A New Approach to Building Artificial Neural Networks

Boris L. Zlotin[1,2], Vladimir N. Proseanic[1,2], Anatoly A. Guin[1,2],
Ivan I. Nehrishnyi Sr[2], Vladimir S. Matsenko[2], Anton N. Belyakov[3[0000-0003-1384-5064]*],
Ivan I. Nehrishnyi Jr[2], Gafur A. Zainiev[1]

[1] Progress, INC. West Bloomfield MI, USA
[2] Omega Server Inc., West Bloomfield MI, USA
[3] Ivanovo State Power Engineering University, Ivanovo, Russia
*ab_pm@mail.ru

Abstract. In 1954, Frank Rosenblatt, created the first artificial neural network - the perceptron based on understanding of operation of brain neurons. This was a brilliant achievement; however, lack at that time of knowledge on how biological neurons worked led to systematic errors in perceptron design and methods of their training. These errors have been repeatedly propagated in most artificial neural networks (ANN). The suggested paper describes the conceptual design of a brand new type of perceptron named PANN (Progressive Artificial Neural Network), free from systematic errors of classical ANN and therefore have various unique properties. Also provided data of the PANN network testing and a link that allows to directly test the proposed neural network.

Keywords: formal neuron, perceptron, neurotransmitter, batch training.

1 Introduction

In 1943 Warren McCulloch and Walter Pitts [1] created a mathematical theory of pattern recognition by a network of neurons, which worked as an intelligent information filter. In 1949, Donald Hubb [2] described the principle of training neural networks to recognize inages. In 1954, Frank Rosenblatt, based on the ideas of McCulloch, Pitts and Hubb, as well as on the biological information, build computer model of a "formal neuron" - an artificial intelligent neural network, which was called "Perceptron" and the method of its training [3].

Rosenblatt's research raised expectations of obtaining a real powerful Perceptron-based Artificial Intelligence (AI) in the near future. This did not happen. In contrast to the brilliant development of Turing Machine computers, neural networks did not produce any practically usable results. Finally, in 1969, Marvin Minsky and Seymour Papert [4] mathematically showed the most serious limitations inherent to then known models of perceptron. As a result, development in this area has stalled for 10 years.

In the 1980s, a new phase in the development of artificial neural networks began - mainly towards the specialization of their functions, design and training methods for

specific purposes and applications. Numerous systems have appeared, such as cognitrons, neocognitrons, Hopfield, Kohonen and other networks, LSTM and other recurrent networks, resonance networks, deep learning, convolutional, etc.

Today it becomes clear that the main reason for most of the difficulties with the creation and training of ANN was an involuntary mistake made by F. Rosenblatt. The knowledge about biological neurons of that time contained a big mistake. Henry Dale received in 1936 the Nobel Prize for the discovery of neurotransmitters - substances that transmit nerve pulses through synapses. He postulated the Dale's Doctrine stating that a neuron uses one neurotransmitter for all its synapses [5]. Rosenblatt transformed the Dale's Principle into rule for ANN: one synapse - one synaptic weight. Let us call this the Rosenblatt Doctrine.

Fig.1 shows the principle of constructing a perceptron based on the Rosenblatt Doctrine. Fig. 1a shows the synapse of a biological neuron, Fig. 1b - formal Rosenblatt neuron and Fig. 1c - the simplest single-layer perceptron built on its basis.

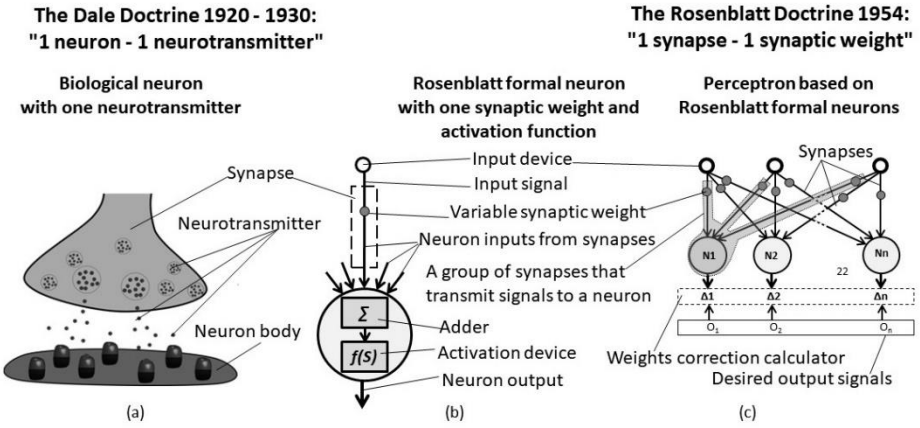

Fig. 1. The principle of construction of the Rosenblatt perceptron: (a) a biological neuron according to Dale's doctrine; (b) a formal Rosenblatt neuron; (c) Rosenblatt's perceptron

This doctrine created the main problem for all types of ANN - that between all synaptic weights in the training process, strong recursive (mutual) feedbacks arise, that is, a change in each weight affects all other weights, its leads to a number of problems:

- Due to the strong feedback between the weights, a change in one weight generates changes in all other weights, the number of which can reach enormous values. In this case, it is necessary to repeat the full training cycle for each weight.
- Training is conducted using methods of "gradient descent", that is, through many small steps of changes in synaptic weights. For complete training, it is necessary to spend thousands, sometimes hundreds of thousands of epochs. In this case, the training time T of a network grows proportionally to the multiplication product of exponents of the number of neurons n and the number of images m: $T \equiv e^n \times e^m$, which requires enormous computing power and leads to an unacceptable increase in time and cost of developing neural networks and their training time.

- To retrain the network, that is, add at least one image or eliminate an unnecessary (for example, erroneous) one, a complete retraining of the entire network is required. Additional training is partially possible, for example, in ART networks, but the other listed problems are true for these networks as well.
- In the process of training a classical ANN, its intelligence is formed, that is, the ability to recognize images close to the trained ones. At the same time, with overfitting it will be effectively recognizing the images that it was trained with, but will not be able to recognize images, even slightly different from them.
- In the training process, due of recursive feedbacks, undesirable nonlinear effects may occur, such as the inability to estimate the training time in advance, and the lack of a guarantee of successful completion of training due to phenomena such as neural network paralysis, freezing, a local minimum, etc. And the probability of problems increases with the number of neurons, the training volume and desired accuracy.
- Due to the nonlinear nature of neural networks, enormous difficulties arise in constructing their structure, choosing the optimal parameters, and the practical impossibility of scaling and increasing the power of a trained network.

The Dale Doctrine was refuted in 1970s [6]. Today it is known that in each biological synapse several transmitters can work, depends on the characteristics of the signal arriving at the synapse. As applied to neural networks, this can be interpreted as the presence on one synapse of several different weights, the choice of which to use should be related to the nature of the input signals. Below we will consider the design of a formal neuron and the construction of a new type of network based on this principle.

2 Progressive Artificial Neural Network and its training

At the beginning of the 21st century, an amateur programmer Dmitry Pescianschi came up with a modification of the formal neuron that closer to real biological neuron with a set of different neurotransmitters, and on its basis - a new design of an artificial neural network that was called later Progressive Artificial Neural Network or PANN [7 - 10].

Fig. 2a shows a synapse of a biological neuron according to modern concepts, having several neurotransmitters, in Fig. 2b - the corresponding new formal neuron, which we called Omega neuron and in Fig. 2c is the PANN network built on the basis of the Omega neuron, which, in its essence, is a new modification of the classical perceptron, which we also call the super-perceptron or supertron.

The main difference between the Omega neuron and the classic formal neuron is that at each synapse there is a set of two or more corrective weights and a distributor that selects one weight corresponding to some characteristic (for example, amplitude) of the input signal, and connects it to neuron. At each specific moment of training, only one weight chosen by the distributor works at the synapse, as in the classical ANN. But at different points in time, different corrective weights perform this role.

An important advantage of the proposed scheme is the presence of several corrective weights at each input, which breaks the chains of recursive feedbacks, reducing the influence of different weights on each other. The absence of a non-linear activation function in PANN also contributes to the weakening of the influence of feedbacks.

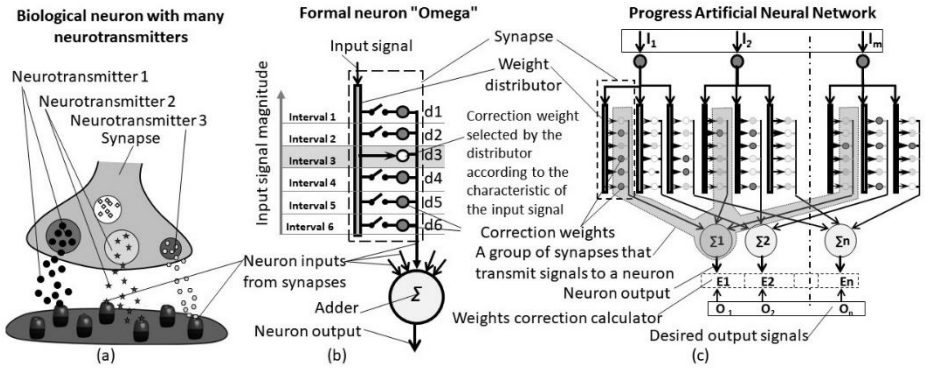

Fig. 2. The principle of building a PANN network: (a) a biological neuron with several neurotransmitters; (b) a formal neuron "Omega"; (c) the PANN network.

This network construction allows:

- Training the weights of each neuron separately, neglecting the influence of the weights of different neurons on each other.
- Eliminating the small steps of training using the gradient descent method and conduct training for the entire value of the error - the difference between the actual and desired values of the neural sums.
- Providing batch training of the neural network at once by whole images containing a large number of pixels rather than by individual pixels - inputs, as is the case with most existing classical networks.

Omega neuron can be implemented in different ways: on a CPU, on graphics cards, specialized or programmable microchips, analog devices, etc.

3 PANN Batch Training

We present a batch training algorithm for a PANN used for image recognition.

Step 1. Preparation

1.1. Selection of basic network parameters.

- The number of network inputs *h* is selected according to the selected image format. For example, for 32×32-pixel images, the number of inputs *h* will be 1024.
- The number of neurons *n* for a single-layer network is equal to the number of outputs. In the simplest case, it can be equal to the number of images selected for training *m* (below we will consider this particular case). But the choice of this number does not really matter, since the network can always be increased by introducing both additional neurons without disrupting the training already carried out.
- The number of levels of correcting weights (intervals into which the range of magnitudes of input signals is divided) must be at least 2. The maximum number is not limited, but, as practice has shown, these numbers are rather small. For example, it was found that for 32×32 image format, 6 - 8 levels are optimal.

1.2. Formation of a set of matrices describing the data on the training sample images and the network structure.

– Input image matrix ***I*** is a one-dimensional vector of length *h* that includes all image pixels in the format selected for analysis

$$\boldsymbol{I}=(I_1, I_2 \ldots I_h).$$

– The commutation matrix of the corrective weights ***C*** (Table 1), determining which of the correction weights corresponds to each pixel of the input image. The number of columns is equal to the product of the number of inputs *k* by the number of levels of weights *d, (k×d)* and the number of rows is equal to the number of outputs *n* equal to the number of neurons. For the convenience of notation, each element of the matrix ***C*** is denoted by three indices $i=\overline{1,n}$, $j=\overline{1,h}$ и $k=\overline{1,d}$, and, that is, the position in the row is denoted by the number of the output *i* and a pair of indices: the number of the input *j* and the number of the corresponding interval *k*.

Table 1. Commutation Matrix ***C***

	Input 1				**Input 2**				**…**	**Input *h***			
	Intervals				**Intervals**				**…**	**Intervals**			
	1	2	…	*d*	1	2	…	*d*	…	1	2	…	*d*
Output 1	$C_{1,1,1}$	$C_{1,1,2}$	…	$C_{1,1,d}$	$C_{1,2,1}$	$C_{1,2,2}$	…	$C_{1,2,d}$	…	$C_{1,h,1}$	$C_{1,h,2}$	…	$C_{1,h,d}$
Output 2	$C_{2,1,1}$	$C_{2,1,2}$	…	$C_{2,1,d}$	$C_{2,2,1}$	$C_{2,2,2}$	…	$C_{2,2,d}$	…	$C_{2,h,1}$	$C_{2,h,2}$	…	$C_{2,h,d}$
…	…	…	…	…	…	…	…	…	…	…	…	…	…
Output *n*	$C_{n,1,1}$	$C_{n,1,2}$	…	$C_{n,1,d}$	$C_{n,2,1}$	$C_{n,2,2}$	…	$C_{n,2,d}$	…	$C_{n,h,1}$	$C_{n,h,2}$	…	$C_{n,h,d}$

It is essential that in each row for each input there is only one unit that determines the use of this particular weight, and all other values are equal to zero.

– The matrix of corrective weights ***W*** is constructed similarly to the matrix ***C***, it has the same dimension and indexing of elements.

– The matrix of the desired output image ***O*** is a one-dimensional vector including the desired output signals corresponding to all the images selected for training

$$\boldsymbol{O}=(O_1, O_2 \ldots O_n).$$

Step 2. Calculation of the matrix of neural sums ∑

The formation of the matrix of neural sums ∑ at each of the neural outputs is performed by multiplying the value of each corrective weight of the matrix ***W*** by the corresponding coefficient $C_{i,j,k}$ from the commutation matrix ***C*** and then adding the values of all weights affecting this neuron. This is done by matrix multiplication of ***C*** and $\boldsymbol{W}^{\boldsymbol{T}}$ (transposed matrix ***W***):

$$\sum = \boldsymbol{C} \times \boldsymbol{W}^{\boldsymbol{T}}.$$

Neural sum matrix ∑ is square, with the number of columns and the number of rows equal to the number of network outputs. Given that all rows in it are equal, for further operations, the matrix is simplified by deleting all rows except one.

Step 3. Calculation of recognition errors *E*

The error of the neural sum for each neuron is equal to the difference between the given value of the desired output signal O_i and the neural sum for this neuron, obtained as a result of step 2. The vector of errors in the recognition of neural sums $\boldsymbol{E}$ is formed by subtracting the vector of neural sums $\sum$ from the vector of the desired output signals $\boldsymbol{O}$:

$$\boldsymbol{E} = \boldsymbol{O} - \sum.$$

Step 4. Calculation of the total correction
The total correction to all weights contributing to the neural sum on each neuron is determined as the quotient of dividing the error of the neural sum on a given neuron by the number of weights contributing to this neural sum, which is equal to h

$$\Delta \boldsymbol{W} = \boldsymbol{E} / h.$$

Step 5. Building a matrix of corrected weights
The training of the system is reduced to adjusting the weights for each neuron by adding a total correction to the existing weights. This operation immediately in one step reduces the errors in the recognition of neural sums for a given image to zero. The corrected value of the weights of the i-th neuron at iteration $t + 1$ is

$$W_i^{t+1} = W_i^t + \Delta W_i.$$

Step 6. Training other images
The remaining images are trained by sequentially repeating steps 2 - 5 for all images. This concludes the first training epoch.
Step 7. Reducing the error
The training of each next image changes some weights that form the previous images, thereby creating new recognition errors on them. If these errors are higher than predetermined permissible value (for example, 0.01), it is necessary to spend additional training epochs, by repeating steps 2 - 5 for all images until the error becomes acceptable.

4 Results of testing PANN networks

The methodology described above for the formation and training of the PANN network was implemented and tested on a regular laptop with the following characteristics: CPU Intel i3-7100U 2.40GHz, RAM 4Gb, OS Microsoft Windows 10 Pro, built-in video card.

Key test findings

- Tests have shown that PANN training epoch is hundreds of times shorter, and in the case of large networks, thousands of times shorter than for classic networks, while the number of training epochs also drops hundreds and thousands of times. Fig. 3a shows the results of training 10 images of 32×32 pixels in the form of the training error dependence on the number of epochs. After the third epoch, the training error becomes significantly less than 1%.

- Fig. 3b shows the experimental curve of the dependence of the training time on the size of the training sample of images, provided that the number of images and the number of neurons is equal, and the training accuracy is less than 1%. The growth of training time with an increase in the number of trained images turned out to be quadratic and rather slow $T = 0.003n^2$, in contrast to the exponential growth characteristic of classical networks. Training converges with a given accuracy in a small number of epochs, also shown in Fig. 3b. In the case of the network with a limited number of neurons, the growth of training time becomes almost linear with the increasing number of images.
- The recognition accuracy of both trained and untrained, but close to trained images by the PANN network is close to the recognition levels of existing neural networks.
- The following assumptions about PANN networks have been confirmed:

— possibility of additional training and unrestricted transitions from recognition to training and back;
— activation function is unnecessary;
— scalability;
— possibility of additions and restructuring during use.

- The possibility of creating generalized images and improving the recognition accuracy using the image voting procedure is shown.
- During testing, despite special efforts, it was not possible to create situations of freezing and overfitting of the network.

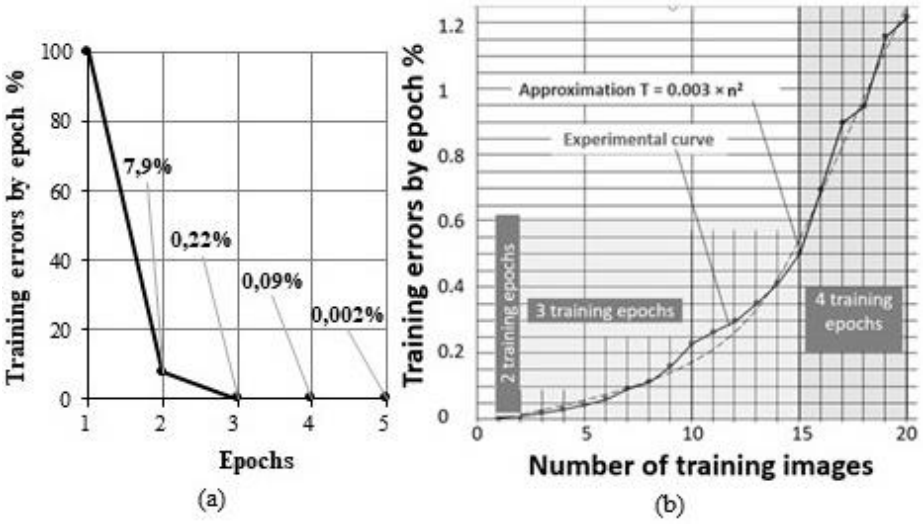

Fig. 3. PANN network test results: (a) - dependence of the training error on the number of epochs; (b) - dependence of training time on the number of input images

Those who wish to get acquainted with the results can find test data on the portal https://www.omega-server.ai/. One can also conduct independent testing of the PANN and train the network with ones' own data for a non-commercial application.

5 Conclusions

Studies of the implementation and training of the PANN network in the form of a simplest perceptron confirm that a significant part of the problems and difficulties in the development of artificial neural networks are the result of Rosenblatt's erroneous doctrine of "one synapse - one synaptic weight". Still, it is quite obvious that emergence of the PANN that corrects the Rosenblatt doctrine does not undermine the brilliant developments in neural networks over the past decades. The simplest PANN network, a supertron, is not a specialized intelligent system, in many respects it is close to the classical perceptron, ready for many practical applications. And, undoubtedly, the next step in the development of PANN will be network hybridization with the best advances in the development of specialized neural networks, the creation on their basis of new advanced versions of multilayer perceptrons, cognitrons, Hopfield, Kohonen networks, deep learning systems, recurrent, resonant, and convolutional networks.

Today PANN networks are at the very beginning of their evolutionary journey. We should expect the emergence of special types of hardware for PANNs based on specialized video cards, reprogrammable microchips, specialized microchips, possibly also various analog systems.

One can also expect the replenishment of object-oriented programming libraries with objects made on the basis of the PANN network, the PANN application in systems such as Internet agents, Secretary, Adviser, in database management systems, for emulation of various software, including cellular automata, Pearl causal machines, hidden Markov networks, etc.

And, obviously, over time, new specialized networks and unknown today applications will appear based on PANN, in particular - advanced artificial intelligence systems.

References

1. McCulloch, W., Pitts, W. A Logical Calculus of Ideas Immanent in Nervous Activity. Bulletin of Mathematical Biophysics 5, 115-133 (1943).
2. Hebb, D.O. The Organization of Behavior. John Wiley & Sons, New York (1949).
3. Rosenblatt, F. Principles of neurodynamics: Perceptions and the theory of brain mechanism. Spartan Books, Washington DC (1961).
4. Minsky, M., Papert, S. Perceptrons: an introduction to computational geometry, MIT Press., Cambridge, MA (1969).
5. Shepherd, G. Foundations of the neuron doctrine. Oxford University Press (1991).
6. Finger, S. Origins of neuroscience: a history of explorations into brain function. Oxford University Press (2001).
7. Pescianschi, D., Boudichevskaia, A., Zlotin, B. and Proseanic, V. Analog and Digital Modeling of a Scalable Neural Network, CSREA Press, Las Vegas US (2015).
8. Patent US 9390373; 2016 Neural network and method of neural network training.
9. Patent US 9619749; 2017 Neural network and method of neural network training.
10. Patent US 10423694; 2019 Neural network and method of neural network training.