Kazimierz Witaszek, Krzysztof Garbala, Mirosław Witaszek, Marcin Rychter

Optimization of neural networks structure selection in modelling spheroidal graphite cast iron for automotive camshafts

JEL: R41 DOI: 10.24136/atest.2019.255
Data zgłoszenia: 28.01.2020 Data akceptacji: 10.02.2020

The present article discusses the process of optimizing the structure of artificial neural networks applied in modelling the wear of spheroidal graphite cast iron (SG cast iron). The networks were trained using the RPROP gradient method with the application of the SNNS package supported by original self-developed software, which enabled automatic creation, training and testing of networks with different sizes of hidden layers. Based on the results of an analysis of learning process and testing a package of 625 networks, the network was selected which – when modelling the process of spheroidal cast iron wear – generates the slightest errors during testing.

Słowa kluczowe: Artificial neural networks, structure optimization, wear, spheroidal cast iron, Stuttgart Neural Network Simulator, Resilient backPROPagation.

Wstęp

With the development of technology, machines have been supporting people to an increasing extent. During operation, machine elements are subjected to a variety of stress, so wear and tear on machine elements and equipment can often be observed as a result of the common phenomenon of friction. It is therefore reasonable to strive to reduce the wear of mechanical components for both economic and ecological reasons. For this purpose, it is necessary to get well acquainted with the wear mechanisms and their dependence on the exploitation factors and materials used. Many researchers have published papers on wear, among whom there is a group using advanced information technology, including artificial neural networks used for modelling or predicting the wear of materials and machine parts [1, 2, 3].

The dynamic development of computer hardware and software observed in the last quarter of a century encourages researchers to seek solutions to scientific issues using IT tools. Recently, a slowdown can be observed in the hardware development of microprocessors measured by the number of embedded transistors, hence researchers are beginning to attach increasing hopes to the development of software algorithms [4]. Advanced IT systems play a very important role both in the world of science and in most branches of our everyday life, where they have achieved a very high level of integration [5]. One of the many interesting categories of IT tools that researchers around the world readily use are the artificial neural networks. They are widely used in many fields of science, especially where research problems are complex and the available knowledge of their relationships is limited [6]. Since numerous wear mechanisms are present in technology and its intensity varies depending on the operating conditions and material factors, it seems reasonable to use artificial neural networks in the analysis of these issues. Elements of transport machines, including parts of motor vehicles often wear out, so they are a significant group of issues studied [7]. Many vehicle parts are made with the use of foundry techniques. For those that require higher strength parameters, ductile irons are used, among others. The study attempts to model the wear of spheroidal graphite cast iron used for camshafts, with the use of artificial neural networks. One of the important issues when using this IT tool is the selection of the network topology and the size of hidden layers. Too few links do not allow for proper modeling of the issue, and too extensive topology negatively affects the ability to generalize knowledge and, as a consequence, the results of the network operation [6]. Therefore, optimization of the neural network topology is a very important issue in virtually every application [8, 9]. This article presents a tool that allows selection of the optimal size of hidden layers of the constructed networks.

1. Research data

To properly carry out the network training/learning process, two separate sets of data were prepared, based on the developed results of wear tests. The first one is the so-called training set, which was used to teach neural networks, while using the second one (validation set) the correctness of the network operation was verified. It did not contain data occurring in the training set [8]. The training and validation data were prepared on the basis of laboratory results of the wear of spheroidal graphite cast iron test EN-GJS-900-2 used among others for camshafts of automotive combustion engines. The wear tests were carried out on a pin-on-disk tribometer type T-01M which was manufactured at the Instytut Technologii Eksploatacji in Radom. Before starting the tests, the samples were heat treated to obtain their hardness within the range of 37 - 53 HRC. The tests were carried out with a friction node load of 49.05 N in the sliding speed range from 0.5 to 2 m/s. The tests were carried out in two variants - for oil-lubricated contact, a friction path of 70,000 m was selected, while for dry sliding conditions - 2,000 m respectively.

The wear rate was calculated using the RegKro program developed for this purpose, which determined it by linearly regressing small fragments of the results of wear tests as a function of the friction path. The results of laboratory wear tests processed by the RegKro program formed the basis for creating a database from which, after supplementing with information on the structure and properties of samples, a training and validation set was generated for experiments with artificial neural networks.

Neural networks were taught based on eight input parameters taking into account operational and material features [10]. Five of them described the conditions of cooperation of the tested samples. They were, subsequently: the presence of oil, pressure, coefficient of friction, sliding speed, contact temperature. The other three input values characterized the sample material. These were: microhardness of the metallic matrix, percentage of graphite area, and the number of graphite precipitates per 1 mm^2. Linear wear rate of the sample material was assigned to the network output. The structure of the data selected for building the training and validation sets is shown in Table 1.

Tab. 1. Structure of data selected for building the training and validation sets

Value description	Type	Value range	Unit
Presence of oil	Input	0, 1	-
Pressure		8,4 … 1125,3	MPa
Coefficient of friction		0,043 … 0,924	-
Sliding speed		0,3 … 2,0	m/s
Contact temperature		29 … 100	ºC
Microhardness of the metallic matrix		390 … 760	HV
Percentage of graphite area		6.77 … 11.66	%
Number of graphite precipitates per 1 mm^2		166 … 285	-
Wear rate	Output	$3{,}62 \cdot 10^{-11}$ … $1{,}85 \cdot 10^{-6}$	-

The values of each input parameter have been standardized to the range of 0…1 [10] according to formula 1:

$$np_i(\mathrm{k}) = \frac{p_i(\mathrm{k}) - \min\ p_i}{\max\ p_i - \min\ p_i} \tag{1}$$

where:
- np_i(k) – standardized value of signal at the i entry of the network for k-th model,
- p_i(k) – value of signal at the i entry of the network for k-th model,
- min p_i(k) – the lowest value of signal at the i entry for the whole set of models,
- max p_i(k) – the highest value of signal at the i entry of the network for the whole set of models,

Due to the observed significant differences in wear rate between the test data in the presence of lubricant compared to dry sliding conditions, it was decided that its value should be log-transformed before standardization.

The finished training set consisted of 746 data records, while the validation set contained 123 of them. After adding headers required by the package simulating the work of artificial neural networks, the data was saved to text files (*.pat).

2. Research data

The Stuttgart Neural Network Simulator (SNNS) software was chosen for creating networks, training and validating them. The software was created at the University of Stuttgart and then further developed at the University of Tubingen. Although SNNS is characterized by an obsolete user interface operating in the X-window (X11) environment, it has been equipped with a rich set of training functions. Thanks to this, it is characterized by considerable universality and, importantly, high speed of operation. The software was made available free of charge under the LGPL Version 2 license. The authors also provided for the possibility of using SNNS software in batch mode bypassing the graphical user interface. Such access to the simulator's core allows its integration with the user's own software [11, 12]. For the purposes of this work, an application operating in a script system has been developed that allows the automatic creation, training and validating the neural nets of various sizes. The application called mkNHH1net was written using the Turbo Pascal programming language and compiled to executable form using the Free Pascal Compiler 3.0.4 software. The operation of the mkNHH1net program is shown schematically in Figure 1.

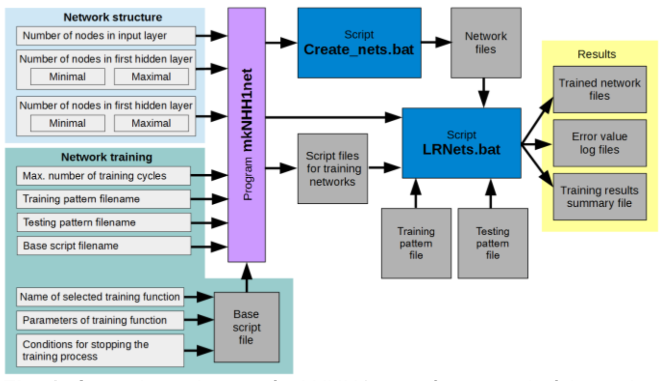
Network structure
Number of nodes in input layer
Number of nodes in first hidden layer
Minimal
Maximal
Number of nodes in first hidden layer
Minimal
Maximal
Network training
Max. number of training cycles
Training pattern filename
Testing pattern filename
Base script filename
Name of selected training function
Parameters of training function
Conditions for stopping the training process
Base script file
Program mkNHH1net
Script Create_nets.bat
Network files
Script files for training networks
Script LRNets.bat
Training pattern file
Testing pattern file
Results
Trained network files
Error value log files
Training results summary file

Fig. 1. Operation scheme of mkNHH1net software and of the scripts generated by it

When starting mkNHH1net, it was necessary to specify the parameters necessary for its operation in the command line. The first group of parameters determined the structure of the network by providing the size of the input layer as well as the minimum and maximum numbers of neurons in both hidden layers. The second group were the parameters defining the network learning process, such as the number of training cycles and the location of the training and validation set files. It was also necessary to indicate the location of the base script, which, after providing headers specific for successively generated networks, was the basis for automatic creation of sets of scripts using the SNNS command line interface (Batchman and ff_bignet) and Windows (cmd). The first script (Create_nets.bat) using the ff-bignet program included in the SNNS package, created successively larger and larger networks within previously specified size limits and saved them to files. According to the base script, the second script (LRNets.bat) trained and validated the previously created networks. The the base script included, among others network learning function declarations. Freshly created networks were initialized by randomly selecting weights in the range from -1 to 1. The networks prepared in this way were subjected to the learning process using the data contained in the training examples.

From among 33 learning functions available under the SNNS package, the heuristic algorithm of Riedmiller and Braun RPROP (Resilient backPROPagation) was selected [13]. The application of the RPROP method was aimed at accelerating the learning process compared to the basic gradient algorithms [14] (e.g. backpropagation). Under the RPROP method, changes in neuron connection weights are described by the relationship (2):

$$\Delta \mathrm{W}_{\mathrm{ij}}(\mathrm{k}) = -\eta_{\mathrm{ij}}^{\mathrm{k}} \mathrm{sgn}\left(\frac{\partial \mathrm{E}[\mathbf{W}(\mathrm{k})]}{\partial \mathrm{W}_{\mathrm{ij}}(\mathrm{k})} \right) \tag{2}$$

where:
- η_{ij}^{k} - learning factor in k-th step for weights of the links of neurons i and j,
- sgn function stands for lack of argument,
- W_{ij}(k) - weights of the links of neurons i and j,
- E[W(k)] – function of target (objective) dependent on W(k) weight vector.

The RPROP algorithm modifies the values of the learning factor η, values in such a way that if the gradient sign was the same in the next two steps η increases, and it decreases when the sign was different.

When calling the RPROP learning function, three parameter values must be entered. They were selected in accordance with the suggestions from the SNNS software documentation [15] and are shown in Table 2.

Tab. 2. Selection of the training method parameters

Parameter	Recommended values [15]	Selected value
η_0 - initial update-value	0 … 0.2	0,1
η_{MAX} - limit for the maximum step size	50	50
η_{MIN} - minimum step size	10-4 … 10-6	10-5

During learning, the weight was modified in the network in the topological order according to the setUpdateFunc() function declared in the base script. The base script enabled the selection of various scenarios to complete the learning process. These were: execution of a given number of training cycles, achievement of the adopted minimum error value, no further learning progress in the defined number of successive training cycles.

The network learning process progress was assessed by observation of changes in the MSE (Mean Square Error) value [16, 17] for the validation set in a single-exit network, calculated by SNNS according to the formula 3 [15]:

$$\mathrm{MSE} = \frac{SSE}{n-q} \tag{3}$$

where:
- n – number of observations (data set amount),
- q – number of estimated parameters (network weights),
- SSE – value of the summary square error determined from the formula 4 [15]:

$$\mathrm{SSE} = \sum_{k=1}^{q}\left(y_i(\mathrm{k}) - d_i(\mathrm{k})\right)^2 \tag{4}$$

where:
- y_i(k) – calculated value of signal at exit i of the network for k-th validating model,
- d_i(k) – required value of signal at exit i of the network for k-th validating model,
- q – number of examples in the validation set.

As a result of the work of programs and scripts, learned networks (*.net) were saved in the files. If the appropriate option was selected in the base script code, it was also possible to save partially learned networks successively after a defined number of learning cycles. Then, in the names of network files, a segment was used consisting of _U characters and a four-digit number corresponding to the number of learning cycles (*_U????.Net). If necessary, it was also possible to enable the option of saving files containing records from training and validation sets in connection with network responses (*.res files). During the network learning process, the accumulative and mean squared error values for the training and validation sets were recorded. Information about the learning process was recorded in text files (*.log). The LRNets.bat script also included instructions that enabled the accurate measurements of the learning time of each network. Collective final results of learning outcomes for all the networks analysed in a given network scenario were also saved to a text file (*.fin). The type of text files was chosen due to the ease of their implementation in the scripts of the SNNS package and the operating system console. The internal organization of the files has been optimized to facilitate their further analysis with external tools in the form of a spreadsheet or a Gnuplot package which was chosen due to the possibility of analyzing and creating 3D charts even with a very large amount of data being processed.

3. Result analysis

Using the mkNHH1net program, a set of 625 networks was created with two hidden layers of all possible combinations of sizes from 1 ... 25 neurons in each layer. Thanks to the generated scripts, these networks were subjected to an automatic training and validating process. Modern computers have considerable computing power, thanks to which properly optimized software can achieve a significant speed of operation. The SNNS package, especially when launched in batch processing mode, without the necessity to emulate the X-Window graphic environment, conducted calculations for the trained networks at a satisfactory speed. Performing 100 learning cycles for the largest of the examined networks, having 25 neurons in two hidden layers, took less than one second. Therefore, with such efficient tools, it is possible to optimize the selection of the network structure with the method known as "brute force", i.e. checking all possible combinations. Initially, it was decided to train all networks through a predetermined number of cycles, and then compare their test results. Due to the nature of the experiment consisting in optimizing the size of two layers, it was decided to present the results in the form of maps on which the colors correspond to the values of the obtained test error. To better illustrate the dependence of network test results on their structure and to reduce the noise of results, a smoothing algorithm embedded in the Gnuplot package was used, based on the exponential function. In addition, the dgrid3d function was used to smooth the edges, which allows oversampling of any data and arranging them in the form of a map. Such mathematical operations allowed better visibility of extremes in the areas of test error graphs. Preliminary analysis of the course of changes in the network error value during learning enabled the determination of the number of training cycles. During the learning process, the values decrease of errors generated by the network for both the training and testing set. However, after exceeding a certain number of training cycles, the test error stops decreasing and may even increase. This is usually the premise for ending the network learning process. The networks generated in the discussed experiment showed this behavior. Examples of results of the learning process of the obtained network set are shown in Figure 2.

It shows that the structure and the number of training cycles have a noticeable impact on the results of training of artificial neural networks. Due to the properties of the RPROP learning algorithm, to achieve satisfactory results it is enough to calculate with the simulator fewer than 100 training cycles. From the input layer, the data goes to the first hidden layer. In the issue under consideration, the number of neurons in this layer cannot be too low. All the tested networks with the first hidden layer containing less than 6 - 7 neurons show large testing errors. The best results are achieved by networks with more than 10 neurons (14 ... 17) in this layer. The sizes of the second hidden layer also have a significant impact on network training outcomes. The most efficient were the networks with a not very developed second hidden layer, with a number of neurons higher than 5 - 6. Increasing this number above 10 - 11 leads to a rapid deterioration of the network test results for virtually all cases studied. It should be emphasized that excessive expansion of the second hidden layer leads to worse results than with a small number of neurons in this layer. The best parameters for training a given number of cycles from among the tested networks were achieved by those with a structure close to 8-14-8-1.

Training networks of different sizes usually requires individual selection of the right number of training cycles. When the weights of the neuron links are optimally selected during this process, the further training is slow, which results in minor changes in error values in subsequent cycles. This fact was used in the structure of the base script, thanks to which it was possible to finish the training process after detecting in ten consecutive cycles test error changes that were smaller than the set relative value. With this algorithm, each network was trained for exactly the number of cycles as needed to achieve the best learning process result for it. The results of such an experiment are shown in Figure 3.

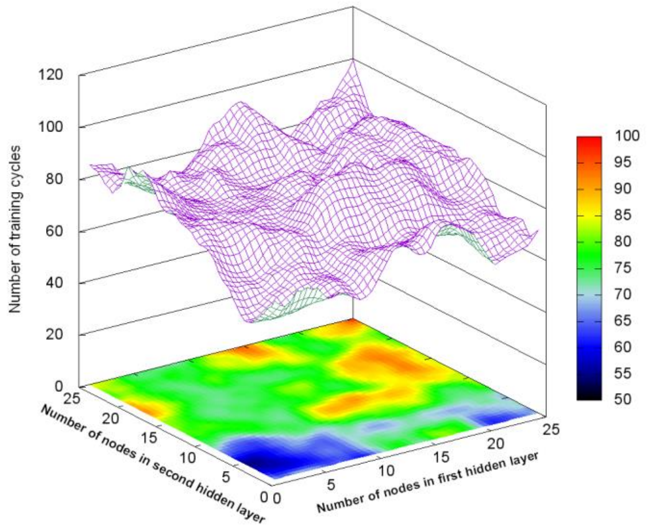

Fig 3. Number of training cycles until the complete training of the networks of various sizes

For better readability, in addition to the color map, a series of data in the form of a three-dimensional surface was marked on it. Figure 3 shows that networks with a poor first and second hidden layer quickly stop learning because they are unable to model the issue under consideration. The presented graph shows several local maxima, one of which corresponds to the network structures that achieve the smallest errors in the learning process.

At the next stage of the analysis, the results of network testing of the trained network packet 625 were made, as presented in Figure 4a. Placing the contour lines on the graph (Fig. 4b) made the sought area of network structures with the lowest testing error more clearly visible. It corresponds to networks with 17-18 neurons in the first hidden layer and nine in the second one. Due to the contour line extension along the x axis, it can be assumed that the selection of the size of the first hidden layer leaves more freedom. However, the change in the number of neurons in the second hidden layer more strongly affects the results of network testing.

In addition to the error evaluation for the test set, an analysis of the final learning error of the network set 625 was made. Its results are shown in Figure 5. The relationships obtained are similar to those previously discussed. It can be seen that due to the learning results themselves, the optimal size of the first hidden layer is 19 neurons. However, such a network has a slightly worse ability to generalize knowledge acquired in the learning process, as indicated by the results of the error obtained for the examples contained in the validation set. For the problem of wear of spheroidal graphite cast iron modeling analyzed in the paper, the optimal topology of the artificial neural network is 8-18-9-1. It is characterized by the lowest testing error (Fig. 4b).

Conclusions

Based on the conducted research, the following conclusions have been formulated:

- The use of a well-documented package simulating the work of artificial neural networks, which has an open architecture, allows to design individual applications supporting the optimization of the network topology.
- Performance of modern computers allows the convenient use of tools optimizing the topology of artificial neural networks modeling the wear of spheroidal graphite cast iron based on the creation, learning and analysis of the operation of ever larger networks.
- It is advisable to use an algorithm that individually selects the number of training cycles for networks with different topologies. Choosing this variant of the base script allowed to obtain networks generating the smallest testing errors.
- Application of the proprietary tool allowed to find the optimal topology of the neural network for modeling the studied wear of spheroidal graphite cast iron. The first hidden layer should consist of 18 neurons and the second one - of nine.
- Considering the results of network testing, a narrower tolerance field occurs in the selection of the sizes of the second hidden layer. Its excessive expansion leads to a significant increase of error in wear modeling.

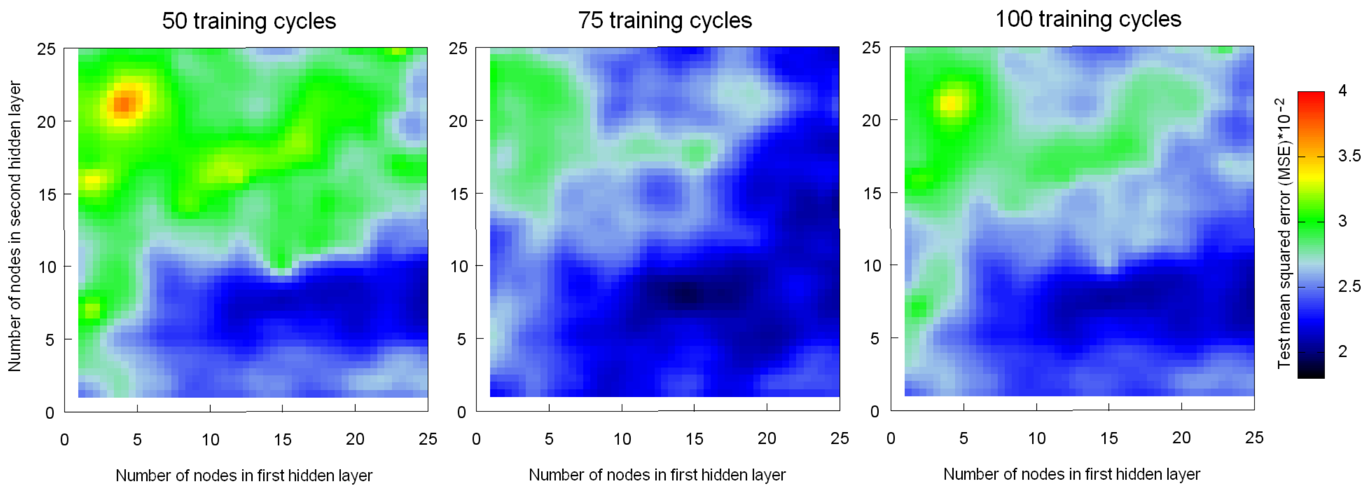

Fig. 2. Testing error of the analyzed networks after passing the determined number of training cycles

References:

1. Xu, J. & Yamada, K. & Seikiya, K. & Tanaka, R. & Yamane, Y.: Effect of different features to drill-wear prediction with back propagation neural network. Precision Engineering.2014. Vol. 38, Iss. 4. P. 791-798.
2. Kadua, R.S. & Awarib, G.K. & Sakhalec, C.N. & Modakd, J.P.: Formulation of Mathematical Model for the Investigation of Tool Wears in Boring Machining Operation on Cast Iron Using Carbide and CBN Tools. Procedia Materials Science. 2014. Vol. 6. P. 1710-1724.
3. Veeresh Kumara, G.B. & Pramoda, R. & Raob, C.S.P. & Shivakumar Goudac, P.S.: Artificial Neural Network Prediction On Wear Of Al6061 Alloy Metal Matrix Composites Reinforced With -Al2O3. Materials Today: Proceedings. 2018. Vol. 5, Iss. 5. Part 2. P. 11268–11276.
4. Cios, K.J. & Mamitsuka, H. & Nagashima, T. & Tadeusiewicz, R.: Computational intelligence in solving bioinformatics problems. Artificial Intelligence in Medicine. 2005. Vol. 35, Iss. 1-2. P. 1-8.
5. Tadeusiewicz, R. & Ogiela, L. & Ogiela, M.R.: The automatic understanding approach to systems analysis and design. International Journal of Information Management. 2008. Vol. 28, Iss. 1. P. 38-48.
6. Tadeusiewicz, R.: Neural networks in mining sciences-general overview and some representative examples. Archives of Mining Sciences. 2015. Vol. 60, Iss. 4. P. 971-984.
7. Manieniyan, V. & Vinodhini, G. & Senthilkumar, R. & Sivaprakasam, S.: Wear element analysis using neural networks of a DI diesel engine using biodiesel with exhaust gas recirculation. Energy. 2016. Vol. 114. P. 603-612.

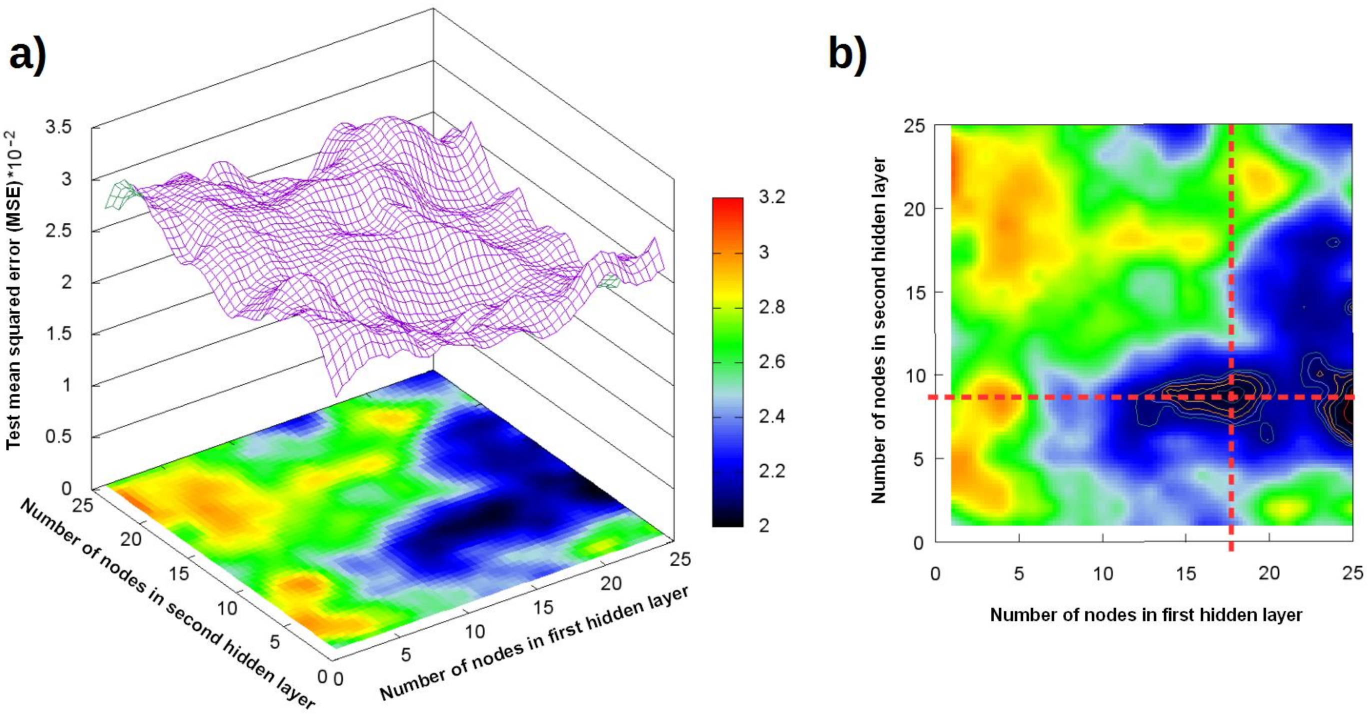

Fig. 4. Relation between the network testing error and the networks' topology for the optimum number of training cycles

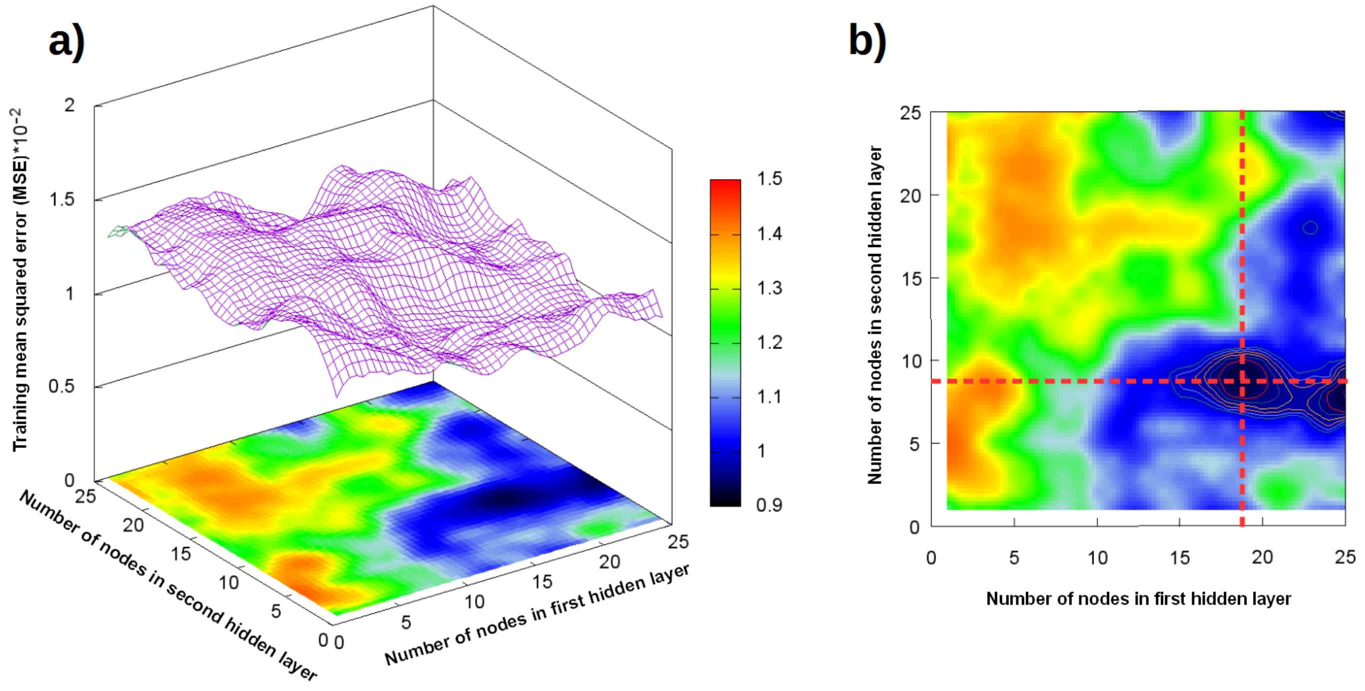

Fig. 5. Dependence of network learning error on their topology for the optimal number of training cycles

8. Szaleniec, J. & Wiatr, M. & Szaleniec, M. & Składzień, J. & Tomik, J. & Oleś, K. & Tadeusiewicz, R.: Artificial neural network modelling of the results of tympanoplasty in chronic suppurative otitis media patients. Computers in Biology and Medicine. 2013. Vol. 43, Iss. 1. P. 16-22.
9. Szaleniec, M. & Tadeusiewicz, R. & Witko, M.: How to select an optimal neural model of chemical reactivity? Neurocomputing. 2008. Vol. 72, Iss. 1-3. P. 241-256.
10. Aleksendrić, D.: Neural network prediction of brake friction materials wear. Wear. 2010. Vol. 268, Iss. 1-2. P. 117-125.
11. Bergmeir, C. & Benítez, J.M.: Neural Networks in R Using the Stuttgart Neural Network Simulator: RSNNS. Journal of Statistical Software. 2012. Vol. 47, Iss. 7. P. 1-26.
12. Olesiuk, D. & Bachmann, M. & Habermeyer, M. & Heldens, W. & Zagajewski, B.: SNNS application for crop classification using hymap data. Proceedings of the Whispers - 2nd Workshop on hyperspectral image and signal processing. 2010. P. 1-4.
13. Riedmiller, M. & Braun, H.: A direct adaptive method for faster backpropagation learning the RPROP algorithm. IEEE International Conference on Neural Networks. 1993. Vol. 1. P. 586-591.
14. Mrówczyńska, M.: Estymacja błędów modelu powierzchni opisanych funkcjami kształtu za pomocą sieci neuronowych. Acta Scientinarum Polonorum, Geodesia et Descriptio Terrarum. 2007. Vol. 6, Iss. 1. P. 15-23.
15. Zell, A. & Mamier, G. & Vogt, M. & Mache, N. & Hübner, R. & Döring, S. & Herrmann, K.U. & Soyez, T. & Schmalzl, M. & Sommer, T. & Hatzigeorgiou, A. & Posselt, D. & Schreiner, T. & Kett, B. & Clemente, G. & Wieland, J.: Stuttgart Neural Network Simulator User Manual, Version 4.2. http://www.ra.cs.uni-tuebingen.de/downloads/SNNS/SNNSv4.2.Manual.pdf
16. Yilmazkaya, E. & Dagdelenler, G. & Ozcelik, Y. & Sonmez, H.: Prediction of mono-wire cutting machine performance parameters using artificial neural network and regression models. Engineering Geology. 2018. Vol. 239. P. 96-108.
17. Shebani, A. & Iwnicki, S.: Prediction of wheel and rail wear under different contact conditions using artificial neural networks. Wear. 2018. Vol. 406-407. P. 173-184.

Optymalizacja doboru struktury sztucznych sieci neuronowych w modelowaniu zużycia żeliwa sferoidalnego na samochodowe wałki rozrządu

W pracy przedstawiono proces optymalizacji struktury sztucznych sieci neuronowych użytych do modelowania zużycia żeliwa sferoidalnego. Sieci uczono metodą gradientową RPROP przy użyciu pakietu SNNS wspomaganego autorskim oprogramowaniem, które umożliwiało automatyczne tworzenie, uczenie i testowanie sieci o różnych wielkości warstw ukrytych. Na podstawie analizy wyników procesu uczenia i testowania pakietu 625 sieci dobrano tę, która modelując proces zużycia żeliwa sferoidalnego generuje najmniejsze błędy podczas testowania.

Słowa kluczowe: Sztuczne sieci neuronowe, optymalizacja struktury, zużycie, żeliwo sferoidalne, Stuttgart Neural Network Simulator, Resilient backPROPagation.

Authors:

Kazimierz Witaszek, PhD. Eng. – Silesian University of Technology, Faculty of Transport and Aviation Engineering, Department of Automotive Vehicle Maintenance, kazimierz.witaszek@polsl.pl

Krzysztof Garbala, PhD. Eng. – AC Spólka Akcyjna, ul. 42Pułku Piechoty 50, 15-181 Białystok, e-mail: krzysztofgarbala@tlen.pl

Mirosław Witaszek, PhD. Eng. – Silesian University of Technology, Faculty of Transport and Aviation Engineering, Department of Automotive Vehicle Maintenance, miroslaw.witaszek@polsl.pl

Marcin Rychter, PhD. DSc Eng. – Vocational State School of Ignacy Mościcki in Ciechanów, Faulty of Engineering and Economics, Gabriela Narutowicza 9, 06-400 Ciechanów, Poland, e-mail: rychter@poczta.fm