Using the Actor-Critic Method for Population Diversity in Neuroevolutionary Synthesis

Serhii Leoshchenko[a], Andrii Oliinyk[a], Sergey Subbotin[a] and Vadym Shkarupylo[b,c]

[a] *National university "Zaporizhzhia polytechnic", Zhukovskogo street 64, Zaporizhzhia, 69063, Ukraine*
[b] *National University of Life and Environmental Sciences of Ukraine, Heroiv Oborony Str. 15, Kyiv, 03041, Ukraine*
[c] *G.E. Pukhov Institute for Modelling in Energy Engineering, NAS of Ukraine, General Naumov Str. 15, Kyiv, 03164, Ukraine*

Abstract

Training methods are used in most applications that use artificial neural networks as mathematical models to solve modeling, diagnosis, prediction, and evaluation problems. However, the rapid development of industries requires methods that would be more automated and less require the involvement of experts. To ensure this level of automation, a more detailed presentation of solutions is necessary. However, when detailing the view, another problem arises: the search space becomes huge, and therefore there is a need for an appropriate scalable and productive search method. To solve both problems, usually firstly it is proposed a powerful solution that combines most of the functions of neural networks from the literature into one representation. Secondly, based on the new concept of the chromosomal spectrum, a new method of preserving diversity is being created, called spectral diversity, which creates a spectrum from the characteristics and frequency of alleles in a chromosome. Combining genetic diversity with a unified representation of neurons allows the method to either surpass or match the neuroevolution of augmenting topologies (NEAT) in most test tasks. In part, the good results can be explained by the novelty of evolution and the good scalability of chromosome sizes provided by the diversity spectrum. This explains the importance of researching the impact of genetic diversity on the success of artificial neural network synthesis. The paper proposes a study that sheds light on a new mechanism of representation and conservation of diversity during neuroevolutionary synthesis.

Keywords

Neuromodels, neuroevolution, ANN, genetic diversity, synthesis, topology, artificial intelligence, training.

1. Introduction

Today, the device of artificial neural networks (ANN) is actively used in solving a number of problems related to the classification and clustering of complex nonlinear systems and objects [1–3]. These tasks may include: diagnostics, forecasting of condition and behavior, modeling and evaluation, and among the tasks can be distinguished: technical and biomedical diagnostics, financial, industrial, medical diagnostics, and etc. [4], [5]. This popularity of the ANN is explained by the high level of applicability when working with complex and nonlinear objects and systems, where the use of classical approaches is insufficient and demonstrates low levels of accuracy of work.

IntelITSIS'2021: 2nd International Workshop on Intelligent Information Technologies and Systems of Information Security, March 24–26, 2021, Khmelnytskyi, Ukraine
EMAIL: sergleo.zntu@gmail.com (S. Leoshchenko); olejnikaa@gmail.com (A. Oliinyk); subbotin@zntu.edu.ua (S. Subbotin); shkarupylo.vadym@nubip.edu.ua (V. Shkarupylo)
ORCID: 0000-0001-5099-5518 (S. Leoshchenko); 0000-0002-6740-6078 (A. Oliinyk); 0000-0001-5814-8268 (S. Subbotin); 0000-0002-0523-8910 (V. Shkarupylo)

As in most cases, during using the ANN, it can be getting a certain model based on certain previous (historical) data about the object (process) under study [6-8]. However, it is worth noting that in most cases, the first model based on the ANN goes through the stage of topological construction or structural synthesis [9-11]. Based on the task and its features, the structure of the future ANN is selected: the number of computing nodes (neurons), the number of layers, the presence of feedback in neurons, the number and depth of hidden layers, additional filters and memory cells in neurons, and so on [9]. The result is the structure of the ANN, which is sent to the second stage – parametric synthesis or training [10]. At this stage, using one of the training methods (usually based on gradient methods), the weights of interneuronal (and sometimes inverse) relationships are determined. So the model is based on the classical principles of learning with a teacher and in the case of technical and biomedical problems demonstrates good results [2], because it uses real historical data [8].

Neuroevolution as an approach to ANN synthesis has emerged as an alternative approach [12-16] that can be used for the basics of unsupervised learning and reinforcement learning [14-16]. Since neuroevolution is based on stochastic evolutionary methods, such synthesis can be performed without taking into account previous historical data about the object (system). Moreover, classical neuroevolution methods perform both postural structural and parametric synthesis, significantly reducing the risks associated with the expert's lack of awareness about the complexity of the ANN for a specific task. However, most neuroevolution methods can also use historical object data, which allows them to be used for tasks with a high degree of accuracy. However, when solving a number of problems related to automating the process of ANN synthesis [12], [13], neuroevolution methods also face a number of problems. For example, a lack of genetic diversity [17-22]. This problem can be caused by a number of factors: insufficient computing power, the use of truncation selection, and an incorrect combination of evolutionary operators [17], [18]. One of the ways to solve this problem is to use methods based on swarm intelligence, but such solutions can also impose certain restrictions and additional requirements on the system. As a result, a situation may occur when methods that seem to be insured against problems with local extremes still have such problems [18].

That is why it is an urgent task to develop mechanisms for maintaining genetic diversity during neurosynthesis.

2. Related works

Indeed, it should be mentioned at the beginning that neuroevolution, as a form of machine learning, appeared as an alternative to training using gradient methods [12-16]. This approach used the principles of most evolutionary algorithms to synthesize neural networks. It is also necessary to distinguish a whole separate group of neuroevolutionary methods: methods that perform the evolution of interneuronal connections and network topologies (TWEANNs) [23], [24], which are individuals in a population.

It is also important to note that the use of neuroevolutionary methods has made it possible to implement approaches to unsupervised learning and reinforcement learning [25-31]. That is why this approach is actively used in industries where the subject area is poorly studied [30], [31], dynamic, or requires a rather complex description, for example, games or controlling robot drives. In these cases, it is quite simple to measure the performance of the neural network, while it is very difficult or almost impossible to implement training with a teacher.

If it is talking about cases when information about the object of research is still available (for example, historical information from sensors installed on the object) [3-6], it should be noted that neuroevolution methods allows to find the most optimal topology of the ANN. In this case, optimality should be understood as the topology model that will provide the greatest accuracy with the simplest structure, since such a structure will be obtained as a result of gradual changes. In addition, we emphasize that in most cases such methods do not have any special problems with local extremums in the search space.

Researching behavioral algorithms and patterns, new methods were later developed that mimic the training of animals in the real world and use a certain assessment of the reward (punishment) for correct (erroneous) actions [26-29]. One of the most popular methods from this group is the actor-

critic method. To date, there are several variations of this method: A2C and A3C [32-34]. These methods are actively used for training, for example, recurrent ANNs, since in this case the use of gradient methods faces a number of difficulties and limitations.

One of the problems with reinforcement learning is that the data coming to the input of the training method is strongly correlated: each subsequent state directly depends on the actions taken by the agent. Learning from highly correlated data leads to retraining. Thus, in order to successfully train a strategy that generalizes to a large number of environmental states, it is still necessary to learn from episodes from different scenarios.

One way to achieve this is to run multiple agents in parallel [25]. All agents are in different states and choose different specific actions according to the stochastic strategy, thereby eliminating the correlation between the observed data. However, all agents use and optimize the same set of parameters.

The idea behind A3C is to run multiple agents in parallel, with each of the agents calculating updates for reward values at each stage. However, instead of just continuing, each agent updates the status and reward common to all agents. Before processing each new episode, the agent copies the current global values of the reward parameter and uses it to determine its own strategy for that episode. Agents do not wait for other agents to finish processing their episodes to update global parameters (hence asynchronous). Therefore, while one of the agents processes one episode, the global value of the reward may change as a result of the actions of other agents.

3. Proposed method

Thus, using the previously presented materials, it can be concluded that maintaining genetic diversity during neurosynthesis can significantly improve the results [35-39]. However, the mechanisms that can be used in this process should be clearly defined [35], [36]. Simply increasing the probability of mutation during synthesis can expand diversity, but it does not guarantee that mutations will lead to a better solution. The use of indicators and markers to intelligently determine the type of mutation can complicate the mutation, but it can also not be the key to successful diversity, moreover, in most cases, this approach is based on determining the complexity of the task and network and does not guarantee an expanded population at the beginning of work [33]. That is why it is necessary to identify and develop a new approach that can be used during neurosynthesis for little-studied tasks and guarantees initial and consistent genetic diversity throughout the entire neurosynthesis process.

Using the A3C method shows good results when used during ANN training [34]. Therefore, our approach suggests using the main strategy of this method during neurosynthesis. Thus, at the beginning of the synthesis, we will create a population consisting of a number of individual actors ($NN_{actors} = \{NN_{actor_1}, NN_{actor_2}, ..., NN_{actor_n}\}$) and one individual critic (NN_{glob_crit}). Additionally, the critical value of the reward (Q) will be determined and the error will be saved from the outputs ($\varepsilon_{outNN_{glob\_crit}}$). Note that from the very beginning, all individuals have access to the training sample. After that, the synthesis process begins, which begins with identifying genetic information among people and encoding this information. Further, based on the logic of the A3C method, the neural network population ($NN_{actors} = \{NN_{actor_1}, NN_{actor_2}, ..., NN_{actor_n}\}$) receives training data at its input, and outputs actions ($action^{out}_{NN_{actor_n}}$) at the output, which should lead to an increase in reward and a decrease in error ($Q \to \max, \varepsilon_{outNN_{glob\_crit}} \to \min$). Although these actions are random and simply depend on the signal passing through the network, this is due to the fact that the neural network has not yet been trained. The global neural network critic also has access to the source data, but also syncs actions with the output of the actors' networks. And at the output, let it predict only the reward that will be received if you apply these actions.

Then the general work can be reduced to the following: what should be the optimal actions at the output of networks that lead to an increase in the reward is not known – parametric synthesis is not possible. And the neural network actor can predict the exact value of the reward (or rather, usually its change), which he will receive if he now applies actions actions. This allows you not only to use the

error change gradient from the critic's network and apply it to actors, but also to determine the topological features of actors. So when synchronizing values from actor networks, we can gradually determine the type of mutation, conditionally dividing the entire population by approaching the critical difference in the initial value:

$$\varepsilon^{out}_{NN_{actor_n}} = \begin{cases} < Value_1, case_1 \\ Value_1 \le \varepsilon^{out}_{NN_{actor_n}} \le Value_2, case_2 \\ \dots \\ > Value_m, case_m \end{cases}. \quad (1)$$

Then, for neural networks with more or less similar sequences, you can determine a certain type of mutation, which allows you to investigate whether the mutation will affect the accuracy of similar neural networks.

At Figure 1 shown the general scheme of operation of the method.

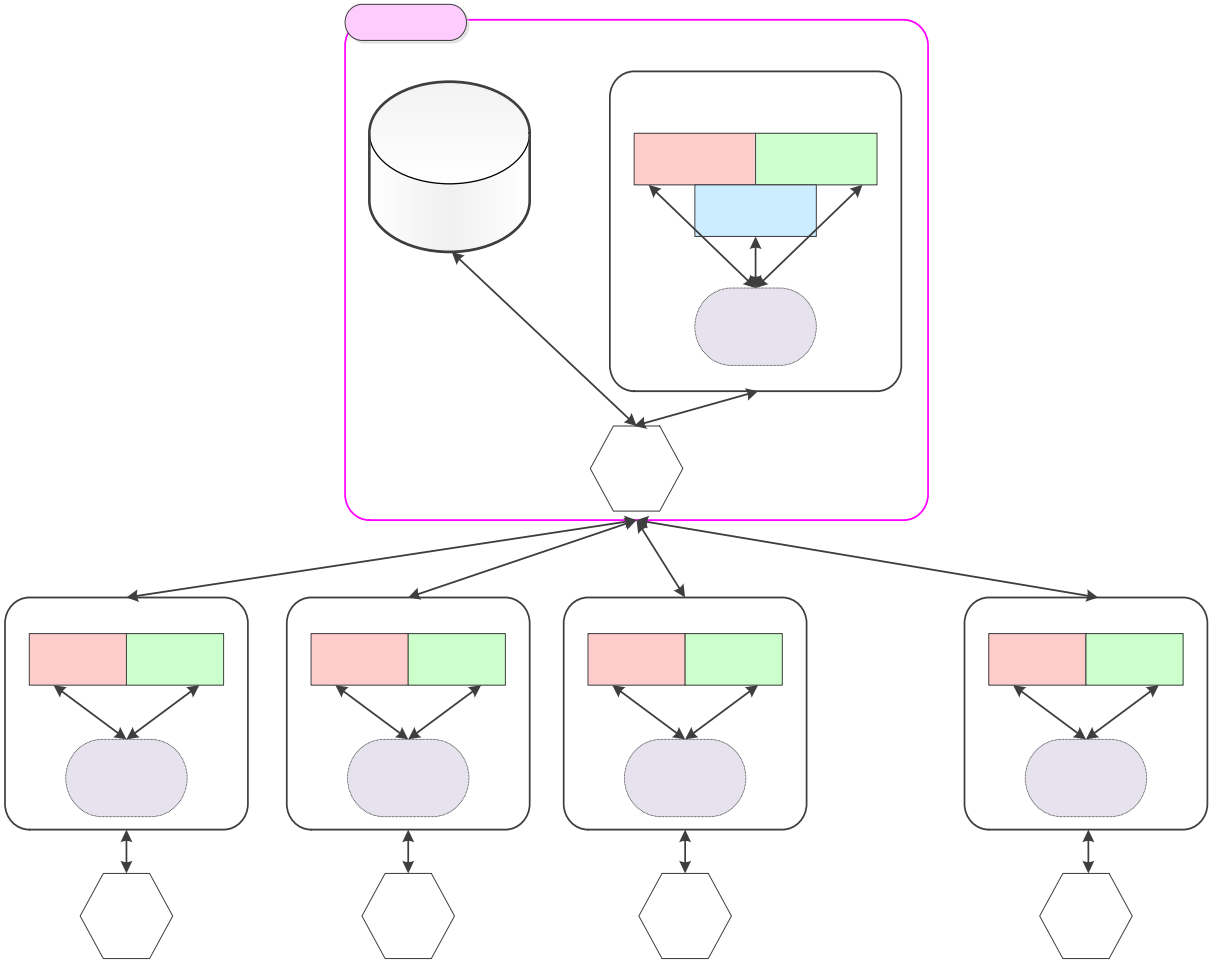

Figure 1: The general scheme of the method

As you can see, in this case, actions are optimized directly by the reward signal. This is the general essence of all Model-free algorithms in reinforcement learning. They are the state-of-the-art at the moment.

Therefore, when using the proposed approach, the advantages can be considered that:

- gradient methods can be used to find optimal actions, which can find the most optimal solutions;
- the ability to use small (and therefore faster learners) neural networks, which will gradually complicate their architectures. Therefore there is a chance to find the optimal value with less resource usage;
- even when using selective clipping, diversity will be maintained and ensured.

This allows us to provide a global strategy: provided that out of all the variety of environmental factors, specific ones are key to solving the problem, the method is quite capable of identifying them. And use it to solve the problem.

4. Results and Discussions

For an experimental study of the method's operation, it was decided to test it on two problems that would differ in scale. This is how the Tic-Tac-Toe Endgame data set was selected [40] and Taiwanese banking Prediction data Set [41]. The main characteristics of the samples are shown in Tables 1 and 2, respectively.

Table 1
Characteristics of the Tic-Tac-Toe endgame data set

Tic-Tac-Toe Endgame Data Set			
Data Set Characteristics:	Multivariate	Number of Instances:	958
Attribute Characteristics:	Categorical	Number of Attributes:	9

Table 2
Characteristics of the Taiwanese Bankruptcy Prediction data set

Taiwanese Bankruptcy Prediction Data Set			
Data Set Characteristics:	Multivariate	Number of Instances:	6819
Attribute Characteristics:	Integer	Number of Attributes:	96

We will compare the work of the proposed method (A3C GA) with the classical neuroevolution genetic algorithm (GA) [42] and the modified genetic algorithm (MGA) [43], [44]. The main modifications of the classical method will consist in the use of adaptive mechanisms at the stages of mutation and crossover. Based on the estimation of the complexity of the problem and the topological complexity of the solutions, the methods allow for more precise and point-based mutation production. The crossing stage is enhanced by the use of uniform crossover, which makes it possible to model other types of crossover (one - and two-point), but ideally allows for multi-parent crossover and inheriting the best genetic features from several parents at once. All this complex of modifications allows to:

- track not only the best solutions for the accuracy of the test (values of the activation function), but also for structural features;
- get the best individuals in new generations with several parental characteristics at once.

For all methods, the same metaparameters specified in Table 3 will be specified.

Table 3
Metaparameters for methods

Metaparameter	Value
Population size	100
Elite size	5%
Activation function (fitness functions)	hyperbolic tangent
Mutation probability (for GA and MGA)	25%
Crossover type	two-point
Types of mutation	
Population size	deleting an interneuronal connection removing a neuron adding interneuronal connection adding a neuron changing the activation function

The test results for the tic-Tac-Toe Endgame data set sample are shown in Table 4, and for the Taiwanese banking Prediction data set sample are shown in Table 5.

Table 4
Test results on the Tic-Tac-Toe Endgame data set

Method	Synthesis Time, s	Error in the training sample	Error in the test sample
GA	2095	0.021	0.14
MGA	1867	0	0.022
A3C GA	1963	0	0.08

Analyzing the presented data, it can be came to the following conclusions. The proposed modification makes it possible to really improve the accuracy of the synthesized resulting ANN. Studying the work, it can be concluded that maintaining genetic diversity can indeed help to obtain the resulting ANN with higher accuracy, but such a mechanism loses out in the time of MGA synthesis, where a criterion mechanism for determining the type of mutation is used.

Table 5
Test results on the Taiwanese Bankruptcy Prediction data set

Method	Synthesis Time, s	Error in the training sample	Error in the test sample
GA	8762	0.017	0.25
MGA	7586	0.011	0.13
A3C GA	10689	0.017	0.24

In the second case, it is noticeable that even classical GA turned out to be faster, and the difference in accuracy does not sufficiently justify such time delays. For additional research, we will consider the system load graphs (CPU and RAM) during synthesis in Figures 2 and 3. measurements were performed every minute using system load monitoring [45].

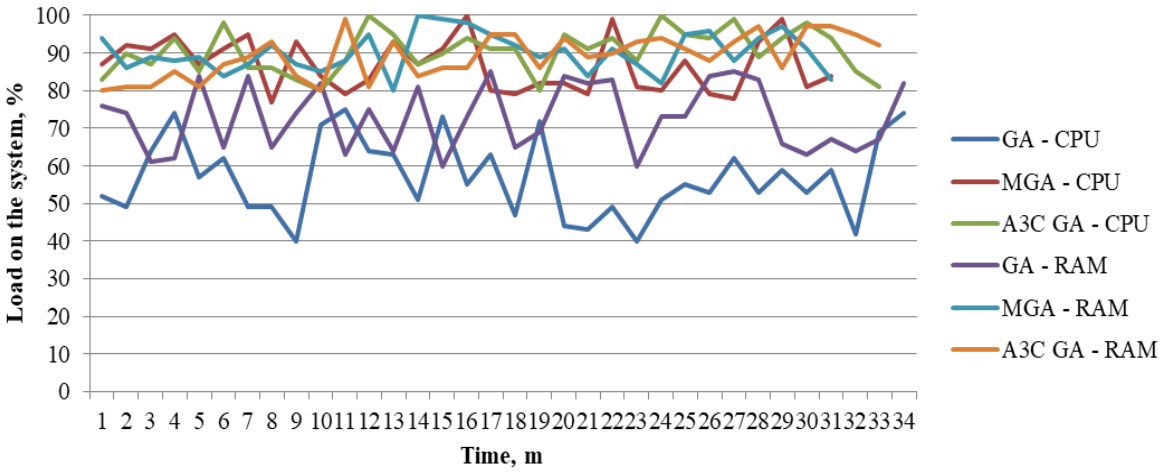

Figure 2: Load on the system when working with the Tic-Tac-Toe Endgame data set

Examining the load graphs, it can be noted that during execution, any modifications load the system additionally through calculations, but it should also be noted that even when using non-complex networks in the second case, the A3C GA reached peak values of 100%. Of course, such load monitoring is not sufficiently objective, but these indicators can still be used for a general description of the load at runtime.

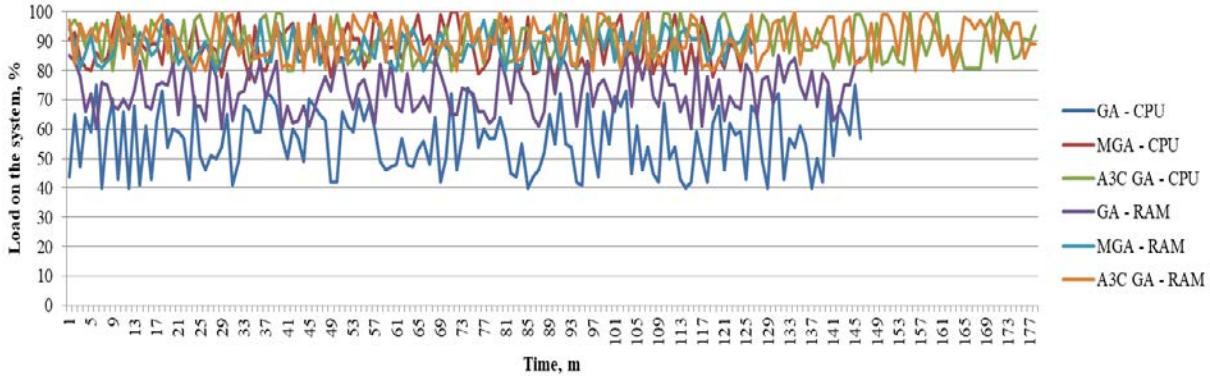

Figure 3: Load on the system when working with the Taiwanese Bankruptcy Prediction data set

Accordingly, the following recommendations can be generated for using this approach:

- at the initial stages of neurosynthesis, simple ANN topologies should be used;
- data for training should be available constantly, so as not to overload the accessors;
- when designing an interacting neuroevolution method, more compact methods of encoding genetic information about individuals should be used.

It should also be noted that such an approach can be difficult to parallelize due to the too frequent exchange of information between agents and the critic. In this case, it is possible to organize separate transfers of only information about the state of the agents and only after a certain completion of the synthesis. But, since access to information is provided through the main agent, such a system can be scattered on lightweight streams of video cards.

5. Conclusion

Numerous works prove that solving the problem of reducing genetic diversity helps to find a more optimal solution in neural network synthesis. The proposed mechanism really showed a certain efficiency: the accuracy of the resulting ANN was increased by 43% (error became from 0.14 to 0.08) compared to the classical GA. The synthesis time was also reduced by 6%. But when testing the new method on large data, some shortcomings were found. Since the ANN population is trained directly on the reward signal, many training examples are required. Tens of millions even for very simple cases. It does not work well on problems with a large number of degrees of freedom. If the method can't immediately identify key factors among a high-dimensional landscape, it probably won't learn at all. The method can also exploit vulnerabilities in the system by focusing on a local extreme – a suboptimal action (if a gradient descent converges to it), ignoring other environmental factors. And the main thing is that when updating data (even if they differ slightly from the original task), the method has to be trained completely again by the ANN.

6. Acknowledgements

The work was carried out with the support of the state budget research projects of the state budget of the National University "Zaporozhzhia Polytechnic" “Intelligent methods and software for diagnostics and non-destructive quality control of military and civilian applications” (state registration number 0119U100360) and “Development of methods and tools for analysis and prediction of dynamic behavior of nonlinear objects” (state registration number 0121U107499).

7. References

[1] V. Sze, Y.-H. Chen, T.-J. Yang, J.S. Emer, Efficient Processing of Deep Neural Networks (Synthesis Lectures on Computer Architecture), Morgan & Claypool Publishers, Williston, 2020.
[2] J. D. Kelleher, Deep Learning (The MIT Press Essential Knowledge series), The MIT Press, Cambridge, 2019.

[3] S. Ansari, Building Computer Vision Applications Using Artificial Neural Networks: With Step-by-Step Examples in OpenCV and TensorFlow with Python, 1st ed., Apress, NY, 2020.
[4] M. Denuit, D. Hainaut, J., Trufin Effective Statistical Learning Methods for Actuaries III: Neural Networks and Extensions (Springer Actuarial), 1st ed., Springer, Berlin, 2019.
[5] Leordeanu Unsupervised Learning in Space and Time (Advances in Computer Vision and Pattern Recognition), 1st ed., Springer, Berlin, 2020.
[6] M. Soroush, M. Baldea, T. F., Edgar Smart Manufacturing: Concepts and Methods, 1st ed., Elsevier, Amsterdam, 2020.
[7] R. S. Sherratt, N., Dey Low-power Wearable Healthcare Sensors, Mdpi AG, Basel, 2020.
[8] S. K. Sharma, B. Bhushan, N. C. Debnath, Security and Privacy Issues in IoT Devices and Sensor Networks, 1st ed., Academic Press, Cambridge, 2020.
[9] V. Sze, Efficient Processing of Deep Neural Networks (Synthesis Lectures on Computer Architecture), Morgan & Claypool Publishers, Williston, 2020.
[10] W. L. Hamilton, Graph Representation Learning (Synthesis Lectures on Artificial Intelligence and Machine Learning), Morgan & Claypool Publishers, Williston, 2020.
[11] Z. Liu, J. Zhou, Introduction to Graph Neural Networks (Synthesis Lectures on Artificial Intelligence and Machine Le), Morgan & Claypool Publishers, Williston, 2020.
[12] K. O. Stanley, J. Clune, J. Lehman, et al., Designing neural networks through neuroevolution. Nature Machine Intelligence 1 (2019) 24–35. doi: 10.1038/s42256-018-0006-z
[13] O. Omelianenko, Hands-On Neuroevolution with Python: Build high-performing artificial neural network architectures using neuroevolution-based algorithms, Packt Publishing, Birmingham, 2019.
[14] A. Bergel, Agile Artificial Intelligence in Pharo: Implementing Neural Networks, Genetic Algorithms, and Neuroevolution, Apress, New York, 2020.
[15] G. Blokdyk, Neuroevolution of augmenting topologies: Second Edition, 5STARCooks, Fort Wayne, 2018.
[16] G. Blokdyk, Neuroevolution of augmenting topologies: A Complete Guide, CreateSpace Independent Publishing Platform, California, 2017.
[17] D. V. Vargas and J. Murata, Spectrum-Diverse Neuroevolution With Unified Neural Models. IEEE Transactions on Neural Networks and Learning Systems, 28 (8) (2017) 1759-1773. doi: 10.1109/TNNLS.2016.2551748.
[18] P. Pagliuca, N. Milano, S. Nolfi, Maximizing adaptive power in neuroevolution. PLOS ONE 13(7) (2018). doi: 10.1371/journal.pone.0198788
[19] A.W.Y. Khang, J.A.J. Alsayaydeh, S.M. Idrus, J.A.B.M. Gani, W.A. Indra, J.B. Pusppanathan, Resource efficient for hybrid fiber-wireless communications links in access networks with multi response optimization algorithm, ARPN Journal of Engineering and Applied Sciences, vol. 16(1) (2021) 45-50.
[20] J.A.J. Alsayaydeh, A. Aziz, A.I.A. Rahman, S.N.S. Salim, M. Zainon, Z.A. Baharudin, M.I. Abbasi, A.W.Y. Khang, Development of programmable home security using GSM system for early prevention, vol. 16(1) (2021) 88-97.
[21] J.A.J. Alsayaydeh, W.A.Y. Khang, W.A. Indra, J.B. Pusppanathan, V. Shkarupylo, A.K.M. Zakir Hossain, S. Saravanan, Development of vehicle door security using smart tag and fingerprint system, ARPN Journal of Engineering and Applied Sciences, vol. 9(1) (2019) 3108-3114.
[22] J.A.J. Alsayaydeh, W.A.Y. Khang, W.A. Indra, V. Shkarupylo, J. Jayasundar, Development of smart dustbin by using apps, ARPN Journal of Engineering and Applied Sciences, vol. 14(21) (2019) 3703-3711.
[23] A. Gaier, D. Ha, Exploring Weight Agnostic Neural Networks, 2019, URL: https://ai.googleblog.com/2019/08/exploring-weight-agnostic-neural.html
[24] K. O. Stanley, J. Clune, Welcoming the Era of Deep Neuroevolution, 2017, URL: https://eng.uber.com/deep-neuroevolution/
[25] PDDM — New Model-Based Reinforcement Learning Algorithm with Improved Scheduler, 2019, URL: https://habr.com/ru/post/470179/
[26] C. Yoon Deriving Policy Gradients and Implementing reinforce, 2018, URL: https://medium.com/@thechrisyoon/deriving-policy-gradients-and-implementing-reinforce-f887949bd63

[27] P. Winder, Reinforcement Learning: Industrial Applications of Intelligent Agents, 1st ed., O'Reilly Media, California, 2020.
[28] R. S. Sutton, A. G. Barto, Reinforcement Learning, second edition: An Introduction (Adaptive Computation and Machine Learning series), 2nd ed., Bradford Books, Cambridge, 2018.
[29] M. Lapan, Deep Reinforcement Learning Hands-On: Apply modern RL methods to practical problems of chatbots, robotics, discrete optimization, web automation, and more, 2nd ed., Packt Publishing, Birmingham, 2020.
[30] T. Hovorushchenko, I. Lopatto, O. Pavlova, Concept of Intelligent Agent for Verification of Considering the Subject Area Information, Proceedings of the 11th International Conference on Dependable Systems, Services and Technologies, IEEE, Kyiv, Ukraine, 2020, pp. 465-469. doi: 10.1109/DESSERT50317.2020.9125081.
[31] T. Hovorushchenko, O. Pavlova, D. Medzatyi, Ontology-Based Intelligent Agent for Determination of Sufficiency of Metric Information in the Software Requirements Advances. Intelligent Systems and Computing 1020 (2020) 447-460. doi: 10.1007/978-3-030-26474-1_32
[32] R. Gilman, K. Wang, Intuitive RL: Intro to Advantage-Actor-Critic (A2C), 2018, URL: https://hackernoon.com/intuitive-rl-intro-to-advantage-actor-critic-a2c-4ff545978752
[33] J. K.H. Franke, G. Köhler, N. Awad, F., Hutter Neural Architecture Evolution in Deep Reinforcement Learning for Continuous Control NeurIPS 2019 MetaLearn Workshop, 2019, URL: https://arxiv.org/abs/1910.12824
[34] Reinforcement Learning: Deep Q Networks, 2020, URL: https://blogs.oracle.com/datascience/reinforcement-learning-deep-q-networks
[35] S. Risi, J. Togelius, Neuroevolution in Games: State of the Art and Open Challenges, IEEE Transactions on Computational Intelligence and AI in Games, 9(01) (2017) 25-41. doi: 10.1109/TCIAIG.2015.2494596
[36] A. Oliinyk, I. Fedorchenko, A. Stepanenko, M. Rud, D. Goncharenko, Combinatorial optimization problems solving based on evolutionary approach. Proceedings of the 15th International Conference on the Experience of Designing and Application of CAD Systems, IEEE, Lviv, Ukraine, 2019, pp. 41–45. doi: 10.1109/CADSM.2019.8779290
[37] A. Oliinyk, S. Skrupsky, S. Subbotin, Experimental research and analysis of complexity of parallel method for production rules extraction. Automatic Control and Computer Sciences, 52(2) (2018) 89-99. doi: 10.3103/S0146411618020062
[38] A. Oliinyk, S. Skrupsky, S. Subbotin, I. Korobiichuk, Parallel method of production rules extraction based on computational intelligence. Automatic Control and Computer Sciences, 51(4) (2017) 215-223. doi: 10.3103/S0146411617040058
[39] A. Oliinyk, T. Zaiko, S. Subbotin, Training sample reduction based on association rules for neuro-fuzzy networks synthesis. Optical Memory and Neural Networks (Information Optics), 23(2) (2014) 89-95. doi: 10.3103/S1060992X14020039.
[40] Tic-Tac-Toe Endgame Data Set, 1991, URL: https://archive.ics.uci.edu/ml/datasets/Tic-Tac-Toe+Endgame
[41] Taiwanese Bankruptcy Prediction, 2020, URL: data set https://www.kaggle.com/chihfongtsai/taiwanese-bankruptcy-prediction
[42] S. Leoshchenko, A. Oliinyk, S. Subbotin, N. Gorobii, T. Zaiko, Synthesis of artificial neural networks using a modified genetic algorithm, 1st International Workshop on Informatics & Data-Driven Medicine, IDDM 2018, CEUR-WS, 2018, pp. 1-13.
[43] S. Leoshchenko, A. Oliinyk, S. Subbotin, S. Shylo, V. Shkarupylo, Method of Artificial Neural Network Synthesis for Using in Integrated CAD 15th International Conference on the Experience of Designing and Application of CAD Systems, CADSM'19, IEEE, 2019, pp. 1-6. doi: 10.1109/CADSM.2019.8779248
[44] S. Leoshchenko, A. Oliinyk, S. Skrupsky, S. Subbotin, N. Gorobii, V. Shkarupylo, Modification of the Genetic Method for Neuroevolution Synthesis of Neural Network Models for Medical Diagnosis, Second International Workshop on Computer Modeling and Intelligent Systems, CMIS-2019, CEUR-WS, 2019, pp. 143-158.
[45] Resource Monitor, 2011, URL: https://docs.microsoft.com/en-gb/archive/blogs/yongrhee/how-to-pull-the-information-that-resource-monitor-resmon-exe-provides