Artificial Neural Networks-Training – Errors, Convergence and Genetic Approaches

Hristo I. Toshev[1], Chavdar D. Korsemov[2] and Stefan L. Koynov[3]

Abstract – **The paper introduces an approach to enhance and speed up the training process of the goal artificial neural networks (ANN). The training and the operation of the ANN is evaluated on the basis of temporally sequential copies of the ANN parameters (generations, offsprings) by means of multilayer and other models. The introduced terminology and mathematical formalism concern the errors, the convergence and the genetic approaches to train ANN.**

Keywords – **multilayer models, neural networks, penalty functions, training.**

I. Introduction

Artificial neural networks (ANN) learning is the most time-consuming step (if any).

Two determining factors: the goal ANN architecture [1] and the corresponding mathematical description [2] feed the learning process. The architecture is the most directly connected with the physical 'nature' of the ANN while the mathematical formalism presents in the discrete time domain the relations between the information streams. The mathematical model is decisive for the ANN learning because it is *defined by the learning rule* but it itself *defines the learning algorithm*.

The ANN learning process consists of the learning paradigm, the learning rule and the learning algorithm. The learning paradigm presets the possible learning rules which in turn preset the possible learning algorithms. The learning paradigm is the most abstract feature and the learning algorithm is the most concrete one. Starting from left in a direction to the right the learning characteristics become more concrete and this corresponds to a 'descent' from the peak of a pyramidal structure towards its base; the peak is analogous to the learning paradigm and the base - to the concrete application task. The very 'descent' during the ANN learning corresponds to a draw up to the final ANN design.

The scheme in Fig. 1 presents the links between these aspects in the learning process for ANN with multilayer models.

The mathematical description is of a principal importance for the self-organizing maps when the mathematical formalism may be presented according to Table 1. The table introduces numbering the layers in the model which is based on the degree of sophistication of the feature for the layer.

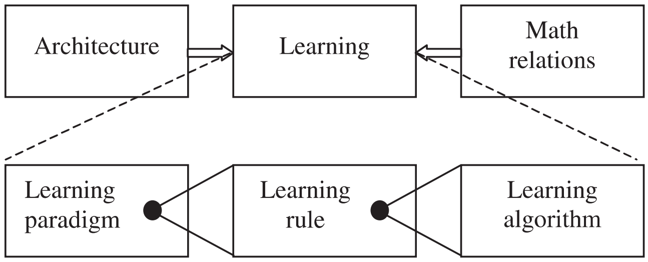

Fig. 1. The ANN learning process with multilayer models

Table 1. Most abstract ANN learning process

ANN LEARNING PARADIGM MULTILAYER MODEL		
Layer No.	**Layer Name**	**Feature of the layer**
1	SL	Learning modes
2	UL	Input-output transform
3	HL	Combined learning rules (error correction, competitive)
Supervised Learning Unsupervised Learning Hybrid Learning		

II. ANN Training: Errors, Convergence and Genetic Approaches

In the training phase 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) training 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*)

[1]Hristo I. Toshev is with the Institute of Information Technologies, Bulgarian Academy of Sciences, Acad. G. Bonchev str., bl. 29A, 1113 Sofia, Bulgaria, E-mail: toshev@iinf.bas.bg

[2]Chavdar D. Korsemov is with the Institute of Information Technologies, Bulgarian Academy of Sciences, Acad. G. Bonchev str., bl. 29A, 1113 Sofia, Bulgaria, E-mail: korsemoviinf.bas.bg

[3]Stefan L. Koynov is with the Institute of Information Technologies, Bulgarian Academy of Sciences, Acad. G. Bonchev str., bl. 29A, 1113 Sofia, Bulgaria, E-mail: baruch@iinf.bas.bg

$h^{(i)}$ for the i-th ANN state generation in the training 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 trained 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 trained 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 train 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 trained 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 training 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 training 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 trained ANN $t_g^{(i)} : \dfrac{N^{a(i)} \cdot n_g^{(i)}}{t_g^{(i)}} = I \geqslant 1$.

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 training a single ANN equals to *one:* $I_{\min} = 1$. It is obtained if the total number of the added neurons in the ANN training process is obtained during the ANN state generation number two ($n = 2$). It means that $I_{\min} \sim N^{a(i)} = N^{a(2)} = 1$ if $n_g^{(2)} = t_g^{(2)}$: 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 analogs 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.

III. ANN Training: Multilayer and Other Models

The approach with penalty functions admits an interpretation with multilayer models for exploring the ANN training 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 training process. The multilayer approach is already implied by the authors to model a serial industrial application [3].

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 trained 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 trained ANN on condition that the penalty functions for the first several offsprings of the state are obtained.

The ANN training 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 [3]; in the ANN training 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:

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

Here: $eval\left(\overline{X}\right)$ – feasible and unfeasible solutions if $\overline{X} \in F$ is the optimal solution of the general nonlinear programming model with continuous variables; $f\left(\overline{X}\right)$ – goal function for optimization; $\lambda(t)$ – updated every generation t in the following way [4]:

$$\lambda(t+1) =$$

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

$f_j\left(\overline{X}\right)$ – constraint violation measure for the j-th constraint such that [5]:

$$f_j\left(\overline{X}\right) = \begin{cases} \max\left\{0, g_j\left(\overline{X}\right)\right\}, & \text{if } 1 \leqslant j \leqslant q \\ \left|h_j\left(\overline{X}\right)\right|, & \text{if } q\text{+}1 \leqslant j \leqslant m \end{cases} ; \qquad (2)$$

Here $g_j\left(\overline{X}\right) \leqslant 0, \quad j = 1, ..., q$ and $h_j\left(\overline{X}\right) = 0, \quad j = q+1, ..., m$ comprise a set of additional constraints $m \geqslant 0$ the intersection of which with S defines the feasible set F;

l – indicator of the constraint type with upper bound $L = 2$:

$$l = \begin{cases} \text{1 : inside a given layer (inside an ANN state generation)} \\ \text{2 : between two layers inside the ANN learning multilayer model (between two successive ANN state generations)} \end{cases}$$

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

Besides multilayer models other types of models can be introduced to model the ANN training process [6]: the homogeneous features (belonging to one class or one group) are ordered in hierarchical systems and the heterogeneous features (belonging to different classes or groups) are clustered in multilayer models; the size of the paper does not allow the detailed introduction of them

IV. Conclusions

The paper presents an approach of the authors to enhance and accelerate the goal artificial neural networks (ANN) training. The training 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 trained 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] A. Jain, “Artificial Neural Networks: A Tutorial”, *IEEE Computer*, pp. 31-44, March, 1996.

[2] B.Kosko, *Neural networks and fuzzy systems. A dynamic systems approach to machine intelligence, Part 1*, Englewood Cliffs, Prentice Hall, N.Y., 1992.

[3] S. Koynov, Ch. Korsemov and H. Toshev, “The artificial-neural-networks-design generalized evolutionary model”, *15th Int. Conf. on Syst. for Automation of Engineering and Research SAER’2001,Proc.* 21-23 September 2001, Varna - St. Konstantin resort, Bulgaria, pp. 191-195, 2001.

[4] J.C.Bean 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.

[5] Zb. Michalewicz, “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.

[6] S. L. Koynov, Ch. D. Korsemov, Hr. I. Toshev and L. M. Kirilov, “Multi-Layer Models and Learning of The Artificial Neural Networks”, *Proceedings of the XXXVII International Scientific Conference on Information, Communication and Energy Systems and Technologies ICEST 2002*, 2-4 October 2002, Nish, Yugoslavia, pp. 113-116, 2002.

[7] S. Koynov, Ch. Korsemov and H. Toshev, “The design of the artificial neural networks as a basis for their generalized evolutionary model”, *15th Int. Conf. on Syst. for Automation of Engineering and Research SAER’2001,Proc.* 21-23 September 2001, Varna - St. Konstantin resort, Bulgaria, pp. 185-190, 2001.

[8] S. Haykin, *Neural networks*. Englewood Cliffs, Macmillan Publishing Co., 1994.

[9] E. Falkenauer, “Applying genetic algorithms to real-world problems”, *Evolutionary algorithms*, L. D. Davis, K. De Jong, M. D. Vose, L. D. Whitley (eds.), Springer, pp. 65-88, 1999.

[10] D.B. Fogel, E. C. Wasson and E. M. Boughton., “Evolving Neural Networks for Detecting Breast Cancer”. *Cancer Letters*, v. 96, pp. 49-53, 1995.

[11] D. Williams and J. B. Gomm, “The introduction of neural network projects in a degree of electrical and electronic engineering”, *12th Int. Conf. on Systems for Automation of Engineering and Research SAER’98, Proc*. 19-20 September 1998, Varna - St. Konstantin resort, Bulgaria, pp. 102-106, 1998.

[12] I. Baruch, A. Martinez and B. Nenkova, “Adaptive Neural Control with Integral-Plus-State Action”, *Cybernetics and Information Technologies*, Vol. 2, No. 1, Sofia, pp. 37-48, 2002