DEEP NEUROEVOLUTION

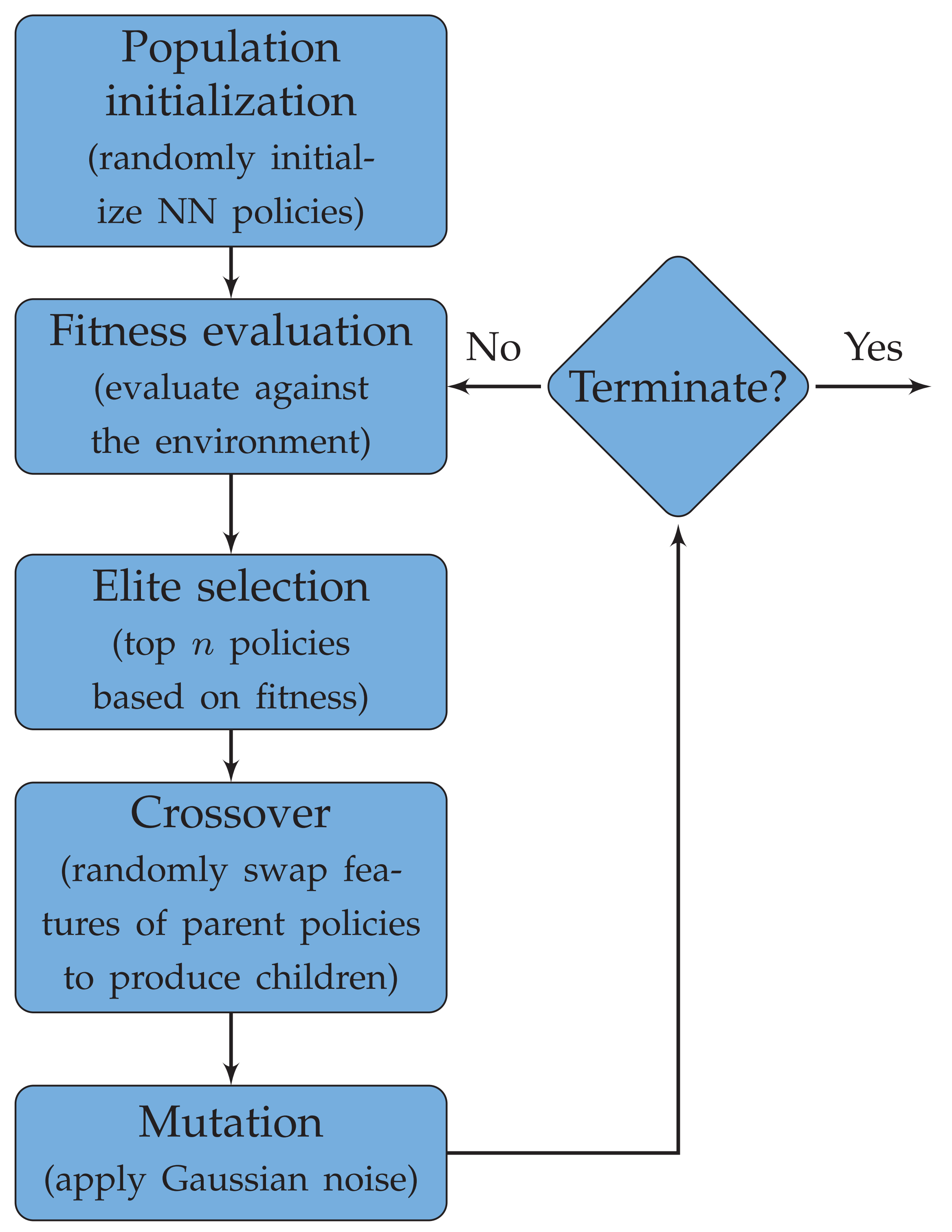
{ HANNES LIIK AND OLEH MATSUK }   UNIVERSITY OF TARTU, INSTITUTE OF COMPUTER SCIENCE

INTRODUCTION

In this project, we have implemented Genetic Algorithms (GAs) to optimize neural network policies for reinforcement learning environments [1]. We present our results on the LunarLander OpenAI gym environment [2], which can be seen in Figures 1-3.

ALGORITHM

Genetic Algorithms


Population initialization (randomly initialize NN policies)
Fitness evaluation (evaluate against the environment)
No
Terminate?
Yes
Elite selection (top n policies based on fitness)
Crossover (randomly swap features of parent policies to produce children)
Mutation (apply Gaussian noise)

REFERENCES

[1] F. P. Such, V. Madhavan, E. Conti, J. Lehman, K. O. Stanley, and J. Clune, "Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning," *CoRR*, vol. abs/1712.06567, 2017.

[2] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," 2016. https://gym.openai.com/envs/LunarLander-v2/.

ENVIRONMENT

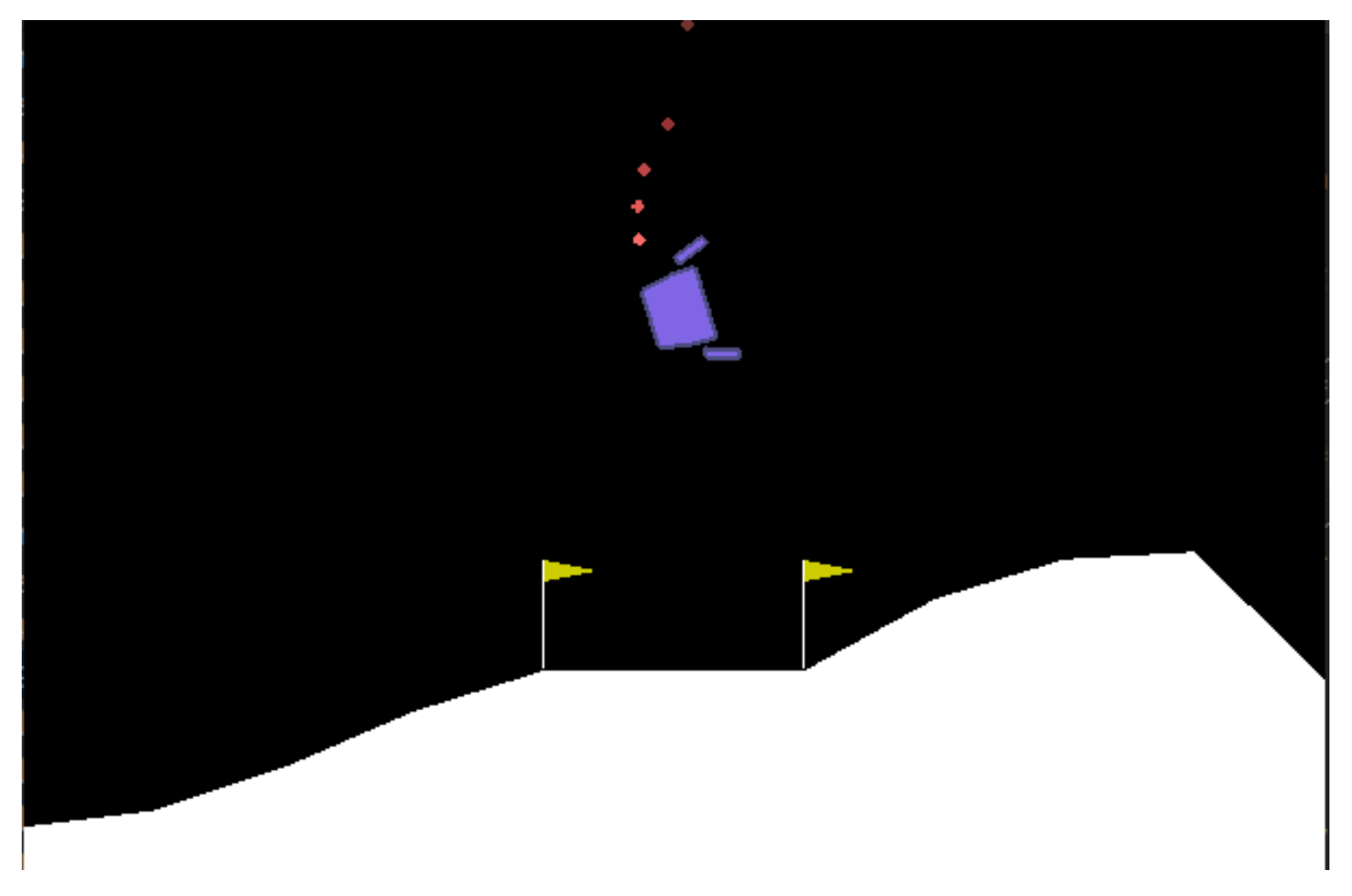


Figure 1: Untrained

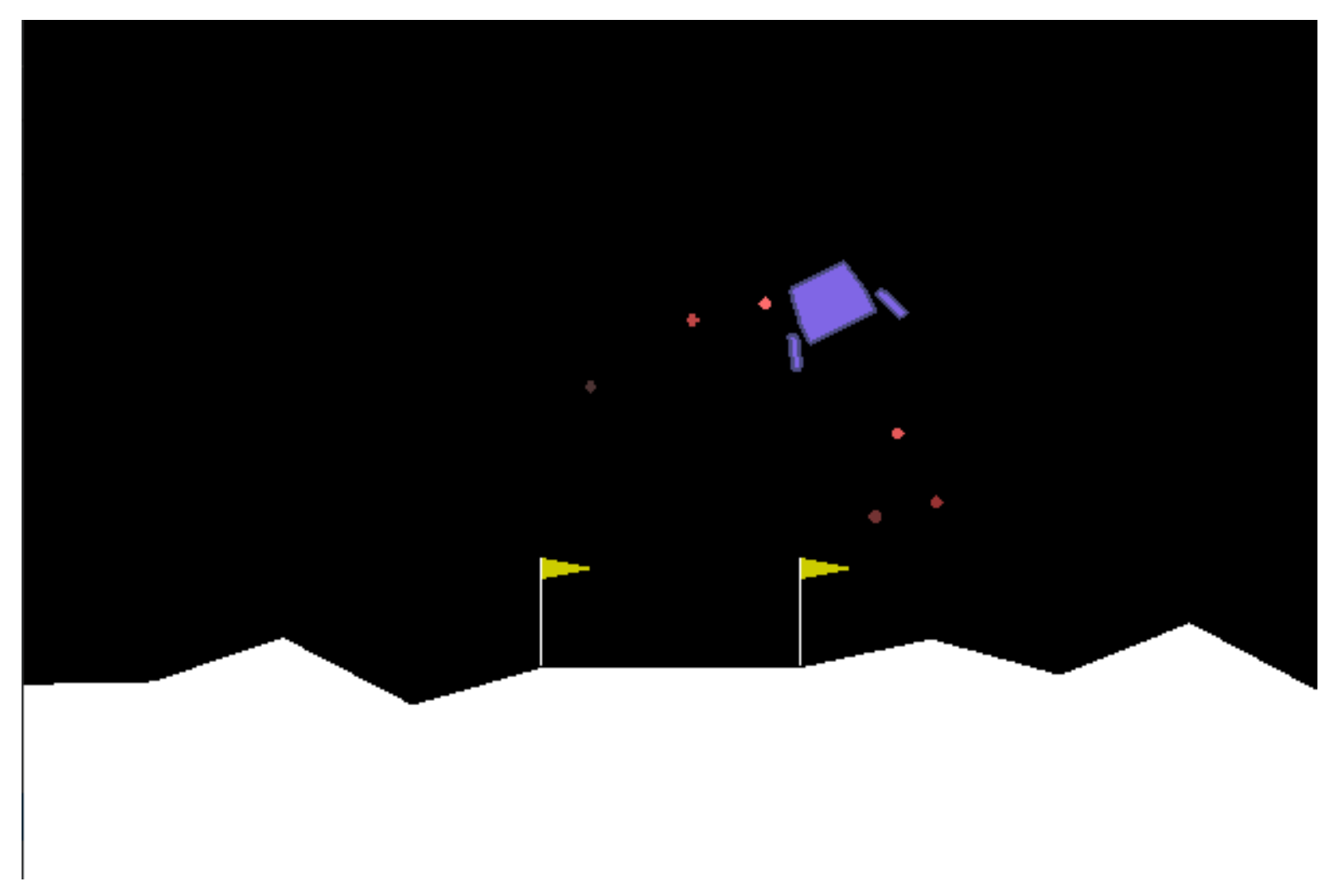


Figure 2: Generation 10

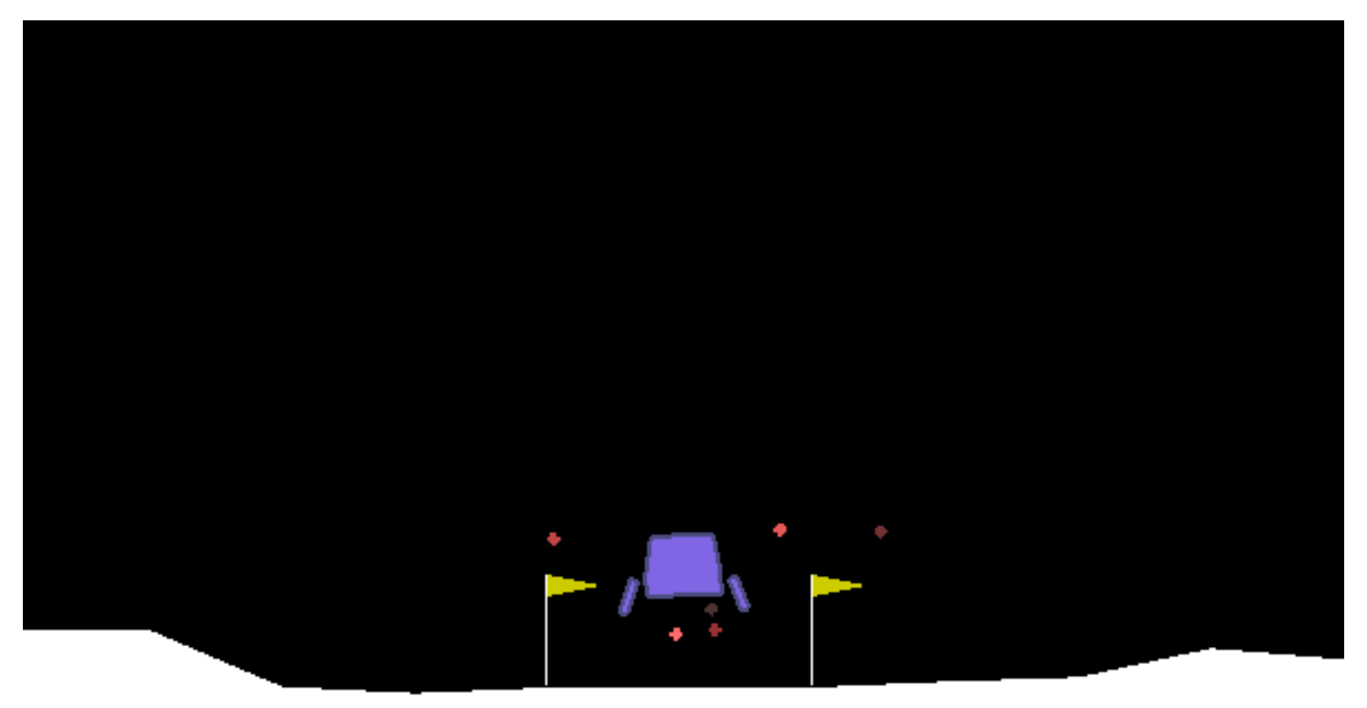


Figure 3: Generation 50

RESULTS

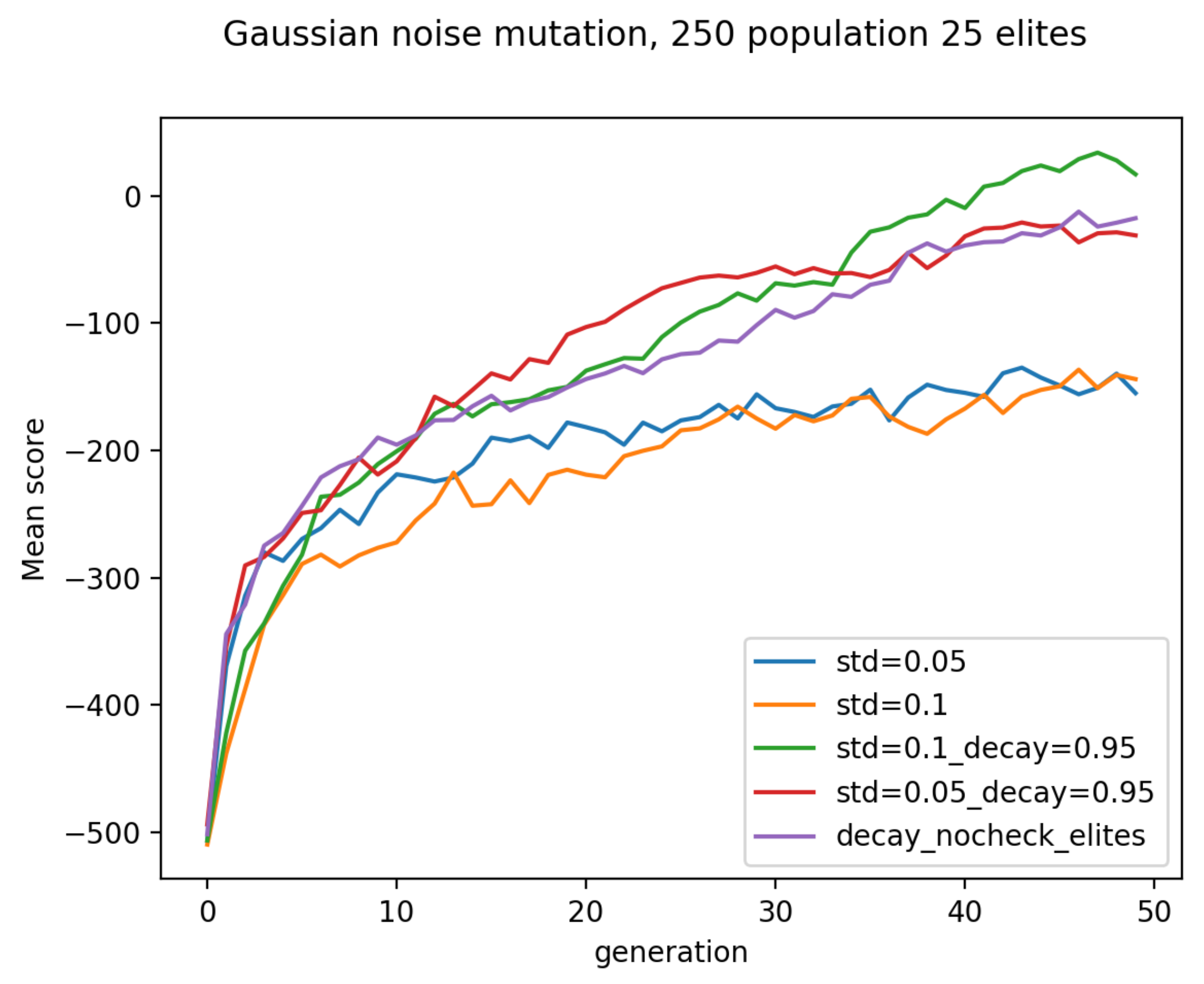


Figure 4: Hyperparameter search, decay and elite checking.

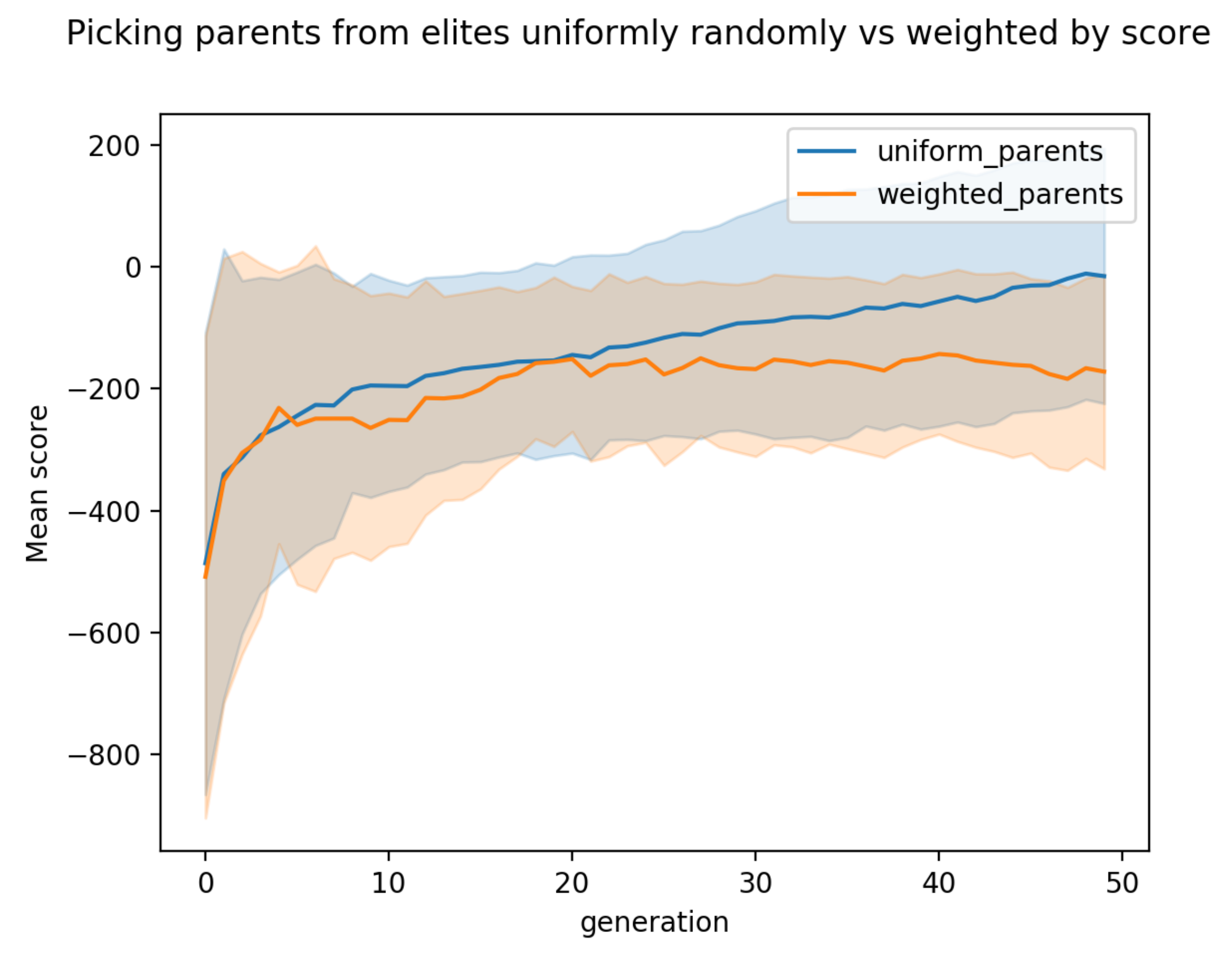


Figure 5: We observe that score-weighted parent selection degrades performance.

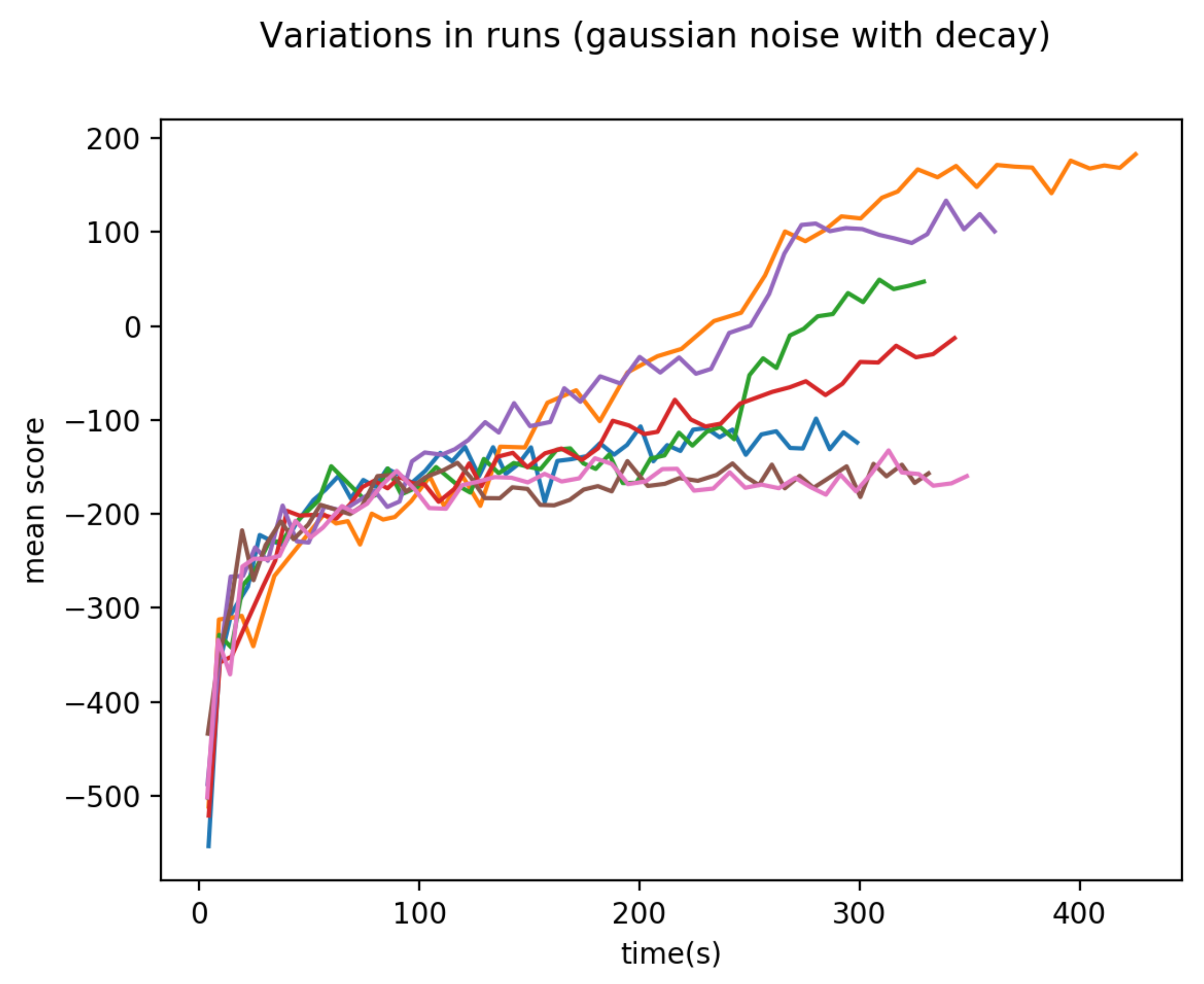


Figure 6: There are vast variations across runs with the same hyperparameters, so we report averages.

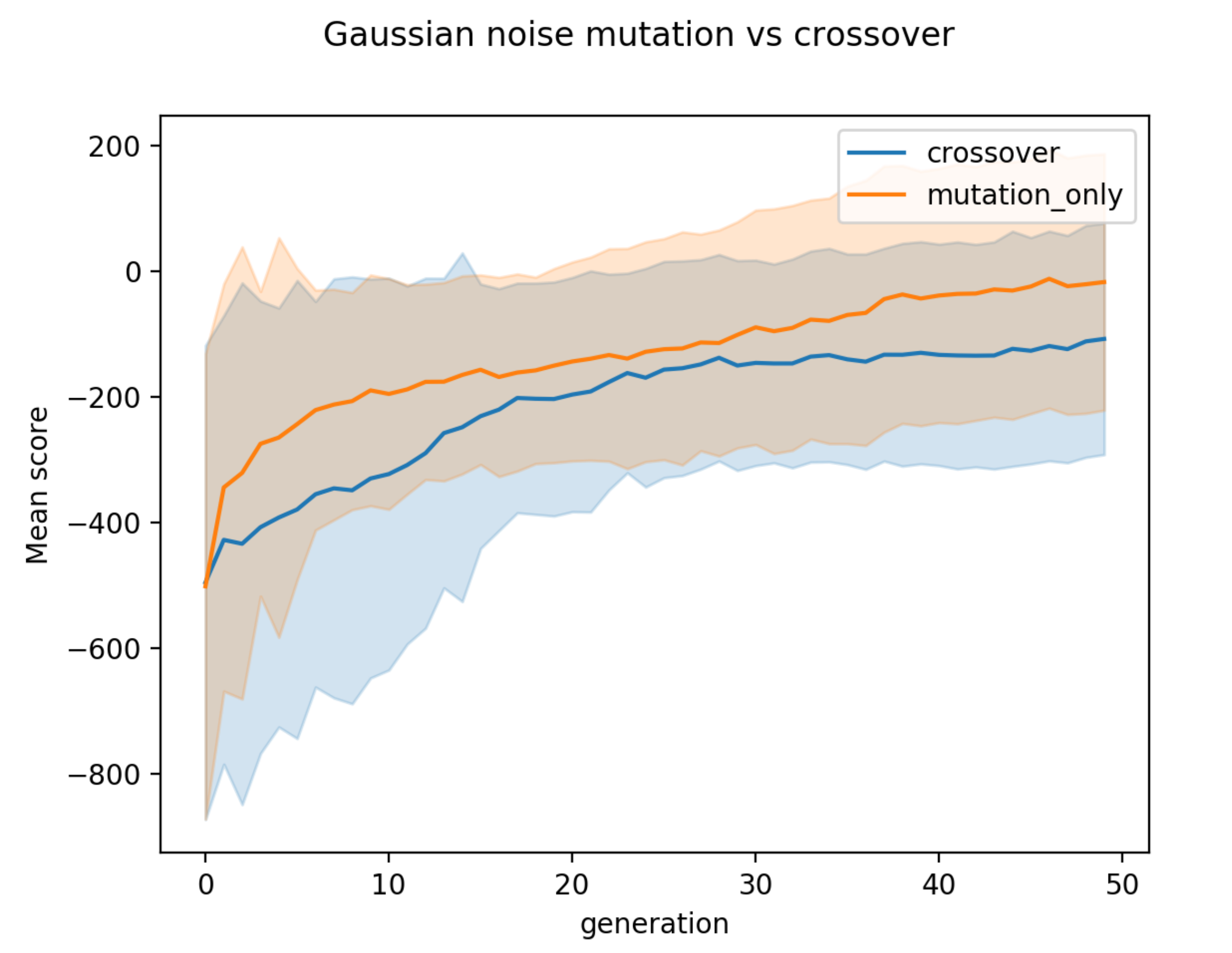


Figure 7: We observed no benefit in using crossovers.

EXPERIMENTS

We experimented with different additions to the vanilla GA. The algorithm solves the task in 5 minutes on a Rocket HPC cluster node (Figure 6).

Mutation decay. Much like learning rate decay in gradient descent, we experiment with annealing the mutation strength after each generation. This would, in principal, allow the algorithm to better settle in a local optimum. We observed noticeable and consistent improvement by applying mutation decay as seen in Figure 4.

True elite checking. A method in which the top n elites are evaluated many times to get a better estimate of their performance and find the "true elite" [1]. The true elite is carried on to the next generation without mutation. We did not notice performance improvements using this method as seen in Figure 4, but it may be necessary for other, more stochastic environments.

Crossover. The act of combining genes (in our case, neural network weights) from different parents. A common part in many GAs, but left unexplored in papers about GAs for reinforcement learning. We implement crossover such that with probability p a network layer is copied from another parent. Our crossover strategy performed worse than the baseline as seen in Figure 7.

Weighted parent selection. Preferring parents with higher scores. We observed that standard uniform parent picking performs better as seen in Figure 5. Perhaps it keeps the gene-pool healthier and results in better exploration.

ADDITIONAL INFORMATION

Source:
github.com/hannesliik/deep_neuroevolution
Visuals:
bit.ly/neuroevolution