Risk-Averse Reinforcement Learning with Long-term Memory

Joseph Murphy, Alex Rickman

April 2019

Abstract

We augment the Q-table update rules of a model-free episodic memory control agent in the Atari Gym environment in an effort to improve learning performance. The update rules we implement are intended to make the agent more robust to noisy environments, in which a one-time high reward for a state-action pair could cause the agent to overestimate future rewards of the same state-action. Building on the implementation of model-free episodic control Q-learning presented in [1], we train agents with various update rules across three Atari games, Qbert, Ms. Pacman, and SpaceInvaders. In some instances, we find that our new agents outperform the default update rule from [1]. We discuss the neuroscientific motivation for model-free episodic control, as well as possible explanations for why our new agents may outperform those of the default implementation.

1 Introduction

In recent years, [2] and others have shown that the successes of deep reinforcement learning (RL) models are limited by their data-inefficiency. With their slow, gradient-based optimization routines, deep RL models may require hundreds of hours of game play to complete training, while human agents can achieve the same, if not higher levels of performance in just a fraction of the time. For the agent, this essentially corresponds to a distant connection between actions, rewards, and the corresponding policy or value function update. Over the past few years, there has been effort to create more efficient agents using approaches that exploit architectures from neuroscience, such as episodic control [3]. To begin, we offer a brief introduction to the principles behind both deep RL and episodic control-based agents, while also describing their strengths and shortcomings.

1.1 Deep Reinforcement Learning

Deep reinforcement learning combines the power of reinforcement learning with deep neural networks. Deep neural networks can be employed to augment RL algorithms by e.g. processing visual data for autonomous vehicles [4], approximating value and quality functions, or learning a policy via policy gradient. As such, in the realm of function approximation deep neural networks have become a useful tool for Q-learning. The success and power of deep RL models is apparent in game play agents such as AlphaGo and now AlphaGo Zero [5]. As noted earlier, one of the limiting factors of deep reinforcement learning, however, is the massive amount of data and time required to train the neural networks. Deep neural networks for Q-learning (DQNs) may take hundreds of millions of frames to achieve human levels of performance in e.g. Atari games, corresponding to hundreds of hours of game play [6]. Additionally, DQN architectures are becoming increasingly complex and difficult to train [6].

1.2 Model-free Episodic Control

Model-free episodic control, outlined in [1], offers an alternative and effective approach to textitQ-learning that avoids the large-data demands of DQNs. In model-free episodic control, the agent learns rapidly by storing discounted rewards from state-action pairs in a table. With this implementation of Q-learning, there are no parameters for the agent to learn (i.e. model-free), and learning increases during episodes by both exploiting knowledge of past rewards from previous state-action pairs as well as the stochastic exploration to new states. In contrast to DQNs, the model-free approach in [1] builds a more direct connection between previous actions and rewards. The model-free episodic control agent is however vulnerable to noisy environments, where a one-time high reward from a certain state-action pair may cause the agent to overestimate the benefit of that decision in future episodes. Below, we offer a scientific motivation for the model-free episodic control agent, and augment the algorithm in [1] to be more robust to noise using several new Q-table update rules.

1.3 Scientific Basis for Episodic Memory Enhancement

Episodic memory—the recall of specific decisions and outcomes of past experiences—is a powerful tool for learning by both organisms and RL models, especially in high-dimensional state spaces where the value-function must be approximated [7]. By combining the evolved ability to recall past rewards and estimate their corresponding uncertainties, humans have been able to thrive in inherently stochastic and ever-changing environments. From a neuroscientific perspective, this amounts to our innate abilities to distinguish risky versus safe situations and take actions accordingly. Our brains have evolved over millions of years to precisely assess risk and perform a cost-benefit analysis of potential outcomes, and we attempt to train an agent to replicate this evolution in a much simpler context. One way to think about these processes is that our brains keep track of probability distributions of rewards over an action space and in sampling from these distributions at decision-time, determine how to proceed. Moreover, it is interesting to note that neuroscience researchers in the past couple decades have identified the key regions of the brain most involved with this risk-reward decision making, namely the bilateral ventromedial, amygdala, and somatosensory cortices [8] which could provide insight for building upon our oversimplified model in future studies.

The episodic memory structure currently in place is a memory efficient lookup table with matrix coordinate (s,a) mapping to pairs of states and actions. At each pair index, the model includes the discounted reward, R, and the number of episodes experienced since the last visit. When the table reaches capacity, the pairs with the longest time since the last visit are deleted, inspired by the human brain's method of forgetting memories that haven't been retrieved in a long time. When a state, action pair is visited by the algorithm, the lookup table at the corresponding index is referenced and the discounted reward at that index is evaluated based on the following update function:

$$Q(s_t, a_t) = max(R_{t,curr}, R_{t,old}). \tag{1}$$

2 Problem Statement and Environment

From the discussion above it is clear that the industry must currently choose between a high accuracy but high training cost and lower accuracy but minimal training cost. The open problem is to achieve the "best of both worlds" whereby we have a highly accurate agent able to be trained under minimal computational expense. For inherently risky environments where there often exists some variance in rewards given the agent's decisions, one typically cannot afford to take the cost-effective method, and must instead rely on powerful Q-Learning techniques. Below we present a series of modifications to the Model Free Episodic Control [1] discussed above framework whereby we design risk-reward

and long-term memory systems equipped to perform well with risky environments in mind under minimal training cost.

3 Technical Approach

This section presents the methods used for our implementation wherein we discuss the baseline model used to benchmark our project and the architecture and motivation used to construct the risk-averse reinforcement learning with long-term episodic memory. In order to withstand risky environments where simply memorizing and replaying a high reward situation might get us into trouble in future runs, we propose several modifications to the maximum replacement update rule currently implemented in Model Free Episodic Control [1]. Each update rule implementation and intuition is discussed in greater detail in the innovations subsection below. Though for this project we only used Atari games, we trained the agents on several games in the hopes that some might be more "risky" than others and thus yield better results when updating more conservatively as opposed to the greedy baseline approach.

3.1 Baseline Model: Model Free Episodic Control

In order to test our innovated update rules, we utilize the out-of-the-box code repository for Model Free Episodic Control[1] available publicly on Github[9] as a baseline. The baseline model updates the episodic memory table via

$$Q(s_t, a_t) = max(R_{t,curr}, R_{t,old}). \tag{2}$$

This greedy approach has proven to be very effective on stable environments, but we aim to present scenarios in which it pays to take a more statistical approach to updating the table.

3.2 Innovations

Our first idea was to instead of taking the maximum of $(R_{t,curr}, R_{t,old})$, taking the average. In this protocol, previous information stored in the table is not thrown out completely. Imagine a case in which our agent makes a risky move and 9 times out of 10 performs horribly but 1 out of 10 achieves the best reward it's ever seen. The current protocol is to greedily see the 1 success only and tell the agent to perform this way in the future. The update rule is thus given by:

$$Q(s_t, a_t) = (R_{t,curr} + R_{t,old})/2. \tag{3}$$

We took this a step further by implementing a weighted update rule, whereby the agent can be tuned to be more or less risk-averse based on the environment at hand. In this situation, we give a strength hyperparameter to the existing table value, and update the memory as follows:

$$Q(s_t, a_t) = \frac{1}{1+\alpha}(R_{t,curr} + \alpha R_{t,old}), \tag{4}$$

where α is the strength hyperparameter specifying how conservative to be, i.e. how much of a risk we are comfortable with taking.

We also decided to experiment with altering the structure of the lookup table altogether. Instead of a state, action pair mapping to a single reward, we modified the mapping to a list of rewards, in essence creating a long-term episodic memory enhancement. This would allow us to produce a running average of rewards, which is more intuitive than an average at every step. The agent operates by seeing the time-average across this list, and we use variance clipping whereby we drop entries with high variance across the entries of the list in order to prevent the agent from accepting risky strategies.

4 Results

In the above section we discussed three modifications made to the standard Model Free Episodic Control[1] implementation, and here we present the corresponding results. We train each agent on three environments from Atari through the Gym environment[10], Qbert, SpaceInvaders, and Ms. Pacman.

4.1 Simple Averaging Update Rules

We first present the default update, standard average, and weighted average playing on the Qbert environment. We use three metrics to measure performance, the average reward obtained per episode, the maximum reward obtained, and the summed reward over all training episodes.

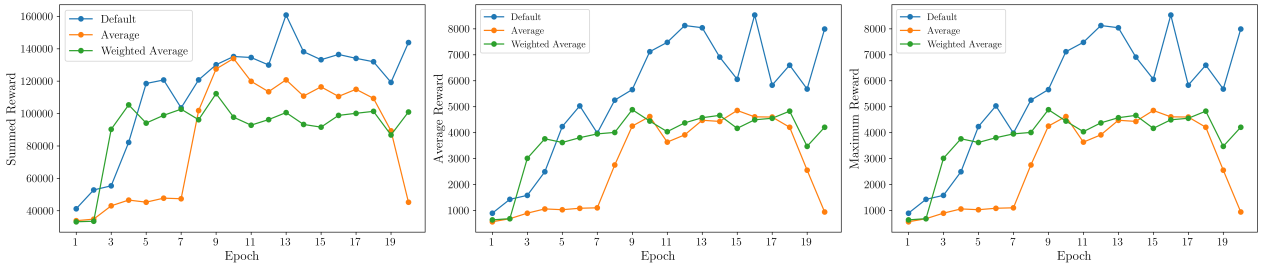

Figure 1: Performance on the Qbert environment over 20 epochs. Weighting strength is set to 5

It is clear from the above plots that the default update rule, in which a greedy maximum update is executed, significantly outperforms the other update rules.

Next, we show the same metrics but for the SpaceInvaders environment:

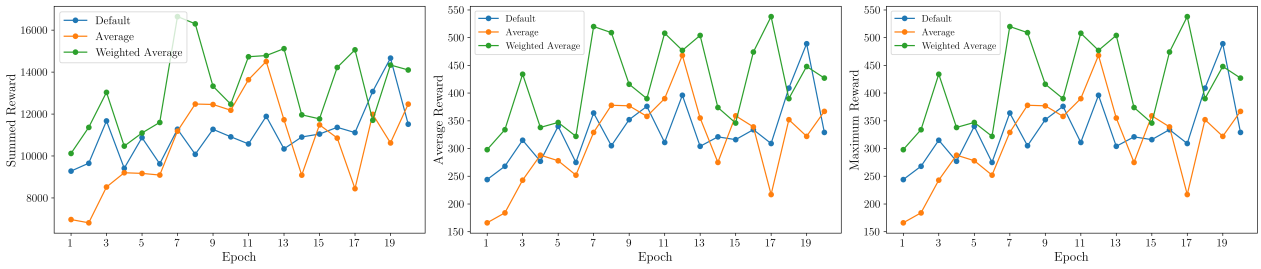

Figure 2: Performance on the SpaceInvaders environment over 20 epochs. Weighting strength is set to 5

It appears now that the tables have turned. We see sweeping improvements from the average updates, surpassing the default performance across all metrics.

Before we discuss the observed trends, we lastly present the results for the Ms. Pacman environment.

We again see better results for the modified update rules than the greedy default.

From the above plots, it is evident that there exists a place for risk-averse algorithms in these architectures. The challenge is in determining in which environments to be greedy and in which to play it safe. We hypothesize that the Qbert environment does not involve as many risk-reward trade-offs as needed for our agent to surpass a greedy approach. However, as of now we do not have metrics in place to quantify the riskiness of an environment. We call upon future research to

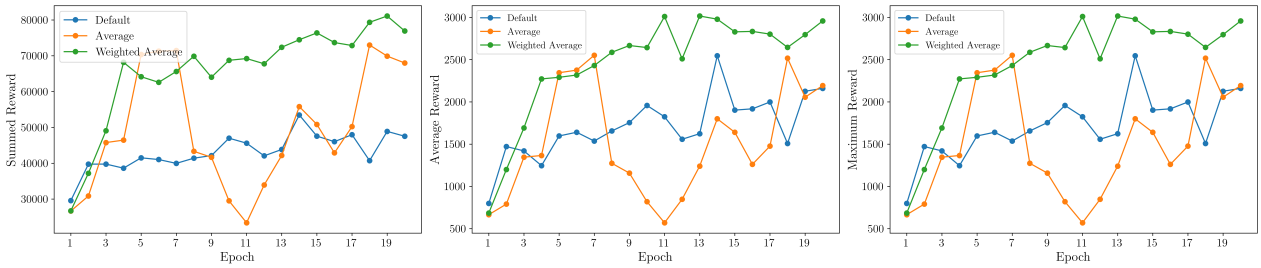

Figure 3: Performance on the Ms. Pacman environment over 20 epochs. Weighting strength is set to 5

develop ways to efficiently and systematically quantify the riskiness of an environment, which gives us a handle on which algorithm will have the advantage.

4.2 Long-Term Memory Averaging Update Rules

In this subsection we present the same metrics on the same environments as above, but for our long-term memory architecture.

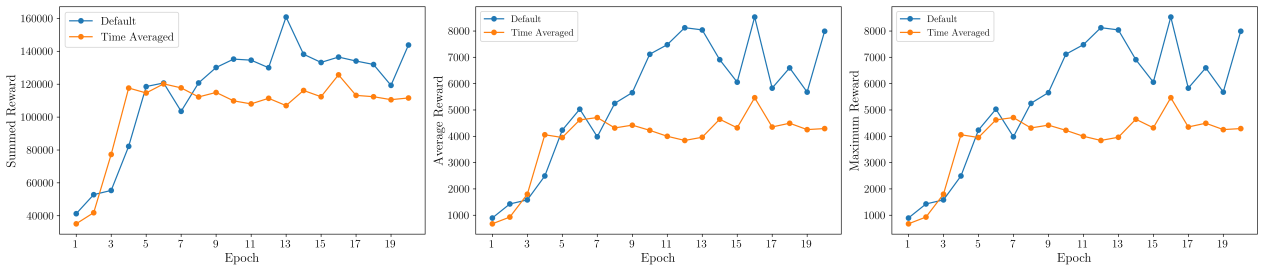

Figure 4: Performance on the Qbert environment over 20 epochs

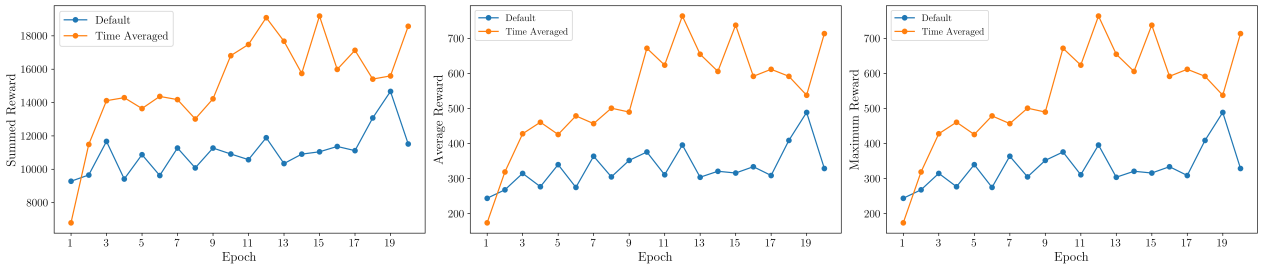

Figure 5: Performance on the SpaceInvaders environment over 20 epochs

We see the same trends as the simple averaging update rules in effect here for the long-term memory architecture. The default update rule outperforms the long-term memory averaging by a moderate margin on Qbert, but falls short of the modified architecture on SpaceInvaders and Ms. Pacman. Again we can only hypothesize that the latter two games inherently involve more riskiness. We also note an interesting feature in the Qbert plots where the default and long-term memory averaging seem to perform roughly the same for the first three epochs, but beyond that point the default continues to climb in reward whereas the averaging plateaus. It seems that the model loses its ability to benefit and adapt from new information once the memory fills up significantly with past

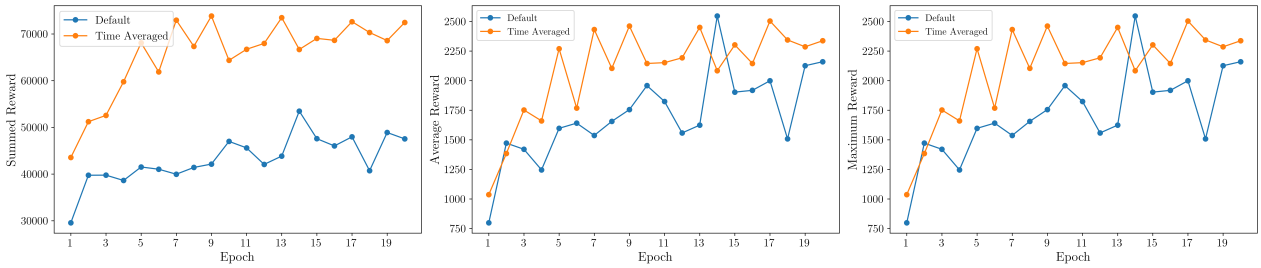

Figure 6: Performance on the Ms. Pacman environment over 20 epochs

rewards. In the other environments however, we don't see this leveling-off and instead see an even larger performance gap than we saw for simple short-term memory averaging. This shows promise of long-term memory averaging techniques, and also prompts further inquiry into determining why the agent sees tremendous benefits on some environments and plateau effects on others.

5 Conclusion

The results above allow us to conclude that statistically motivated update rules outperform greedy maximum reward updates on some but not all environments. We performed these experiments on three Atari environments from Gym[10]. On the Qbert environment, we found the default greedy update rule from Model Free Episodic Control[1] to dominate, however, on SpaceInvaders and Ms. Pacman, the statistically motivated averaging update rules (standard average, weighted average, and running average with variance clipping) outperform the greedy algorithm.

We hope to see future work in reinforcement learning aiming to quantify the riskiness of an environment from which can systematically employ a risk-averse or greedy algorithm tailored specifically to the environment at-hand. Furthermore, since these methods have shown significant promise, the next steps are to develop and test increasingly complex update rules. We are specifically interested in seeing an exponentially weighted average update rule for short-term memory [11], and a decaying weighted average update for long-term memory in which the average would weight past rewards less and less the further back you go, motivated by older memories being "fuzzier" and less certain in the brain.

Finally, we would like to thank Professors Thomas Dean and Rishabh Singh, Course Assistant Shreya Shankar, and all of the guest speakers throughout the quarter for providing us with the tools, resources, and motivations needed to accomplish this project.

6 Code

Our code repository, along with installation and execution instructions, can be found at https://github.com/arickman/CS379C_final_project.

7 Member Contributions

As a group of just two, Joseph and Alex worked closely together on all aspects of this project, from initial research, to planning, to implementation, to figure generation and writing. The partners met frequently to iterate on different project ideas before the proposal, and continued to work together to produce update rules and trained agents during implementation.

References

[1] C. Blundell, B. Uria, A. Pritzel, Y. Li, A. Ruderman, J. Z. Leibo, J. Rae, D. Wierstra, and D. Hassabis, "Model-Free Episodic Control," *arXiv e-prints*, p. arXiv:1606.04460, Jun 2016.

[2] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman, "Building Machines That Learn and Think Like People," *arXiv e-prints*, p. arXiv:1604.00289, Apr 2016.

[3] M. Lengyel and P. Dayan, "Hippocampal contributions to control: the third way," in *Advances in neural information processing systems*, pp. 889–896, 2008.

[4] S. Wang, D. Jia, and X. Weng, "Deep Reinforcement Learning for Autonomous Driving," *arXiv e-prints*, p. arXiv:1811.11329, Nov 2018.

[5] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, p. 354, 2017.

[6] A. Irpan, "Deep reinforcement learning doesn't work yet." https://www.alexirpan.com/2018/02/14/rl-hard.html, 2018.

[7] S. J. Gershman and N. D. Daw, "Reinforcement learning and episodic memory in humans and animals: An integrative framework," *Annual Review of Psychology*, vol. 68, no. 1, pp. 101–128, 2017. PMID: 27618944.

[8] A. Bechara, "Neurobiology of decision-making: Risk and reward," *Seminars in clinical neuropsychiatry*, vol. 6, pp. 205–16, 08 2001.

[9] A. Stier, "Model-Free-Episodic-Control implementation based on DeepMinds paper: http://arxiv.org/abs/1606.04460," 2016.

[10] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," 2016.

[11] A. Wong, "Exponentially weighted average for deep neural networks," *Data Driven Investor*, 03 2019.