ISSN 2224-087X. Електроніка та інформаційні технології. 2024. Випуск 25. С. 93–101
Electronics and information technologies. 2024. Issue 25. P. 93–101

UDC 004.4 DOI: https://doi.org/10.30970/eli.25.8

MUTATION INFLUENCE AND PROBABILITY IN NEURO-GENETIC EVOLUTION AND NEAT. STRATEGY FOR ADJUSTING DYNAMIC MUTATION PARAMETERS IN RESPONSE TO FITNESS STAGNATION

V. Pretsel, R. Shuvar

Ivan Franko National University of Lviv,
50 Drahomanova St., UA-79005 Lviv, Ukraine
v.pretsel@gmail.com, roman.shuvar@lnu.edu.ua

Optimization tasks are one of the hot nowadays challenges in science and techniques which can be resolved efficiently by evolutionary algorithms. Although they have proved their efficiency in neuroevolution they have to compete with other methods of neural network training. They began to show their advantages with the development of computing technologies and proved themselves well with problems where more traditional methods, such as gradient descent, cannot be applied. It would be great to investigate already existing neuroevolutionary methods or develop new ones.

A different configuration of training parameters can have a significant impact on the result. Thus the investigation in this direction is actual. Inspecting the influence of training parameters and various strategies for tuning these parameters we found interesting and actual the challenge of automatic adjustment of training parameters during training.

The main goal of the paper is an investigation of the training parameter's influences on the training results of two popular neuroevolution methods: the neuro-genetic evolution with a static topology of the neural network and the neuroevolution method of augmenting topologies - NEAT. A mutation step (mutation strength) is a studied parameter for the first method, and for the NEAT we have two parameters: a mutation step and a probability of structural mutations. The proposed strategy for the automatic adjustment of the studied parameters aims to solve the problem of reaching a local extremum, which didn't allow to achieve the best possible results of training.

All experiments were performed on the same task of a rocket controlling the flight from the Earth to the Moon, which requires enhanced accuracy of the neural network for successful training. The results of the training were analyzed according to the following indicators: average fitness value in the generation, best fitness value in the generation, and percentage of successful individuals in the generation. The conducted experiments have demonstrated that the requirements with high accuracy force usage of lower values of the mutation strength for traditional genetic algorithms, but this creates risks of getting stuck in local extrema or convergence to a suboptimal solution.

The NEAT experiments have shown that the task's requirements for high accuracy didn't allow the best results to be achieved. The proposed strategy for decreasing the mutation strength and the probability of mutations was able to reduce the risk of convergence to a suboptimal solution in both methods. Generally, this approach was able to justify itself. Further investigations of this strategy on par with already existing approaches might reveal other advantages which should be investigated more deeply.

Keywords: genetic algorithms, neuroevolution, artificial neural networks, NEAT

Introduction

There are a lot of developed methods which offer different approaches to artificial neural network training. The neuroevolution methods involve training the neural networks by genetic algorithms updating the edge weights of neural networks or even their structure. In some scenarios, neuroevolution methods are being used effectively with reinforcement learning to resolve the problems related to artificial life or evolutionary robotics [1]. It's also been said, that some reinforcement learning problems, might be resolved by neuroevolution methods that might even outperform traditional RL approaches [2].

Neuroevolution, at its core, is an optimization technique that uses a probabilistic approach to evolving neural network architectures and weights [3-4]. Unlike conventional methods like supervised learning, neuroevolution operates in a more dynamic and self-adapting manner, making it particularly suitable for tasks where the optimal structure of the neural network is not known beforehand or may change over time. This adaptability is particularly advantageous in complex and dynamic environments, where traditional approaches may struggle to converge efficiently.

Traditional neuro-genetic evolution uses basic principles of genetic algorithms to change only weights in structures of predefined neural networks [5]. NeuroEvolution of Augmenting Topologies (NEAT) methods extend this by introducing new mechanisms of evolving ANN structure simultaneously with its weights [6]. This innovation addresses one of the key challenges in neuroevolution, allowing the algorithm to explore a broader solution space and discover more sophisticated architectures.

As with machine learning methods, the success of neuroevolution depends on a careful selection of parameters. Parameters such as mutation rates and crossover rates play a pivotal role in shaping the evolutionary process and ultimately influencing the performance of the evolved neural networks. Striking the right balance is crucial, as overly aggressive mutations may hinder convergence, while conservative settings may impede the exploration of diverse solutions.

To optimize the performance of neuroevolution, researchers have explored strategies to dynamically adjust these parameters during training. Adaptive methods, such as those inspired by the concept of self-adaptation in evolutionary algorithms, have shown promise in achieving a more fine-tuned and efficient convergence [7-8]. These dynamic strategies contribute to the adaptability of neuroevolution, allowing it to navigate the evolving landscape of problem complexity.

While there exist some methods of dynamic adjusting of training parameters, they lack considering fitness values during training, and also usually are applied to traditional neuroevolution, not considering other methods. In this work, we tend to analyze how mutation parameters affect the training process for both neuro-genetic evolution and NEAT, and based on that, propose a strategy for dynamically changing these parameters during training.

Methods and materials

Running a bunch of experiments for each settings configuration we support the investigation of the dependence of training results on parameters. Keeping a certain number of results for the same experimental conditions will allow us to get more accurate results.

The chosen task for training is defined as the Moon's rotation around the Earth in a 2-dimensional simulation implemented with the Unity game engine [9], a neural network needs to control the rocket flight from the Earth to the Moon. The tested population is more fitted when the rocket is coming near the Moon which could be defined as the fitness function as $f = 1/d$, where d is the distance between the rocket and the Moon. During a population selection for the

next generation, we count on the highest value of fitness function because it means that the rocket is the closest to the Moon. If the rocket reached the Moon the individual would be considered successful. The distance from Earth to the Moon and the size of the Moon in the calculation are close to real and require high precision which could be provided by low values of mutation strength and probability.

The architecture of the trained neural network has 8 input nodes and 3 output nodes that will be considered for further analysis: the fitness of the best individual in the population, the average fitness of the population, the percentage of successful individuals in the population, and the number of epochs. During multiple simulations for each training configuration, we have counted the training that failed due to hitting local extrema.

For both neuro-genetic evolution and NEAT we use real number representation for chromosomes. It means that both methods use uniform perturb mutation: the weights of the neural network edges are changed by adding a random number to its previous value. Controlling the limits on how much this value can be perturbed, we use the mutation strength, also known as the mutation step, and define it as a fraction of max possible weight value.

Parameter settings

For the neuro-genetic algorithm, we use the following configuration: size of population 30, crossover probability 50%, probability of gene mutation 5%, mutation strength 0.05. We use a neural network with 1 hidden layer of 7 nodes for this method.

For NEAT we use a configuration that overall is similar as it is in the original work [6]. The population's size equals 100. The coefficients for measuring compatibility were $c_1 = 1.0$, $c_2 = 1.0$, and $c_3 = 0.3$, the compatibility threshold was equal to $\delta_t = 3.0$. The champion of each species with more than five networks was copied into the next generation unchanged. There was a 30% chance of a genome having its connection weights mutated, and each weight had a chance of mutation 5%, in which case each weight had a 90% chance of being uniformly perturbed and a 10% chance of being assigned a new random value. Mutation strength was 0.005. In each generation, 25% of offspring resulted from mutation without crossover. The interspecies mating rate was 0.1%. The probability of adding a new node was 1.5% and the probability of a new link mutation was 2.5%. For all experiments, we start to evolve neural networks from structures that don't have hidden nodes.

For both methods, we used hyperbolic tangent as a transfer function for all nodes. All these parameters are taken as a basis and only the investigated parameters are changed in different experiments.

Dynamic mutation parameters strategies

The main idea of changing parameters during training is detection of the fitness stagnation. Fitness stagnation can be defined as the absence of improvement in the best or average fitness in the population for a certain number of generations in a row. When stagnation is detected, we will decrease mutation parameters' values. So, starting with parameters that have higher values, we will use the exploration approach, and decreasing it we will gradually switch to exploitation.

For neuro-genetic algorithms, we will decrease the mutation strength and/or probability, while other parameters remain constant. Later experiments have shown, that for neuro-genetic algorithms it makes sense to decrease parameter values after just a few generations without fitness improvement, otherwise decreasing it after a higher number of epochs can prolong the training process without much positive effect.

Similar to a NEAT method, but first we decrease the probability of structural mutations: those are node and connection mutations. Along with this, we decrease random weight mutations, since changing weights' values to those that completely differ from previous ones can harm not only individual but also species' effectiveness. By decreasing the chances of structural mutations we stick more to weight mutations, and after the structural mutations chances reach a threshold of 1E-4, we stop to execute them. Instead, we perform only weight mutations for evolved structures, and with the same principle as for the neuro-genetic algorithm, start to decrease the mutation strength. The picked number of epochs for stagnation detection here is greater than in neuro-genetic algorithms since structural mutations require more time to search for an optimal solution.

Results and analysis

In brief, most of the results are expected but the introduced strategy shows a significant increase in the required epochs number, so may not be justified for every task. But in this task, it shows good results since it achieves its main goal to reduce the risk of falling into local extrema.

Investigation of mutation strength for neuro-genetic evolution. The charts in Figure 1 and Figure 2 show the training process with 0.05 and 0.001 mutation strength, all other parameters remain the same. It's clear that a lower mutation strength demonstrates better results: while a high strength value shows only 0.4 average fitness value and 30% of successful individuals, a low mutation strength allows one to reach 1 fitness and 100% successful individuals. Figure 3 shows how mutation strength affects the results of training. At the same moment, for all strength configurations training reaches its best results nearly in 20 epochs. Even though we received better results, for the low values of mutation strength, it needs to be noted, that for experiments with a low mutation strength the risk of failing the experiment appears: for 0.0025 strength 10% of experiments failed and for 0.001 strength 30% of experiments were failed resulting in significantly lower results than average. It can be explained by the problem of falling into local extrema, or in other words finding a suboptimal solution, which is the reason why training gets stuck at a certain point and doesn't improve after that.

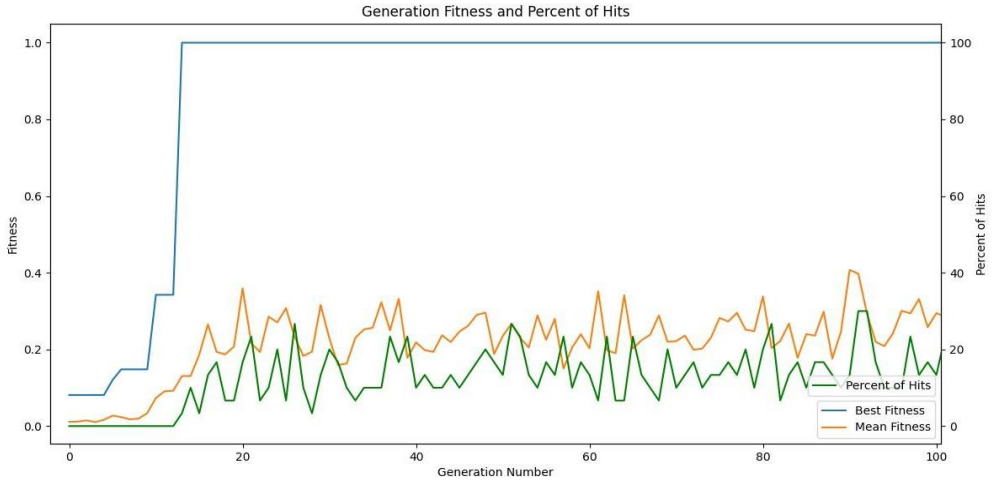

Fig. 1. Training process for neuro-genetic evolution with a static mutation strength 0.05

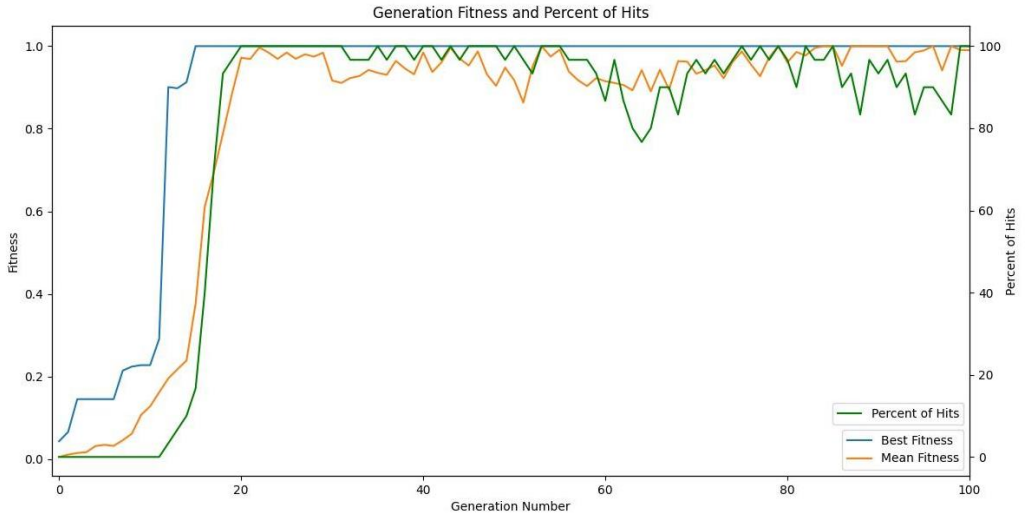

Fig. 2. Training process for neuro-genetic evolution with a static mutation strength 0.001

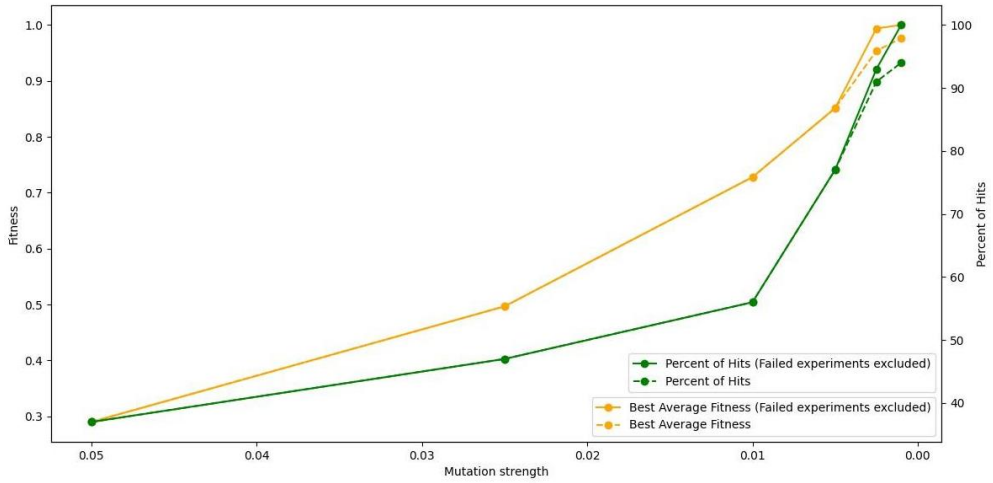

Fig. 3. Dependence of successful individuals (the percentage of hits) and best reached average fitness in population on used mutation strength. Dashed lines consider experiments that failed due to getting stuck in a local extrema

Investigation of dynamic mutation strength strategy for neuro-genetic evolution. In this experiment, the mutation strength starts evolving from 0.05 and decreases this value by half for every 5 generations in a row that haven't shown improving the average fitness value. Figure 4 depicts the training process with this configuration. Noticeably, the required amount of epochs increased significantly, nearly to 150 epochs, but all experiments were successful: reached 1 average fitness and 100% of hits in the population, also no failed experiments due to falling to local extrema were detected.

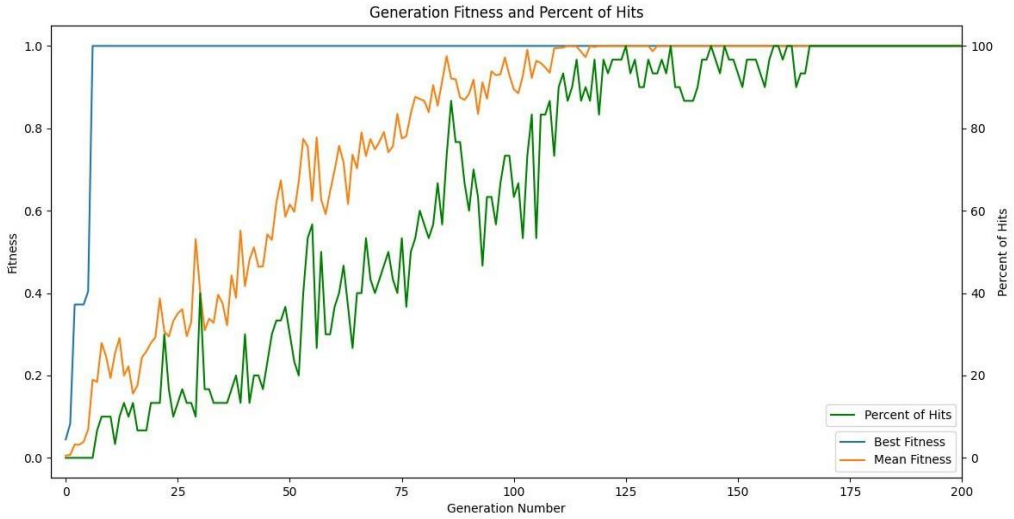

Fig. 4. Training process for neuro-genetic evolution with a dynamic mutation strength

We consider this result as successful, although the approach of decreasing the parameters with every generation [7] may show the same result with fewer generations needed. But here we should admit the advantage that in our approach we don't need to know in advance the required number of epochs and minimum value of the parameter.

Investigation of mutation strength for NEAT. Using a similar approach as in neuro-genetic evolution, we tried to investigate how weight mutation strength affects the training results. One of the best results was achieved with a configuration of 0.005 mutation strength, 10% probability of random weight and 90% probability of perturbed weight mutation. Other parameters remain as default. The results are shown in Figure 5. We observe that only 86 percent of successful individuals can be reached and 0.89 of average fitness at best. On average, 64% of successful individuals and 0.64 of average fitness were observed. This is not the best result can be explained by the fact that introducing new structural elements to the neural network complicates the process of weight tuning.

Investigation of dynamic mutation strength strategy for NEAT method. For the dynamic approach in NEAT, we decrease the parameter values for each 20 generations in a row without average fitness improvement. We start with a mutation strength of 0.05, other parameters are default. Similar to experiments with neuro-genetic evolution, the number of required epochs slightly increased, but the desirable results were finally achieved: we could achieve 100% successful individuals and 1 fitness value for 80% of the experiments, but the rest of experiments still demonstrated the problem of falling into local extrema. An example of the training process is depicted in Figure 6.

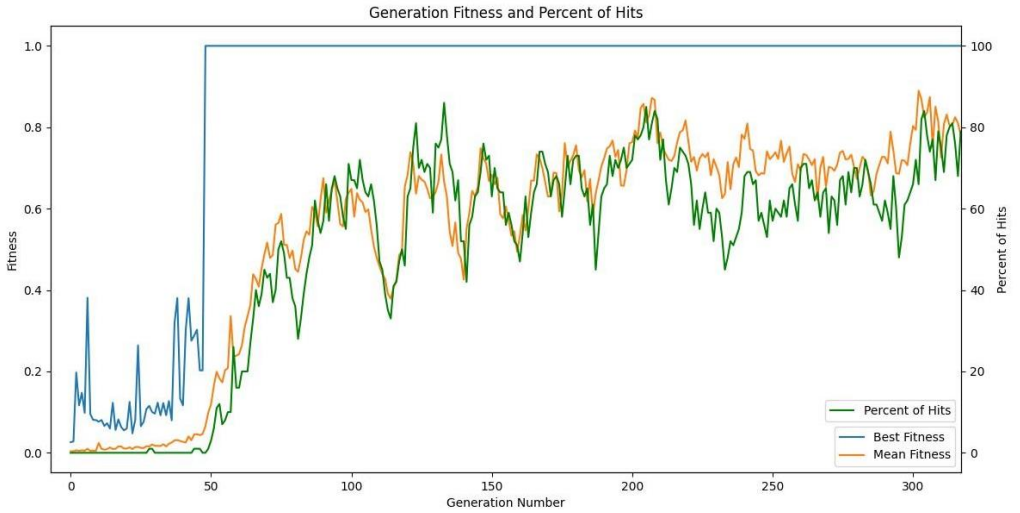

Fig. 5. Training process for NEAT with static mutation parameters

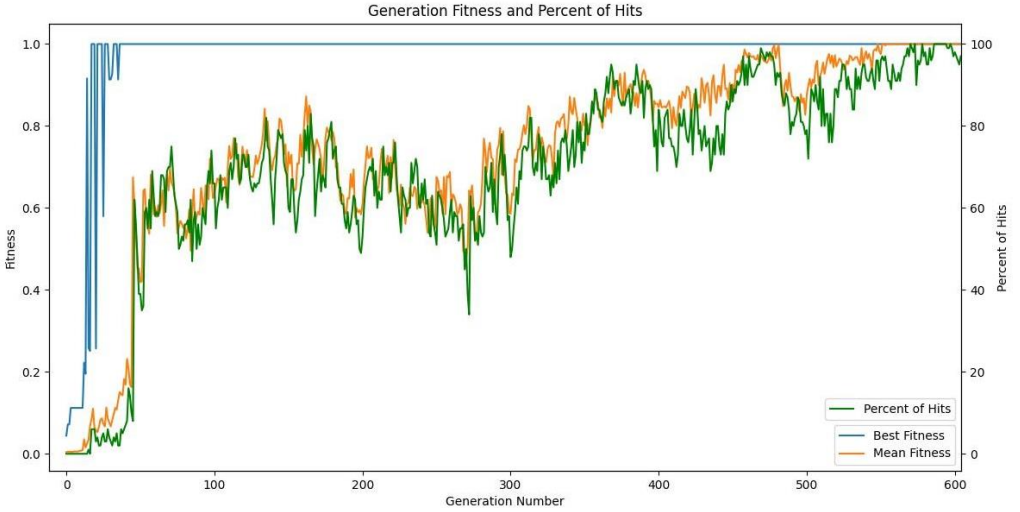

Fig. 6. Training process for NEAT with dynamic mutation parameters

Conclusion

In this paper, we investigated the parameters and mutation influence on the results of neuro-genetic evolution and NEAT training. We used the average fitness in the generation as the main metric for the results of experiments. Experiments with neuro-genetic evolution demonstrated that for the best possible results, we need to use the mutation strength with low values, but this may cause the risk of getting stuck in a local best solution, which leads to the inability to achieve the best results in all experiments. The same problems were observed in experiments with the NEAT method, where the desired results couldn't be achieved without additional modifications.

The proposed strategy with decreasing mutation parameters' values has shown good results: although it increased the number of needed generations significantly, it allowed for improvement of the training results. For neuro-genetic evolution, we managed to achieve the best fitness guaranteed in all experiments. Also, we didn't need to tune mutation strength manually, since this strategy allows us to start with high mutation strength values and automatically decrease it to the needed level. A similar strategy also worked for the NEAT method, allowing to reach the desired results in 80% of the performed experiments.

The proposed strategy may not work optimally for every task since it requires more epochs for training, but it offers its benefits, which were mentioned above. The next steps of the proposed investigation might be a comparison with other strategies for dynamically adjusting training parameters and the ability to combine our approach with other methods which can lead to effectiveness improvement in resolving the various tasks.

References

[1] *Lehman, J., & Miikkulainen, R. (2013).* "Neuroevolution". *Scholarpedia*, 8(6), 30977. https://doi.org/10.4249/scholarpedia.30977

[2] *Ahmad, F., Isa, N. a. M., Osman, M. K., & Hussain, Z. (2010).* "Performance comparison of gradient descent and Genetic Algorithm based Artificial Neural Networks training". *10th International Conference on Intelligent Systems Design and Applications.* https://doi.org/10.1109/isda.2010.5687199

[3] *Goldberg, D. E. (1989).* "Genetic Algorithms in Search, Optimization, and Machine Learning". *The University of Alabama: Adisson-Wesley Publishing Company, Inc., 412.*

[4] *Poli, R., Langdon, W. B., & McPhee, N. F. (2008).* "A Field Guide to Genetic Programming". *Lulu Enterprises, UK Ltd, 242.*

[5] *Ronald, E. M. A., & Schoenauer, M. (1994).* "Genetic Lander: An experiment in accurate neuro-genetic control". In *Lecture Notes in Computer Science, 452–461.* https://doi.org/10.1007/3-540-58484-6_288

[6] *Stanley, K. O., & Miikkulainen, R. (2002).* "Evolving Neural Networks through Augmenting Topologies". *Evolutionary Computation, 10(2), 99–127.* https://doi.org/10.1162/106365602320169811

[7] *Pham, D. T., & Karaboğa, D. (1997).* "Genetic algorithms with variable mutation rates: application to fuzzy logic controller design". *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering, 211(2), 157–167.* https://doi.org/10.1243/0959651971539975

[8] *Hassanat, A. B. A., Almohammadi, K., Alkafaween, E., Abunawas, E., Hammouri, A. M., & Prasath, V. B. S. (2019).* "Choosing Mutation and Crossover Ratios for Genetic Algorithms - A Review with a New Dynamic Approach". *Information, 10(12), 390.* https://doi.org/10.3390/info10120390

[9] Unity homepage. URL https://unity.com/

ВПЛИВ ВЕЛИЧИНИ ТА ЙМОВІРНОСТІ МУТАЦІЇ НА НАВЧАННЯ ДЛЯ ТРАДИЦІЙНОГО ГЕНЕТИЧНОГО АЛГОРИТМУ ТА NEAT. СТРАТЕГІЯ ЗМІНИ ПАРАМЕТРІВ МУТАЦІЇ ПРИ ВИЯВЛЕННІ СТАГНАЦІЇ ПРИСТОСОВАНОСТІ

В. Прецель, Р. Шувар

Львівський національний університет імені Івана Франка,
вул. Драгоманова 50, 79005 Львів, Україна
v.pretsel@gmail.com, roman.shuvar@lnu.edu.ua

На даний момент існує значна кількість методів нейроеволюції. В той час, коли генетичним алгоритмам доводиться конкурувати із іншими методами тренування нейромереж, перші почали демонструвати свої переваги з розвитком обчислювальної техніки і добре себе проявили у задачах, з якими більш традиційні методи, такі як градієнтний спуск, не можуть застосовуватись. Це говорить про доцільність подальшого розвитку цього напрямку. Окрім розробки нових методів нейроеволюції, досліджуються вже існуючі. Різна конфігурація параметрів навчання може мати значний вплив на результат, і окрім вивчення впливу цих параметрів також розроблюються різні стратегії підбору цих параметрів або навіть їх автоматичної зміни під час навчання.

Метою даної роботи є дослідити вплив параметрів на перебіг навчання двох популярних методів нейроеволюції: традиційного генетичного алгоритму зі статичною топологією нейромережі та методу нейроеволюції аугментуючих топологій – NEAT. В першому методі досліджуваним параметром буде величина мутації, а для NEAT – величина мутації і ймовірності структурних мутацій. Окрім того, для обох методів запропоновано стратегію динамічної зміни досліджуваних параметрів, які повинні вирішити проблему потрапляння в локальний екстремум, що загрожує отриманню не найкращих можливих результатів навчання. Всі досліди проводились на одній прикладній задачі з керуванням польоту ракети від Землі до Місяця, яка вимагає значної точності тренованих ваг нейромережі для успішності навчання. Результати навчання аналізувались за такими показниками: середня пристосованість в поколінні, найкраща пристосованість в поколінні, відсоток успішних особин в поколінні. Проведені досліди продемонстрували, що при наявності вимог до високих показників точності, необхідним є малі значення величини мутації для традиційних генетичних алгоритмів, але це створює ризики потрапляння в локальний екстремум. Досліди NEAT показали, що вимоги задачі до значної точності не дозволяли досягти найкращих результатів. Запропонована стратегія зі зменшення величини мутації та ймовірності мутацій змогла зменшити ризик сходження до субоптимального рішення в обох методах, тому такий підхід вцілому зміг себе виправдати. Подальші дослідження цієї стратегії разом із порівнянням із іншими існуючими підходами можуть проявити й інші свої переваги, і тому показують доцільність продовження пошуків.

Ключові слова: генетичні алгоритми, нейроеволюція, штучні нейронні мережі, NEAT

Стаття надійшла до редакції 06.12.2023.
Прийнята до друку 23.02.2024.