Application of the Hierarchic Memetic Strategy HMS in Neuroevolution

Mateusz Sokół and Maciej Smołka(✉)

Institute of Computer Science, AGH University of Science and Technology, Kraków, Poland
mateusz.sokol@outlook.com,
smolka@agh.edu.pl
https://www.informatyka.agh.edu.pl/en/

Abstract. Quite recently some noteworthy papers appeared showing classes of deep neural network (DNN) training tasks where rather simple one-population evolutionary algorithms (EA) found better solutions than gradient-based optimization methods. However, it is well known that simple single-population evolutionary algorithms generally suffer from the problem of getting stuck in local optima. A multi-population adaptive evolutionary strategy called Hierarchic Memetic Strategy (HMS) is designed especially to mitigate this problem. HMS was already shown to outperform single-population EAs in general multi-modal optimization and in inverse problem solving. In this paper we describe an application of HMS to the DNN training tasks where the above-mentioned single-population EA won over gradient methods. Obtained results show that HMS finds better solutions than the EA when using the same time resources, therefore proving the advantage of HMS over not only the EA but in consequence also over gradient methods.

Keywords: Neuroevolution · Deep neural networks · Evolutionary algorithm

1 Introduction

Deep Neural Networks (DNN) in their versatility have become one of the most popular techniques to be used in many areas, including advanced applications such as image recognition [6] or reinforcement learning [8]. In the DNN training domain gradient methods, such as Adam [4], are state-of-the-art techniques for problems such as the supervised learning. As the development advanced new methods emerged to solve restrictions emerging from existing approaches, including architectural solutions, e.g. deep residual learning [3]. One of the Deep Learning (DL) fields that has been in the spotlight for many years is reinforcement learning (RL). The specificity of optimization problems stated in this field,

This research was supported in part by PLGrid Infrastructure and by the funds of Polish Ministry of Education and Science assigned to AGH University of Science and Technology.

D. Groen et al. (Eds.): ICCS 2022, LNCS 13351, pp. 422–429, 2022.
https://doi.org/10.1007/978-3-031-08754-7_49

i.e. the multimodality and deceptiveness of the loss (objective) function, forced to search for new model training approaches in both gradient based methods [5] and those coming from other optimization techniques.

Neuroevolution, as an application of evolutionary algorithms in the neural networks domain, offers a multitude of approaches to be used: from architecture search to plain model training. Evolution as a substitution for gradient based training algorithms was recently successfully applied in RL settings [1,7,9]. In [9], which was the main inspiration for our work, a simple Genetic Algorithm (GA) was able to outperform state-of-the-art gradient methods, such as Deep Q-Learning (DQN) and Actor-Critic, for selected Atari games. In the same paper authors also considered the problem of getting stuck in local minima and environment deceptiveness in RL, proposing "Novelty Search" method for GA that rewards an agent for picking new actions, therefore enhancing exploration. Similar research was done for Evolution Strategies (ES) in [1], as ESs are even more prone to local minima traps, and a set of new "Novelty Search" methods for ESs was proposed. Another proposition described in [7] made one step further and showed that abandoning fitness maximization completely and focusing solely on rewarding novelty in actions also offers a viable approach for tackling environment deceptiveness.

Some advances in evolving both network topology and weights were also made. The work in [13,14] showed that as the human brain exhibits regularities and modularity in its structure a similar approach in DNN encoding can allow to encode trained neural networks previously unavailable due to their scale. In their first work [13] authors propose a graph based indirect encoding technique and apply this method in [14] to evolve both architecture and weights showing that DNN with 8 million connections can be effectively trained with an evolutionary algorithm.

One of the evolution-based methods that has not been applied in the DNN training is Hierarchic Memetic Strategy (HMS). HMS, introduced in [12] is a multi-population strategy with populations organized in a hierarchy according to the accuracy of performed search. It has been specifically designed to address ill-conditioned problems with high risk of getting stuck in local minima, with a special attention for the inverse problems solution. In this work we explore the application of HMS to training DNNs in the RL setting.

2 Hierarchic Memetic Strategy (HMS)

HMS is a complex multi-stage stochastic global optimizer and inverse problem solver. It has been developed to address the ill-conditioning of the considered problem, i.e., various kinds of multimodality of the objective function. It started with solving problems with multiple isolated local minima where it outperformed other global optimization methods (cf. [12]). Currently, the strategy is able to approximate the shape of sets of local minima in problems where those local minima sets have positive Lebesgue measure (cf. [11] and the references therein). The core of HMS is a multi-population evolutionary strategy. It is itself a global optimization method able to detect multiple isolated local minima. In this work

we use only this evolutionary core, so we will concentrate on it in the description below. For the description of the full HMS as well as the analysis of its asymptotic properties we refer the reader to [10] and the references therein.

The HMS core is a multi-population strategy where single-population evolutionary algorithm instances (i.e., *demes*) form a fixed-height parent-child hierarchy with a single root and a different parametrization on each level. A single step of the strategy, called *metaepoch*, consists of a series of standard evolutionary epoch executions in each deme. HMS can utilize various GAs as deme engines. In the past it was used mostly with the simple evolutionary algorithm (SEA), but also with such strategies as the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) or Non-dominated Sorting GA II (NSGA-II). In this work we utilize a no-crossover evolutionary algorithm used in [9]. In the sequel we shall call the algorithm *single-population GA* or simply *GA*.

After each metaepoch we decide whether to spawn new demes based on current state of the strategy. Spawning, in HMS framework called *sprouting*, creates a new child deme around the current local minimum found in the parent deme if a given sprouting condition is satisfied: see Fig. 1. A typical sprouting condition disallows spawning new demes in the proximity of already explored regions. The lifecycle of a deme is driven by Sprouting Condition and Local Stop Condition (LSC) to control search process and avoid unnecessary evaluations in low-quality regions. With Global Stop Condition (GSC) that determines when we should stop our search we end up with a full-fledged framework that allows us to perform broad search on high levels (i.e., closer to the root) to look for promising regions and high-precision search on the lower levels. In case a leaf deme gets stuck around a low-quality minimum a properly set LSC will purge it to avoid unnecessary evaluations. In this work the parametrization of each level consists of mutation power, population count, LSC and sprouting condition.

3 Application in Neuroevolution

In this section we will explore and explain the proposed method for incorporating HMS in the neuroevolution setting for RL environments. As shown in [9] the fact that a simple GA can outperform gradient methods suggests specific characteristics of the loss function, such as multimodality or deceptiveness. These

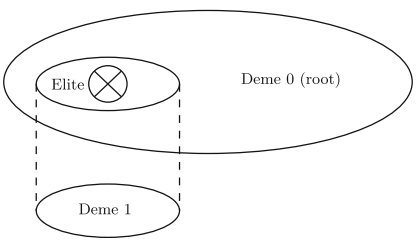

Fig. 1. Idea of HMS sprouting operation

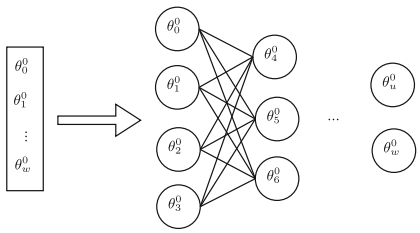

Fig. 2. Genotype direct encoding

are target issues for HMS, i.e., a method aimed at avoiding getting stuck in local minima, that are expected to show the advantage of HMS over single-population GA and consequently over gradient methods.

The first decision to be made when applying evolutionary algorithm in the DNN domain is the representation. As a neural network is not directly suitable for genotype operations we need an encoding technique to easily and unambiguously translate genotype into a network. In [2] a distinction was proposed into three types of encoding: direct, indirect and parametric. In this work we use a special kind of direct encoding with compression introduced in [9] for Atari problems. In the direct encoding a genotype determines only weights and biases. As shown in Fig. 2 genotype, which is an array of genes being floating point values, is unambiguously translated into a neural network. A gene θ_m^n at index m for some individual in epoch n represents a specific weight. This one-to-one direct encoding allows to interpret a gene as a concrete value for some weight or bias in the target DNN. From now on we will call this translation process a DNN materialization. It is worth noticing that in our experiments m exceeds 1.5 million, so the search domain is rather big.

In our experiments the evaluation of an individual is the process of running an individual in RL environment for a given number of iterations, determined by episode duration. For each step the model makes a decision which action to take by performing a forward pass for an input, which is a current observation. A reward r_i is given in i-th step and the final fitness of the model is the sum of all rewards G acquired through the episode: $G = r_1 + r_2 + \cdots + r_T$. The episode duration depends on the environment and selected actions: if in a game we lose all available lives the episode ends. Also the exact definition of r_i varies among games: from the collected item number to the time of staying alive.

4 Experiments

The methodology described in the previous section was applied to different RL environments, following the work in [9]. Main experiments were conducted in the Atari 2600 environment[1].

In all experiments we used the same type of LSC and sprout condition. The LSC stopped a deme after a given number of metaepochs without a significant improvement in the best objective value. The sprout condition prohibited sprouting when the nearest child-deme centroid was closer than a given minimal distance.

For Atari games GA parametrization and DNN architecture was the same as in [9] and only HMS parameters were adapted to conform with desired number of evaluations in total. As [9] compares GA to gradient methods, our work only covered HMS to GA comparison. It's also important to stress that GA algorithm to which we are comparing exactly follows the one proposed in [9]. Selected DNN architecture was originally proposed in [8] and is comprised of

[1] https://gym.openai.com/envs/#atari.

Table 1. Atari 2600: global parameters

Parameter	2-level HMS	Single-pop GA
Environment	Atari 2600	Atari 2600
GSC (nr of epochs)	40	40
Metaepoch length	3	–
Number of levels	2	1

Table 2. Eval. number

Game	GA	HMS
Frostbite	39000	35250
Kangaroo	39000	31950
Zaxxon	39000	33750
Venture	39000	31950
Asteroids	39000	34200

convolutional layers and fully connected ones with ReLU activation function. Also each experiment was performed for 40 epochs. The selection of parameters for the experiments are shown in Table 1 and 3.

In this work we report the results of experiments with selected five Atari games, which were also used in [9]. These are: Frostbite, Kangaroo, Zaxxon, Venture and Asteroids. Here, similarly to [9], we also got best results for the Frostbite game where the best individual was able to play the game and finish with a score on par with a human player. What is the most important, HMS was able to reach those scores much faster and using less iterations. The same applies to the Zaxxon game and also to some degree to the Kangaroo game, for which score was not that much better but resulting individual learned basic behaviour of dodging falling items. Results of these experiments are presented in Fig. 3. Each plot shows the medians and the 95% bootstrapped confidence intervals obtained from evaluating the best individual in a given epoch multiple times in the target environment, each time with a different initial seed. The maximum number of evaluations for both competing algorithms is shown in Table 2. For single-population GA the number was the same for each game, whereas in HMS this number varies but for each game it is less than in GA. The results show that even in the worst case (Asteroids) HMS performs at least as well as than GA. In remaining four cases HMS performs better and its advantage over GA is the most prominent in Frostbite and Zaxxon cases.

Table 3. Atari 2600: local parameters

Parameter	HMS level 0	HMS level 1	Single-pop GA
Mutation probability	1.0	1.0	1.0
Mutation power	0.05	0.002	0.002
Population count	600	150	1000
Number of promoted	45	20	20
LSC (no improvement)	–	3	–
Sprouting condition	0.5	–	–

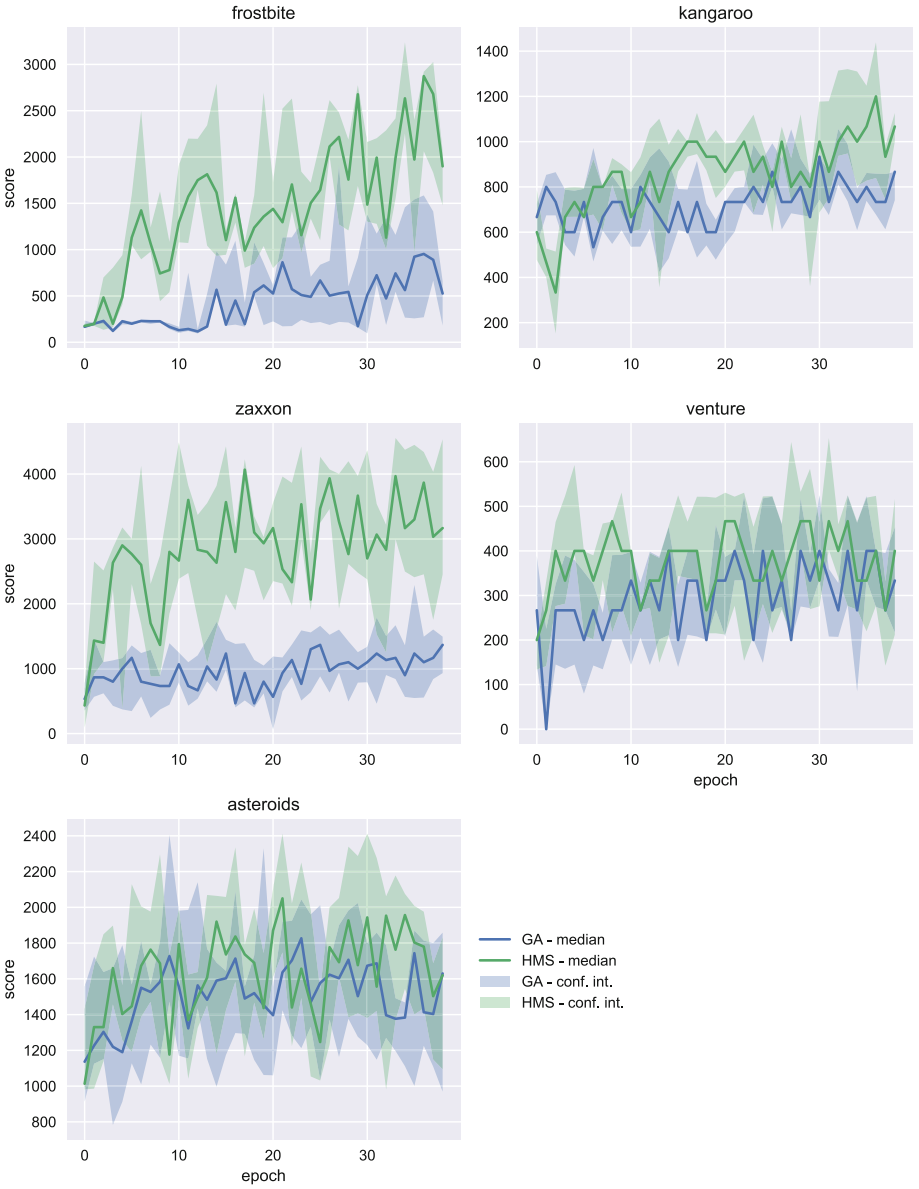

Fig. 3. Results for Atari games experiments.

The implemented framework for conducting experiments is available in a public GitHub repository[2]. A showcase playlist with recorded episodes performed by the best individuals for selected environments is also available[3].

[2] https://github.com/mtsokol/hms-neuroevolution.
[3] https://bit.ly/hms-neuroevolution-playlist.

5 Conclusions

In this work we showed that HMS as an evolutionary optimization method designed for multimodal and deceptive fitness functions outperforms simple GA and consequently also gradient methods in selected RL problems. In our opinion it makes HMS a noticeable competitor in RL area. The obtained results are also very promising in the context of the application of HMS in other problems involving DNN training. Apart from considering new problems with a fixed DNN structure further studies will include the application of HMS in DNN architecture learning problems.

References

1. Conti, E., Madhavan, V., Petroski Such, F., et al.: Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems (NIPS 2018), pp. 5032–5043. Curran Associates Inc., Red Hook (2018)
2. Fekiač, J., Zelinka, I., Burguillo, J.: A review of methods for encoding neural network topologies in evolutionary computation. In: European Conference on Modelling and Simulation (2016)
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778 (2015)
4. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1703.00548 (2017)
5. Konda, V., Tsitsiklis, J.: Actor-critic algorithms. In: Advances in Neural Information Processing Systems, vol. 12 (2000)
6. LeCun, Y., Boser, B., Denker, J.S., et al.: Backpropagation applied to handwritten zip code recognition. Neural Comput. **1**(4), 541–551 (1989)
7. Lehman, J., Stanley, K.: Abandoning objectives: evolution through the search for novelty alone. Evol. Comput. **19**, 189–223 (2011)
8. Mnih, V., Kavukcuoglu, K., Silver, D., et al.: Human-level control through deep reinforcement learning. Nature **518**, 529–533 (2015)
9. Petroski Such, F., Madhavan, V., Conti, E., et al.: Deep neuroevolution: genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning. arXiv preprint arXiv:1712.06567 (2018)
10. Sawicki, J., Łoś, M., Smołka, M., Schaefer, R.: Understanding measure-driven algorithms solving irreversibly ill-conditioned problems. Nat. Comput. (2021). https://doi.org/10.1007/s11047-020-09836-w
11. Sawicki, J., Łoś, M., Smołka, M., Schaefer, R., Álvarez-Aramberri, J.: Approximating landscape insensitivity regions in solving ill-conditioned inverse problems. Memetic Comput. **10**(3), 279–289 (2018). https://doi.org/10.1007/s12293-018-0258-5
12. Smołka, M., Schaefer, R., Paszyński, M., Pardo, D., Álvarez-Aramberri, J.: An agent-oriented hierarchic strategy for solving inverse problems. Int. J. Appl. Math. Comput. Sci. **25**(3), 483–498 (2015)

13. Stanley, K.: Compositional pattern producing networks: a novel abstraction of development. Genet. Program Evolvable Mach. **8**, 131–162 (2007)
14. Stanley, K.O., D'Ambrosio, D.B., Gauci, J.: A hypercube-based encoding for evolving large-scale neural networks. Artif. Life **15**(2), 185–212 (2009)