Constructive algorithms as a method of achieving plasticity of ANNs

I V Kovalev1 and N V Yakasova1,2

1 Siberian Federal University, Svobodny Ave., 79, Krasnoyarsk, 660041, Russia
2 Khakassia state University named after N. F. Katanov, Lenin Ave., 90, Abakan, Republic of Khakassia, 655017, Russia

E-mail: ice@khsu.ru

Abstract. Artificial neural networks are a model of natural neural networks. For natural neural networks, the incessant genesis causes their plasticity and stability. In a changing environment, artificial neural networks must provide not only stability, but also plasticity. The solution to the problem is constructive algorithms that allow you to create architectures with an optimal structure.

1. Introduction

Artificial neural networks (ANNs) have come a long way from the ideas of McCallock and Pitts (1943) to modern neural networks, which differ in a variety of architectures and application areas.

Despite the fact that artificial neural networks were created as a model of natural neural networks, they are, according to modern representations of neurophysiologists, a very remote rather primitive model of them [1].

Multilayer perceptrons, radial networks, associative machines, and many other networks are successfully used in practice while having several disadvantages that limit their use on a wider range of tasks. Real tasks have a variety of input information, which is virtually impossible to foresee, which leads to the need to repeatedly return to retraining the network in the operating conditions. The learning process itself is long in time and is accompanied by the solution of particularly problems: the formation of a training sample, the order of presentation of examples, the selection of training parameters, the solution of issues to getting into a local minimum, etc.

Another problem is the lack of stability of the ANNs in combination with plasticity, that is, the ability to remember new information without losing the ability to perceive and process old information, as it happens in the natural neural network. As a result, the ANN meeting with a new image that violates the classification known to it cannot process it correctly and requires retraining [2].

While real-world tasks, such as knowledge management tasks, self-control of systems, are impossible without the presence of a plastic memory in ANN that could take into consideration the previous experience and respond flexibly to current changes, the emergence of new knowledge, and change of the context [3].

An important question that still has no clear answer is the choice of the optimal neural network architecture. Most of the methods used are varieties of deconstructive (reduction) methods, which are characterized by the initial definition of the network architecture based on some theoretical or empirical considerations (the number of layers, types of connections between neurons, activation functions are...
defined, network parameters are initialized). After that, training and optimization of the network are performed, as a result of which "extra" connections and neurons are usually removed from the network.

The problem with this approach is that the learning process usually takes a long time, it is rather difficult to predict the network parameters in advance (if not impossible), and the resulting network is no longer able to solve the problem when the classification conditions change.

Attempts to solve these problems have been made repeatedly, so there were constructive algorithms, cascading architectures, and adaptive neural networks.

2. Neurogenesis as a natural constructive algorithm

Based on recent studies of brain plasticity and neurogenesis, it can be noted that natural neural networks are characterized by continuous neurogenesis, which ensures the plasticity of the natural neural network in the conditions of the variability of the external world [1].

At the same time, the researchers note that the high neurogenesis of the old age of the organism is significantly reduced but does not stop. It goes in a qualitative rather than quantitative form.

In young organisms, neurogenesis is intensive in quantitative terms and is certainly related to the learning process. One of the theories accepted by the scientific community that explains this process is the pattern separation theory, which reduces to the fact that new neurons appear in response to the emergence of non-standard situations that cannot be reduced to already known patterns [2].

It is noted that the more intense neurogenesis in the young state, the less it is expressed in old age. A natural neural network is characterized by some saturation in the learning process, under the influence of which the neural network is actively formed. In the future, the brain remains plastic, but since the already trained brain rarely meets non-standard situations, the process of neurogenesis slows down. Thus, natural neural networks develop from simpler structures (it is possible to talk about some template networks) to more complex ones.

Therefore, the development of ANN built on the principles of reduction contradicts the principles laid down to neural networks by nature: development from simple to complex [3].

The implementation of the idea of complicating ANN in the learning process is embodied by constructive algorithms. They are considered when creating perceptron, radial, polynomial networks as one of the ways to reduce network training costs.

3. The constructive algorithms of the ANN

The idea of constructive algorithms directly follows from the training methods of multilayer neural networks, in which, during the training process, recustomized of the weights of neurons in hidden layers occurs. If learning was unsuccessful, the neurons are added to the hidden layers and produced training again. That leads to the idea of gradually adding neurons to the network during training.

To form a network, a fairly simple structure is usually chosen, which changes during the learning process by adding new neurons whose connections are trained together with the existing network structure. Constructive algorithms at the initial stage have a small number of neurons in hidden layers, the number of which increases during the learning process.

The difference between such networks is that the skills acquired by the network earlier are not lost in the learning process with an increase in the number of neurons. The architecture is refined in the learning process, which allows training the network for new examples.

Constructive algorithms differ in the way they add new neurons to the network and initialize them. The simplest algorithms form a network by adding or splitting individual neurons.

The initialization of neurons can be performed by random numbers from a given range (or by predetermined values) or by copying the weights of one of the neurons. With this initialization of neurons by random numbers, the error function increases with the introduction of each new neuron in the network, which may ultimately lead to redundancy of neurons in hidden layers.

Copying the weights of a neuron is done by the splitting method: among neurons, one is selected, the change vector of which has two preferred directions. During training, such neurons often lead to
problems associated with significant fluctuations in the error function (with the backpropagation method) or falling into a local minimum (when learning with the integral error function).

The vector of changes in the synaptic weights is stored in a covariance matrix, from which the eigenvectors and eigenvalues of the matrix are calculated, and thus the stochastic gradient rise and Gram-Schmidt orthogonalization are performed. The splitting of neurons solves the problems of the first method, but the computation time exponentially increases with the growth of the network dimension.

A faster version of the algorithm uses the splitting of the neuron with the largest value of the functional

$$F_i = \frac{\sum_{k=1}^{N} |\delta w_i^k|}{\sum_{k=1}^{N} |\delta w_i^k|}$$

where is the change vector of the synaptic weights of the i-th neuron as a result of using the k-th training example [5].

Instead of the found neuron, two new ones are introduced into the network, the synaptic weights of each of which are the sum of the weights of the found neuron with some noise. The connections of new neurons with the neurons of the next layer will repeat the connections of the found neuron, but at the same time, they decrease in value by half. Thus, it is guaranteed that the error function will not increase. The disadvantage of this algorithm can be considered a slightly larger number of neurons in the hidden layer than the previous algorithm.

Another version of the neuron splitting method is reduced to sensitivity analysis based on Hessian matrices for the second derivatives of the error function, modulo the second derivatives determine the parameters for zeroing (parameters with small values). This method is more demanding on computing resources. The resulting network receives the optimal structure, the network learning process is significantly accelerated and provides faster convergence of the learning process.

Constructive algorithms have to solve two problems: the choice of architecture and weight settings. As the authors of V.L. Matrosov, Z.M. Shibzukhov in [6] the choice of a mathematical model of a neuron determines the characteristics of the ANN, the process of selecting weight coefficients and its architecture. The choice of a model of classical neurons leads to the generation of multilayer networks, which are characterized by significant computational complexity. A more complex mathematical model of a neuron, which brings it closer to natural neurons in its properties, simplifies the architecture of the ANN. An example of such neurons can be considered ΣP-neurons.

ΣP-neuron is the best algebraic model of a natural neuron, it is based on the multilinear function of the total signal

$$y = \text{out}(\theta + \sum w_i \prod_{i \in i_k} x_i), i_k \subseteq \{1, \ldots, n\}$$

ΣP-neuron can represent arbitrary boolean functions. If ΣP-neurons are modified, they can be used to approximate discrete functions over finite discrete sets, and continuous functions [6].

In [7–10], the construction of ANN from algebraic ΣP-neurons using direct constructive combinatorial-algebraic procedures based on an ordered sequence of training examples was shown. In one pass, only one ΣP-neuron is added and trained, its weights are adjusted in accordance with minimization of the ranks of the multiplicative terms.

Additional restrictions are used to optimize network architecture. A constructive approach to teaching a wide class of algebraic ΣP-neural networks and some of their generalizations was theoretically substantiated and developed in [11].

4. Cascading models

Another approach to the formation of the ANN structure is demonstrated by the creators of cascading networks. The cascade model was proposed by Falman in 1990. In contrast to architectures with fixed topology, network formation begins with a minimal network, which grows in the learning process. The
model was developed as an attempt to overcome the main limitation of the popular back-propagation error algorithm - the slow learning rate. Falman and his colleagues identify two problems: step size and the attempt of each neuron to approach the approximated function.

The cascade algorithm solves the problems of the backpropagation method by implementing two ideas. The first is that in the learning process, new blocks are sequentially added to the hidden layer, one at a time, which do not change in the subsequent. The second is related to an attempt to achieve maximum correlation between the output of a new block and the current error size, which should be eliminated.

The cascade correlation network consists of neurons integrated into a developing cascade. A new neuron connects to all nodes of the input and hidden layers. All neurons, both the input layer and the hidden, are connected to the neurons of the output layer.

At the first stage, a network is created consisting only of the input and output layers of a virtually single-layer perceptron. Output neurons can have any activation function. The number of inputs and outputs is determined by the task and does not change further. All inputs are connected to all outputs; and the connection weights are refined during the training process.

Each new neuron forms a separate network layer. The initial network, which consists of two layers, is trained to minimize the value of the objective function, after which hidden neurons begin to be added to the network. For training in the original method, the quickprop algorithm was used, which had fast convergence. In practice, any methods can be used for training.

When a hidden neuron is included in the network, it is first added as a candidate. It connects to all network inputs and all outputs of existing hidden neurons. A candidate neuron is trained using the same training set as the original perceptron. The weights of connections are trained but the output is not output anywhere. The entire training set is used to train the neuron. Weights are selected so as to increase the correlation between the activity of the neuron and the value of the error of the network at the output.

In practice, it is more profitable to simultaneously train several candidate neurons (from 5 to 10), from which, after training, a neuron that reaches the maximum value of S. is selected. Candidate neurons can have different activation functions. Parallel training of several neurons avoids getting into a local minimum or adding a neuron with unsuccessful weights to the network. The process of network formation is completed upon reaching an acceptable error. The resulting network is different from a layered network and therefore it is not a perceptron [12].

A cascade network may differ in the heterogeneity of neurons, which provides greater plasticity. In [13], the principle of constructing a hidden layer using cascade algorithms using a configurable sigmoid activation function is used. The result is a network that, in comparison with a similar homogeneous network, has the advantage of flexibility, stability of training, and optimality of architecture.

Cascade-correlation architecture has several advantages over destructive algorithms: it learns quickly enough, the network determines its size and topology (the network is close to optimal), and it does not require the backpropagation of error signals through network connections [12].

In the work of Avedyan E.D., Barkana G.V. and Levina I.K. [15] it was shown that a nonlinear scheme for constructing cascading ANN can be used, while the network is in no way inferior to networks obtained using the linear scheme. The characteristics that are important for operation are kept at the same level: the approximation accuracy order is \( O \left( \frac{1}{n} \right) \), the approximation rate does not depend on the dimension of the input space. Moreover, the class of approximable functions for cascading ANN is wider than for two-layer ANNs [14].

More modern approaches to the use of cascade architecture methods were demonstrated in [15]. Authors E.V. Bodyansky, V.V. Volkova, S.D. Gromov considered a cascading evolutionary ANN with neo-fuzzy neurons as nodes with the Falman-Lebier correlation criterion for learning, the Shink pattern recognition criterion. The network is constructed of blocks based on neo-fuzzy neurons (presented in [16]).

The neo-fuzzy neuron is a nonlinear synapse that converts the input signal into a fuzzy zero–order Takagi-Sugeno output [17]. Because the network is cascaded, its configurable parameters are determined linearly.
Neo-fuzzy neurons provide “an important advantage in learning speed, computational simplicity, and the possibility of finding the global minimum of the learning criterion in real time” [15].

Experimentally, L. Prechelt showed that the learning algorithms of a neo-fuzzy-neural network are significantly faster than the correlation algorithms and can be performed both in batch and in adaptive mode. [16].

The disadvantage of the neo-fuzzy-neural network is the “curse of dimension” associated with the scattered partition of the training sample [17].

5. Constructive algorithms in deep learning networks

Recently, deep learning networks have gained further distribution. In this regard, interest in constructive algorithms is growing. This is primarily due to the fact that the presence of a large number of hidden levels in deep learning networks accordingly provokes the presence of a significant number of customizable parameters. And this, given the large-scale training sample, increases the difficulty of obtaining ANNs with optimal architecture. Since the tasks of reduction algorithms are the initial determination of the number of hidden layers and initialization of compounds, this usually determines the convergence of learning.

A constructive algorithm for building a deep learning network is proposed in [18]. The network begins with one hidden layer with one neuron, then in the learning process, neurons are added to the hidden layer until the maximum number of neurons in the layer is reached, or the learning termination condition is met. When the next layer is saturated, a new layer is created and the iterative process is repeated.

More often used are hybrid algorithms combining the advantages of constructive and deconstructive algorithms. The construction of the network begins with a certain minimal configuration, to which, in the process of training, constructive and deconstructive methods are alternately applied, forming the optimal architecture. One of the advantages of the hybrid approach is the ability to overcome the problem of noisy data [19, 20].

6. Conclusion

The topic of constructive algorithms is studied to a large extent less than deconstructive algorithms. This is probably due to a variety of approaches, the specificity of methods and algorithms that distinguishes the work of different authors. While for deconstructive algorithms a large number of typical architectures and methods have been developed for solving various kinds of problems.

Nevertheless, there is a periodic surge of interest in this area, which is associated with the attractiveness of the idea of obtaining a quickly trained neural network that combines plasticity and stability.

The evolution of natural neural networks and the experience of creating constructive algorithms demonstrate the promising possibilities of using constructive algorithms in combination with the complication of neural elements and their differentiation, combining them with the reduction to obtain networks with the plasticity and stability of natural neural networks.

References

[1] Kempermann G 2008 The neurogenic reserve hypothesis: what is adult hippocampal neurogenesis good for? Trends Neurosci 31 163–9
[2] Aimone J B, Deng W and Gage F H 2011 Resolving new memories: a critical look at the dentate gyrus, adult neurogenesis, and pattern separation neuron 70 589–96
[3] Kolobashkina L V 2020 Biologically Inspired Algorithm for Increasing the Number of Artificial Neurons Advances in Intelligent Systems and Computing 948 225-31
[4] Palyanova A Yu, Palyanova N V and Khairulin S S 2012 About problems of modeling neural networks of living organisms bulletin of nsu 10(3) 46-57
[5] Kruglov V V and Borisov V V 2002 Artificial Neural Networks. Theory and Practice (Moscow: Hotline-Telecom) pp 39-41
[6] Matrosov V L and Shibzukhov Z M 2009 Constructive algebraic approach to training of ΣP-neural networks *Teacher of the XXI Century* 1 199-203
[7] Shibzukhov Z M 2007 *Constructive Training of Algebraic ΣΠ-Neural Networks and Correct Σπ-Extensions* (Moscow) p 38
[8] Shibzukhov Z M 2003 Recurrent Method of Constructive Training of Algebraic ΣΠ-Neurons and ΣΠ-Neuromodules *Reports of the Russian Academy of Sciences* 388(2) 174-6
[9] Shibzukhov Z M 2004 Correct ΣΠ Extensions of a Valid Class of Algorithms *Reports of the Russian Academy of Sciences* 394(4) 462-4
[10] Shibzukhov Z M 2006 *Constructive Methods of Training ΣP-Neural Networks* (Moscow: MAIK Nauka) p 159
[11] Fahlman S E and Lebiere C 1991 *The Cascade-Correlation Learning Architecture* (Pittsburgh: School of Computer Science Carnegie Mellon University) p 13
[12] Sharma S K and Chandra P 2012 A constructive algorithm with adaptive sigmoidal function for designing single hidden layer feedforward neural network *Advanced Materials Research* 403-408 3867-74
[13] Avedyan E D, Barkan G V and Levin I K 1999 Cascade neural networks *Automaton. and telemech* 3 38-54
[14] Bodyansky E V, Volkova V V and Gromov S D 2011 Cascade evolutionary neural network with neo-fuzzy neurons as nodes *Eastern European Journal of Advanced Technologies* 4/3 (52) 55-8
[15] Prechelt L 1997 Intesigation of cascag familyof learning algorithms *Neural Networks* 10(5) 885-96
[16] Zaichenko Yu P and Gamidov G I 2017 Cascade neuro-fuzzy networks in forecasting problems in securities markets *System Research & Information Technologies* 2 92-102
[17] Bodyansky E V, Vinokurova E A and Mulesa P P 2016 Diagnosing a wavelet-neuro-fuzz system with adaptive wavelet membership functions in problems of multidimensional data analysis *USIM* 2 34-40
[18] Zemouri R, Omri N, Morello B, Devalland C, Arnould L, Zerhouni N and Fnaiech F 2018 Constructive Deep Neural Network for Breast Cancer Diagnosis *IFAC* 51 98-103
[19] Hou M, Taohua L, Yunlei Ya, Hao Zh, Hongjuan L, Xiugui Yu and Xinge L 2017 A new hybrid constructive neural network method for impacting and its application on tungsten price prediction *Applied Intelligence* 10
[20] Zemouri R, Omri N, Fnaiech F, Zerhouni N and Fnaiech N 2019 A new growing pruning deep learning neural network algorithm (GP-DLNN) *Neural Computing and Applications* 10