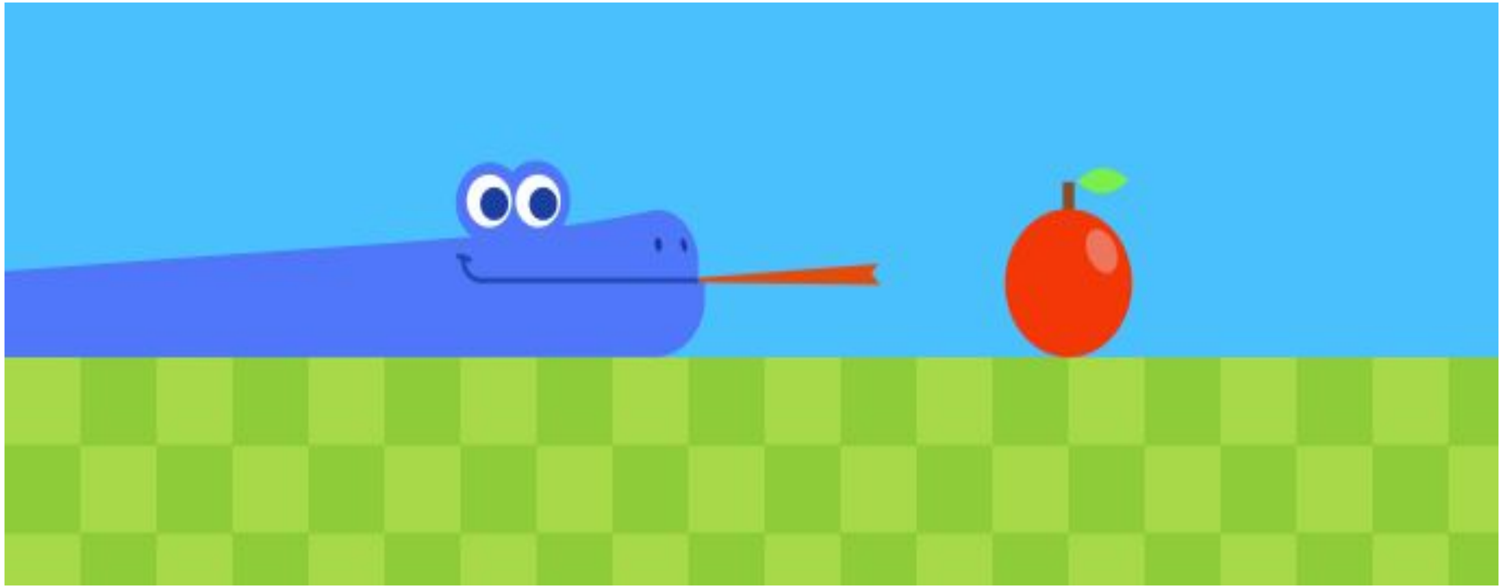
Deep Neural Network In Snake Game



Deep Learning Final Project

Students Team

Ahmed Magdy (07)
Mohamed Kamal El-Shazly (59)
Hesham Medhat (71)

Problem Statement

Getting our friendly snake to eat apples and grow!

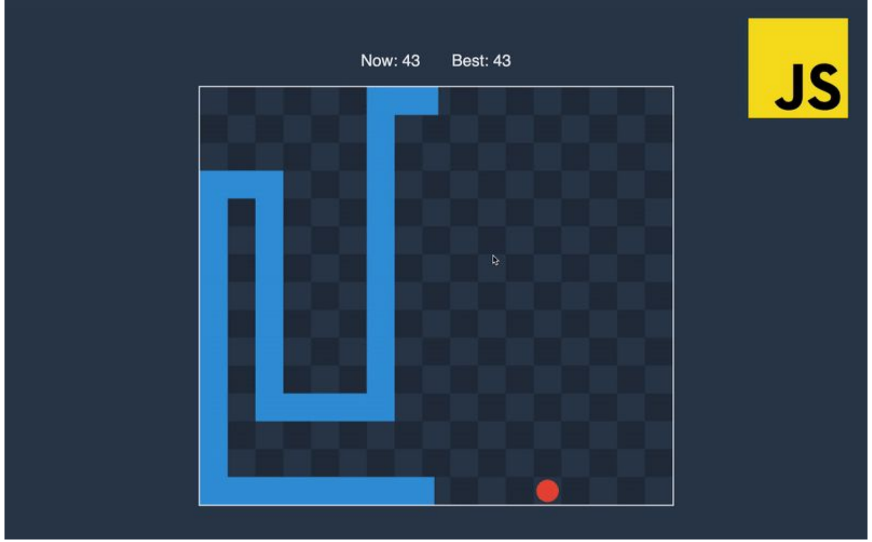
Detailed description coming next!

Game Overview

The classical game Snake was a world’s amusement since 1977. The game is about challenging a snake on the screen to be able to manage moving around a 2-dimensional grid to get rewards (apples/food).

Game Rules

Upon eating food, the snake's body grows.



The snake has the choices of either:

- Continuing to move forward in the given direction.
- Take a left turn.
- Take a right turn.

The snake dies upon contact with its own body or a wall.

Why Snake? A childhood challenge!

Applying what we have learnt in the Artificial Intelligence course last semester, in addition to what we are learning in this semester in Deep Learning; exploring the capabilities of deep neural networks in solving what used to be a childhood challenge for us.

Visualization for Investigation

We will simulate the game for the network, and make use of that facility of observing results and how the mind of the neural network works as it makes decisions attempting to win the game with top score. Rather than charts and graphs, observing the simulation of the game would be most educational and beneficial to us as we explore deep neural networks and how changing hyper-parameters reflects on its decision-making process.

Related Work

- *"Autonomous Agents in Snake Game via Deep Reinforcement Learning"* paper is taking screenshots for the play grid and making analysis to fetch the location of snake and apple using CNN and use reinforcement learning to control the snake. This approach achieve accuracy around 17 point in grid with dimension = 12X12

- This article approached this problem by using forward neural network to learn the snake which direction it should take to max its score.

Our Approach

For the sake of challenging our feature engineering skills and exploring the genetic learning capabilities of the deep neural networks, we will let the genetic algorithm (GA) and neural network(NN) play the snake game, and Instead of backpropagation, we will update weights using GA to find the best parameters.

Will this Work??

In December 2017, Uber AI Labs release five papers, related to the topic of neuroevolution, a practice where deep neural networks are optimised by evolutionary algorithms. One of them is *"Deep Neuroevolution: Genetic Algorithms are a Competitive Alternative for Training Deep Neural Networks for Reinforcement Learning"*

Uber AI Lab Paper

“Using a simple Genetic Algorithm (GA) it is possible to optimise DNNS and that GAs are a competitive alternative to gradient based methods when applied to Reinforcement Learning (RL) tasks such as learning to play Atari games, simulated humanoid locomotion or deceptive maze problems.”

Progress

- Make more investigation about the problem. ✓
- Study genetic algorithms and how they work with neural networks. ✓
- Find a suitable game simulator. ✓

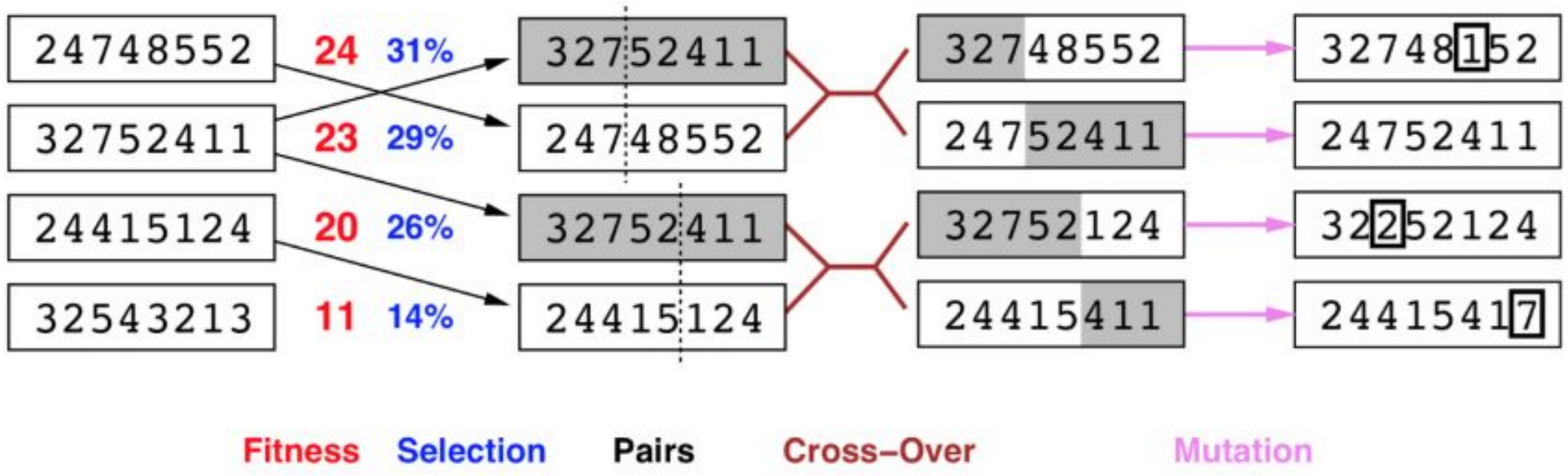
Neural Network Architecture

Using Transfer Learning, we will try to use the same architecture proposed in the related work and some other atari games like:

- One hidden layer with 128 neurons followed by linear output layer (treat it as linear regression problem).
- One hidden layer with 25 neurons followed by softmax layer.

The architecture is subject to change depending on the results in terms of number of layers, neurons, and activation.

Genetic Algorithm



Algorithm Steps

1. Creating an initial population (randomly generated weights).
2. Deciding the fitness function.
   a. +A for eating an apple
   b. +B for non-fatal move
   c. -C for colliding with boundaries
   d. -D for colliding with self
3. Play a game for each individual in the population

Algorithm Steps

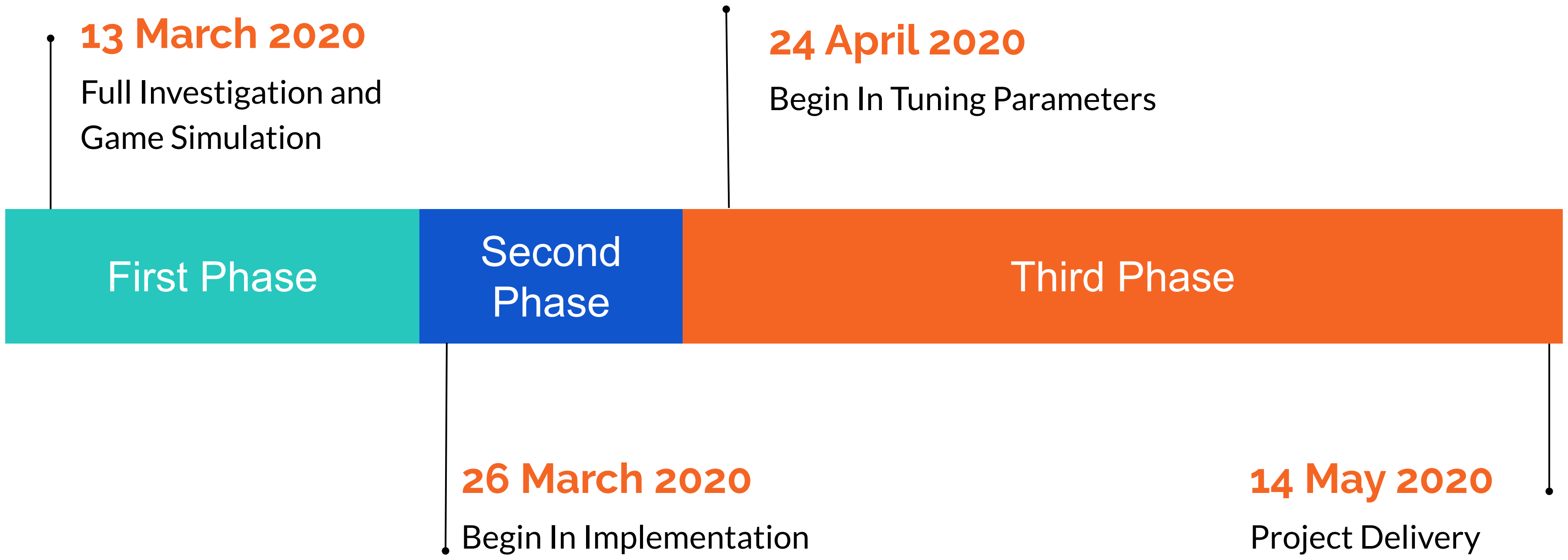
4. Sort individuals based on the fitness function score.
5. Select a the top “x” individuals.
6. Create the new population from the selected individuals using Crossover.
7. Apply mutation to the new population with a small probability.
    a. Re-assigning a random weight to some weight(s)
8. Go to step 3 and repeat until the stopping criteria is satisfied.
    a. Maximum fitness doesn’t change.
    b. Maximum iterations exceeded.

Input Factors and Initial Results

1. Is left blocked or is there any obstacle in left (1 or 0)
2. Is front blocked or is there any obstacle in front (1 or 0)
3. Is right blocked or is there any obstacle in right(1 or 0)
4. Angle between head of snake and apple
5. Snake current position (X, Y)
6. Apple current position (X, Y)

These inputs are subject to change.

Time Plan



13 March 2020

Full Investigation and Game Simulation

26 March 2020

Begin In Implementation

24 April 2020

Begin In Tuning Parameters

14 May 2020

Project Delivery

References

- https://www.researchgate.net/publication/327638529_Autonomous_Agents_in_Snake_Game_via_Deep_Reinforcement_Learning?fbclid=IwAR2tYSFsQtAeB9rH1evuCkyi7Gk4gsXoEEOZGbiCo4WpKsOU1zbrRFxjT0I
- https://becominghuman.ai/lets-build-an-atari-ai-part-1-dqn-df57e8ff3b26
- https://towardsdatascience.com/today-im-going-to-talk-about-a-small-practical-example-of-using-neural-networks-training-one-to-6b2cbd6efdb3
- https://towardsdatascience.com/deep-neuroevolution-genetic-algorithms-are-a-competitive-alternative-for-training-deep-neural-822bfe3291f5
- https://github.com/TheAILearner/Snake-Game-with-Pygame