HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

ARTIFICIAL INTELLIGENCE PROJECT REPORT

Project: Self-playing Flappy Bird AI with Neural Network and Genetic Algorithm

LECTURER: Huong Thanh Le

TEAM: 21

No	Full Name	Student ID	Class
1	Phi Hoang Long	20184288	ICT 01 – K63
2	Vu Tung Lam	20184283	ICT 01 – K63
3	Nguyen Van Luc	20176812	ICT 01 – K62

Hanoi, 11/2019

Table of Contents

INTRODUCTION

Development

Flappy Bird is a mobile game developed by Vietnamese video game artist and programmer Dong Nguyen (Vietnamese: *Nguyễn Hà Đông*), under his game development company dotGears.

The bird character was originally designed in 2012 for a cancelled platform game. The gameplay was inspired by the act of bouncing a ping pong ball against a paddle for as long as possible. Initially the game was significantly easier than it became in the final version. However Nguyen said he found this version to be boring and subsequently tightened up the difficulty. He described the business plan of a free download with in-game advertisements as "very common in the Japanese market".

Gameplay

Flappy Bird is a side-scrolling mobile game featuring 2D retro style graphics. The objective is to direct a flying bird, named "Faby" who moves continuously to the right, between sets of Mario-like pipes. If the player touches the pipes, he lose.

Faby briefly flaps upward each time that the player taps the screen; if the screen is not tapped, Faby falls because of gravity.

Each pair of pipes that he navigates between earns the player a single point.

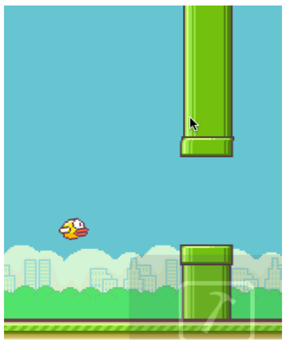


Fig 1. Flappy Bird interface

ABOUT THE PROJECT

Task environment

- Performance measures: safely dodging all obstacles.
- Environment: bird location, top pipes, bottom pipes, ground (fully observable; single learning agents).
- Actuators: bird movements.
- Sensors: screen touches.

Problem statement

1. **Inputs**

- Bird’s height.
- Top pipe’s height, Bottom pipe’s height, Gap width.

2. **Outputs**

Bird’s continuous decisions whether to jump or not, each time the agent gets the full information of randomly created inputs.

3. **Goal**

Teach the AI bot to self-play the game until it is unbeatable.

For the optimal strategy, we will use Neural Network for bird’s self-decisions and Genetic Algorithm for faster goal reaching by regenerating a population of birds in each generation.

Problem representation

1. **Graphic (Inputs)**

In Flappy Bird game, we use 3 classes representing the bird, the pipes and the ground, which later will be used for inputs information and graphic design. In the following subsection, we just focus on the main inputs used for the algorithm.

a) Bird

For the bird character, its horizontal position is kept unchanged while its height is initialized first then will depend on the number of screen touches.

- Coordinate in Oxy: X (horizontal initial position - fixed)   Ex: 150,
  Y (vertical initial position - unfixed).

b) Pipes

The pipes in the game will be scrolled back so that it looks like the bird is flying forward. Input for the pipes will be its gap width and height.

- Gap width: fixed Ex: 120
- Pipe height: random range Ex: 100-300
- Top pipe height = Pipe height – the height of the PIPE loaded image
- Bottom pipe height = Pipe height + Gap width

Note:

Losing condition: Bird location matches either a pipe location with no gap or the ground/sky.

2. Other variables

- Generation
- Score
- Clock
- Fitness index

- Initial state: a population of birds in the initially-set location, which will also be regenerated in the next generation after all of its predecessors are all dead.
- Final state: a number of birds survived that will keep playing infinitively after a process of learning from neural network.

Algorithms and Heuristics

1. Neural Network

In programming, artificial **neural network** is computational model/algorithm for machine learning that is inspired by structure and functional aspects of biological neural networks.

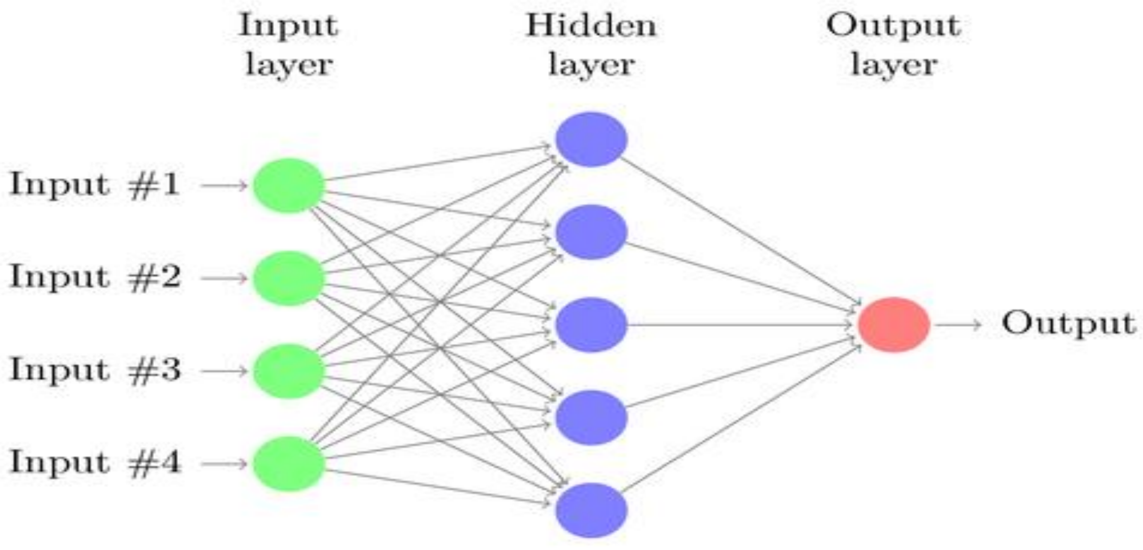


Fig 2. An example of a Neural Network

- Every neural network has one input layer, one output layer and one or more hidden layers.
- Each circle represents a neuron and a neuron has a connection to every neuron in the next layer.
- Each connection has weight value and each neuron has a bias value.

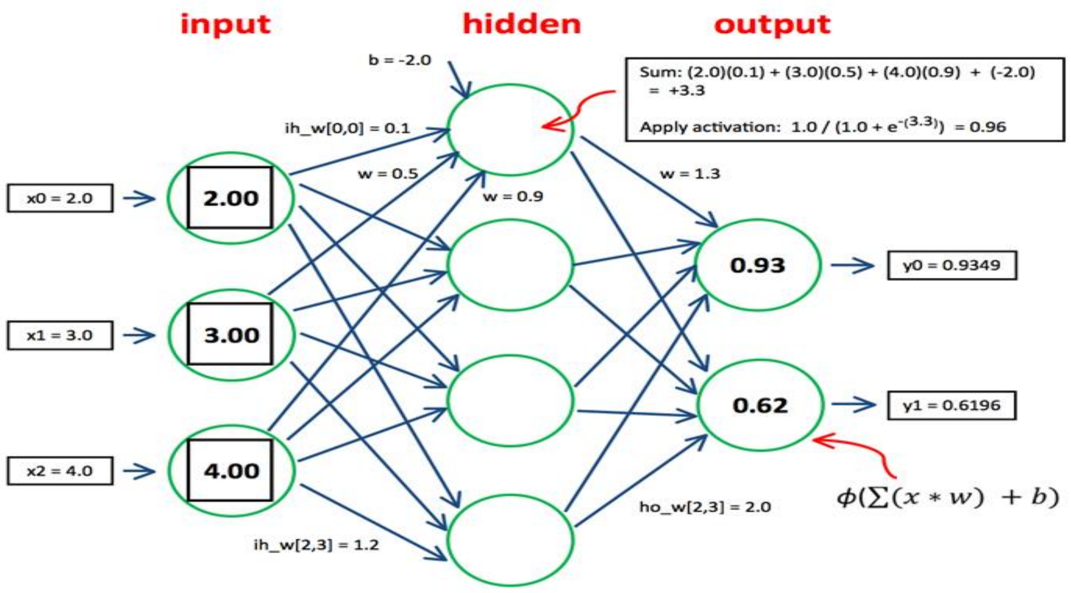


Fig 3. An example of what's happening in the Neural Network

When neural network calculates the output, it does a lot of math with weight and bias using aggregation function and activation function.
If we change one of the weights or bias values, the final output will also be changed.
In other words, training neural network means finding and adjusting weight and bias values that gives the best output that we want.

2. Genetic Algorithm

Genetic algorithm is a search heuristic that is inspired by Charles Darwin's theory of natural evolution. This algorithm reflects the process of natural selection where the fittest individuals are selected for reproduction in order to produce offspring of the next generation.

The process of natural selection starts with the selection of fittest individuals from a population. They produce offspring which inherit the characteristics of the parents and will be added to the next generation. If parents have better fitness, their offspring will be better than parents and have a better chance at surviving. This process keeps on iterating and at the end, a generation with the fittest individuals will be found.

This notion can be applied for a search problem. We consider a set of solutions for a problem and select the set of best ones out of them.

***Five phases** are considered in a genetic algorithm:

a) Initial Population

The process begins with a set of individuals which is called a **Population**. Each individual is a solution to the problem you want to solve.

An individual is characterized by a set of parameters (variables) known as **Genes**. Genes are joined into a string to form a **Chromosome** (solution).

b) Fitness Function

The **fitness function** determines how fit an individual is (the ability of an individual to compete with other individuals). It gives a **fitness score** to each individual. The probability that an individual will be selected for reproduction is based on its fitness score.

c) Selection

The idea of **selection** phase is to select the fittest individuals and let them pass their genes to the next generation.

Two pairs of individuals (**parents**) are selected based on their fitness scores. Individuals with high fitness have more chance to be selected for reproduction.

d) Crossover

Crossover is the most significant phase in a genetic algorithm. For each pair of parents to be mated, a **crossover point** is chosen at random from within the genes.

Offspring are created by exchanging the genes of parents among themselves until the crossover point is reached and then they will be added to the population in the next generation.

e) Mutation

In certain new offspring formed, some of their genes can be subjected to a **mutation** with a low random probability. This implies that some of the bits in the bit string can be flipped.

Mutation occurs to maintain diversity within the population and prevent premature convergence.

***Pseudocode:**

```
START
Generate the initial population
Compute fitness
REPEAT
    Selection
    Crossover
    Mutation
    Compute fitness
UNTIL population has converged
STOP
```

Implementation

1. Neural Network

In our work, each unit (bird) has its own neural network which is used as its Artificial Intelligence brain for playing the game. It consists of 3 layers as follows:

• **Input layer.** An input layer three neurons shows the agent's perspective and represents what it sees:

- Bird Height
- Top pipe location
- Bottom pipe location

• **Hidden layer.** Number of neurons in the hidden layer were varied from 1, 5,10 and 20 and the performance of each was studied.

• **Output Layer.** An output later with one neuron which provides an action of flap if output>0.5 else do nothing.

2. Genetic Algorithm

The main steps of our genetic algorithm implementation:

Step 1: Random neural network helps in creating the initial population of 20 birds.
Step 2: Each unit has its own neural network which helps them to play autonomously.
Step 3: Fitness function is calculated for each bird separately.
Step 4: When all die, evaluate the current population to the next.
Step 5: Go back to the second step.

To be simple, we design fitness function as following principles:

- In the first generation, initialize all birds in the population with a fitness of 0.
- Each time a bird passes a pair of pipes, increase its fitness by 0.1.
- Each time a bird dies but then will still be chosen for the next generation, decrease its fitness by 1.

Replacement Strategy:

- Keep 2 birds with the highest fitness and pass them directly to the next generation
- Choose randomly the rest of the population with a probability for each bird of 20% to mutate and/or crossover for the next generation to add some variations

3. List of classes and functions

a) Class:

Bird: Represent birds with their states, action.

Pipes: Represent pipes with their size, position, motion and status whether collision or not.

Base: Represent the ground with its position and motion.

b) Function:

draw_window(win, birds, pipes, base, score, GEN, pipe_ind): Draw a whole window with birds, pipes, base and other parameters

eval_genomes(genomes, config): Evaluate a genome with passed configuration.

neat.nn.FeedForwardNetwork.create(g, config): Receives a genome and returns its phenotype.

neat.nn.FeedForwardNetwork.activate(inputs) : Feeds the inputs into the network and returns the resulting outputs.

Full code link : https://github.com/drag-pi/AIteam

Result

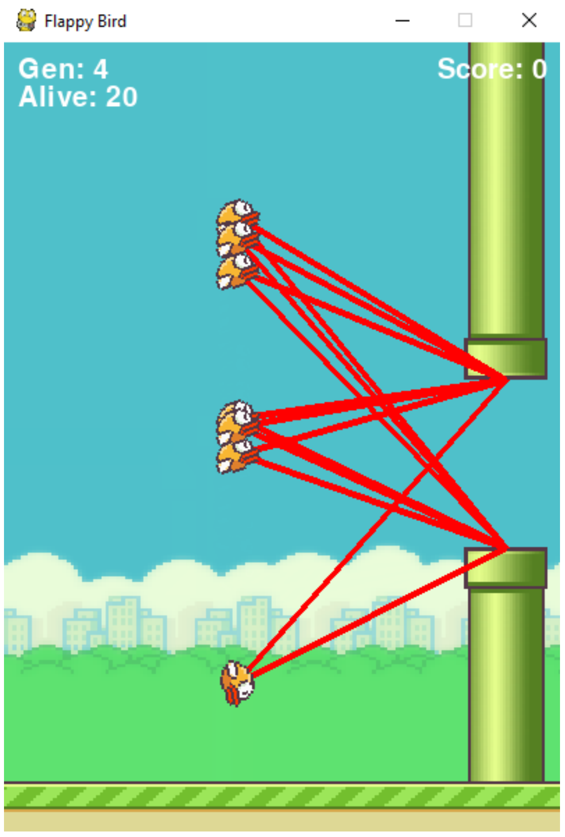
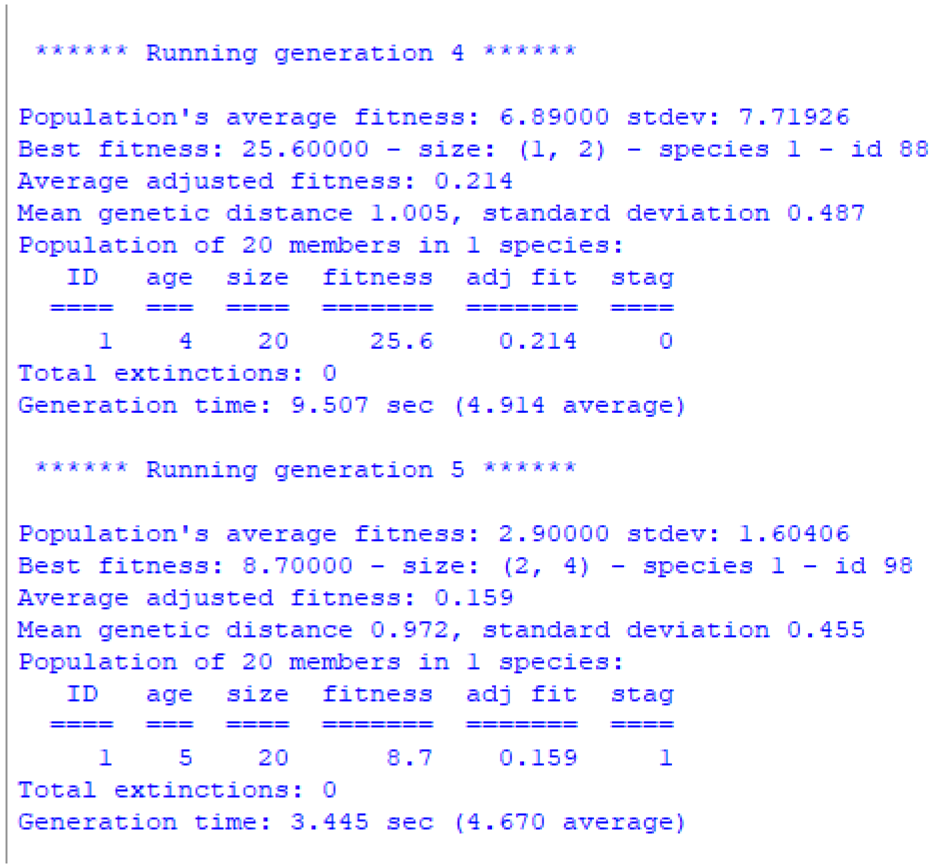


Fig 4. Game interface sample



```
 ****** Running generation 4 ******

Population's average fitness: 6.89000 stdev: 7.71926
Best fitness: 25.60000 - size: (1, 2) - species 1 - id 88
Average adjusted fitness: 0.214
Mean genetic distance 1.005, standard deviation 0.487
Population of 20 members in 1 species:
   ID   age  size  fitness  adj fit  stag
  ====  ===  ====  =======  =======  ====
     1    4    20     25.6    0.214     0
Total extinctions: 0
Generation time: 9.507 sec (4.914 average)

 ****** Running generation 5 ******

Population's average fitness: 2.90000 stdev: 1.60406
Best fitness: 8.70000 - size: (2, 4) - species 1 - id 98
Average adjusted fitness: 0.159
Mean genetic distance 0.972, standard deviation 0.455
Population of 20 members in 1 species:
   ID   age  size  fitness  adj fit  stag
  ====  ===  ====  =======  =======  ====
     1    5    20      8.7    0.159     1
Total extinctions: 0
Generation time: 3.445 sec (4.670 average)
```

Fig 5. Game statistics report

Comments

1. About results:

- The AI bot successfully reaches the immortal state
- No anticipations could be made about at what generation the AI bot is intelligent enough **(randomly)**
- APIs are difficult to use because of light overuse of the given module (neat), also the module is not fully developed yet.

2. About UI

- No representations or graphs of neural network and genetic algorithm made for deeper understanding
- Interface is still too simple, only used for research

REFERENCES

1. *Lectures of Artificial Intelligence*, Prof. Le Thanh Huong, Hanoi University of Science and Technology

2. Flappy Bird introduction, https://en.wikipedia.org/wiki/Flappy_Bird

3. Module Neat-python for Neural Evolution and Genetic Algorithm, https://neat-python.readthedocs.io/en/latest/index.html

4. Neural Network introduction, Woojae Kim, https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3

5. Genetic Algorithm introduction, Vijini Mallawaarachchi, https://towardsdatascience.com/neural-network-plays-flappy-bird-e585b1e49d97

6. Flappy Bird AI tutorial, Tech With Tim, https://github.com/techwithtim/NEAT-Flappy-Bird