Genetic Algorithm: Dino Game

AI/ML Project - Team Nazgul

Pushpendra Jakhar - 170050049
Drumil Trivedi - 170020016

INTRODUCTION

We often find ourselves playing the classic dino game on Google Chrome when we travel and there's no internet, and obviously the impending question, what's your high score ? So we took it upon ourselves to make a bot that aces the game. After some research we found that genetic algorithms work well for the problem!

GENETIC ALGORITHM AND PARTS

Every genetic algorithm involves 5 parts, and they are defined w.r.t to the game as follows

Initial Population

Defines the population for the first iteration

- Our population involves 100 Dinos per generation.
- Initially as there is no knowledge of the game, they are initialized randomly.

Fitness Function

The function helps us determine the healthy section of the population who will spawn the next generation.

- The fitness function in our case is the score of the bots.
- The probability that an individual will be selected for reproduction is based on its game score.

Selection

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

- We expect that the one with the highest scores has understood the semantics of the game as compared to its peers.
- Two pairs of individuals (**parents**) are selected from the set of top 10 high scorers. Individuals with high fitness have more chances to be selected for reproduction.

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.

- The set of genes here are the parameters of a neural network that help define actions.
- The crossing over is a random split and concatenation of parent parameters.

Mutation

In certain new offspring formed, some of their genes can be subjected to a **mutation** with a low random probability

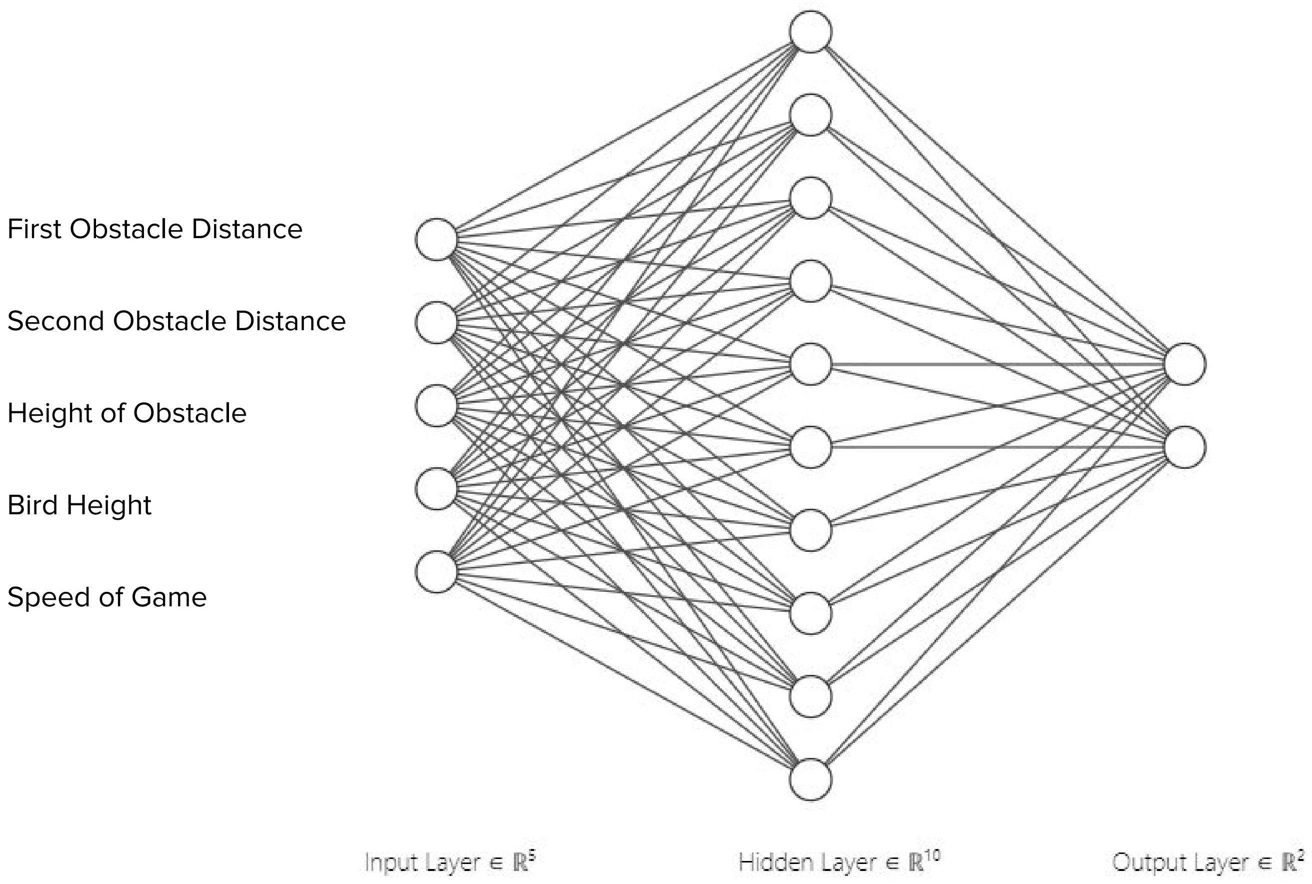
- Here by mutation we added some form of scaled noise to the parameters and it happens 20% of the time.

- We added a scaled noise by adding a N(0,0.8) scaled weights to the original weights

DESIGN & EXPERIMENTATIONS

Basic Neural Net

It involves 5 inputs as shown below . And two outputs i.e. whether to jump or duck.


First Obstacle Distance
Second Obstacle Distance
Height of Obstacle
Bird Height
Speed of Game
Input Layer ∈ $\mathbb{R}^5$
Hidden Layer ∈ $\mathbb{R}^{10}$
Output Layer ∈ $\mathbb{R}^2$

Initially we started with 3 input nodes - nearest object distance, object height and gamespeed and had one output node.

Over training for 50 generations we were getting high scores around 500. We observed that in most of the cases the Dino was crashing with the pteranodons (high flying objects which cannot be avoided unless a bot ducks). So we included bird height as well which would be 0 if the cacti was the object the height otherwise .

This increased the high scores to around 700. We observed that it was crashing due to jumping very close to the object and thus crashing into the second one when the 2 obstacles were close. This indicated a lack of information about the second in line object.

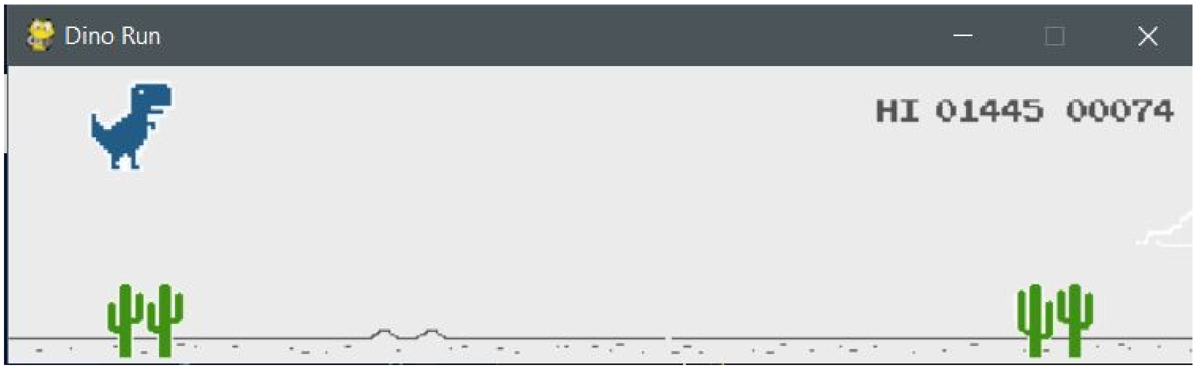
After including the distance of the second in line object it achieved high scores of about 1500.

After which the reason for gameover was due to the huge game speed.

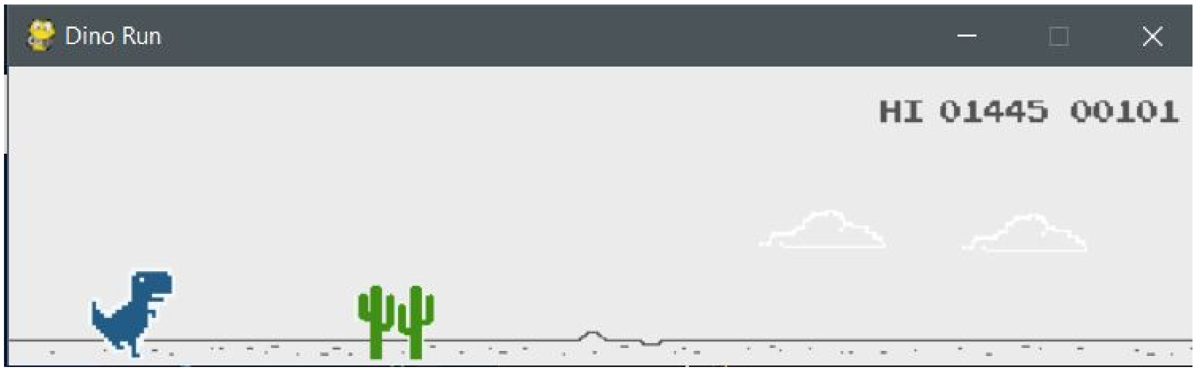
We also combated the forgetting through generations by keeping the highest scorers alive through generations and randomly considering them in the following progeny.

RESULTS

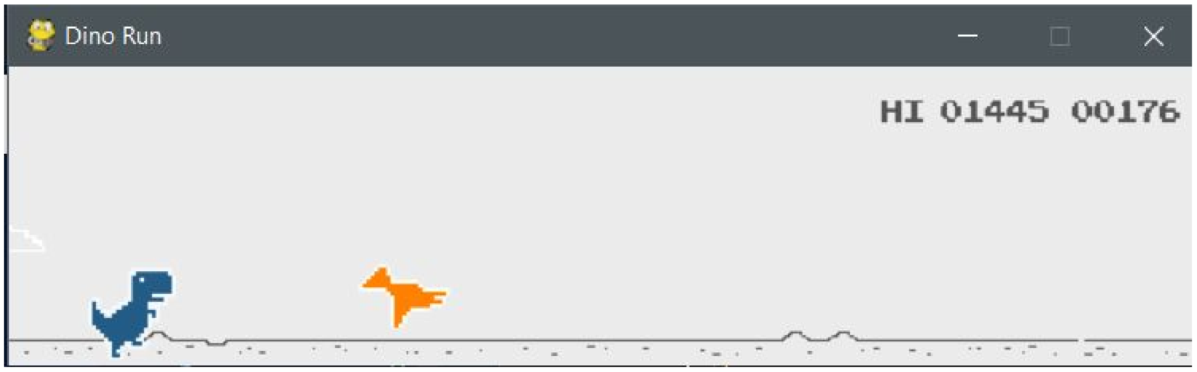
Shown in the snapshots we see the Dino in the final run at work.



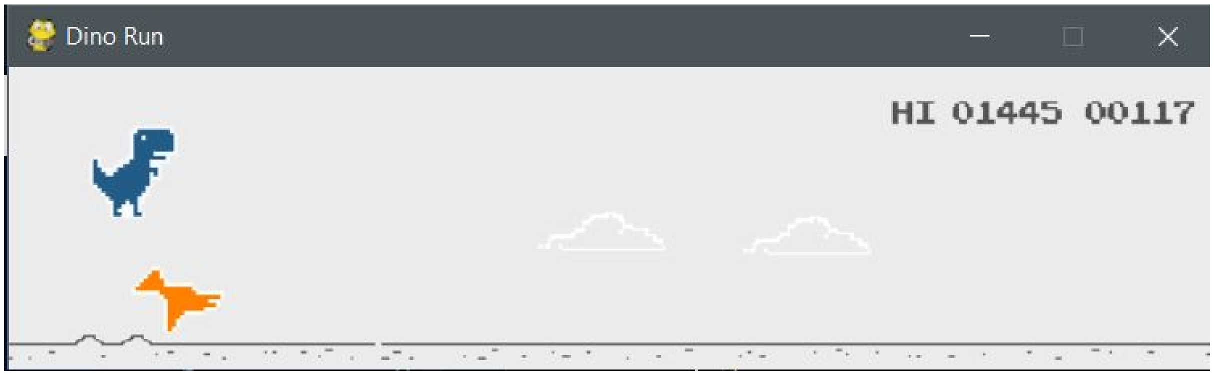
a) Neatly jumping over cacti



b) Not jumping until an obstacle is close enough



c) Landing after an obstacle



d) Jumping over the pteranodon

REFERENCES

- Genetic Algorithm - https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3
- Game Engine- https://code-projects.org/dino-game-in-python-with-source-code/#
- https://www.pygame.org/docs/tut/PygameIntro.html
- https://numpy.org/doc/