IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Neuroevolution – Training An Ai Agent In Dynamic Environments

[1] K.Vanithasri , [2] U.Aiman Rabiya , [3] A.Farnaz Sulthana

1Assistant Professor, Department of Computer Science and Engineering, Aalim Muhammed Salegh College Of Engineering (Affiliated to Anna University), Chennai - 600055, Tamil Nadu, India.

2,3Department of Computer Science and Engineering, Aalim Muhammed Salegh College Of Engineering (Affiliated to Anna University), Chennai - 600055, Tamil Nadu, India

Abstract - The field of artificial intelligence (AI) is constantly on the lookout for algorithms that can effectively handle complex problems in dynamic environments. Neuroevolution, a powerful technique that merges the capabilities of neural networks (NNs) and genetic algorithms, has emerged as a promising solution. This project applies neuroevolution to train an autonomous agent for Flappy Bird. NNs receive data on the bird's position and obstacles. The process initializes a population with random movements, evaluates performance, selects the fittest, and uses their genetic material for a new generation. This iterative process continues until proficient birds emerge. The project demonstrates neuroevolution's effectiveness in training agents for dynamic environments. By combining NNs' learning with genetic algorithms' optimization, it offers a promising approach for complex tasks. Results show the genetic algorithm's ability to optimize behavior over generations, enabling birds to learn and adapt, achieving significant performance improvements. Although focusing on Flappy Bird, these principles can apply to robot control, autonomous navigation, and game design. Neuroevolution in game design can create challenging, engaging content, blurring lines between human and AI creativity.

Keywords – Artificial Intelligence, Neural Networks, Flappy Bird, Autonomous Agent, Optimization

I. INTRODUCTION

The field of artificial intelligence (AI) is constantly seeking innovative techniques to tackle complex problems in dynamic environments. This project explores the application of neuroevolution, a powerful technique that merges the capabilities of neural networks (NNs) and genetic algorithms, to train an autonomous agent to play the challenging mobile game Flappy Bird. By integrating reinforcement learning principles, agents can optimize their behavior using genetic algorithms, exploring a vast search space for optimal strategies. By leveraging neuroevolution's ability and combining with evolutionary algorithms such as neural networks, this project aims to train an AI agent to learn, adapt to navigate successfully through the game's complex environment. This project is motivated by the goal of training Al agents to play games autonomously.

Flappy Bird, with its challenging mechanics requiring precise timing and coordination, provides an interesting problem for Al training. This research investigates the effectiveness of neuroevolution in a dynamic gaming environment, aiming to unlock its potential for broader AI applications.

Flappy Bird offers an ideal platform for AI research due to its unique characteristics:

Dynamic Environment: The game presents a dynamic environment where the bird's position and the approaching pipes are constantly changing, requiring the AI agent to

adapt its strategy in real-time.

Precise Control: Success hinges on the ability to make split-second decisions and execute precise adjustments to the bird's flight path.

Continuous Learning: Unlike games with discrete levels, Flappy Bird requires the AI agent to continuously learn and improve its decision-making throughout the ongoing gameplay.

Neural networks excel at processing information and making informed decisions, mimicking the human brain's structure and function. Reinforcement learning principles, where agents learn through trial and error, allow neural networks to optimize their behavior within a specific environment. This combination provides a powerful framework for training AI agents to learn and adapt within game worlds.

Neuroevolution, the core technique of this project, merges the strengths of neural networks and genetic algorithms. Inspired by natural selection, it allows a population of neural networks to evolve over generations. Networks performing well based on a defined fitness function are more likely to be selected for reproduction, passing their successful traits to the next generation. This iterative process leads to progressively improved neural networks, enabling them to tackle increasingly complex tasks.

II. PROBLEM STATEMENT & OBJECTIVES

The problem addressed in this research is training an Al agent to play simplified version of the Flappy Bird game using neuroevolution and a geneti algorithm. The challenge lies in developing an Al-controlled bird that can navigati through a series of obstacles, such as pipes, by optimizing its movement patterns. The goal is to evolve a generation of birds that demonstrate proficiency in playing the game, showcasing the effectiveness of the genetic algorithm approach in training Al agents for complex tasks.

The objectives of this research project are as follows;

A. Apply Neuroevolution and Genetic Algorithms: Implement neuroevolution principles and genetic algorithms to train Al agents to play the Flappy Bird game.

This involves combining neural networks with genetic algorithms an utilizing evolutionary processes to optimize the Al behavior.

B. Evolve Improved Al Behavior: Evolve a population of Al controlled birds over successive generations, aiming to improve their gameplay performance Through the process of selection, mutation, and crossover, seek to discover and propagate more effective strategies and behaviors.

C. Evaluate Training Effectiveness: Assess the effectiveness of the genetic algorithm approach in optimizing Al behavior and achieving proficiency it playing Flappy Bird.
Analyze the performance of the evolving generations of birds and track their progress in terms of gameplay metrics, such as distance.

III. EXISTING SYSTEM

Traditional methods for training AI agents to play Flappy Bird often rely on supervised learning.
This approach involves training a single AI agent ("bird") through the following steps:

A. Data Provision: The agent receives game state data as input. This data typically includes information about the bird's position, the location of the pipes, and other relevant game elements.

B. Action Selection: The agent selects an action (e.g., jump or stay still) based on the provided data and its internal model.

C. Performance Feedback: The agent receives a reward or penalty depending on its performance (e.g., distance traveled, pipes cleared).

D. Model Adjustment: The agent's internal model (typically a neural network) is iteratively updated based on the received feedback.

However, supervised learning for Flappy Bird has limitations:

Limited Exploration:

This approach relies solely on the experience of a single agent. This can restrict the exploration of alternative strategies and limit the ability to adapt to unseen game variations. The agent may become stuck in a "locally optimal" solution, performing well on the training data but failing to generalize to unseen scenarios.

Single Agent Bottleneck:

The reliance on a single agent hinders the exploration of a broader range of strategies. Supervised learning doesn't allow for the exchange of knowledge and adaptation based on the performance of other agents within the game environment. These limitations highlight the need for alternative approaches that can overcome the challenges of supervised learning in complex and dynamic environments like Flappy Bird.

IV. PROPOSED SYSTEM

This project breaks away from traditional methods by employing a neuroevolutionary approach with a genetic algorithm. Here's the key difference:

A. Beyond a Single Bird: Instead of training a solitary bird, we leverage a population of diverse birds. This allows for exploration of a wider range of strategies within the Flappy Bird environment.

B. Evolving Through Generations: A genetic algorithm plays a central role. It evaluates each bird's performance and selects the most successful ones (parents) to reproduce the next generation. This "survival of the fittest" approach ensures that strong traits influencing successful gameplay are passed on.

C. Strength in Numbers: By promoting diversity and variation within the population, this method fosters inter-bird learning and adaptation over generations. This collaborative approach holds promise for achieving superior gameplay performance compared to supervised learning with a single bird.

D. Iterative Learning and Improvement: The process of evaluation, selection, and variation is repeated over multiple generations. As generations progress, the population evolves towards superior performance. The birds within the population learn from each other and adapt their strategies, leading to progressively better performance in navigating the challenges of Flappy Bird

V. SYSTEM DESIGN ARCHITECTURE & FLOW

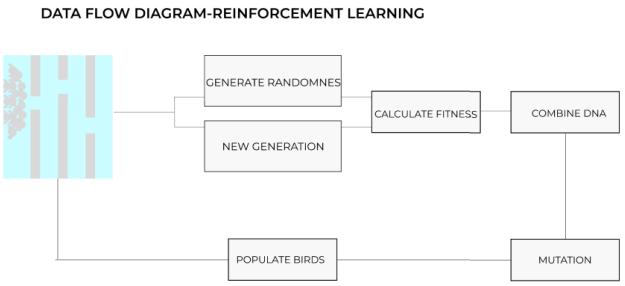

Figure 1. Data Flow Diagram – Reinforcement Learning

The genetic algorithm approach is a key component of the neuroevolution process used to train the Al agent in playing Flappy Bird. It leverages the principles of evolution, selection, and genetic variation to iteratively improve the Al agent's performance. Here is an explanation of the genetic algorithm approach;

A. Initialization:

The genetic algorithm starts by creating an initial population of Al agents with random genetic representations. Each agent is associated with a neural network that controls its decision making in the game.

B. Evaluation:

The fitness of each Al agent is evaluated by simulating their gameplay in the Flappy Bird environment. The fitness function considers factors such as the distance covered, the number of pipes cleared, or the survival time. The fitness score quantifies how well an Al agent performs in the game and serves as a measure of its success.

C. Selection:

The selection process aims to choose the fittest individuals from the current population to form the next generation. Fit individuals have a higher chance of being selected, while less fit individuals have a lower chance. Various selection strategies can be employed, such as tournament selection or roulette wheel selection, to determine which individuals are chosen to proceed to the next generation.

D. Genetic Variation:

Genetic variation ensures diversity in the population and introduces exploration and exploitation. This process involves applying genetic operators like mutation and crossover.

E. Mutation:

Mutation randomly alters the genetic representation of an Al agent, introducing small changes to its neural network parameters. This stochastic process enables exploration of new solutions and helps prevent premature convergence to suboptimal solutions.

F. Crossover:

Crossover Involves combining genetic information from two parent Al agents to create offspring. This process mimics the reproduction and inheritance mechanisms observed in natural evolution. Different crossover techniques, such as single-point crossover or uniform crossover, can be utilized to exchange genetic material between parent agents.

G. Reproduction and Generation Update:

The new offspring, resulting from mutation and crossover, replace a portion of the less fit individuals in the current population. This reproduction and replacement process ensures the population evolves towards better performance over generations. The updated population becomes the next generation, and the evaluation, selection, and genetic variation steps are repeated.

By iteratively repeating the evaluation, selection, and genetic variation steps, the genetic algorithm optimizes the Al agent's genetic representation over multiple generations. This process drives the Al agent towards improved performance and the ability to play Flappy Bird more effectively.

The admin, or administrator, is an authorized person responsible for creating elections, setting the election time, and adding candidates to the voting contract.

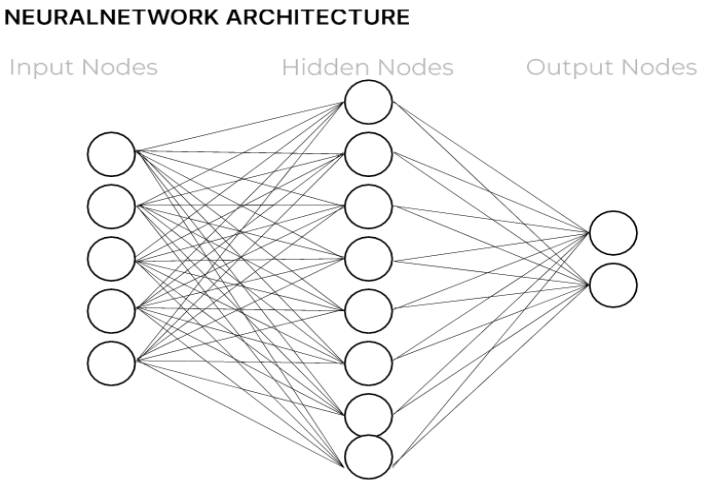

Figure 2. Neural Network Architecture

The neural network architecture used for controlling the Al agent in the Flappy Bird game consisted of 5 input nodes, 8 hidden nodes, and 2 output nodes. Here is a breakdown of the architecture:

A. Input Nodes:

There were 5 input nodes that represented the key features or states of the game environment. These input nodes provided relevant information to the neural network for making decisions during gameplay. Examples of the input nodes could include the bird's vertical position, the distance to the nearest pipe, and the bird's current velocity.

B. Hidden Nodes:

The neural network had 8 hidden nodes, forming a hidden layer between the input and output layers. The hidden nodes performed complex computations and transformations on the input data, extracting meaningful patterns and representations.

C. Output Nodes:

There were 2 output nodes that determined the Al agent's actions in the game. Each output node represented a possible action, such as flapping or not flapping, enabling the Al agent to control the bird's movement. The neural network architecture described above allows the Al agent to process the input data, learn from the game environment, and make decisions based on the output values.

The hidden nodes serve as intermediaries, enabling the neural network to capture and learn complex relationships between the input and output layers.

It's important to note that the specific number of input nodes, hidden nodes, and output nodes can vary depending on the requirements of the game and the complexity of the problem. Feel free to adjust and customize these numbers based on your specific implementation.

SYSTEM IMPLEMENTATION

The Flappy Bird game was developed from scratch using the p5.js library, incorporating custom physics, collision detection algorithm, and random pipe generation. Here is an overview of the implementation details:

A. Game Engine and Graphics:

The game was implemented using the p5.js library, which provides a simple and intuitive framework for creating interactive web-based games. Custom graphics and animations were designed to recreate the visual elements of the original Flappy Bird game, including the bird, pipes, background, and game over screen.

B. Physics and Motion:

Custom physics were developed to handle the bird's motion, gravity, and interaction with the game environment. The bird's movement was controlled using keyboard input or mouse clicks, simulating the flapping motion to navigate through the pipes.

C. Collision Detection:

A custom collision detection algorithm was implemented to detect collisions between the bird and the pipes. The algorithm ensures accurate detection and handles collisions by determining whether the bird has intersected with any pipe or if it has hit the ground or ceiling.

D. Random Pipe Generation:

A random pipe generation algorithm was devised to create a continuous stream of pipes with varying heights and gaps. The algorithm determines the position and dimensions of each pipe, providing a dynamic and challenging gameplay experience.

Testing Data:

Comprehensive testing was conducted to evaluate the performance of the trained Al agent in playing the Flappy Bird game. The testing phase included the following:

1. Performance Evaluation: Metrics such as pipes cleared, total score, and average gameplay duration were recorded to assess the agent's proficiency.

2. Generalization Testing: The agent was tested on unseen levels to evaluate its ability to adapt to novel game scenarios.

3. Sensitivity Analysis: Modifying game conditions tested the agent's flexibility and responsiveness to changes in pipe speed, gravity, and gap size.

4. Comparison with Baseline: Performance was compared with a baseline model to quantify improvement achieved through the training process.

Testing provided insights into the agent's performance, adaptability, and generalization capabilities. Results contribute to the overall assessment of the Al agent's competence in playing the Flappy Bird game

VI. RESULTS

The training process using the genetic algorithm and neuroevolution yielded promising results in training the Al agent to play the Flappy Bird game.
Key observations and insights include:

A. Learning Progression: Initial generations exhibited random movements, but subsequent generations improved gameplay skills, navigating through pipes more successfully.

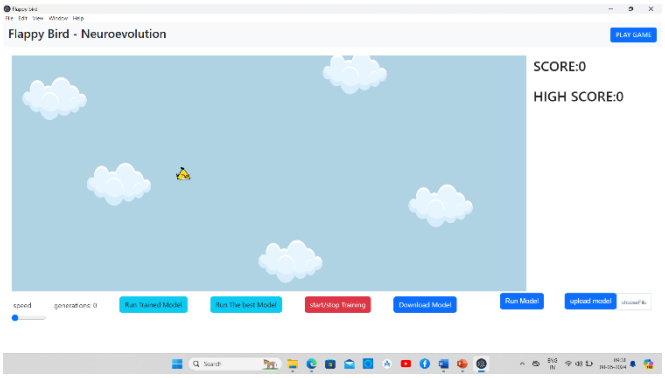

Figure 3. Initial Generation

B. Fitness Evaluation and Selection: The fitness evaluation function assessed agent performance, selecting better-performing agents as parents for the next generation. Selective breeding led to a steady improvement in the Al agent's performance.

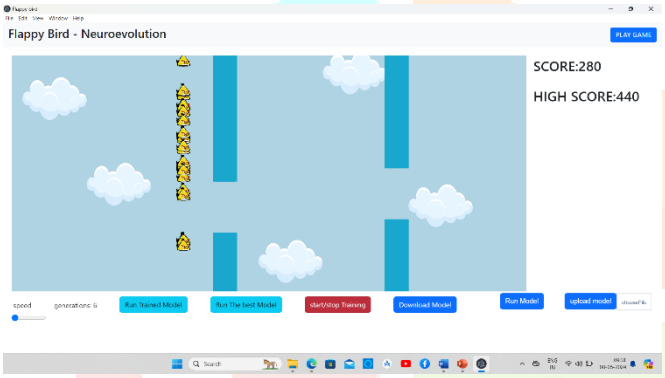

Figure 4. Fitness Evaluation

C. Genetic Diversity and Exploration: Genetic mutations introduced diversity, exploring new strategies and behaviors. Certain mutations led to significant advancements, while others had minimal impact.

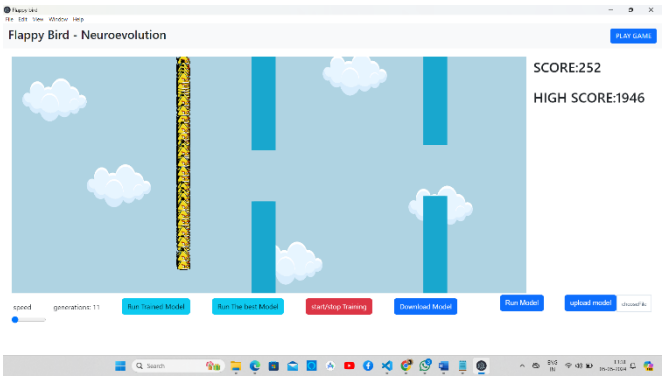

Figure 5. Genetic Mutations

D. Training Convergence and Limitations: Training converged towards an optimal solution, but performance reached a plateau due to neural network architecture and algorithm limitations.

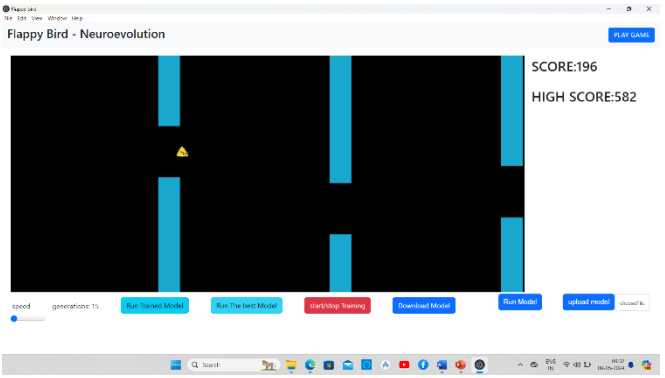

Figure 5. Trained Model

E. Generalization and Transfer Learning: The trained Al agent generalized its skills, performing well on unseen Flappy Bird game levels, suggesting transfer learning potential.

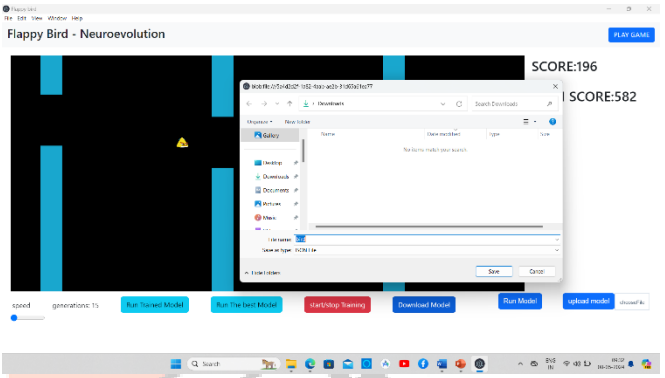

Figure 6. Transfer Learning

These results highlight the effectiveness of the genetic algorithm and neuroevolutionary approach. Further research is needed to enhance algorithm parameters, network architecture, and fitness evaluation. Scaling up to complex games and real-world scenarios will provide broader insights.

VII. CONCLUSION

In In this project, a neuroevolutionary approach using a genetic algorithm was successfully employed to train an Al agent to play the Flappy Bird game. The implemented neural network architecture, with 5 input nodes, 8 hidden nodes, and 2 output nodes, facilitated the agent's decision-making process based on learned patterns and representations. The experimental results demonstrated the effectiveness of the genetic algorithm-based training process, showcasing the evolution of generations of Al agents and their improved

performance.

This project contributes to the understanding of neuroevolution and genetic algorithms in Al training for gaming applications.

While further enhancements can be explored, such as optimizing genetic algorithm parameters and refining fitness evaluation functions, this project establishes the potential of neuroevolution and genetic algorithms for training Al agents in complex game environments. The success achieved opens avenues for future research in applying similar techniques to more challenging games or real-world problems.

In conclusion, this project exemplifies the capability of neuroevolution and genetic algorithms to train Al agents in playing games, providing valuable insights into the intersection of Al and gaming.

REFERENCES

[1] Verena Heidrich-Meisner, Christian Igel, (2009). Neuroevolution strategies for episodic reinforcement learning.

[2] Shiffman, D. (2012). The Nature of Code: Simulating Natural Systems with Processing. Self-published.

[3] Stanley, K. O., & Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. Evolutionary Computation, 10(2), 99-127.

[4] Floreano, D., & Mattiussi, C. (2008). Bio-inspired artificial intelligence: Theories, methods, and technologies. MIT Press.

[5] Kureychik V.V., Kureychik V.M., Rodzin S.I. The theory of evolutionary computation. Moscow: Fizmatlit, 2012. 260 pages

[6] Tsoy YU.R., Spitsyn V.G. The evolutionary approach to the setting up and training of artificial neural networks // Neuroinformatics. 2006. Vol 1. № 1. pp. 34-61