БЪЛГАРСКА АКАДЕМИЯ НА НАУКИТЕ • BULGARIAN ACADEMY OF SCIENCES

ПРОБЛЕМИ НА ТЕХНИЧЕСКАТА КИБЕРНЕТИКА И РОБОТИКАТА, **70**
PROBLEMS OF ENGINEERING CYBERNETICS AND ROBOTICS, **70**

София • 2018 • Sofia

An Approach to Training Artificial Neural Networks with Genetic Algorithms

Chavdar Korsemov, Stefan Koynov, Hristo Toshev

Institute of Information and Communication Technologies 1113 Sofia
Emails: ***chkorsemov@mail.bg***, ***slk@iinf.bas.bg***, ***hr_toshev@mail.bg***

Abstract: *The paper introduces characteristic features of genetic algorithms (GA) that distinguish them from traditional optimization methods and procedures for searching and learning at the stages of preliminary preparation for solving practical problems with GA. Genetic approaches for learning of artificial neural networks are proposed, evolutionary approaches with penalty functions included.*

Keywords: *Genetic algorithms, artificial neural networks, penalty functions; artificial neural networks learning*

1. Introduction

At the end of the 50s scientists from different countries [1], [2], [3] explored in detail evolutionary systems and independently came to the conclusion that they could use the theory of evolution as an instrument for optimization in the process of solution for problems of different nature with the main goal creating a population of eventual solutions using some of the most characteristic peculiarities of nature – heredity, changeability, selection and so on

Genetic algorithms are a method for search based on the selection of the best species in the population in analogy to the theory of evolution of Ch. Darwin.

Their origin is based on the model of biological evolution and the methods of random search. From the bibliographical sources [1], [2], [3], [5] it is evident that the random search appeared as a realization of the simplest evolutionary model when the random mutations are modeled during random phases of searching the optimal solution and the selection is modeled as “removal” of the unfeasible versions.

From the point of view of the information change, the evolutionary search is a sequential transformation of a single fuzzy (imprecise) set of some solutions into another one. The transform itself can be named a searching algorithm or a genetic algorithm (GA).The GA is not simply a random search, but an efficient usage of information in the evolutionary process [5], [7].

The main goal of GA-s is twofold:

- abstract and formal explanation of the adaptation processes in evolutionary systems;
- modeling natural evolutionary processes for efficient solution of determined class of optimization and other problems.

During the last years a new paradigm is applied to solve optimization problems GA-based and modifications of GA. GA realize searching a balance between efficiency and quality of solutions at the expense of selecting the strongest alternative solution from undetermined and fuzzy solutions [7].

2.Characteristic Features of the Genetic Algorithms

We can show more precisely the real necessity to use GA- if we try to present some of their most characteristic peculiarities that distinguish them from traditional optimization methods for search and learning [3], [6], [7]:

- GA operate basically not with the problem parameters, but with a coded set of parameters;
- they realize the search not by improving the solution, but by using several alternatives of the defined set of solutions;
- they use a goal function, not different increments for evaluation of quality of the accepted solutions;
- the accepted rules for analysis of the optimization problem are not determined, but probabilistic.

In operational mode, the GA chooses a set of natural parameters of the optimization problem and it codes them into a sequence with a finite length in some alphabet. The GA operates until a given number of iterations is fulfilled (algorithmic iterations) or during some iteration a solution with a definite quality is obtained or a local optimum is found, i.e. a premature convergence is observed and it is impossible to find an exit from this state [4].

GA-s provide a series of advantages for solving problems in practice. One advantage is the adaptation to the changing environment. In real life the problem that is postulated for solution can undergo vast modifications during the process of its solving. The usage of traditional methods requires that all calculations must be performed from scratch and this leads to big losses of machine time. In the case of

evolutionary approaches the population can be analyzed, supplemented and modified in accordance with the modified conditions. For this reason the complete selection is an option, not obligation.

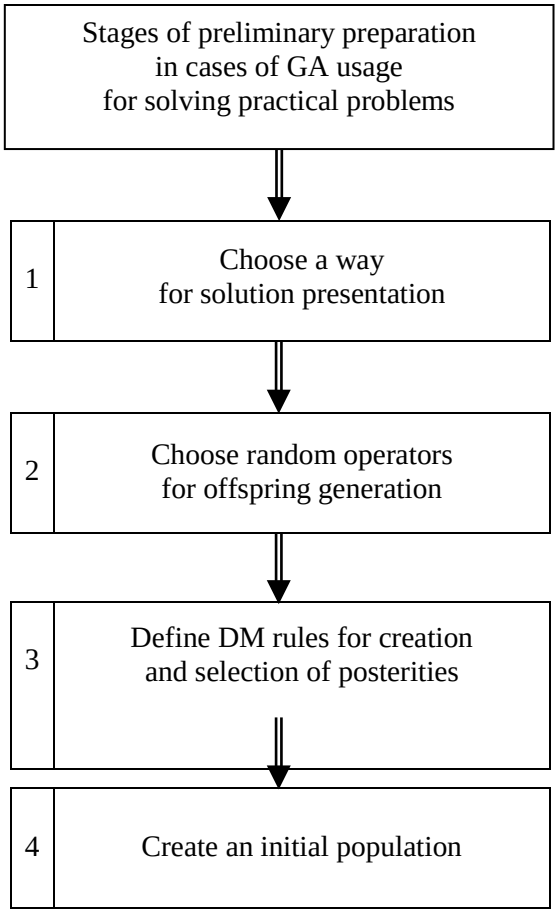

Fig. 1 Stages of preliminary preparation

Another important peculiarity in the process of solving practical problems supported by GAs is the necessity to execute some preliminary stages (Fig. 1):

- *Choose a way for solution presentation* (stage 1 in Fig. 1). The chosen structure must allow coding of all possible solutions and it also must admit and ensure their estimate [4]. It has been proven that no ideal structures for presentation exist, therefore the creation of a good structure requires the ensurance of possibilities for application of heuristic structures, analysis and selection;

- *Choose random operators for posterity generation* (stage 2 in Fig. 1). This problem is rather sophisticated due to the existence of a big number of possibilities. Besides the usage of the two basic types of reproduction, sexual and asexual

(cloning), it is possible to use also operations that do not exist in real nature. For example, the usage of material from three and even more parents that is accompanied by voting for the choice of parents. There are no limitations for the usage of different operators and also it is senseless to copy laws of nature and their limitations mechanically and blindly [1], [2,], [4], [5];

- *Define DM rules for creation and selection of posterities* (stage 3 in Fig. 1). Just like the case of choosing random operators, here a set of ways to perform the selection also exists. The simplest choice is the case to accept only the best solutions and to ignore the rest of the decisions. Very often this rule turns out to be little effective and that good decisions may come also from bad ones, i.e. not just from the best ones. Therefore this leads to the logical assumption that the possibility for choice of the best decision must be the most important principle at this stage;
- *Create an initial population* (stage 4 in Fig. 1). If there is a lack of knowledge for the considered problem then solutions are accessible according to some random choice from the set of possible solutions. This points to a process of generation of random problems when every single problem itself is a definite solution. On the other hand during the process of creation of the primary population it is possible to use data received during the experiment of solving the same problem with other GA-s. If these solutions are really valuable then they will outlast and produce a posterity if they do not perish together with other weak individuals meanwhile [2], [5].

3. Genetic approaches in ANN learning

In the ANN-learning phase [8], [9], [13-17] at an equal predefined number of iterations control copies of the network parameters, the input and output vectors are made. So a series of successive temporal generations of the goal ANN is obtained which is presented in the form of a multilayer model; the interlayer connections are modeled with penalty functions. The definitions which follow below define different types of errors for the artificial neural networks (ANN) learning process. They link the ANN with the genetic and evolutionary approaches to the problem of the ANN learning.

DEFINITION 1: Control total error $E_g^{(i)}$ is the ratio of the erroneous outputs (*misses*) $m^{(i)}$ to the correct outputs (*hits*) $h^{(i)}$ for the i-th ANN state generation in the learning process, i.e. $E_g^{(i)} = m^{(i)} / h^{(i)}$.

NOTE. The value of $E_g^{(i)}$ usually does not exceed 10^{-2}.

DEFINITION 2: Accumulated total error E_g^i for the last i generations is the sum of the control total errors $E_g^{(i)}$ for the last i generations, i.e. $E_g^i = \sum_i E_g^{(i)}$.

The accumulated total error E_g^i serves both as an indicator of the ANN parameters state evolution and implicitly for the expected number of state offsprings of the being trained ANN.

DEFINITION 3: Error of the learned ANN is any error E_t which does not exceed the allowed operational error of the already trained ANN, i.e. $E_t < E_d$.

COROLLARY 1. The error of the already learned ANN E_t is less than the maximal value of the control total error $\left(E_g^{(i)}\right)_{\max}$ for the last I state generations, i.e. $E_t < \left(E_g^{(i)}\right)_{\max}$.

The two theorems below evaluate the number of iterations necessary to learn an ANN and an interval of the estimate for the necessary ANN state offsprings is proposed.

THEOREM 1: Let the abscissa axis is the number of iterations to change the weights between two successive states of the being learned ANN. Then the iteration interval between any two successive states during the training of a single ANN $t_g^{(i)}$ must be less than the product of the overall added in the successive state generations neurons $N^{a(i)}$ by a factor of the number of the weight corrections for the these states $n_g^{(i)}$: $N^{a(i)} \cdot n_g^{(i)} > t_g^{(i)}$.

Proof: The worst case is when the new state generation has just *one* new added neuron. If this is the case of learning for the next successive generations then the iteration interval for adding new neurons will be estimated based on the statistics of the averagely added neurons multiplied by the number of the average weight corrections per one added neuron.

Theorem 2: The maximal number of state generations for the learning of an ANN I is equal to the product of the overall added in the next state generation neurons $N^{a(i)}$ multiplied by the number of the weight corrections $n_g^{(i)}$ divided by the iteration interval between two successive states of the being learned ANN $t_g^{(i)}$: $\dfrac{N^{a(i)} \cdot n_g^{(i)}}{t_g^{(i)}} = I \geq 1.$

Proof: The proof follows from the previous Theorem 1 by dividing the number of iterations for all new added neurons $N^{a(i)} \cdot n_g^{(i)}$ and the iteration interval between two state generations of the ANN $t_g^{(i)}$

COROLLARY 2: The minimal number of state generations for learning a single ANN equals to *one:* $I_{\min} = 1$. It is obtained if the total number of the added neurons in the ANN learning process is obtained during the ANN state generation number two $(n = 2)$. It means that $I_{\min} \sim N^{a(i)} = N^{a(2)} = 1$ iff $\mathrm{n}_{\mathrm{g}}^{(2)} = t_g^{(2)}$: the iteration interval between two successive ANN state generations $t_g^{(i)}$ is adjusted equal to the current number of weight corrections $n_g^{(i)}$ for tuning the ANN to achieve $E_t < E_d$

The following below definitions link the types of convergence (or divergence) in the automation theory and their analogues in the genetic approach.

DEFINITION 4: Genetic divergence is the tendency the values of the traced parameters to deviate from their average stable states for an offspring series for the species for large values of the number of generations.

COROLLARY 3: The variance and the r.m.s. divergence increase if the genetic divergence increases.

DEFINITION 5: Genetic convergence is the tendency the values of the traced parameters to converge to their average stable states for an offspring series for the species for large values of the number of generations.

COROLLARY 4: The variance and the r.m.s. divergence decrease if the genetic divergence decreases.

DEFINITION 6: Asymptotic genetic convergence implies that the traced parameters are genetically convergent and that their values lie in a predefined small neighborhood around their average values for an offspring series for the species for large values of the number of generations.

4. Penalty functions approach in ANN learning

The approach with penalty functions admits an interpretation with multilayer models for exploring the ANN learning process; the definitions and the theorems in the previous chapter allow the evaluation of the penalty functions which comprise the multilayer model for the ANN learning process. The multilayer approach is already implied by the authors to model a serial industrial application [10].

The series of penalty functions between the separate temporal generations described with the multilayer model naturally converges to a penalty function corresponding to the already learned ANN, so the temporal series of the parameters, of the input and output vectors of the ANN are the analog to the popular mathematical series. The goal of the method is to achieve an estimate of the penalty functions of an arbitrary given sequence number of the ANN state generations by varying the number of iterations between every two state replicas of the being learned ANN on condition that the penalty functions for the first several offsprings of the state are obtained.

The ANN learning multilayer model consists of layers which correspond to the successive ANN state parameter records (offsprings). The successive layers may be numbered in two mutually opposite directions: from the periphery to the core or v.v. The default direction of numbering is based on the physical nature of the model [10]; in the ANN learning process it is the number of iterations, therefore the greater layer numbers correspond to the successive ANN state generations. Let the whole search space is denoted with S and the feasible subspaces of the solutions are denoted with F. Then the following mathematical description may be formulated for the case of the ANN training process from the point of view with penalty functions:

$$eval\left(\overline{X}\right) = f\left(\overline{X}\right) + \sum_{l=1}^{L} a_l \left[\lambda(t) \sum_{j=1}^{m} f_j^{\,2}\left(\overline{X}\right) \right]^l$$

where:

$eval(\overline{X})$ — feasible and unfeasible solutions if $\overline{X} \in \boldsymbol{F}$ is the optimal solution of the general nonlinear programming model with continuous variables;

$f(\overline{X})$ — goal function for optimization;

$\lambda(t)$ — updated every generation t in the following way [11]:

$$\lambda(t+1) = \begin{cases} (1/\beta_1) \cdot \lambda(t), \text{ if } \overline{B}(i) \in \boldsymbol{F} \text{ for all} t-k+1 \le i \le t \\ \beta_2 \cdot \lambda(t), \text{ if } \overline{B}(i) \in \boldsymbol{S} - \boldsymbol{F} \text{ for all} t-k+1 \le i \le t \\ \lambda(t), \quad \text{else} \end{cases}$$

$f_j(\overline{X})$ — constraint violation measure for the *j*-th constraint such that [12]:

$$f_j(\overline{X}) = \begin{cases} max\{0, g_j(\overline{X})\}, & \text{if } 1 \le j \le q \\ |h_j(\overline{X})|, & \text{if } q+1 \le j \le m \end{cases}$$

Here $g_j(\overline{X}) \le 0$, $j = 1,\ldots,q$ and $h_j(\overline{X}) = 0$, $j = q+1,\ldots,m$ is a set of additional constraints $m \ge 0$ the intersection of which with $\boldsymbol{S}$ defines the feasible set $\boldsymbol{F}$.

l — indicator of the constraint type with upper bound $L = \{ 2 \mid 3 \}$:

$$l = \begin{cases} 1: \textit{inside} \text{ a given } \textit{layer} \text{ (the lowest constraint level);} \\ 2: \textit{between} \text{ two } \textit{layers} \text{ inside a given multilayer model (the moddle constraint level);} \\ 3: \textit{between} \text{ two multilayer } \textit{mod els} \text{ (the highest constraint level).} \end{cases}$$

a_l — coefficient array reflecting the weights of the different constraint levels in the formula. It is adjusted heuristically.

4. Conclusions

The presented in the paper characteristic features of GA demonstrate basic differences from traditional optimization methods and procedures and in some cases they offer better possibilities for solving certain classes of optimization problems. The stages of preliminary preparation for solving practical problems with GA are very essential in the context of the general strategy for solving such problems, therefore their strict execution is crucial to obtain good solutions. Genetic approaches are offered for learning of artificial neural networks, evolutionary approaches with penalty functions included. The learning and the operation of ANN are estimated via their successive temporal generations by multilayer and other models. The goal of the method is to achieve an estimate of the penalty functions of

an arbitrary given sequence number of the ANN state generation by varying the number of iterations between every two state replicas of the being learned ANN on condition that the penalty functions for the first several offsprings of the state are obtained; an interval of the estimate for the necessary ANN state offsprings is proposed.

References

1. Holland, J. H., "Adaptation in Natural and Artificial Systems", the MIT Press, 1992.
2. Goldberg, D., "Genetic Algorithms in Search, Optimization and Machine Learning", Addison-Wesley, 1989.
3. Mitchell, M., "An Introduction to Genetic Algorithms", MA: MIT Press, 1996.
4. Koza, J. R., "Genetic Programming 2", Cambridge MA: MIT Press, 1994.
5. Emelianov, V. V., V. M. Kureichik, V. V. Kureichik, "Theory and Practice of Evolutionary Modelling", Moscow, 2003. (in Russian)
6. Karova, M., "A Review of Selection, Mutation and Recombination in Genetic Algorithms", Proceedings of the XXXVIII International Scientific Conference on Information, Communication and Energy Systems and Technologies ICEST 2003, Sofia, pp 314-317, 2003.
7. Goldberg, D., "Web Courses", http://www.engr.uiuc.edu/OCCE, 2000.
8. Haykin, S. Neural networks. Englewood Cliffs, Macmillan Publishing Co., 696 p., 1994.
9. Kasabov, N., "Foundations of Neural Networks, Fuzzy Systems and Knowledge Engineering", the MIT Press, 1996.
10. Koynov, S., Ch. Korsemov, Hr. Toshev, "The Artificial-Neural-Networks-Design Generalized Evolutionary Model", Proceedings of the 15th International Conference on Systems for Automation of Engineering and Research SAER 2001, (Eds. Popov A., Romansky R.), Varna, Bulgaria, pp. 191-195, 2001.
11. Bean, C. and A. B. Hadj-Alouane, A dual genetic algorithm for bounded integer programs, Department of Industrial and Operations Engineering, The University of Michigan, TR 92-53, 1992
12. Michalewicz, Zb. "The significance of the evaluation function in evolutionary algorithms", Evolutionary algorithms, L. D. Davis, K. De Jong, M. D. Vose, L. D. Whitley (eds.), Springer, pp-151-166, 1999.
13. Van Rooij, A. J. F., L. C. Jain and R. P. Johnson, Neural Network Training Using Genetic Algorithms. Machine Perceptron & Artificial Intelligence. World Scientific Publisher, 1997.
14. Nair A., Srinivasan P., Blackwell S., Alcicek C., Fearon R., De Maria A., Panneershelvam V., Suleyman M., BeattieC., Petersen S., and others . Massively parallel methods for deep reinforcement learning. arXiv preprintarXiv:1507.04296, 2015
15. Hessel M., Modayil J., Van Hasselt H., Schaul T., Ostrovski G., Dabney W., Horgan D., Piot B., Azar M., and Silver D. Rainbow: Combining improvements in deep reinforcement learning, arXiv preprint arXiv:1710.02298, 2017
16. Salimans T., Ho J., Chen X., and Sutskever I. Evolution strategies as a scalable alternative to reinforcement learning. arXiv preprint arXiv:1703.03864, 2017
17. Such, F. P., V. Madhavan, E. Conti, J. Lehman, K. O. Stanley, J. Clune, Deep Neuroevolution: Genetic Algorithms are a Competitive Alternative for Training Deep Neural Networks for Reinforcement Learning, Cornell University Library, **arXiv:1712.06567v3** [cs.NE], 2018

Подход к обучению искусственных нейронных сетей с генетическими алгоритмами

Чавдар Корсемов, Стефан Койнов, Христо Тошев

Институт информационных и коммуникационных технологий, 1113 Sofia

Emails: ***chkorsemov@iinf.bas.bg***, ***slk@iinf.bas.bg***, ***hr_toshev@mail.bg***

Р е з ю м е

В статье представлены характерные особенности генетических алгоритмов (ГА), которые отличают их от традиционных методов оптимизации и процедур поиска и обучения на этапах предварительной подготовки к решению практических задач с ГА. Предложены генетические подходы к обучению искусственных нейронных сетей, включены эволюционные подходы со штрафными функциями.