УДК 004.891.3

ОБУЧЕНИЕ ЯДЕРНЫХ НЕЙРОННЫХ СЕТЕЙ

М. Ф. БАЙМУХАМЕДОВ,
доктор технических наук, профессор, проректор по науке,
Костанайский социально-технический университет им. З. Алдамжар
(Республика Казахстан, 110010, г. Костанай, ул. Герцена, д. 27)
А. М. БАЙМУХАМЕДОВА,
доктор философских наук,
Башкентский университет
(Турция, 06810, г. Анкара, ул. Ескишекир, д. 45)

Ключевые слова: *нейронная сеть, статическая настройка, синоптический вес, весовые коэффициенты, смещение, слои, алгоритм обучения, входной сигнал, выходной сигнал.*

Рассматривается статическая настройка нейронной сети, предшествующая обучению ядерной нейронной сети. При статической настройке осуществляется выбор синоптических весов и смещений для первого и скрытых слоев нейронной сети. Далее приводятся основные математические соотношения, которые используются при выполнении алгоритма обучения ядерной нейронной сети. В алгоритме обучения параллельно с настройкой синоптической карты выполняется подбор нелинейных функций за счет изменения смещения их аргументов. Предлагаемая модель обучения обеспечивает процесс адаптации нейронной сети для достижения минимума функциональной оценки и качественного решения определенной задачи нейронной сетью. Признаками, которыми должна обладать задача, чтобы применение нейронных сетей было оправдано, и нейронная сеть могла бы ее решить, являются:

– отсутствует алгоритм или не известны принципы решения задач, но накоплено достаточное число примеров;
– проблема характеризуется большими объемами входной информации;
– данные неполны или избыточны, зашумлены, частично противоречивы.

NUCLEATE NEURAL NETWORKS TRAINING

M. F. BAIMUKHAMEDOV,
doctor of technical sciences, professor, vice-rector for science,
Kostanay Social and Technical University of Z. Aldamzhar
(27 Herzen Str., 110010, Kostanay, Kazakhstan Republic)
A. M. BAIMUKHAMEDOVA,
doctor of philosophical sciences,
Baskent University
(45 Eskişehir Yolu, 06810, Ankara, Turkey)

Keywords: *a neural network, static adjustment, synoptic weight, weight factors, displacement, layers, algorithm of training, an entrance signal, a target signal.*

The static adjustment of a neural network previous training of a nuclear neural network is considered. By static adjustment the choice of synoptic scales and displacement for the first and the latent layers of a neural network is carried out. Further the basic mathematical parities which are used at performance of algorithm of training of a nuclear neural network are resulted. In the training algorithm along with synoptic chart adjustment the nonlinear functions selection by changing their arguments displacement is performed. The offered model of training provides the process of neural network adaptation to achieve the minimum of a certain estimating functional and the qualitative solution of a certain problem by neural network. Features that the task should have in order to justify the application of neural networks, and the neural network could solve it, are:

– there are no algorithm or known principles for tasks solution, but there is a sufficient number of examples;
– the task is characterized by large volumes of input data;
– the data is either incomplete or redundant, noisy, partially contradictory.

Положительная рецензия представлена Н. А. Потехиным, доктором экономических наук,
профессором Уральской государственной сельскохозяйственной академии.

Artificial intelligence systems which are based on algorithmic languages have been developed according to the World Program of “The fifth computer generation”. Now it is replaced by another program – “Calculations in the real world”. Its top priority is to develop information systems that can operate in the absence of a man in a “natural” outside world.

A great deal of this program is given to the creation of artificial neural networks. At the same time a great majority of works on neuro-informatics are dedicated to the transfer of various tasks solution algorithms to such neural networks. The chapter shows that there is a large class of problems for which the connections are formed according to evident/explicit formulas (neural associative memory systems, statistical processing, filtration, etc.), as well as tasks requiring implicit process. This process is called training.

It is noted that the effective operation of neural networks requires parallelism and artificial neural networks are one of the powerful means of parallel computers programming.

At designing of information systems on the basis of neural networks the training of nuclear neural networks is played the important role.

The problem description. Before training it is necessary to set initial values of weight factors in the neural network in this or other way. The weight factors are usually initialized randomly. Statistical adjustment is intended to improve the initialization algorithm on the basis of additional information about the data [1, 2]. The purpose of static adjustment is:

1) reduction of input symbols real levels to the operating range of the neural network by the expedient choice of synoptic weights and displacement in the first layer;

2) reduction of the network output signals to the range of true values by the expedient choice of synoptic weights and displacement in the last layer;

3) setting the initial state of the network into the area of general position, that is, maximum sensitivity to parameter variation by optimal choice of initial values of displacements and synoptic weights for the neurons of hidden layers.

Let us mark the weight factors of synoptic maps for layer m as:

$$w_i^m(a,b) \tag{1}$$

where *i* is the number of the nucleus, and *m* is number of the layer.

Data processing for the nucleus A_i^m is defined by the expression

$$s_i^m(v) = \sum_u x_i^m(u) w_i^m(u,v) \tag{2}$$

where u = 1,2, ..., p.

In the training algorithm along with synoptic chart adjustment the nonlinear functions selection by changing their arguments displacement is performed. Formally, displacement is realized by adding a fictitious (dummy) $x_i^m(*)$ coordinate, which usually has a constant value equal to +1, and, besides, for each neuron nucleus an adjusted synoptic weight value $w_i^m(*,b)$ is added. This technique allows you not to change the form of the expression (2), assuming that the number of summands in this sum is increased by one. Static adjustment of the first layer

Source information:

For each input variable $x(u)$ the following is considered to be known

− the average value of the variable: $\overline{x}$

− change range: Δx

Optimality principles*:* For each input neuron the input variables change in the authentic levels range should have about the same change of the neuron output signal.

The level of displacement for each neuron must be set so that to provide at the average the maximum value of the derivative for the activation function in the training sampling.

Design equations:

$$\left|w_i^1(u,v)\right| \le \frac{1}{\Delta x_i^1(u)} \tag{3}$$

Sign (digit) of the weighting factor is chosen in a random way. There is a formula, used to calculate the displacement value of the activation function

$$w_i^1(*,v) = -\sum_{u=1}^{p} x_i^{-1}(u) w_i^m(u,v) \tag{4}$$

Statictical adjustment of hidden layers. In the hidden layers neurons with sigmoid or tangential activation function are typically used, so in the worst case

$$\Delta x_i^m(u) = 2 \tag{5}$$

All speculations on the first layer synoptic weights choice are valid for hidden layers. Using formula (5) for hidden layers, we obtain:

$$\left|w_i^m(u,v)\right| \le \frac{1}{2} \tag{6}$$

Sign of synoptic weight is chosen in a random way. After synoptic weights are set, the displacement value is calculated by the formula

$$w_i^m(*,v) = -\sum_{u=1}^{p} x_i^{-m}(u) w_i^m(u,v) \tag{7}$$

Value of the averages $x_i^{-m}(u)$ is determined directly from the network, provided that for the input of the first layer average values of the input variables have been forced and the displacement and balance set-up in the previous layers have been implemented optimally.

Statictical adjustment of the last layer. In order to obtain the true values of the output variables of the network, the activation functions in the last layer should be linear. For output variables values of the average $y_j^{-m}(v)$ and change range of values $\Delta y_j^n(v)$ are assumed known.

Modules of the weight factors are determined by the expression

$$\left| w_j^n(u,v) \right| = w_j^n(v) = \frac{\Delta y_i^n(v)}{p} \qquad (8)$$

The sign is chosen in a random way. To calculate the displacement of activation functions the following formula is used.

$$w_j^n(*,v) = y_j^{-n}(v) - \sum_{u=1}^{p} x_j^{-n}(u) w_j^n(u,v) \qquad (9)$$

Let us present the basic mathematical relations, used in the performance of the training algorithm for nuclear neural network [3].

The expression for the learning criterion J is (looks like) this:

$$J = \sum_j \sum_i (y_j^n(v) - z_j(v))^2 \qquad (10)$$

where $y(v)$ are coordinates of the output vector of the neural network $z(v)$, is the desired value of the output vector coordinates, j is the number of the nucleus in the output layer, v is the number of axons within the nucleus.

For each layer m gradient coordinates are determined by partial derivatives.

$$\frac{dJ}{dw_i^m(a,b)} = \delta_i^m(b) * x_i^m(a) \qquad (11)$$

where $\delta_i^m(b)$ is the generalized error, $x_i^m(a)$ are coordinates of the receptor field of the nucleus i of the layer m, $w_i^m(a,b)$ are elements of the nucleus synoptic map. The following expression can recursively compute the generalized errors for all layers except the last layer.

$$\delta_i^m(b) = \frac{df_i^m}{ds_i^m(b)} * \sum_v \delta_j^{m+1}(v) * w_j^{m+1}((b)\rho_{ij}^m, v) \qquad (12)$$

where $s_i^m(b)$ is the coordinate of the state (the argument of the activation function)

$$y_i^m(b) = f_i^m(s_i^m(b)) \qquad (13)$$

$$s_i^m(b) = \sum_a x_i^m(a) w_i^m(a,b) \qquad (14)$$

$$\rho_{ij}^m = (\mu_i^m)^{-1} q^m \sigma_j^{m+1} \qquad (15)$$

Formally the argument displacement of the nonlinear function is realized by adding the fictitious coordinate $x_i^m(*)$, usually having a constant value equal to + 1, and, besides, customizable values $w_i^m(*,b)$, that are considered accessory of the synoptic chart and must be set up in the process of education, are added.

For the last layer the generalized error is determined by the expression:

$$\sigma_j^n(v) = 2 * \frac{df_j^n}{ds_j^n(v)} (y_j^n(v) - z_j(v)) \qquad (16)$$

and the coordinates of the gradient are calculated by the formula

$$\frac{dJ}{dw_j^n(u,v)} = \sigma_j^n(v) x_j^n(u) \qquad (17)$$

In accordance with the principle of gradient search, the new values of the synaptic weights are determined from the old ones by additive correction in the direction of the anti-gradient:

$$W := W - \gamma \nabla J(W) \qquad (18)$$

Here γ is the learning (educational) quotient, which is usually determined empirically.

The offered model of training provides the process of neural network adaptation to achieve the minimum of a certain estimating functional, for example, the solution quality of the assigned task by network.

Cascading of neural networks. There are technologies that allow combining several neural modules into a single neural network. The cascading procedure is designed to interface neural networks on data flow and error back propagation. This allows the use of the generic method for forward propagation modular networks training with arbitrary structure [4, 5].

Cascading procedure presupposes that each neuron module has an output module for the generalized error vector, which is formed in the first layer of the neural network. Generalized error vector is used to build the reference vector for the preceding neural module.

The output vector of generalized errors for nuclear neural network is determined by

$$\sigma_B(U) = 0.5 \sum_v \sigma_{Bi}^1(v) W_{Bi}^1(u,v) \qquad (19)$$

where

$W_{Bi}^1(u,v)$ is a synoptic map of nucleus Bi, the first layer of the neural network;

$\sigma_{Bi}^1(v)$ are generalized errors of nucleus Bi, the first layer of the neural network;

$U = (u)(\sigma_{Bi}^1)^{-1}$ is the global number of input layer receptor.

In case of cascade connection modules the neural network learning can be made with different speeds of training for the incoming modules.

The beginning of the modern mathematical modeling of neural computation was put in 1982 by Hopfield's works, which formulated the mathematical model of associative memory at the neural network using Hebbian rules for network programming [6, 7].

But not only the model itself has caused the emergence of other authors works on the subject as the neural network computing power function introduced by Hopfield. This is an analog of the Lyapunov's function in dynamic systems.

It is shown that for a single-layer neural network with links such as "all for all" convergence to one of a finite set of equilibrium points (dots) is typical; they are local minimum of the energy function which contains the whole structure of relationships in the network in itself.

Other researchers also understood this neural network dynamics. However, Hopfield and Tank have shown how to construct the power function for a given optimization

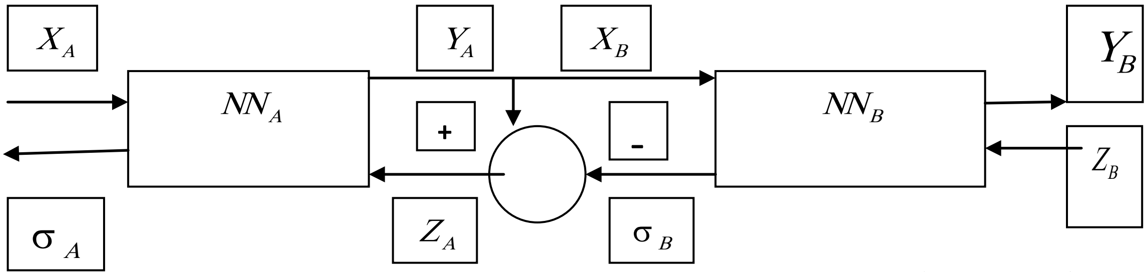

Fig. 1. Cascade connection of two neural modules

task and how to use it for the task imaging in the neural network. This approach was developed for other combinatorial optimization problems solution too. Hopfield's approach attractiveness is that the neural network for a specific task can be programmed without training iterations. Connections weights are computed on the basis of the power function type that is designed for this task.

Boltzmann's machine proposed and researched by Jeffrey E. Hinton and R. Zemel is the development of the Hopfield's model for combinatorial optimization problems solution and for artificial intelligence tasks.

Problems solved on the basis of neural networks. Features that the task should have in order to justify the application of neural networks, and the neural network could solve it, are [8, 9, 10]:

- there are no algorithm or known principles for tasks solution, but there is a sufficient number of examples;
- the task is characterized by large volumes of input data;
- the data is either incomplete or redundant, noisy, partially contradictory.

Thus, neural networks are well suited for image recognition and classification tasks solution, optimization and forecasting. The following is a list of possible industrial applications of neural networks, on the basis on which either already established commercial products created, or demonstration prototypes are realized.

Banks and insurance companies:

- Automatic checks and financial documents pick-up;
- Verification of signatures;
- Risk assessment for loans;
- Prediction of changes in economic indicators.

Administrative services:

- Automatic documents pick-up;
- Automatic barcodes recognition.

Petroleum and Chemical Industry:

- Geological information analysis;
- Equipment failure identification;
- Mineral deposits exploration on aerial photographs data;
- Impurities composition analysis;
- Process control.

Armaments industry and Aeronautics:

- Audio signal processing (separation, identification, localization, noise elimination, interpretation);
- Radar signals (target detection, identification and sources localization);
- Infrared signals processing (localization);
- Information generalization;
- Automatic piloting.

Industrial production:

- Manipulators control;
- Quality control;
- Processes control;
- Failure detection;
- Adaptive robotics;
- Voice control.

Security personnel:

- Face, voice and fingerprints detection.

Biomedical industry:

- X-ray photograph analysis;
- Deviation in the electrocardiogram detection.

TV and communication:

- Adaptive network connection control;
- Image squeezing and restoration.

The presented list is far from being complete. Every month mass media report about new commercial products based on neural networks. So, different equipment which can, for example, monitor water quality, find plastic bombs in the luggage of passengers, etc., is produced. Investment banks experts by means of software neuropackage make short-term forecasts of currency fluctuations.

The major commercial hardware products based on neural networks are, and probably in the near future will be, neuro LSI (large-scale integration circuit). Different types of neuro LSI the parameters of which often differ by many times are produced. Among them is ETANN model of Intel. This LSI, made according to Micron Technology, is an implementation of a neural network with 64 neurons and 10 240 synapses [11].

Among the cheapest is the model MD 1220 neuro LSI of Micro Devices. This LSI implements a neural network with 8 neurons and 120 synapses.

Among the currently developed neuro LSI there is a model of Adaptive Solutions (USA) and Hitachi (Japan). Neuro LSI of Adaptive Solutions is likely to become one of the fastest: the processing speed is 1.2 billion connections. (Neural network contains 64 neurons and 262144 synapses). Neuro LSI of Hitachi allows realizing of the neural network with up to 576 neurons. These neuro LSIs undoubtedly will become the basis for new neurocomputers and specialized multiprocessor products.

Most of today's neuro-computers are merely a personal computer or workstation, which include additional neuro-board. These include, for example, a series of FMR computers of Fujitsu. Such systems have an unquestionable right to exist, because their capacity is sufficient for the development of new algorithms and solutions of a large number of applications by means of neuromathematics methods. However, the most interesting are specialized neuro-computers which immediately implement the principles of the neural network. Typical examples of such systems are Mark family computers TRW (the first implementation of the perceptron, developed by Rosenblatt, was called Mark I). Mark III model of TRW is a workstation with up to 15 Motorola processors with mathematical coprocessors. All processors are connected by a VME bus. The system architecture that supports up to 65.000 virtual processing elements with more than 1 million adjustable connections enables to process up to 450 thousand interconnections. Mark IV is a uniprocessor supercomputer with a pipelined architecture. It supports up to 236 thousand virtual processing elements that can process up to 5 million interconnects. Mark family PCs have a common modeling environment ANSE (Artificial Neural System Environment) which provides models software compatibility. In addition to these models TRW firm offers a Mark II package – a software emulator of the neural network.

Another interesting model is NETSIM neurocomputer, created by Texas Instruments based on the development of Cambridge University. Its topology is a three-dimensional lattice of standard computing nodes based on processors. NETSIM computer is used for modeling of neural network models such as the Hopfield – Kohonen network and neural network with back-propagation. Its performance reaches 450 million interconnects.

Computer Recognition Systems (CRS) sells a series of WIZARD/CRS 1000 neuro-computers for the processing of video images. CRS 1000 model has already found application in industrial automatic control.

Today, the market offers many models of neuro-computers. In fact, there may be a lot more, but the most powerful and advanced models are still created on the military orders.

Conclusion. The major commercial hardware products based on neural networks are, and probably in the near future will be, neuro LSI (large-scale integration circuit). Different types of neuro LSI the parameters of which often differ by many times are produced.

The characteristic example of the successful application of neural computation in the financial sector is credit risk management. Another very important area of application of neural computation in the financial sector is the prediction of the situation at the stock exchange.

The standard approach to this problem is based on rigidly fixed set of “rules of the game”, which will eventually lose their effectiveness due to changes in the terms of trading at the stock exchange. In addition, systems based on this approach, are too slow for situations that require immediate solutions. That is why the main Japanese companies operating at the securities market decided to use the method of neural computation. They fed into a typical system based on neural network a total of 33 years of business activity of several organizations, including turnover, the previous value of the shares, income levels, etc. Self learning from real-world examples, the system has shown greater neural network prediction accuracy and better performance: compared with the statistical approach gave improved performance in general by 19 %.

Литература

1. Тенк Д. У., Кхопфилд Д. Д. Коллективные вычисления в нейронных электронных схемах // В мире науки. 2012. № 2. С. 44–53.

2. Куссул В. М., Байдук Т. Н. Построение архитектуры нейронных сетей для распознавания формы объектов и изображений // Автоматика. 2011. № 5. С. 56–61.

3. Джефри Е., Хинтон М. Как обучаются нейронные сети // В мире науки. 2013. № 12. С. 103–107.

4. Цуприков С. Нейронные вычисления принимают на вооружение финансистов // Мир компьютеров. 2011. № 7. С. 57–58.

5. Масалович А. Я. От нейрона к компьютерному нейрону // Компьютеры + программы. 2012. № 1. С. 20–23.

6. Оптимальная проблема нейронной структуры // Информационные технологии и контроль : мат. 2-ой междунар. науч. конф. Алматы, 2014. С. 56–64.

7. Кувелс П., Ли Х. Л. Блочные угловые структуры и проблема погрузки в гибких производственных системах // Опер. Реш. 2011. Т. 39. № 4. С. 666–676.

8. Конноли Т. М., Бегг К. Е. Проектирование базы данных, реализация и поддержка. Теория и практика. М. : Вильямс, 2012. 1120 с.

9. Введение в компьютерную безопасность : руководство НИСТ. Проект. Администрация технологии американского Министерства торговли, 2011. 310 с.

10. Мамаев Е. Microsoft Сервер SQL 2000. СПб., 2001. 1280 с.

11. Баймухамедов М. Ф. Информационные системы : учебник. Алматы, 2013. 384 с.

References

1. Tenk D. U., Khopfild D. D. Collective calculation in neural electronic schemes // In the science world. 2012. № 2. P. 44–53.

2. Kussul V. M, Baidyk T. N. Working out of architecture neural networks for recognition of the form of objects on the image // Automatics. 2011. № 5. P. 56–61.

3. Dzheffri E. H. As neural networks are trained // In the science world. 2013. № 11–12. P. 103–107.

4. Tsuprikov S. Neuron calculations undertake on arms of financiers // Computerworld. Moscow, 2011. № 7. P. 57–58.

5. Masalovich A. I. From neuron to neuron-computer // Computers + programs. 2012. № 1. P. 20–23.

6. Optimal problem of Net Structure. // “Information Technologies and Control” : proc. of the 2nd intern. scient. symp. Almaty, 2014. P. 56–64.

7. Kouveles P., Lee H. L. Block angular structures and the loading problem in flexible manufacturing systems // Oper. Res. 2011.Vol. 39. № 4. P. 666–676.

8. Konnoli T. M., Begg K. E. Database designing, realization and support : the theory and practice. M. : Wiliams, 2012. 1120 p.

9. An Introduction to Computer Security : the NIST Handbook. U. S. Department of Commerce, 2011. 310 p.

10. Mamayev E. Microsoft SQL Server 2000. SPb., 2001. 1280 p.

11. Baimukhamedov M. F. Information Systems : textbook. Almaty, 2013. 384 p.