Solving the Flappy Bird Game through A Neural Network improved with Various Genetic Algorithms

Bilal Mohiuddin
Dhanani School of Science and Engineering
Habib University
Karachi, Pakistan
bm03751@st.habib.edu.pk

Muhammad Ajlal Bawani
Dhanani School of Science and Engineering
Habib University
Karachi, Pakistan
mb03748@st.habib.edu.pk

***Abstract*—Flappy bird was invented as simple tapping game wherein a bird exists within a gravitational field as pipes with small gaps approach the bird-agent. The objective of the game is for the bird-agent to pass through as many pipes as possible. In this project, we have tried to implement a neural network for the game that makes use of genetic algorithms to improve itself with each generation of the genetic algorithm. At the end of the report, the efficiency of each genetic algorithm coupled with the neural network is discussed.**

I. Introduction

Flappy Bird is a simple mobile game in which, as mentioned before, a bird-agent exists within a gravitational field as pipes advance towards it. Each pipe contains a small gap of a standard size. Upon tapping our screen, the bird gains an upward velocity which can then be used to help it pass through the oncoming pipes. The objective of the game is for the user to pass through as many pipes as possible before he/she inevitably collides with one.

We have attempted to create agents that "solve" the game by utilizing a personal neural network. Each neural network improves over time through a genetic algorithm that is applied to it. As with all genetic algorithms, a population is initially created. Each chromosome represents a game solving bird-agent. Each agent has its own neural network. The performance of the game improves as the number of generations increases.

July 5, 2020

A. Genetic Algorithms

Genetic Algorithms aim to replicate the natural process of evolution by finding the optimum solution to a problem through replicating evolutionary processes. A chromosome represents a single solution within our solution space. We start off by populating our solution space randomly with a fixed number of chromosomes. Each chromosome is tested against the solution space. Using different selection policies, we then select parents which, through crossover and mutation processes are used to create children chromosomes. Usually higher performing chromosomes are preferred as parents to create better performing children. A few chromosomes are eliminated to keep the population size constant.

1) Crossover & Mutation: Crossover involves taking a few features of both parents in order to create a new child chromosome. Mutation on the other hand, assigns a random value to a certain feature of the chromosome. This ensures the diversity of the population.

2) Selection Schemes: Parents are selected using the following selection schemes. Our objective is to select higher performing chromosomes as parents (those with higher fitness values/scores) to get closer to an optimal solution with each generation. However, each scheme must have a degree of randomness involved to ensure the space is explored properly. The following selection schemes were used:

- Truncation; wherein best fitness parent are selected
- Hybrid; similar to truncation but based on a certain probability, we sometimes select a parent with low fitness.
- Tournament Select; the parent with higher fitness out of two is selected.

3) Survival Schemes: A similar concept applies for survival schemes. The following survival schemes were used:

- Truncation; wherein lower fitness chromosomes are removed from the population.
- Random Select; which involves randomly selecting chromosomes to be eliminated.
- Binary Tournament Select; the chromosome with lower fitness out of two is removed.

B. Neural Networks

Neural networks consists of multiple layers and are trained over time to provide us with an expected output for a given set of inputs.The input layer consists of nodes that are equal in number to the number of inputs in the input data set. The output layer outputs the probability that a given output is likely to occur. It may also be configured to provide a range of outputs. The number of layers between the input and the middle layer depends on the number of variables computed in order to reach our end state. An activation function, usually the sigmoid function is used to provide an output. Each edge between nodes existing in different layers has a weight associated with it. The weight indicates the degree of affect a node will have on our output and over time, is configured to best match our expectations of the system. A bias may also be included in our calculations. In our algorithm, each bird-agent has a personal neural network associated with it. Using 5 different parameters, the bird decides whether to fly upwards or not when it is a certai distance away from a pipe-gap.

C. The Neuroevolution Algorithm

Our genetic algorithm takes the weight of one bird agent and uses crossover with the weights of another bird-agent to create a new bird-agent that has a neural network with weights from its parents. There is no bias involved within our algorithm. Conmbining these two algorithms is what gives us our final, proposed, neuroevolution algorithm.

II. The Algorithm

A. Setting Up The Environment

The game window is a simple 640×480 window. Each pipe is a simple pygame rectangle. Our population consists of ten birds, each with its own neural network. Every bird starts out with a velocity of zero, while a gravitational field with gravitational field strength of 0.08 acts on each bird.

B. Game Mechanics

The neural network tells each agent to fly after calculating whether it is feasible to do so. When it is, each agent gets an upward velocity of 3. Newton's equations of motion of displacement and velocity are used to then calculate the resultant displacement and velocity of each bird-agent.

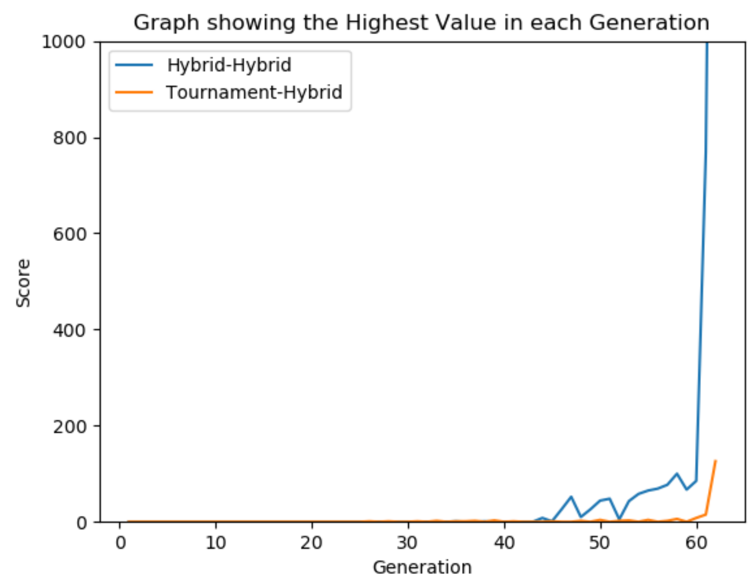
C. Neuroevolution

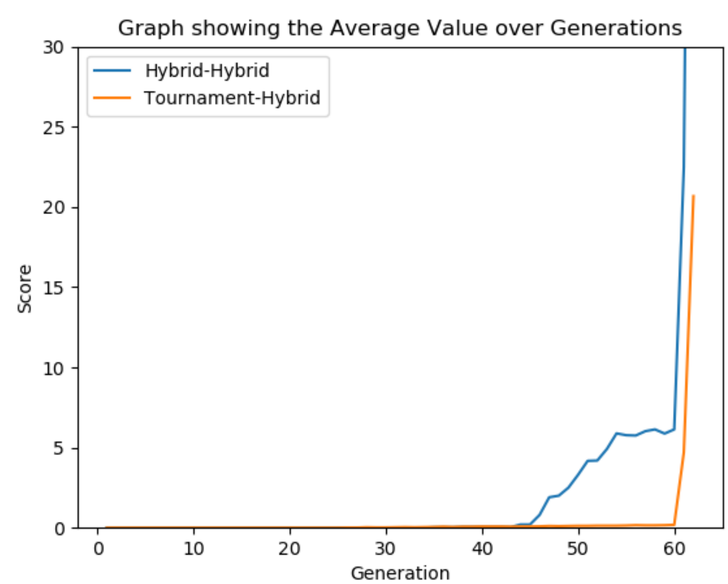
The neural network has 4 layers including the the input and the output layer. Each layer, except for the output layer is initialized with random weights and every one of these layers contain 5 nodes each. The 5 nodes in the input layer represent the bird's position, its velocity, the dimensions of the gap that it is approaching, and the position of said gap. The sigmoid function is used as an activation function. Mutation occurs with a certain probability by replacing any of the weights with a random new value.

The population contains 10 bird-agents, each with its own neural network. The fitness of each bird is evaluated on the basis of the number of frames it has survived in the game. After a set of parents have been selected through the above specified selection schemes, crossover takes place between two of the parents. The process of crossover is carried out by randomly selecting a weight from either the father or the mother chromosome. This process is repeated until an entire new neural network can be formed. This neural network is then assigned to the new child bird-agent. Using the survival schemes defined above, a new population is formed. The new population may consist of parent+children bird-agents, or it may select a new generation from the entire pool of chromosomes.

III. Data and Analysis

We have first fixed the survival selection scheme, such that the previous generation is overthrown by parents of previous generation and children of the current generation. We will run our algorithm for the different parent selection schemes which we have implemented and will compare the results. Then we will fix the parent selection scheme(hybrid) and will run the algorithm for different survival selection scheme. We will run the algorithm 5 times for 200 generations each time and will take average of the outcome. The results of these runs are shown in a table towards the end of the section. Below is a graph comparing two schemes,the hybrid-hybrid, and the Tournament-Hybrid scheme, where the scheme mentioned first is the selection scheme and the later is the survival scheme.





As can be seen above, the hybrid-hybrid scheme far outweighs any other. The first graph shows how the best scoring agent evolves with each generation while the second displays the change in average values over generations.

The following table contains a summary of all data collected and was used to compare pairing of different selection and survival schemes.

Sel.Scheme	Sur. Scheme	Avg Score	Best Score
H	H	70.12	2401
TR	H	62.58	1863
TM	H	0.42	56
H	H	70.12	2401
H	R	0.24	21
H	TM	13.65	114

Where H stands for Hybrid
Where TR stands for Truncate
Where TM stands for Tournament
Where R stands for Random

IV. Proposed Future Work

In this section we will discuss further work which can be done on the basis of this paper. One of the approach which we can use to solve the flappy bird game is by implementing deep reinforcement learning. In reinforcement/Q learning we have multiple states in which our agent exist, and can perform multiple actions in a particular state. The agent gets reward if a good action is performed, or penalty if a bad action is performed. Through several iterations, we are able to determine a policy on which the agent can act, by looking at its track record of award/penalty. Coming back to our example, here state would be the position of the bird. At every state there would be only two possible actions either to jump or not. We just have to make a trail for the bird for few consecutive state, lets assume for 3 states. So we will see that if a bird is in a state and it decides to jump, then will it get reward(pass pipe) or penalty(collide with pipe or floor) in the next 3 states. It would be a good comparison to make between the deep reinforcement learning and Nero evolution, and to see that which one gives better result.

V. Conclusion

Neuroevolution algorithms are an efficient way of combining two algorithms in order to make use of the best of both. Data Analysis shows that the Hybrid-Hyrid scheme pairing is the one that performs best over time, while a hybrid-random scheme performs the worst. We beleive that in this case it is best to adopt a slightly more exploitative approach, as finding even a local maxima allows our bird agent to pass through incoming pipes.

Acknowledgment

The authors would like to thank Dr. Saleha Raza, our professor, for her help throughout the course and for providing us with the opportunity to work on this project.

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.