The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Playing SNES Games with NeuroEvolution of Augmenting Topologies

Son Pham, Keyi Zhang, Tung Phan, Jasper Ding, Christopher L. Dancy
Department of Computer Science, Bucknell University, Lewisburg, PA 17837
{son.pham, keyi.zhang, tung.phan, cd032, christopher.dancy}@bucknell.edu

Abstract

Teaching a computer to play video games has generally been seen as a reasonable benchmark for developing new AI techniques. In recent years, extensive research has been completed to develop reinforcement learning (RL) algorithms to play various Atari 2600 games, resulting in new applications of algorithms such as Deep Q-Learning or Policy Gradient that outperform humans. However, games from Super Nintendo Entertainment System (SNES) are far more complicated than Atari 2600 games as many of these state-of-the-art algorithms still struggle to perform on this platform. In this paper, we present a new platform to research algorithms on SNES games and investigate NeuroEvolution of Augmenting Topologies (NEAT) as a possible approach to develop algorithms that outperform humans in SNES games.

Introduction

Video games that require some strategy and with reasonably sized state spaces are great platforms for RL research. Traditionally, games used in RL research has been limited to board games like Chess, Checkers or more recently Go and Poker because they tend to have well-defined representation. Recently, RL research has used more visual games such as Atari 2600 games because these games have a fairly large state-space and encourage an approach that can generalize representation of visual inputs.

Algorithms like Deep Q-Network (Mnih et al. 2015) have been able to outperform humans in Atari 2600 games, but still struggle to perform well on SNES games due to the environment being far more complicated (Bhonker et al. 2016). In this extended abstract, we present two early contributions with following 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).

Interfacing with SNES Environment

We used BizHawk Emulator (TASVideos 2017), an open-sourced SNES emulator as the foundation for our environment. BizHawk contains many supporting features such as save-playback and 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 and ranking in real time.

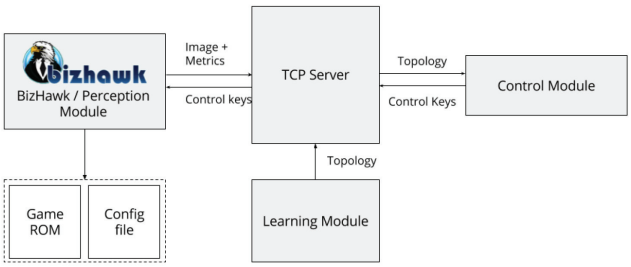

Figure 1: Architecture of the RL platform.

We modified the source code so that it became a perception module. This module queries a Config file that contains all memory locations of specific game information such as score or distance. The module then constantly streams the visual input and these additional information through a TCP server. The TCP server then sends all input information to a control module, which then returns buttons pressed that can be fed back directly to BizHawk emulator.

This setup allows one to add a new game to the platform (by simply writing a Config file for each available game ROM) without writing new specific supporting code. The TCP server alternates the request between the two modules, meaning that the game will only proceed if it receives button output from the latest input, effectively eliminating any input delay problem. Separating the control and perception module also makes the task of writing the training mechanism in the learning module much easier.

NEAT Agent for Top Gear

We used this platform to develop an artificial agent to play Top Gear (1992), a car racing game in which one has to complete the race better than other 19 racers. We developed our agent to be a fully connected network that connect pixel inputs with 10 output buttons. Unlike other neural network algorithms (e.g., those using gradient descent), NEAT doesn't require back-propagation, which means the framework does not need to build up a reservoir of "training samples." Instead, the neural network (called *genome*) will modify its weights (called *genes*) through *random mutation* and *crossover*. All genomes will then directly control the agent in simulation runs to determine the "survival of the fittest."

Random mutation and crossover of genome can creatively generate potentially desirable behavior that is not yet optimized to achieve high fitness score. To protect this innovation, NEAT groups similar genomes together and forces them to share fitness score, a process known as *speciation*. The difference or *distance* δ between two genomes can be calculated as:

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

which is a linear combination of the number of excessive (E) and disjoint (D) genes and average weight differences of matching genes $\overline{W}$. If the distance is below a threshold δ_t then the two genomes are said to belong the same species and will share the *adjusted fitness* f_i' calculated as:

$$f_i' = \frac{f_i}{\sum_{i=1}^{N} sh\big(\delta(i,j)\big)}, sh\big(\delta(i,j)\big) = \begin{cases} 1, & \delta(i,j) < \delta_t \\ 0, & \delta(i,j) \geq \delta_t \end{cases}$$

in which f_i is the original fitness value and $\delta(i,j)$ is the difference between gene i and every other gene j. The speciation process elegantly handles the *exploration-exploitation dilemma* in that it allows innovation to be protected (exploration) while forcing ineffective species to go extinct, creating new room to optimize and mutate high-performing genomes (exploitation).

The fitness value of each genome will be determined after each evaluation run based on a function of in-game metrics. Originally, we only used Race Ranking for our fitness function. However, because initial genomes are not sophisticated enough to even finish the race, they all share the same reward for ranking 20th. This leads to the *credit assignment problem* in which many networks with great potential behaviors are regarded as the same as the ones without because they have the same reward. To alleviate the problem, we used a fitness function that both rewards long-term goals such as Ranking and short-term goals such as "moving forward" (Distance) and "keeping the speed high" (Speed). The fitness of genome i after finishing the race at rank R, distance L and speed $V(t)$ is as followed:

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

Results

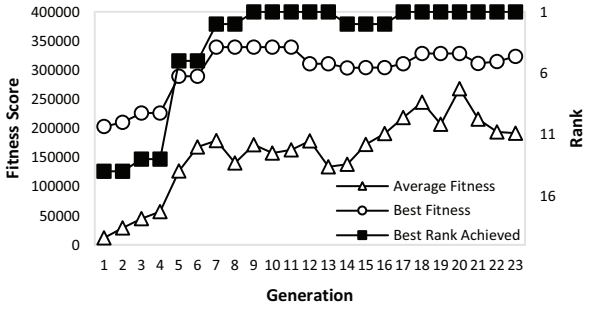

Figure 2: Fitness and Rank achieved through each generation.

In the first few generations, the agent behaves very erratically because the controlling neural network is initialized to be random. 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 very competent by generation 10. We used Top Gear's two-player game mode to test our agent against human players in the same environment and found that the generation-10 agent consistently performs better than humans.

Conclusion

As RL moves on to tasks that are more correlated with those in real life, the development of new RL algorithms should tackle more complicated games both in terms of representation and strategy. Thus, an RL research platform for SNES games from simpler Atari 2600 games is a natural progression. We also present NEAT as a potential alternative to play SNES games compared to traditional back-propagation approaches such as Deep Q-Network. Our NEAT agent only needs 200 training games to achieve better-than-human level of performance. Our approach takes only screen pixels as input, meaning that it is not limited to racing games alone and can apply to all genres of games. Future work will expand to other SNES games and compare NEAT's performance on these games with those of other existing learning frameworks.

References

Bhonker, N., Rozenberg, S. and Hubara, I., 2016. Playing SNES in the Retro Learning Environment. *arXiv preprint arXiv:1611.02205*.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G. and Petersen, S., 2015. Human-level control through deep reinforcement learning. *Nature*, *518*(7540), pp.529-533.

Stanley, K.O. and Miikkulainen, R., 2002. Evolving neural networks through augmenting topologies. *Evolutionary computation*, *10*(2), pp.99-127.

TASVideos, 2017. BizHawk, available at: https://github.com/TASVideos/BizHawk