IMPROVED CUCKOO SEARCH ALGORITHM FOR FEEDFORWARD NEURAL NETWORK TRAINING

Ehsan Valian, Shahram Mohanna and Saeed Tavakoli

Faculty of Electrical and Computer Engineering, University of Sistan and Baluchestan, Iran

ehsanvalian@gmail.com,mohanna@hamoon.usb.ac.ir,tavakoli@ece.usb.ac.ir

ABSTRACT

The cuckoo search algorithm is a recently developed meta-heuristic optimization algorithm, which is suitable for solving optimization problems. To enhance the accuracy and convergence rate of this algorithm, an improved cuckoo search algorithm is proposed in this paper. Normally, the parameters of the cuckoo search are kept constant. This may lead to decreasing the efficiency of the algorithm. To cope with this issue, a proper strategy for tuning the cuckoo search parameters is presented. Then, it is employed for training feedforward neural networks for two benchmark classification problems. Finally, the performance of the proposed algorithm is compared with that of the standard cuckoo search. Simulation results demonstrate the effectiveness of the proposed algorithm.

KEYWORDS

Classification, cuckoo search algorithm, feedforward neural network, optimization

1. INTRODUCTION

A typical artificial neural network (ANN) has two types of basic components. They are neurons which are processing elements and links which are interconnections between neurons. Each link has a weighting parameter. Each neuron receives stimulus from other neurons, processes the information, and produces an output. Neurons are categorized to input, output and hidden neurons. The first and the last layers are called input and output layers, respectively, and the remaining layers are called hidden layers [1].

Consider N_k as the number of neurons in the k^{th} layer. Let w_{ij}^{k} , represents the weight of the link between the j^{th} neuron of the $(k-1)^{th}$ layer and the i^{th} neuron of the k^{th} layer. Suppose w_{i0}^{k} is an additional weighting parameter for each neuron, representing the bias for the i^{th} neuron of the k^{th} layer. The weighting parameters are initialized before training the neural network. During training, they are updated iteratively in a systematic manner [2]. Once the neural network training is completed, the weighting parameters remain fixed throughout the usage of the neural network as a model.

The process of training an ANN is to adjust weights and biases. The back-propagation (BP) learning has become the most popular method to train feedforward ANNs in many domains [3]. However, one limitation of this technique, which is a gradient-descent technique, is that it requires a differentiable neuron transfer function. Also, as neural networks generate complex error surfaces with multiple local minima, the BP tends to converging into local minima instead of a global minimum [4].

In recent years, many improved learning algorithms have been proposed to overcome the handicaps of gradient-based techniques. These algorithms include a direct optimization method using a polytope algorithm [5], a global search technique such as evolutionary programming [6],

DOI : 10.5121/ijaia.2011.2304

and genetic algorithm (GA) [7]. The standard gradient-descent BP is not trajectory-driven, but population driven. However, the improved learning algorithms have explorative search features. Consequently, these methods are expected to avoid local minima frequently by promoting exploration of the search space.

Because of computational drawbacks of conventional numerical methods in solving complex optimization problems, researchers may have to rely on meta-heuristic algorithms. Over the last decades, many meta-heuristic algorithms have been successfully applied to various engineering optimization problems [8-14]. For most complicated real-world optimization problems, they have provided better solutions in comparison with conventional numerical methods.

To imitate natural phenomena, most meta-heuristic algorithms combine rules and randomness. These phenomena include the biological evolutionary processes, such as genetic algorithm (GA) [15-16], evolutionary algorithm [17-19] and differential evolution (DE) [20]), animal behaviour, such as particle swarm optimization (PSO) [21], tabu search [22] and ant colony algorithm (ACA) [23]), as well as physical annealing processes, such as simulated annealing (SA) [24].

The Cuckoo Search (CS) developed in 2009 by Yang and Deb [25-26], is a new meta-heuristic algorithm imitating animal behaviour. The optimal solutions obtained by the CS are far better than the best solutions obtained by efficient particle swarm optimizers and genetic algorithms [25]. This paper develops an Improved Cuckoo Search (ICS) algorithm for unconstrained optimization problems. To enhance accuracy and convergence rate of the CS, the ICS employs an improved method for generating new solution vectors.

The paper is organized as follows. In section 2, the cuckoo search algorithm is briefly reviewed. The improved cuckoo search algorithm is presented in section 3. In section 4, this algorithm is employed to train a feedforward neural network. We end this paper with some conclusions in section 5.

2. CUCKOO SEARCH ALGORITHM

To describe the CS more clearly, the breed behavior of certain cuckoo species is briefly reviewed.

2.1 Cuckoo Breeding Behaviour

The CS was inspired by the obligate brood parasitism of some cuckoo species by laying their eggs in the nests of host birds. Some cuckoos have evolved in such a way that female parasitic cuckoos can imitate the colours and patterns of the eggs of a few chosen host species. This reduces the probability of the eggs being abandoned and, therefore, increases their re-productivity [27]. It is worth mentioning that several host birds engage direct conflict with intruding cuckoos. In this case, if host birds discover the eggs are not their own, they will either throw them away or simply abandon their nests and build new ones, elsewhere.

Parasitic cuckoos often choose a nest where the host bird just laid its own eggs. In general, the cuckoo eggs hatch slightly earlier than their host eggs. Once the first cuckoo chick is hatched, his first instinct action is to evict the host eggs by blindly propelling the eggs out of the nest. This action results in increasing the cuckoo chick's share of food provided by its host bird [27]. Moreover, studies show that a cuckoo chick can imitate the call of host chicks to gain access to more feeding opportunity.

The CS models such breeding behaviour and, thus, can be applied to various optimization problems. Yang and Deb [25-26], discovered that the performance of the CS can be improved by using Lévy Flights instead of simple random walk.

2.2 Lévy Flights

In nature, animals search for food in a random or quasi-random manner. Generally, the foraging path of an animal is effectively a random walk because the next move is based on both the current location/state and the transition probability to the next location. The chosen direction implicitly depends on a probability, which can be modeled mathematically. Various studies have shown that the flight behavior of many animals and insects demonstrates the typical characteristics of Lévy flights [28]. A Lévy flight is a random walk in which the step-lengths are distributed according to a heavy-tailed probability distribution. After a large number of steps, the distance from the origin of the random walk tends to a stable distribution.

2.3 Cuckoo Search Implementation

Each egg in a nest represents a solution, and a cuckoo egg represents a new solution. The aim is to employ the new and potentially better solutions (cuckoos) to replace not-so-good solutions in the nests. In the simplest form, each nest has one egg. The algorithm can be extended to more complicated cases in which each nest has multiple eggs representing a set of solutions [25-26]. The CS is based on three idealized rules:

• Each cuckoo lays one egg at a time, and dumps it in a randomly chosen nest;

• The best nests with high quality of eggs (solutions) will carry over to the next generations;

• The number of available host nests is fixed, and a host can discover an alien egg with probability $p_a \in [0,1]$. In this case, the host bird can either throw the egg away or abandon the nest to build a completely new nest in a new location [25].

For simplicity, the last assumption can be approximated by a fraction p_a of the n nests being replaced by new nests, having new random solutions. For a maximization problem, the quality or fitness of a solution can simply be proportional to the objective function. Other forms of fitness can be defined in a similar way to the fitness function in genetic algorithms [25].

Based on the above-mentioned rules, the basic steps of the CS can be summarized as the pseudo code, as follows [25]:

```
begin
   Objective function $f(x)$,   $x = (x_1,.....,x_d)^T$
   Generate initial population of
        n host nests $x_i$ $(i = 1,2,...,n)$
   While $(t < MaxGeneration)$ or (stop criterion)
      Get a cuckoo randomly by Lévy flights
        evaluate its quality / fitness $F_i$
      Choose a nest among n (say, j) randomly
      if $(F_i > F_j)$,
           replace j by the new solution;
      end if
      A fraction $(p_a)$ of worse nests
            are abandoned and new ones are built;
      Keep the best solutions
            (or nests with quality solutions);
      Rank the solutions and find the current best
   end while
   Postprocess results and visualization
end
```

when generating new solutions $x_i(t+1)$ for the i^{th} cuckoo, the following Lévy flight is performed

$$x_i(t+1) = x_i(t) + \alpha \oplus \text{Lévy}(\lambda) \tag{1}$$

where $\alpha > 0$ is the step size, which should be related to the scale of the problem of interest. The product $\oplus$ means entry-wise multiplications [26]. In this research work, we consider a Lévy flight in which the step-lengths are distributed according to the following probability distribution

$$\text{Lévy } u = t^{-\lambda}, 1 < \lambda \le 3 \tag{2}$$

which has an infinite variance. Here, the consecutive jumps/steps of a cuckoo essentially form a random walk process which obeys a power-law step-length distribution with a heavy tail .

It is worth pointing out that, in the real world, if a cuckoo's egg is very similar to a host's eggs, then this cuckoo's egg is less likely to be discovered, thus the fitness should be related to the difference in solutions. Therefore, it is a good idea to do a random walk in a biased way with some random step sizes [25].

3. Improved Cuckoo Search

The parameters p_a, λ and α introduced in the CS help the algorithm to find globally and locally improved solutions, respectively. The parameters p_a and α are very important parameters in fine-tuning of solution vectors, and can be potentially used in adjusting convergence rate of algorithm. The traditional CS algorithm uses fixed value for both p_a and α. These values are set in the initialization step and cannot be changed during new generations. The main drawback of this method appears in the number of iterations to find an optimal solution. If the value of p_a is small and the value of α is large, the performance of the algorithm will be poor and leads to considerable increase in number of iterations. If the value of p_a is large and the value of α is small, the speed of convergence is high but it may be unable to find the best solutions.

The key difference between the ICS and CS is in the way of adjusting p_a and α. To improve the performance of the CS algorithm and eliminate the drawbacks lies with fixed values of p_a and α, the ICS algorithm uses variables p_a and α .In the early generations, the values of p_a and α must be big enough to enforce the algorithm to increase the diversity of solution vectors. However, these values should be decreased in final generations to result in a better fine-tuning of solution vectors. The values of p_a and α are dynamically changed with the number of generation and expressed in Equations 3-5, where NI and gn are the number of total iterations and the current iteration, respectively.

$$P_a(gn) = P_{a\,max} - \frac{gn}{NI}\left(P_{a\,max} - P_{a\,min}\right) \tag{3}$$

$$\alpha(\mathrm{gn}) = \alpha_{\max} \exp(\mathrm{c.gn}) \qquad (4)$$

$$\mathrm{c} = \frac{1}{\mathrm{NI}} \mathrm{Ln}\left(\frac{\alpha_{\min}}{\alpha_{\max}}\right) \qquad (5)$$

4. Simulation Results, Analysis and Discussion

When ANN training is initiated, the iterative process of presenting the training patterns of the dataset to the network's input continues until a given termination condition is satisfied. This usually happens based on a criterion indicating that the current achieved solution is presumably good enough to stop training. For instance, one common termination criterion in the BP is the difference between the current value of sum of squared errors (SSE) and that obtained in the previous iteration [29].

Figure 1 illustrates a small scale sample ANN. Each vector represents a complete set of ANN weights including biases. The objective function is to minimize SSE [30]. The squared difference between the target output and actual output determines the amount of error. This is represented by for each pattern and each output unit as shown in Figure 1. Since the values of ANN weights are usually within the same range we simplify the CS and ICS model by setting.

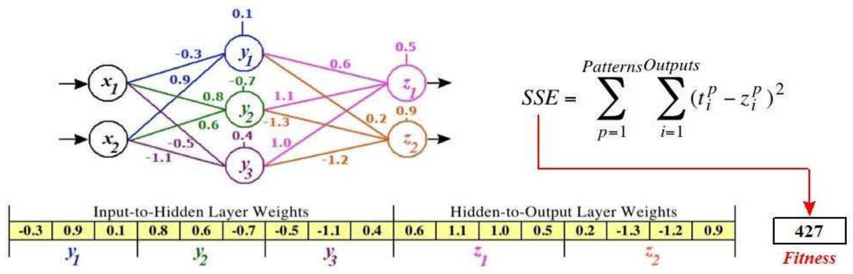

Figure 1. A small scale sample ANN [31]

In this study, two benchmark classification problems, which are classification of Iris, and breast cancer, are studied. According to four properties of one flower, the categories of Iris are identified in the Iris problem. This problem consists of 150 examples; all of them can be classified into three categories whilst each category accounts for one third of the examples. 105 examples are used for training and the rest are used for testing. In the breast cancer problem, we correctly diagnose breast lumps as either benign or malignant based on data from examination of cells. This problem consists of 699 examples. Each example includes 9 inputs and 2 outputs. 500 examples are used for training and the rest are used for testing. To evaluate the performance of the proposed algorithm, we compare its computational results with those provided by CS, and ICS. For the CS and ICS algorithms, the number of iterations is 750 for the first benchmark problem and 500 for the second one .

For the CS algorithm, the number of populations is 10, whereas other algorithm parameters are $\alpha = 0.25$, $p_a = 0.1$ and $\lambda = 1.5$. The parameters of the ICS are $p_a(\min) = 0.05$, $p_a(\max) = 0.5$, $\alpha(\min) = 0.01$, $\alpha(\max) = 0.5$ and $\lambda = 1.5$.

For both benchmark problems, the number of hidden neurons for all algorithms is 8. After 20 iterations, the results of classification are demonstrated in Table 1. In this table, the parameter 'Std' represents the standard deviation. The results show that the CS performs better than the

ICS in terms of classification accuracy and computational expenses. Furthermore, Table 1 shows that the ICS has a better overall performance in almost all criteria.

Table1. Simulation results for benchmark problems

Algorithm		**Breast cancer dataset**			**Iris dataset**		
		SSE	**Training accuracy**	**Testing accuracy**	**SSE**	**Training accuracy**	**Testing accuracy**
CS	Min.	43.66	95.80	92.46	29.94	89.52	80.00
	Average	52.99	96.91	96.13	32.53	92.14	95.89
	Max.	65.79	**98.40**	97.99	35.48	97.14	**100**
	Std.	5.31	0.61	1.33	**1.85**	4.00	5.36
ICS	Min.	**37.74**	**96.20**	**94.97**	**25.17**	**91.43**	**97.78**
	Average	**41.71**	**96.98**	**96.86**	**29.38**	**94.10**	**98.45**
	Max.	**47.23**	97.60	**98.49**	**32.29**	**97.14**	**100**
	Std.	**2.30**	**0.40**	**0.91**	2.18	**1.79**	**1.63**

5. CONCLUSIONS

In this paper, an improved cuckoo search algorithm enhancing the accuracy and convergence rate of the standard cuckoo search algorithm was proposed. Unlike the standard cuckoo search in which the parameters are kept constant, the ICS parameters are tuned. This results in increasing the efficiency of the algorithm. To evaluate the performance of the proposed algorithm, it is employed for training feedforward neural networks for two benchmark classification problems. Comparing the results provided by the proposed algorithm with those given by the CS demonstrates the effectiveness of the improved cuckoo search.

REFERENCES

REFERENCES

[1] Zhang QJ , Gupta KC, (2003) Neural Networks for RF and Microwave Design- From Theory to Practice, *IEEE Transactions on Microwave Theory and Techniques*, Vol.51, No .4 ,pp 1339-1350.

[2] Wang F, et al, (1999) Neural network structures and training algorithms for microwave applications, *International Journal of RF Microwave Computer-Aided Engineering*, Vol. 9,pp 216-240.

[3] Chronopoulos AT, Sarangapani J, (2002) A distributed discrete time neural network architecture for pattern allocation and control, *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS'02), Florida, USA* pp,204-211.

[4] Gupta JND, Sexton RS, (1999) Comparing backpropagation with a genetic algorithm for neural network training, *The International Journal of Management Science*, Vol. 27,pp 679-684.

[5] Curry B, Morgan, P, (1997) Neural Networks: A Need for Caution, OMEGA. *International Journal of Management Sciences*, Vol. 25, pp 123–133.

[6] Salchenberger L, et al, (1992) Neural Networks: A New Tool for Predicting Thrift Failures. *Decision Sciences*, Vol.23, pp 899–916.

[7] Sexton RS, Dorsey RE, Johnson JD, (1998) Toward Global Optimization of Neural Networks: A Comparison of the Genetic Algorithm and Backpropagation. *Decision Support Systems*, Vol. 22, pp 171–186.

[8] Sanchis J, Martanez MA, Blasco X, (2008) Integrated multiobjective optimization and a priori preferences using genetic algorithms, *Information Sciences*, Vol. 178. No. 4, pp 931–951.

[9] Zhang JJ, Zhang YH, Gao RZ, (2006) Genetic algorithms for optimal design of vehicle suspensions, in: *IEEE International Conference on Engineering of Intelligent Systems*, pp, 1–6.

[10] Qing AY, (2006) Dynamic differential evolution strategy and applications in electromagnetic inverse scattering problems, *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 44, No. 1, pp 116–125.

[11] Sickel JHV, Lee KY, Heo JS, (2007) Differential evolution and its applications to power plant control, in: *International Conference on Intelligent Systems Applications to Power Systems*, ISAP, pp, 1–6.

[12] Marinakis Y, Marinaki M, Dounias G, (2008) Particle swarm optimization for pap- smear diagnosis, *Expert Systems with Applications*, Vol. 35,No. 4,1645–1656.

[13] Ghosh S, Biswas S, Sarkar D, Sarkar PP, (2010), Mining Frequent Itemsets Using Genetic Algorithm, *International Journal of Artificial Intelligence & Applications (IJAIA),* Vol.1, No.4, 133-143.

[14] Banerjee S, Tithi Dey M , Dutta S, (2010), Difficult Channel Generation Using Genetic Algorithm, *International Journal of Artificial Intelligence & Applications (IJAIA)*, Vol.1, No.4, 145-157.

[15] Holland JH, (1975) *Adaptation in natural and artificial systems*, University of Michigan Press, Ann Arbor, MI.

[16] Goldberg DE, (1989) *Genetic algorithms in search, optimization and machine learning*, Addison Wesley, Boston, MA.

[17] Fogel LJ, Owens AJ, Walsh MJ, (1996) *Artificial intelligence through simulated evolution*, John Wiley, Chichester, UK.

[18] De Jong K, (1975) *Analysis of the behavior of a class of genetic adaptive systems*, Ph.D. Thesis, University of Michigan, Ann Arbor, MI.

[19] Koza JR, (1990) *Genetic programming: a paradigm for genetically breeding populations of computer programs to solve problems*, Rep. No. STAN-CS-90-1314, Stanford University, CA.

[20] Storn R, (1996) Differential evolution design of an IIR-filter, in: *IEEE International Conference on Evolutionary Computation, Nagoya*, pp 268–273.

[21] Kennedy J, Eberhart RC, (1995) Particle swarm optimization, in: *Proceedings of IEEE International Conference on Neural Networks*, pp 1942–1948.

[22] Glover F, (1977) Heuristic for integer programming using surrogate constraints. *Decision Sci*, Vol. 8, No. 1, pp 156–166.

[23] Dorigo M, Maniezzo V, Golomi A, (1996) Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on SMC*, Vol. 26, No. 1, pp 29–41.

[24] Kirkpatrick S, Gelatt C, Vecchi M, (1983) Optimization by simulated annealing. *Science*, Vol. 220, pp 671–680.

[25] Yang XS, Deb S, (2009) Cuckoo search via Lévy flights, *Proceeings of World Congress on Nature & Biologically Inspired Computing, India*, pp 210-214.

[26] Yang XS, Deb S, (2010) Engineering Optimisation by Cuckoo Search, Int. J. *Mathematical Modelling and Numerical Optimisation*. Vol. 1, No. 4, pp 330–343.

[27] Payne RB, Sorenson MD, Klitz K, (2005) *The Cuckoos*, Oxford University Press.

[28] Brown C, Liebovitch LS, Glendon R, (2007) Lévy flights in Dobe Ju/hoansi foraging patterns, *Human Ecol*. Vol. 35, pp 129-138.

[29] Fausett L. (1994) *Fundamentals of Neural Networks Architectures, Algorithms, and Applications*. New Jersey: Prentice Hall.

[30] Hassoun MH (1995) Fundamentals of Artificial Neural Networks. Massachusetts: MIT Press, Cambridge.

[31] Kattan A, Abdullah R, Abdul-Salam R, (2010) Harmony search based supervised training of artificial neural networks, *International Conference on Intelligent Systems, Modelling and Simulation*, pp 105-110.

Authors

Ehsan Valian received his BSc degree in electrical engineering from Razi University of Kermanshah, Iran in 2007. Currently, he is an MSc student at University of Sistan and Baluchestan, Iran. His areas of research include evolutionary algorithms and numerical optimization.

Shahram Mohanna received his BSc and MSc degrees in electrical engineering from University of Sistan and Baluchestan, Iran and University of Shiraz, Iran in 1990 and 1994, respectively. He then joined University of Sistan and Baluchestan, Iran. In 2005, he obtained his PhD degree in electrical engineering from University of Manchester, England. As an assistant professor at University of Sistan and Baluchestan, his areas of research include design of microwave circuits, antenna design and applied electromagnetic. Dr. Mohanna has served as a reviewer for several journals and a number of conferences.

Saeed Tavakoli received his BSc and MSc degrees in electrical engineering from Ferdowsi University of Mashhad, Iran in 1991 and 1995, respectively. In 1995, he joined University of Sistan and Baluchestan, Iran. He earned his PhD degree in electrical engineering from University of Sheffield, England in 2005. As an assistant professor at University of Sistan and Baluchestan, his research interests are evolutionary multi-objective optimization, control of time delay systems, PID control design, robust control, and jet engine control. Dr. Tavakoli has served as a reviewer for several journals including IEEE Transactions on Automatic Control, IEEE Transactions on Control Systems Technology, IET Control Theory & Applications, and a number of international conferences