MANCHESTER
1824
The University of Manchester

Department of Computer Science
Project Report 2020

Reinforcement learning for a learnable agent in classic arcade games

Author: Sean J Parker

Supervisor: Dr. Konstantin Korovin

Abstract

Reinforcement learning for a learnable agent
in classic arcade games

Author: Sean J Parker

The aim of the project is to investigate the performance of Gismos and to design and construct a super multi-functional Gismo.
The novel aspects of the new Gismo are described. The abstract should perhaps be about half a page long.
The results of testing, which show the abject failure of the Gismo, are presented.
In the conclusions proposals for rectifying the deficiences are outlined.

Supervisor: Dr. Konstantin Korovin

Acknowledgements

I would like to thank my parents, my school teachers, my friends, my wonderful supervisor, and all my wonderful fellow students for their unswerving support during my project. Without your help none of this would have been possible.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

My project aims to replicate some of the reinforcement learning algorithms that can be used to play classic Atari 2600 games. It also compares the results of different tests with these algorithms such as varying the hyperparameters of the network. By observing the effects on the trained agents[1] when the hyperparameters are changed we can deduce a set of optimal values such that the networks can play three different Atari 2600 games. Overall, the main features of the project are the following:

- Agents with raw pixel game data as input, outputting a set of values for the best action.
- Agents attempt to find an optimal model of the environment without any prior knowledge.
- Visualization of the agent "brain" to provide insight into what information the agents is learning.

1.1 Motivation

Over the past 10 years there has been significant improvement in the RL (reinforcement learning) algorithms. One reason is that the computing power has become cheaply available by using discrete graphics cards. For example, for my project I used a Nvidia GeForce GTX 1070 that provides 1920 CUDA cores that can be used to accelerate training of neural networks. Despite this, RL algorithms are massively computationally expensive and hence take a long time to train.

Over recent years one of the pioneers in this area is DeepMind, which was acquired by Google in 2014, and they developed the DQN (deep q-network) algorithm in 2013 which they demonstrated could learn directly from the raw pixel data of games in order to achieve either human-level or super-human level performance.

This research was expanded upon by DeepMind and OpenAI which is based on the original DQN by DeepMind. This research focused on trying to approximate a Q-function and thereby infer the optimal policy. On the otherhand, there has recently been a focus on other methods such as A3C and PPO which instead seek to directly optimise in the policy space of the environment.

[1] Agent. In this case, agent refers to a trained neural network that takes actions in a chosen environment.

1.2 Objectives

Further to what was described in section 1 there was a few main objectives of the project. Firstly, I chose three games on which I decieded to train the agents, Pong, Breakout, and Space Invaders which are shown below in Figure 1.1.

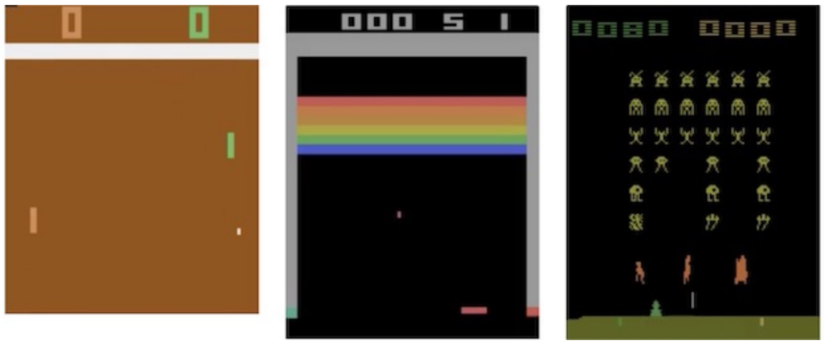

Figure 1.1: Screenshots of Pong, Breakout, Space Invaders (left to right).

Secondly, I wanted to find a way to explore the internals of a trained agent, in order to give further insight into what the agent is trying to learn. The reason for this is a researcher could use this information to determine, for example, where the agent has learnt to focus on the frame. Additionally, it provides a insight into how to optimally tune the hyperparameters which is described in section TODO: include section.

TODO: include gantt chart

1.3 Report structure

My report is divided into three main sections. Firstly, describing the background of the problem, then going onto giving details of my implementation and finally project evaluations/conclusions.

Chapter 2

Background

This chapter will cover some of the background material required for the following sections, it will cover the history of reinforcement learning (RL) and it's evolution to the current state-of-the-art. Additionally, it will cover the related work to this project and also cover some details of the past research papers for which this project has been based upon.

2.1 Reinforcement learning

Reinforcement learning is an area of machine learning has has been under active research since the late 1980s (TODO: ref watkins phd). One of the earliest applications of RL is for the game of backgammon. TD-Gammon is a computer program developed by Gerald Tesauro in 1992 to play the game of backgammon using *'temporal-difference learning'*.

2.1.1 Deep reinforcement learning

Following from section 2.1 on RL, this section talks about the combination of two areas, deep learning and reinforcement learning methods, called deep reinforcement learning (DRL). Deep learning (DL) is a common class of machine learning methods that has been of much research focus over the past decade and can deal with high-dimensional sensory input; for the case of Atari this is 84x84x4 greyscale images after pre-processing of the raw Atari frames. On the other hand, reinforcement learning allows us to create an agent which can learn an optimal policy to navigate some environment in order to optimise its reward.

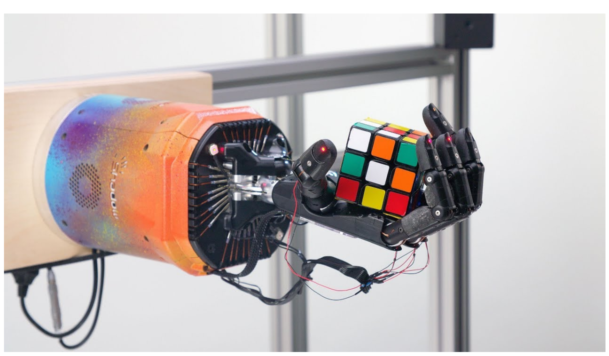

Figure 2.1: OpenAI Robot solving a Rubik's cube

Through the combination of these methods, it has proven to provide solutions to previously intractable problems [1] in areas such as robotics, computer vision and healthcare. For example, in 2019 OpenAI developed a robotic hand that could solve a Rubik's cube 2.1, trained using deep reinforcement learning. As an extension to DRL, end-to-end reinforcement learning is a method for single layered neural network, trained by reinforcement learning. Figure 2.2 shows, diagramatically, how DL and RL are used together in order to produce a single end-to-end model. In this simple architcture, there are two main components, the agent and the environment. Throughout all DRL methods, this is common to see, an agent observes some state and reward from the environment after taking an action.

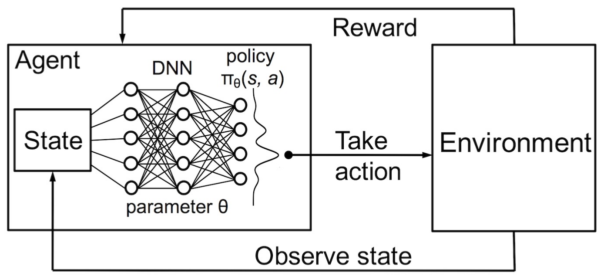

Figure 2.2: Representation of E2E RL models

2.2 DQN on Atari 2600

sdgsdsdfg

2.3 CNN Visualisation

sdfsdfsdfsdfsdf

Chapter 3

Design

3.1 Markov decision process

3.2 Reinforcement learning

3.2.1 Exploration vs Exploitation

3.3 Q-Learning

3.4 Q-Learning enhancements

3.4.1 Double Q-Learning

3.4.2 Duelling Q-Learning

References

[1] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath. A brief survey of deep reinforcement learning. *CoRR*, abs/1708.05866, 2017.