Making use of a Genetic Algorithm, the program was made to teach a bot how to play the Flappy Bird  
game and survive as long as possible.  
Expect from the game building block, which was assembled via Pygame, the fully connected neural  
network architecture and the genetic learning algorithm were constructed from scratch.  
This was achieved by mutated each bird weight by a random offset and crossover the best birds from a  
given a flock of each generation.

Flappy Bird is a mobile game that was proudly developed by a Vietnamese game maker. It has grown into somewhat of an intro project for new computer science students.

To be honest, building an AI that learns to play a simple game will not have any real-world application. However, I find this project is a great first step into reinforcement learning, an area that is not my forte. The final result was very satisfying, I can spend hours watching my “children” find success.

My implementation is a simpler version of NEAT, inspired by the idea of mutation and crossover. The algorithm in this project is my own take on the 2 mentioned concepts with my existing knowledge of Neural Networks.

Result

The following is an analyst of the result. More technical details can be found by scrolling down.

The results show that the model is powerful enough to convert the game. In some lucky instances, the game can be beaten at only 23 generations.

Result

Fitness graph

Viewing from the clip, we can see that the first few generations are truly a disaster. All members of the flock jump up and down with no indication of direction or strategy. However, Some lucky ones pass through the first pair of pipes

These genetic get to pass down the line and help more birds to defeat the pair.

General

Statistic

Since the game is run on many random variables. I believe

Game

A virtual environment is needed to train the reinforcement agent, in other words, the game. The game is built using Pygame as a foundation, this provides a free environment for the project since the whole pipeline is implemented through Python. By making the game using the same language, the AI can interact directly with the program with ease.

This is my first homemade game, fortunately, tutorials are abundant for this intro program on the internet, one of the reasons I choose this game to dive into the genetic algorithm.

The Clear Code youtube channel has a great tutorial on this. Following his instruction and assets, I was able to make a replica within a day.

https://www.youtube.com/watch?v=UZg49z76cLw&ab\_channel=ClearCode

Sorry if my code is a bit messy for anyone looking through them ☹.

Here is some of the adjustable variable to balance the game difficulty:

• The distance from the above and below the pipe

• The time interval between two spawning pairs of pipes.

• The gravitational force on the birds.

• The jumping force of the birds.

With these parameters, we can test out the limitation of our learning algorithm with different difficulties.

The original NEAT algorithm is complicated and very effective in training over the generation. It only generates additional nodes when needed, allows the model to involve increasingly complex over the generation, and remains lightweight.

The general idea behind the following algorithms is that each bird will have a fully connected Neural Network as a “brain”. Birds are selected using their fitness score within a flock, and learn using 2 methods, mutation and cross-over.

The Neural Network

This is the controlling unit of each bird, the deciding factor in the game. Each brain is created through my implementation of fully connected NN, the file call NN.py can be found on the Github link.

The following is the default parameter for each bird:

* 5 number of input
* 2 number of output ( jump or not to jump)
* 1 hidden layer
* 6 nodes in each hidden layer
* Tanh activation function

Each of these parameters can be changed through the input variable. ( vẽ NN)

The input

The bird vision contains 5 views:

* Distance to the closest pare of pipe (X-axis)
* Distance to the top pipe (Y-axis)
* Distance to the bottom pipe (Y-axis)
* The bird current Y-coordinate
* How fast the bird is falling or jumping up

Mỗi point 1 hình vẽ.

Each of these inputs is normalized the game parameter before being passed to the Neural Network.

The fitness

Fitness is a way to measure how long a bird can survive through the environment, calculated using the clock on the game. Each given generation contains a flock of 100 birds, the 10 top bird is selected base on their fitness to carry on to the next generation. These top 10 birds will be called 10 types.

Mutation

Mutation is the ability of the bird to change its genetic by itself through generations. This is achievable by randomly (mutation rate) modify the weight of each bird by a small margin (mutation margin).

The default value for mutation rate is 30%, and mutation margin is a random value of a normal distribution of mean 0 and standard deviation of 0.1

A loop is created to go through each bird weight matrix. At each weight, a random variable is generated to decide the weight should mutate or not. The action will be taken if the mentioned variable is smaller than 0.3.

Cross-over

Cross-over is a method for the top birds to combine their genetic to produce offspring, that potentially be better than the parents. My home-made cross-over algorithm contains 3 main components :

1. Replication
2. Pairing
3. Cross-swap

With the combination of the three steps, hopefully, the algorithm successfully mimics the heredity process of nature.

Replication

From each round, the top 10 birds with the highest fitness are selected. Each bird will be replicated 10 times using deep copy. This produces a new flock of 100, 10 of each type of bird.

Each of these birds is exactly the same as the original with no differences.

Pairing

Now since we have a new flock of 100 birds, 10 of each, the next task is to pair them with each other to create “families”. The method is simple, but a bit hard to explain.

First, one of each type is reserved to keep the original top 10 birds for the next generation. This is because the process of mutation and cross-over can cause the loss of useful genetic material from the flock.

The reservation step can keep that useful information for later on.

The second step boiled down to the problem of generating unique pairs from 90 birds. We can view the flock of birds as a 9x9 matrix. The algorithm simply pairs the n row with the n column.

With this method, we have successfully pair 90 birds.

Cross-swap

Cross-swap provides a way for a pair of birds to exchange information. It archives this by randomly swapping the neurons of two birds.

Again the weights of 2 neural networks can be view as 2 matrixes. There is a 40% chance (swap rate) that an individual weight from the first matrix will be swapped the weight in the same coordinate from the second matrix.

The swap rate is default and can be changed.