AI FLAPPY BIRD USING NEAT

Sharan Ram M

UG Scholar

Rajalakshmi Engineering College

AR.Sharvesh

UG Scholar

Rajalakshmi Engineering College

Abstract

This paper explores the application of the NeuroEvolution of Augmenting Topologies (NEAT) algorithm to the classic game Flappy Bird, aiming to develop an AI that can autonomously learn to play the game through evolutionary computation techniques. NEAT, a form of neuroevolution, evolves artificial neural networks (ANNs) by optimizing both the network weights and topologies, facilitating the discovery of increasingly complex and efficient structures over successive generations. In this study, we initialize a population of simple neural networks and subject them to the Flappy Bird environment, where their performance is evaluated based on the distance traveled and obstacles navigated. Through iterative processes of mutation, crossover, and selection, the networks evolve to exhibit enhanced gameplay strategies. Key results demonstrate that NEAT can effectively evolve ANNs that achieve high levels of proficiency in Flappy Bird without human intervention. The evolved networks exhibit behaviors such as precise timing for flapping actions and anticipation of future obstacles, akin to strategies employed by human players. The study highlights the robustness of NEAT in handling the dynamic and unforgiving nature of Flappy Bird, showcasing its potential for broader applications in game AI development and real-time decision-making systems. The success of NEAT in this context underscores its capability in creating adaptable and efficient neural architectures for complex control tasks.

1. INTRODUCTION

NeuroEvolution of Augmenting Topologies (NEAT) is an innovative evolutionary algorithm developed by Kenneth O. Stanley and Risto Miikkulainen. NEAT distinguishes itself by evolving both the connection weights and the topologies of neural networks, allowing it to create increasingly complex and efficient network structures over time. Traditional neuroevolution techniques typically operate with fixed-topology networks, focusing solely on optimizing connection weights. NEAT overcomes this limitation through a dynamic approach that incrementally complexifies the networks by adding new nodes and connections.

A key feature of NEAT is its use of speciation to maintain diversity within the population. By grouping similar networks into species, NEAT protects innovative topologies from premature elimination, fostering the exploration of novel network structures. Another critical component is the use of historical markings, which track gene origins and ensure proper alignment during the crossover process, preserving the functionality of inherited structures.

NEAT's ability to evolve both network structure and weights has proven effective in a wide range of applications, from game playing and robotics to control systems and pattern recognition. Its approach facilitates the discovery of optimal network architectures for complex problems, making NEAT a powerful tool in the field of neuroevolution and machine learning.

2. LITERATURE SURVEY

Self-Learning Car Simulation Using NEAT:

- This research project focuses on self-driving car development using NEAT. Self-driving cars have the potential to revolutionize transportation by improving road safety and reducing traffic congestion.
- The simulation environment is created using Unity, a popular game engine. The self-driving car learns from its experiences, adapts to new environments, and enhances its performance over time.
- The NEAT algorithm optimizes the car’s performance by evolving the neural network’s structure and weights.

NEAT in Robotics (Stanley and Miikkulainen, 2004):

- **Study Overview**: The study titled "Competitive Coevolution through Evolutionary Complexification" applied NEAT to robotic control tasks.
- **Methodology**: NEAT was employed to evolve neural networks for controlling robots in complex environments.
- **Findings**: The evolved neural networks demonstrated advanced control capabilities, adapting effectively to the complexities of their environments.
- **Implications**: This research showcased NEAT's applicability in robotics, highlighting its potential in developing adaptive and efficient control systems for real-world applications.

Clune et al. (2011):

- **Focus**: Comparison of NEAT with other encoding methods for evolving neural networks in robotic control.
- **NEAT Performance**: Highlighted NEAT's robustness and efficiency in evolving adaptable and scalable network structures.

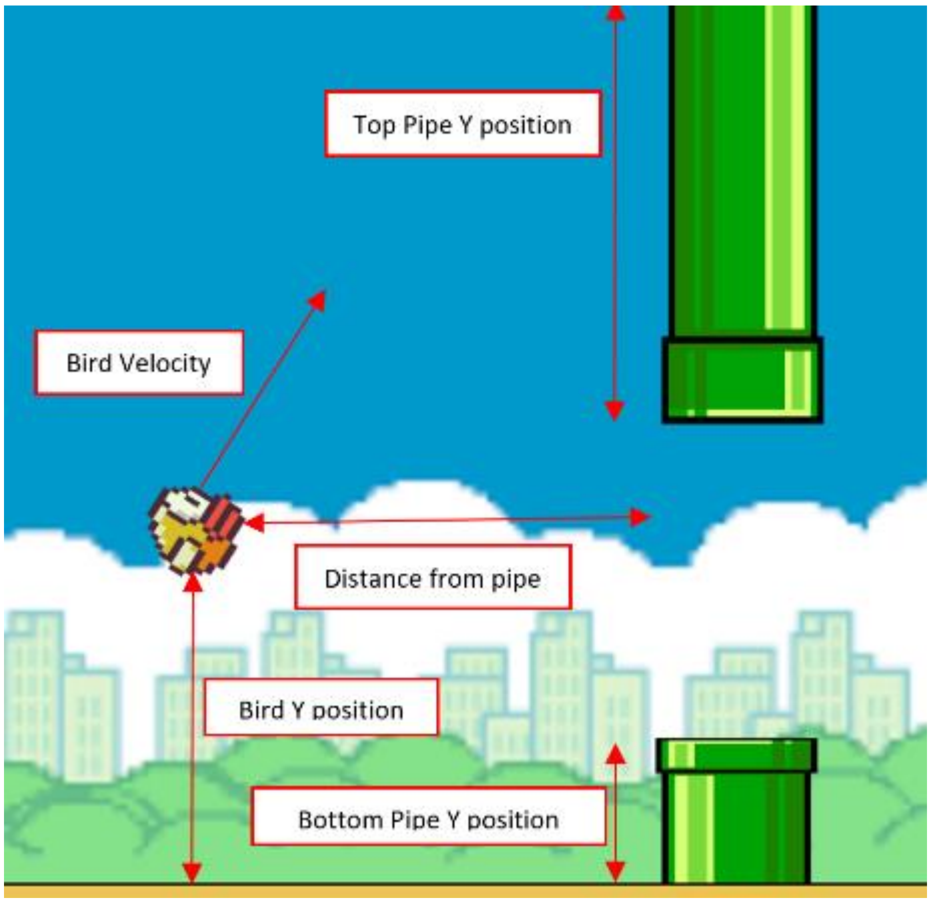
- **Implications**: Signifies NEAT's suitability for real-world applications requiring adaptive and efficient control systems in robotics.

3. Model architecture

The model architecture of an AI playing Flappy Bird using NEAT typically involves the following components:

3.1 Input and Output Layers

Inputs are derived from the bird's current y-position, the distance to the top and bottom pipes. A single output neuron decides whether the bird should jump or not based on the activation value. It's common to normalize these inputs to a range between 0 and 1. Normalization ensures that the neural network receives consistent input values regardless of the actual game dimensions, making it easier to learn patterns and behaviors.

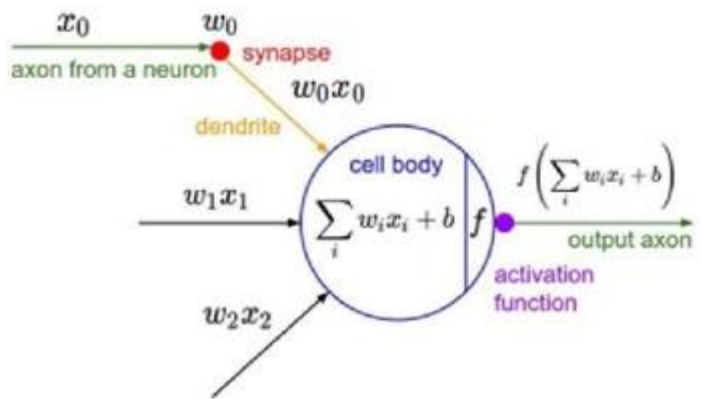


3.2 NEAT Parameters:

1. **Fitness Function**: The fitness function in the context of using NEAT to train an AI to play Flappy Bird is crucial for guiding the evolutionary process. It determines how well each individual in the population (each genome, representing a neural network) performs in the game.The function will provide a fitness score based on the bird's performance.

2. **Selection process:** The NEAT algorithm employs powerful selection process to drive the evolution of the AI agents. These operators ensure the most successful neural networks are propagated, while introducing beneficial mutations and crossovers to explore new solutions.

3. **Mutation:** Adding nodes can create new pathways for information processing, potentially leading to more sophisticated decision-making. Removing nodes can simplify the bird's brain, reducing computational complexity.



3.3 Activation Function:

The activation layer, often referred to simply as the output layer in the context of neural networks, is a critical component of the AI model architecture for Flappy Bird using NEAT. This layer is responsible for making the final decision based on the processed inputs. The activation layer typically consists of a single neuron. This neuron outputs a value that determines the bird's action: to flap or not to flap.

Tanh: The tanh (hyperbolic tangent) activation function is another common choice for the activation layer in neural networks. It maps input values to a range between -1 and 1, which can be beneficial for certain types of problems, including reinforcement learning tasks like Flappy Bird.

$$\mathrm{tanh}(x)=1+e-2x2-1$$

3.4 Reproduction:

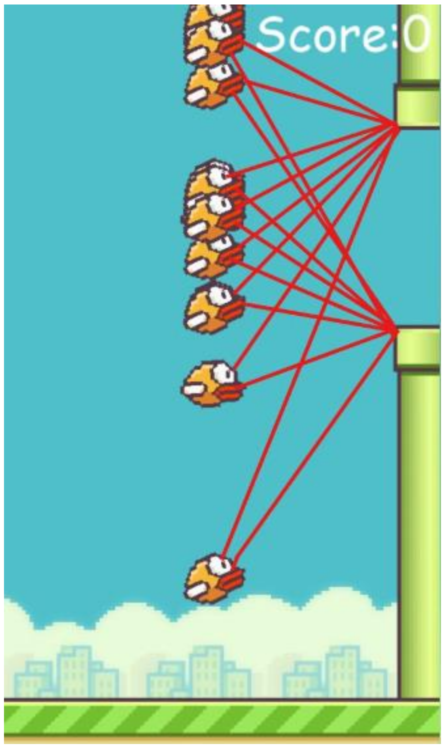
After the mutation and crossover the best performing birds or genome are allowed to bred to produce a desirable offspring. Less fit individuals are replaced by these off-springs and then these off springs go through the same process for number of times.

4. Implementation Methods

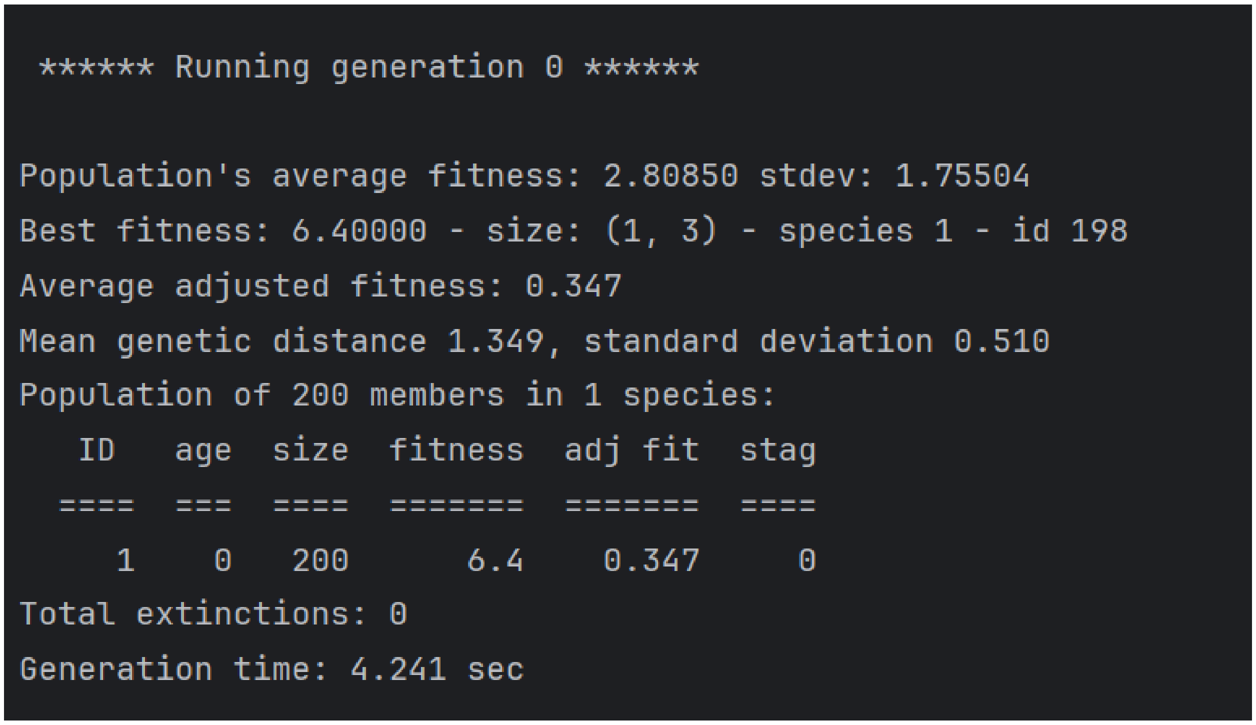
The system requirements for training and evaluating models are 16 GB of RAM, an integrated GPU, and an Intel Core i5-7700 CPU running at 3.60 GHz.

5.RESULTS

This is the visual representation of how each genome decides when it sees a pipe



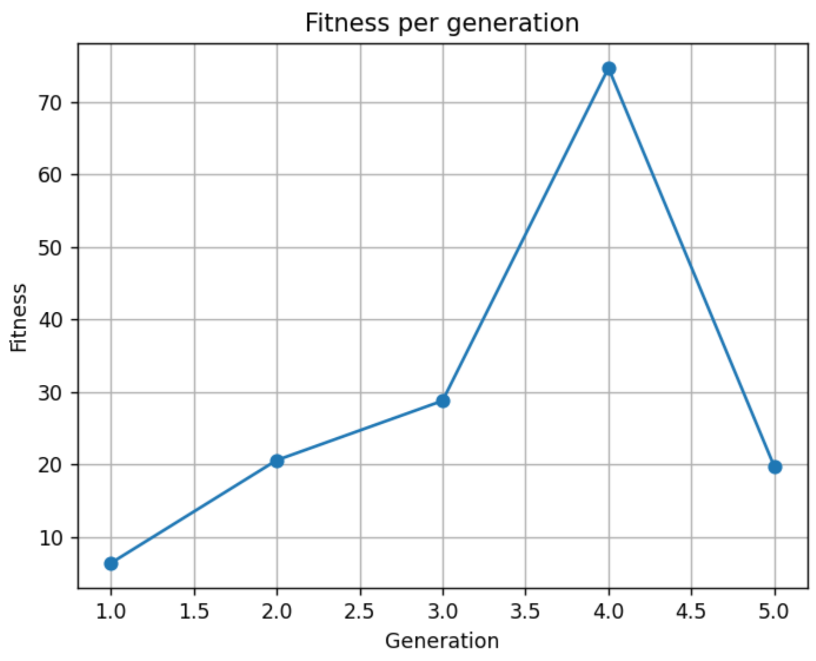
Details about each generation:



```
 ****** Running generation 0 ******

Population's average fitness: 2.80850 stdev: 1.75504
Best fitness: 6.40000 - size: (1, 3) - species 1 - id 198
Average adjusted fitness: 0.347
Mean genetic distance 1.349, standard deviation 0.510
Population of 200 members in 1 species:
   ID   age  size  fitness  adj fit  stag
  ====  ===  ====  =======  =======  ====
     1    0   200      6.4    0.347     0
Total extinctions: 0
Generation time: 4.241 sec
```

Graphical Representation of each generation and their fitness score



6. CONCLUSION

The conclusion of using NEAT (NeuroEvolution of Augmenting Topologies) for training an AI to play Flappy Bird is multifaceted, reflecting both the achievements and potential areas for further exploration. Here's a comprehensive summary of the conclusions drawn from employing NEAT in this context:

1. **Autonomous Gameplay**: NEAT enables the development of an AI agent capable of autonomously playing Flappy Bird without explicit programming. The AI learns to navigate the game environment, avoiding obstacles and maximizing its score through evolutionary processes.

2. **Adaptability**: NEAT's ability to dynamically evolve neural network structures allows the AI to adapt and improve its performance over successive generations. The AI learns effective strategies for playing the game, evolving increasingly sophisticated behaviors

3. **Complexity Management**: NEAT effectively manages the complexity of neural networks by starting with simple structures and gradually increasing complexity as needed. This approach optimizes computational resources and prevents overfitting to the training data.

REFERENCES

1. Stanley, Kenneth O., and Risto Miikkulainen**.** "Evolving Neural Networks through Augmenting Topologies." *Evolutionary Computation* 10.2 (2002): 99-127.

- This seminal paper introduces the NEAT algorithm and details its methodology and effectiveness in evolving neural networks.

2. Clune, Jeff, Kenneth O. Stanley, Robert T. Pennock, and Charles Ofria**.** "On the Performance of Indirect Encoding Across the Continuum of Regularity." *IEEE Transactions on Evolutionary Computation* 15.3 (2011): 346-367.

- This research compares indirect encoding methods, including NEAT, demonstrating the strengths and weaknesses of different approaches in evolving neural networks.

3. Gauci, Jason, and Kenneth O. Stanley**.** "Autonomous Evolution of Topographic Regularities in Artificial Neural Networks." *Neural Computation* 22.7 (2010): 1860-1898.

- The paper discusses advancements in NEAT and its ability to autonomously evolve neural network structures with topographic regularities, relevant for spatial and visual processing tasks.