Deep Reinforcement Learning and Transfer Learning with FlappyBird

Cedrick Argueta
Department of Computer Science
Stanford University
Stanford, CA 94305
cedrick@cs.stanford.edu

Austin Chow
Department of Computer Science
Stanford University
Stanford, CA 94305
archow@cs.stanford.edu

Cristian Lomeli
Department of Computer Science
Stanford University
Stanford, CA 94305
clomeli@cs.stanford.edu

Abstract

Reinforcement learning's growth in popularity in recent years is partly due to its ability to play some video games with a level of mastery that no human can reach. Transfer learning is popular in the field of deep learning, and using pre-trained models on certain tasks speeds up training time and increases performance significantly. In this project we aim to apply transfer learning to the popular video game *FlappyBird* and analyze its performance compared to traditional reinforcement learning algorithms.

1 Introduction

Reinforcement learning is a technique for solving certain decision tasks, where an agent attempts to maximize a long-term reward through trial and error. This trial and error paradigm is better explained through the lens of exploration and exploitation: the agent must decide whether it explores and learns new information, or exploits the current policy to maximize reward. We will primarily explore the applicability of deep Q-learning, a deep reinforcement learning algorithm, to the FlappyBird game. In addition, we intend to explore the impact of transfer learning, a machine learning technique where a model developed for a specific task is reused as the starting point for a model in a second task. Being able to transfer learn will allow us to develop a more general model for different tasks, saving time and effort. In particular, we aim to see whether transfer learning will speed up training time to reach the same performance level as the baseline, or whether transfer learning will allow the agent to reach even higher performance levels than the baseline.

2 Related Works

The impetus behind our project is DeepMind's paper, "Playing Atari with Deep Reinforcement Learning". Mnih et al. were able to show how a Deep Q-Network could take raw image pixels from an Atari game and estimate the Q function. [**?**]

Their model also takes advantage of recent advances in convolutional neural networks for image processing, a target network for stabilizing policy updates, and experience replay for a more efficient use of previous experience. Transfer learning is the idea that generalization occurs both within tasks and across tasks. [**?**] Deep reinforcement learning may benefit from transfer learning, especially since convolutional neural networks are often used for playing games. Since convolutional neural networks often learn similar features in the first, second, etc. layers across a range of image classification tasks, it's possible that transfer learning in the context of reinforcement learning for video games can exploit this.

3 Game Mechanics

The game of FlappyBird can be described as follows: a bird flies at a constant horizontal velocity v_x and a variable veritcal velocity v_y. The bird can flap its wings, giving it a boost in altitude and velocity v_y. The aim of the game is to avoid randomly generated pipes that restrict the flying area, leaving only a small gap for the bird to fly through.

We model the problem as a Markov decision process with no knowledge of the transition probabilities or reward function at every state. The transition probabilities are unknown, since each state consists of the deterministic bird position and velocity, along with the non-deterministic pipe positions. The only reward signal received from the game in its standard implementation is when the bird flies past a pipe, giving us a reward of 1. This sparse reward makes it impossible for us to get an explicit reward for each (s, a, s') tuple. The start state s_{start} is the bird at some constant height, with no pipes on the screen. The actions for every state are the same: the bird can flap its wings or not. The only exception to this are the end states s_{end}, where the bird collides with a pipe or the ground. In these cases, there are no actions to take, and the episode ends.

4 Approach

The goal of our project is twofold: we aim to evaluate deep reinforcement learning algorithms on the FlappyBird game, and also experiment with transfer learning to analyze the impact it makes on the training process.

Vanilla learning methods like Q-learning are not well suited to this problem, since the state space of the game is very large. The position of the bird and pipes are continuous values, and as such we have an almost-zero probability of reaching a particular state that we've seen before. Thus, it will be necessary to use either function approximation to generalize to unseen states or some form of deep learning that can extract features automatically. We envision using Deep Q-networks as a nice foray into deep learning, given our experience with Q-learning.

Furthermore, we will demonstrate the ability for policies learned through deep reinforcement learning on FlappyBird to transfer to other games, such as those on OpenAI's gym platform. [**?**] This has been demonstrated before, particularly through the use of convolutional neural networks for playing games directly from pixels.

4.1 Input / Output

The input to our training models will be a prepossessed version of the PLE Games Version of Flappybird. The input to the model will specifically be an 84 x 84 x 4 array, in which the 84 x 84 section of the array is a scaled down black and white image corresponding to a single frame of the game, and the x 4 corresponds to an array of four time separated frames from the game. The output of our models is the optimum action (either jump or no jump), for any input array.

4.2 Infrastructure

The infrastructure for the game comes mostly from the PyGame Learning Environment and keras-rl packages. The PyGame Learning Environment provides a nicely wrapped implementation of FlappyBird, complete with sprites and the relevant game mechanics built in. Keras-rl provides a deep reinforcement learning framework that gives us a simple interface for training agents. We take advantage of these two here, writing simple wrappers for the PLE game instance so that it is compatible with keras-rl.

The keras-rl package additionally provides an implementation of several algorithms that are applicable to FlappyBird, particularly Deep Q-networks and the SARSA algorithm.

4.3 Baseline and Oracle

The baseline for this project is an agent trained with SARSA, with no transfer learning involved.

The oracle for this project is the high score of a human who can play the game extremely well. Since the game isn't particularly difficult for humans and gameplay doesn't change dramatically throughout the game, it's possible for humans to play nearly indefinitely. After looking at high scores online through leaderboards and videos, the max legitimate score that we found was 999.

The gap between the baseline and oracle shows us ...

5 Challenges

References

[1] Matthias Plappert. keras-rl, 2016.