Unifying On-Policy and Off-Policy Learning in TD Learning and Actor-Critic Methods

Riashat Islam [1] Raihan Seraj [2]

Abstract

Unifying algorithms in model-free reinforcement learning may bring success in a wide variety of simulated domains. There are a multitude of TD control algorithms in model-free RL, including SARSA and Q-learning, which are either on-policy or off-policy algorithms. On-policy algorithms may offer stable learning, but at the cost of high sample complexity. Off-policy algorithms are often sample efficient but biased. In this work, we present a novel approach for the first time to unify on-policy learning with off-policy learning in TD control. We aim to combine the stability of on-policy TD learning with the efficiency of off-policy learning. In this paper, we present a unified approach where TD our control algorithm either uses on-policy sampled action or off-policy samples depending on the amount of exploration, extending ideas from the Q(σ) algorithm (Asis et al., 2017) which unifies existing TD control algorithms for multi-step returns. Our approach can also be extended for actor-critic policy gradient methods, actor-critic(σ) where the critic is updated either on or off policy based on exploration. The unified approach is to trade-off the benefits and limitations between on-policy and off-policy policy gradient methods in RL. We also present the Q(σ, λ) algorithm that uses eligibility traces and trace decay λ parameter for the Q(σ) algorithm. Our results are demonstrated with linear function approximation to show the effectiveness of our novel approach to unify and trade-off on-policy and off-policy TD learning.

1. Introduction

Model-free reinforcement learning (RL) is a promising approach for solving goal directed behaviour, where an agent interacts with an unknown environment to solve a particular task (Sutton & Barto, 1998). Temporal difference learning methods (Sutton, 1988) are a fundamental component of RL algorithms that allows learning to occur directly from raw experience. TD learning often consists of on-policy and off-policy methods. SARSA is an on-policy TD learning algorithm that learns the agent policy directly from raw experience. In contrast, Q-learning (Watkins, 1989)is a popular off-policy algorithm where a target policy of the agent is learnt, but a different behaviour policy is used to generate the behaviour of the agent. An exploratory behaviour policy is often used to explore the full state and action space.

Both on-policy and off-policy methods in RL have their own advantages and limitations. On-policy methods such as Monte-Carlo policy gradient methods (also known as REINFORCE) (Peters & Schaal, 2006; Schulman et al., 2015) often suffer from high variance estimates and requires large number of on-policy samples. Off-policy methods such as Q-learning and actor-critic methods (Silver et al., 2014), in contrast are sample efficient as they can use all samples including off-policy samples to explore the state space. However, off-policy methods often suffer from convergence issues, and an importance sampling ratio term is included to correct for the difference in the target and behaviour policy. Using importance sampling to learn an off-policy often creates high variance estimates.

2. Proposed Approach

In this work, we unify the framework of on-policy and off-policy learning based on approaches from the Q(σ) algorithm (Asis et al., 2017). The Q(σ) algorithm unifies TD control methods such as SARSA and Tree Backup (Precup et al., 2000), where based on the σ parameter, we can either follow a SARSA update or an Expected SARSA (van Seijen et al., 2009) update (same as Tree Backup for n-step

[1] McGill University, Montreal, Canada [2] McGill University, Montreal. Correspondence to: Riashat Islam <riashat.islam@mail.mcgill.ca>, Raihan Seraj <raihan.seraj@mail.mcgill.ca>.

returns). Tree backup can learn off-policy without the need for importance sampling ratio that considers an expectation over all actions in the backup; in contrast SARSA update is based on a sampled action. We take a step further to unify on-policy and off-policy methods based on the σ parameter.

The intuitions and contributions behind our work are as follows. Based on the σ parameter, we either follow an on-policy update (SARSA) or a Tree Backup (Expected SARSA) off-policy update, where σ parameter is dependent on the amount of exploration of the state space. By varying the σ parameter, we want to balance between on-policy and off-policy learning. We want to develop methods that are both sample efficient and stable, by combining the advantages of on-policy learning with efficiency of off-policy learning. Our approach considers using a "mixture" policy (consisting of on-policy and off-policy samples) to learn a target policy. As an agent starts to act in an unknown environment, we want to consider an off-policy behaviour policy to explore the environment. The TD target would then consider an expectation over all possible exploratory actions (as in Tree Backup), since in an unseen state, the agent should consider all possible actions but then takes an action based on the target policy. However, once an agent has sufficiently learnt about unseen states in environment, we want to consider an on-policy update (as in SARSA).

The contributions from this work can be summarised as follows. We first demonstrate the use of both the on-policy and off-policy Q(σ) algorithm with linear function approximation, with different σ schedules (static, varying and exploration dependent σ parameter). We then present our unified approach that uses a mixture of on-policy and off-policy learning. This balances between either using on-policy samples or off-policy samples based on amount of exploration, trading off the advantages between on-policy and off-policy learning. Our approach is also demonstrated in the actor-critic policy gradient methods. We present an actor-critic (σ) approach that combines on-policy actor-critic (SARSA critic) and off-policy actor-critic (Tree Backup critic) depending on σ parameter. The critic can now be updated using either an on-policy critic (SARSA critic) or an off-policy ciritc (Tree Backup) depending on the σ parameter (and using similar σ schedules described above. Our unifying framework is then used as a mixture critic update where the critic uses a mixture consisting of on-policy and off-policy samples dependent on amount of exploration. Additionally, we present the Q(σ, λ) algorithm where the update uses λ-returns with the use of eligibiity traces in case of function approximation. This uses the computationally efficient bakward view TD learning with eligiblity traces, but extended for the Q(σ) algorithm.

The code for our implementations can be found at : https://github.com/Riashat/Q_Sigma_Tree_Backup_Eligibility_Traces.git

3. Q(σ) : Unifying On-Policy and Off-Policy Learning

In this section, we derive the key idea to unifying on-policy learning with off-policy learning. Our algorithm can be derived by observing the Q(σ) algorithm update equation below. The generalised TD error for one-step Q(σ) can be written, in case of function approximation as:

$$G_t^{(1)} = R_{t+1} + \gamma \big[\sigma_{t+1} Q_t^w(s_{t+1}, a_{t+1}) + (1 - \sigma_{t+1}) V_{t+1}^w \big] \tag{1}$$

$$\delta_t = G_t^{(1)} - Q_{t-1}^w(s_t, a_t) \tag{2}$$

In the on-policy case, the actions are sampled from the target policy $a \sim \pi(.|s_t)$. Similarly, for the off-policy Q(σ) one step update, the same update equation 2 is followed, except the actions are sampled from $a \sim \mu(.|s_t)$ where μ is the exploratory behaviour policy. For off-policy Q(σ), the value function V^w is computed based on expectation over all possible exploratory actions.

We first present results for the windy grid world environment for both the on-policy and off-policy Q(σ) algorithm with linear function approximation. In the windy grid world, the agent can take four possible actions of right, left, up and down. In the middle of the gridworld, the agent can get affected by n upward wind which shifts the resultant next state upwards. At each step, the agent can receive a constant negative reward of -1 until the goal state is reached.

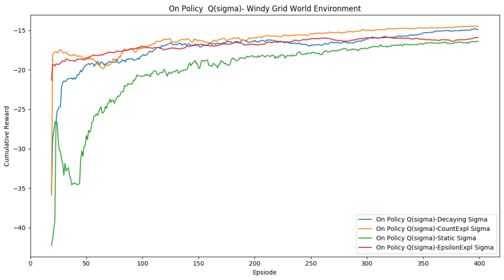

Figure 1. On-Policy $Q(\sigma)$

The obtained result for on-policy $Q(\sigma)$(Figure 1) shows that that a higher average cumulative reward was obtained when σ was varied by keeping a count of each state visited. The Experiment was performed by varying sigma by four different means. For decaying sigma we start with a sigma value of 1 and decay this value at the end of each episode by a factor of 0.95. For count based exploration sigma we keep a count of each of the states visited in an episode and vary the value of σ to either 1 or 0. We used the notion that when $\sigma = 1$ the algorithm performs a SARSA update and when $\sigma = 0$ the algorithm performs a Tree Backup update which is an expectation over all possible actions. Since we wanted to perform SARSA after a state has been visited beyond a certain threshold. We used a threshold of 5 in this case and consider that once a state has been visited more than 5 times it is sufficient to perform a SARSA update rather than taking an expectation over all possible actions. We next performed the experiment by varying the value of σ to 1 or 0 randomly which was generated from a binomial distribution with a probability of 0.5. Finally we varied the value of σ based on exploration strategy where during the exploration phase we kept $\sigma = 0$ which performs a Tree Backup update. The results were obtained by taking the average cumulative reward over 20 runs.

We also take a similar measure of varying the value of σ for off-policy $Q(\sigma)$ and obtained the following results.

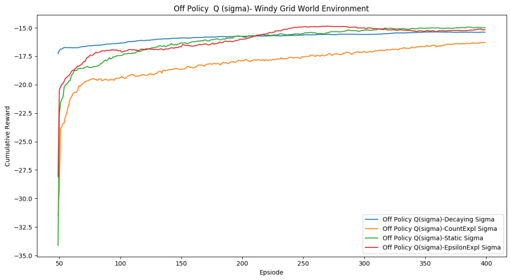

Figure 2. Off-Policy $Q(\sigma)$

The results for off-policy $Q(\sigma)$ algorithm in Figure 2 shows that a lower average cumulative reward was obtained when sigma was varied based on keeping a count of the number of times a state has been visited in an episode. This is justified by the fact that when the frequency of a state being visited exceeds a threshold of 5, we are using a value of $\sigma = 1$ which is infact a SARSA update. However, this SARSA update is being performed by sampling the off policy actions instead of on-policy actions as a result this leads to the algorithm yielding a lower average cumulative reward compared to other forms of variation of σ.

Our proposed approach for unifying in order to obtain a mixture of on-policy and off-policy learning can then be demonstrated by considering the TD target below

$$V_{t+1}^{w} = \sum_{a} \mu(a|s_{t+1}) Q^{w}(s_{t+1}, a^{\mu}) \tag{3}$$

$$G_t^{(1)} = R_{t+1} + \gamma\big[\sigma_{t+1} Q_t^{w}(s_{t+1}, a_{t+1}^{\pi}) + (1-\sigma_{t+1}) V_{t+1}^{w}\big] \tag{4}$$

Equation 4 shows that depending on the σ parameter being either $\sigma = 0$ (SARSA update) or $\sigma = 1$ (Tree Backup update), we either consider an on-policy update with actions from $a \sim \pi(.|s_t)$ or an off-policy update with actions from $a \sim \mu(.|s_t)$. Additionally, if σ is varied dynamically, we are considering a mixture where the target consists of action value functions using both on-policy and off-policy action samples. For our proposed approach, we demonstrate that the σ parameter can be varied with either $\sigma = 0$ or $\sigma = 1$ depending on the size of exploration.

Considering count-based exploration in grid world environments, if the agent has visited a state N number of times, then we should consider an on-policy SARSA update (with $\sigma = 1$), since the agent has explored the environment sufficiently. Similarly, if agent has visited the state less than N number of times, then we consider that the state has not been visited frequently, and so when arriving at that state, the agent should consider a tree backup update, with expectation over all possible actions (for $\sigma = 0$). This expectation is over all possible exploratory actions.

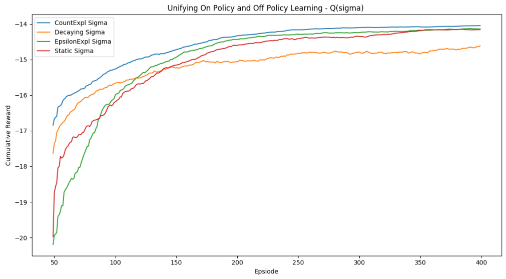

Figure 3. Unifying On-Policy and Off-Policy Learning

The results obtained for unifying on-policy and off-policy $Q(\sigma)$ learning is given in Figure 3. The experiment was then performed by choosing the four possible ways to vary σ. The results obtained thus show that varying sigma based on keeping a count of the states visited and on exploration, yields an overall higher average cumulative reward compared to randomly varying the value of σ by 0 and 1 and decaying σ in each episode by a common factor.

We then compare the unified $Q(\sigma)$ learning with on-policy and off-policy $Q(\sigma)$ the resutls of which is given in the following.

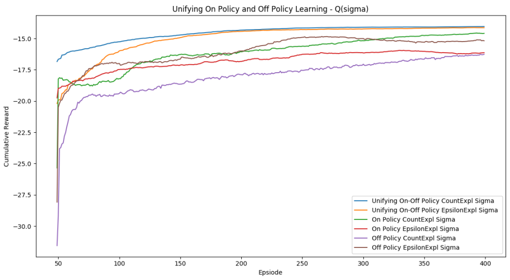

Figure 4. Comparing Unified $Q(\sigma)$ with On-Off Policy $Q(\sigma)$ Algorithm

Our results show that unifying on-policy and off-policy Q-sigma yields a better result for for count based sigma exploration rather than the on-policy and off-policy $Q(\sigma)$(Figure 4).Thus we deduce that following a strategy of changing σ based on exploration in a unified algorithm gives a superior performance compared to on-policy and off-policy $Q(\sigma)$ alone.

4. Unified Actor-Critic(σ)

The actor-critic is a widely used architecture based on the policy gradient theorem (Sutton et al., 1999; Silver et al., 2014; Peters & Schaal, 2008; Bhatnagar et al., 2007), where the actor adjusts the policy parameters θ based on the gradient of the cumulative reward $\nabla_\theta J(\theta)$ by gradient ascent. The actor updates uses the approximate action-value function $Q^w(s,a)$ with parameter vector w. The critic then estimates the action-value function such that $Q^w(s,a) \approx Q^\pi(s,a)$, using TD learning algorithm. The critic and actor updates can be written as follows, for on-policy actor updates

$$\delta_t = R_{t+1} + \gamma Q^w(s_{t+1}, a_{t+1}) - Q^w(s_t, a_t) \tag{5}$$

$$w_{t+1} = w_t + \alpha \delta_t \nabla_w Q^w(s_t, a_t) \tag{6}$$

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta \log \pi_\theta(a|s) Q^w(s,a) \tag{7}$$

Similarly, for the off-policy actor critic (Degris et al., 2012), the critic updates are based on off-policy TD learning such as Q-learning, such that the TD error is now

$$\delta_t = R_{t+1} + \gamma Q^w(s_{t+1}, \pi_\theta(a|s_{t+1})) - Q^w(s_t, a_t) \tag{8}$$

4.1. On-Policy Actor-Critic (σ)

By observing equations 5 and 8, we can similarly derive a unified actor-critic approach, where the critic update would depend on the σ parameter. In other words, the critic can either be updated by using on-policy SARSA updates, or by using an off-policy tree backup update, depending on the σ parameter. The critic update can therefore be written as

$$G_t^{(1)} = R_{t+1} + \gamma \big[\sigma_{t+1} Q^w(s_{t+1}, a_{t+1}) + (1 - \sigma_{t+1}) V^w(s_{t+1})\big] \tag{9}$$

$$\delta_t = G_t^{(1)} - Q^w(s_t, a_t) \tag{10}$$

$$w_{t+1} = w_t + \alpha \delta_t \nabla_w Q^w(s_t, a_t) \tag{11}$$

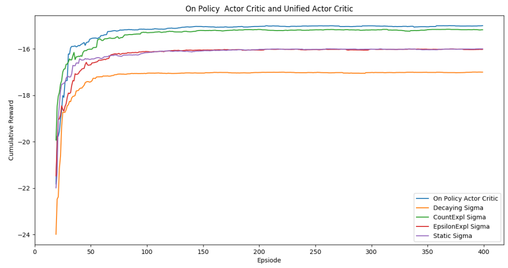

Figure 5. On-Policy Actor Critic and Unified Actor Critic

Figure 5 however shows that the naive on-policy actor-critic algorithm performs better than our proposed actor-critic(σ) algorithm. This is because the on-policy critic in actor-critic uses SARSA updates which is an efficient on-policy algorithm known to perform well with guaranteed convergence in case of function approximation. Our proposed σ varying shedules does not perform well for the on-policy actor-critic (σ) since the σ parameter at times encourages the tree backup update which is known to perform well during off-policy learning, considering expectation over all samples. The SARSA update is based on an action sampled using the current policy and therfore the naive on-policy actor-critic is supposed to be working better.

4.2. Off-Policy Actor-Critic (σ)

Similarly, we can derive an off-policy actor-critic (σ) where the critic would now be updated based on the off-policy samples, and the σ parameter would unify off-policy n-step SARSA with off-policy tree backup algorithm, similar to equations in 10 and 11. The tree backup update in the critic would use an expectation over off-policy action samples for $\sigma = 1$ and sample an off-policy action for $\sigma = 0$.

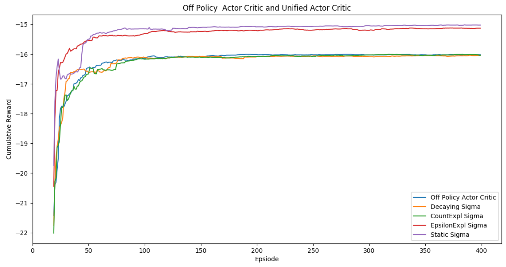

Figure 6. Off-Policy Actor Critic and Unified Actor Critic

Figure 6 however shows that our proposed actor-critic(σ) algorithm performs better than the naive off-policy critic. This is because the σ parameter encourages between SARSA and Tree Backup now, where the actions are from the exploratory policy. The tree backup considers expectation over all possible exploratory actions compared to Q learning in off-policy critic, our proposed critic update rule based on the σ parameter has better converges guarantees and performs significantly better than the niave off-policy actor-critic algorithm.

5. Unifying On-Policy and Off-Policy Learning in Actor-Critic (σ)

From above, using an off-policy one-step SARSA update for the critic with $\sigma = 1$ might not lead to convergence of the action-value function. We therefore unify on-policy and off-policy for actor-critic methods. The critic estimate would now use samples from both the target policy and the exploratory behaviour policy. The tree backup estimate would use all off-policy actions for expectation whereas the SARSA update would be based on on-policy samples. The σ parameter can again be varied based on the amount of exploration. Considering count based or ϵ-greedy exploration, during the exploration phase (where states have not been suffiently visited or for large ϵ values in ϵ-greedy, the σ parameter during exploration phase can be set to 0 such that the critic is updated based on an expectation over all off-policy actions. During exploitation, $\sigma = 1$ and we use on-policy sampled action for SARSA update in the critic. This can be written for the TD error in the critic as follows

$$G_t^{(1)} = R_{t+1} + \gamma\big[\sigma Q^w(s_{t+1}, a_{t+1}) + (1-\sigma)V_{t+1}\big] \quad (12)$$

$$V_{t+1} = \sum_a \mu(a|s_t)Q^w(s_t, a_t) \quad (13)$$

$$\delta_t = G_t^{(1)} - Q_t^w(s_t, a_t) \quad (14)$$

$$w_{t+1} = w_t + \alpha\delta_t\nabla_w Q^w(s_t, a_t) \quad (15)$$

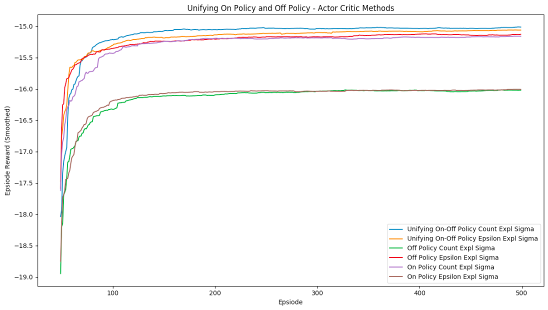

Figure 7. Unifying On-Policy and Off-Policy Actor Critic Methods

Interestingly, figure 7 shows that our proposed approach of unifying on-policy learning with off-policy learning in case of actor-critic methods performs better than our proposed actor-critic(σ) on-policy and off-policy algorithms, with exploration dependent σ. The result in figure 5 shows a promising result that a mixture of on-policy and off-policy learning can be used on actor-critic policy gradient methods. This may open doors for further research as policy gradient methods, being on-policy and off-policy has their own limitations. Therfore, a method that can unify both on-policy and off-policy learning can perhaps lower such limitations and drawbacks, such as sample complexity ahd higher variance of on-policy policy gradients, Monte-Carlo methods. Similarly, naive off-policy policy gradient methods often suffer from convergence issues. A method to unify both can reduce sample complexity and variance in gradeitn estimates, and also may ensure better convergence guarantees.

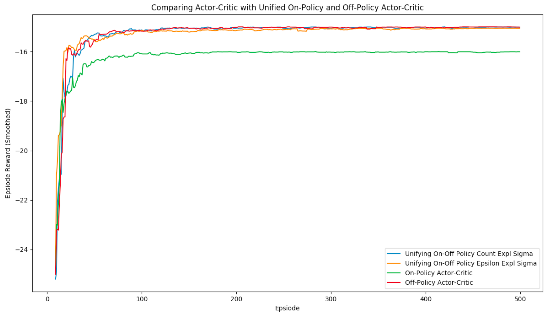

Figure 8. Comparing Actor Critic with Unified On-Policy and Off-Policy Actor Critic

Figure 8 further compares our proposed unified on-policy and off-policy actor-critic method with the naive on-policy actor-critic and off-policy actor-critic. Result in figure 8 shows that our proposed approach performs as much better as the naive on-policy actor-critic (where the critic is updated using SARSA), and significantly better than the off-policy actor-critic, with linear function approximation in the windy grid world environment.

6. Q(σ, λ) with Eligibility Traces

As an addition to our work, we also propose to the Q(σ, λ) algorithm which has a simple incremental implementation using eligibility traces. The Q(σ, λ) algorithm unifies one-step Q(σ) with Monte-Carlo methods. With the use of eligibiliy traces and the λ trace decay parameter, we can create algorithms with Monte-Carlo methods at one end (with $\lambda = 1$) and one step TD learning (with $\lambda = 0$). The Q(σ, λ) algorithm can therefore be unified for one step and multi-step returns with a backward view implementation using eligibility traces. For intermediate values of λ and with different σ schedules, we can balance between SARSA and Tree Backup updates in the n-step returns for the Q(σ, λ) algorithm. In this case, we also investigated Q(σ, λ) with dynamically varying σ and σ dependent on the amount of exploration, and results are shown for the case of linear function approximation. One future avenue of research would be to again use a mixture policy, combining on-policy and off-policy learning in multi-step n-step returns using eligibility traces. The following equations show the TD error and the eligiblity trace updates for the Q(σ, λ) algorithm

$$\delta_t = R_{t+1} + \gamma \big[\sigma Q^w(s_{t+1}, a_{t+1}) + (1-\sigma) V^w\big] - Q^w(s_t, a_t) \tag{16}$$

$$e_t \leftarrow \sigma\gamma\lambda e_{t-1} + (1-\sigma)\gamma\lambda\pi e_{t-1} + \nabla_w Q^w \tag{17}$$

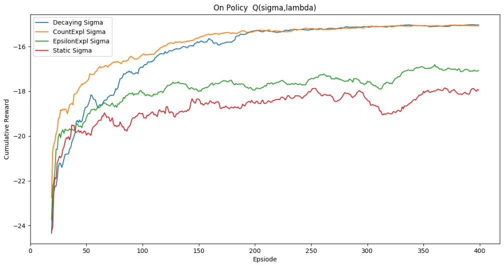

Figure 9. On-Policy $Q(\sigma, \lambda)$

Fig 9 shows the results for on-policy $Q(\sigma, \lambda)$ algorithm. Our results show that varying σ with a count based exploration strategy gives a higher average cumulative reward. The experiment was carried out keeping $\lambda = 0.8$.

Similar variations of σ was made for off-policy $Q(\sigma, \lambda)$ shown in Fig 6.2

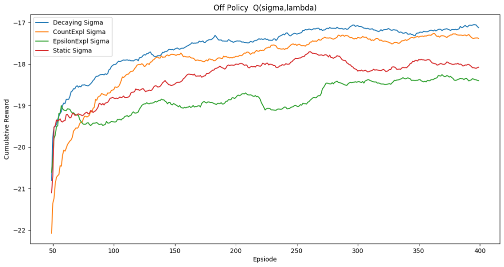

Figure 10. Off-Policy $Q(\sigma, \lambda)$

7. Conclusion and Future Work

We presented a unified approach to combine on-policy learning with off-policy learning, inspired from the unifying Q(σ) algorithm. We studied our unified mixture policy approach on both TD style algorithms and actor-critic policy gradient methods. We presented a σ scheduling parameter depending on amount of exploration of the environment. If the agent encounters an unexplored state, then it considers an expectation over all actions as in the tree backup algorithm, whereas if the state has been visited sufficiently, then it should exploit and sample an action based on current policy as in the SARSA algorithm. We show that our combined mixture on-polic and off-policy algorithm can perform better than the naive approach. We also presented our approach in case of actor-critic methods, where the critic is updated based on the unified on and off policy. This is to balance between the stable, unbiased on-policy gradient estimation with the sample efficient off-policy critic. Our method of unified actor-critic performs better than the naive on-policy and off-policy actor-critic algorithm. Finally, as an extension to the Q(σ) algorithm, we presented the Q(σ, λ) algorithm with eligibility traces. Our results were all demonstrated in the windy grid world environment with linear function approximation.

Our approach was limited to the case of linear function approximation. However, it would be more interesting to consider our unified mixture on-policy and off-policy approach to the case of non-linear function approximations as in DQNs for solving large scale problems. A unifying on-policy and off-policy approach for policy gradient methods may reduce variance of policy gradient REINFORCE type estimations and also make actor-critic methods more sample efficient. Furthermore, we presented results for one-step TD control. It would be interesting to see whether similar performance can be achieved with multi-step methods. It would also be interesting to have theoretical proofs and justifications for our proposed Q(σ, λ) algorithm, and show its performance on other environments.

Acknowledgements

The authors thank Doina Precup and Pierre-Luc Bacon for insights and discussions contributing to the results presented in this paper, and the entire McGill Reasoning and Learning Lab for providing the environment to nurture and support this research.

References

Asis, Kristopher De, Hernandez-Garcia, J. Fernando, Holland, G. Zacharias, and Sutton, Richard S. Multi-step reinforcement learning: A unifying algorithm. *CoRR*, abs/1703.01327, 2017. URL http://arxiv.org/abs/1703.01327.

Bhatnagar, Shalabh, Sutton, Richard S., Ghavamzadeh, Mohammad, and Lee, Mark. Incremental natural actor-critic algorithms. In *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*, pp. 105–112, 2007.

Degris, Thomas, White, Martha, and Sutton, Richard S. Linear off-policy actor-critic. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*, 2012. URL http://icml.cc/2012/papers/268.pdf.

Peters, Jan and Schaal, Stefan. Policy gradient methods for robotics. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2006, October 9-15, 2006, Beijing, China*, pp. 2219–2225, 2006. doi: 10.1109/IROS.2006.282564. URL http://dx.doi.org/10.1109/IROS.2006.282564.

Peters, Jan and Schaal, Stefan. Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190, 2008. doi: 10.1016/j.neucom.2007.11.026.

Precup, Doina, Sutton, Richard S., and Singh, Satinder P. Eligibility traces for off-policy policy evaluation. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000), Stanford University, Stanford, CA, USA, June 29 - July 2, 2000*, pp. 759–766, 2000.

Schulman, John, Levine, Sergey, Abbeel, Pieter, Jordan, Michael I., and Moritz, Philipp. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pp. 1889–1897, 2015.

Silver, David, Lever, Guy, Heess, Nicolas, Degris, Thomas, Wierstra, Daan, and Riedmiller, Martin A. Deterministic policy gradient algorithms. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pp. 387–395, 2014.

Sutton, Richard S. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988. doi: 10.1007/BF00115009. URL http://dx.doi.org/10.1007/BF00115009.

Sutton, Richard S. and Barto, Andrew G. Reinforcement learning: An introduction. *IEEE Trans. Neural Networks*, 9(5):1054–1054, 1998. doi: 10.1109/TNN.1998.712192. URL http://dx.doi.org/10.1109/TNN.1998.712192.

Sutton, Richard S., McAllester, David A., Singh, Satinder P., and Mansour, Yishay. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999]*, pp. 1057–1063, 1999.

van Seijen, Harm, van Hasselt, Hado, Whiteson, Shimon, and Wiering, Marco A. A theoretical and empirical analysis of expected sarsa. In *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning, ADPRL 2009, Nashville, TN, USA, March 31 - April 1, 2009*, pp. 177–184, 2009. doi: 10.1109/ADPRL.2009.4927542. URL http://dx.doi.org/10.1109/ADPRL.2009.4927542.

Watkins, Christopher John Cornish Hellaby. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, UK, May 1989. URL http://www.cs.rhul.ac.uk/~chrisw/new_thesis.pdf.