ELSEVIER

Information Sciences 112 (1998) 1–6

INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL

Improvement of cascade correlation learning algorithm with an evolutionary initialization [1]

Hualou Liang *, Guiliang Dai

Computing Center, Institute of High Energy Physics, Chinese Academy of Sciences, P.O. Box 918(7), Beijing 100039, People's Republic of China

Received 4 September 1994; received in revised form 5 August 1996; accepted 15 August 1997

Abstract

Cascade Correlation (CC) as a constructive algorithm for artificial neural networks stands out for solving hard classification problems. In this note we propose an extension of this technique that uses genetic algorithm for finding a good initialization point and yields improved results. The performance of our method is demonstrated on benchmark problems.

1. Introduction

The Cascade Correlation (CC) learning architecture [1] as a constructive algorithm for neural networks stands out for solving hard classification problems. The most characteristic feature is the dynamic creation of hidden units without guessing the size and topology of the network in advance, which renders the algorithm much fast during the learning process compared to the back-propagation algorithm. However, it also endures deep structure caused by many one-unit hidden layers. Several researchers [2,3] have begun to seek robust methods for improving the performance of the CC network. Following the same spirit we have attempted an approach which combines the advantages of CC in training neural network with those of Genetic Algorithm (GA) [4] in optimizing the initialization.

* Corresponding author.

[1] This work is supported by National Nature Science Foundation of China.

0020-0255/98/$19.00
PII: S0020-0255(97)10071-8

The GA has been applied in connectionist learning both to optimize the weights of a fixed size network and to find the topology of the network. However, GA is not always capable of finding the best solution. Problem oriented solutions will outperform GA with respect to convergence speed and probably also with respect to the accuracy of the result. Therefore, in general optimization problems GA can be applied as a first "smart" guess. This solution can then be used as an initialization for a classical, problem oriented search algorithm. This led us to use the GA to find a good initialization point. The GA should be considered as a pre-training algorithm because most of the time the solution provided by this algorithm is close but not equal to the global optimum. To achieve a more accurate learning, a gradient descent algorithm is necessary.

2. The model

In our approach, we use genetic algorithm in conjunction with the CC learning architecture. GA is first applied over all possible sets of weights in the cascade correlation learning architecture and then the gradient descent technique (for instance Quickprop [6]) is applied to converge to a solution. This technique can automatically construct the architecture of a neural network to give it a suitable size for a specific problem and is free of the competing conventions problem [7]. An overall outline of the proposed method is as follows.

1. *Initialize the network.* Set up initial CC architecture.
2. *Train output layer.* The output layer weights are optimized by the genetic algorithm using populations of chromosomes on which the weights of the output layer are encoded. If the error cannot be reduced significantly in a few generations or the timeout (i.e. the maximum number of generations allowed) has been reached, then use Quickprop to adjust the weights of the output layer. If the learning is completed following a predefined criterion, then stop; else if the error could not be reduced significantly in a few consecutive epochs or the timeout has been reached, then go to the next step.
3. *Initialize candidate units.* Create and initialize a random population sized to fit the problem. Each string in the population represents weights linking a candidate unit to input units, all pre-existing hidden units and bias unit.
4. *Train candidate units.* Perform genetic search in weight space of candidate unit and use Quickprop to adjust weights of candidate unit so as to maximize the correlation between activation of the candidate unit and the error of the network. If the correlation cannot be improved significantly in a few consecutive epochs or the timeout has been reached, then go to the next step.
5. *Insert a new hidden unit.* Select the candidate unit with the highest correlation value and insert it in the network as a new hidden unit. Now freeze

·its incoming weights and initialize the newly established outgoing weights. Go to the Step 2.

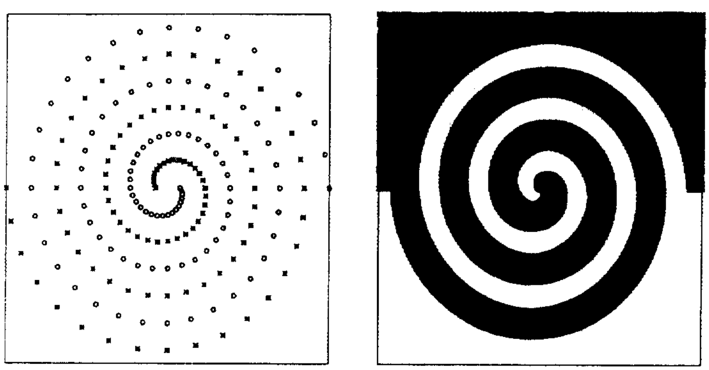
The new feature of this method as opposed to the normal CC algorithm is in the use of GA to find the initial weights for the network before applying Quickprop. As will be discussed in Section 3, it results in a more efficient convergence of the network.

3. Results

3.1. The two spirals problem

The two spirals problem [5] was chosen as the primary benchmark for this study because it is an extremely hard problem for algorithms of the back-propagation family to solve. The goal of this problem is to learn to discriminate between two sets of training points which lie on two distinct spirals in the x–y plane. These spirals coil three times around the origin and around one another, as shown in Fig. 1(a). Fig. 1(b) shows two spiral regions which can be used to address generalization. Here the generalization performance is defined as the ratio of the number of points that are correctly classified to the size of the test sample. Experimental results of the genetic CC compared to those for the regular CC are given in Table 1.

It is evident from Table 1 that the performance of the CC algorithm is improved by using GA. The effect of increasing the number of hidden nodes, one by one from 1 to 20, on the classification date produced from the two spirals by our network is depicted in Fig. 2. The total sum squared error, averaged over



(a) (b)

Fig. 1. The two spirals problem, training set in (a), testing set in (b).

Table 1
Performance of a genetic CC compared to that of a regular CC

	No. of hidden units			Runs	Performance (%)	STD [a]
	Max	Min	Average			
Genetic CC	23	15	20	30	75.5	1.8
Regular CC	43	13	26	30	72.4	5.6

[a] STD stands for standard deviation.

30 experiments, is shown in Fig. 3, as a function of the number of hidden units in both genetic and regular CC algorithm.

3.2. 8-Input parity

Since parity has been a popular benchmark among other researches, we tested our algorithm on 8-input parity problem. As a test of generalization, we ran 30 trials of genetic CC and regular CC algorithm on this problem, training on 50% of the 256 patterns and testing on the rest. The results are given in Table 2, which clearly show the superiority of the proposed algorithm.

4. Conclusions

In this paper we have addressed the issue of improving the performance of the CC algorithm by supplementing it with a GA. Our experiments demon-

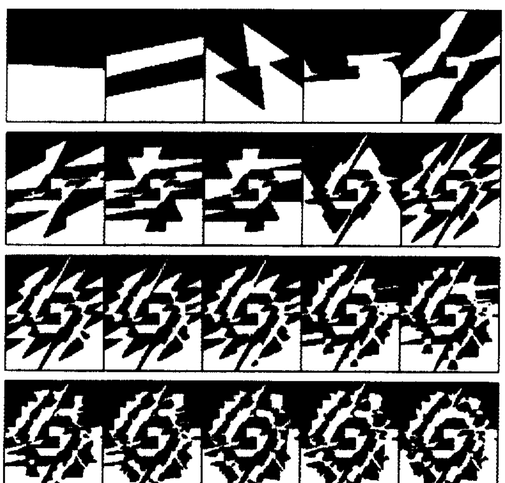


Fig. 2. Evolution of the network in the two spirals problem.

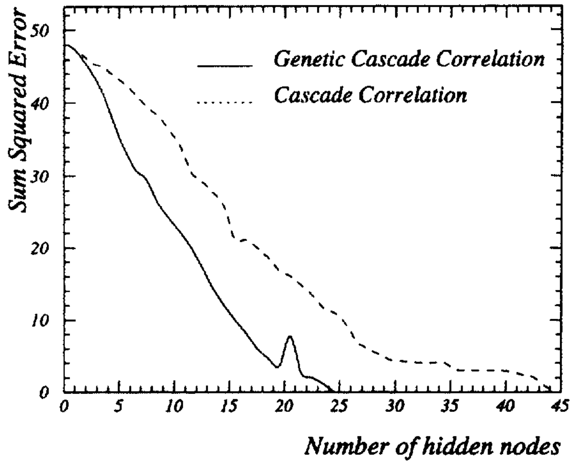


Fig. 3. Sum squared error vs. average number of hidden units for the two spirals problem.

Table 2
Performance of a genetic CC compared to that of a regular CC

	Hidden units	Test errors			
		Max	Min	Average	STD
Genetic CC	4	9	3	6	3.7
Regular CC	4	18	3	11	7.6

strated that the network obtained with our technique is of small size and has comparable performance with the standard CC algorithm. Although it is not always easy to assess the additional cost of applying GA as opposed to Quickprop, it is expected that training time will be increased. However the improvement of the performance may make it worthwhile. It is also worth noting that, some successful applications with our algorithm have been demonstrated both in high energy physics experiment [8] and in biomedical signal processing [9].

References

[1] S.E. Fahlman, C. Lebiere, The Cascade Correlation Learning Architecture. Technical Report CMU-CS-90-100, School of Computer Science, Carnegie Mellon University, 1990.

[2] D.S. Phatak, I. Koren, Connectivity and performance tradeoffs in the cascade correlation learning architecture, IEEE Transactions on Neural Networks 5 (6) (1994) 930–935.

[3] I.G. Smotroff, D.H. Friedman, D. Connoly, Self organizing modular neural networks, in: International Joint Conference on Neural Networks, Seattle, 1991, pp. 187–192.

[4] D.E. Goldberg, Genetic Algorithm in Search, Optimizing, and Machine Learning, Addison-Wesley, Reading, MA, 1989.

[5] K.J. Lang, M.J. Witbrock, Learning to tell two spirals apart, in: Proceedings of the 1988 Connectionist Models Summer School, Morgan-Kaufmann, 1988, pp. 52–59.

[6] S.E. Fahlman, An empirical study of learning speed in back-propagation networks, Technical Report CMU-CS-88-162, School of Computer Science, Carnegie Mellon University, 1988.

[7] D. Whitley, T. Starkweather, C. Bogart, Genetic algorithms and neural networks: Optimizing connections and connectivity, Parallel Computing 14 (1990) 347–361.

[8] H.L. Liang et al., Classification of the primary cosmic ray composition with neural networks, High Energy Physics and Nuclear Physics 21 (3) (1997) 205–210.

[9] Z. Lin et al., Application of combined genetic algorithm with cascade correlation to diagnosis of delayed gastric emptying from electrogastrograms, Proceedings of the Ninth Annual Conference IEEE Eng. Med. Biol. Soc., Chicago, 1997 (in press).