https://doi.org/10.59849/aidd.2024.3

Analytically Computable Neural Networks Perceptron Type

Heydarov Polad Shahmaly
Institute of Control Systems, Baku, Azerbaijan,
E-mail: plbaku2010@gmail.com

Abstract. The paper describes the main provisions of the developed technology for analytical calculation of the values of weights of neural networks perceptron type. Analytical formulas are given by which the values of the weights are calculated. The basic proposed architecture of a perceptron-type neural network is presented, on the basis of which the values of the perceptron weights can be calculated. The work also presents numerous experimental results that show the capabilities and advantages of analytical calculation of perceptron weights. In particular, experiments have shown that a perceptron with analytically calculated weight values learns much faster and is more resistant to a decrease in the size of the training database.

Keywords: neural networks, convolutional neural networks, backpropagation algorithm, perceptron, calculation of weights

Currently, artificial neural networks are widely used throughout the world and are the main component for artificial intelligence technologies. Despite this, modern neural networks have limitations and disadvantages, such as: the need for a huge amount of training database to train a neural network, lengthy training, lack of transparency of the neural network architecture, and others. This paper describes the main provisions of the developed technology for analytical calculation of the values of neural network weights, which allows one to bypass these limitations and difficulties.

The weights are calculated based on the selected small set of selected images. For example, for a fully connected neural network, the values of one weight of the first layer are calculated using formulas (1), (2) based on Figure 1 [1].

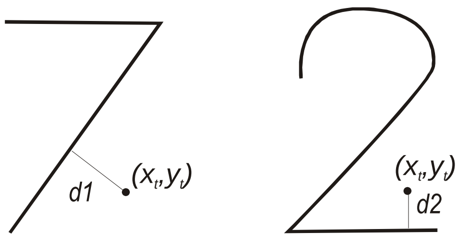


Fig.1 Distances d_1 and d_2 for point (x_t, y_t).

Here for the point (x_t, y_t) the shortest distances to each image will be equal to d_1 and d_2. Accordingly, the weight w_{ij} for this point will be defined as:

$$w_{ij} = d_1^2 - d_2^2, \tag{1}$$

where d_1^2 and d_2^2 - replaces $\left(y_{etal1\,j} - y_{testij}\right)^2$, $\left(y_{etal2\,i} - y_{testij}\right)^2$ and are determined [1] respectively by the expressions:

$$d_1^2 = (x_1 - x_t)^2 + (y_1 - y_t)^2, \tag{2}$$
$$d_2^2 = (x_2 - x_t)^2 + (y_2 - y_t)^2,$$

where (x_1, y_1), (x_2, y_2) are the coordinates of the nearest active pixels for two images.

The architecture of the proposed perceptual type incompletely connected neural network is shown in Fig. 2 [1-5]. The value of all weights of the second and third layers for this neural network is equal to 1. This neural network can also be easily converted into a fully connected multilayer perceptron neural network.

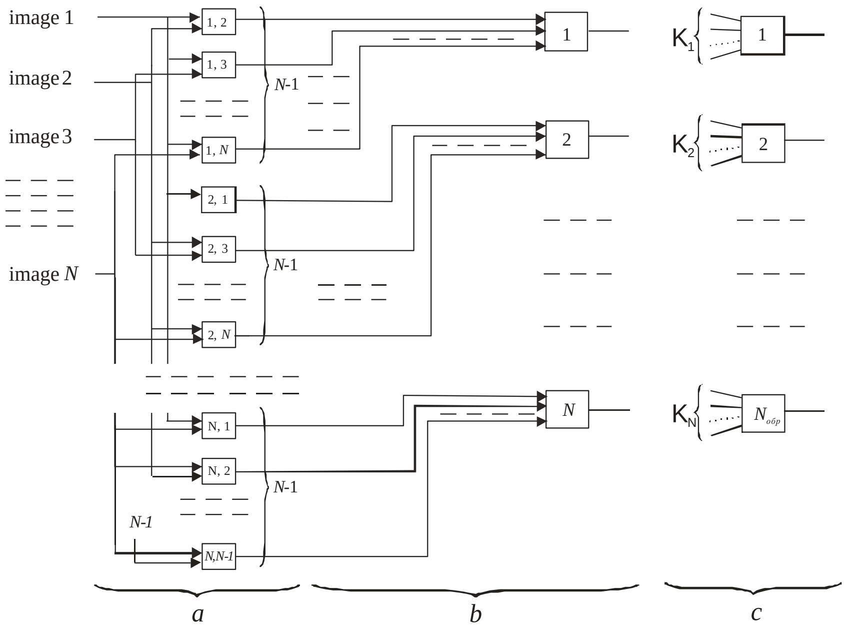
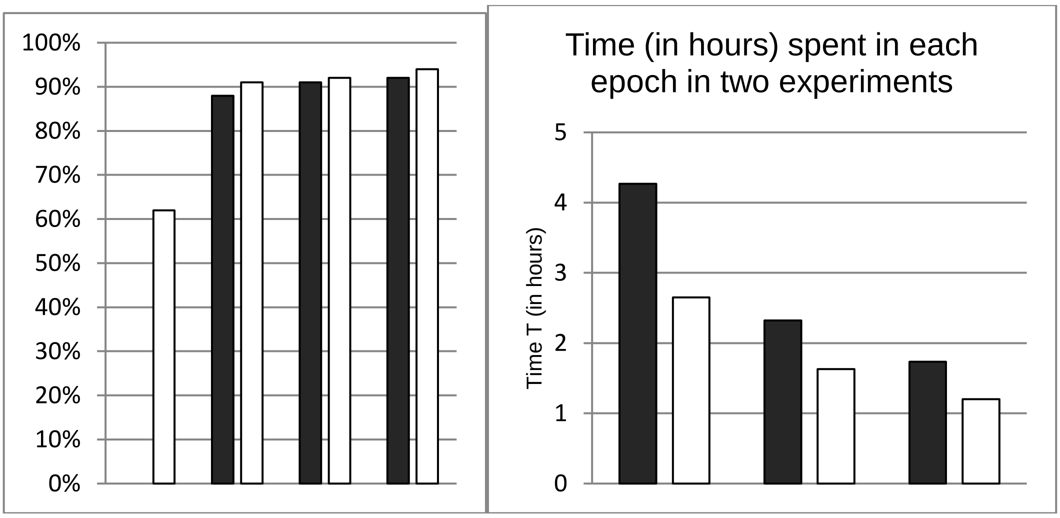


Fig.2. Three-layer neural network based on metric recognition methods.

The performance of this technology was tested on the well-known MNIST digit database, which consists of a 60,000 training database and a 10,000 control digit database. Two comparative experiments were carried out on training a three-layer perceptron with calculated weights and training a neural network with random generation of weights [2]. Training was carried out over three epochs. 30 images were randomly selected as reference images. In Figure 3 you can see the results of these experiments. Analytical calculation of weights without training made it possible to immediately achieve 62% (Fig. 3*a*). Training a neural network with calculated weights allowed us to speed up the neural network training procedure by 33% (Fig. 3b), and increase the effectiveness of the neural network by 2% (94%-92%).



a *b*

Fig.3. (*a*) Plot of the percentage of correctly recognized images of the MNIST benchmark, (*b*) Plot comparing the time spent per epoch for the two experiments.

It was also shown that the values of the weights can be calculated not only using formulas (1), (2) but also, for example, using formulas for electrostatic field parameters such as intensity and potential

(Fig. 4) [3, 4]. Weight values are calculated based on physical formulas for potential difference or intensity (Fig. 5).

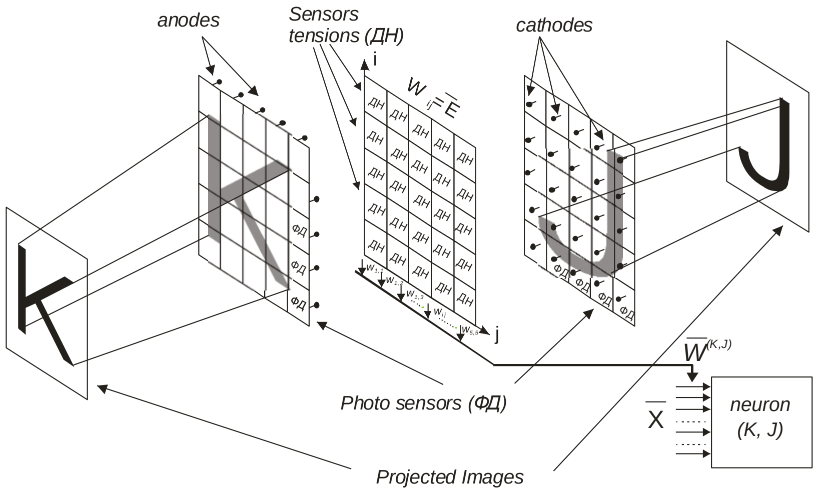


Fig.4. Block diagram demonstrating obtaining the weight values of the first layer neuron for projected images *"K, J"* using photosensors and tension sensors.

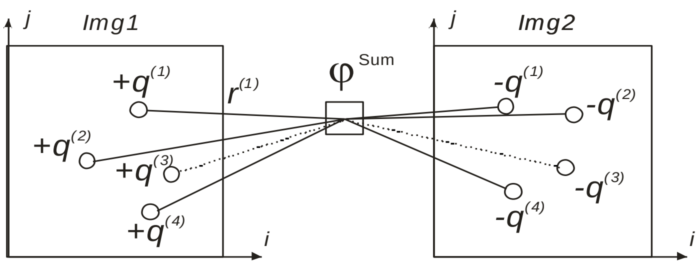


Fig.5. Scheme for determining the potential-weight in one cell of the weights table (plane of potentiometer sensors) of the first layer.

The results of the calculated neural network based on physical parameters are shown in Table 1 for the case of using only 50 images [4]. The recognition result without training reaches 70%. This approach shows that a neural network can almost instantly calculate the weight values of the neural network and immediately begin recognizing objects without training.

Table 1. Test results on the MNIST control base using 50 images.

s0 = 852	i0 = 980	p0 = 87%
s1 = 1123	i1 = 1135	p1 = 99%
s2 = 516	i2 = 1032	p2 = 50%
s3 = 585	i3 = 1010	p3 = 58%
s4 = 775	i4 = 982	p4 = 79%
s5 = 588	i5 = 892	p5 = 66%
s6 = 555	i6 = 958	p6 = 58%
s7 = 616	i7 = 1028	p7 = 60%
s8 = 633	i8 = 974	p8 = 65%
s9 = 726	i9 = 1009	p9 = 72%

10000, 6969, 70%	

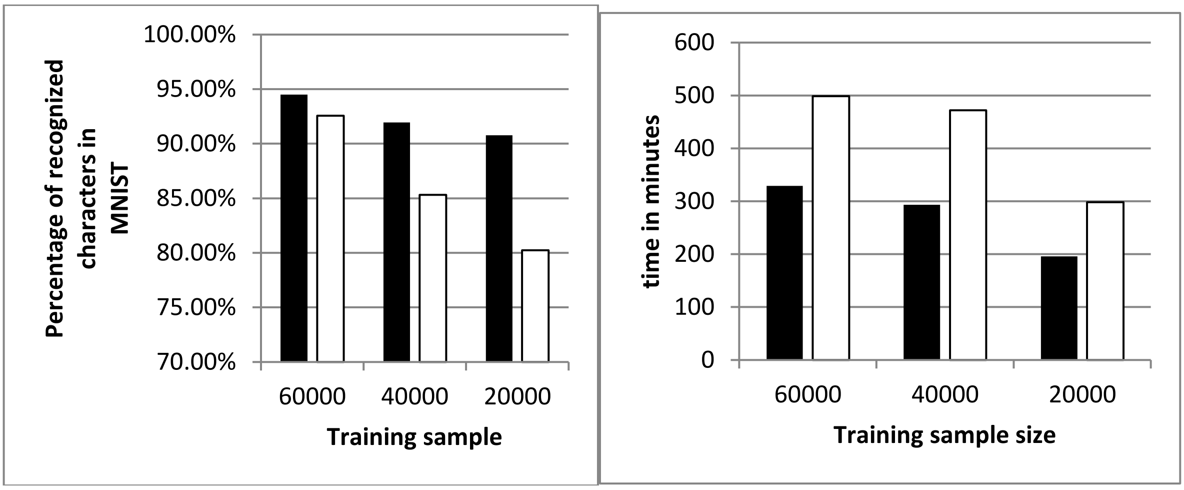


Fig. 6. Dependence of the percentage of recognized characters on the MNIST control base (*a*) and the training time of the neural network (*b*) with calculated weight values (black color) and random generation of weights (white color) on the size of the training sample.

In addition, it was shown that neural networks with analytically calculated weights require significantly less training sample than neural networks trained in the classical way. In Fig.6 shows that reducing the size of the training sample allows you to maintain the performance of the neural network within 90%-95%, when for a neural network with randomly generated values, training with small database volumes leads to a sharp decrease in recognition performance (Fig. 6) [5].

Calculation of neural network weights has also been implemented for a deep convolutional neural network. For example in Fig. 7 shows the analytically calculated kernel of one channel of the first convolutional layer of a convolutional neural network. Based on the obtained kernels, all channels of the convolutional neural network are calculated without using learning algorithms. In this case, the value of the weights of the fully connected neural network of the convolutional neural network is calculated in the same way as was shown above (1), (2).

0	0	0	0	0
0	0	0	0	0
0	0	0	0	255
0	0	0	255	255
0	0	255	255	255

-1	-1	-1	-1	-1
-1	-1	-1	-1	-1
-1	-1	-1	-1	1
-1	-1	-1	1	1
-1	-1	1	1	1

kernel for channel				
b = 170				
-0,3	-0,3	-0,3	-0,3	-0,3
-0,3	-0,3	-0,3	-0,3	-0,3
-0,3	-0,3	-0,3	-0,3	0,28
-0,3	-0,3	-0,3	0,28	0,28
-0,3	-0,3	0,28	0,28	0,28

a *b* *c*

Fig.7. Calculation of the feature kernel for the channel of the first convolutional layer, (*a*) - image feature, (*b*) - channel kernel without correction, (*c*) - kernel with corrections.

In Fig. 8. the selected features are shown in the reference images and in the test image based on the calculated values of the kernel weights. From Fig. 8 it can be observed that the selected features of the test image “2” are similar to the selected features of the reference image “2”. Which corresponds to correct recognition of the input image.

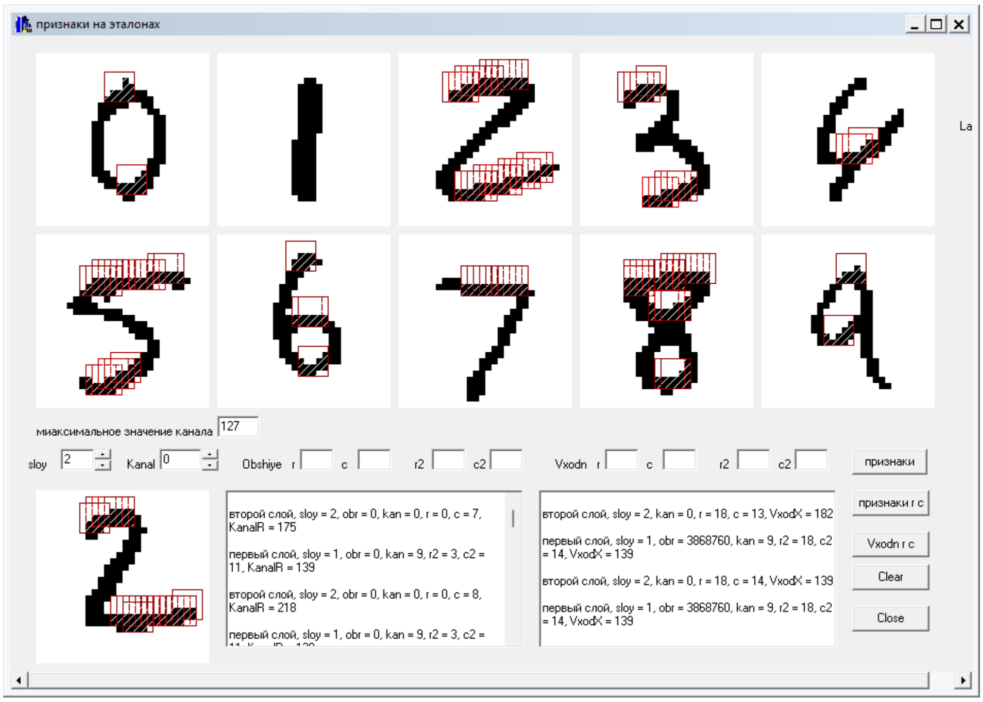


Fig.8. All features on all images and the input image “2”, corresponding to the first (zero) real channel of the second convolutional layer.

In Fig.9 shows the result of an analytically calculated convolutional neural network. The result was immediately 58% without performing neural network training. The number of layers of the convolutional neural network was 2 convolutional layers, 1 pooling channel and 3 layers of a fully connected neural network. In total, 6 layers were used in this network.

Thus, in the works [1-5] it was shown that the values of the weights of a multilayer perceptron can be calculated analytically, which makes it possible to immediately obtain a workable neural network. The resulting neural network can be trained using backpropagation algorithms. Training a perceptron with calculated weights requires significantly less time and significantly smaller volumes of training databases.

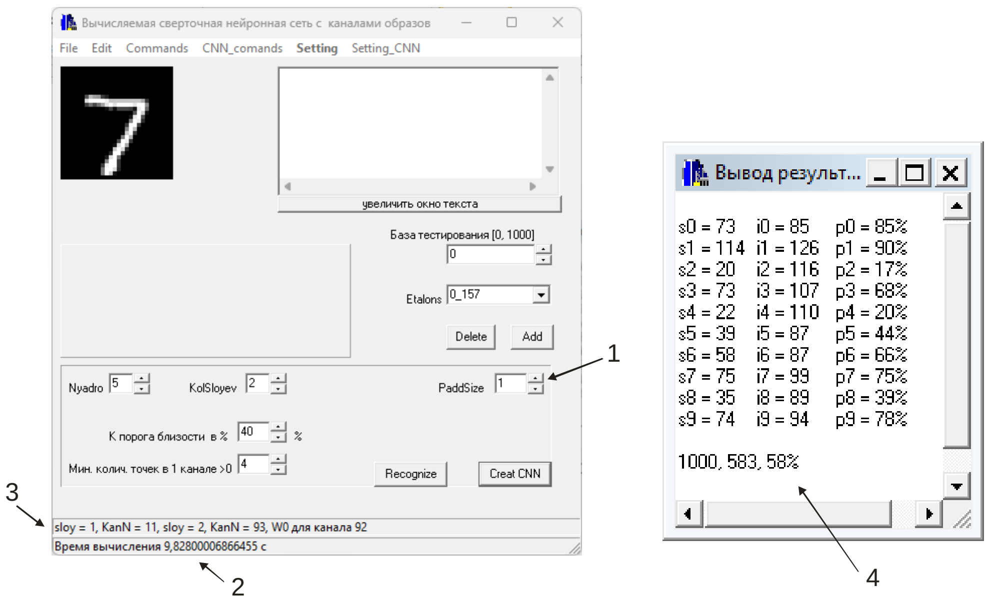


Fig.9. Creating a neural network and checking the performance of the resulting neural network based on the MNIST control database.

References

1. Geidarov, P.Sh. Neural networks on the basis of the sample method // Automatic Control And Computer Sciences,–2009. V. 43. No. 4,–p. 203-210. **Scopus.**
2. Geidarov, P. Sh. Comparative Analysis of the Results of Training a Neural Network with Calculated Weights and with Random Generation of the Weights // Automation and Remote Control, NewYork, - 2020. V. 81, no. 7, - p. 1211-1229. **Scopus, Web of Science (SCIExp).**
3. Geidarov P.Sh. On the Possibility of Determining the Values of Neural Network Weights in an Electrostatic Field \\ Scientific and Technical Information Processing, 2022, Vol. 49, No. 6, pp. 1–13. **Scopus, Web of Science (E SC I).**
4. Geidarov P.Sh. Experiment for Creating a Neural Network with Weights Determined by the Potential of a Simulated Electrostatic Field \\ Scientific and Technical Information Processing, 2022, Vol. 49, No. 6, pp. 1–13. **Scopus, Web of Science (ESCI).**
5. Geidarov P.Sh. A Comparative Analysis of a Neural Network with Calculated Weights and a Neural Network with Random Generation of Weights Based on the Training Dataset Size \ \ Optical Memory and Neural Networks, 2022, Vol. 31, No. 3, pp. 309–321. **Scopus, Web of Science (ESCI).**