BULGARIAN ACADEMY OF SCIENCES
PROBLEMS OF ENGINEERING CYBERNETICS AND ROBOTICS • 2021 • Vol. **77**, pp. 11-19
p-ISSN: 2738-7356; e-ISSN: 2738-7364
https://doi.org/10.7546/PECR.77.21.02

Encog Gradient Training Algorithms Evaluation

Petar Tomov

Bulgarian Academy of Sciences
Institute of Information and Communication Technologies
Acad. Georgi Bonchev Str., block 2, office 514
1113 Sofia, Bulgaria
E-mail: petar.tomov@iict.bas.bg

Abstract: Encog Machine Learning Framework is a software programming library for artificial intelligence solutions. It has a wide range of data structures and algorithms. This study investigates the capabilities of Encog for gradient-based training of three-layers perception. The available algorithms are compared in forecasting time series.

Keywords: *Artificial neural networks, Encog, gradient training.*

1. Introduction

Text classification is an essential part of applications like web searching, filtering, language identification, readability, etc. with active usage of neural networks. A review of machine learning models in an interdisciplinary scientific field combining computer science, artificial intelligence, and linguistics can be found in (Mankolli & Guliashki, 2020). Artificial neural networks have been a very popular tool for solving various problems including e-commerce (Wei et al., 2021)) wind energy (Elyasichamazkoti & Khajehpoor, 2021), finance forecasting (Tomov et al., 2019), etc. Among the different types of artificial neural networks, the most widely used is the multilayer perceptron (Gardner & Dorling, 1998). The multilayer perceptron is a directed weight graph organized in layers (Jain et al, 1996). In the most commonly used embodiment, the multilayer perceptron has full connectivity between two adjacent layers (Lecun et al., 1998). Each node in the graph serves as an artificial neuron. At the input of each neuron, input signals are received, which are summed, in the general case. The sum is then transmitted for

normalization by an activation function, which in most cases is a sigmoid or hyperbolic tangent (Zamanlooy & Mirhassani, 2014). The normalized value is provided as an output value of the neuron. The signals entering the input layer are formed in the environment external to the artificial neural network (Lin & Lee, 1994). The input signals in all subsequent layers are products of previous layer signals and weights of the connections (Amit, 2019). The process of artificial neural network training is to find optimal values for all weights in the graph (Abdull Hamed, 2012). For some cases, it is possible to compose hierarchical knowledge graphs (Yoshinov et al., 2020).

There are many different training algorithms. The most popular and most efficient training algorithms are gradient-based exact numerical algorithms (Bengio, 2012). Gradient training is based on the total error produced as an output of the artificial neural network (Mustafa et al., 2012). The gradient gives the direction of the total error slope (Verma et al., 2016). The weights are adjusted in the opposite direction. The error propagates back from the output layer to the input layer (Rumelhart et al., 1986). The weights inside the artificial neural networks are adjusted according to the error propagated back (Wang et al., 2005). The adjustment is performed so that each weight drives a reduction of the total error. Gradient-based training is fast but often leads to a local optimum (Ilonen et al., 2003). Backpropagation of the error is the most popular training, but there are many derivatives and improvements (Riedmiller, 1994).

This study compares some of the gradient-based training algorithms provided in the Encog machine learning software library. A three-layer perceptron is trained for financial time series forecasting. After the introductory section, the paper is organized as follows: Section 2 presents common gradient-based training algorithms; Section 3 presents the experiments and obtained results and Section 4 contains some conclusions and guidance for further research.

2. Gradient-Based Trainings

Backpropagation is the most common feed-forward neural network training algorithm. It has a learning rate parameter. The learning rate determines the amount to which the weights will be modified in each iteration. One major problem with backpropagation is that the magnitude of the partial derivative is often straitening to the training of the neural network. The other propagation methods of Resilient or Manhatten address this issue in distinct ways.

A problem with backpropagation is that partial derivatives can be too large or too small. The other issue is that the learning rate is a single value for the whole process of training. The resilient propagation training uses a separate update value (as the learning rate) for every connection. These updated values are automatically estimated. It is not the case with the learning rate in backpropagation which is

usually predefined. Quick propagation is a method based on Newton's method that involves a quadratic approximation of the preceding gradient step and the present gradient. In this way, it is expected to be close to the minimum of the network's total error function. The suggestion is that the network's total error function is locally nearly square. As similar to an upward open parabola. A general problem is the artificial neural network can behave chaotically during the training process due to big step sizes.

Scaled conjugate gradient training is based on conjugate directions. This algorithm does not perform a line search at each iteration, unlike other conjugate gradient algorithms. If a line search for each iteration, it will make the training computationally inefficient. Training can be used for neural networks when network functions have derivatives.

Manhattan propagation attempts to solve the partial derivatives magnitude problem by only indicating the sign of the derivative. The weights adjustment is a constant value. The exact value of the constant is experimentally determined. The basic approach is starting with a higher value and decreasing it. Manhattan propagation is like a simplified version of resilient propagation.

3. Experiments and Results

Encog supported gradient trainings are compared on a three-layered feed-forward neural network. The task of the network is the forecasting time series. Two separate datasets were used. The first dataset is a simple sine function as it is shown in Fig. 1.

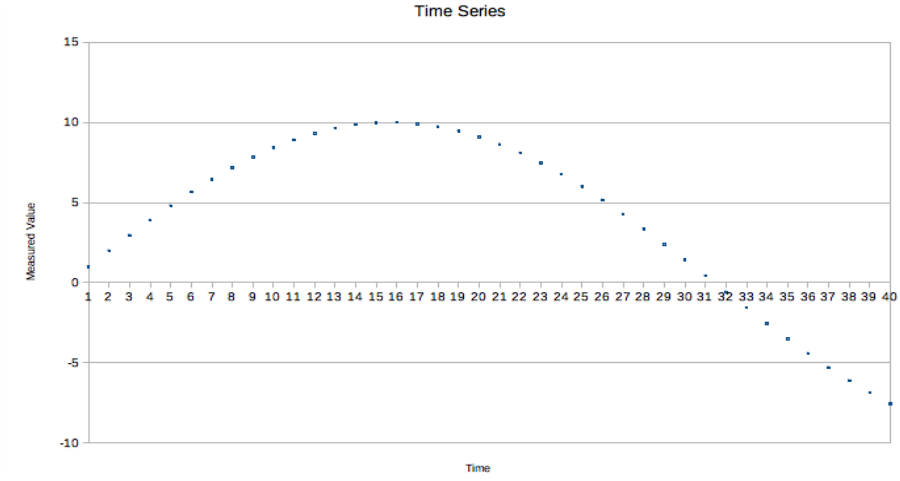

Fig. 1. Simple sine function dataset

The experimental results show clear outperformance of backpropagation training when it is done on simpler data (see Fig. 2 to Fig. 6).

The dependency of the training loops over time is almost linear, as is shown in Fig. 2a-6a. For backpropagation, it could be seen, from Fig. 2b, that after 5 seconds, the total error decreases significantly to 7 decimal places. For the resilient propagation, total network error oscillates around values near to 6 decimal places, as it is shown in Fig. 3b. Quick propagation is pretty inefficient as a training algorithm with a single spike in total network error around 3 decimal places in 50th second, shown in Fig. 4b. A scaled conjugate gradient about 5th second gives a sharp slope-down with values for the total error around 8 decimal places, which is shown in in Fig. 5b.

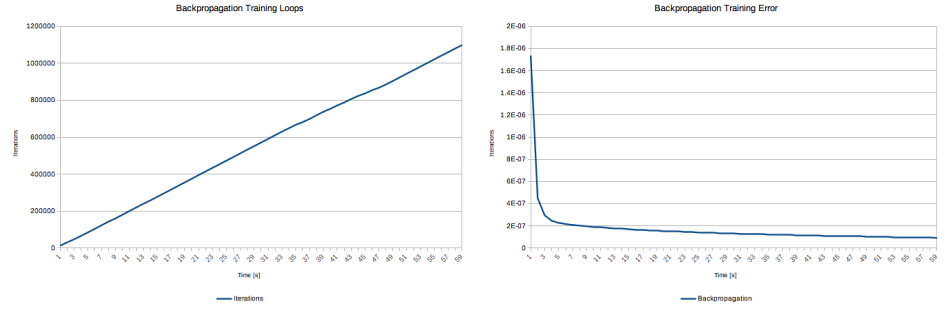

Fig. 2. Backpropagation training – number of iterations (left), total error (right)

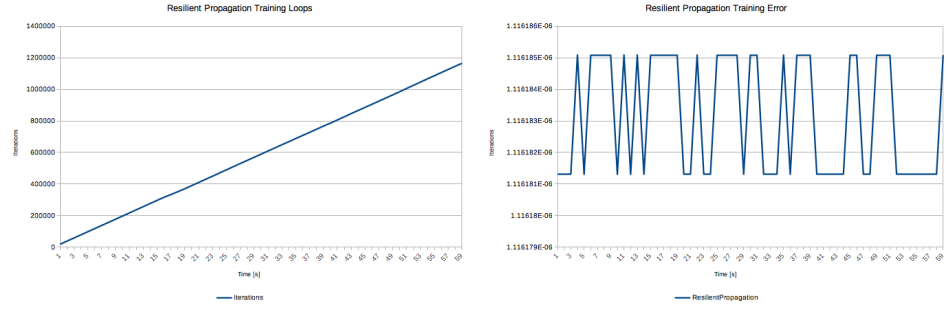

Fig. 3. Resilient propagation training – number of iterations (left), total error (right)

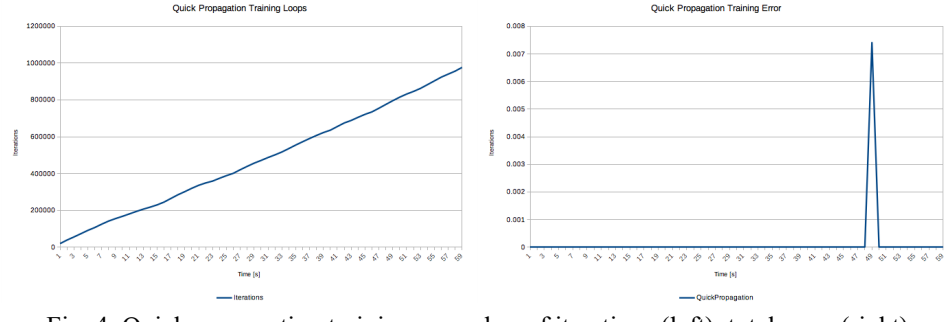

Fig. 4. Quick propagation training – number of iterations (left), total error (right)

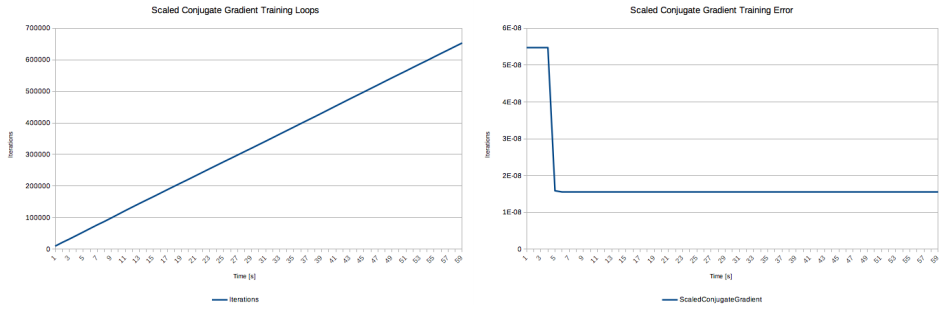

Fig. 5. Scaled conjugate gradient training – number of iterations (left), total error (right)

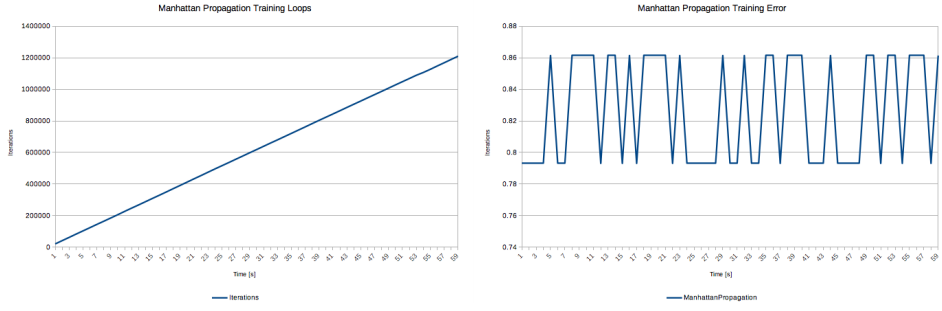

Fig. 6. Manhattan propagation training – number of iterations (left), total error (right)

Manhattan propagation is oscillating between 0.8 and 0.9, which cannot be accepted as an efficient artificial neural network training. With a simpler time series, backpropagation and scaled conjugate gradient are the most convenient training algorithms.

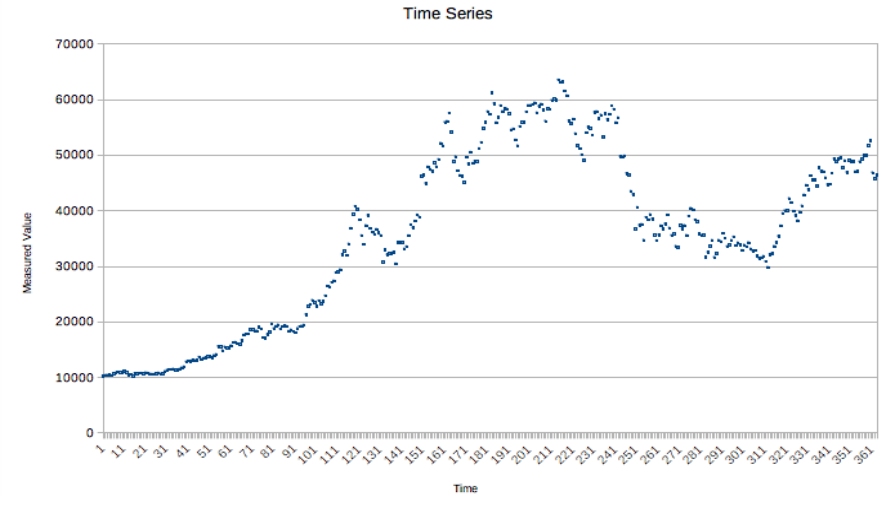

Fig. 7. Bitcoin historical price dataset

The second dataset consists of Bitcoin's historical price in US dollars (Fig. 7). With much more complicated data, resilient propagation outperforms the other four algorithms as it is shown in Fig. 8 to Fig. 12. Training loops in dependency of the time are still in almost linear functional relation, which is visible in Fig. 8a-12a. The backpropagation algorithm starts from almost 0.003 total error and slightly goes almost to 0.002 in 60 seconds, shown in Fig. 8b. Resilient propagation goes sharp slope down in first 4 seconds, after that the slope gets smaller and it gets almost flat after the 15 seconds, visualized in Fig. 9b. Quick propagation starts promising in the first 7 seconds, but after that convergence gets almost flat, as shown in Fig. 10b. The scaled conjugate gradient gets sharp down, in the beginning, first 7 seconds and the convergence gets lower slope after that, as illustrated in Fig. 11b.

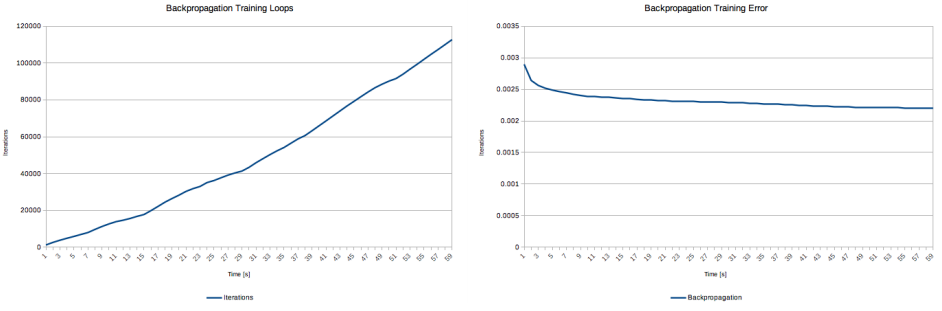

Fig. 8. Backpropagation training – number of iterations (left), total error (right)

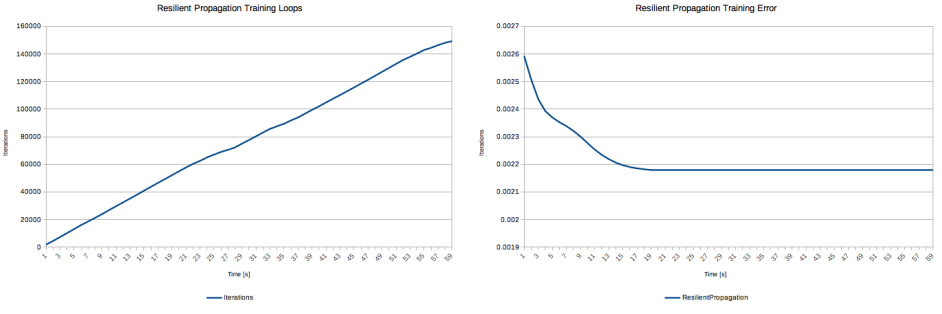

Fig. 9. Resilient propagation training – number of iterations (left), total error (right)

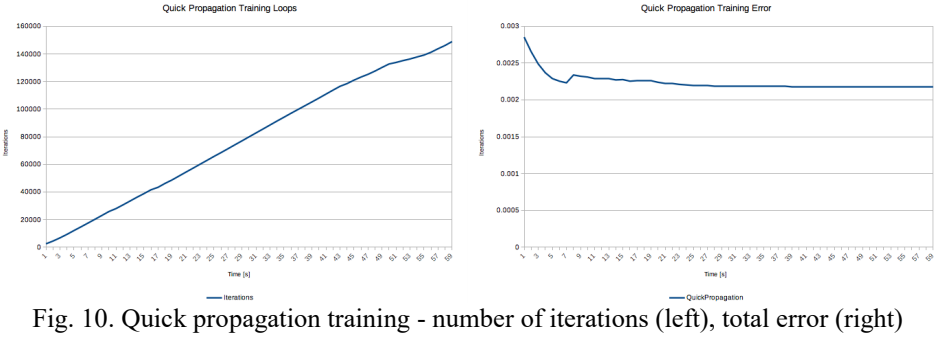

Fig. 10. Quick propagation training - number of iterations (left), total error (right)

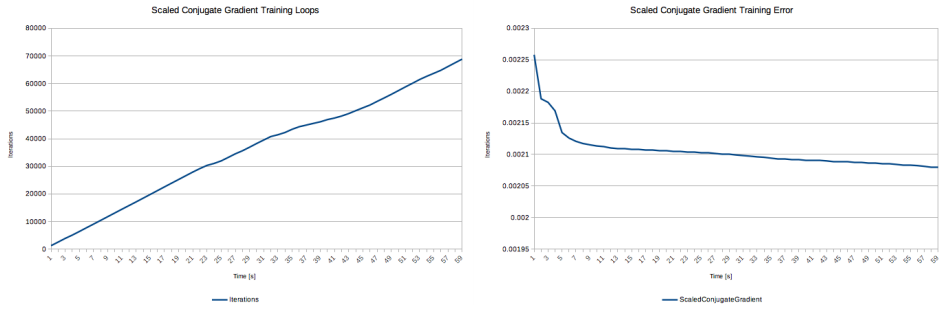

Fig. 11. Scaled conjugate gradient training – number of iterations (left), total error (right)

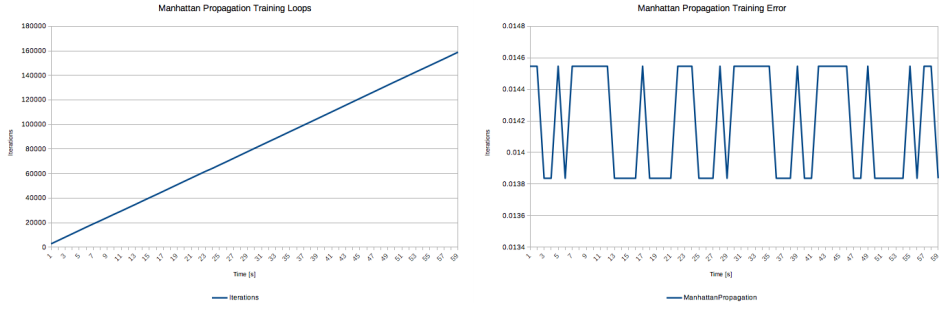

Fig. 12. Manhattan propagation training – number of iterations (left), total error (right)

Manhattan propagation oscillates between 0.0146 and 0.0138, which makes it inefficient for artificial neural networks training even with bigger time series, as shown in Fig. 12b.

In both cases, Manhattan propagation gives less promising results than all other four algorithms. Reduction of the training time is not always acceptable when the achieved results are not better.

4. Conclusion

In the present study, a comparison of Encog supported gradient training algorithms has been done. Such training is very promising when applied to feed-forward artificial neural networks used in time series forecasting problems. The results achieved clearly show that a hybrid implementation of compared algorithms can give extra efficiency. It can be concluded that the compared algorithms can be useful in real industrial applications.

As for directions for further research, it will be interesting evolutionary algorithms to be compared and gradient-based training to be extended with such optimization heuristics.

Acknowledgments

This research is funded by Velbazhd Software LLC and it is partially supported by the Bulgarian National Science Fund by the project "Mathematical models, methods and algorithms for solving hard optimization problems to achieve high security in communications and better economic sustainability", KP-06-N52/7/19-11-2021.

References

1. Abdull Hamed, H.N., Shamsuddin, S.M., Salim, N.: Particle Swarm Optimization for Neural Network Learning Enhancement. Jurnal Teknologi 49(1), 13-26, 2012.
2. Amit, Y.: Deep Learning with Asymmetric Connections and Hebbian Updates, Frontiers in Computational Neuroscience 13, pp. 18, (2019), DOI 10.3389/fncom.2019.00018.
3. Bengio, Y.: Practical Recommendations for Gradient-Based Training of Deep Architectures. In: Montavon G., Orr G.B., Müller KR. (eds) Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science, vol. 7700. Springer, Berlin, Heidelberg. (2012), https://doi.org/10.1007/978-3-642-35289-8_26.
4. Elyasichamazkoti, F., Khajehpoor, A.: Application of machine learning for wind energy from design to energy-Water nexus: A Survey. Energy Nexus 2,100011, 2021, https://doi.org/10.1016/j.nexus.2021.100011.
5. Ilonen, J., Kamarainen, JK., Lampinen, J.: Differential Evolution Training Algorithm for Feed-Forward Neural Networks. Neural Processing Letters 17, 93–105 (2003). https://doi.org/10.1023/A:1022995128597.
6. Jain, A. K., Mao, J., Mohiuddin, K.M.: Artificial neural networks: a tutorial. Computer 29(3), 31-44 (1996), doi: 10.1109/2.485891.
7. Lecun ,Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. In: Proc. of the IEEE, vol. 86(1)1, pp. 2278-2324, 1998, doi: 10.1109/5.726791.
8. Lin, Ch.-T., Lee, C.S.G.: Reinforcement structure/parameter learning for neural-network-based fuzzy logic control systems. IEEE Transactions on Fuzzy Systems 2(1), 46-63 (1994), doi: 10.1109/91.273126.
9. M.W Gardner, S.R Dorling, Artificial neural networks (the multilayer perceptron) – a review of applications in the atmospheric sciences. Atmospheric Environment 32(14-15), 2627-2636 (1998), https://doi.org/10.1016/S1352-2310(97)00447-0.
10. Mankolli, E., Guliashki, V.: Machine Learning and Natural Language Processing: Review of Models and Optimization Problems. In: Dimitrova V., Dimitrovski I. (eds) ICT Innovations 2020. Machine Learning and Applications. Communications in Computer and Information Science, vol. 1316 (2020). https://doi.org/10.1007/978-3-030-62098-1_7.
11. Mustafa, M.R., Rezaur, R.B., Saiedi, S. et al. River Suspended Sediment Prediction Using Various Multilayer Perceptron Neural Network Training

Algorithms – A Case Study in Malaysia. Water Resour Manage 26, 1879–1897 (2012), https://doi.org/10.1007/s11269-012-9992-5

12. Riedmiller, M.: Advanced supervised learning in multi-layer perceptrons – From backpropagation to adaptive learning algorithms. Computer Standards & Interface 16(3), 265-278 (1994), https://doi.org/10.1016/0920-5489(94)90017-5.
13. Rumelhart, D., Hinton, G. & Williams, R. Learning representations by back-propagating errors. Nature 323, 533–536 (1986). https://doi.org/10.1038/323533a0.
14. Tomov, P., Zankinski, I., Balabanov, T. Training of Artificial Neural Networks for Financial Time Series Forecasting in Android Service and Widgets. Problems of Engineering Cybernetics and Robotics, vol. 71, 2019.
15. Verma, A.K., Singh, T.N., Chauhan, N.K. et al. A Hybrid FEM–ANN Approach for Slope Instability Prediction. J. Inst. Eng. India Ser. A 97, 171–180 (2016). https://doi.org/10.1007/s40030-016-0168-9.
16. Wang, H.B., Xu, W.Y., Xu, R.C.,: Slope stability evaluation using Back Propagation Neural Networks, Engineering Geology 80(3-4), 302-315 (2005), https://doi.org/10.1016/j.enggeo.2005.06.005.
17. Wei, C., Wang, Q. & Liu, C. Application of an artificial neural network optimization model in e-commerce platform based on tourism management. Journal on Wireless Communications and Networking 93 (2021). https://doi.org/10.1186/s13638-021-01947-x.
18. Yoshinov, R., Kulikov, I., Zhukova, N.: Methods of composing hierarchical knowledge graphs of telecommunication networks. Problems of Engineering Cybernetics and Robotics 72, 69–78, (2020), https://doi.org/10.7546/PECR.72.20.07.
19. Zamanlooy, B., Mirhassani, M.: Efficient VLSI Implementation of Neural Networks With Hyperbolic Tangent Activation Function. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 22(1), 39-48, 2014, doi: 10.1109/TVLSI.2012.2232321.