We developed a method of training artificial neural networks for intelligent decision support systems. A distinctive feature of the proposed method consists in training not only the synaptic weights of an artificial neural network, but also the type and parameters of the membership function. In case of impossibility to ensure a given quality of functioning of artificial neural networks by training the parameters of an artificial neural network, the architecture of artificial neural networks is trained. The choice of architecture, type and parameters of the membership function is based on the computing resources of the device and taking into account the type and amount of information coming to the input of the artificial neural network. Another distinctive feature of the developed method is that no preliminary calculation data are required to calculate the input data. The development of the proposed method is due to the need for training artificial neural networks for intelligent decision support systems, in order to process more information, while making unambiguous decisions. According to the results of the study, this training method provides on average 10–18 % higher efficiency of training artificial neural networks and does not accumulate training errors. This method will allow training artificial neural networks by training the parameters and architecture, determining effective measures to improve the efficiency of artificial neural networks. This method will allow reducing the use of computing resources of decision support systems, developing measures to improve the efficiency of training artificial neural networks, increasing the efficiency of information processing in artificial neural networks

Keywords: artificial neural networks, efficiency of information processing, decision support systems

UDC 681.324.01
DOI: 10.15587/1729-4061.2022.251637

DEVELOPMENT OF A METHOD FOR TRAINING ARTIFICIAL NEURAL NETWORKS FOR INTELLIGENT DECISION SUPPORT SYSTEMS

Qasim Abbood Mahdi
PhD, Head of Department
Department of Computer Technology
Al Taff University College
Karrada str., 3, Karbala, Iraq, 31001
Andrii Shyshatskyi
PhD, Senior Researcher
Research Department of Electronic Warfare Development*
Oleksandr Symonenko
PhD, Senior Teacher**
Nadiia Protas
PhD, Associate Professor
Department of Information Systems and Technologies
Poltava State Agrarian University
Skovorody str., 1/3, Poltava, Ukraine, 36000
Oleksandr Trotsko
Corresponding author
PhD, Associate Professor**
E-mail: s.trockiy@gmail.com
Volodymyr Kyvliuk
PhD, Associate Professor
Department of Logistic Support
The National Defence University of Ukraine named after Ivan Cherniakhovsky
Povitrofloski ave., 28, Kyiv, Ukraine, 03049
Artem Shulhin
PhD, Head of Research Department***
Petro Steshenko
PhD, Leading Researcher***
Eduard Ostapchuk
Deputy Head of the Research Department
Research Department for the Development of Protection
and Survivability of Weapons and Military Equipment*
Tetiana Holenkovska
Researcher
Research Department of the Development of Communications
and Technical Information Protection*
*Central Scientific Research Institute of Armament
and Military Equipment of the Armed Forces of Ukraine
Povitrofloski ave., 28, Kyiv, Ukraine, 03049
**Department of Automated Control Systems
Military Institute of Telecommunications
and Information Technologies named after Heroes of Kruty
Moskovska str., 45/1, Kyiv, Ukraine, 010011
***Research Department
State Scientific-Research Institute of Aviation
Andryushchenka str., 6V, Kyiv, Ukraine, 01135

Received date 31.12.2021
Accepted date 11.02.2022
Published date 28.02.2022

How to Cite: *Mahdi, Q. A., Shyshatskyi, A., Symonenko, O., Protas, N., Trotsko, O., Kyvliuk, V., Shulhin, A., Steshenko, P., Ostapchuk, E., Holenkovska, T. (2022). Development of a method for training artificial neural networks for intelligent decision support systems. Eastern-European Journal of Enterprise Technologies, 1 (9 (115)), 35–44. doi: https://doi.org/10.15587/1729-4061.2022.251637*

1. Introduction

Decision support systems (DSS) have become the basis for solving information and calculation problems in everyday life and to solve very specific (special) tasks. DSS are actively used in processing large data sets, providing information support to the decision-making process of decision-makers. The basis of existing DSS is artificial intelligence methods [1–11].

The creation of intelligent DSS was a further development of classical DSS, the main tool of which is evolving artificial neural networks (ANN). Intelligent Decision-Maker Support System (iDMSS) is an interactive computer system designed to support decision-making in various fields of activity regarding poorly structured and unstructured problems, based on the use of models and procedures for data processing and knowledge based on artificial intelligence technologies.

Evolving ANN have versatile approximating properties. Evolving ANN provide stable operation in conditions of nonlinearity, a priori certainty, stochasticity and chaos, various kinds of disturbance and interference.

Despite their successful application to a wide range of data mining problems, these systems have a number of disadvantages. The most significant shortcomings are the following:

– complexity of system architecture selection. As a rule, a model based on the principles of computational intelligence has a fixed architecture. In the context of ANN, this means that the neural network has a fixed number of neurons and connections. Therefore, adapting the system to new data received for processing, which differ from previous data, may be problematic;

– formation of «dead» neurons in the layers during ANN functioning;

– batch training and multi-epoch training requires considerable time resources. Such systems are not adapted to work online with a fairly high flow rate of new data for processing;

– many of the existing computational intelligence systems cannot determine the evolving rules by which the system develops, and can also present the results of their work in terms of natural language.

Thus, the task of developing new methods (approaches, techniques) for training ANN, which will solve these difficulties, is relevant.

2. Literature review and problem statement

In [3], an analysis of the properties of ANN, which were used in predicting the air pollutant concentration, was carried out. The use of an extreme training machine for ANN is proposed, which provides high generalization efficiency at extremely high training rates. The disadvantages of the approach include the accumulation of ANN errors during calculations, the impossibility to choose the parameters and type of the membership function, the formation of «dead» neurons during training.

The work [4] presents the modeling of bank capital management adequacy. This modeling is based on trend forecasting models. A multilayer perceptron is used for calculations. The training of this perceptron is limited only to training synaptic weights, and only activated neurons. No other training mechanisms are presented in this study.

The work [5] presents an operational approach to spatial analysis in the maritime sector to quantify and reflect related ecosystem services. The disadvantages of this method include the impossibility of flexible adjustment (adaptation) of evaluation models while adding (excluding) indicators and changing their parameters (consistency and significance). Also, ANN training is limited to classical training of the weights of active neurons.

The work [6] presents a machine learning model for automatic identification of requests and providing information support services exchanged between members of the Internet community. This model is designed to process a large number of messages from users of social networks. The disadvantages of this model are the lack of mechanisms for assessing the adequacy of decisions and high computational complexity. Training is limited to the synaptic weights of the ANN.

The work [7] demonstrates the use of ANN to detect cardiac arrhythmias and other heart diseases. The backpropagation algorithm is used as a method of ANN training. The disadvantage of this approach is the limited training of only synaptic weights, without training the type and parameters of the membership function.

In [8], it is proposed to use ANN to detect avalanches. The backpropagation algorithm is used as a method of ANN training. The disadvantage of this approach is the limited training of only synaptic weights, without training the type and parameters of the membership function.

The work [9] presents the use of ANN to identify anomaly detection problems in home authorization systems. The «winner-take-all» algorithm is used as a method of Kohonen ANN training. The disadvantages of this approach are the accumulation of errors due to the presence of inactivated and dead neurons in the training process, the limited training of only synaptic weights and the need to store previously calculated data.

The work [10] presents the use of ANN to identify problems in detecting abnormalities in a human encephalogram. The method of fine-tuning of ANN parameters is used as a method of ANN training. The disadvantages of this approach are the accumulation of errors in the training process, the limited training of only synaptic weights without training the type and parameters of the membership function.

The work [12] presents the use of machine learning methods, namely ANN and genetic algorithms. A genetic algorithm is used as a method of ANN training. The disadvantage of this approach is the limited training of only synaptic weights, without training the type and parameters of the membership function.

The work [13] presents the use of machine learning methods, namely ANN and differential search method. In the course of the study, a hybrid method of ANN training was developed, based on the use of backpropagation and differential search algorithms. The disadvantages of this approach are the limited training of only synaptic weights, without training the type and parameters of the membership function.

In [14], the methods of ANN training were developed using a combined approximation of the response surface, which provides the smallest training and forecasting errors. The disadvantage of this method is the accumulation of training errors and the inability to change the ANN architecture during training.

The work [15] shows the use of ANN to evaluate the efficiency of a unit using the previous time series of its performance. SBM (Stochastic Block Model) and DEA (Data Envelopment Analysis) models are used for ANN training. The disadvantages of this approach are the limited choice of network architecture and training of only synaptic weights.

The work [16] describes the use of ANN to evaluate geomechanical properties. The backpropagation algorithm is used as a method of ANN training. The characteristics of the backpropagation algorithm are improved by increasing the training sample. The disadvantages of this approach are the limited training of only synaptic weights, without training the type and parameters of the membership function.

The work [17] presents the use of ANN to assess traffic intensity. The backpropagation algorithm is used as a method of ANN training. The performance of the backpropagation

algorithm is improved using bandwidth connections among each layer, so that each layer presents only a residual function relative to the results of the previous layer. The disadvantage of this approach is the limited training of only synaptic weights, without training the type and parameters of the membership function.

The analysis of scientific works [1–17] showed that well-known training methods are used for artificial neural networks. These methods are usually focused on training synaptic weights or membership functions. The use of known algorithms (methods, techniques) for training artificial neural networks, even with improved characteristics, does not meet the existing and future requirements for them, namely:

– increasing the amount of information that artificial neural networks can process;

– increasing the reliability of decision-making of intelligent decision support systems;

– increasing the speed of adaptation of the architecture and parameters of artificial neural networks in accordance with emerging tasks;

– increasing the efficiency of systems with limited computing capabilities, such as on-board aircraft observers, by improving evaluation accuracy;

– prevention of deadlocks in the training of artificial neural networks;

– ensuring the predictability of the process of training artificial neural networks;

– ensuring the unambiguity of decisions made by intelligent decision support systems;

– ensuring the calculation of large data sets for a single epoch without saving previous calculations.

3. The aim and objectives of the study

The aim of the study is to develop a method for training artificial neural networks for intelligent decision support systems, which allows processing more information while making unambiguous decisions.

To achieve the aim, the following objectives were set:

– to develop an algorithm for training artificial neural networks for intelligent decision support systems;

– to evaluate the effectiveness of training artificial neural networks experimentally.

4. Materials and methods

During the study, the general provisions of artificial intelligence theory were used to solve the problem of training artificial neural networks in intelligent decision support systems. Thus, artificial intelligence theory is the basis of this study. The study used an advanced genetic algorithm and evolving artificial neural networks. Simulations were performed using the MathCad 2014 software (USA) and Intel Core i3 PC (USA).

The Kohonen network [2, 18–24] refers to self-organizing networks. This means that they do not receive the desired output signal upon receipt of the input training vector, and as a result of training, the network divides the input signals into classes, thus forming topological maps.

It should be noted that the Kohonen self-organizing map implements the mapping of the input space of dimension n into the output space of dimension m.

The self-organizing map has a very simple architecture with direct information transfer. In addition to the zero (receptor) layer, it contains a single layer of neurons, which is often called the Kohonen layer [25–32].

Let us consider the self-organizing map architecture in more details. The network input receives an n-dimensional input signal. The network contains a single layer of m neurons that form rectangular lattices on the plane.

Neurons are characterized by their location in the network. Each neuron in the Kohonen layer is connected to each input of the zero (input) layer by direct connections and to all other neurons by cross connections.

Fig. 1 presents a 1D Kohonen map.

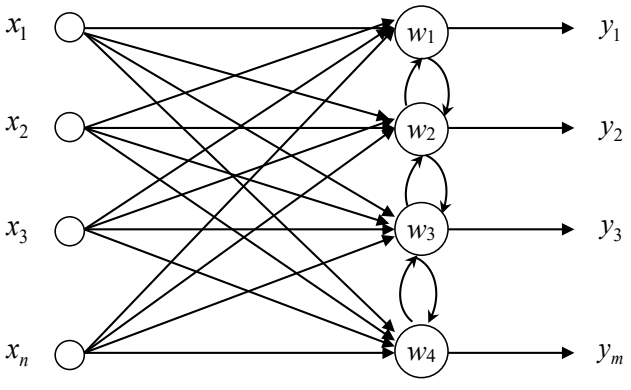

Fig. 1. 1D Kohonen map

In the training process, neighboring neurons affect each other more strongly than those located further away. It is the lateral connections in the network that excite some neurons and inhibit others.

Each neuron of the Kohonen layer forms a weighted sum of signals:

$$f(x,w)=\sum_{i=1} w_i x_i.$$

If the synapses are accelerating, then $w_{ij}>0$. If the synapses are inhibitory, then $w_{ij}<0$.

Given the above, the classic training procedure of the Kohonen network is to adjust the synaptic weights, without taking into account other network training opportunities, such as the type and parameters of the membership function and network architecture.

5. Results of research on the development of a method for training artificial neural networks

5. 1. Development of an algorithm for training artificial neural networks

Fig. 2 shows the proposed algorithm for training an artificial neural network. The improvement of this training algorithm lies in improving procedures 2, 3, 8 of the previously developed method of training artificial neural networks [2, 18, 32].

Briefly, the main steps of the proposed method are:

Step 1. Initialization of the initial values of synaptic weights.

Step 2. Determination of neuronal weights.

Step 3. Correction of neuronal weights and determination of neighborhood function.

Step 4. Formation of the first cluster.

Step 5. Checking the threshold value.

Step 6. Checking the architecture's capabilities to process the amount of information coming to its input.

Step 7. Evolution of the system architecture.

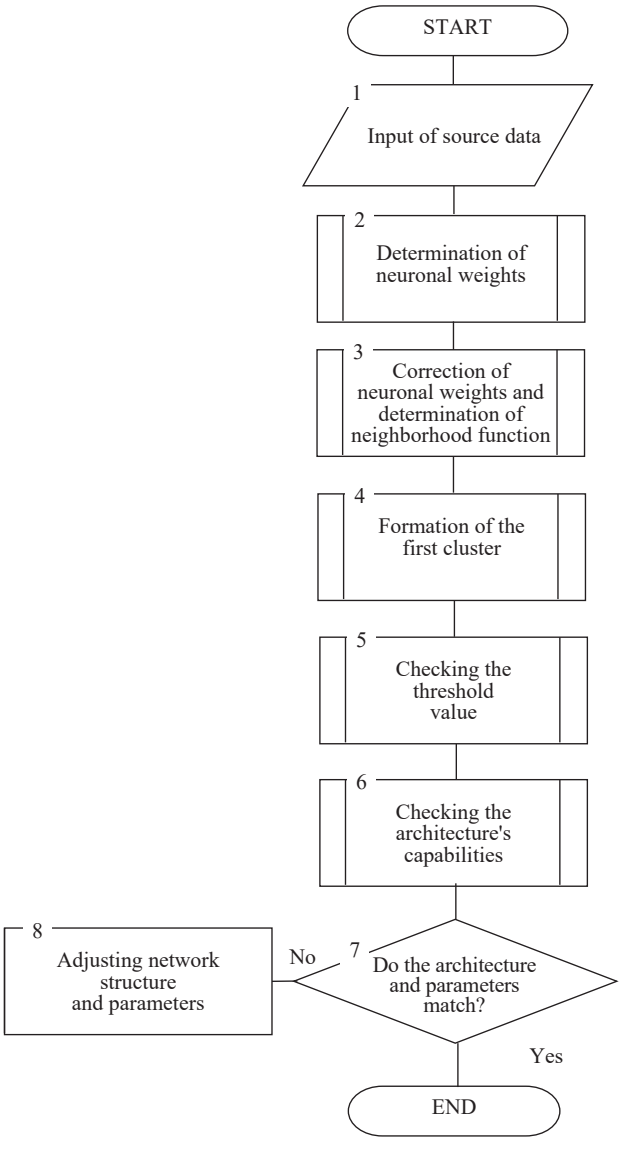

Fig. 2. Algorithm of functioning and training of an evolving artificial neural network

We describe *steps 2, 3* and *8* in detail. The essence of the improvement is genetic-competitive training, supplemented by the introduction of various strategies for genetic optimization of weights of «dead» neurons located in the source layer of the network. Also, the type of uncertainty of the training sample is additionally taken into account (the approach is detailed in [32]). The proposed stochastic optimization reduces the number of training echoes of the Kohonen network while reaching a given maximum value of vector quantization error and while constructing centroids, uncertainty coefficients (total uncertainty, partial uncertainty, full awareness) are additionally taken into account when selecting initial values of cluster centers.

Before starting the Kohonen network training algorithm, the input vectors are pre-normalized [33, 34]:

$$\tilde{x}_i = \frac{x_i}{\sqrt{\sum_i x_i^2}} = \frac{x_i}{\|x\|},\ i = 1,2...,N. \quad (1)$$

The Kohonen network training algorithm can be described as a sequence of steps:

Step 1. Input of source data. At this stage, the initial values of synaptic weights $w_{ij}=0$ are initialized.

One commonly used initialization method is to assign synaptic weights values equal to randomly selected vectors from multiple observations.

Step 2. Determination of neuronal weights. At this stage, the normalized signal vector $\tilde{x}$ is fed to the input of the system and the weight vector (neuron) closest to $\tilde{x}$ is selected, i.e., the vector for which the Euclidean distance to x is the smallest:

$$\arg\min_j \left\|\tilde{x} - w_j\right\|,\ j = 1,2,...,l. \quad (2)$$

The following steps are performed:

2. 1. Setting the parameters of the Kohonen network (the size of the original network $I \times J$, the number of training epochs $T \geq 1$, the initial width of the neuron neighborhood σ_0, the coefficients τ, κ_0, η).

2. 2. Zeroing of the counter of current iterations $t:=0$, initialization of the weight factors w_{ij} $(1 \leq i \leq I, 1 \leq j \leq J)$ of the output lattice neurons randomly, preparation of training data $\{x_k\}_{k=1}^M$, choosing an optimization strategy G' for the weights of the output lattice neurons.

Also, at this stage, the current width of the centroid edges is calculated:

$$\sigma(t) = \sigma_0 \cdot \exp\left\{-\tau \cdot \frac{t}{T-1}\right\} \text{ for } T>1$$

and

$$\sigma(t) = \sigma_0 \text{ for } T=1. \quad (3)$$

Step 3. Correction of neuronal weights and determination of neighborhood function.

We denote as w_{ij} the weight vector of the neuron, which has coordinates (i, j) on the input lattice of the Kohonen network (i is the row number, j is the column number). The training process is aimed at minimizing the half sum of squares of the distances between the input vectors $\{x_k\}_{k=1}^M$ of the training sample and the neuron vectors of the output lattice $(1 \leq i \leq I, 1 \leq j \leq J)$.

$$E\left(w_{11},...,w_{IJ}\right) = \frac{1}{2}\sum_{k=1}^{M}\left(D\left(w_{i_k^* j_k^*}, x_k,\right)\right)^2 \to \min_{w_{11},...,w_{IJ}}, \quad (4)$$

where

$$D(w,x) = \sqrt{(w-x)^T \cdot (w-x)}$$

is the function of the distances between a pair of vectors in Euclidean space;

$$\left(i_k^*, j_k^*\right) = \overbrace{\arg\min_{\substack{1\leq i\leq I \\ 1\leq j\leq J}} D\left(w_{ij}, x_k\right)}^{F_{IJ}:\mathbb{R}^n \to \{1,...,I\}\times\{1,...,J\}}$$

are the coordinates of the neuron on the source network layer, whose weights are closest to the vector x_k.

The value $1/M \cdot E\left(w_{11},...,w_{IJ}\right)$ is the error of vector quantization [24]. Using the gradient descent method, we obtain the following formula for updating the weight vectors: w_{ij} $(1 \le i \le I, 1 \le j \le J)$:

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij},$$

$$\Delta w_{ij} = -\kappa \frac{\partial E\left(w_{11},...,w_{IJ}\right)}{\partial w_{ij}} =$$

$$= \kappa \sum_{k=1}^{M} \left(\left[(i,j) = F_{IJ}(x_k) \right] \left(x_k - w_{ij} \right) \right), \quad (5)$$

where κ is some positive constant or function with the range of values [0, 1], setting the training rate. Note that in (5), only part of the vectors from the training sample with the smallest discrepancy between each of them w_{ij} is used to update each particular weight vector w_{ij}. In other words, the vector is modified if and only if it is closest to the training vector x_k within a given metric space. The correction of the vector w_{ij} is carried out by a value directly proportional to the difference between the input vector x_k and the weight vector w_{ij}. Thus, there is competition between the output lattice neurons for the right to be selected as the candidates closest to the input vector x_k. The neuron satisfying this requirement is called the winning neuron with coordinates $\left(i_k^*, j_k^*\right)$. Note that when rationing the vectors w_{ij} and x_k, the minimization of $E(w_{11},...,w_{IJ})$ is equivalent to maximizing the sum of their scalar products:

$$E\left(w_{11},...,w_{IJ}\right) = \frac{1}{2} \cdot \sum_{k=1}^{M} \left(w_{i_k^* j_k^*} - x_k \right)^T \cdot \left(w_{i_k^* j_k^*} - x_k \right) =$$

$$= \max_{w_{11},...,w_{IJ}} \sum_{k=1}^{M} w_{i_k^* j_k^*}^T \cdot x_k - M. \quad (6)$$

To reduce competition between neurons, a rule is introduced that allows you to update not only the weights of the winning neuron, but also other neurons in its vicinity. To this end, the previously introduced characteristic function $[(i,j=F_{IJ})(x_k)]$ is replaced by an exponential Gaussian function, the value of which reflects the attenuating dependence of changes in neuronal weights with increasing distance from neurons to the winning neuron at the level of their coordinates on the original lattice. The closer the neuron is to the winning neuron, the greater the multiplicative factor of updating its weights. The parameter σ is called the effective neighborhood width [35–37] of the winning neuron, which can be interpreted as the current value of the neighborhood radius of the winning neuron. A feature of the Kohonen network training algorithm is a decrease in the value of σ over time:

$$\sigma(t) = \sigma_0 \cdot \exp\left\{ -\tau \cdot \frac{t}{T-1} \right\} \ (t = 0,...,T-1).$$

Here, the parameter σ_0 sets the initial value of the neighborhood radius of the winning neuron, which is usually set to $\sqrt{I^2 + J^2}$. The parameter τ is selected so that in the last training epoch, the minimum number of weight vectors of neurons or only one vector of the winning neuron is updated. Thus, it was $\tau = \ln(\sigma_0)$. The training rate factor is chosen so that in the initial epochs of the algorithm, the weight vectors of most neurons are updated at the highest rate. Then, as the number of epochs increased and the width of the neighborhood narrowed, fewer and fewer neuronal vectors were modified at a lower rate. This technique allows you to build clusters whose elements are first adapted to the general characteristics of the approximating set, and then specify its individual features. The most common representatives with such a characteristically descending dependence are the functions $\kappa(t) = \kappa_0 (t+1)^{-1}$, $\kappa(t) = \kappa_0 \cdot \exp\{-\eta t\}$ [35–37].

The essence of improving procedure 3 is to use the Kohonen network competitive training algorithm, supplemented by genetic operators:

3. 1. Initialization of the current set of active neurons $V^+ := \varnothing$.

3. 2. Performing *steps 3. 2. 1–3. 2. 8* of each vector x_k $(k = 1, ..., M)$.

3. 2. 1. Normalization of the weight factors of neurons w_{ij} by component-wise division by $\|w_{ij}\| \left(w_{ij} \ne \vec{0} \right)$.

3. 2. 2. Normalization of the vector x_k by component-wise division by $\|x_k\| \left(x_k \ne \vec{0} \right)$.

3. 2. 3. Calculation of the distances between the vector x_k and each weight vector w_{ij} of the neuron:

$$d_{ijk} = D\left(x_k, w_{ij}\right) = \sqrt{\sum_{l=1}^{n} \left(x_{kl} - w_{ijl} \right)^2}.$$

3. 2. 4. Determining the coordinates of the winning neuron for the vector x_k:

$$\left(i_k^*, j_k^*\right) = F_{IJ}(x_k) = \arg \min_{\substack{1 \le i \le I \\ 1 \le j \le J}} d_{ijk}. \quad (7)$$

3. 2. 5. Determining the current vicinity of the winning neuron $\left(i_k^*, j_k^*\right)$:

$$V^* = V\left(i_k^*, j_k^*\right) = \left\{ (i,j) \middle| \begin{pmatrix} 1 \le i \le I \\ 1 \le j \le J \end{pmatrix} \wedge \right.$$

$$\left. \wedge \left(i - i_k^* \right)^2 + \left(j - j_k^* \right)^2 < \sigma^2(t) \right\}. \quad (8)$$

3. 2. 6. Modification of the weights of neurons with coordinates:

$$(i,j) \in V^* : w_{ij} := w_{ij} + \kappa(t) \cdot \varphi(i,j,t) \cdot \left(x_k - w_{ij} \right),$$

where the function is

$$\varphi(i,j,t) = \exp\left\{ \frac{\left(i - i_k^* \right) + \left(j - j_k^* \right)^2}{2\sigma^2(t)} \right\}.$$

3. 2. 7. Expansion of the set of active neurons:

$$V^+ := V^+ \bigcup \left\{ i_k^*, j_k^* \right\}.$$

3. 2. 8. Applying the genetic optimization strategy G' to the weights w_{ij} for:

$$(i,j) \in V^\dagger, V^* \setminus \left\{ \left(i_k^*, j_k^* \right) \right\}.$$

3. 3. Applying the genetic optimization strategy G' to the weights w_{ij} for:

$$(i,j) \in V^\dagger, V^\dagger = V_{\left(i_M^*, j_M^* \right)} \setminus V^+.$$

3. 4. Increasing the counter of current iterations: t: t+1.

3. 5. Go to the next step 3. 4 if the condition $t \geq T$ is met, otherwise go to step 2.

3. 6. Exclusion of «dead» neurons with coordinates $(i', j') \in V^{\dagger}$ on the original lattice, where

$$V^{\dagger} = \left\{(i,j) \middle| \forall k \in \{1,...,M\}(i,j) \neq F_{IJ}(x_k)\right\}.$$

3. 7. Calculation of the activation threshold of the remaining neurons with coordinates:

$$(i'', j'') \notin V^{\dagger} : h^{-}_{i'',j''} = \min_{1 \leq k \leq M} \left\{ d^{-1}_{i''j''k} \middle| (i'', j'') = F_{IJ}(x_k) \right\}.$$

Step 6. Checking the architecture's capabilities to process the amount of information coming to its input with optimization of the artificial neural network architecture.

After modification of weights when presenting the training vector (*step 3. 2. 8*) or after performing several epochs of competitive training (*step 3. 3*), stochastic optimization is additionally applied. This optimization is based on a genetic algorithm of stochastic optimization of the weights of certain neurons of the original lattices of the Kohonen map. To this aim, the weights of each neuron at the edges of the current winning neuron and never activated are given as a sequence of genes acting as the minimum unit for the input argument of the crossover operator (O_1). As a result of this operator, a pair of new chromosomes is formed, in which randomly selected regions of the chromosome gene are rearranged. Each gene is a set of bits that can be considered as a separate component of the vector associated with the corresponding winning neuron or one of the neurons in its vicinity. When using the mutation operator (O_2), there is a permutation of a pair of randomly selected bits within one gene, when using the inversion operator (O_3), there is an inversion of the value of the randomly selected bit. Both of these operators apply only to part of the mantissa of the 64-bit «objective gene».

To simulate the process of neuronal evolution, several methods have been developed to generate offspring. The first approach A_1 is to apply the operators of crossover, mutation or inversion of arbitrary neurons that are currently close to the winning neuron. The second approach A_2 is based on geometric considerations about the mutual position of neurons on the original lattice relative to the winning neuron. Since the weight vector of each neuron equidistant from the winning neuron is modified with the same coefficient as a result of the selected training algorithm, the crossover operator is proposed to be applied to such neurons. The third approach A_3 involves applying the crossover operator only to the most adapted individuals, while the mutation or inversion operators will be used for the remaining neurons. Here, the inverse value of the mean deviation was chosen as a fitness function when the given neuron with the weight w_{ij} recognized elements of the training $\{x_k\}_{k=1}^{M}$ sample:

$$\{x_k\}_{k=1}^{M} : \psi_{ij} = \left(\frac{\sum_{k=1}^{M} \|x_k - w_{ij}\|}{M} \right)^{-1}$$

for step 3.5 or

$$\psi_{ijk} = \|x_k - w_{ij}\|^{-1}$$

for *step 3. 2. 8*. In addition, several strategies were developed to select a neuron generation method: fixed choice G_1 of a certain approach, sequential or random selection of all approaches G_2 and the choice based on the roulette mechanism from [32]. In the G_3 strategy, the offspring of neurons generated with the chosen approach are more adapted than their predecessors. Then the probability of choosing such an approach in the future increases compared to other approaches, otherwise the probability of choosing it decreases.

Each of the approaches A_1, A_2, A_3 is used to create two daughter Kohonen maps, in which only neurons from the environment of the winning neuron of the parent Kohonen map can be changed. These approaches can be described as a sequence of actions in which only *step 3* differs:

Step 6. 1. V:=∅.

Step 6. 2. Arbitrary choice of the operator O_{i^*} ($i^* \in \{1,2,3\}$); (3_{A_1}).

Step 6. 3. Application of the operator O_{i^*} to the next neuron with coordinates $(i', j') \in V^{\dagger} \setminus V$ from the current vicinity V^* of the winning neuron (if O_{i^*} is the crossover operator, you need to additionally select another neuron with coordinates (i'', j'') (3_{A_2}) other than (i', j') of the crossover operator to a pair of neurons with coordinates $(i', j') \in V^{\dagger} \setminus V$ and

$$(i'', j'') \in U = \\ = \left\{(i,j)\right\}_{(i,j) \in V^* \setminus V \setminus \{(i',j')\} \wedge \left(i - i^*_{k(M)}\right)^2 + \left(j + j^*_{k(M)}\right)^2 = \left(i' - i^*_{k(M)}\right)^2 + \left(j' - j^*_{k(M)}\right)^2}$$

from the vicinity V^* of the winning neuron with coordinates $(i^*_{k(M)}, j^*_{k(M)})$ (if $U=\varnothing$, it is necessary to apply twice the mutation or inversion operator to the neuron with coordinates (i', j') (3_{A_3}). Using the crossover operator to neurons with coordinates:

$$(i', j') = \arg \max_{(i,j) \in V^{\dagger} \setminus V} \psi_{ij}(k)$$

and

$$(i'', j'') = \arg \max_{(i,j) \in V^* \setminus V \setminus \{(i',j')\}} \psi_{ij}(k)$$

(if $V^{\dagger} \setminus V \setminus \left\{(i', j')\right\} = \varnothing \vee \#\left(V^{\dagger} \setminus V\right) \leq K \cdot \#V^*$, it is necessary to use twice the mutation or inversion operator to the neuron with coordinates (i', j'), where $0 \leq K \leq 1$ is a constant detere mining the relative number of neurons to which only the mutation or inversion operator should be applied).

Step 6. 4. Adding two new neurons generated by the O_{i^*} operator, one in each of the two Kohonen daughter maps, keeping each of these neurons in position (i', j') on the original lattice.

Step 6. 5. Adding coordinates (i', j') to the set:

$$V : V := V \bigcup \left\{(i', j')\right\}.$$

Step 6. 6. Go to step 6. 2, if $V \neq V^{\dagger}$, otherwise stop.

Each of the strategies G_1, G_2, G_3 manipulates the choice of approaches A_1, A_2, A_3. The G_1 strategy is the most primitive of the analyzed and consists in choosing one of the three approaches A_1, A_2, A_3 in *step 6. 1* of the algorithm of genetic-competitive training of the Kohonen network.

Therefore, if there are N observations and m clusters with centroids c_j, calculations of all memberships and the adjusted coordinates of the centroids are estimated according to the relation:

$$\begin{cases} u_j(k)=\left(1+\left(\dfrac{\|x(k)-c\|^2}{\mu_j}\right)^{\frac{1}{\beta-1}}\right)^{-1}, \\ c_j=\dfrac{\sum\limits_{k=1}^{N}u_j^{\beta}(k)x(k)}{\sum\limits_{k=1}^{N}u_j^{\beta}(k)}, \\ \mu_j(k)=\dfrac{\sum\limits_{k=1}^{N}u_j^{\beta}(k)\|x(k)-c_j\|^2}{\sum\limits_{k=1}^{N}u_j^{\beta}(k)}, \\ \psi_{ij}(k)=\left(\dfrac{\sum_{k=1}^{M}\|x_k-w_{ij}\|}{M}\right)^{-1}. \end{cases} \quad (9)$$

The system of equations (9) is essentially a batch information processing algorithm so that when the observation $x(N+1)$ is received, all calculations must be performed again. With a sufficiently high data flow rate, the approach may be ineffective.

To this end, recurrent procedures should be developed that do not require storing previously used data. These recurrent procedures can be implemented on the basis of a two-layer adaptive neural fuzzy network with such an architecture.

The first hidden layer of the network is formed by ordinary Kohonen neurons N_j^K, connected by lateral connections, through which the competition process is realized. The source network layer, formed by nodes N_j^u, is designed to calculate the levels of membership of each observation $x(k)$ to each j-th cluster, $j=1, 2, 3, ..., m$. To configure the centroids of clusters, a recurrent self-learning procedure is used, which has the form [10]:

$$\begin{cases} c_j(k+1)=c_j(k)+\dfrac{u_j^{\beta}(k)}{k+1}\left(x(k+1)-c_j(k)\right), \\ u_j(k+1)=\dfrac{1}{1+\left(\dfrac{\|x(k+1)-c_j(k+1)\|^2}{\mu_j(k)}\right)^{\frac{1}{1-\beta}}}, \\ \mu_j(k+1)=\dfrac{\sum\limits_{p=1}^{k+1}u_j^{\beta}(p)\|x(p)-c_j(k+1)\|^2}{\sum\limits_{p=1}^{k+1}u_j^{\beta}(p)}, \\ \psi_{ij}(k+1)=\left(\dfrac{\sum\limits_{M=1}^{k+1}\mu_j(k+1)\|x_k-w_{ij}\|}{M}\right)^{-1}. \end{cases} \quad (10)$$

It is easy to see that the first expression (10) is a WTM self-learning rule with a narrowing neighborhood function $(k+1)^{-1}u_j^{\beta}(k)$.

Steps 4, 5 7 are described in detail in previous studies [18, 32], and *steps 2, 3, 6* are described respectively by expressions (1) to (10).

5. 2. Experimental evaluation of the effectiveness of training artificial neural networks

Simulation of the proposed method in the MathCad 14 software environment is performed. For the experiment, we used a two-dimensional artificially generated data set with different degrees of uncertainty (for the convenience of calculation, 1,500 of each type of uncertainty). Thus, the generated data set consists of data:

– that are fully reliable (full awareness of the object state), so intelligence data are complete and reliable;

– partial uncertainty (missing information on individual assessment indicator);

– complete uncertainty (no information about the monitoring object).

The data set contained 15 clusters with different levels of overlap. The data sample contained 4,500 observations. The data were submitted for processing in sequential mode.

To compare the quality of clustering, FCM and a system based on the evolving fuzzy clustering method (EFCM) with different threshold parameter values, EFCM system with the training method from [18, 32], K-means++ and K-medoids (Partitioning Around Medoid) were used.

For the next experiment, a data set describing the characteristics of the monitored object was used. Each observation was described by seven parameters:

– the number of personnel;

– total number of personnel;

– number of organizational and staff structures;

– number of weapons and military equipment samples;

– number of radio communication devices;

– number of types of weapons and military equipment samples and the type of radio communication devices.

Before clustering, the observational features were normalized in the interval [0, 1].

The Xie-Beni index was used as a criterion for assessing clustering quality.

Table 1 presents the comparative results of clustering.

Table 1

Comparative results of clustering

System	Number of clusters	Algorithm parameters	XB (Xie-Beni Index)	Clustering time, s
FCM (Fuzzy C-Means)	15	delta=0.1	0.1903	1.4
K-means++	15	delta=0.1	0.1361	0.69
K-medoids	15	delta=0.1	0.1256	0,651
EFCM	15	Dthr=0.24	0.148	0.54
EFCM	15	Dthr=0.19	0.139	0.49
System (batch mode) proposed in [18, 32]	15	delta=0.1	0.12	0.37
System (online mode) proposed in [18, 32]	15	delta=0.1	0.1127	0.25
Proposed system (batch mode)	15	delta=0.1	0.1006	0.37
Proposed system (online mode)	15	delta=0.1	0.10045	0.25

The study of the developed methodology showed that this training method provides on average 11–15 % higher efficiency of training artificial neural networks and does not accumulate training errors (Table 1).

These results can be seen in the last rows of Table 1 as the difference in the Xie-Beni index.

6. Discussion of the results of the development of artificial neural network training method

The advantages of this method are achieved by performing a sequence of additional procedures, namely 2, 3, 6, shown in Fig. 2.

Analytical dependencies for these procedures are given in the formulas, namely:

– dependencies (2), (3) – *step 2* «Determination of neuronal weights»;

– dependencies (4)–(8) – *step 3* «Correction of neuronal weights and determination of neighborhood function»;

– dependencies (9), (10) – *step 6* «Checking the architecture's capabilities».

Thus, there is an adjustment not only of synaptic weights, but also the architecture of artificial neural networks, which provides fewer training epochs.

The main advantages of the proposed evaluation method are as follows:

– enables processing more information by reducing the advanced training algorithm;

– increased reliability of the results, no accumulation of errors during training artificial neural networks. This is achieved through an improved procedure for correcting synaptic weights and the architecture of the artificial neural network;

– wide scope of use (decision support systems);

– easy mathematical calculations;

– prevention of the «overtraining» phenomenon by adjusting the network architecture;

– no need to save the results of previous calculations.

The disadvantages of the proposed method include:

– the need to use high-speed genetic algorithms to adjust synaptic weights;

– lower accuracy of estimation on a separate estimation parameter;

– loss of results accuracy during the readjustment of the artificial neural network architecture.

This method will allow you:

– to train artificial neural networks;

– to make adjustments necessary to control the aircraft during refueling in the air and ensure the successful connection of the rod with the refueling cone;

– to identify effective measures to increase the efficiency of training artificial neural networks through an improved procedure of synaptic weights and network architecture;

– to reduce the use of computing resources of decision support systems;

– to develop measures aimed at improving the efficiency of training artificial neural networks;

– to increase the efficiency of information processing in artificial neural networks.

The limitations of this study include:

– the need for reliable and complete training and test samples;

– the need to take into account the time for collecting, processing and summarizing information;

– to ensure a given degree of training efficiency in the software implementation of the method, it is necessary to take into account the computational capabilities of the hardware.

The method is proposed for use in intelligent decision support systems, which adjust the angular velocities of aircraft based on the specified and current angular velocities, stabilization of the set roll angle by forming the set value of angular velocity.

It is also proposed to use the method:

– in the contour of the specified angular velocity (for example, the contour of angular velocity in roll) when synthesizing an algorithm for stabilizing a given angle;

– when compensating for inconsistencies at the final stage of aircraft docking by forming a given angular pitch velocity of the aircraft state observer.

Areas of further research should be aimed at reducing computational costs when processing various types of data in special-purpose systems.

7. Conclusions

1. An algorithm for the method of training artificial neural networks for intelligent decision support systems has been developed.

The improvement of information processing efficiency and reduction of evaluation errors are achieved by:

– training not only the synaptic weights of the artificial neural network, but also the type and parameters of the membership function;

– training the architecture of artificial neural networks;

– calculation of data for one epoch without having to store previous calculations. This reduces the processing time as there is no need to access the database;

– no accumulation of errors during training artificial neural networks as a result of processing the information received at the input of artificial neural networks.

2. The application of the proposed method on the example of clustering of the monitored object is given. This example showed an increase in the efficiency of artificial neural networks at the level of 10–18 % by the Xie-Beni index and the efficiency of information processing using additional training procedures for artificial neural networks.

Acknowledgments

The team of authors is grateful for assistance in preparing the paper to:

Doctor of Technical Sciences, Professor Oleksiy Kuvshinov – deputy head of the educational and scientific institute of the Ivan Chernyakhovsky National Defense University of Ukraine.

PhD, Associate Professor Oleksandr Bashkirov – leading researcher at the Central Research Institute of Armament and Military Equipment of the Armed Forces of Ukraine.

References

1. Kalantaievska, S., Pievtsov, H., Kuvshynov, O., Shyshatskyi, A., Yarosh, S., Gatsenko, S. et. al. (2018). Method of integral estimaK tion of channel state in the multiantenna radio communication systems. Eastern-European Journal of Enterprise Technologies, 5 (9 (95)), 60–76. doi: https://doi.org/10.15587/1729-4061.2018.144085

2. Kuchuk, N., Mohammed, A. S., Shyshatskyi, A., Nalapko, O. (2019). The method of improving the efficiency of routes selection in networks of connection with the possibility of self-organization. International Journal of Advanced Trends in Computer Science and Engineering, 8 (1.2), 1–6. Available at: http://www.warse.org/IJATCSE/static/pdf/file/ijatcse01812sl2019.pdf
3. Raskin, L., Sira, O. (2016). Method of solving fuzzy problems of mathematical programming. Eastern-European Journal of Enterprise Technologies, 5 (4 (83)), 23–28. doi: https://doi.org/10.15587/1729-4061.2016.81292
4. Katranzhy, L., Podskrebko, O., Krasko, V. (2018). Modelling the dynamics of the adequacy of bank's regulatory capital. Baltic Journal of Economic Studies, 4 (1), 188–194. doi: https://doi.org/10.30525/2256-0742/2018-4-1-188-194
5. Manea, E., Di Carlo, D., Depellegrin, D., Agardy, T., Gissi, E. (2019). Multidimensional assessment of supporting ecosystem services for marine spatial planning of the Adriatic Sea. Ecological Indicators, 101, 821–837. doi: https://doi.org/10.1016/j.ecolind.2018.12.017
6. Çavdar, A. B., Ferhatosmanoğlu, N. (2018). Airline customer lifetime value estimation using data analytics supported by social network information. Journal of Air Transport Management, 67, 19–33. doi: https://doi.org/10.1016/j.jairtraman.2017.10.007
7. Kachayeva, G. I., Mustafayev, A. G. (2018). The use of neural networks for the automatic analysis of electrocardiograms in diagnosis of cardiovascular diseases. Herald of Dagestan State Technical University. Technical Sciences, 45 (2), 114–124. doi: https://doi.org/10.21822/2073-6185-2018-45-2-114-124
8. Zhdanov, V. V. (2016). Experimental method to predict avalanches based on neural networks. Ice and Snow, 56 (4), 502–510. doi: https://doi.org/10.15356/2076-6734-2016-4-502-510
9. Kanev, A., Nasteka, A., Bessonova, C., Nevmerzhitsky, D., Silaev, A., Efremov, A., Nikiforova, K. (2017). Anomaly detection in wireless sensor network of the «smart home» system. 2017 20th Conference of Open Innovations Association (FRUCT). doi: https://doi.org/10.23919/fruct.2017.8071301
10. Sreeshakthy, M., Preethi, J. (2015). Classification of human emotion from deap EEG signal using hybrid improved neural networks with Cuckoo Search. Brain: Broad Research in Artificial Intelligence and Neuroscience, 6 (3-4), 60–73. Available at: https://brain.edusoft.ro/index.php/brain/article/view/519
11. Chica, J., Zaputt, S., Encalada, J., Salamea, C., Montalvo, M. (2019). Objective assessment of skin repigmentation using a multilayer perceptron. Journal of Medical Signals & Sensors, 9 (2), 88. doi: https://doi.org/10.4103/jmss.jmss_52_18
12. Massel, L. V., Gerget, O. M., Massel, A. G., Mamedov, T. G. (2019). The Use of Machine Learning in Situational Management in Relation to the Tasks of the Power Industry. EPJ Web of Conferences, 217, 01010. doi: https://doi.org/10.1051/epjconf/201921701010
13. Abaci, K., Yamacli, V. (2019). Hybrid Artificial Neural Network by Using Differential Search Algorithm for Solving Power Flow Problem. Advances in Electrical and Computer Engineering, 19 (4), 57–64. doi: https://doi.org/10.4316/aece.2019.04007
14. Mishchuk, O. S., Vitynskyi, P. B. (2018). Neural Network with Combined Approximation of the Surface of the Response. Research Bulletin of the National Technical University of Ukraine «Kyiv Politechnic Institute», 2, 18–24. doi: https://doi.org/10.20535/1810-0546.2018.2.129022
15. Kazemi, M., Faezirad, M. (2018). Efficiency estimation using nonlinear influences of time lags in DEA Using Artificial Neural Networks. Industrial Management Journal, 10 (1), 17–34. doi: http://doi.org/10.22059/imj.2018.129192.1006898
16. Parapuram, G., Mokhtari, M., Ben Hmida, J. (2018). An Artificially Intelligent Technique to Generate Synthetic Geomechanical Well Logs for the Bakken Formation. Energies, 11 (3), 680. doi: https://doi.org/10.3390/en11030680
17. Prokoptsev, N. G., Alekseenko, A. E., Kholodov, Y. A. (2018). Traffic flow speed prediction on transportation graph with convolutional neural networks. Computer Research and Modeling, 10 (3), 359–367. doi: https://doi.org/10.20537/2076-7633-2018-10-3-359-367
18. Dudnyk, V., Sinenko, Y., Matsyk, M., Demchenko, Y., Zhyvotovskyi, R., Repilo, I. et. al. (2020). Development of a method for training artificial neural networks for intelligent decision support systems. Eastern-European Journal of Enterprise Technologies, 3 (2 (105)), 37–47. doi: https://doi.org/10.15587/1729-4061.2020.203301
19. Bodyanskiy, Ye., Pliss, I., Vynokurova, O. (2013). Flexible wavelet-neuro-fuzzy neuron in dynamic data mining tasks. Oil and Gas Power Engineering, 2 (20), 158–162. Available at: http://nbuv.gov.ua/UJRN/Nge_2013_2_18
20. Haykin, S. (2009). Neural networks and learning machines. Pearson, 906. Available at: http://dai.fmph.uniba.sk/courses/NN/haykin.neural-networks.3ed.2009.pdf
21. Nelles, O. (2001). Nonlinear System Identification. Springer, 786. doi: https://doi.org/10.1007/978-3-662-04323-3
22. Wang, L.-X., Mendel, J. M. (1992). Fuzzy basis functions, universal approximation, and orthogonal least-squares learning. IEEE Transactions on Neural Networks, 3 (5), 807–814. doi: https://doi.org/10.1109/72.159070
23. Kohonen, T. (1995). Self-Organizing Maps. Springer, 362. doi: https://doi.org/10.1007/978-3-642-97610-0
24. Kasabov, N. (2003). Evolving Connectionist Systems. Springer, 307. doi: https://doi.org/10.1007/978-1-4471-3740-5
25. Sugeno, M., Kang, G. T. (1988). Structure identification of fuzzy model. Fuzzy Sets and Systems, 28 (1), 15–33. doi: https://doi.org/10.1016/0165-0114(88)90113-3
26. Ljung, L. (1999). System Identification. Theory for the User. PTR Prentice Hall, Upper Saddle River, 609. Available at: https://www.twirpx.com/file/277211/

27. Otto, P., Bodyanskiy, Y., Kolodyazhniy, V. (2003). A new learning algorithm for a forecasting neuro-fuzzy network. Integrated Computer-Aided Engineering, 10 (4), 399–409. doi: https://doi.org/10.3233/ica-2003-10409
28. Narendra, K. S., Parthasarathy, K. (1990). Identification and control of dynamical systems using neural networks. IEEE Transactions on Neural Networks, 1 (1), 4–27. doi: https://doi.org/10.1109/72.80202
29. Petruk, S., Zhyvotovskyi, R., Shyshatskyi, A. (2018). Mathematical Model of MIMO. 2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T). doi: https://doi.org/10.1109/infocommst.2018.8632163
30. Zhyvotovskyi, R., Shyshatskyi, A., Petruk, S. (2017). Structural-semantic model of communication channel. 2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T). doi: https://doi.org/10.1109/infocommst.2017.8246454
31. Alieinykov, I., Thamer, K. A., Zhuravskyi, Y., Sova, O., Smirnova, N., Zhyvotovskyi, R. et. al. (2019). Development of a method of fuzzy evaluation of information and analytical support of strategic management. Eastern-European Journal of Enterprise Technologies, 6 (2 (102)), 16–27. doi: https://doi.org/10.15587/1729-4061.2019.184394
32. Koshlan, A., Salnikova, O., Chekhovska, M., Zhyvotovskyi, R., Prokopenko, Y., Hurskyi, T. et. al. (2019). Development of an algorithm for complex processing of geospatial data in the special-purpose geoinformation system in conditions of diversity and uncertainty of data. Eastern-European Journal of Enterprise Technologies, 5 (9 (101)), 35–45. doi: https://doi.org/10.15587/1729-4061.2019.180197
33. Gorokhovatsky, V., Stiahlyk, N., Tsarevska, V. (2021). Combination method of accelerated metric data search in image classification problems. Advanced Information Systems, 5 (3), 5–12. doi: https://doi.org/10.20998/2522-9052.2021.3.01
34. Levashenko, V., Liashenko, O., Kuchuk, H. (2020). Building Decision Support Systems based on Fuzzy Data. Advanced Information Systems, 4 (4), 48–56. doi: https://doi.org/10.20998/2522-9052.2020.4.07
35. Meleshko, Y., Drieiev, O., Drieieva, H. (2020). Method of identification bot profiles based on neural networks in recommendation systems. Advanced Information Systems, 4 (2), 24–28. doi: https://doi.org/10.20998/2522-9052.2020.2.05
36. Kuchuk, N., Merlak, V., Skorodelov, V. (2020). A method of reducing access time to poorly structured data. Advanced Information Systems, 4 (1), 97–102. doi: https://doi.org/10.20998/2522-9052.2020.1.14
37. Shyshatskyi, A., Tiurnikov, M., Suhak, S., Bondar, O., Melnyk, A., Bokhno, T., Lyashenko, A. (2020). Method of assessment of the efficiency of the communication of operational troop grouping system. Advanced Information Systems, 4 (1), 107–112. doi: https://doi.org/10.20998/2522-9052.2020.1.16