An Implementation of Neural Network with Evolutionary Algorithm in Gaming Application

Manisha (M.tech 2nd Year)
Er. Lovely Malik
Assistant Professor
Department of Computer Science and Engineering,
Ganpati Institute of Technology & Management, Bilaspur(Yamunanagar)

ABSTRACT

This paper focusses on Artificial Intelligence which is recently growing in many areas. We see various developments for technical systems, image processing and multimedia solutions. We can find applications of intelligent technologies for smart management of energetic systems and simulation models in thermal sciences also gaming industry is using various aspects of intelligent technologies. There are many interesting system models which support interactions in real time. Also computer systems support interesting application of gamification ideas. Among techniques of artificial intelligence very high impact is visible from evolutionary methods. Algorithms are based on various models of life from animals, insects or other. Systems which use various models seen in nature are used in image processing to track points of interest over images. Artificial intelligence (AI) techniques have been proven highly successful at the problems of navigation, task prioritization, and problem avoidance. Traditionally, humans have encoded rule-based AIs to create the behavior necessary to allow an automaton to achieve a specific task or set of tasks. Genetic programming (GP), however, has been proven to allow a computer to create human-competitive results. This thesis report is about artificial intelligence implementation in the famous game- "Snake". This application is implemented with the help of artificial intelligence by using Genetic Algorithm and Artificial Neural Network.

The Genetic Algorithm (GA) is a stochastic global search method that mimics the metaphor of natural biological evolution. Genetic Algorithms operate on a population of potential solutions applying the principle of survival of the fittest to produce (hopefully) better and better approximations to a solution. At each generation, a new set of approximations is created by the process of selecting individuals according to their level of fitness in the problem domain and breeding them together using operators borrowed from natural genetics. This process leads to the evolution of populations of individuals that are better suited to their environment than the individuals that they were created from, just as in natural adaption.

Keywords: |**Artificial neural network, evolutionary algorithm, evolutionary design, evolutionary training.**

1. INTRODUCTION

The initial inventors and founding fathers like —Alan Turing and all others were very much inspired by the very basic vision of implanting computer programs with initial stage of very basic intelligence which can have the ability to self−replicate. The initial programs aim to infuse the features of logic

adaptability as its one of core processing logic. These initial steps made these programs capable to learn and to regulate their surrounding atmospheres. These early research communities and program developers imitated routine tasks and several scientific and mathematical problems by inculcating logical intelligence into the programs. Thus they charted the initial path to achieve their visions. So it is very noteworthy that from the very early days computers were considered and applied not for data crunching operations only but also to carry out tasks which adheres activities of brains and related learning and decision making process. This is like very closely following biological evolution. These biologically inspired computing activities were slow then after for some years as computer systems went through very daunting phases of resource developments and up gradation in terms of compute power, storage etc., but since the early 1980s research started harnessing the mentioned capabilities for research centered, intelligence demanding computational tasks. Computer scientists used various nature-proven, biology-inspired and complexity capable methods like neural networks and machine learning to solve these problem domains.

In that evolving tradition, a new dimension has joined which is known as "evolutionary computation" family. The family has many members like genetic programing, genetic algorithms and Evolution Strategies.

simultaneously computational parallelism is the need of an hour. In this case many processors need to be evaluated sequences at the same time and an appropriate intelligent tactic should be employed for selecting the very succeeding set of sequence.

1.1 EVOLUTIONARY COMPUTATION

The transformative calculation is a mark adhered to various calculations motivated by Darwin's hypothesis of organic development by normal determination, and furthermore to the logical field that reviews these calculations. What exactly degree these calculations really imitate the components of organic advancement differs uncontrollably, yet most PC researchers will in general negligence the natural understanding of transformative calculations and treat them just as worldwide analyzers. Seen from this point of view, developmental calculations are hugely flexible, as they can enhance any issue as long as the arrangement can be communicated as a grouping of parameters, and the wellness (nature) of an applicant arrangement can be estimated rapidly and dependably. Contrasted with other enhancement methods, practically zero space information is required, however, better outcomes can, for the most part, be accomplished while implanting some area information.

The requirement of several problems is very important that it has to perform well in varying environment and continue to give "good" and "desirable" results. [1] So in a nutshell a computer program has to accommodate the surrounding changes and need to show very prompt learning and assessment making behavior. A new algorithm paradigm has to chart out systematic approach for realizing a computational task or even a new scientific unearthing. This leads to solve many high complexity problems using a novel branch of computer engineering called Artificial Intelligence.AI experts do believe that it would be immediate and easy to encode the principles that would present knowledge on a program

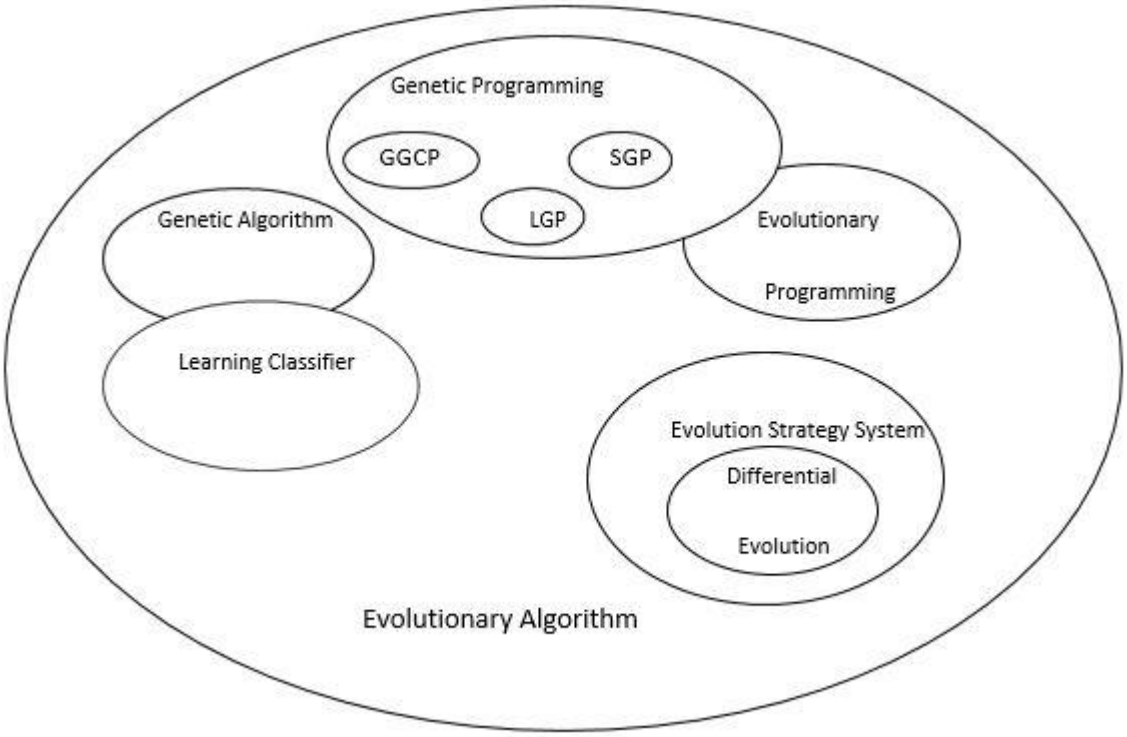
1.2 THE FAMILY OF EVOLUTIONARY ALGORITHM

The family of evolutionary algorithms encompasses five members, as given below:

1. Genetic algorithms (GAs). GAs subsumes all evolutionary algorithms which have bit strings as search space G.
2. The set of evolutionary algorithms which explore the space of real vectors $X \subseteq Rn$ is called evolution strategies.
3. For Genetic Programming (GP) there is provision of two definitions: GP includes all evolutionary algorithms that grow programs, algorithms, and these alike. Second, all EAs that evolve tree-shaped individuals are instances of Genetic Programming.
4. Learning Classifier Systems (LCS) are online learning approaches which assign output values to given input values. They on the inside use a genetic algorithm to find new rules for this mapping.
5. Evolutionary programming (EP) is an evolutionary approach that treats the occurrences of the genome as different species rather than as individuals. Over the decades, it has more or less merged into Genetic Programming and the other evolutionary algorithms.

Usually the EA are divided in four key groups, presented from the oldest to the most current in the following subsections:

- Evolutionary Programming
- Evolution Strategy
- Genetic Algorithm
- Genetic Programming



2 RESULT AND PLAYING THE GAME

2.1 EVOLUTIONARY ALGORITHM

Each snake in the evolutionary race has its own food to feed on, so there is no competition in the feeding case, the only competition, the selection criterion, is that the snake which fed the most gets to propagate more. Hence, each SnakeInstance has a Snake, FoodGenerator and a BackpropogationNetwork, snakeNet of its own. To mutate the snake as it propagates, the class has a method MutateNetwork() and a method to reproduce by copying other snake's neural weights, by the method 'ReproduceNetwork()'.

Food Generator is another class which randomly generates food inside the canvas, which is a of 600*600, which also generates lines to be used by RefreshCanvas() to update them in the UI. How it's being done is open to inquisition in the code provided with this article, because the main purpose is the explanation of the implementation of the Evolutionary Algorithm.

In the timer Tick event handler in the *MainWindow.cs*, we write the logic for the game to run, where in succession, first, the snake's sensors are updated, the sensor values are then fed into a neural network and the network's output is calculated. The outputs are then compared for snake to take the decision regarding whether to MoveForward(), TurnLeft() or TurnRight().

```
void timer_Tick(object sender, EventArgs e)
{
   foreach (SnakeInstance snake in evolution.SnakeInstances.Where(r=>r.IsDead == false))
   {
      snake.snake.FoodSensor.ReadSensor(snake.snake, snake.food.Food.First());
      if (PlayNetwork.IsChecked == true)
      {
         snake.snakeNet.ApplyInput(new double[]
         {
snake.snake.FoodSensor.Front, snake.snake.FoodSensor.Back,
         snake.snake.FoodSensor.Left, snake.snake.FoodSensor.Right });
         snake.snakeNet.CalculateOutput();

         IEnumerable<double> outputs = snake.snakeNet.ReadOutput();

         if (outputs.ToArray()[0] > Math.Max(outputs.ToArray()[1], outputs.ToArray()[2]))
            snake.snake.MoveForward();

         if (outputs.ToArray()[1] > Math.Max(outputs.ToArray()[2], outputs.ToArray()[0]))
            snake.snake.TurnLeft();

         if (outputs.ToArray()[2] > Math.Max(outputs.ToArray()[1], outputs.ToArray()[0]))
            snake.snake.TurnRight();
```

```
        }
    }
    RefreshCanvas();
}
```

When a snake dies by touching itself, touching the wall or by exhausting itself, an event, SnakeDead is fired, which is used to update a variable IsDead in our SnakeInstance, this variable is used in the evolutionary process. Now, many instances of this snake are initialized and simulation runs until all the snakes are dead and then sorted according to the number of times the snake fed, and the genes (or the neural weights) are propagated and mutated according to the code, which is written in the event handler to the SnakeDead event.

Hence, the simulation is run and re-run, and the result plotted out into another window. Here is a plot of the fitness parameter of different generations.

Generation 1: In the initial generation i.e. generation 1 as we can see length of the snake is 6.

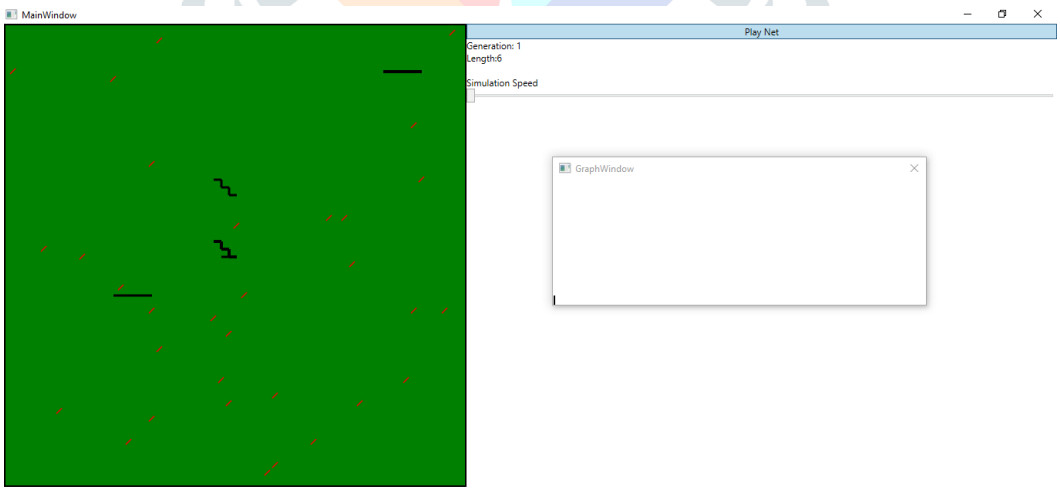


Fig. 2.1. Generation 1 of the snake

Generation 5: In the generation 5, we can see length of the snake is increased to 10.

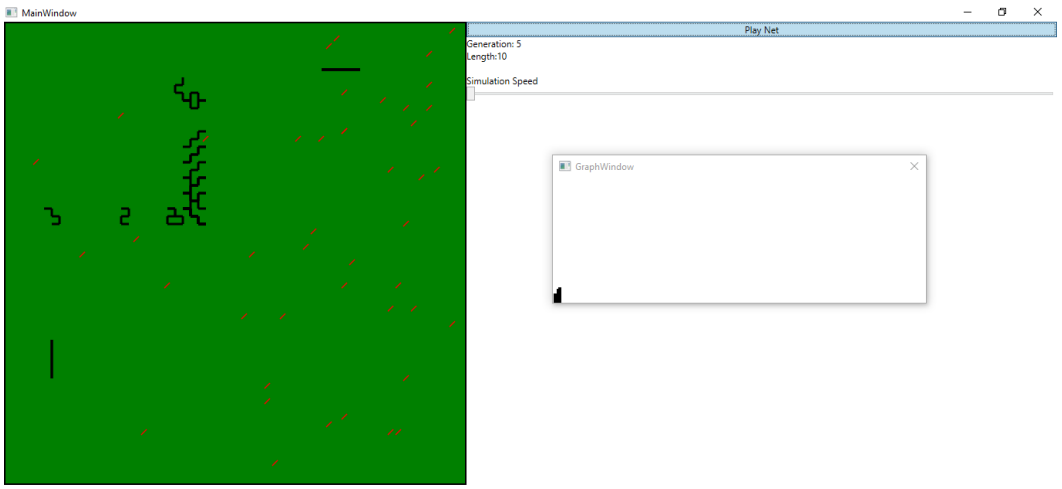


Fig. 2.2. Generation 5 of the snake

Generation 7: In the generation 7, we can see length of the snake is increased to 11.

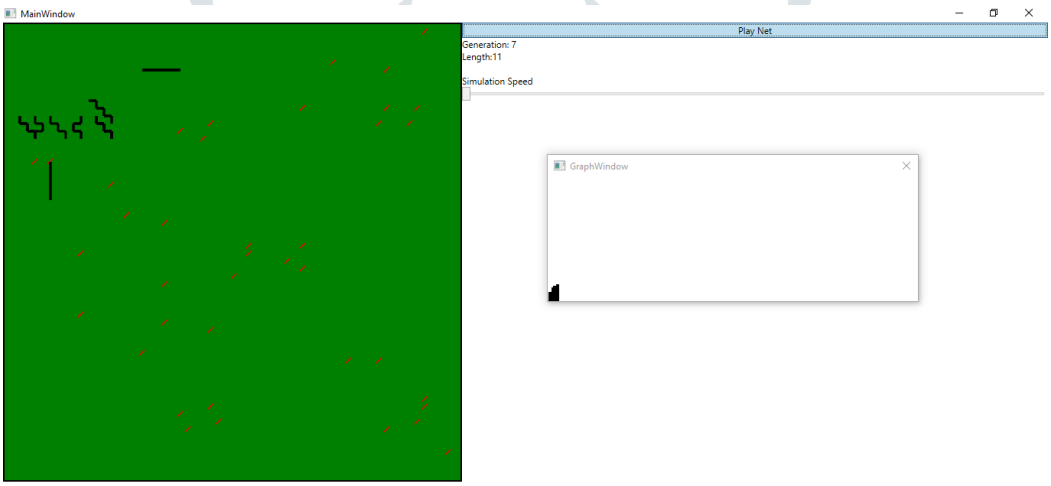


Fig. 2.3. Generation 7 of the snake

Generation 21: In the generation 21, we can see length of the snake is increased to 28.

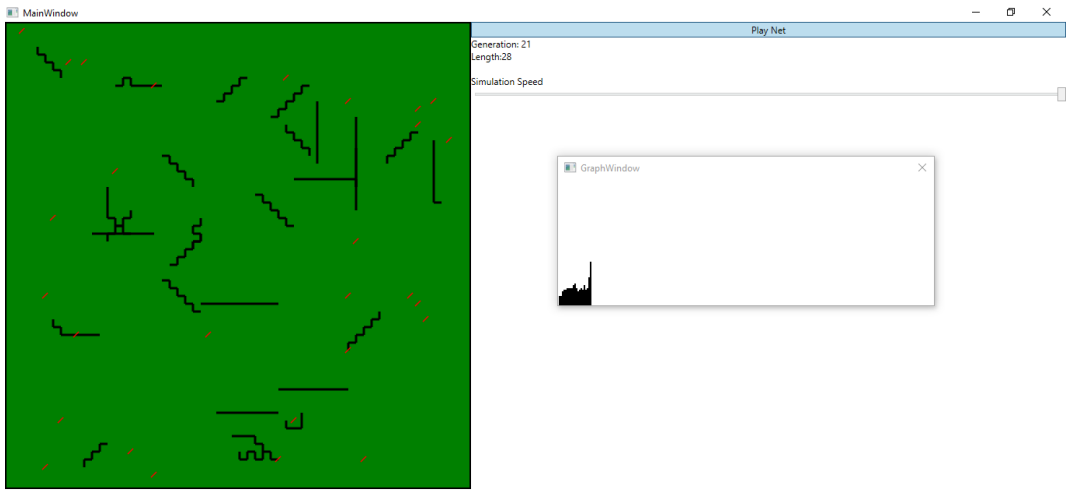


Fig. 2.4. Generation 21 of the snake

3.1 CONCLUSION

In this study, we have successfully implemented the use of artificial intelligence in the famous game – Snake. This gaming application is successfully carried out by using “Genetic Algorithm” and “Artificial Neural Network”.

In this, we had successfully plotted the fitness parameter of different generations of the snake. Fitness parameter of snake against its generation is represented with the help of graph which clearly shows the oscillation of length of the snake in each generation.

From this implementation we can conclude that fitness of the snake is increasing rapidly initially, then oscillates due to random nature of the food generation. The algorithm used in this implementation is improved version of previous work carried out by others, as our algorithm deals with a better selection procedure which leads to better fitness parameter to offspring so that it provides improved results generation after generation.

4.1 FUTURE SCOPE

We have seen Artificial Intelligence is recently growing in many areas. We see various developments for technical systems, image processing and multimedia solutions and in many more areas. The succinct discussion in this dissertation shows that, in spite of the large efforts to improve the previous algorithm carried out by others, the result of our algorithm is better than the previous one, but still there is a chance to improve this algorithm.

We can expect to achieve hundred percent accuracy in this field, because there is always a space for improvement.

REFERENCES

1. Fundamentals of Artificial intelligence: Russell, Stuart J.; Norvig, Peter (2009). Artificial Intelligence: A Modern Approach (3rd ed.). Upper Saddle River, New Jersey: Prentice Hall.
2. Fundamentals of Artificial intelligence: Neapolitan, Richard; Jiang, Xia (2018). Artificial Intelligence: With an Introduction to Machine Learning. Chapman & Hall/CRC.
3. Fundamentals of Genetic Algorithms: https://en.wikipedia.org/wiki/Genetic_algorithm
4. Fundamentals of Genetic Algorithms: An Overview of Genetic Algorithms : Part 1, Fundamentals David Beasley Department of Computing Mathematics, University of Cardiff, Cardiff, CF24YN, UK.
5. Implementation of artificial intelligence in Snake game using genetic algorithm and neural networks: Piotr Białas Faculty of Applied Mathematics Silesian University of Technology Kaszubska 23, Gliwice, 44-100, Poland
6. Application of Genetic Programming to the Snake Game: https://www.gamedev.net/articles/programming/artificial-intelligence/application-of-genetic-programming-to-the-snake-r1175/
7. Solving the Classic Game Snake using Artificial Intelligence: https://towardsdatascience.com/slitherin-solving-the-classic-game-of-snake-with-ai-part-3-genetic-evolution-33186e6be110
8. Snake Game with Genetic Algorithm: https://theailearner.com/2018/11/09/snake-game-with-genetic-algorithm/
9. Designing AI: Solving Snake with Evolution: https://becominghuman.ai/designing-ai-solving-snake-with-evolution-f3dd6a9da867
10. Train a Neural Network to play snake game using Genetic Algorithm: https://pythonawesome.com/train-a-neural-network-to-play-snake-using-a-genetic algorithm/
11. AI Techniques for Game Programming by Mat Buckland.
12. Using Genetic Algorithms to Automate the Chrome Dinosaur Game: https://heartbeat.fritz.ai/using-genetic-algorithms-to-automate-the-chrome-dinosaur-game-part-2-1c0007334297
13. Automated Snake Game Solver via AI Search Algorithms by Shu Kong.
14. Y. Shichel, E. Ziserman, and M. Sipper, "GP-Robocode: using genetic programming to evolve Robocode players," Lecture Notes in Computer Science.
15. J. Muñoz, G. Gutierrez, and A. Sanchis, "Multi-objective evolution for car setup optimization," UK Workshop on Computational Intelligence (UKCI).
16. C.-Y. Cheng, Y.-H. Chen, and T.-C. Chiang, "Intelligent level generation for Super Mario using interactive evolutionnary computation," The 27th Annual Conference of the Japanese Society for Artificial Intelligence.
17. Snake Game AI: Movement Rating Functions and Evolutionary Algorithm- based Optimization by Jia-Fong Yeh, Pei-Hsiu Su, Shi-Heng Huang, and Tsung-Che Chiang Department of Computer Science and Information Engineering, National Taiwan Normal University, Taiwan (2016).
18. Snake- Game – AI- Solver by neel gajjar (2018): https://github.com/neelgajjar/Snake-game-AI-Solver.