Analyzing the Role of Temporal Differencing in Deep Reinforcement Learning

Alexandre Beaulne
alex@mgnr.io
ID 20087309

Amina Madzhun
amina.madzhun@umontreal.ca
ID 20052277

1 Introduction

Our project has for objective to reproduce a subset of results the results from TD or Not TD: Analyzing the Role of Temporal Differencing in Deep Reinforcement Learning [Amiranashvili et al., 2018]. In recent years, techniques combining Reinforcement Learning (RL) and Deep Learning (DL) have emerged [Mnih et al., 2015] to produce superhuman performance at some control tasks. However, the behavior of deep neural architectures when used as function approximators in RL is still poorly understood. The chosen paper aims to improve this understanding.

Background

RL is an area of machine learning concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward [Sutton and Barto, 2018]. In the traditional RL framework, an agent's evolution is *entirely* defined by a sequence of state, action and reward tuples. This sequence forms a partially observable markov decision process (POMDP). The objective of the agent is to pick the action in any state that will maximize the cumulative rewards encountered till the end of the sequence (the *episode*).

More rigourously, a standard RL problem develops over discrete time steps involving an episodic setup (starting an episode at $t = 0$ and finishing it at terminal time step $t = T$). The agent evolves in a stochastic environment with its current state s_t at time step t, and can can perform an action a_t from a set of actions. This action leads the environment to a new state s_{t+1}, and gives a reward r_t to the agent or finishes the episode. The goal of the agent is to learn a policy $\pi(a_t|s_t)$ to obtain the maximum expected return over time which is the total amount of reward that it can get from the current time step to the end of the episode, with future rewards being reduced by discount factor γ for each time step.

For a given policy π the value function and the action-value function are defined as:

$$V^{\pi}(s_t) \doteq \mathbb{E}_{\pi}[\hat{R}_t|s_t], \quad Q^{\pi}(s_t, a_t) \doteq \mathbb{E}_{\pi}[\hat{R}_t|s_t, a_t]$$

where $\hat{R}_t$ is cumulated discounted reward

$$R_t^{\gamma} = \sum_{i=t}^{T} \gamma^{i-t} r_i \tag{1}$$

or a truncated sum

$$R_t^{\tau} = \sum_{i=t}^{t+\tau} r_i \tag{2}$$

τ in this case is a fixed number of steps; the *horizon*.
The optimal state and state-action value functions are respectively given by

Submitted as course project for IFT6135 H2018 Representation Learning

$$V^*(s_t) \doteq \max_\pi V^\pi(s_t), \quad Q^*(s_t, a_t) \doteq \max_\pi Q^\pi(s_t, a_t) \tag{3}$$

Value based algorithms to find the optimal policy require computing or approximating one of the value function from equation (3).

2 Paper Summary

RL algorithms to find optimal policies involve exploring trajectories and backtracking observed rewards to previous visited state-action pairs. The depth and breath of the search vary between different algorithms (Figure (1)).

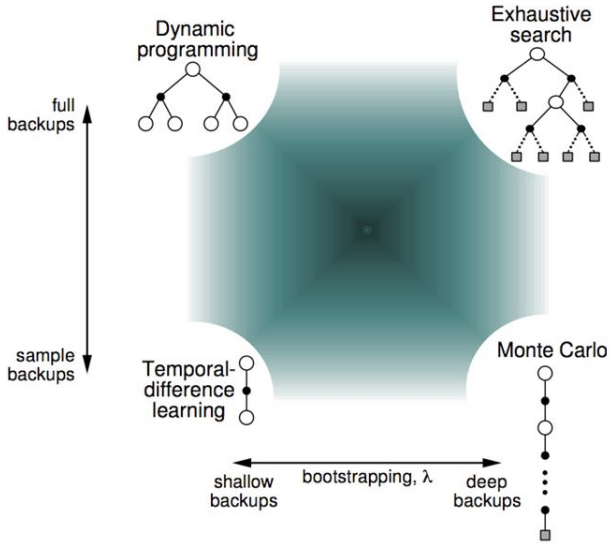

Figure 1: Different backtracking schemes Sutton and Barto [2018]

The large size of the state space of the problems considered preclude full breadth search, which leaves sampling as the only viable exploration scheme. Sampling search can be divided into two approaches, depending on the depth of the trajectory sampled. Monte Carlo (MC) search samples a trajectory till the end, while temporal differencing (TD) samples a single step, and approximate the value of the rest of the trajectory with the current value function at the next state (i.e. *bootstrap*). Naturally, the complete spectrum of hybrid exploration depths, i.e. rollout length, from 1 step (TD) to the end of the episode (MC) is possible. Before the advent of deep network as function approximators, it has been empirically demonstrated that TD is superior to MC [Sutton, 1996]. However, it is unclear whether those results still hold with the use of deep neural nets as function approximators. This question is explored in [Amiranashvili et al., 2018].

The authors two sets of experiments. In one set, they benchmark three different algorithms (A3C, n-step Q-learning and $Q_M C$) on games from the Atari and VizDoom environments, varying rollout lengths. The second set of experiments are *controlled*, fixed algorithms were tested against environments whose reward randomness, delay and sparsity were carefully adjusted.

The authors want to investigate the role of Temporal Differencing (TD) in modern deep Reinforcement Learning and observe the use of different techniques in controlled comparable experiments to evaluate the advantages of using one approach versus another. They reproduce classic results as well as use complex (3D) environments to test for tasks requiring perception, navigation and control, while separating influencing factors. They discover that Monte Carlo estimation (MC) can be feasible alternative and can be even competitive to TD, in some cases better in perception and control of visually complex 3D environments dealing with reward sparsity, delayed rewards etc. The work is discussed in more details further.

2.1 Motivation

In RL setup numerous results in the past have proven that for low-dimensional feature representations and linear function approximators Temporal Difference learning outperforms Monte Carlo estimation. However, in deep RL the spectrum of problems has been broadened by using deep network architectures for function approximation. This allowed to handle high dimensionality inputs, extremely large state space and more sophisticated environments with partial observability, still TD serves as the basis for most of developed algorithms. One of the problems in deep RL is that the choice of algorithm for given task is based on empirical results.
Dosovitskiy and Koltun [2016] successfully developed a Monte Carlo prediction based approach (Direct Future Prediction) which outscored TD, albeit the algorithm involved the use of specific concepts and components. The achieved results could induce the aspiration to study the existing solutions on standard problems within the framework of comparable experiments and motivate to explore the ways to combine advantages of existing approaches in deep RL.

2.2 Proposed approach

The more complicated model architectures become with developing new solutions, the more serious becomes the issue of defining causes for perplexing performance or the correctness of results. To surpass potential mistakes in conclusion for an analysis of algorithms, the authors try to separate the factors which could influence the performance:

1. complexity of the environment
   Experiments on 1D to 3D environments
2. algorithm configurations
   Consider three essential aspects:
   - TD or MC (infinite) varying rollout length in the update, -> n-step Q learning
   - infinite vs finite horizon -> Q_MC
   - pure based vs A3C not to be limited in conclusions -> A3C from Mnih
3. environment properties
   consider different factoring by criterias: sparse, delayed rewards, terminal state, reward type, perceptual complexity

+ additional help/observations in terms of health gathering<...> problems

3 Experiments

describe controlled experiments, VizDoom environments, perception

3.1 Motivation for the experiments

3.2 Reproducing the main results

4 Discussion

Mnih et al. [2015]
OpenAI's Gym [OpenAI]
[PyTorch]
Amiranashvili et al. [2018]
Mnih et al. [2016]

	#steps	Seaquest	Pong
20-step A3C (Mnih et al., 2016)	80M	2300	11.4
Q_{MC} (Amiranashvili et al., 2017)	60M	12708	-4.2
20-step Q (Amiranashvili et al., 2017)	60M	4276	8.9
20-step A3C (Amiranashvili et al., 2017)	60M	2021	20.6
20-step Q (our results)	20M	100	21
20-step A3C (our results)	20M	2702	21

Table 1: Our results at two Atari games against selected publications. Higher scores are better for both games.

References

Artemij Amiranashvili, Alexey Dosovitskiy, Vladlen Koltun, and Thomas Brox. Analyzing the role of temporal differencing in deep reinforcement learning. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=HyiAuyb0b`.

Alexey Dosovitskiy and Vladlen Koltun. Learning to act by predicting the future. *CoRR*, abs/1611.01779, 2016. URL `http://arxiv.org/abs/1611.01779`.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. ISSN 00280836. URL `http://dx.doi.org/10.1038/nature14236`.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1928–1937, New York, New York, USA, 20–22 Jun 2016. PMLR. URL `http://proceedings.mlr.press/v48/mniha16.html`.

OpenAI. Gym. URL `https://gym.openai.com/`.

PyTorch. Pytorch. URL `http://pytorch.org/`.

Richard S. Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. In *Advances in Neural Information Processing Systems 8*, pages 1038–1044. MIT Press, 1996.

Richard S. Sutton and Andrew G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 2nd edition, 2018.