results of various medical tests carried out on a patient, it contains 13 attributes and 270 examples. For the breast cancer set, the first 349 examples of the whole data set were used for training, the following 175 examples for validation, and the final 175 examples for testing. For the heart disease set, the first 134 examples were used for training set, the following 68 examples for the validation set, and the final 68 examples for the testing set. Table 1 show the experimental results averaged 30 runs for the testing data set.

Table 1. Accuracy of the evolved ANNs by PSONN

	Breast cancer data set			Heart disease data set		
	Mean	Min	Max	Mean	Min	Max
Error	1.322	0.153	3.362	11.724	9.958.	13.637
Error rate	0.011	0.000	0.051	0.146	0.112	0.186

Table 2. Comparison with other work

	Breast cancer data set			Heart disease data set		
	FNNCA	HDANNS	PSONN	MSM1	HDANNS	PSONN
Error rate (Mean)	0.0145	0.0115	0.0112	0.1653	0.1478	0.1460

We compare the results of PSONN with other works. For the breast cancer problem, Setiono and Hui have proposed a new ANNs constructive algorithm called FNNCA [13]. Prechelt also reported results on manually constructed ANN using HDANN [12]. For the heart disease problem, Bennet reported a testing result with their MSM1 method [14], and the best manually designed ANN's also gave a result. The results are show in Table 2.

From the results, we know that PSONN is able to evolve both the architecture and weights of ANNs, and the ANNs evolved by PSONN has better accuracy and generalisation ability than the other algorithms.

4 Conclusions

PSONN perfectly harmonises the architecture and weights evolution of artificial neural networks, which effectively alleviate the noisy fitness evaluation problem and the moving target problem. ANNs evolved by PSONN has very compact ANNs with good generalization ability.

Acknowledge

This work is supported by National 973 Fundamental Research Program of China. The author thanks Prof. Jun Gu of Hong Kong University of Science and Technology for his encouragement and support in this research.

References

[1] N.Burgess. "A constructive algorithm that converges for real-valued input patterns". *Int. J. Neural Syst.*, vol 5, no. 1, pp. 59-66, 1994

[2] R. Reed. "Pruning algorithms-A survey", *IEEE trans. Neural Networks*, vol 4, pp. 740-747, 1995.

[3] D. B. Fogel. *Evolutionary computation: toward a new philosophy of machine intelligence.* New York: IEEE Press, 1995.

[4] G. F. Miller, P. M. Todd, and S. U. Hegde. "Designing neural networks using genetic algorithms". *In Proc. 3rd Int. Conf. Genetic Algorithms Their Applications,* CA: Morgan Kaufmann, pp. 379-384, 1989.

[5] R. K. Belew, J. MchInerney and N. N. Schraudolph. "Evolving networks: Using GAs with connectionist learning". *Computer Sci. Eng. Dept., Univ. California-San Diego, Tech. Rep.* CS90-174 revised, Feb. 1991.

[6] Kennedy, J., and Eberhart, R. C. "Particle swarm optimization", *in Proc. IEEE International Conference on Neural Networks,* IEEE Service Center, Piscataway, NJ, pp. 39-43, 1995.

[7] X. Yao. "A review of evolutionary artificial neural networks". *Int. J. Intell. Syst.*, vol. 8, no. 4, pp. 539-567, 1993.

[8] J. R. McDonnell and D. Waagen. "Evolving recurrent perceptrons for time-series modeling". *IEEE Tran. Neural Networks,* vol. 5, no. 1, pp. 24-38, 1994.

[9] V. Maniezzo. "Genetic evolution of the topology and weight distribution of neural networks". *IEEE Trans. Neural Networks*, vol. 5, no. 1, pp. 39-53, 1994.

[10] Fahlman, S.E., and Lebiere, C.. "The cascade-correlation learning architecture". *In D.S.Touretzky. Advances in neural information processing systems 2,* pp. 524-532, San Mateo, CA: Morgan Kaufmann, 1990.

[11] P.J. Angeline, G.M.Sauders, and J.B. Pollack. "An evolutionary algorithm that constructs recurrent neural networks". *IEEE Trans. Neural Networks,* vol. 5, pp. 54-65. 1994.

[12] L. Prechelt. "Some notes on neural learning algorithm benchmarking". *Neurocomputing,* vol. 9, no. 3, pp. 343-347. 1995.

[13] R. Setiono and L. C. K. Hui. "Use of a quasinewton method in a feedforward neural network construction algorithm". *IEEE Trans. Neural Network,* vol. 6, pp. 740-747, 1995.

[14] K. P. Bennett and O. L. "Mangasarian. Robust linear programming discrimination of two linearly inseparable sets". *Optimization Methods Software,* vol. 1, pp. 23-34, 1992.