Reinforcement Learning and Gamification: a Framework for Integrating Intelligent Agents In Retro Video Games

Nemanja Josipovic[1], Aleksandar Petrovic[1] Aleksa Cuk[1],
Milos Antonijevic[1]*, Dejan Jovanovic[2],
and Nebojsa Budimirovic[1]

[1] Singidunum University, Faculty of Informatics and Computing, Danijelova 32, 11000 Belgrade, Serbia
[2] College of Academic Studies "Dositej", Vojvode Putnika 7, 11000 Belgrade, Serbia
office@singidunum.ac.rs
https://singidunum.ac.rs/
[3] Singidunum University, Belgrade, Serbia
nemanja.josipovic.21@singimail.rs, nebojsa.budimirovic.20@singimail.rs, aleksandar.petrovic@singidunum.ac.rs, vgajic@singidunum.ac.rs, acuk@singidunum.ac.rs, mantonijevic@singdinum.ac.rs*, dejanjovanovic@akademijadositej.edu.rs

Abstract. This work explores the benefits that reinforcement learning (RL) and unsupervised learning (UL) have over supervised learning (SL). Using RL and Python simulations are produced that closely mimic those observed in the real world, and an agent is trained using a form of genetic algorithm. The system created can be used to simulate real-world scenarios and offer solutions. Simulations were performed upon a clone of the game "Pong". The system successfully adapted to the game the more it played, and improved its capability to successfully get high-scoring results. The system is flexible and dynamic, being able to adapt to different simulation environments. In the future, more complex examples will be tested on the system, with the goal of real-world scenario simulation.

Keywords: genetic algorithm · artificial intelligence · python · gaming · simulation · reinforcement learning

1 Introduction

The usage of simulations with reinforcement learning (RL) is highly beneficial in situations that strive to replicate a situation present in real-world scenarios. Trying to train an agent to find a solution to a problem directly without the simulated environment could incur far greater costs or even a potential risk to human life. For instance, the usage of self-driving on roads with real drivers, especially in the early stages would produce catastrophic results when looking at it from an economic and safety perspective. Training a reinforcement-learning-based self-driving algorithm with more data does not always lead to better performance, which is a safety concern [17].

N. Bacanin and H. Shaker (eds.), *Proceedings of the 2nd International Conference on Innovation in Information Technology and Business (ICIITB 2024)*,
Advances in Computer Science Research 113,
https://doi.org/10.2991/978-94-6463-482-2_7

Reinforcement learning (RL) is a learning framework that improves a policy in terms of a given objective through interaction with an environment where an agent perceives the state of that environment [20]. The main benefit of RL is having a system that is able to adapt to different environments since it can be trained in real-time and through trial and error come to a conclusion. It also doesn't require labeled data to begin with making it ideal for scenarios where labeled data is scarce.

RL is extensively used in complex, non-linear problems that require dynamic changes in the environment. It is suitable for use in training self-operating vehicles such as cars, drones, and complex robots. In recent years deep RL has made massive progress in constructing agents for playing board games and digital games. Still, most of the works require huge computational resources for a large scale of environmental interactions or self-play for the games [60].

This work uses an entity-component framework for real-time simulations. Entity component system (ECS) is very dynamic since it gives us an opportunity to choose if a certain system will be enabled or not. The framework used is generic and can be applied to a wide array of simulations. In this work, the focus is on simulating a clone of the game "Pong". Metaheuristics will be applied to the agent with the goal of getting the best possible results.

This work has the goal of achieving the following contributions:

- Introduction of a reinforcement learning framework that leverages the genetic algorithm to attain optimized agent performance.
- Simulation of a Pong clone to facilitate agent training and evaluation.
- The application of the genetic algorithm (GA) to improve agent performance.

The remainder of this work has the following structure: Section 2. Describes preceding works that helped inspire and support this research. The methodology is discussed in Section 3 in detail. Simulation outcomes are provided and discussed in Section 4. The work is concluded in Section 5. followed by proposals for future works.

2 Related works

Zero-shot action recognition is the task of recognizing action classes without any visual examples. The challenge is to map the knowledge of seen classes at training time to that of novel unseen classes at test time. A novel model that learns a clustering-based representation of visual-semantic features, optimized with RL was used. The visual-semantic representation helped improve the representation. And the RL yields better, cleaner clusters. The results were remarkable improvements across datasets and tasks over all previous state-of-the-art [24].

Multiple methods of automatic hyperparameter optimization (HPO) methods are tested to avoid manual trial-and-error that is rather time-consuming. All algorithms used were applied to supervised machine learning (ML). Most state-of-the-art systems optimize pre-processing, model selection, and post-processing rather than a single algorithm. For HPO tools, there is a general trade-off between handling many tasks and specializing in a few narrow-focused tasks. With the increase in the number of tasks, the system requires more development time to set up and the search space is larger such that the efficiency might be suboptimal [16].

The optimization of hyperparameter can be simplified as how many function evaluations will be performed on every optimization to select the best hyperparameter in that model. Hybridization of the genetic algorithm with local search methods was tested. The author first investigated the hyperparameter search method focusing on image classification of CIFAR-10 datasets (dataset that consists of 60000 32x32 colored images spread across 10 different classes each with 6000 images. Of the 60000 images, 50000 are for training, and 10000 for testing). Grid search was insufficient, but random search used proved to be able to get to around 80% on the CIFAR-10 leaderboard, while the state-of-the-art accuracy seems to be around 90% [2].

The use metaheuristics on real-world problems is broad and some examples are provided in the following text. Medicine [36, 61, 35, 66, 29, 68, 65, 40], phishing and intrusion detection [62, 1, 52, 32, 51, 13, 31, 10], credit card frauds identification [26, 43, 42], environmental monitoring and pollutants tracking [38, 28, 6, 37, 18], economic problems [30, 34, 56, 23, 47, 45], defect identification in software testing [71], spam emails [64, 49], plant classification[12, 70], predicting green energy production [54, 4, 33], automotive traffic predictions [46, 19], global optimization problems [15, 9], cloud computing problems [48, 7], marine vessel classification and trajectory prediction [44, 58], feature selection [67, 14, 27, 63, 5], enhancing the audit opinion [57], and general optimization of machine learning models [8, 53, 11, 39, 50, 69, 3, 21, 59].

3 Methods

3.1 Entity component system

Entity component system (ECS) is a software development architectural pattern which follows composition over inheritance principle, aiming to achieve a flexible and dynamic way to manage entities in a large-scale real-time applications. By separating data from logic, ECS achieves a modular system, which allows for memory friendly storage of data in contiguous memory areas, and easily parallelized application logic [25]. An entity-component approach aims to decouple the characteristics and functionality of a game entity into smaller, self-constricted components [22]. It gives opportunity for creating a complex virtual environment while maintaining a simple extensible design. Additionally, ECS can facilitate parallel processing and multithreading, allowing for better use of modern hardware and faster execution times. However, it should be noted that the performance gains from ECS are not always guaranteed and depend on the specific application and implementation. Providing the ability to efficiently manage data and optimize program execution, ECSs, as well as the wider field of data-oriented design, have attained popularity in the realms of modeling, simulation, and gaming. Each entity in the ECS consist of components that have a value. The value can be of any data type. All entities are iterated over by each system in order. Those systems primarily read and change component values of those entities in a loop until the simulation is over. For simulations, a clone of Pong is used with the following integrated entities:

- Artificial neural networks (ANN).
- GA.

The system can be expanded with other entities to create any kind of environment for running different simulations.

3.2 Feedforward neural networks

Due to the development of powerful data processing and storage systems, very large amounts of information are available, whether in the form of numbers or in the form of symbols. Therefore, the ability to retrieve information that is known to be present in the data, but that is difficult to extract, becomes crucial.

ANN consists of many neurons that represent local data points that are always in communication with their environment. Each neuron must be able to receive an input signal, process the information, and give an output. In a neural network, neurons must be connected to at least one other neuron, and each connection is assigned a real number known as the weight coefficient. This coefficient signifies the relative importance of the connection compared to others within the network. One of the primary advantages of neural networks is their capacity to use some a priori unknown information hidden in data (but not extract it). This process is called learning or training and is mathematically accomplished by changing the weight coefficients until specific criteria are met. Training processes for AAN can be split into two categories: supervised and unsupervised learning. Supervised learning knows the desired output is known beforehand and thus adjusting the weight coefficients is done so that the desired outputs are as close as possible. Contrary, unsupervised learning has no knowledge of the desired output, instead, the system is provided with a group of facts (patterns) and then left to settle down to a stable state over a certain number of iterations.

The most popular type of neural network is a multi-layer feed-forward (MLF) neural network. Neurons inside the MLF neural network are ordered into layers. The output value (activity) of the ith neuron xi is determined by Eq. 1 and Eq. 2.

$$x_i = f(\xi_i) \tag{1}$$

$$\xi_i = \theta_i + \sum_{j \in r_i^{-1}} \omega_{ij} x_j \tag{2}$$

where Γ_i^{-1} represents a mapping function that assigns for each neuron i a subset that consists of all predecessors of the given neuron i, a connection between the i-th and j-th neuron is characterized by $\omega_P ij$ (weight coefficient), and the i-th neuron by θ_i, ξ_i is the potential of the i-th neuron and function $f(\xi_i)$ is the transfer function (the summation in Eq. 2 is carried out over all neurons j transferring the signal to the i-th neuron). The transfer function is described by Eq. 3.

$$F(\xi) = \frac{1}{1 + exp(-\xi)} \tag{3}$$

The supervised adaptation process varies the threshold coefficients θ_i and weight coefficients ω_{wj} to minimize the sum of the squared differences between the computed and required output values. This is accomplished by minimization of the objective function E:

$$E = \sum_0 \frac{1}{2}(x_0 - \hat{x}_0)^2 \tag{4}$$

where x_0, and $\hat{x}_0$, are vectors composed of the computed and required activities of the output neurons and summation runs over all output neurons 0.

The backpropagation method is used to train modern feedforward neural networks. This algorithm is called back-propagation because the output error propagates from the output layer through the hidden layers to the input layer. There are two sets of data that MLF uses and those are:

- Training set: Used for training the model, the weights in the network begin with an arbitrary value. Each iteration of the whole set is called an epoch. In each epoch, weights are adjusted accordingly in the direction that reduces the error until they converge to the locally optimal set of values. The weights can be updated after the presentation of each training pattern (pattern mode) or after the presentation of all training examples (batch mode).
- Prediction set: Examples are processed one at a time, producing an estimate of the output value based on the input values. The resulting error is used as an estimate of the quality of prediction of the trained network.

Some advantages of MLF neural networks include adaptability without the assistance of the user (learning), non-linear property of the neurons (nonlinearity), construction of a link between input and output values (input-output mapping), and graceful degradation of performance amid increasing amounts of noise (robustness). One of the biggest problems MLF neural networks experience is the fact that ANNs cannot explain their prediction, and they tend to have a lot of weights which in turn can make training ANNs take too long [55].

3.3 GA

In nature, an individual in a population competes with each other for virtual resources like food, shelter, and so on. Also in the same species, individuals compete to attract mates for reproduction. Due to this selection, poorly performing individuals have less chance to survive, and the most adapted or "fit" individuals produce a relatively large number of offspring [41]. GA is inspired by evolution. They apply recombination on a data structure that resembles a chromosome while preserving critical information stored inside. They are applied to a wide array of problems. Populations of typically random chromosomes are generated. Those structures are then evaluated and scores are assigned to each one so that the reproduction of those with a better score is more probable. The 'goodness' of a solution is typically defined with respect to the current population.

The genetic algorithm differs from other search methods in that it searches among a population of points, and works with a coding of parameter set, rather than the parameter values themselves. It also uses objective function information without any gradient information. The transition scheme of the genetic algorithm is 2 probabilistic, whereas traditional methods use gradient information. Because of these features of the GA, they are used as general-purpose optimization algorithms. They also provide means to search irregular space and hence are applied to a variety of function optimization, parameter estimation, and machine learning applications.

Genetic Algorithms	Explanation
Chromosome	Solution
Genes	Part of solution
Locus	Position of gene
Alleles	Values of gene
Phenotype	Decoded solution
Genotype	Encoded solution

Table 1: Explanation of Genetic Algorithm terms

The basic principles of GA involve 3 major steps that are iterated upon:

- Selection – Choosing with increased odds chromosomes that have a higher fitness score thus giving them more chances to "reproduce".
- Crossover – From the selected population pick a part of the chromosomes and pair them up so that their genes can switch places making new chromosomes.
- Mutation – Taking some chromosomes and replacing their genes with new values.

This process continues until the stopping criteria are met. An important characteristic of GA is the coding of variables that describe the problem. The most common coding method is to transform the variables to a binary string or vector; GA performs best when solution vectors are binary. If the problem has more than one variable, multi-variable coding is constructed by concatenating as many single variables coding as the number of variables in the problem. [15]

3.4 Pong clone

The clone version of Pong used in simulations consists of paddles that hit and reflect the ball. On the top, there is only one paddle that always shares the position of the x-axis with the ball, so it can never lose by letting the ball get past it. On the bottom, a number of paddles are instantiated, and they use the ANN to move and try to hit the ball. On the side, there are walls that reflect the x-axis of the ball if they are hit:

$$ball.X = -ball.X \tag{5}$$

When the paddle hits the ball, it reflects it on the y-axis and also changes the velocity of the x-axis to a random number, so that the angle of the ball is always between 0° and 180°:

$$ball.Y = -ball.Y \tag{6}$$

$$ball.X = (rand.random(0-1) - 0.5) * horizontalSpeed \tag{7}$$

This clone of Pong is an excellent base for reinforcement learning because of its simplicity and the good objective function that is built into the game. This is a pilot project that will be expanded in future works.

4 Simulations

In simulations that were run, 100 bottom paddles (agents) were created per generation. The fitness score of the agent went up by 1 if it managed to hit the ball. In the case of an agent missing the ball, the agent was deleted, and the total number of agents "alive" was reduced by one in the current generation. If there were no agents present, the generation would end, and a new one created with the use of GA. Every population was documented based on the total population score (sum of all paddles in a generation) and the agent that had the best score in that generation.

Number of generations simulated	Best population fitness score	Best agent fitness score
4121	856	17
1820	584	16
509	509	10
1883	1217	24
14205	1005	20

Table 2: Experimental resuts.

The following table indicates that the longer training times give better overall results, although a large sample is needed since the GA has a lot of elements

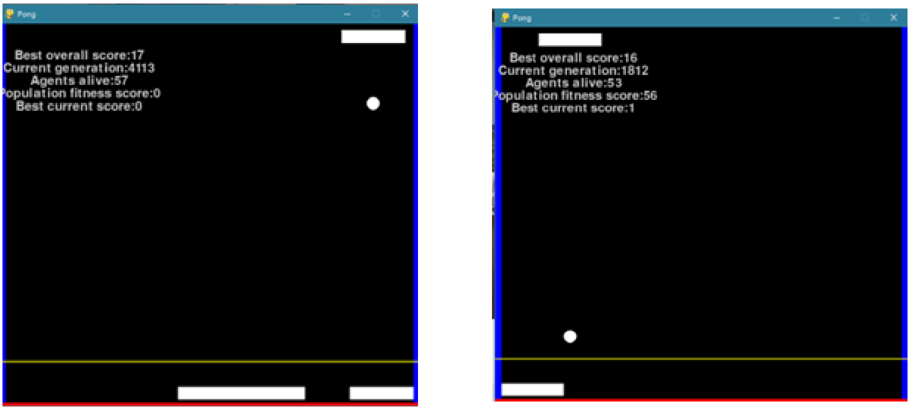

Fig. 1: Examples of simulation screen

of randomness, therefor if the system is trained on an insufficient amount of populations it could produce unsatisfactory results.

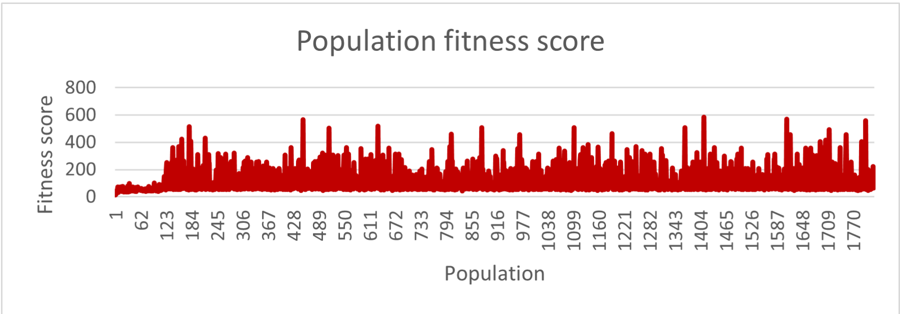

Fig. 2: Graph of a single simulation run showing fitness score increasing with longer training time

5 Conclusion

The system made for this article seems to fulfill its goal of training the agent to play a clone of Pong while incorporating ANN and GA. It can also be used for much more complex simulations by replacing existing entities and components, making it very flexible. In the future, this project will be expanded upon, increasing in complexity while trying to simulate some examples that could lead to resolving practical problems people face. Based on simulations that were run, it can be observed that the system improves steadily over time, but the speed of improvement varies since the GA has a lot of randomness. The system doesn't yet have an option to save the training progress, so every test was run in a single time interval, which limits the number of generations that can be simulated because of the hardware constraints. The speed at which a generation is tested can be increased by increasing the speed of the ball, but the visualization aspect of the simulations then becomes redundant. In the future, an option to save progress and a better visualization will be done to better reflect a more complex environment that will be worked upon. The goal is to get as close as possible to recreating a real-world scenario inside the environment and produce a successful solution to a problem that people are facing.

References

1. AlHosni, N., Jovanovic, L., Antonijevic, M., Bukumira, M., Zivkovic, M., Strumberger, I., Mani, J.P., Bacanin, N.: The xgboost model for network intrusion detection boosted by enhanced sine cosine algorithm. In: International Conference on Image Processing and Capsule Networks. pp. 213–228. Springer (2022)
2. Aszemi, N.M., Dominic, P.: Hyperparameter optimization in convolutional neural network using genetic algorithms. International Journal of Advanced Computer Science and Applications **10**(6) (2019)
3. Bacanin, N., Antonijevic, M., Vukobrat, N., Bezdan, T., Zivkovic, M.: Enhanced seagull optimization algorithm adapted for artificial neural network training. In: ICT with Intelligent Applications: Proceedings of ICTIS 2021, Volume 1. pp. 753–761. Springer (2022)
4. Bacanin, N., Jovanovic, L., Zivkovic, M., Kandasamy, V., Antonijevic, M., Deveci, M., Strumberger, I.: Multivariate energy forecasting via metaheuristic tuned long-short term memory and gated recurrent unit neural networks. Information Sciences **642**, 119122 (2023)
5. Bacanin, N., Petrovic, A., Zivkovic, M., Bezdan, T., Chhabra, A.: Enhanced salp swarm algorithm for feature selection. In: International conference on intelligent and fuzzy systems. pp. 483–491. Springer International Publishing Cham (2021)
6. Bacanin, N., Sarac, M., Budimirovic, N., Zivkovic, M., AlZubi, A.A., Bashir, A.K.: Smart wireless health care system using graph lstm pollution prediction and dragonfly node localization. Sustainable Computing: Informatics and Systems **35**, 100711 (2022)
7. Bacanin, N., Simic, V., Zivkovic, M., Alrasheedi, M., Petrovic, A.: Cloud computing load prediction by decomposition reinforced attention long short-term memory network optimized by modified particle swarm optimization algorithm. Annals of Operations Research pp. 1–34 (2023)
8. Bacanin, N., Stoean, C., Zivkovic, M., Jovanovic, D., Antonijevic, M., Mladenovic, D.: Multi-swarm algorithm for extreme learning machine optimization. Sensors **22**(11), 4204 (2022)
9. Bacanin, N., Stoean, R., Zivkovic, M., Petrovic, A., Rashid, T.A., Bezdan, T.: Performance of a novel chaotic firefly algorithm with enhanced exploration for tackling global optimization problems: Application for dropout regularization. Mathematics **9**(21), 2705 (2021)

10. Bacanin, N., Zivkovic, M., Hajdarevic, Z., Petrovic, A., Budimirovic, N., Antonijevic, M., Strumberger, I.: Detection of botnet using extreme learning machine tuned by enhanced sine cosine algorithm. In: International Conference on Advances and Applications of Artificial Intelligence and Machine Learning. pp. 125–137. Springer (2022)
11. Bacanin, N., Zivkovic, M., Salb, M., Strumberger, I., Chhabra, A.: Convolutional neural networks hyperparameters optimization using sine cosine algorithm. In: Sentimental Analysis and Deep Learning: Proceedings of ICSADL 2021. pp. 863–878. Springer (2022)
12. Bacanin, N., Zivkovic, M., Sarac, M., Petrovic, A., Strumberger, I., Antonijevic, M., Petrovic, A., Venkatachalam, K.: A novel multiswarm firefly algorithm: An application for plant classification. In: International Conference on Intelligent and Fuzzy Systems. pp. 1007–1016. Springer (2022)
13. Bacanin, N., Zivkovic, M., Stoean, C., Antonijevic, M., Janicijevic, S., Sarac, M., Strumberger, I.: Application of natural language processing and machine learning boosted with swarm intelligence for spam email filtering. Mathematics **10**(22), 4173 (2022)
14. Bezdan, T., Cvetnic, D., Gajic, L., Zivkovic, M., Strumberger, I., Bacanin, N.: Feature selection by firefly algorithm with improved initialization strategy. In: 7th conference on the engineering of computer based systems. pp. 1–8 (2021)
15. Bezdan, T., Petrovic, A., Zivkovic, M., Strumberger, I., Devi, V.K., Bacanin, N.: Current best opposition-based learning salp swarm algorithm for global numerical optimization. In: 2021 Zooming Innovation in Consumer Technologies Conference (ZINC). pp. 5–10. IEEE (2021)
16. Bischl, B., Binder, M., Lang, M., Pielok, T., Richter, J., Coors, S., Thomas, J., Ullmann, T., Becker, M., Boulesteix, A.L., et al.: Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **13**(2), e1484 (2023)
17. Cao, Z., Jiang, K., Zhou, W., Xu, S., Peng, H., Yang, D.: Continuous improvement of self-driving cars using dynamic confidence-aware reinforcement learning. Nature Machine Intelligence **5**(2), 145–158 (2023)
18. Damaševičius, R., Jovanovic, L., Petrovic, A., Zivkovic, M., Bacanin, N., Jovanovic, D., Antonijevic, M.: Decomposition aided attention-based recurrent neural networks for multistep ahead time-series forecasting of renewable power generation. PeerJ Computer Science **10** (2024)
19. Dogandzic, T., Petrovic, A., Jovanovic, L., Bacanin, N., Jovanovic, A., Zivkovic, M.: Speeding classification by a deep learning audio analysis system optimized by the reptile search algorithm. In: International Joint Conference on Advances in Computational Intelligence. pp. 73–88. Springer (2022)
20. Doya, K., Friston, K., Sugiyama, M., Tenenbaum, J.: Neural networks special issue on artificial intelligence and brain science (2022)
21. Gajic, L., Cvetnic, D., Zivkovic, M., Bezdan, T., Bacanin, N., Milosevic, S.: Multilayer perceptron training using hybridized bat algorithm. In: Computational Vision and Bio-Inspired Computing: ICCVBIC 2020. pp. 689–705. Springer (2021)
22. Garcia, F.E., de Almeida Neris, V.P.: A data-driven entity-component approach to develop universally accessible games. In: Universal Access in Human-Computer Interaction. Universal Access to Information and Knowledge: 8th International Conference, UAHCI 2014, Held as Part of HCI International 2014, Heraklion, Crete, Greece, June 22-27, 2014, Proceedings, Part II 8. pp. 537–548. Springer (2014)

23. Golubovic, S., Petrovic, A., Bozovic, A., Antonijevic, M., Zivkovic, M., Bacanin, N.: Gold price forecast using variational mode decomposition-aided long short-term model tuned by modified whale optimization algorithm. In: International Conference on Data Intelligence and Cognitive Informatics. pp. 69–83. Springer (2023)
24. Gowda, S.N., Sevilla-Lara, L., Keller, F., Rohrbach, M.: Claster: clustering with reinforcement learning for zero-shot action recognition. In: European Conference on Computer Vision. pp. 187–203. Springer (2022)
25. Härkönen, T.: Advantages and implementation of entity-component-systems (2019)
26. Jovanovic, D., Antonijevic, M., Stankovic, M., Zivkovic, M., Tanaskovic, M., Bacanin, N.: Tuning machine learning models using a group search firefly algorithm for credit card fraud detection. Mathematics **10**(13), 2272 (2022)
27. Jovanovic, D., Marjanovic, M., Antonijevic, M., Zivkovic, M., Budimirovic, N., Bacanin, N.: Feature selection by improved sand cat swarm optimizer for intrusion detection. In: 2022 International Conference on Artificial Intelligence in Everything (AIE). pp. 685–690. IEEE (2022)
28. Jovanovic, G., Perisic, M., Bacanin, N., Zivkovic, M., Stanisic, S., Strumberger, I., Alimpic, F., Stojic, A.: Potential of coupling metaheuristics-optimized-xgboost and shap in revealing pahs environmental fate. Toxics **11**(4), 394 (2023)
29. Jovanovic, L., Bacanin, N., Zivkovic, M., Antonijevic, M., Petrovic, A., Zivkovic, T.: Anomaly detection in ecg using recurrent networks optimized by modified metaheuristic algorithm. In: 2023 31st Telecommunications Forum (TELFOR). pp. 1–4. IEEE (2023)
30. Jovanovic, L., Jovanovic, D., Bacanin, N., Jovancai Stakic, A., Antonijevic, M., Magd, H., Thirumalaisamy, R., Zivkovic, M.: Multi-step crude oil price prediction based on lstm approach tuned by salp swarm algorithm with disputation operator. Sustainability **14**(21), 14616 (2022)
31. Jovanovic, L., Jovanovic, D., Antonijevic, M., Nikolic, B., Bacanin, N., Zivkovic, M., Strumberger, I.: Improving phishing website detection using a hybrid two-level framework for feature selection and xgboost tuning. Journal of Web Engineering **22**(3), 543–574 (2023)
32. Jovanovic, L., Jovanovic, D., Antonijevic, M., Zivkovic, M., Budimirovic, N., Strumberger, I., Bacanin, N.: The xgboost tuning by improved firefly algorithm for network intrusion detection. In: 2022 24th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC). pp. 268–275. IEEE (2022)
33. Jovanovic, L., Kljajic, M., Petrovic, A., Mizdrakovic, V., Zivkovic, M., Bacanin, N.: Modified teaching-learning-based algorithm tuned long short-term memory for household energy consumption forecasting. In: International Conference on World-wide Computing and Its Applications. pp. 347–362. Springer Nature Singapore (2023)
34. Jovanovic, L., Milutinovic, N., Gajevic, M., Krstovic, J., Rashid, T.A., Petrovic, A.: Sine cosine algorithm for simple recurrent neural network tuning for stock market prediction. In: 2022 30th Telecommunications Forum (TELFOR). pp. 1–4. IEEE (2022)
35. Jovanovic, L., Petrovic, A., Zivkovic, T., Antonijevic, M., Bacanin, N., Zivkovic, M.: Exploring the potential of generative adversarial networks for synthetic medical data generation. In: 2023 31st Telecommunications Forum (TELFOR). pp. 1–4. IEEE (2023)

36. Jovanovic, L., Zivkovic, M., Antonijevic, M., Jovanovic, D., Ivanovic, M., Jassim, H.S.: An emperor penguin optimizer application for medical diagnostics. In: 2022 IEEE Zooming Innovation in Consumer Technologies Conference (ZINC). pp. 191–196. IEEE (2022)
37. Mani, J., Shaker, H., Jovanovic, L., et al.: Sunspot occurrence forecasting with metaheuristic optimized recurrent neural networks. Theoretical and Applied Computational Intelligence **1**(1), 15–26 (2023)
38. Markovic, I., Krzanovic, J., Jovanovic, L., Toskovic, A., Bacanin, N., Petrovic, A., Zivkovic, M.: Flood prediction based on recurrent neural network time series classification boosted by modified metaheuristic optimization. In: International Conference on Advances in Data-driven Computing and Intelligent Systems. pp. 289–303. Springer (2023)
39. Milutinovic, N., Gajevic, M., Krstovic, J., Petrovic, A., Bacanin, N., Antonijevic, M.: Performance of arithmetic optimization algorithm for elm tuning applied to iot security. In: 2022 30th Telecommunications Forum (TELFOR). pp. 1–4. IEEE (2022)
40. Minic, A., Jovanovic, L., Bacanin, N., Stoean, C., Zivkovic, M., Spalevic, P., Petrovic, A., Dobrojevic, M., Stoean, R.: Applying recurrent neural networks for anomaly detection in electrocardiogram sensor data. Sensors **23**(24), 9878 (2023)
41. Mirjalili, S.: Evolutionary algorithms and neural networks. Studies in computational intelligence **780** (2019)
42. Petrovic, A., Antonijevic, M., Strumberger, I., Jovanovic, L., Savanovic, N., Janicijevic, S.: The xgboost approach tuned by tlb metaheuristics for fraud detection. In: Proceedings of the 1st International Conference on Innovation in Information Technology and Business (ICIITB 2022). vol. 104, p. 219. Springer Nature (2023)
43. Petrovic, A., Bacanin, N., Zivkovic, M., Marjanovic, M., Antonijevic, M., Strumberger, I.: The adaboost approach tuned by firefly metaheuristics for fraud detection. In: 2022 IEEE World Conference on Applied Intelligence and Computing (AIC). pp. 834–839. IEEE (2022)
44. Petrovic, A., Damaševičius, R., Jovanovic, L., Toskovic, A., Simic, V., Bacanin, N., Zivkovic, M., Spalević, P.: Marine vessel classification and multivariate trajectories forecasting using metaheuristics-optimized extreme gradient boosting and recurrent neural networks. Applied Sciences **13**(16), 9181 (2023)
45. Petrovic, A., Jovanovic, L., Zivkovic, M., Bacanin, N., Budimirovic, N., Marjanovic, M.: Forecasting bitcoin price by tuned long short term memory model. In: 1st International conference on innovation in information technology and business (ICIITB 2022). pp. 187–202. Atlantis Press (2023)
46. Petrovic, A., Strumberger, I., Antonijevic, M., Jovanovic, D., Mladenovic, D., Chabbra, A.: Firefly-xgboost approach for pedestrian detection. In: 2022 IEEE Zooming Innovation in Consumer Technologies Conference (ZINC). pp. 197–202. IEEE (2022)
47. Petrovic, A., Strumberger, I., Bezdan, T., Jassim, H.S., Nassor, S.S.: Cryptocurrency price prediction by using hybrid machine learning and beetle antennae search approach. In: 2021 29th Telecommunications Forum (TELFOR). pp. 1–4. IEEE (2021)
48. Predić, B., Jovanovic, L., Simic, V., Bacanin, N., Zivkovic, M., Spalevic, P., Budimirovic, N., Dobrojevic, M.: Cloud-load forecasting via decomposition-aided attention recurrent neural network tuned by modified particle swarm optimization. Complex & Intelligent Systems pp. 1–21 (2023)

49. Radomirovic, B., Petrovic, A., Zivkovic, M., Njegus, A., Budimirovic, N., Bacanin, N.: Efficient spam email classification logistic regression model trained by modified social network search algorithm. In: Computational Intelligence and Blockchain in Complex Systems, pp. 39–55. Elsevier (2024)
50. Salb, M., Bacanin, N., Zivkovic, M., Antonijevic, M., Marjanovic, M., Strumberger, I.: Extreme learning machine tuning by original sine cosine algorithm. In: 2022 IEEE World Conference on Applied Intelligence and Computing (AIC). pp. 143–148. IEEE (2022)
51. Salb, M., Jovanovic, L., Bacanin, N., Antonijevic, M., Zivkovic, M., Budimirovic, N., Abualigah, L.: Enhancing internet of things network security using hybrid cnn and xgboost model tuned via modified reptile search algorithm. Applied Sciences **13**(23), 12687 (2023)
52. Savanović, N., Toskovic, A., Petrovic, A., Zivkovic, M., Damaševičius, R., Jovanovic, L., Bacanin, N., Nikolic, B.: Intrusion detection in healthcare 4.0 internet of things systems via metaheuristics optimized machine learning. Sustainability **15**(16), 12563 (2023)
53. Stankovic, M., Gavrilovic, J., Jovanovic, D., Zivkovic, M., Antonijevic, M., Bacanin, N., Stankovic, M.: Tuning multi-layer perceptron by hybridized arithmetic optimization algorithm for healthcare 4.0. Procedia Computer Science **215**, 51–60 (2022)
54. Stoean, C., Zivkovic, M., Bozovic, A., Bacanin, N., Strulak-Wójcikiewicz, R., Antonijevic, M., Stoean, R.: Metaheuristic-based hyperparameter tuning for recurrent deep learning: application to the prediction of solar energy generation. Axioms **12**(3), 266 (2023)
55. Svozil, D., Kvasnicka, V., Pospichal, J.: Introduction to multi-layer feed-forward neural networks. Chemometrics and intelligent laboratory systems **39**(1), 43–62 (1997)
56. Todorovic, M., Petrovic, A., Toskovic, A., Zivkovic, M., Jovanovic, L., Bacanin, N.: Multivariate bitcoin price prediction based on lstm tuned by hybrid reptile search algorithm. In: 2023 16th International Conference on Advanced Technologies, Systems and Services in Telecommunications (TELSIKS). pp. 195–198. IEEE (2023)
57. Todorovic, M., Stanisic, N., Zivkovic, M., Bacanin, N., Simic, V., Tirkolaee, E.B.: Improving audit opinion prediction accuracy using metaheuristics-tuned xgboost algorithm with interpretable results through shap value analysis. Applied Soft Computing **149**, 110955 (2023)
58. Toskovic, A., Petrovic, A., Jovanovic, L., Bacanin, N., Zivkovic, M., Dobrojevic, M.: Marine vessel trajectory forecasting using long short-term memory neural networks optimized via modified metaheuristic algorithm. In: International Conference on Trends in Sustainable Computing and Machine Intelligence. pp. 51–66. Springer (2023)
59. Vesic, A., Marjanovic, M., Petrovic, A., Strumberger, I., Tuba, E., Bezdan, T.: Optimizing extreme learning machine by animal migration optimization. In: 2022 IEEE Zooming Innovation in Consumer Technologies Conference (ZINC). pp. 261–266. IEEE (2022)
60. Zhao, D., Tu, S., Xu, L.: Efficient learning for alphazero via path consistency. In: International Conference on Machine Learning. pp. 26971–26981. PMLR (2022)
61. Zivkovic, M., Bacanin, N., Antonijevic, M., Nikolic, B., Kvascev, G., Marjanovic, M., Savanovic, N.: Hybrid cnn and xgboost model tuned by modified arithmetic optimization algorithm for covid-19 early diagnostics from x-ray images. Electronics **11**(22), 3798 (2022)

62. Zivkovic, M., Jovanovic, L., Ivanovic, M., Bacanin, N., Strumberger, I., Joseph, P.M.: Xgboost hyperparameters tuning by fitness-dependent optimizer for network intrusion detection. In: Communication and intelligent systems: Proceedings of ICCIS 2021, pp. 947–962. Springer (2022)
63. Zivkovic, M., Jovanovic, L., Ivanovic, M., Krdzic, A., Bacanin, N., Strumberger, I.: Feature selection using modified sine cosine algorithm with covid-19 dataset. In: Evolutionary computing and mobile sustainable networks: Proceedings of ICECMSN 2021, pp. 15–31. Springer (2022)
64. Zivkovic, M., Petrovic, A., Bacanin, N., Djuric, M., Vesic, A., Strumberger, I., Marjanovic, M.: Training logistic regression model by hybridized multi-verse optimizer for spam email classification. In: Proceedings of International Conference on Data Science and Applications: ICDSA 2022, Volume 2. pp. 507–520. Springer (2023)
65. Zivkovic, M., Petrovic, A., Bacanin, N., Milosevic, S., Veljic, V., Vesic, A.: The covid-19 images classification by mobilenetv3 and enhanced sine cosine metaheuristics. In: Mobile Computing and Sustainable Informatics: Proceedings of ICMCSI 2022, pp. 937–950. Springer Nature Singapore Singapore (2022)
66. Zivkovic, M., Petrovic, A., Venkatachalam, K., Strumberger, I., Jassim, H.S., Bacanin, N.: Novel chaotic best firefly algorithm: Covid-19 fake news detection application. In: Advances in Swarm Intelligence: Variations and Adaptations for Optimization Problems, pp. 285–305. Springer International Publishing Cham (2022)
67. Zivkovic, M., Stoean, C., Chhabra, A., Budimirovic, N., Petrovic, A., Bacanin, N.: Novel improved salp swarm algorithm: An application for feature selection. Sensors **22**(5), 1711 (2022)
68. Zivkovic, M., Stoean, C., Petrovic, A., Bacanin, N., Strumberger, I., Zivkovic, T.: A novel method for covid-19 pandemic information fake news detection based on the arithmetic optimization algorithm. In: 2021 23rd international symposium on symbolic and numeric algorithms for scientific computing (SYNASC). pp. 259–266. IEEE (2021)
69. Zivkovic, M., Vesic, A., Bacanin, N., Strumberger, I., Antonijevic, M., Jovanovic, L., Marjanovic, M.: An improved animal migration optimization approach for extreme learning machine tuning. In: International Conference on Intelligent and Fuzzy Systems. pp. 3–13. Springer (2022)
70. Živković, M., et al.: Plant classification using firefly algorithm and support vector machine. In: Sinteza 2022-International Scientific Conference on Information Technology and Data Related Research. pp. 443–448. Singidunum University (2022)
71. Zivkovic, T., Nikolic, B., Simic, V., Pamucar, D., Bacanin, N.: Software defects prediction by metaheuristics tuned extreme gradient boosting and analysis based on shapley additive explanations. Applied Soft Computing **146**, 110659 (2023)