Playing SNES Games with NeuroEvolution of Augmenting Topologies

Computer Science department, Bucknell University, Lewisburg, PA

Bucknell UNIVERSITY

BACKGROUND

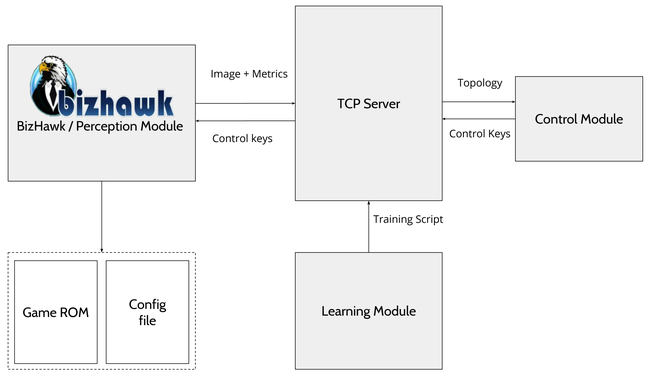
Teaching a computer to play video games has generally been seen as a reasonable benchmark for developing new AI techniques. Recent research efforts have focused on Atari 2600 games, resulting in algorithms such as Deep Q-Learning or Policy Gradient that outperform humans. However, games from Super Nintendo Entertainment System (SNES) are far more complicated and suggest demands for an alternative to the usual optimization approaches commonly seen in Deep Q-Learning or Policy Gradient.

OUR CONTRIBUTION

We present two contributions in our work:

- Introducing an environment to interface SNES games so that researchers can train RL algorithms on these games.
- Investigating NeuroEvolution of Augmenting Topologies (NEAT) (Stanley and Miikkulainen 2001) as a possible approach to develop RL algorithms for SNES games by creating a NEAT agent to play the game Top Gear (1997)

OUR PLATFORM



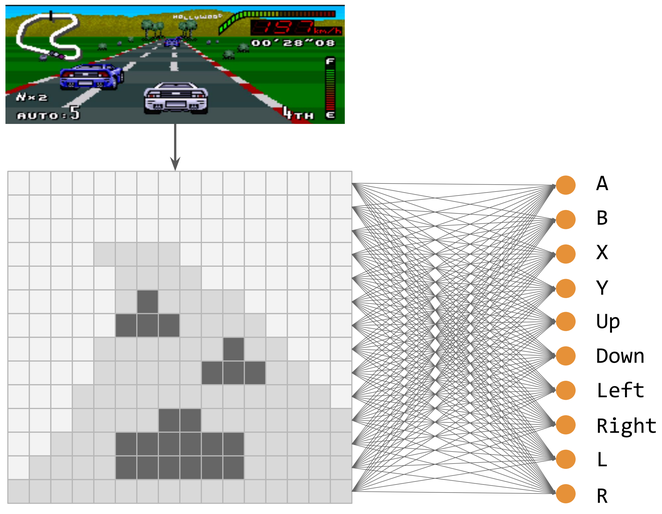
BizHawk is a popular open-source SNES emulator with many supporting features such as save-playback, speed up, that are very helpful for setting up training environment. It also stores the entire RAM map, meaning that we can search for specific information of the game such as score, distance or ranking.

We modified the source code of the emulator so that it becomes part of our TCP setup, with a central TCP Server controls the flow of information from the Control Module to the actual game. In addition, our modified Bizhawk component can read extra information related to the game. Which means that if there is a new game to be added to the platform, users only need to specify a Config file with a new reward function and addresses to specific memory location in the RAM map.

SETUP FOR TOP GEAR (1992)

Pixels input is processed so that cars and road can be differentiated. The input is then downscaled from 256 x 144 to 16 * 14. This input will then be fed directly into a fully connected neural network that maps every input to 10 button outputs according to every button in the SNES controller.

The weights of the network are initially random. However, the weights and the structure of the neural network change overtime due to the mutation. Structures with weight configurations that perform well will survive through generations of training.

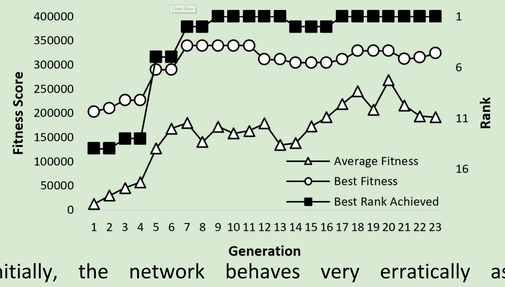


REWARD FUNCTION

The fitness of each network is evaluated through a "fitness function" based on several in-game reward metrics such as Race Ranking, Distance, and Speed. Originally, we only use Race Ranking for our fitness function. However, because initial networks are so weak, all networks usually rank 20th without even completing the race. This leads to the *credit assignment* problem in which many networks with great potential behaviors are regarded as the same the ones without because they share the same reward. To alleviate the problem, we have a fitness function that both rewards long-term goals such as Ranking and short-term goals such as "moving forward." The fitness function is as follow:

$$f_i = -aR + bL + c\int \frac{dV(t)^2}{dt}$$

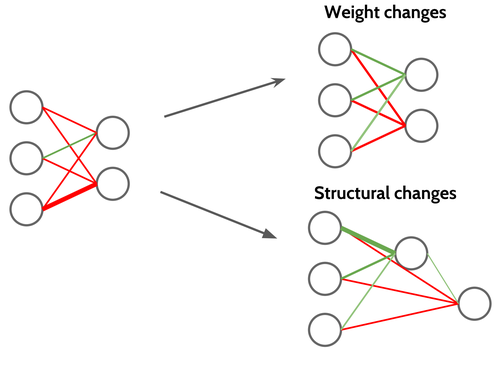
RESULTS



Initially, the network behaves very erratically as expected. However, the agent already starts to stick to the road by generation 3 and steer properly by generation 5. By generation 8, it learns to surpass other cars, and becomes a competent player that performs better than humans already by agent 10.

GENETIC APPROACH

Unlike classic back-propagation approach, NEAT doesn't need a reservoir of training samples to evaluate and modify its connectivity. Instead, connections and weights are modified randomly during mutation phase.



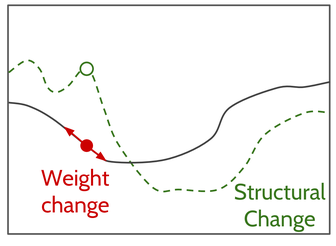
SURVIVAL OF THE FITTEST

PROTECTING INNOVATION

Structural network changes create a new reward landscape that has potential for better optimum. However, these changes also tend to not produce desirable behavior right away and need time to optimize.

To protect this innovation, the speciation process is developed. Similar *genomes* will be grouped into one species and forced to share reward score so that newer and yet-to-be-optimized genomes are protected by older and working ones. The similarity between two genes are calculated using the following equations:

$$\delta = \frac{c_1}{N}E + \frac{c_2}{N}D + c_3\bar{W}$$



If the distance is below a threshold δt then the two genomes are said to belong the same species and will share the adjusted fitness calculated as:

$$f_i' = \frac{f_i}{\sum_{i=1}^{N} sh\big(\delta(i,j)\big)}$$

$$sh\big(\delta(i,j)\big) = \begin{cases} 1, & \delta(i,j) < \delta_t \\ 0, & \delta(i,j) \geq \delta_t \end{cases}$$

ACKNOWLEDGEMENT

This project would not be possible without the support by Professor Christopher L. Dancy and Bucknell Computer Science Department.

REFERENCE

1. Bhonker, N., Rozenberg, S. and Hubara, I., 2016. Playing SNES in the Retro Learning Environment. arXiv preprint arXiv:1611.02205.
2. TASVideos, 2017. BizHawk, available at: https://github.com/TASVideos/BizHawk