Representation and Search in RL: Literature Review

Chris Dipple
Stanford University, XCS229ii Milestone 2
chris@dipple.org

Abstract

Reinforcement Learning is critically dependent on two things. How it represents the problem space and how it searches that problem space. The representation can be tabular, or a function, typically learned with a neural net. Tabular representations of continuous state spaces can get very large and accuracy depends on the granularity. Neural networks require a lot of training and are inherently expensive in machine cycles. In both cases sample efficiency can be poor. Sample efficiency can be improved with better search strategies, and/or modelling. We propose exploring these two issues with some classic Reinforcement Learning problems.

Problem Statement

This research is motivated by the need for edge computing on low cost devices such as IoT, and autonomous vehicles for space exploration etc. In particular physics based environments. Environments where some degree of learning is required locally because the environment is not entirely static and/or centralised learning resources are not immediately available.

Tabular representations of Reinforcement Learning state spaces are simple when the state space is small, both in terms of discrete values modelled and the number of dimensions required. Unfortunately they get very large when dimensionality is high and/or each dimension has many distinct values (or value ranges in the case of discretised continuous state spaces). However often a state space is only 'sparely' explored, especially in higher dimensions, when a Reinforcement Learning algorithm is learning a policy. Potentially a great deal of space can be saved by not pre-allocating space for all possible states, especially in higher dimensional spaces. Also, granularity of the state space may only need to be low in certain places but of higher granularity elsewhere, can it be adaptive?

Another characteristic of Reinforcement Learning problems is the exploration/exploitation trade off is poorly managed and much time and energy is often wasted with simple search strategies such as epsilon greedy which can thrash about locally. The credit assignment problem cannot be qualified in many problems until some reward (not necessarily the best reward) has been identified, so efficient initial search is desirable. There are well known techniques for regret reduction, which can be helpful, but again only once some reward has been identified. Intrinsic rewards are very helpful, but complex. Are there better techniques, simple techniques, for exploring state space and finding rewards in a sample efficient manner that can help bootstrap the learning process and reduce thrashing?

Project Scoping

- The system will use OpenAI Gym as a simulator and source of data.
- The system will output a model for running CartPole and MountainCar, with discrete actions for now. Later to be generalised to more complex models, many with continuous action spaces.
- Success criteria will be efficient and stable learning with minimal resources (no GPU or TPU required or excessive memory). Above all it should be as sample efficient as possible given it will likely be model free for now.
- There are many challenges. RL is a huge area of study and we are addressing some of the fundamental issues. This project is a beginning not an end.
- There are aspects of RL problems that might be used to characterise them, over and above the common characterisations (Continuous or not, stochastic or not and others). For example CartPole receives rewards on every iteration, until it fails to do so. While MountainCar does not receive any rewards until it achieves it's objective, and to do so it cannot be greedy all the time. How to identify different search space characteristics and tackle different problems automatically with a common framework is the objective.
- Search strategies and making the most of a simple and common 'functional' representation (namely discretisation) is the limited objective of this project. It is possible that a crude solution, achieved quickly could be reused to train a more complex solution which could then be fine tuned.

General Problem/Task Definition

There are five references. A mix of discretisation schemes and state space search strategies. Some papers can be simply implemented, such as [Sutton, 2018] for tile coding is referenced in other papers as it acts as a building block. Other papers are aspirational, such as [Ecoffet, 2021] and [Badia, 2020].

The general problem area covered is functional representation of policies and search space principles to quickly uncover rewards, to bootstrap useful policy learning. Different papers take different approaches and there is no definitive, single approach.

The primary objective of this study is to explore relatively simple methods, that perform well across multiple criteria while efficiently traversing the problem space and mapping it's contours to quickly discover a reliable policy. Different problems carry different challenges, so any scheme must be adaptable and flexible. The Agent7 paper [Badia, 2020] suggests taking a range of policies and exploration strategies that adapt to the problem space. This would seem to be the way forward, hence the papers chosen. Often the problems can look very different, yet it would seem that the brain uses essentially that same subcomponent, the cortical column, thousands of times over, utilising different types of input while processing a a generic way. This would seem a sensible way forward with Reinforcement Learning too. An adaptable generic component, that can operate and cooperate with similar units to solve a bigger problem while tuning it's own representation and problem exploration strategies. See [Hawkins, 2021] as a background reference.

In the specific domain of primary interest, which is physics based problems, the use of Kalman filters would be sensible and this will be explored in future work. Even in other domains where there are common problems of orientation and identification of material or even immaterial objects such as abstract concepts, work with the brain suggests tools like the Kalman filter may be useful. However, we are trespassing into philosophy, later.

Concise Summary

The webpage by [Sutton, 2018] is a simple reference to the tiling algorithm, referenced in his book. This may perform better than simple discretisation and/or produce faster learning. Other papers listed below cover more complex schemes. Of particular interest is the trade off between system resources and sample complexity while supporting rapid learning. The overlapping tiling method described here is an essential milestone in discretisation and representation.

[Sinclair, 2019] is a description of an adaptive discretisation method for metric spaces. A uniform grid, while simple is inefficient, this method provides finer grained detail in those parts of the state and or action space that carry more information. It will do this dynamically as the system explores the State x Action space. The result is better results while using less memory and adapting to the problem.

The new version of the [Ecoffet, 2021] paper is published in Nature so two references are included [Ecoffet, 2020], the earlier one is not behind a paywall. The paper is a methodology to remember previously visited states that are interesting and return to them at a later point in time. The main problems inherent in prior solutions, they claim, is forgetting to get to older explored states and failing to be able to return to those states and continue exploration from those remembered states. They claim much better performance on Atari games with sparse, multistep and complex rewards, such as Montezuma's revenge.

The Agent57 [Badia, 2020] paper from DeepMind again tackles the problem of playing games in the Atari suit that were previously not played well. The paper addresses weakness with previous attempts by DeepMind to play Atari games where success was patchy. Some games achieved super human results but others, in particular those with sparse, multistep rewards were previously played poorly due to spare rewards and the tendency of Neural Networks to forget previously learning when it has not been seen again recently. The approach they take is to use a parametrised family of policies that span the range from mostly exploration to highly exploitative.

[Ghiassian, 2020] this paper is bit of a wild card, and is included here to give context and demonstrate the use of discretisation, even when Deep Neural Nets are used as the main functional approximation technique. The hypothesis is discretisation helps with stability and over generalisation. The resulting systems are more robust and less sensitive to hyper parameter settings in the neural network. It helps address the catastrophic interference problem. The paper is a detailed, exploration of the these issues and as such is good background.

[Weng, 2020] is a useful summary of intrinsic search methodologies that span a range of approaches from the simple to more complex. It is a useful survey and reference that lists salient points and references to the literature. It also gives an outline of the history and thinking in this space and what works and when and what does not.

Compare and Contrast

The papers referenced above have been chosen to be interrelated and mutually reinforcing. Many are on the leading edge of research into Reinforcement Learning and hence map out a way forward. The purpose in this paper is to take some of the simpler ideas, use those simplified ideas with classic reinforcement learning problems and create a deeper understand and highlight some common themes. It is also expected that some approachs will work for some problems and not for others, for example simple policy gradient methods like REINFORCE might be expected to behave poorly on the MountainCar problem. Indeed it takes a while, even with value based methods for naive search strategies like epsilon greedy to 'luck' upon a reward. This is crucial, finding some direction to gain rewards and exploring the problem space methodically. Indeed, the essence of the problem is, 'What does methodical exploration mean, relative to the space being explored now'? It is expected that there will be a beneficial correspondence between the exploration strategy used and the functional representation.

It is interesting to compare and contrast the Agent57 [Badia, 2020] paper and the Go-Explore [Ecoffet, 2021] papers. They both look at solving the harder Arari games such as Montezuma' Revenge. The approach taken is different though. They both take some inspiration from prior work, such as NGU (Never Give Up) by the same authors. The Agent57 paper suggests that extending NGU to be adaptive with it's exploration strategy over the lifetime of the agent and tuning some other parameters resulted in many benefits. An additional meta-controller to select policies to guide the exploration/exploitation trade-off when parametrised by exploration and discount factors. This theme of using meta/second order information to guide algorithms will likely see much more use in the future.

Go-Explore, in contrast to Agent57 does not build directly on top of NGU. It addresses the exploration problem by explicitly remembering past states visited which can later be returned to and used as the basis for further exploration. This does require 'restorable environments', which is not a feature of frameworks like OpenAi gym. Go-Explore uses two phases of exploration, the initial exploration phase and a later 'robustification' phase to make it tolerant to variants and stochastic noise. This latter robustification phase is resource intensive, so Go-Explore concentrates on the 11 Atari games other papers have struggled with, namely those with complex and sparse rewards. A later version of the paper (the one in Nature) learns a policy that can return to previously 'saved' locations without actually restoring the state directly but by navigating from the beginning to the same place. This strategy is reminiscent of Answer Set Programming systems such as clingo and older logic programming systems like Prolog from the Symbolic AI and planning communities. Which can backtrack to saved decision branch points. Causal inference also comes to mind too.

Go-Explore and Agent57 take different approches to solve similar problems. However, this is best viewed as a spectrum of possible approaches to similar problems. Mixing and matching is likely to the the eventual way forward. Both papers are written by private companies, and hence there is a level of competition here, this of course does not just happen between companies, but academically too. The best way forward is cooperation, as always.

The [Ghiassian, 2020] Breaking Generalisation paper is interesting primarily because it explores the issues created by using Deep Neural Networks for function approximation in Reinforcement Learning. DNNs suffer from Catastrophic Interference, previously leant data is slowly forgotten under the mass of new inputs available for reinforcement learning systems. The Go-Explore paper addressed this problem by allowing restarts. The Agent57 paper by the use of a meta-controller. Here the authors propose another, complimentary approach to reduce inappropriate generalisation. This is probably a defining characteristic of RL, generalisation, at least globally is not necessarily your friend as it is in supervised learning. The space is 'lumpy' and the lumps should not be forgotten or smoothed out or performance will suffer. The solution this paper proposes is to project the observations onto higher dimensions and use a discretisation method such as defined in [Sutton, 2018] with tile encoding. The idea of discretisation, then processing a neural network proposed here is simple and suggests a way to incrementally increase the complexity of the function approximator as time goes on to achieve higher levels of performance while keeping control of resource usage. In other words train with discretisation, then use the discretisation to train a DNN and then combine the two for fine tuning. Just an idea.

The [Sinclair, 2019] Adaptive Discretization paper takes a slightly different approach. It defines an adaptive discretisation framework that has higher fidelity for regions of the State x Action that naturally demonstrate more structure. This brings up the question of what is an optimal discretisation and how can it operate dynamically. This scheme is model free, in and of itself. Again, there is potential to combine this methodology with others, such as the Breaking Generalization paper.

The [Weng, 2020] reference is a useful summary of intrinsic reward mechanisms. As such, it has overlap with all

other papers. For our purposes we will concentrate on common and simple techniques that can be generally applied.

Bias and variance are present in all that is done in Reinforcement Learning. However, the techniques for taming them are different and they are inherent in what is done and how data is generated in a complex way that is not amenable to the techniques developed in ML Theory. They are however present and important. Monte Carlo returns are an obvious case in point, they are high variance which makes learning harder. For policy gradient algorithms like REINFORCE an approximation of the action-advantage function can be used $a_\pi(a,s)=q_\pi(s,a)-v_\pi(s)$ which can be approximated with $G_t-V(S_t)$ which is the return of the trajectory minus the state-value function at time t. This action advantage function tends to vary around zero and as good action will have positive advantage and poor actions negative action this acts as a damper and helps reduce variance when substracted from the policy gradient term. Similarly the use of replay buffers in Temporal Difference bootstrapping methods such as DQN and similar value based methods helps reduce inherent bias.

TD methods tend to underfit, as they cannot lookahead [and TD(lambda) is a way to address this problem on a sliding scale]. To summarise, a full Monte Carlo roll out has high variance and low bias while Boot straping has high bias as in TD(lambda=0), and low variance.

There is an interesting phenomena where a combination of bootstrapping, off-policy learning (such as Q learning) and function approximation can lead to unstable states. Boot strapping means we have bias, off policy means the main policy is only updated periodically and function approximation may see one state as very similar to the next. A combination of these factors, can lead to erratic behaviour and poor learning. This is a partial motivator for Policy Gradient methods (see above), but they have their own problems, so combined methods, namely Actor-Critic methods can be used.

Future Work

The papers referenced here provide a number of directions in which the Reinforcement Community is progressing. This is exciting and in many instances there is common ground to explore. This project will use some of the simpler mechanisms to determine better search strategies and the relationship between search strategies and simple function approximation, such as discretisation. Understanding generic learning principles is the primary objective, not just achieving the ‘best’ score ever for a particular problem. Ideally we can achieve as sort of ‘deep’ transfer learning, at least within a class of similar problems.

Later in this project we would like to attempt to design a mechanism to relate actions to changes in the state space. A map of the space, and the location of significant artefacts (rewards, maxima, minima, turning points etc) would be a possible way forward, then movement relative to these features. In a physics based environments an action will change some aspect of the the state space. This is how humans operate, we think, if I do that, then that happens. I want to achieve such and such, so do more of this. Typically a sequence of actions results in something interesting, not individual single actions. In CartPole for example a single action may not upset the balance catastrophically, providing there is a corrective sequence. Similarly, MountainCar needs to do a subsequence of similar actions, to build up momentum, before ‘reversing’ those actions to use that momentum to move in the opposite direction. In both environments, there are ‘outside’ force(s) operating too, ‘gravity’, that we need to work with to achieve an objective. A Kalman filter might help here, if there is a consistent error in a particular dimension with common characteristics then perhaps there is an unknown ‘force’ acting in the environment that we can estimate and model, hence improving the model in the Kalman filter and hence the RL agent. Indeed, in effect there are multiple actions, the explicit ones we control are trying to optimise and implicit ones that is always on (at least in some part of the space). This is something longer term, for example we need to build up momentum as actions are cumulative and we need to work with the ‘implicit’ actions. This can be used for exploration too. The problem is, how to represent this generically.

An interesting problem raised above is the potential and inappropriate generalisation of powerful function approximators such as Deep Neural Nets when applied to RL. It is especially interesting that a non-linear function such as a DNN can approximate a space defined by physics that is inherently linear, though multi-dimensional. This would imply we are missing features of the environment and the DNN, being non-linear, is ‘filling in’, if we are lucky, the missing pieces. There is much work to be done here to understand these spaces and how best to learn the contours of said spaces.

The idea of discretisation, then processing by a neural network proposed above [Sinclair, 2019] is worth pursuing. In many circumstances we suspect adaptive discretisation on it’s own will give good results. Reading the recent literature on DNN base RL one gets the impression that many of the innovations are simply there to reduce the inherent problems introduced by the use of Neural Networks in the

first place. Certainly the hardware manufactures have benefited. A deeper understanding of the use of DNNs in Reinforcement Learning is warranted.

An area of particular interest would be the use of Symbolic and other reasoning systems, such as causal reasoning, to model the policies created by RL systems. This is not well researched and the two communities have little in common at present. There would be many advantages, if it can be made to work, as symbolic reasoning systems are inherently explainable unlike DNNs. The use of meta-controllers as demonstrated by Agent57 is another promising area.

The appeal of Reinforcement Learning is the fact that it seems to address the problem of acting in the real world. The idea of learning from rewards goes back to the early days of AI. However the practical problems of getting this to work in the real world are immense. It is almost like we are in pre-Copernican times trying to apply geocentric models of the universe (which would appear to be obviously true from our individual personal experience). We make some progress, then invent more and more elaborate epicycles to explain and address contradictions we encounter. The answer, is step back and look again, cooperate with rivals trying to solve similar problems but have a different approach, in other branches of AI, in related disciplines and in Neuroscience. The components needed are probably known, however, they are not yet being put together in the right combination(s).

Bibliography

Badia, 2020: Adrià Puigdomènech Badia and Bilal Piot and Steven Kapturowski and Pablo Sprechmann and Alex Vitvitskyi and Daniel Guo and Charles Blundell, Agent57: Outperforming the Atari Human Benchmark, 2020

Ecoffet, 2020: Adrien Ecoffet and Joost Huizinga and Joel Lehman and Kenneth O. Stanley and Jeff Clune, First return, then explore, 2020

Ecoffet, 2021: Ecoffet, Adrien and Huizinga, Joost and Lehman, Joel and Stanley, Kenneth O. and Clune, Jeff, First return, then explore, 2021

Ghiassian, 2020: Sina Ghiassian and Banafsheh Rafiee and Yat Long Lo and Adam White, Improving Performance in Reinforcement Learning by Breaking Generalization in Neural Networks, 2020

Hawkins, 2021: Jeff Hawkins, A Thousand Brains: a New Theory of Intelligence, 2021

Sinclair, 2019: Sinclair, Sean R. and Banerjee, Siddhartha and Yu, Christina Lee, Adaptive Discretization for Episodic Reinforcement Learning in Metric Spaces, 2019

Sutton, 2018: Sutton, Rich, Tile Coding Software -- Reference Manual, Version 3.0, 2018

Weng, 2020: Weng, Lilian, Exploration Strategies in Deep Reinforcement Learning, 2020, https://lilianweng.github.io/lil-log/2020/06/07/exploration-strategies-in-deep-reinforcement-learning.html