SOCIETY. INTEGRATION. EDUCATION
Proceedings of the International Scientific Conference. Volume V, May 25th-26th, 2018. 260-271

DATA MINING TEACHING POSSIBILITIES USING MATLAB

Pēteris Grabusts

Aleksejs Zorins
Rezekne Academy of Technologies, Latvia

Abstract. *The teaching experience in the study process shows that students are better at perceiving graphical information rather than analytical relationships. Many training courses run on models that were previously available only in mathematics or physics. The use of Matlab package for the implementation of various algorithms in the Information Technology fields could be a possible solution. Often, the analytical solution is much simpler than the visual Matlab model, but for the purposes of perspective training it gives understanding of the usefulness of using such models. In the previous articles the authors had given examples of how Matlab's possibilities could be used for economic research purposes (optimal tax rate searching and modelling market equilibrium price). Students are very interested in modern data mining methods, such as artificial neural networks. In the research part of the study, the modelling capabilities in data mining studies are demonstrated by neural network examples.*
Keywords: *Data Mining, Matlab, modelling, neural networks, perceptron, teaching.*

Introduction

The purpose of studying various models of simulation is to form students' theoretical knowledge and practical skills in applying simulation modelling methods in research of solving specific problems in modelling of real applications - artificial neural networks (ANN). During the process of studying the course, students get acquainted with the means of simulation modelling in processes of functioning the systems, master the methods of simulation, the typical stages of modelling the processes which form the "chain": building of the conceptual model and its formalization - algorithmization of the model and its computer realization - simulation experiment and interpretation of the results of simulation; master the practical skills of implementing modelling algorithms for studying the characteristics of complex systems of information technologies.

Implementing such capabilities in a universal programming language is a very difficult task. Currently, there are quite a few software products that allow modelling processes. At the same time, there is now a product that can solve these problems very effectively - the MATLAB package, which contains tools for implementing artificial neural networks and fuzzy logic. It should be noted that

http://dx.doi.org/10.17770/sie2018vol1.3091

knowing artificial neural networks and fuzzy logic gives students the skills to do the research and stimulates them for scientific work (Kay, 1984; Karel & Tomas, 2015; Karris, 2006; Smith, 2013; Xue & Chen, 2013).

The aim of the article is to show Matlab suitability for the purpose of visualizing simulation models of various Data Mining (DM) disciplines. To reach the aim, the following research tasks have been set: identification of Matlab possibilities for neural network realization; demonstrate visualization models on the basis of examples. Common research methods are used in this research: descriptive research method, statistical method, mathematical modelling, neural network algorithms.

Within the framework of the work, a practical example is presented that will enable students to understand and start learning the modern analysis of large data (Big Data) with the help of neural network.

Neural Networks Concept

Artificial Neural Networks (ANN) is a generalized term for a certain class of algorithms that has a very important property - the ability to learn from examples, obtaining hidden regularities from data, and the data may be incomplete, distorted, and even contradictory (Fausett, 1994; Russel & Norvig, 1995; Bishop, 1995). If there is some connection between the input and output data (even if the traditional methods do not show it), the neural network is able to tune in to it with a certain accuracy. In addition, many neural networks make it possible to evaluate the significance of individual input data, to reduce the amount of data without the loss of relevant data, and make it possible to identify a near-critical situation. In many cases, neural networks allow to find regularities that are virtually impossible to detect by analyzing data manually.

The core element of artificial neural networks is neuron, that unlike a biological neuron, is known as formal, technical or mathematical neuron. The formal neuron (see Figure 1) has the following structure:

1) the neuron has n input channels x_i $(i = 1,\dots,n)$ and one output channel;
2) for each input channel x_i, a number w_i is assigned, called the weight of this channel. The weighting vector is $w = \{w_1, w_2, \dots, w_n\}$;
3) weighted signals of all inputs are summed up:
$$a = \sum_i w_i x_i$$
4) summarized signal is modified by the activation function and fed to the output channel.

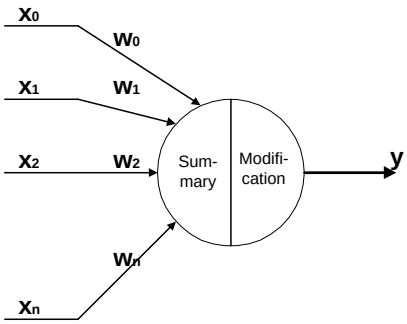

Figure 1. **Formal neuron**

In general, the neural network has the following structure:

1) the network consists of many interconnected neurons;
2) neurons are arranged in levels or layers;
3) the first layer is the input buffer that inputs the signal vector to the network; the last layer is the output buffer that determines the network response to the input signal vector; between the first and last layer there may be one or more hidden layers (interlayers) that perform the aggregator functions of network operation;
4) the interlayer connections may be complete, accidental or by element groups;
5) a network with layers and signal processing functions is called perceptron.

The network training, i.e., the weight adaptation process, building neuronal responses from the input signal vector to the output buffer is essential for the functioning of the neural network. The output buffer signals should match the advisable result. The functioning of the network was improved by applying weight ratios (Rashid, 2016).

Single-layer perceptron - able to train to classify input vectors that are linearly separable. In the training process the perceptron training algorithm is used (Fausett, 1994), which is very suitable for students in acquiring the basics of functioning of neural networks. Because of the limited volume of the article, it is not possible to publish the full Perceptron learning algorithm and students should understand it individually.

Matlab Opportunities for Researching Neural Networks

The Matlab package has several development versions and it has become one of the most powerful universal mathematical systems. It contains its own built-in programming language, powerful graphical visualization tools. Matlab can be

used practically in every field of science and technology and is widely used in mathematical modelling (Esfandiari, 2013; Kiusalaas, 2016).

The popularity of Matlab is largely due to the fact that, in addition to the powerful programming environment, a number of expansion packets have been developed - specific tools (Toolboxes) for various practical applications. The latest Matlab versions feature about 50 different additional tools.

Neural Networks add-on NNTool gives the possibility to create neuron networks, training, modelling, export and import of neural networks and data without interacting with the Matlab team. It can be done by using interface (GUI) capabilities of graphical user. Such types of means are effective working with a package only at an early stage, as there are several limitations in choosing more complex network configurations. So NNTool allows to process only single layer and dual layer networks. All of this makes the Matlab extension pack suitable for initial information acquisition of the neural network and allows to experiment with different types of networks in order to get the best training that allows you to conclude on the quality of network training.

Research part

The configuration of a computer plays an important role in its application- it can be an office computer for working with a text editor, an electronic table, and an Internet connection. Home computer would need a larger hard drive, a good video card and a good monitor. The gaming computer should have a powerful video card, a large hard drive and a good monitor. Of course, such a breakdown is very provisory, over time the criteria have changed several times.

In order to ease the choice for the client, it is proposed to create a training system based on the neural network technology, which, with existing computer configuration data, would be able to identify the type of chosen computer. This would facilitate and speed up the selection of computer configurations. Such a system should be self-trained and based on already introduced examples of computer configurations. In order to complete this task, it is necessary to select an appropriate data structure to be used in future for the training and further testing of the neural network. When looking at different computer configurations (although they are outdated - it does not change the substance of the matter), a simplified data model can be created in the following way (see Table 1).

Computer analysis was performed and, as a result, data was received. For the neural network to use these data in the network entrance for the training process, it is necessary to encode the input data example, or vector. The following criterion has been chosen: if the desired computer component matches the computer configuration parameter, let us encode this parameter with 1, if it does not match -

with 0. Thus, the example computer configuration will be encoded as follows: [1 0 1 0 1 0 1 0 1 0 0 0 1 0].

Table 1 **Possible Computer Configuration Indicators**

N.	Computer component	Possible values
1.	Processor Type	Celeron
		Pentium
2.	Processor Tact Frequency	1.8 to 2.4 Ghz
		higher than 2.4 Ghz
3.	The amount of RAM	2048 to 4096 Mb
		greater than 4 Gb
4.	Hard dick capacity	250 to 500 Gb
		greater than 500 Gb
5.	Video card type	integrated
		256 to 1024 Mb memory
		greater than 1024 Mb
6.	Display size	19” display
		21” display
		greater

Table 2 **Hypothesis data distribution**

N.	Computer Configuration	Computer Type
1.	1 0 1 0 1 0 1 0 1 0 0 1 0 0	Office computer (0)
2.	1 0 1 0 1 0 0 1 1 0 0 0 1 0	
3.	1 0 1 0 1 0 0 1 1 0 0 0 1 0	
4.	1 0 1 0 1 0 0 1 0 1 0 0 1 0	
5.	1 0 1 0 0 1 0 1 0 1 0 0 1 0	
6.	1 0 1 0 1 0 0 1 0 1 0 0 1 0	
7.	1 0 1 0 1 0 0 1 0 1 0 0 1 0	
8.	1 0 1 0 0 1 0 1 0 1 0 0 1 0	
9.	0 1 1 0 0 1 0 1 0 1 0 0 1 0	Gaming computer (1)
10.	0 1 1 0 1 0 0 1 0 1 0 0 1 0	
11.	0 1 1 0 1 0 0 1 0 0 1 0 0 1	
12.	0 1 0 1 1 0 0 1 0 0 1 0 1 0	
13.	0 1 0 1 0 1 0 1 0 0 1 0 0 1	
14.	0 1 0 1 0 1 0 1 0 0 1 0 0 1	

In the planning stage of the experiment, the hypothesis was put forward - computers with numbers from 1 to 8 with the Celeron processor could relate to the same type, which we would conditionally call the *Office Computer* class. Computers with numbers 9 to 14 will be assigned to the *Gaming Computer* class. In this way, we have two types of data or classes, and, as a result of experiment, with a help of neuron networks we should be sure that the neural network is

capable to separate the two-class vectors in the training process. Table 2 shows the appropriate computer breakdown by classes.

NNTool is launched in the Matlab environment (see Figure 2).

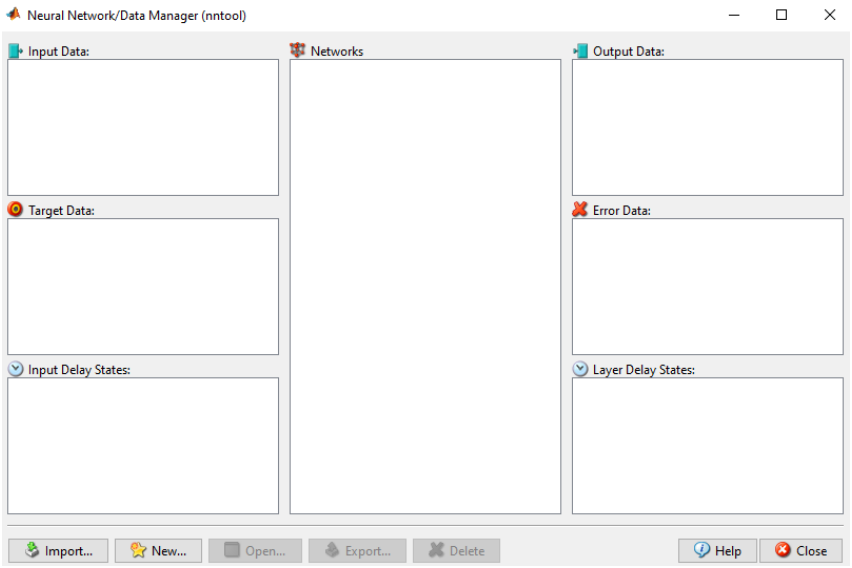

Figure 2. **Matlab NNTools environment window**

Select *NEW/DATA* and enter data and targets for training. Conditionally we name the data from Figure 2 as *data_in* and define them as *Inputs* (see Figure 3) in accordance with Matlab's syntax. At the end we press the button *Create*.

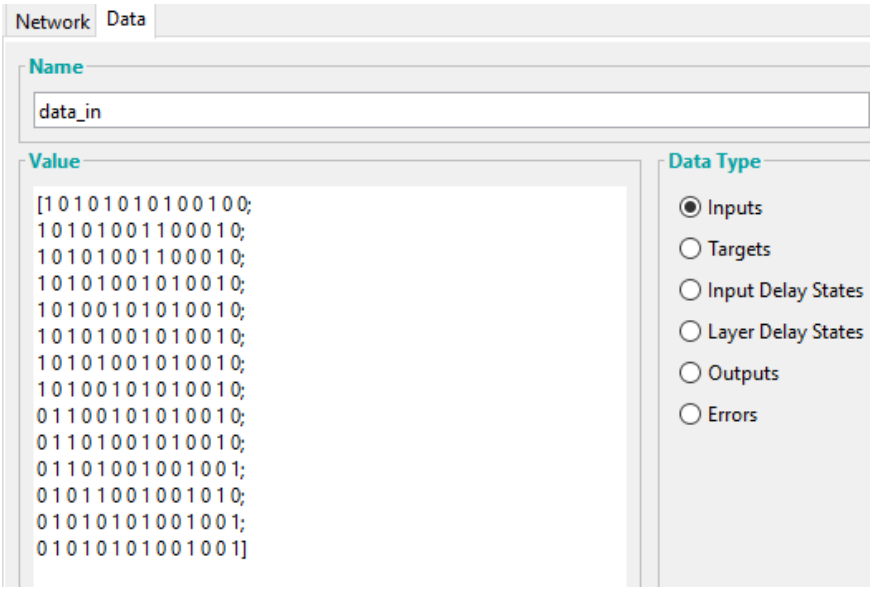

Figure 3. **Experiment input data**

Similarly, we name target data from Figure 2 as *data_target* and define them as *Targets* (see Figure 4).

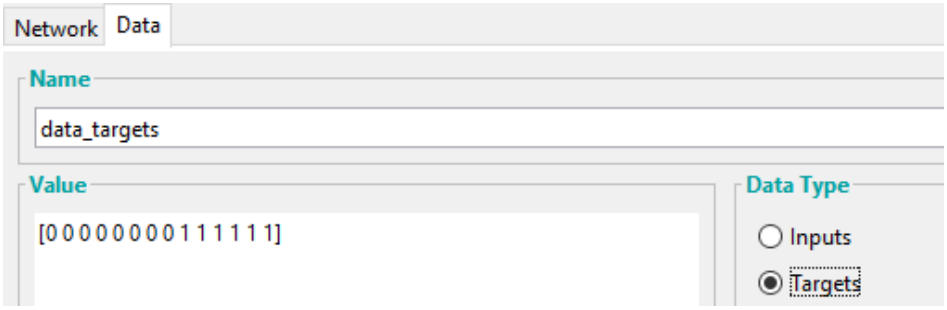

Figure 4. **Experiment target data values**

After clicking the button Create, we return to the neural network design mode *New\Network*. We will call the network *Perceptron*.

Network type select *Perceptron.*

Input data select *data_in.*

Target data select *data_targets* and validate by clicking the button *Create* (see Figure 5).

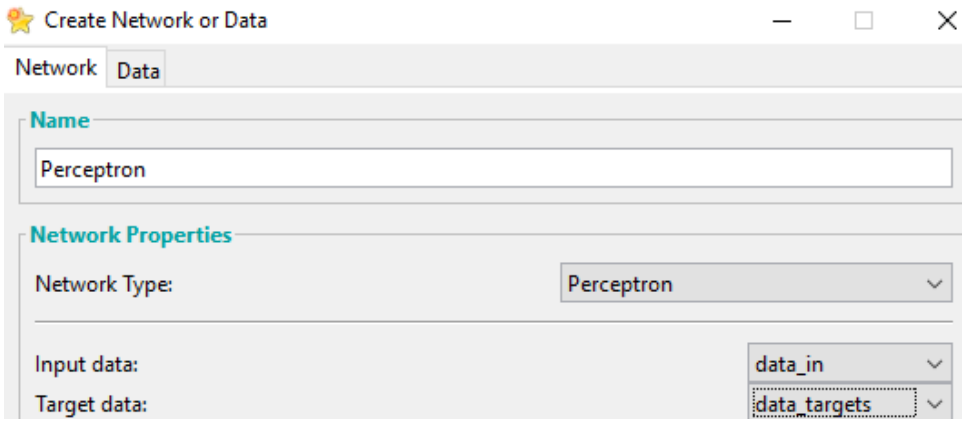

Figure 5. **Network data configuration window**

As a result, a neuron network entitled *Perceptron* was created (see Figure 6).

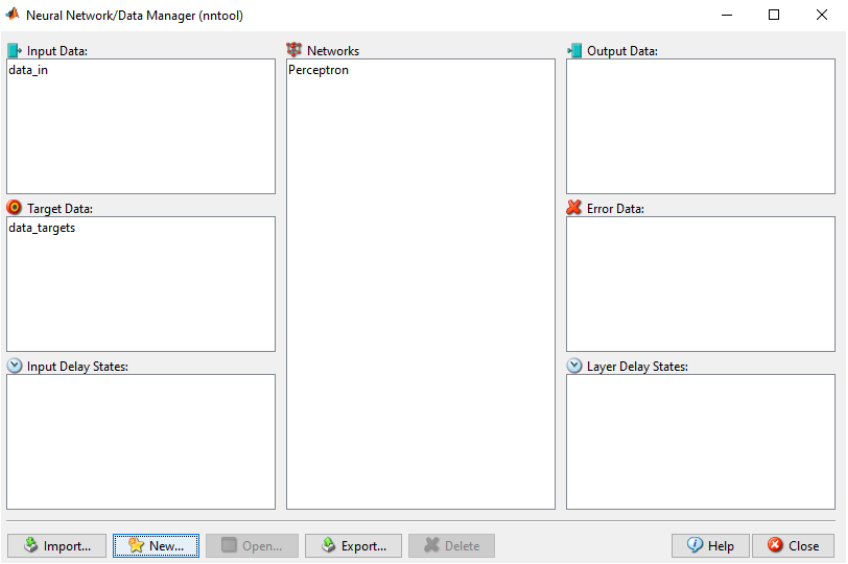

Figure 6. **Created neural network**

By choosing *Perceptron* neural network, its structure is displayed - 14 input vectors, 1 layer and 1 output (see Figure 7).

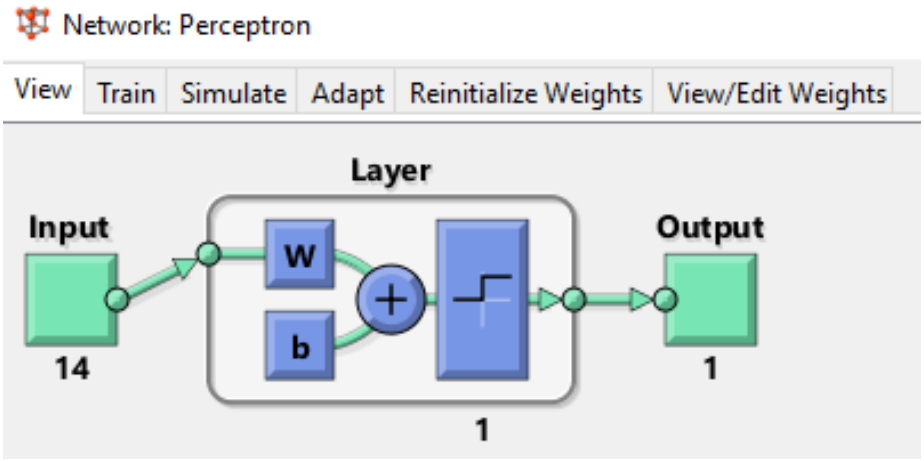

Figure 7. **Neural network structure**

It's time to train the neural network - in the distribution *Train* we set all the initial settings and start network training *Train network* (see Figure 8).

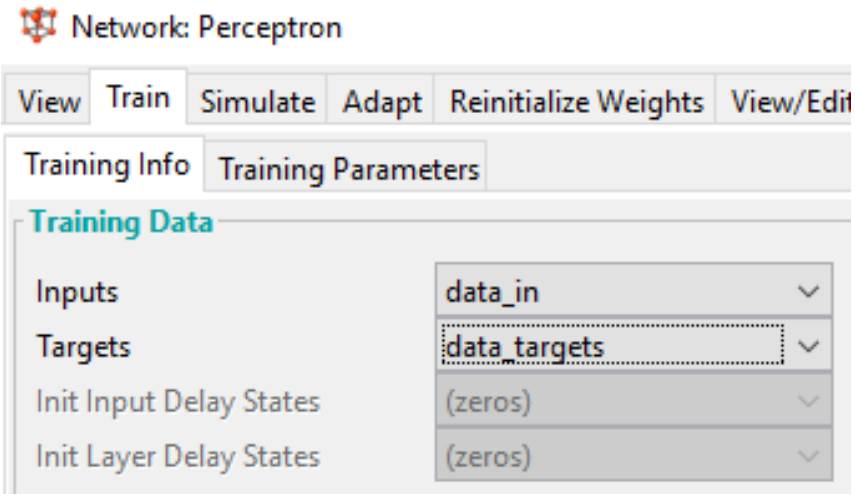

Figure 8. **Network training**

By default, 1000 algorithm iterations are accomplished, the number and training parameters of which can be changed. The following window appears as a result of the training (see Figure 9).

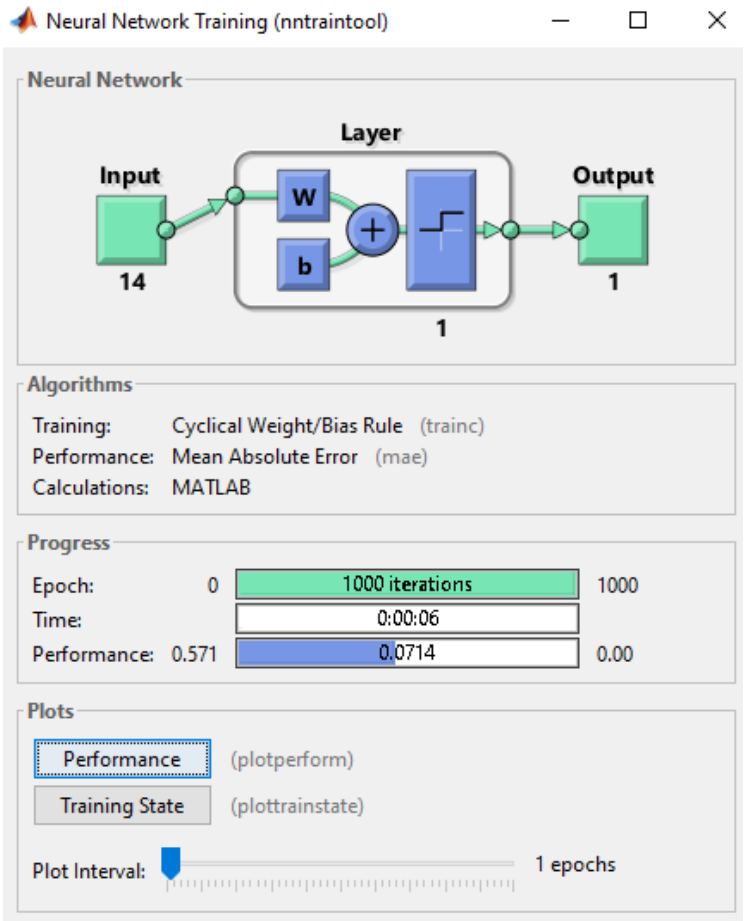

Figure 9. **Neural network training window**

It can be concluded that the neural network has been trained in the 57 iteration of the training algorithm (performance 0,571), which is also shown in the graph *Plots/Performance* (see Figure 10).

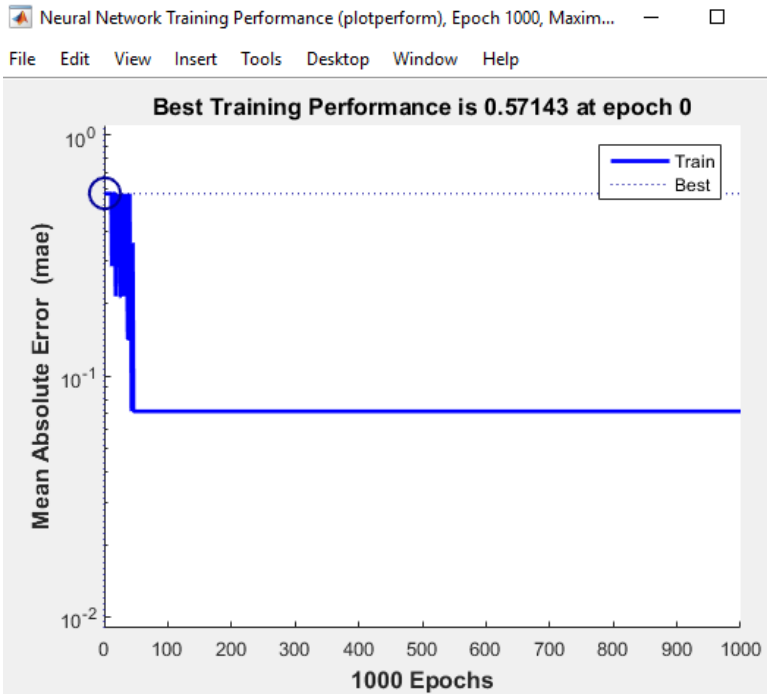

Figure 10. **Best training performance**

It can be concluded that a neural network is capable to recognize the given computer configuration successfully by referring it to the office computer type or to the gaming computer type.

The following steps should be taken to improve network training:

- complement the number of training examples (e.g., by collecting statistics on computers sold over a given period of time);
- introduce new types of computers if needed (e.g., *Server computer* type);
- change the training data structure by adding new configuration parameters (e.g., Cache memory volume), or by detailed description of the component range;
- change the network training parameters.

In this way, an optimal neural network configuration could be obtained, which would help to identify new computer configurations.

Conclusions

This paper justifies the usefulness of the introduction of simulation models in the initial training process, when simulation models can also be introduced for the acquisition of analytical relationships for modelling purposes. It enables to perceive not only the possibilities of using the formulas, but also to visualize various relationships in graphic form.

A single class of artificial neural networks - Perceptron training - was studied in the work. An encoding of information is considered in the form in which it can be used in artificial neural networks. The Perceptron modelling algorithm is described and a numerical example of an algorithm's operation is realized.

In the field of data analysis, the Matlab package NNTool was investigated with the aim of finding out the possibilities of processing the neural networks assigned to solve practical problems. Packages of this type make it easier for the user to work in the decision-making field, increasing the efficiency and quality of decisions. An example of applying a suitable system for the computer configuration task is seen. For demonstration purposes, the systems were trained with a set of 14 examples; each example contained a vector with encoded computer configuration with 14 parameters. It has been experimentally stated that neuron networks are able to train according to the given training algorithm and successfully perform testing, which results in the test data example being assigned to the corresponding computer configuration type or class.

It is concluded that neuron network class tasks can be successfully applied at the beginning of the data analysis. Such systems can show good results in the selection and analysis of information.

Thus, it can be concluded that Matlab neural network modelling tool is a very suitable tool not only for calculations in engineering, but can also serve as a visualization tool of simulation models in various artificial intelligence applications.

References

Bishop, C. M. (1995). Neural networks for pattern recognition. Oxford: Oxford University Press.

Esfandiari, R. S. (2013). *Numerical Methods for Engineers and Scientists Using MATLAB*. Chapman & Hall/CRC.

Fausett, L. (1994). *Fundamentals of Neural Networks: Architectures, Algorithms and Applications*. New York: Prentice Hall International Inc.

Kay, C. (1984). *Mathematics for Computer Programmers*. New Jersey: Prentice Hall.

Karel, P., & Tomas, Z. (2015). Multimedia Teaching Aid for Students of Basics of Control Theory in Matlab and Simulink. *Procedia Engineering, Volume 100*, 150–158.

Karris, S. T. (2006). *Introduction to Simulink ® with Engineering Applications*. Orchard Publications.

Kiusalaas, J. (2016). *Numerical Methods in Engineering with MATLAB, 3e*. Cambridge University Press.

Rashid, T. (2016). *Make Your Own Neural Network.* CreateSpace Independent Publishing Platform.

Russel, S., & Norvig, P. (1995). *Artificial Intelligence: A Modern Approach*. Englewood Cliffs, NJ: Prentice Hall.

Smith, D. (2013). *Engineering Computation with MATLAB, 3e*. Pearson Education Inc.

Xue, D., & Chen, Y. (2013). *System Simulation Techniques with MATLAB and Simulink*. John Wiley & Sons, Inc.