Evolving Unsupervised Neural Networks for slither.io

Mitchell Miller, Megan Washburn, and Foaad Khosmood

California Polytechnic State University, Department of Computer Science and Software Engineering

LEADERBOARD:

CAL POLY SAN LUIS OBISPO

NEATAPTIC

SYNOPSIS

We create Slither.io bots using neuroevolution of augmenting topologies (NEAT) and compare their performance to that of the best open source bot available online (high-performing expert system). With a fitness function based on the final size of the agent, our results show steady improvement in average score.

PRIOR WORK

The implementation of rtNEAT (real-time NEAT) in NeuroEvolvingRobotic Operatives (NERO) improved the performance and engagement of AI agents by learning from players and allowing the agents to develop original behavior [3].

NEUROEVOLUTION

This method was first developed by Kenneth O. Stanley and Risto Miikkulainen in 2002 to pose a solution to control and sequential decision tasks [4]. The evolutionary structure of NEAT works by altering the associated networks' node and connection layout from generation to generation; this greatly expands the search space of traits in the artificial neural network (ANN).

TOOLS

Neataptic [5] | Slither.io Agent Manager [2]
Slither.io agent by Eskandary [1]

PARAMETER ADJUSTMENT

We chose three different sets of parameters to see how the fitness of the genomes progressed after 100 generations of running our neuroevolution. See Table 1 for the difference in each runs parameters. All other parameters were held constant.

Table 1: Run Parameters

Run	Number of Games	Selection	Initial Population	Mutations
1	5	Power (p=4)	Random Hidden Nodes	All
2	10	Tournament (k=10, p=1)	Random Hidden Nodes	All
3	10	Tournament (k=10, p=1)	Perceptron	Add/Change (no removal)

SLITHER.IO NEAT SYSTEM

In order to run a neural-evolution for Slither.io, a system that could run web browser games headlessly (without rendering the game interface to a screen) for many days without intervention was necessary. It involved modifying the Slither.io agent Manager and integrating Neataptic to orchestrate our neuroevolution, along with modifying Eskandary and Cailliau's Slither.io agent to be a shell that accepts a neural network as its decision-making brain.

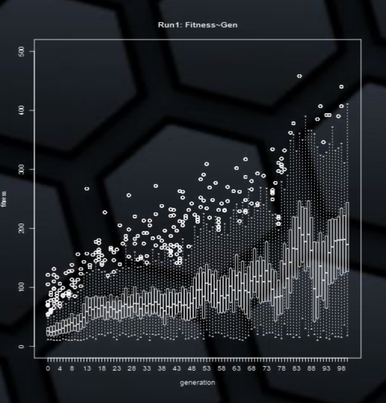

Figure 1: Boxplot of each population's fitness after running 100 generations for Run 1.

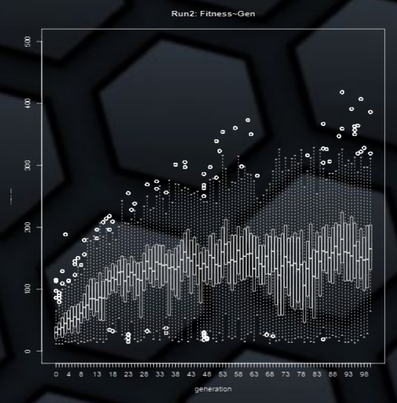

Figure 2: Boxplot of each population's fitness after running 100 generations for Run 2.

DESIGN AND EXPERIMENT

We observed two main attributes when running our neuroevolutionfor Slither.io. Primarily, we focused on the resulting scores of our generated agent to compare with with Eskandary and Cailliau'sexpert-system agent. Secondly, we looked at the performance of our neuralevolution given different parameters. Additionally, we focused on what parameter adjustments would improve the speed of our neuroevolution.

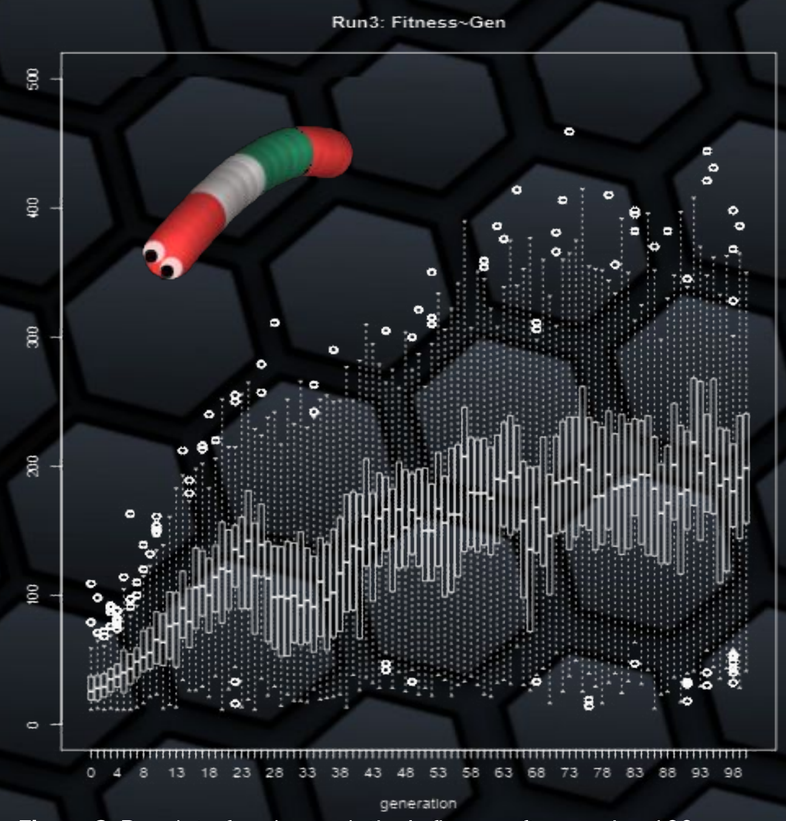

Figure 3: Boxplot of each population's fitness after running 100 generations for Run 3.

CONCLUSIONS

Regardless of being unable to match the score performance of Eskandary and Cailliau's expert-system agent, our generated agents exhibited a variety of unique behaviors, reacting to other players in a manner similar to both human players and the expert-system agent, with no supervision.

REFERENCES

1. Ermiya Eskandary and Théophile Cailliau. 2016. Slither.io Bot. https://github.com/ErmiyaEskandary/Slither.io-bot
2. K00Klust. 2016. Slither.io Bot Manager. https://github.com/K00sKlust/slither.io-bot-manager
3. K. O. Stanley, B. D. Bryant, and R. Miikkulainen. 2005. Real-time neuroevolution in the NERO video game. IEEE Transactions on Evolutionary Computation 9, 6 (Dec 2005), 653–668. https://doi.org/10.1109/TEVC.2005.856210
4. Kenneth O. Stanley and Risto Miikkulainen. 2002. Evolving Neural Networks Through Augmenting Topologies. evolutionary Computation 10, 2 (2002), 99–127. http://nn.cs.utexas.edu/?stanley:ec02
5. Thomas Wagenaar. 2018. Neataptic. https://wagenaartje.github.io/neataptic/