SILESIAN UNIVERSITY OF TECHNOLOGY PUBLISHING HOUSE

SCIENTIFIC PAPERS OF SILESIAN UNIVERSITY OF TECHNOLOGY 2020
ORGANIZATION AND MANAGEMENT SERIES NO. 149

DATA FUSION IN THE DECISION-MAKING PROCESS BASED ON ARTIFICIAL NEURAL NETWORKS

Janusz DUDCZYK[1]*, Łukasz RYBAK[2], Zdzisław JEZIERSKI[3]

[1] Institute of Computer Science and Technology, Stefan Batory State University; jdudczyk@pusb.pl, ORCID: 0000-0001-7169-6824
[2] Institute of Computer Science and Technology, Stefan Batory State University; lrybak@pusb.pl, ORCID: 0000-0002-2920-7326
[3] Institute of Security Sciences, Stefan Batory State University; zdzislawjezierski@wp.pl, ORCID: 0000-0002-2428-7160
* Correspondence author

Purpose: The term data fusion is often used in various technologies, where a significant element is the ability of combining data of different typology coming from diverse sources. Currently, the issue of DF is developing towards interdisciplinary field and is connected with 'agile' data (information) synthesis concerning phenomena and objects. Optimal environment to carry out data fusion are SN (Sensor Networks), in which DF process is carried out on a data stage, most often automatically with the use of probable association algorithms of this data. The purpose of this article was an implementation of a neural network and its adaptation in the process of data fusion and solving the value prediction problem.
Design/methodology/approach: The conducted experiment was concerned with modelling artificial neural network to form radiation beam of microstrip antenna. In the research the MATLAB environment was used.
Findings: The conducted experiment shows that depending on the type of output data set and the task for ANN, the effect of neural network's learning is dependent on the activation function type. The described and implemented network for different activation functions learns effectively, predicts results as well as has the ability to generalize facts on the basis of the patterns learnt.
Research limitations/implications: Without doubts, it is possible to improve the model of a network and provide better results than these presented in the paper through modifying the number of hidden layers, the number of neurons, learning step value or modifying the learning algorithm itself.
Originality/value: The paper presents the implementation of the sensor network in the context of the process of data fusion and solution prediction. The paper should be read by persons which research interests are focused at the decision support by the information and communication technologies.

Keywords: data fusion, decision-making process, sensor networks, artificial neural network.

Category of the paper: Research paper.

1. Introduction

Recently, there has been a huge scientific interest in problems connected with data fusion. Generally, 'data fusion' means a synergy of information, which is used in many different fields of science and technology such as medicine, robotics and army. In a contemporary SN (Sensor Network), this fusion takes place on a data level, and this concept comes down to 'multi-sensor data fusion' carried out in SN. Depending on the level on which the fusion is carried out (which leads to synergy effect), it may be divided into data fusion, information fusion and knowledge fusion. Such an attitude is directly correlated with the flow and data processing procedure in multi-sensor information systems (Figure 1), where data are a set of single facts, calculations and observations. A piece of information appears as a result of collection, processing and aggregation of data. The knowledge at the end of this chain is the ability to use information and build a conceptual model. This knowledge is directly connected with 'the Continuum of Understanding', defined by Nathan Shedroff in the aspect of OODA (Observe Orient Decide Act) decision making model (Dudczyk and Rybak, 2018; Rybak and Dudczyk, 2019).

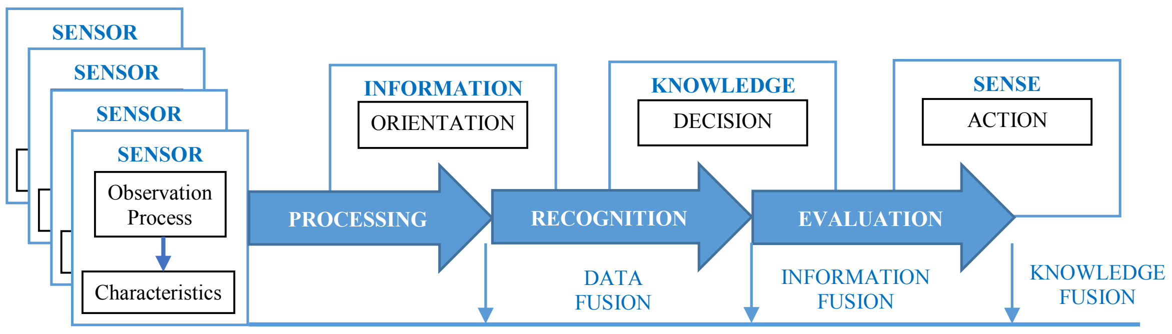

Figure 1. Data flow scheme in multi-sensor information sensor. The source: own study.

Data fusion is a process of effective combination of data, which comes from many different sensors. It is based on a wide range of processes concerning data analysis, synthesis, processing and correlation for the purpose of its further aggregation. Information fusion is a process of combination of information (processed data), whereas knowledge fusion is a process of synthesis concerning deeply analysed knowledge. The fusion process is often carried out offline. In case of a multi-sensor platform (for different types of sensors), the fusion process should be examined on an information platform. Methods used in the problem concerning data fusion include classical reasoning, Bayesian reasoning, Dempster-Shafer theory, Dezert-Smarandache theory, fuzzy set theory, cluster analysis, expert system, adaptive neural networks and genetic algorithms.

Taking into consideration methods and techniques of data fusion as far as sensor networks are concerned, it is necessary to define the speed of data processing in such networks as well as the accuracy of the received data and acceptable load of sensor network. The above-mentioned is significant as defining particular threshold values of those above makes it possible to answer the question whether a particular data fusion technique can be used in a sensor network or not.

2. Application of artificial neural network in the process of data fusion and solution prediction

Nowadays, artificial neural networks are useful not only in a data fusion process but also while solving decision making problems. Without ANN it would be impossible or partially possible to solve such cases effectively. The use of artificial neural network has a number of advantages due to its flexibility of applications, as a result it is hard to list all feasible ways of using it. Nevertheless, artificial neural networks enable us to solve cases concerning recognition, classification and identification of non-trivial patterns such as speech or images. ANN makes it also possible to predict solutions on the basis of initial data and thus provides an increase in accuracy or value estimation on the basis of a large number of parameters in the data fusion process. The possibilities of the implemented neural network will be used to predict the so far unknown solution, on the basis of known and presented solutions of an artificial neural network. On the basis of the presented fragmentary knowledge, ANN will need to find a general solution of the problem and use it effectively with the so far unknown value parameters.

2.1. Cost function and evaluation results of artificial neural network

In order to assess the quality of a created artificial neural network and make it possible to optimize it, it is necessary to have a way to evaluate the accuracy of the results coming from the network. In order to do it, a good option is to present a data set to an artificial neural network. Solutions to such data should be already known and compared with reactions from the network. The above mentioned leads to a decision process, where ANN carries out input data fusion. In order to compare already known results and results coming from the network, a cost function is introduced. The cost function is also necessary for the ANN's learning process to be carried out, on the basis of already known results. The most frequent methods to calculate the cost function are normed vector spaces L^1 and L^2,which definitions derive from Lebesgue norm, and their index means a particular case of a generalized norm L^p, (Anatriello, 2014). The L^1 norm, Least Absolute Deviations, (Hurvich and Tsai, 1990) is described by the equation (1), (where S – is a total cost/error for n example, i – is the index of a particular example, y_i – is a known solution, whereas z_i – is the value of output signal in the neural network for i example). It is the fastest method, however, there is no guarantee that an unequivocal solution exists.

$$S = \sum_{i=1}^{n} |y_i - z_i| \tag{1}$$

Geometrical interpretation in two-dimensional space, the distance between A and B calculated with the use of the L^1 norm (Figure 2) may be equated with any distance of a curve with horizontal coordinates and vertical coordinates of the curve's consecutive points are non-decreasing. It can be easily noticed why there are so many potential solutions for which the distance is the same. The L^2 norm, according to the equation (2) is the most difficult to calculate in terms of computing power, however, it has better parameters concerning stability of the sought after solution, where S – is a total cost/error for n example, i – is a particular example's index, y_i – is a known solution, whereas z_i – is the value of the output signal in a neural network for i example.

$$S = \sum_{i=1}^{n} (y|i - z_i)^2 = ||Y - Z|| \tag{2}$$

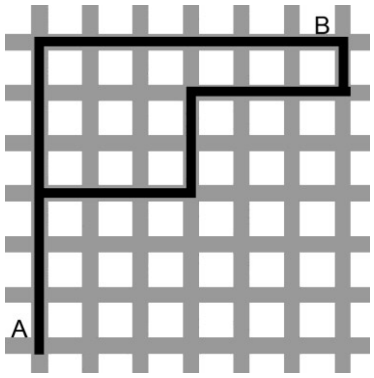

Figure 2. Calculating distances in two-dimensional metric space with two examples of ways. Source: own study.

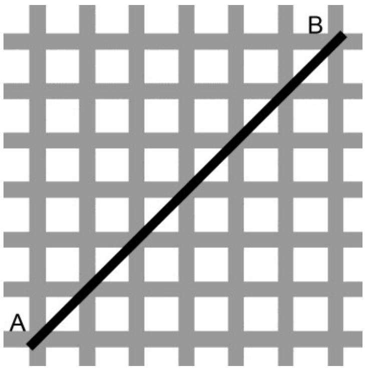

Figure 3. Calculating distances in two-dimensional metric space. Source: own study.

Geometric interpretation in two-dimensional space, the distance between A and B, calculated with the use of the L^2 norm (Figure 3) may be equated with a line that is coming directly between and may be equated with the Euclidean distance between two points. In further dissertation, the L^2 norm is used to calculate the AAN's cost function. It is necessary to define and introduce the aim function in order to deliberate on the quality of results coming from the AAN. Furthermore, one of the elements necessary for a neural network to learn and to evaluate is defining the aim function. During the prediction process, it is important to test a network before, i.e. on a particular data set, and the aim function makes it possible to asses if a network achieves the expected result as well as points out the direction of changes in order to improve the above result.

2.2. Matrix calculation in artificial neural network

Artificial neural network can be presented intuitively with the use of a graph of connections however, it is easier to present the problem of solution prediction and input data fusion in algebraic form. If n means a number of features which a neural network will take into consideration, and m is a number of examples that will be analysed, then the value i of feature for j example can be presented with the equation (3), where i – means the feature's index, n – is the number of all features, j – is the index's example and m – is the number of all examples.

$$x_{ij}\{i \in N: i < n; j \in N: j < m\} \tag{3}$$

All features for a single example then form a vector, where each component of the vector is a value of the next feature, according to the equation (4).

$$X_j = [x_{j1}, x_{j2}, x_{j3} .. x_{jn}] \tag{4}$$

Taking into consideration of the above, it is worth presenting all examples as an **X** matrix, according to the equation (5), where in the next columns of the matrix, there are further features, and lines of the matrix are sets of features describing e.g. an object which is tested.

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & x_{13} \cdots x_{1n} \\ x_{21} & x_{22} & x_{23} \cdots x_{2n} \\ x_{31} & x_{32} & x_{33} \cdots x_{3n} \\ \vdots & \vdots & \vdots \ddots \vdots \\ x_{m1} & x_{m2} & x_{m3} \cdots x_{mn} \end{bmatrix} \tag{5}$$

By analogy, if m is a number of different examples which will be analysed and for which there is a known correct solution, then the value of the solution is in accordance with the equation (6). By analogy, solution values will create a vector according to the equation (7), where T – is a transposition sign.

$$y_j\{j \in N: j < m\} \tag{6}$$

$$\mathbf{Y} = [y_1, y_2, y_3 \dots y_m]^{\mathrm{T}} \tag{7}$$

A pair of vectors $\{X_1, Y_1\}$ is a single example, which consists of features and the expected solution, a pair of $\{\mathbf{X}, \mathbf{Y}\}$ matrices, however, describes the whole tested area. If the *k* layer of a network with the index has *m* neurons and the next layer has *n*, then synaptic weights of layer with *k* index can be presented as a matrix in accordance with equation (8), where w_{mn} – is connection weight between *m*-this neuron of the *k*-1 layer and the neuron of the *k* layer. In particular, if a *k*-1 layer does not exist, it means that the weight will be used for appropriate value of the **X** input matrix's component.

$$\mathbf{W}^k = \begin{bmatrix} w_{11} & w_{12} & w_{13} \cdots w_{1n} \\ w_{21} & w_{22} & w_{23} \cdots w_{2n} \\ w_{31} & w_{32} & w_{33} \cdots w_{3n} \\ \vdots & \vdots & \vdots \ddots \vdots \\ w_{m1} & w_{m2} & w_{m3} \cdots w_{mn} \end{bmatrix} \tag{8}$$

As there is no need for a matrix to be square, the whole one way neural network, in which there is only one activation function used, can be described explicitly as a $\{f, W\}$ pair where *f* – is the neuron activation function, while **W** – is an ordered sequence of layers described with the use of matrix. An output signal of the first layer of neuron can be described with the following equation (9), while an output signal of *m* layer can be defined in accordance with the equation (10), or recursively with the equation (11).

$$z^1 = f(\mathbf{X} \times \boldsymbol{W}^1) \tag{9}$$

$$Z^m = f \dots (f(f(f(\mathbf{X} \times \mathbf{W}^1) \times \mathbf{W}^2) \times \mathbf{W}^3) \dots \times \mathbf{W}^m) \tag{10}$$

$$Z^m = \begin{cases} f(\mathbf{X} \times \mathbf{W}^m) \wedge & \text{for } m = 1 \\ f(Z^{m-1} \times \mathbf{W}^m) \wedge & \text{for } m > 1 \end{cases} \tag{11}$$

ANN solves a known problem if $Z^m = Y$, where Z^m – is a network resulting signal vector of *m* layer network, while *Y*– is a vector of a known solution, which means that the last network layer returns incorrectly expected result vector. In practice, a network should return a result vector with the least possible error, according to the equation (12), where S – is the aim function, *S(W)* – is a total error of a network, *n* – means the number of examples, y_i – means the value of a known solution for i -this example, z_i^m – means the network value of resulting signal *m* – this layer network for i example, Z^m – is a resulting signal vector of the network, *m* – the layer network, while Y – is a vector of a known solution.

$$S(W) = \sum_{i=1}^{n} \frac{(y_i - z_i^m)^2}{2} = \frac{\|Y - Z^m\|}{2} \tag{12}$$

A ½ coefficient is often added in order to simplify the first derivative for the aim function. Carrying out the described procedure makes it possible to fast calculate the resulting values of a neural network of a feedforward type. The introduced algorithm not only calculates resulting values for a network with set weights, but also is a base to introduce algorithms which enable a network to learn.

2.3. Preparing a set of learning data and feature scaling process

In the process of preparing a set of learning data, the process of feature scaling is carried out in accordance with (13), where i – is an index of a particular feature, x_i' – is a single feature value after the feature scaling process, x_i – is a feature value before this process, $\acute{X}$ – is an arithmetic mean of the feature, calculated for the whole population while σ – is standard deviation of a feature.

$$x_i' = \frac{x_i - \acute{X}}{\sigma} \tag{13}$$

For networks which produce results in form of a [0,1] set of real numbers from the interval, it is necessary to carry out transformation of resulting values and project them into an interval, where there is a network in accordance with the equation (14), where i – is an index of a particular resulting value, Y – is a resulting vector, y_i' – is a single resulting value before feature scaling process, $min(\ldots)$ – is a minimum value among known resulting values, while $max(\ldots)$ – is the maximum value among all known resulting values.

$$y_i' = \frac{y_i - min(Y)}{max(Y) - min(Y)} \tag{14}$$

In order to receive a result in an appropriate scale it is necessary to carry out a reverse process for the received results, in accordance with the equation (15).

$$y_i = y_i' * \left(max(Y) - min(Y)\right) + min(Y) \tag{15}$$

All data, on which prediction algorithms are presented, i.e. algorithms based on forward propagation, gradient descent and backpropagation are normalized before, and the received values are reduced to numerical form.

3. The achieved results, prediction process realization

Prediction of values on the basis of many features is similar to the problem of function interpolation or extrapolation for many variables. Some extremely efficient numerical algorithms are used in these cases, however, if the dependence of resulting values on features is more complicated than linear and hard to precise with the use of polynomial or a function with a simple structure, numerical methods become more complicated in terms of analysis and calculation. Simultaneously, the level of complexity to solve the problem increases in a nonlinear way with relation to the number of the analysed variables. A neural network deals with the prediction problem of complicated relations much better than numerical methods. It is also a relatively simple model with increased flexibility as changes in assumptions require only small corrections in a model of an artificial neural network.

3.1. Presentation of neural network learning process, comparative effectiveness analysis of different types of activation function

Speed and learning possibilities of ANN for the presented data set depend mainly on the implemented network structure and accepted activation function. In order to do it, a simple model of a network has been prepared, with three layers with one output neuron.

$$W^1 = \begin{bmatrix} 0.02 & 0.06 & 0.07 \\ 0.31 & 0.04 & 0.09 \\ 0.05 & 0.61 & 0.09 \end{bmatrix}, W^2 = \begin{bmatrix} 0.04 & 0.78 & 0.39 \\ 0.39 & 0.49 & 0.71 \\ 0.59 & 0.24 & 0.51 \end{bmatrix}, W^3 = \begin{bmatrix} 0.03 \\ 0.39 \\ 0.71 \end{bmatrix} \tag{16}$$

The designed ANN was initialized with particular weights and then 500 steps of the learning procedure started with a particular speed value of the learning coefficient $\alpha = 0{,}1$.

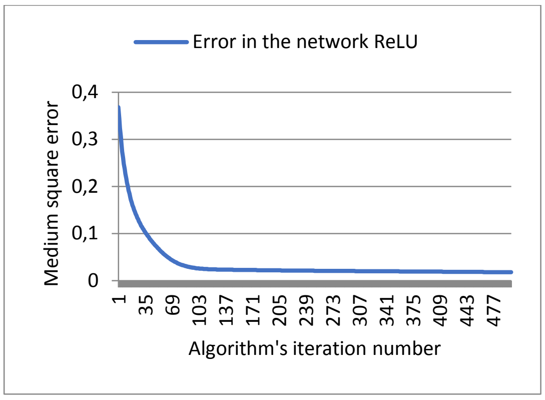

Figure 4. Evolution of a network with activation function - ReLU. Source: own study.

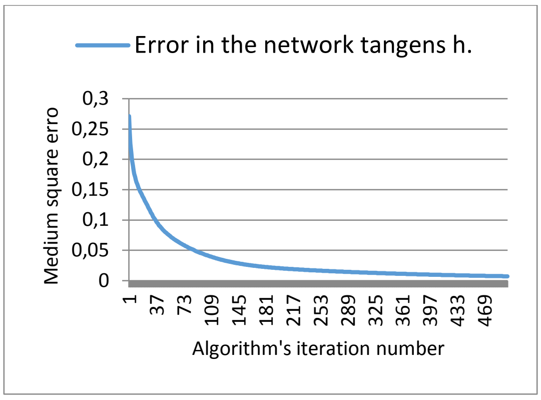

Figure 5. Evolution of a network with activation function- hyperbolic tangent. Source: own study.

From the graph (Figure 4) it can be concluded that a network with activation function reLU (rectifier Linear Unit) has an error of ~0.025 after about 90 steps and since then the progress slows down. After 500 steps, the total error for this sample is 0.01774545. From the graph presented in the Figure 5 it can be concluded that a network with activation function, hyperbolic tangent has no clear point, where learning has no effects. After about 100 steps, an error is 0.043 and it is bigger than in reLU, however, after 500 steps, the total error for this sample is 0.00714321, which is the best result among all tested activation functions of artificial neural network.

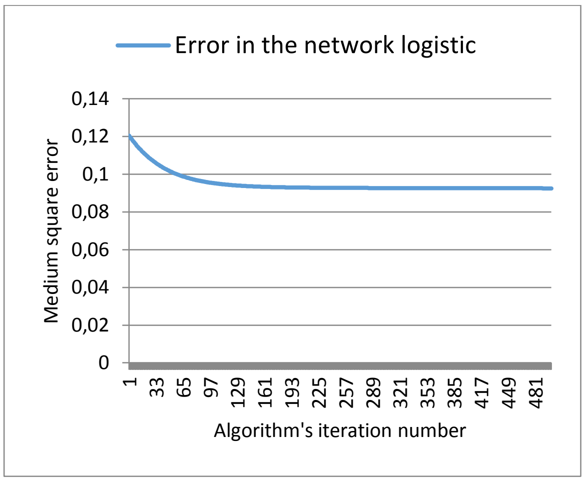

Figure 6. Evolution of a network with activation function – logistic. Source: own study.

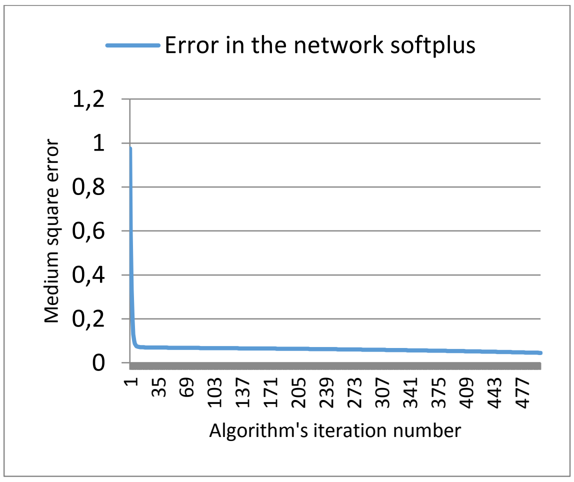

Figure 7. Evolution of a network with activation function – SoftPlus. Source: own study.

Activation function – logistic, which results are presented in Figure 6, provides no satisfying results, and the error after 500 steps is 0.09255047. From the graph presented in Figure 7 it can be concluded that a network with activation function – SoftPlus (so-called sigmoid for step function – so the super-soft and super-smooth function – reLU) reacts to learning with lightning speed and after just 10 steps the error decreases to 0.074. Further learning still decreases slowly this value even to 0.04556837. Figure 8 presents in general the received results for six different activation functions of an artificial neural network.

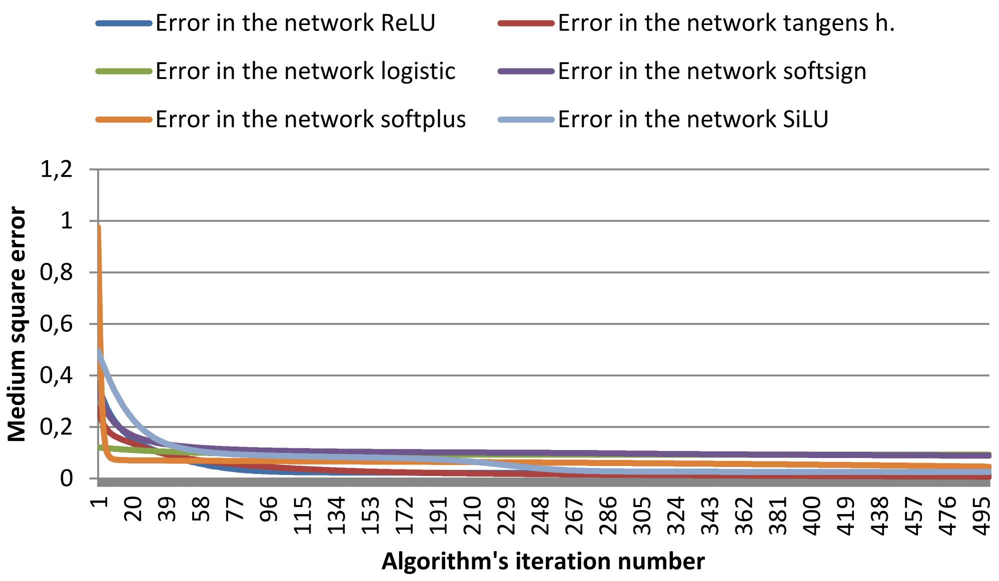

Figure 8. Overall presentation of evolution of a network for six different activation functions. Source: own study.

Table 1.
The comparison of results of artificial neural network's learning for a few different activation functions

Activation function type	The minimum error value after 500 steps
Hyperbolic tangent	0.00714321
reLU	0.01774545
SiLU	0.02476649
SoftPlus	0.04556837
SoftSign	0.08839626
Logistic	0.09255047

The fastest adjustment occurred in case of ANN with the activation function – SoftPlus although its error was initially the biggest. It could have been predicted before as this function has a tendency to grow fast, thus changes in synaptic weights are very dynamic. The smallest errors occurred in case of the activation function – reLU and hyperbolic tangent. Both of these functions also coincide fast with the solution, so in the process of solution prediction it is useful to apply them.

3.2. Comparative analysis of effectiveness of different types of activation functions in value prediction for unknown data sets

A true ability test of ANN should be carried out on a data set with which the network has had no contact before. In order to check how particular activation functions deal with an unknown data set, a set of five examples is prepared that are unknown to the implemented network. These examples are used to evaluate predictive effectiveness of ANN. The fact that the network learns a data set does not mean that the network can deal with unknown data. It may turn out that ANN is overfitted and learned so well that it may lose the ability of abstractive estimation. In order to carry out tests, ANN with three layers and one output neuron is implemented.

$$W^1 = \begin{bmatrix} -0.69677980 & -0.13805204 & -0.23110544 \\ 0.97319321 & 0.44082334 & 0.98965416 \\ -0.04369450 & -0.381727090 & 0.64158716 \end{bmatrix}$$

$$W^2 = \begin{bmatrix} -0.76102662 & -0.64227951 & -0.83463912 \\ -0.72989451 & 0.88778036 & 0.72958331 \\ -0.36639262 & -0.63425962 & -0.84816601 \end{bmatrix} \tag{17}$$

$$W^3 = \begin{bmatrix} 0.57042009 \\ 0.39659853 \\ -0.72826755 \end{bmatrix}$$

The implemented ANN is initialized with weights in accordance with (17), and then 300 steps of the procedure are started. During the learning procedure, the error is monitored both for a known and an unknown data set.

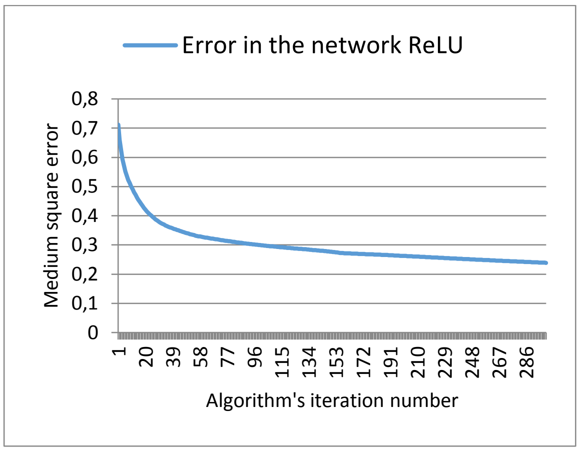

Figure 9. Evolution of the error in the network reLU for a known data set. Source: own study.

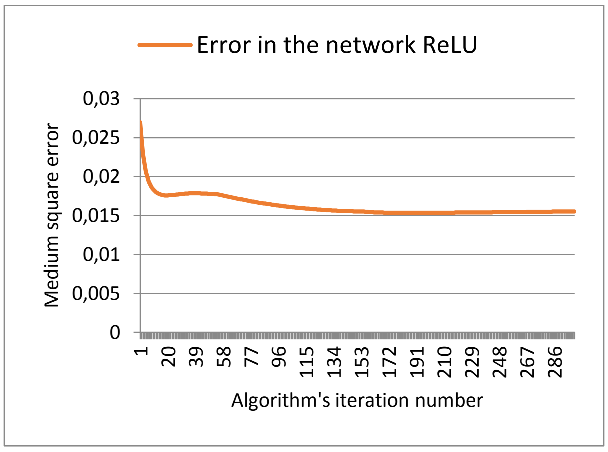

Figure 10. Evolution of the error in the network reLU for an unknown data set. Source: own study.

Figures 9 and 10 present a depiction of a layout of errors in learning ANN for the activation function – reLU in case of known and unknown learning data sets. Although the error for a known learning data set constantly decreases (Figure 9), for the data set which is unknown for the network, after achieving about 180-190 steps the error starts to increase. These are clear signs of ANN overlearning and after 190th step the network learning is pointless.

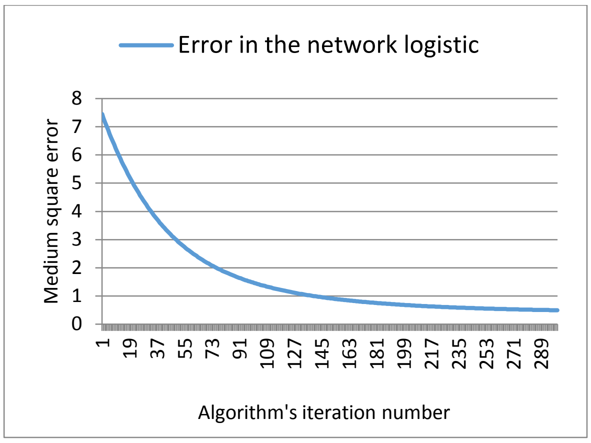

Figure 11. Evolution of the error in the network - logistic for a known data set. Source; own study.

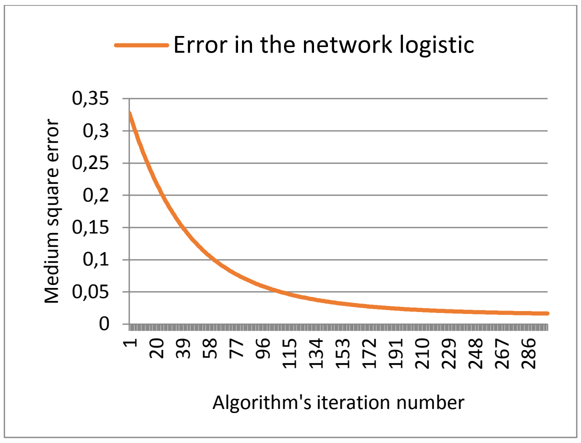

Figure 12. Evolution of the error in the network – logistic for an unknown data set. Source: own study.

Figures 11 and 12 present a depiction of a layout of errors in learning ANN for the function -logistic in case of a known and unknown learning data sets. On the basis of the figure above it can be noticed that ANN with the function type- deals with this task quite well. Although the total error is still bigger than in other activation functions used, there are no signs of overlearning.

Table 2.
Comparison of results of learning artificial neural network for a few different activations

Activation function type	Minimum error value after 300 steps of learning algorithm	Minimum error value after 300 steps of learning algorithm for an unknown data set	Average minimum error value
SoftPlus	0.52357342	0.01328551	0.268429465
ReLU	0.23911525	0.01535639	0.12723582
Logistic	0.49193433	0.01646150	0.254197915
SiLU	0.41188665	0.01699956	0.214443105
SoftSign	0.29392890	0.01779733	0.155863115
Hyperbolic tangent	0.15532112	0.01901063	0.087165875
Gausiann	3.01723971	0.26007022	1.638654965

The described and implemented ANN, based on the activation function – hyperbolic tangent deals with the learning process in the best way (Table 2). However, the best results for an unknown data set are surprisingly provided by the activation function – SoftPlus. Without doubts, it is possible to improve the model of a network and provide better results than these presented above by modifying the number of hidden layers, the number of neurons, learning step value or modifying the learning algorithm itself. It is important to underline that the best results are given by neural networks for which activation functions are selected on the layer level.

4. Conclusion

The aim of this article is implementation of a neural network and its adaptation in the process of data fusion and solving the value prediction problem. The conducted experiment shows that depending on the type of output data set and the task for ANN, the effect of neural network's learning is dependent on the activation function type. The described and implemented network for different activation functions learns effectively, predicts results as well as has the ability to generalize facts on the basis of the patterns learnt.

References

1. Anatriello, G. (2014). Iterated grand and small Lebesgue spaces. *Collectanea Mathematica*, *65(2)*, 273-284.
2. Cho, S.B., and Kim, J.H. (1995). Combining multiple neural networks by fuzzy integral for robust classification. *IEEE Transactions on Systems, Man, and Cybernetics, 25(2),* 380-384. doi.10.1109/21.364825.
3. Dudczyk, J., and Rybak, Ł. (2018). Adaptive Decision Support System in Network Centric Warfare Process. *Elektronika: konstrukcje, technologie, zastosowania*, *nr 7*, 9-42. doi: 10.15199/13.2018.7.10.
4. Hurvich. C., and Tsai, C.-L. (1990). Model selection for least absolute deviations regression in small samples. *Statistics & Probability Letters, 9(3)*, 259-265.
5. Jolly, K.G., Ravindran, K.P., Vijayakumar, R., and Kumar, R.S. (2006). Intelligent decision making in multi-agent robot soccer system through compounded artificial neural networks. *Robotics and Autonomous Systems, 55(7),* 589-596. doi.org/10.1016/j.robot.2006.12.011.
6. Kahraman, C., Ruan, D., and Dogan, I. (2003). Fuzzy group decision-making for facility location selection. *Information Sciences, 157*, 135-153. Doi.org/10.1016/S0020-0255(03)00183-X.
7. Rybak, Ł., and Dudczyk, J. (2019). Increasing the information superiority on the modern battlefield through the use of virtual reality systems. *Security and Defence Quarterly*, *25(3).*
8. Zhou, J., and Civco, D.L. (1996). Using Genetic Learning Neural Networks for Spatial Decision Making in GIS. *Photogrammetric Engineering & Remote Sensing, 62(11),* 1287-1295. Doi. 0099-1112/96/6211-1287$3.00/0.