Adv Syst Sci Appl 2025; **1**; 22-29
Published online at https://ijassa.ipu.ru.

Application of Modular Neural Networks for Image Recognition in Foggy Computing Environments

Nikolay Verskov*, Natalia Kuchukova

North-Caucasus Federal University, Stavropol, Russia

Abstract: The paper considers various approaches to the decomposition of artificial neural networks for the purpose of their application on fog computing nodes. Based on the requirements for the organization of fog computing, a method of dividing the input information into subspaces by means of wavelet transform and subsequent proportional division of all layers of the neural network is proposed. The proposed approach achieves a significant gain in the amount of information transferred between modules compared to the currently used layer-by-layer partitioning. In addition, the proposed method optimizes the load on fog computing nodes by partially utilizing the modules.

Keywords: artificial neural networks, image recognition, fog computing, wavelet transform, proportional division of layers of the neural network.

1. INTRODUCTION

Today, the IoT is becoming a significant part of society and the volume of connected devices is growing rapidly. Existing approaches based on cloud computing technologies are unable to provide the required quality of service due to the increasing latency of data transmission [10]. Therefore, more and more attention is being paid to the fog computing (FC) model. The FC model involves placing computing power at the edge of the network [11], i.e., close to the hardware in use [8]. This approach allows to significantly reduce the cost of data transmission, as well as to increase the privacy of user data.

More and more tasks, such as virtual assistant, computer vision and object recognition, require the use of artificial neural networks (ANNs). Highly accurate ANN models have significant volumes and require large computational resources. With the cloud computing model, there is no problem in using ANNs, computing resources in the cloud are sufficient. However, the shift in focus to fog computing requires the placement of ANNs on FC nodes, whose computing power is much lower than cloud computing. In this regard, multiple FC nodes are required to be used simultaneously to support the high-fidelity ANN model.

Application of ANN is usually divided into 2 stages: training and intended use. The growth of requirements to model accuracy is directly related to the growth of the number and volume of ANN layers, which, in turn, requires large expenditures of computational resources that FC nodes do not possess. To solve the emerging contradiction, different approaches are used today:

1. The training process is performed on the cloud server and the ANN is deployed on FC nodes only directly for the intended application [9].
2. Utilizing the client-server computing paradigm [12], which requires infrastructure support.
3. ANN model compression [12], which requires infrastructure support or special training steps.
4. Using a distributed programming model such as MapReduce [3].

* Corresponding author: vernick61@yandex.ru

5. Using mathematical methods of dividing the input information into modules with subsequent separate processing of each module at FC nodes [19].

Approach 1 is widely applicable and can be used if the computational power of the FC node is sufficient for ANN operation. This approach is not modular and allows to perform the most energy-intensive training operation on the server. Approach 2 is also non-modular and moves energy-intensive operations from the FC domain to the cloud server. This approach does not solve the problem of service time delay and is actually a veiled form of cloud computing. Model compression (approach 3), as a rule, requires additional costs for ANN optimization, but allows to obtain a performance gain of 20-30% compared to approach 1. The use of distributed programming model (approach 4) is the most popular today. This approach allows to divide the ANN into layers (by one or other attribute) and execute one or more layers on a separate FC node [2, 12]. A significant disadvantage of this approach is the heavy load on data transmission channels, since the amount of information transferred between layers is very large. At the same time, this approach allows training and intended use of ANN without accessing the cloud server. Approach 5 has a significant advantage over approach 4: with its help, the ANN is divided into modules not along the layers, but across, i.e., all the layers of the module reduce their volume so that the computational power of the FC node is sufficient for its functioning [19]. Due to the fact that almost all module layers are on the same node, the channel load is significantly reduced. Another advantage of the proposed approach is the non-uniform distribution of features within modules, i.e., the use of a smaller number of modules only slightly reduces the recognition accuracy of ANN and allows optimizing the load of FC nodes.

One of the key characteristics of FC is scalability and dynamism [7], i.e. FC should be adaptive in nature. Adaptability, in turn, implies support for the following key mechanisms: elasticity of computation, pooling of resource capabilities, adjustment to changes in data load and changes in network composition [10]. This paper is devoted to the use of the latter approach for building scalable and dynamic ANNs on FC nodes.

2. JUSTIFICATION OF USING WAVELET TRANSFORM TO CREATE MODULAR ANNS FOR FC

The proposed approach is based on a well-known concept in the theory of information transmission - power spectral density (PSD). If $x(t)$ is a periodic signal with period T, then the average power per period can be defined as

$$P = \frac{1}{T}\int_{-T/2}^{T/2} x^2(t)dt.$$

Parseval's theorem for a real periodic signal $x(t)$ has the form

$$P = \frac{1}{T}\int_{-T/2}^{T/2} x^2(t)dt = \sum_{n=-\infty}^{\infty} |c_n|^2, \tag{2.1}$$

Where c_n are the complex coefficients of the Fourier series. The power spectral density $P(f)$ of a periodic signal $x(t)$, which is a real, even and non-negative function of frequency, gives the distribution of signal power over the frequency range, is defined as follows:

$$P(f) = \sum_{n=-\infty}^{\infty} |c_n|^2 \delta(f - nf_0) \tag{2.2}$$

Here f_0 is the center (average) frequency.

The input values of the ANN can be considered as a discrete periodic signal, since all input values have the same digit capacity, and the period of the signal coincides with the digit capacity of the ANN input. This discrete periodic signal at the ANN input defines a

finite set of classifiable images. Differences in images belonging to the same cluster can be considered as additive noise on the ideal (averaged) image.

It follows from Riemann's theorem that the Fourier coefficients c_n of the function $f \in L_R(-\pi,\pi)$ tend to zero when $n \to \infty$ [17], i.e., the partial sums $P_k(f), k \in \left[n_1^k, n_2^k\right]$, of expression (2) will also decrease as n increases. This means that if we divide the spectrum of the signal at the ANN input into 2 equal parts, the PSD in the left (low-frequency) part will be higher than in the right one. Based on (2.1), it can be stated that the power of features in the signal representing the low-frequency part will be higher than in the high-frequency part.

In [4] it is proved that in the presence of nonlinearity in ANN, it is possible to construct connections and select coefficients between neurons in such a way that ANN can compute any continuous function from its inputs with any accuracy. Consequently, an ANN trained with the low frequency part of the signal will perform image recognition with higher accuracy than an ANN trained with the high frequency part of the signal.

The Fourier transform of a numerical sequence is a 2π-periodic function. Therefore, when one speaks of low and high frequencies, one means close to zero or ±π, and the sampling step $\Delta t = 1$. In this case, the Kotelnikov (Nyquist) frequency is equal to π [1, 16]. For ideal low-pass $H(\omega)$ and high-pass $G(\omega)$ filters, the transfer characteristics are described by expressions:

$$H(\omega) = \begin{cases} 1, \omega \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right], \\ 0, \omega \in \left[-\pi, -\frac{\pi}{2}\right) \cup \left(\frac{\pi}{2}, \pi\right] \end{cases}, G(\omega) = \begin{cases} 0, \omega \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right] \\ 1, \omega \in \left[-\pi, -\frac{\pi}{2}\right) \cup \left(\frac{\pi}{2}, \pi\right] \end{cases}. \quad (2.3)$$

From expression (2.3) it is not difficult to find the coefficients of ideal filters (by performing the inverse Fourier transform):

$$h = \begin{cases} 1/2, n = 0 \\ 0, n = 2k \\ \frac{(-1)^k}{(2k+1)\pi}, n = 2k+1 \end{cases}, \; g = \begin{cases} 1/2, n = 0 \\ 0, n = 2k \\ \frac{(-1)^{k+1}}{(2k+1)\pi}, n = 2k+1 \end{cases}. \quad (2.4)$$

When the signal is decomposed into high- and low-frequency components $\{x_n\} \to \{x_n^g\}, \{x_n^h\}$ the number of samples doubles, but using the Kotelnikov-Shenon theorem it is possible to discard half of the samples [16]. This process is called decimation. Thus, to realize the process of dividing the input sequence into high- and low-frequency parts it is enough to perform its convolution with coefficients of ideal filters followed by decimation.

Consider 2 Haar filters with coefficients [1, 6, 16]:

$$h_0 = \frac{1}{2}, h_1 = \frac{1}{2}; g_0 = \frac{1}{2}, g_1 = -\frac{1}{2}.$$

The convolution of the low-pass filter kernel $\{h_i\}$ with the input sequence $\{x_j\}$ will result in the following sequence

$$a_n = \sum\nolimits_{k \in Z} h_k x_{n-k} = \frac{1}{2} x_n + \frac{1}{2} x_{n-1} \quad (2.5)$$

and convolution of the high-pass filter kernel $\{g_k\}$ with the input sequence $\{x_j\}$ the sequence will be obtained

$$d_n = \sum_{k \in Z} g_k x_{n-k} = \frac{1}{2} x_n - \frac{1}{2} x_{n-1} \tag{2.6}$$

The sequence a_n, defined by expression (2.5), is obtained as the arithmetic mean of two neighboring values of $\{x_j\}$, reducing the oscillation frequency, while the sequence d_n (2.6), being essentially the first derivative of the input values of $\{x_j\}$, reflects its oscillations. Thus $x_k = a_k + d_k$, i.e., the input values $\{x_k\}$ have been decomposed into a low-frequency part a_k and a high-frequency part d_k and can be recovered by a postal addition of the components.

Turning to the wavelet transform, we define in the space $L^2(R)$ the orthonormalized Haar basis [16] as

$$\varphi(x) = \begin{cases} 1, x \in [0,1) \\ 0, \notin [0,1) \end{cases}. \tag{2.7}$$

In [16] it is shown that the spaces V_j, generated by the system of functions

$$\varphi_{j,n} = \sqrt{2^j}\varphi(2^j x - n), \tag{2.8}$$

created on the basis of the parent function (2.7), form an infinite system of nested subspaces $V_0 \subset V_1 \subset V_2 \subset \ldots$. This system of subspaces can be used to go from an arbitrary function $f(x)$ from $L^2(R)$ to its more or less exact approximation in the space $V_j \in Z$, the spaces V_j are being scaled versions of the space V_0.

Thus, the use of scaled versions of spaces for ANN training will allow to create modular neural networks with cross-layer separation [19], since the dimensionality of spaces decreases proportionally to the level of nesting due to decimation. At the same time, the dimensionality of the ANN module can be reduced in proportion to the level of nesting. In this case, the accuracy of space approximation V_j determines the classification accuracy of the ANN module from more accurate (low-frequency part) to less accurate (high-frequency part). The ANN modules can be used separately, for example, for optimization (dimensionality reduction) of ANN, or together. A single linear layer was used to combine the results of the modules in [19]. However, the Haar wavelet transform assumes an inverse transform [6], but due to the nonlinearity of the ANN, the inverse transform kernel cannot be used, so the inverse transform layers need to be trained.

3. EXPLORING THE POTENTIAL OF A MODULAR CROSS-LAYER PARTITIONED ANN FOR USE IN FC

The task of using ANN at FC nodes can be divided into several subtasks: wavelet transformation of the input signal with decimation, division of the initial ANN into modules (according to the number of input information module), training of the network and its application as intended.

Thanks to the McCulloch-Pitts model [13], it has become possible to realize spectral decomposition operations in direct propagation ANNs and convolutional layers for almost any, including random signals. Here, the most important feature is the correlation radius of the information field [15]. Most conveniently, this operation can be performed using convolution layers, which allow to perform, for example, wavelet transform simultaneously with several kernels. This operation is considered in detail in [20]:

$$\begin{cases} x(n) * h(n) = \sum_k x(k)h(n-k) \\ x(n) * g(n) = \sum_k x(k)g(n-k) \end{cases},$$

If the signal at the input of the ANN is a one-dimensional sequence, one convolution layer with kernels $h(n)$ and $g(n)$ is sufficient to perform the Haar wavelet transform. In image processing, the input signal is a two-dimensional matrix, and 3 convolution layers will be required to perform the two-dimensional Haar wavelet transform (Fig. 1).

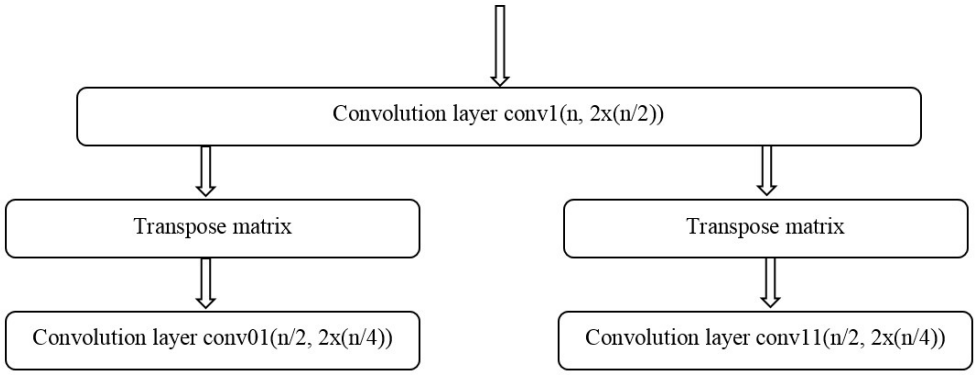

Fig. 1. Two-dimensional wavelet transform

As noted earlier, the division of ANN into modules is most often performed layer-by-layer [18]. In this case, the volume of the ANN does not change, and the lack of computing resources is compensated by using several computing devices connected in series with the placement of several layers of the network on each of them. The authors rely on the thesis that it is impossible to divide one layer into 2 or more computing devices [18]. As it was shown in [19], such separation is possible not at the level of the ANN layer, but at the level of input information. It was proposed to divide the input information into several modules and to carry out joint or separate processing of modules on ANNs, proportionally reduced compared to the original network. The authors in this paper set out to compare the performance of the approaches proposed in [18] and [19] for FC nodes.

For this purpose, 2 sets were used: the simplest one, MNIST [14], for visual demonstration of comparative performance and a much larger set GTSRB [5] (German Traffic Sing Recognition Benchmark) for working with AlexNet-like network. For the first experiment, an ANN with two convolution layers and a classifier of two linear layers was used on a personal computer running the "Windows 10 Home" operating system with an Intel© Core™ i7-10510U processor and 16 GB of RAM.

Each of the layers was placed in a separate ANN, which were placed sequentially, i.e. the output of the previous ANN is the input of the subsequent one. Thus, the entire network was partitioned layer by layer and trained for later use at the FC nodes. It is clear that with such an architecture, it is impossible to remove at least one ANN, because in this case a complete retraining is required. For comparison, a similar ANN was divided across the layers into 4 modules and trained. At the same time, 3 convolution layers were added to the ANN architecture to perform the two-dimensional Haar wavelet transform to the first layer and 3 layers at the end of the network for the inverse transform. The training results are presented in Table 1.

Table 1. Training results of ANNs with longitudinal and cross-layer separation

	Maximum training cycle time	Minimum training cycle time	Average training cycle time	Mean square deviation
Separation of ANNs along the layers	0.21533	0.14007	0.16213	0.01076
Separation of ANNs across layers	0.21266	0.14084	0.16620	0.00917

Table 2 summarizes the amount of data transferred between modules.

Table 2. Dimensionality of data transferred between modules

	INS input	1 module output	2 module output	3 module output	4 module output
Separation of ANNs along the layers	28x28x100	32x14x14x100	3136x100	1000x100	10x100
Separation of ANNs across layers	28x28x100	4x196x100	4x10x100		10x100

Table 1 shows that the time characteristics of both types of architectures are very close. Taking into account that there are 6 more layers in the ANN with cross-layer partitioning (forward and inverse two-dimensional wavelet transform), the difference in the average training time by 0.004 sec does not look critical. It became possible due to parallelization of operations between modules. Here, it should be noted that the number of training cycles in ANN with cross-layer partitioning turned out to be ≈1.5 times greater than in ANN with layer-by-layer partitioning: 314 vs. 192 cycles. This is quite understandable, since in addition to the training of 4 modules, 3 layers of the inverse two-dimensional wavelet transform are subject to training. As noted earlier, introduction of nonlinearity into ANN leads to impossibility of the inverse transform using the standard kernel.

Table 2 gives an idea of the amount of data transferred between modules. With layer-by-layer partitioning of the ANN, the amount of data transferred is proportional to the size of the layer (the size of the batches is the same and equals 100). At the same time, the amount of data to be transmitted through communication channels is 8 times less at cross-layer separation of ANN after the first layer, 78.4 times less after the second layer, and not required after the third layer. At the same time, the number of modules in cross-layer partitioned ANN is much larger: 6 vs. 4, since 2 more modules are required to perform forward and inverse wavelet transform.

But the biggest advantage in cross-layer partitioning of ANNs is that not all 4 modules can be used in network operation, but for example 2 or 3 (Table 3).

Table 3: Comparison of recognized quality and execution time using different number of modules

	1 module	2 module	3 module	4 module
Recognition quality, %	96.08	97.18	97.72	98.11
Average turnaround time, sec.	0.014	0.025	0.037	0.047

During the experiment, zero tensors were fed instead of the output data of the missing modules. At the same time, once trained ANN does not require retraining if 2, 3 or 4 modules are used depending on the availability of FC computational resources.

From the conducted experiment the advantages of cross-layer separation of ANN become clear: firstly, the amount of transferred information between layers is tens of times less than at layer-by-layer separation, secondly, the possibility of using not all trained modules, which allows to reduce the load on FC nodes at some reduction of recognition accuracy. The disadvantages of the proposed method include the need for two additional modules: to perform the wavelet transform and to combine the data using linear or convolution layers.

The following libraries for the Python 3.7 (64 bit) compiler: Numpy (v. 1.21.4), PyWavelets (v. 1.3.0), PyTorch (v. 1.12.1) were used for the experiment using the GTSRB dataset [5]. The experiment was conducted on a server running the Linux 2operating system, CPU E5-2690V4 processors, 1 TB of RAM. The purpose of the experiment was to evaluate the feasibility of the proposed cross-layer separation method for volumetric high-precision AlexNet-like ANNs, as well as to assess the advantages of using linear and convolutional layer modules for combining the results of the linear and convolutional layer modules.

The German Traffic Sign Recognition Test (GTSRB) includes 43 different types of road signs divided into 39,209 training and 12,630 test color images. The pictures depict a variety of lighting and environments. To reduce the volume of ANN, the authors allowed themselves

to convert color images of signs into grayscale. Such approach allowed, on the one hand, to reduce the volume of ANN due to the lack of necessity to process each color, on the other hand, increased the complexity of the problem due to the lack of 3 processing channels.

Table 4 shows comparative data on the recognition quality and performance of two AlexNet-like networks, one of which used a linear layer as a unifying layer and the other used 3 convolutional layers simulating the inverse Haar wavelet transform with the possibility of training kernels.

Table 4. Comparison of the features of the unifying linear and convolutional layers

	1 module	2 module	3 module	4 module
Recognition accuracy, line layer, %	78.57	96.24	99.38	99.49
Recognition accuracy, convolution layer, %	99.23	99.29	99.31	99.34
Recognition time of test images, line layer, c	14.8	21.84	28.47	35.65
Recognition time of test images, convolution layer, c	15.04	21.93	28.32	35.35

Table 4 shows that the temporal characteristics of both architectures are very close and it can be argued that there are no advantages. But the recognition quality of the ANN with a convolutional layer showed a significant advantage: even 1 low-frequency module performs recognition with a quality almost as good as a full network of 4 modules. Application of linear layer shows a significant decrease in recognition quality - by more than 20%. And only 3 modules out of 4 show quality comparable to full ANN. However, it should be noted that this approach does not always give such an advantage. A similar experiment on the MNIST dataset showed almost the same drop in recognition quality when the number of modules decreases for ANNs with linear and convolutional combining layer. Obviously, such an effect appears at significant redundancy of the network and/or input data.

CONCLUSION

In this paper, methods of partitioning ANNs into modules were investigated. The proposed method of cross-layer partitioning of ANN and further modular application showed a significant advantage over the currently used layer-by-layer partitioning, primarily in terms of the volume of data transferred between modules. In addition, the proposed method allows to use not all trained modules at the same time, but only a part of them. In this case, no additional training of the network is required when excluding a part of modules. The layer-by-layer separation of ANNs does not allow using this approach. These advantages allow the proposed ANN partitioning principle to be used in fog computing when the number and computational power of FC nodes are not known in advance. In this case, once trained ANN can be placed on the number of FC node that is available without the need for additional training. In addition, when FC nodes are released, the used ANN can be augmented to increase the recognition accuracy. Thus, the scalability and dynamism of computation, which is one of the most important indicators of FC, are achieved.

ACKNOWLEDGEMENTS

The research was supported by the Russian Science Foundation Grant No. 24-21-00149, https://rscf.ru/en/project/24-21-00149/.

REFERENCES

1. Ahmed, N. & Rao, K. R. (1975) *Orthogonal Transfarms for Digital Signal Processing*. Berlin, Germany: Springer-Verlag.
2. Bahtin, V. V. (2022) A monolithic neural network separation algorithm for implementing foggy computing in programmable logic devices, *Bulletin of PNRPU.*

Series: Electrical Engineering, Information technology, control systems, **41**, 123–145, [in Russian], doi: 10.15593/2224-9397/2022.1.06

3. Dean, J. & Ghemawat, S. (2008) Mapreduce: Simplified data processing on large clusters, *Communications of the ACM*, **51**, 107–113.
4. Gorban, A. N. (1998) Generalized approximation theorem and exact representation of polynomials from several variables by superpositions of polynomials from one variable, Russian Math. (Iz. VUZ), **42**(5), 4–7, [in Russian].
5. *GTSRB – German Traffic Sign Recognition Benchmark* [Online]. Available https://www.kaggle.com/datasets/meowmeowmeowmeowmeow/gtsrb-german-traffic-sign
6. Haar, A. (1909) Zur theorie der orthogonalen funktionensysteme, *Mathematische Annalen*, **69**, 331–371, [in German].
7. Iorga, M., Feldman, L., Barton, R., Martin, M. J., Goren, N. et al. (2018) Fog Computing Conceptual Model, *NIST Special Publication* **500-325**, 1–14.
8. Jiang, W., Lu, S., Jiang, Y., Chen, J., Ye, F., et al. (2020) Evolutionary dynamics modeling of symbolic social network structure equilibrium, *China Commun,* **17**(10), 229–240.
9. Keshavarznejad, M., Rezvani, M. H. & Adabi, S. (2021). Delay-aware optimization of energy consumption for task offloading in fog environments using metaheuristic algorithms, *Cluster Computing*, **24**, 1825–1853, doi: 10.1007/s10586-020-03230-y.
10. Kirsanova, A. A., Radchenko, G. I. & Chernykh, A. N. (2020) Overview of fog computing organization technologies, *Bulletin of SUSU. Series: Computational Mathematics and Computer Science*, **9**(3), 35–63, doi: 10.14529/cmse200303.
11. Mahmud, R., Kotagiri, R. & Buyya, R. (2018) Fog computing: a taxonomy, survey and future directions. In: *Internet of Everything*. Berlin, Germany: *Springer,* 103–130.
12. Mao, J., Chen, X., Nixon, K. W., Krieger, C. & Chen, Y. (2017) MoDNN: Local distributed mobile computing system for Deep Neural Network, *Proc. of Design, Automation & Test in Europe Conference & Exhibition (DATE)* (Lausanne, Switzerland), 1396–1401, doi: 10.23919/DATE.2017.7927211.
13. McCulloch, W. S. & Pitts, W. (1943) A logical calculus of the ideas immanent in nervous activity, *The bulletin of mathematical biophysics*, **5**(4), 115–133.
14. Qiao, Yu. (2007) The MNIST database of handwritten digits, [Online]. Available: https://archive.ics.uci.edu/dataset/683/mnist+database+of+handwritten+digits
15. Rytov, S. M., Kravtsov, Yu. A. & Tatarsky, V. I. (1978) *Introduction to statistical radiophysics. Part 2. Random fields*. Moscow: Nauka, Main Editorial Office of Physical and Mathematical Literature, [In Russian].
16. Smolencev, N. K. (2019) *Basics of wavelet theory. Wavelets in MATLAB*. Moscow, Russia: DMK Press, [In Russian].
17. Tikhonov, A. N. & Sveshnikov, A. G. (1967) *Theory of functions of a complex variable*. Moscow, USSR: Nauka, [in Russian].
18. Ushakov, Y. A., Polezhaev, P. N., Shukhman, A. E. & Ushakova, M. V. (2018) Distribution of the neural network between mobile device and cloud infrastructure services, *Modern Information Technology and IT-education*, **14**(4), 903–910, [in Russian], doi: 10.25559/SITITO.14.201804.903-910.
19. Vershkon, N. A., Babenko, M. G., Kuchukova, N. N., Kuchukov, V. A. & Kucherov, N. N. (2024) Cross-layer separation of artificial neural networks for image classification, *Computer Optics Journal*, **48**(2), 312–320, [in Russian], doi: 10.18287/2412-6179-CO-1278.
20. Vershkov, N., Babenko, M., Tchernykh, A., et al. (2022) Optimization of Artificial Neural Networks using Wavelet Transforms, *Program Comput Soft*, **48**, 376–384, https://doi.org/10.1134/S036176882206007X

