Flappy Bird Decision AI

Kamil Bojanczyk *kamilien1@gmail.com*, Zhipeng Hu *zhipengh@stanford.edu*, and Chao Xu *cxu1@stanford.edu*

***Abstract*—The game Flappy Bird is a popular single-player game whose objective is to continue flying for as long as possible. The input is to go up, or flap by pressing the up arrow. The constraints are that the game ends if the bird touches the ground or pipes. We would like to use Reinforcement Learning and MDP to learn how to play Flappy Bird, and beats human player.The code for this project can be found at `https://github.com/cxu1/CS221-Project`**

I. Introduction

We like the flappy bird game setting because it is a good reduction of real world maneuver problem, for example navigation of manufacturing robots and autonomous drones. In this particular simple game setting, the vertical velocity and gravity are the only natural influence factors of the environment. These two factors represent a simple dynamics process compared to real world, where we would have to take into account many more variables.

II. Related Work

Perhaps the best-known success story of reinforcement learning is TD-gammon, a backgammon- playing program which learn entirely by reinforcement learning and self-play, and achieved a super-human level of play [1]. Also, DeepMind Technolgies developed first deep learning model for successfully learning control policies directly from high-dimensional sensory input for Atari 2600 games [2].

This is a similar project [3] on apply AI to Flappy Bird using Deep Q Learning algorithm described in Playing Atari with Deep Reinforcement Learning and shows that this learning algorithm can be further generalized to the notorious Flappy Bird. There are also other student projects that attempted to apply AI for gaming. One was done for the Artificial Intelligence class (CS221) for the classical snake game[4], where they used A* to find a path to the food whose consumption gives the maximum score, and used the difference between food value and Manhattan distance to the food as heuristic to maximize the snake length.

III. Project Scope

The first objective is to design an algorithm that successfully plays the Flappy Bird game and beats a beginner human player. Moreover, we will improve our algorithm to increasing score, reduce iteration time, and simplify the state space.

A next step is to augment the game to make it replicated to a larger extent a real world problem. The real world here is a the natural game environment a human experiences. Going further, the real world could represent robots and drones that face more complicated interactions with manufacturing and flight scenarios. We consider the following additions to make the world more realistic:

1) Drones deal with sudden gusts of wind: adding time-changing y-direction wind to the bird
2) Drones may be flying around other drones: dynamic pipe movement in the y direction
3) Cameras on robots face obstacles from glare, color, and other variables: adding random noise to measuring the distance to the pipe
4) Robots have to navigate through factory environments with people: Multiple gaps in the pipes to allow for two or more paths through the pipes; pipes that move slowly in the y-direction.

IV. Methodology

A. Example of Input and Output

The input of the algorithm would be a set of training actions, either flap (1) or don't flap (0). The algorithm then determines Q-values from reinforcement training paired with a weight vector and a feature vector on actions. The output would be the score of the game, a cumulative sum of the number of pipes the flappy bird has passed through.

B. Implementation

We constructed a mathematical model for the Flappy Bird game using Markov Decision Process (MDP) for modeling decision making where results are under game engine's control. And Reinforcement Learning would be used for teaching software agents what actions should take under different circumstances that can maximize cumulative reward or the game scores.

1) Markov Decision Process can be modeled as following:
   - States have two parameters: Horizontal position, Vertical position of the flappy bird on the screen.
   - Actions can be either flap (1) or don't flap (0)
   - Rewards can be constructed as: +1 if flying or -10000 if hits the pipe/game ends.
2) Deep Q-Learning
   The following algorithm [2] can be implemented for Reinforcement Learning:

Initialize replay memory D to size N
for episode = 1, M do
Initialize state s_1
for t = 1, T do
  With probability select random action a_t
  otherwise select $a_t = max_a Q(s_t, a; \theta_i)$
  Execute action a_t in emulator and observe $r_t and s_{t+1}$
  Store transition (s_t, a_t, r_t, s_{t+1}) in D

Sample a minibatch of transitions (s_j, a_j, r_j, s_{j+1}) from D
Set y_j:=
r_j for terminal s_{j+1}
$r_j + \gamma * max_(a')Q(s_{j+1}, a\prime; \theta_i)$ for non-terminal s_{j+1}
Perform a gradient step on $(y_j - Q(s_j, a_j; \theta_i))^2$ with respect to θ
end for
end for

C. Evaluation Metric

We would perform some test with human players, and collect their scores and then perform a standard deviation analysis of players' scores and compare the results with the scores coming from Deep Q-Learning. The scores from Deep Q-Learning should have much higher average value than human players, either beginner or experts.

V. Preliminary results

A. Baseline

For our baseline, we included the following features:

- Human player?

B. Oracle

For our oracle, we have the following features *after* a movie is released. These are:

- Relaxation that width of the pipe is zero?

VI. Challenges

The major challenge for our AI algorithm would be how to make it running effective and sufficiently quick since we need to ensure the game running smoothly and able to make decisions within milliseconds. Also, the state space for this problem is very large as there are huge numbers of pixels in vertical and horizontal direction. We need to find a way to reduce the state space or a new algorithm that can make the learning possible.
The second challenge for this project would be tackling with the complexity and uncertainties described in the project scope, including sudden gusts of wind and facing obstacles. There would be a huge challenge for identify correct feature vectors and weights vectors to make the learning possible.

References

[1] Gerald Tesauro. *Temporal difference learning and td-gammon* Communications of the ACM, 38(3):5868, 1995.
[2] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. *Playing Atari with Deep Reinforcement Learning*. NIPS, Deep Learning workshop
[3] Yenchen Lin, *Using Deep Q-Network to Learn How To Play Flappy Bird*, Github Source, https://github.com/yenchenlin/DeepLearningFlappyBird
[4] Abhinav Kumar Rastogi, Ayesha Mudassir Khwaja, Shabaz Basheer Patel. *Agent for the snake game*. Stanford CS221 Projects.