OPTIMIZING SIMULATION MODELS OF AN ARTIFICIAL NEURAL NETWORK FOR DIGITAL RECOGNITION

*Yoana Ivanova**

Department of Telecommunications
New Bulgarian University
Bulgaria

* Corresponding Author, e-mail: yivanova@nbu.bg

Abstract: This paper is considered to be a continuation of a previous publication devoted to adaptive digitalization methods and digital transformation trends for security (IJIITS, № 3, 2020). The aim of this research is to present how digitized data obtained by using scanning software can be imported in a simulation model of an Artificial Neural Network (ANN) with backpropagation of error built in software SIMBRAIN contributing to improvement of its capability to self-learn in digital recognition by the proposed method of optimization.

Key words: digitized data, optimization, simulation modelling, artificial neural network, backpropagation error.

1. INTRODUCTION

Artificial Neural Networks (ANNs) are able to recognize typographic symbols (letters, punctuation marks), pictographic symbols (emojis), emotional expressions presented by graphical images, as well as more complex sequences of symbols (words, phrases) and speech. For example, ANNs with backpropagation of error (BPE) can recognize a great variety of images that have been digital transformed in a binary code by a scanning software with a conventional type of interface.

The product selected for this research allows the user to draw an image with a high level of abstraction and low level of complexity (2D object or a rendered image of a 3D object) in a matrix of pixels. Therefore, *frmScanner* used is suitable mainly for digital transformation of letters, numbers, emojis, signs, diagrams and some geometric primitives or shapes. Actually, such a digitization process can be performed manually, but the advantage of the usage of a software is that each part of the graphical image can be digital transformed in a numerical value automatically.

As it is known, a physical object can be represented as a point cloud by digital transformations. Although BPE is often determined as a highly time-consuming algorithm [1, 2], it is still known as one of the advanced methods for classification of objects on the basis of their main geometric parameters [3] and not only. Therefore, using basic geometric figures the author aims to demonstrate the process of transforming a 2D digital image in a binary code for subsequent recognition from ANNs, as well as to propose a reliable method of optimization in order to reduce the calculation time and to minimize the error.

SIMBRAIN is an open source software for simulating neural circuits written in Java which is characterized by intuitive design [4]. It is very suitable for the main purpose of this research related to modelling a neural network that is self-learning to correctly recognize digital images in a relatively short time.

The main purpose of the author is to propose a reliable algorithmic method for optimization of a neural network with BPE using a narrowly tuned simulation model built in a software with capabilities for precise settings. Its contribution is mostly expressed in the conception for realizing the optimization as a two-stage process performed using two absolutely independent software resources the results of which are expected to jointly affect the final simulation outcome. The first stage of the overall optimization process is related to the matrix of pixels, while the focus of the second stage is the achieving maximum efficiency of the ANN.

Besides, a good practice for selecting a suitable product for simulation modelling requires a scientific justification of the choice based on a number of expert criteria and practical knowledge for the software products. Therefore, the comparative analysis of products which could be used for simulating neural networks further contributes to this work. The preliminary analysis results show that the main advantage of SIMBRAIN is related to the possibility of conducting an in-depth empirical research of a very specific neural network selected among a wide variety of built-in simulation models based on complex mathematical algorithms.

In Section 2 is described a preliminary simulation study including a comparative analysis between the closely specialized simulation software SIMBRAIN and the universal simulation product NetLogo [5] in order to establishing the capabilities of the products for simulation of a neural network with BPE. This part of the paper includes a presentation and summary analysis of related work in the field of simulation modelling of ANNs. In this paper the error backpropagation algorithm is explained in Section 3, while the experimental research related to optimization of simulation models of ANNs with BPE according to selected criteria is presented in Section 4. The simulation results are summarized and analysed in Section 5.

2. RELATED WORK

The error backpropagation algorithm can be difficult for realization in a simulation environment if the level of complexity of the neural network increases due to the addition of a large number of neurons in the layers. Consequently, a simulation of an ANN with BPE can not be directly completed in every simulation software even if it has a wide range of applications and an advanced built-in model library.

In SIMBRAIN the simulation starts directly after entering the input parameters, while in NetLogo it is required the basic programme code of the built-in model of a perceptron to be transformed by the user for the purposes of the specialized research. For example, each new artificial neuron needs not only to be added in the ANN, but also to be placed in the right place by accurate setting of the coordinates, dimensioned and colored appropriately.

The training algorithm in the model in NetLogo is also BPE. *"The propagate phase propagates the activation values of the input nodes to the output node of the network. In the backpropagate phase, the error in the produced value is passed back through the network layer by layer"* [6].

Actually, if the main aim of a study is not the researcher to develop professional skills for modelling neural networks by a programme language, but to simulate directly advanced optimization processes, then in this case SIMBRAIN can be the preferable choice because of the time reducing for conducting experiments.

This contributes not only to the effectiveness of the working process, but also to the reliability of the generated results, because the risk of human mistake in the mathematical algorithms is also minimized. The only inaccuracies may be a result of incorrectly entered input parameters (for example binary codes of neurons or their activation values) which can be easily verified by the user and corrected if it is necessary.

There are examples of ANNs with BPE built in SIMBRAIN modelled for initially recognition of basic symbols with middle level of complexity like letters, numbers or emojis that could be presented in matrixes with minimum resolution (2x2 or 3x3) [7]. They are especially suitable for education purposes and precede the optimization process that is needed if the digitized symbols are characterized by a high level of complexity as in the current research.

This means that if a symbol is complex, the ANN needs a very recognizable pattern of it. Therefore, the matrix should be with high resolution (in this case 8x8) and respectively the neural network requires more neurons in the input and processing layers. As a result the calculation time and the BPE increases which lowers the overall efficiency of ANN. Initially, optimization can be performed by conducting a series of experiments in a reliable simulation environment, which is a relatively inexpensive but an advanced method.

The comparative analysis of the experimental researches in SIMBRAIN and NetLogo is shown in Table 1. The following abbreviations are accepted: BPE_{opt} – an optimized ANN with BPE; MR – matrix resolution; IN – number of input neurons; PL – number of processing layers; PN – number of processing neurons respectively in each of the two processing layersPL_1 and PL_2; ON – number of output neurons; E – the value of BPE.

Table 1.

Software	***ANN***	***Symbols***	***MR***	***IN***	***PL***	***PN***	***ON***	***E***
NetLogo	*BPE*	*built-in (basic)*	2x2	4	1	2	1	0.0000
SIMBRAIN	*BPE*	*digitized (basic)*	3x3	9	1	5	2	0.0000
	BPE_{opt}	*digitized (middle complexity)*	4x4	16	2	PL_1 – 5 PL_2- 4	2	0.0006
SIMBRAIN	BPE_{opt}	*digitized (high complexity)*	8x8	64	2	PL_1 –12 PL_2- 12	2	0.0029

3. ESSENCE OF THE ERROR BACKPROPAGATION ALGORITHM

The process of designing a neural network with backpropagation of error can be expressed in the following steps:

- **the digital image is presented at the input layer;**
- **the neurons in the processing layer evaluate their outputs using the digital image;**
- **applying the target to the output layer;**
- **setting random values for the weights (w_1, w_2, …w_j) of the neurons in the processing layer.**

The backpropagation error (E) can be expressed by a concept presented in the equation [8]:

$$E = \frac{1}{2}\Sigma\ (target - out)^2 \qquad (1)$$

where:

- ✓ *target – the predicted result;*
- ✓ *output (out) - the actual result.*

- **changing the input parameters in order to minimize the error;**
- **calculating the outputs taking into account the bias (b) [9].**

"The neuron's output (0 or 1), is determined by whether the weighted sum $\Sigma w_j x_j$ *is less than or greater than some threshold value".* The perceptron operates on the basis of the rule:

- *output = 0 if* $\Sigma w_j x_j \leq$ *threshold, where* $x_1, x_2, \ldots x_j$ *are inputs;*
- *output = 1 if* $\Sigma w_j x_j \geq$ *threshold.*

It is possible to use the bias instead of the threshold, as follows:

- *output = 0 if* $w*x + b \leq 0$;
- *output = 1 if* $w*x + b > 0$.

- **calculating the sigmoid or logistic function (σ)**

- *sigmoid neuron* – it has inputs (x_1, x_2, …x_j),weights (w_1, w_2, …w_j) and a bias (b), but its inputs can take also values between 0 and 1 and respectively the output is not 0 or 1. It can be calculated by the formula:

$$\text{out} = \sigma(\text{wx} + \text{b}) = \frac{1}{1+\exp(-\Sigma w_i x_j - b)} \quad (2)$$

$$\sigma(\text{z}) = \frac{1}{1 + e^{-z}} \quad (3)$$

There are different cases according the value of z:

- *z is a large and positive number:* if $z \equiv w*x + b$ then $e^{-z} \approx 0$ and $\sigma(z) \approx 1$;
- *z is a negative number:* if $z \equiv w*x + b$ then $e^{-z} \to \infty$, and $\sigma(z) \approx 0$ [10].

4. OPTIMIZATION OF SIMULATION MODELS OF ANN FOR RECOGNITION OF GEOMETRIC PRIMITIVES

The neural network is designed to recognize the two geometric primitives together. This means that there is a sequence of 8 matrixes per each model to be recognized – 4 spatial orientations of the square and 4 of the cube. Table 2 contains the input parameters inserted respectively in the simulation model before the optimization of the image (M_1) when the square is filled and after its optimization (M_2) when the square is not filled and the number of neurons in the three processing layers is 5. Besides, M_1 and M_2 are optimized by the number of processing layers (N) and neurons (n) in each of them to prove experimentally how the error depends on these parameters.

Table 2.

Parameters	***M1***			***M2***	
Geometric	*filled square*	*outlined cube*		*outlined square*	*outlined cube*
Matrix	***resolution***	***SO***		***resolution***	***SO***
	8x8	8		8x8	8
Processing layers	***N***	2	3	2	3
Number of neurons in all layers	n_{input}	64	64	64	64
	$n_{process}$	4	5	4	5
	n_{output}	2	2	2	2

The selected optimization criteria by the author are as follows:

- ***whether the geometric primitives are filled or not*** – figures can be completely filled with a color or only outlined. It is necessary to examine if this is important for the optimization given that the filled parts are digitized with 1 and the empty ones with 0. Consequently, ANN recognizes the filled and outlined digital images in a different way. The purpose of the author is to analyse in which of the cases this ANN is more effective.
  As can be seen in Fig. 1 the cube can be recognizable if it is partially filled unlike the square. Besides the square has 4 visually identical spatial orientation (SO), while the cube has 4 recognizable spatial orientations (Fig. 2). In connection with the filling of the figures, the two neural networks differ because of the square which is represented in two ways.
  The sequences of spatial orientations start from the square and continue to the orientations of the cube. Each subsequent orientation is a rotation of the image at 90 degrees in a clockwise direction.
- ***number of processing layers (N)*** – it is necessary to be noted that when the resolution of the matrix is more than 4x4 and the neurons in the input layer are more than 16, minimum of 2 processing layers are recommended, while for a matrix 3x3 and 9 neurons in the input layer, 1 processing layer is enough [7]. The neurons in the output layer are two, because the ANN mist recognize two geometric primitives.
- ***number of neurons in the processing layers (n).***

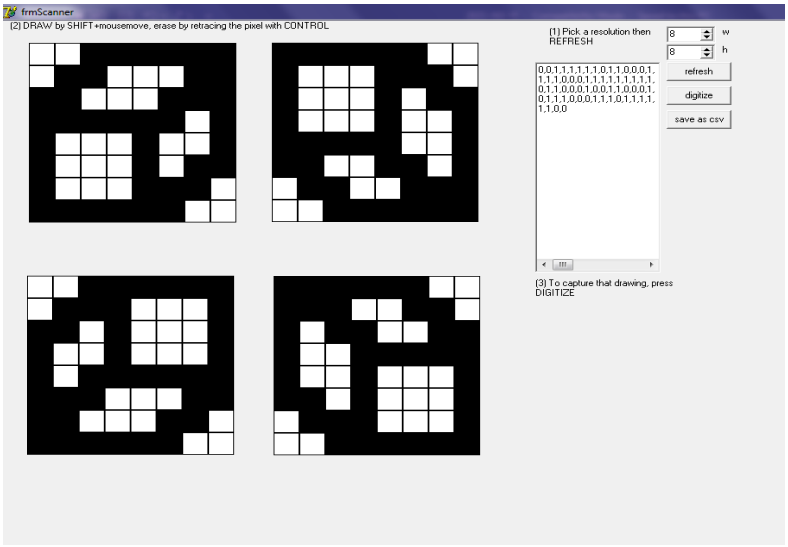

Fig. 1. A representation of an outlined cube and a filled/outlined square in a matrix with a size 8x8.

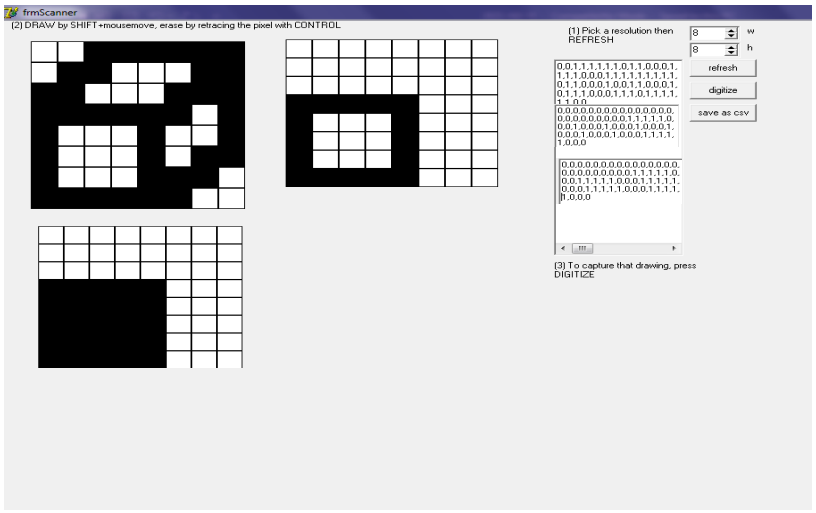

Fig. 2. Main spatial orientations of a cube represented in a matrix with a size 8x8.

The binary codes for all 64 neurons in the input layer and the two neurons in the output layer are inserted respectively in the tables *Input data* and *Target data* shown in Fig. 3. The first simulation model of the ANN M_1 is presented in Fig. 4. The full optimization algorithm is presented in Fig. 5.

#	Neuron_1	Neuron_2	Neuron_3	Neuron_4	Neuron_5	Neuron_6	Neuron_7	Neuron_8
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	0.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0
6	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0
7	0.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0
8	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0

#	Neuron_73	Neuron_74
1	1.0	0.0
2	1.0	0.0
3	1.0	0.0
4	1.0	0.0
5	0.0	1.0
6	0.0	1.0
7	0.0	1.0
8	0.0	1.0

Fig. 3. The binary codes inserted in the tables Input data and Target data.

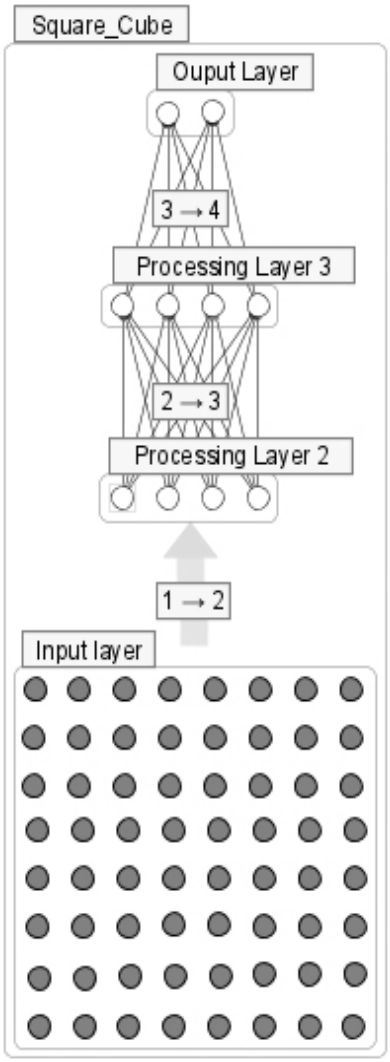

Fig. 4. The basic simulation model of the ANN for digital recognition built in SIMBRAIN.

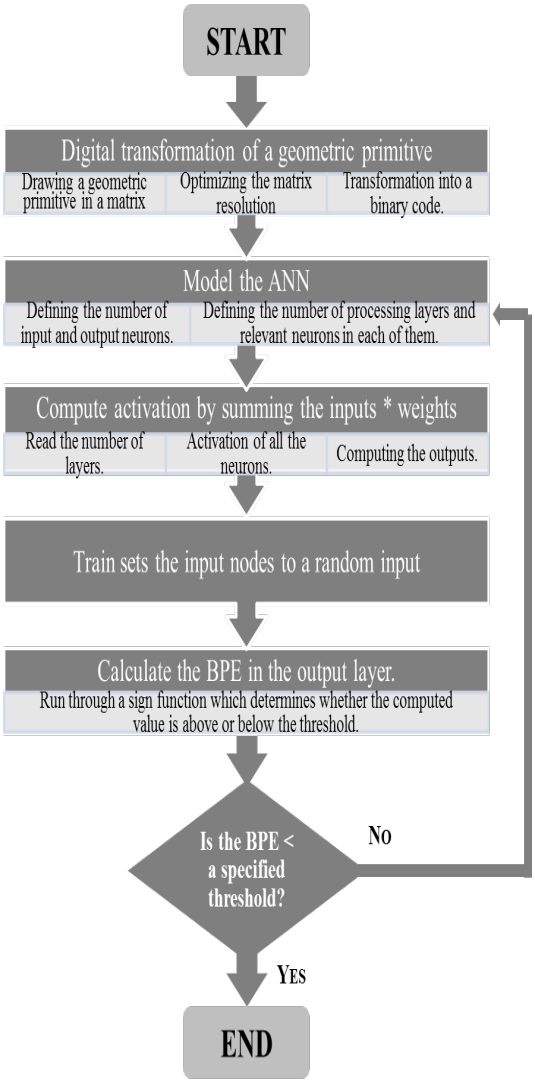

Fig. 5. An algorithmic representation of all steps of the optimization process.

5. ASSESSMENT AND ANALYSIS OF THE SIMULATION RESULTS

The summary assessment and analysis can be made based on the generated simulation results in a total of four scenarios presented in Table 3. Considering the machine learning as an iterative process, it is logical that if I is the number of iterations, then the backpropagation error *(E) = f (I)*.

Table 3.

Parameters	*M_1*				*M_2*		
N	***2***		***3***		***2***		***3***
n	***4***	***24***	***5***	***45***	***4***	***24***	***5***
I	173731	173870	173806	173771	206628	206764	206450
E	0.0042	0.0029	0,0038	0.0053	0.0042	0.0029	0,0038

In Fig. 6 are shown selected screenshots taken during the simulation process in M_1 respectively before and after the optimization by changes in the number of layers and neurons in them. It is obvious that a minimal increase of N and n reduces the error in M_1, but the error increases if n is 3 times more.

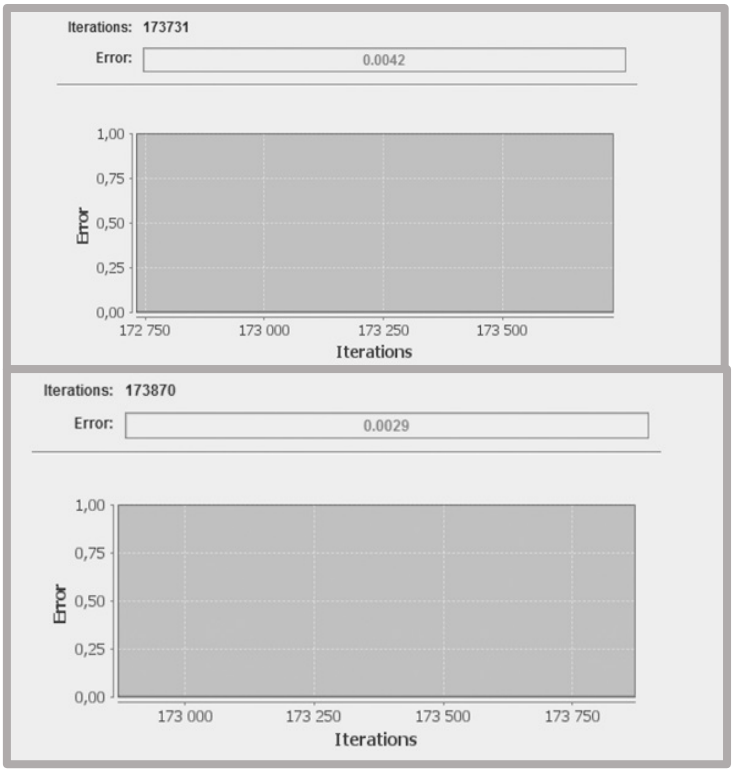

Fig. 6. Screenshot taken during the simulation of M_1 before the optimization and after the first level of the optimization.

With a regard to M_2 it can be concluded that the filling in a figure does not significantly affect BPE even in case that the iterations are more compared to M_1. Consequently, the ANN with two processing layers with 12 neurons per each of them is the most effectively optimized in the current empirical research completed in SIMBRAIN.

Last but not least the result of this simulation with an average duration of one scenario $D < 3.10^2$ [s] can be determined as satisfactory according the reference used, because it is recommended that the network error should tend to 0, but it does not need to be absolutely 0.0000. This experimental research can be generally compared to an analogues study completed by the author in SIMBRAIN. In this case E = 0.0006, but the simulation model is characterized by a middle level of complexity (size of the matrix: 4x4; number of neurons in the input layer: 16; N = 2, $n \in [4;5]$).

6. CONCLUSION

The described simulation method of optimization of ANNs can be extended in a several directions. For example, neural networks with backpropagation of error can be used in ASR (Automatic speech recognition) to isolate words. Some of their advanced applications are in ITSs (Intelligent Transport Systems) for voice services like voice input in vehicles, voice user interfaces and voice dialing [11]. There are speech recognition systems "based on vector quantization at the acoustic level" [12], that can be described as an optimal encoding method of transforming the components of the input vector in a binary code or "binary words" [13].

In terms to the optimization the next step of the research can be directed to analysing the network behaviour depending on the activation of individual groups of neurons selected by specific criteria. Besides, a comparative analysis of results obtained by using different products for simulating ANNs is a good practice for validation, verification and capability evaluation of software for simulation and visualization especially in case that an empirical study in a physical environment requires costly resources or poses potential risks.

Besides the distributed neural networks are widely used in Internet of Things (IoT) and "Machine-to-Machine (M2M) communications that *"ensure communications without human participation"* [14] *"to decrease the cloud usage and increase the independence of this environment"* [15].

REFERENCES

[1] Yildirim, M., R. Sokullu, Deep-learning-based Decoding for Phase Shift Keying. *International Journal on Information Technologies and Security (IJITS)*, ISSN 1313-8251, Vol. 13, No 1, 2021, pp. 39-51.

[2] Zhang, M., H. You, H., P. Kadam, S. Liu, J. C.-C. Kuo. *PointHop: An Explainable Machine Learning Method for Point Cloud Classification*. IEEE, 2019, Available at: https://arxiv.org/pdf/1907.12766.pdf (visited on 12.07.2021).

[3] Song, W., S. Zou, Y. Tian, et al. Classifying 3D objects in LiDAR point clouds with a back-propagation neural network. *Human-centric Computing and Information Sciences*, Vol. 8, article 29, 2018, https://doi.org/10.1186/s13673-018-0152-7.

[4] SIMBRAIN. *SIMBRAIN 3.0 Documentation.* Available at: http://simbrain.net/Documentation/docs/SimbrainDocs.html (visited on 10.07.2021)

[5] Wilensky, U. *NetLogo*. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL, 1999. Available at: http://ccl.northwestern.edu/netlogo/ (visited on 15.06.2021).

[6] Rand, W., U. Wilensky. *NetLogo Artificial Neural Net - Multilayer model*. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL, 2006. Available at: http://ccl.northwestern.edu/netlogo/models/ArtificialNeuralNet-Multilayer (visited on 12.06.2021).

[7] Whitehouse, P. *PDub's Simbrain Tutorial, IPT-A VIRTUAL APPROACH*, 2018. Available at: http://www.wonko.info/ipt/iis/ai/simbraintut/index.html (visited on 15.06.2021).

[8] Mazur, M. *A Step by Step Backpropagation Example*. 2015, Available at: https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/ (visited on 09.07.2021).

[9] Gurney, K. Chapter Six: Multilayer nets and backpropagation. In book *An Introduction to Neural Networks* (2nd ed.), ISBN 0-203-45151-1, Taylor & Francis e-Library, London and New York, 2004, pp. 102-103.

[10] Nielsen, M. A. Chapter 1: Using neural nets to recognize handwritten digits. In book *Neural Networks and Deep Learning,* Determination Press, San Francisco, 2015, pp. 3-9.

[11] Joshi, S. C., A. N. Cheeran. MATLAB Based Back-Propagation Neural Network for Automatic Speech Recognition. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering,* ISSN (Print): 2320 – 3765, ISSN (Online): 2278-8875; Vol. 3, No 7, 2014, pp. 1-7, https://doi.org/10.15662/ijareeie.2014.0307016.

[12] Hegde, R., A. M. Appaji, M. D. Rao, Vector Quantization (VQ) Based Speech Recognition System on TMS320C6713DSK. *2013 Texas Instruments India Educators' Conference*, 4-6 April 2013, Bangalore, India, ISBN:978-0-7695-5146-3, IEEE, 2014, https://doi.org/10.1109/TIIEC.2013.24.

[13] Gersho, A., S. Gupta, S.-W. Wu. Chapter 6 - Vector Quantization Techniques in Image Compression. In book *Handbook of Visual Communications*, Academic Press, Inc., 1995, pp. 189-222, https://doi.org/10.1016/B978-0-08-091854-9.50011-7.

[14] Romansky, R. A Survey on Digital World Opportunities and Challenges for User's Privacy. *International Journal on Information Technologies & Security*, ISSN 1313-8251, Vol. 9, No 4, 2017, pp. 97–111.

[15] De Coninck, E., T. Verbelen, B. Vankeirsbilck, S. Bohez, P. Simoens, P., Demeester, B Dhoedt. Distributed neural networks for Internet of Things: the Big-Little approach. *Mandler B. et al. (eds) Internet of Things. IoT Infrastructures. IoT360 2015. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 170,* Ghent, Belgium, Springer, Cham, 2016, pp. 484-492, https://doi.org/10.1007/978-3-319-47075-7_52.

Information about the author:

Dr. Yoana A. Ivanova – Teaching assistant in the Department of Telecommunications at NBU; Area of scientific research: Applications of Information Technologies in Security, Communication and Information Systems and Technologies in Security; Professional area: 5.3. "Communication and computer equipment"; Doctoral Program: "Automated Systems for Information processing and Management".

Manuscript received on 18 August 2021