Continuous Tactical Optimism and Pessimism

Kartik Bharadwaj$^+$, Chandrashekhar Lakshminarayan$^+$, and Balaraman Ravindran$^+$
$^+$Robert Bosch Centre for Data Science and Artificial Intelligence,
Department of Computer Science and Engineering,
Indian Institute of Technology Madras, Chennai, India

22nd May 2023

Abstract

The Tactical Optimism and Pessimism (TOP)[1] framework from reinforcement learning can be deployed in robotics by incorporating it into the decision-making process of an autonomous system. The TOP framework allows a robot to dynamically adjust its optimism or pessimism level based on the uncertainty in its environment and its level of risk tolerance. By being tactically optimistic, the robot takes exploratory actions to gather more information about its environment, which can be useful in unfamiliar or uncertain situations. Conversely, by being tactically pessimistic, the robot takes cautious actions to mitigate risks and avoid potentially dangerous situations. This framework enables the robot to strike a balance between gathering information and ensuring safety, thereby enhancing its decision-making capabilities in complex and uncertain real-world scenarios. TOP hypothesize that the efficacy of an optimistic strategy depends on the environment, the learning stage, and the overall context in which a learner is embedded. Therefore, they propose to view optimism/pessimism as a spectrum and investigate procedures that actively move along that spectrum during the learning process. TOP formulates the optimism/pessimism dilemma as a $k-$armed bandit problem. However, deciding arm values and the number of arms depends on each environment, and finding an optimal set of arms becomes more of a hyper-parameter search. In this work, we propose learning the degree of optimism/pessimism while the agent interacts online with the environment.

1 Introduction

Reinforcement learning (RL) has been increasingly successful, particularly with the use of deep neural networks for value function approximation. Despite recent progress [2, 3], these methods still require millions of interactions with the environment to achieve satisfactory performance on moderately complex control problems. This means that deploying these algorithms can be prohibitively expensive in systems where collecting samples is costly. Another challenge arises due to the use of function approximators that can introduce positive bias in value computation. This can cause an overestimation of the expected reward, which might result in the exploration of states and actions that would not otherwise be explored. However, without a proper understanding of the nature of the overestimation, such exploration can be risky. There are two opposing views on addressing this tension in the literature on RL approaches to continuous-control problems: one seeks to correct the overestimation. At the same time, the other side argues that being optimistic can be helpful in encouraging exploration. Alternatively, RL agents can benefit by varying their optimism and pessimism based on the task they are trying to accomplish. For instance, in the exploration-exploitation trade-off, increasing optimism allows the agent to explore more, as they are more likely to take actions that have not been tried before. In contrast, increasing pessimism makes it more likely for the agent to choose actions that have proven to be effective in the past. Moreover, in some tasks, pessimistic agents are more likely to avoid taking risky actions whose consequences can be severe, while optimistic agents are more risk-seeking likely to take risks in the hope of obtaining a larger reward. Also, varying optimism and pessimism can be helpful in dealing with non-stationary environments, where the true value of states and actions may change over time. By adapting to the changing environment, the agent can maintain good performance and avoid getting stuck in obsolete policies. Tactical Optimism and Pessimism (TOP) [1] hypothesize that the degree of action-value estimation bias and subsequent efficacy of an optimistic strategy depends on the environment, the learning stage, and the overall context in which a learner is embedded. Therefore, they propose to view optimism/pessimism as a spectrum and investigate procedures that actively move along that spectrum during the learning process. They measure two forms of uncertainty that arise during learning: aleatoric uncertainty and epistemic uncertainty, and aim to control their effects. TOP acknowledges the inherent uncertainty in the level of estimation bias present and estimates the optimal approach on the fly by formulating the optimism/pessimism dilemma as a $k-$armed bandit problem. However, deciding arm values and the number of arms depends on each environment, and finding an optimal set of arms becomes more of a hyper-parameter search. In this work, we propose learning the degree of optimism/pessimism while the agent interacts online with the environment.

2 Related Work

The recent success in deep reinforcement learning has been attributed to improvements to off-policy actor-critic algorithms. Specifically, the Deterministic Policy Gradient (DPG) [4] method, which underpins this work, is based on a deterministic policy. DDPG [5] combines the deterministic policy gradient [4] with off-policy learning using neural networks architecture based on DQN [6]. This effort leads to an off-policy actor-critic architecture in which the actor's gradients rely solely on derivatives via the trained critic. This indicates that boosting the critic's evaluation immediately improves the actor gradients.

Optimistic algorithms are built upon the principle of "optimism in the face of uncertainty" (OFU). They operate by maintaining a set of statistically plausible models of the world and selecting actions to maximize the returns in the best plausible world. Such algorithms were first studied in the context of multi-armed bandit problems [7], and went on to inspire numerous algorithms for reinforcement learning.

The off-policy exploration strategy used in OAC [8] is designed to optimize the upper confidence bound on the critic, which is derived from an epistemic uncertainty estimate on the Q-function obtained using the bootstrap method [9]. Unfortunately, due to the difficulties associated with maintaining low estimation error when using function approximation, attempts to establish an upper bound on the true value function have had limited success.

Tactical Optimism and Pessimism (TOP) [1] integrates DPG with distributional value estimate, just like in D4PG [10]. D4PG uses a categorical distribution, whereas TOP uses two critics and a quantile representation. TOP models the trade-off between optimism and pessimism as a discrete multi-armed bandit problem. While optimism can help with exploration if there is a considerable estimating error, a more pessimistic strategy may be required to stabilize learning. Furthermore, both techniques have resulted in algorithms that are backed by substantial empirical data. TOP seeks to unify these seemingly contrary views by postulating that the respective contributions of optimism and pessimism elements can change depending on the environment. Unlike TOP, which uses $k-$armed bandits, our approach is to learn the degree of optimism/pessimism β to actively vary the level of optimism/pessimism in its value estimates.

3 Off-policy Reinforcement Learning

Reinforcement learning considers the framework where an agent interacts with its environment with the goal of learning to maximize its cumulative reward. Ideally, an environment is cast as a Markov Decision Process (MDP), formally defined as a tuple $\langle \mathcal{S}, \mathcal{A}, p, r, \gamma \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the space of possible actions, $p : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$ is the transition function, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\gamma \in [0, 1)$ is the discount factor. For a given policy π, the return $Z_\pi = \sum_t \gamma^t r_t$, is a random variable describing the sum of discounted rewards, starting at state s_t, and observed along one episode obtained by unrolling policy π until some time horizon T, potentially infinite. γ determines the priority of short-term rewards. Given a set of policies parameterized by θ, $\{\pi_\theta : \theta \in \Theta\}$, the goal is to update θ so as to maximize the expected return, or discounted cumulative reward, $J(\theta) = \mathbb{E}_\pi[\sum_t \gamma^t r_t] = \mathbb{E}[Z_\pi]$.

Actor-critic algorithms provide a solution for maximizing the expected cumulative reward in Deep RL by using two separate components: the actor and the critic. The actor, represented by the policy π, is trained to make decisions that maximize the expected return. The critic, in the form of a value function, evaluates the actions of the policy by predicting the expected return under the current policy, $Q_\pi(s, a) := \mathbb{E}_{s_i \sim p_\pi, a_i \sim \pi}[Z_t | s_t = s, a_t = a]$. In continuous control, parameterized policies π_θ are updated by taking the gradient of the expected return $\nabla_\theta J(\theta)$. In the actor-critic framework, the policy can be updated using gradient ascent through the deterministic policy gradient algorithm [4]:

$$\nabla_\theta J(\theta) = \mathbb{E}_\pi[\nabla_a Q_\pi(s, a)|_{a=\pi(s)} \nabla_\theta \pi_\theta(s)] \tag{1}$$

In Q-learning, the action-value function can be learned using temporal difference learning [11], [12], an update rule based on the Bellman equation [13]. The Bellman equation is a fundamental relationship between the value of a state-action pair (s, a) and the value of the next state-action pair (s', a'):

$$Q_\pi(s, a) = r + \gamma \mathbb{E}_{s', a'}[Q_\pi(s', a')], a' \sim \pi(s') \tag{2}$$

For large continuous state space, the value can be estimated with a differentiable function approximator $Q_\phi(s, a)$, with parameters ϕ. In deep Q-learning [6], the network is updated by using temporal difference learning with a secondary frozen target network $Q'_\phi(s, a)$ to maintain a fixed target y over multiple updates:

$$y = r + \gamma Q'_\phi(s', a'), a' \sim \pi'_\theta(s'), \tag{3}$$

where actions are selected from a target action network π'_θ. The weights of the target networks are periodically updated to match the weights of the current network by some proportion τ at each time step as:

$$\theta' \leftarrow \tau\theta + (1 - \tau)\theta' \tag{4}$$

This update can be applied in an off-policy fashion, sampling random mini-batches of transitions from an experience replay buffer [14].

4 Our approach

4.1 Learning the degree of optimism as a $k-$armed bandit problem

It would be beneficial for an agent to vary its degree of optimism or pessimism β as the reward distribution varies across environments. TOP casts the problem of choosing β as a $k-$armed bandit framework to adjust its degree of optimism. Before the start of every episode m, the agent picks arm d_m from a set of D arms, each taking a discrete value $\{\beta_d\}_{d=1}^D$. These arms follow a distribution $\mathbf{p}_m \in \delta_D$. Bandit feedback is of the form $f_m = R_m - R_{m-1}$, which tells us the absolute level of performance associated with selecting an arm. Weights of the $k-$arms are learned using the Exponential Weighted Average Forecasting algorithm [15].

4.2 Learning β using gradient-based algorithms

Rather than choosing β as a hyper-parameter, we want to learn it. Many optimization algorithms assume a setting where information about the gradient of the loss function with respect to the parameters being optimized is available. We can update β for each episode m or sample from the replay buffer. Updating β has a direct effect on the exploration strategy and sample efficiency. To update β for each sample from the replay buffer, we would need the gradient signal characteristic of the potential update to the value β, which will lead to faster convergence towards optimal action-value estimates. Since we cannot define an action-values distribution in continuous state-action pairs, it is challenging to update reliably β. Similar to updating β for each sample from the replay buffer, updating β for each episode m is also challenging. The non-stationarity of the action-values estimates also adds to the difficulty of finding a reliable gradient estimate to update β. In essence, obtaining reliable knowledge of the relationship between β and Q(s, a) may not be possible, implying that the gradient-based algorithms may be infeasible or undependable. Moreover, the cost of achieving effective convergence depends on the number of required iterations and the cost per iteration, which is usually higher in gradient-based algorithms. Additionally, the convergence rates when using function approximators such as neural networks may not accurately reflect practical convergence rates in limited sample sizes.

One workaround to update β for each episode m is to use absolute feedback $f_m = R_m - R_{m-1}$. While this does not tell us anything about the relative feedback of using a specific value of β over time, f_m serves as a good proxy, especially in our case, where the domain of $\beta \in \mathbb{R}$, and where we need to find the optimal β in 1M timesteps. Hence, current gradient-based algorithms act as a bottleneck to update β. Instead, we use gradient-free optimization algorithms like SPSA which we describe in the next subsection.

4.3 Simultaneous perturbation Stochastic Approximation (SPSA)

Simultaneous Perturbation Stochastic Approximation (SPSA)[16] is a general recursive optimization algorithm that is specifically useful when information associated with the gradient vector of the loss function is not available or is too resource intensive to compute. Instead, gradient-free algorithms similar to SPSA are based on approximating the gradient formed from noisy measurements of the loss function.

Let $L(\theta)$ be a differentiable loss function where θ is an n-dimensional vector. There exists a θ^* at which $\frac{\partial L}{\partial \theta} = 0$. SPSA starts with an initial parameter vector $\hat{\theta_0}$. Its update rule uses the stochastic approximation scheme given by:

$$\hat{\theta}_{k+1} = \hat{\theta}_k - \eta_k \hat{g}_k(\hat{\theta}_k) \tag{5}$$

Table 1: Average reward over five seeds on Mujoco tasks, trained for 1M time steps. ± values denote one standard deviation across trials. Values within one standard deviation of the highest performance are listed in bold

Task	*cTOP*	*TOP-TD3*	*TD3*
Hopper-v2	3630 ± 120	**3695 ± 152**	3367 ± 136
Walker2d-v2	**5672 ± 447**	5408 ± 87	4529 ± 360
HalfCheetah-v2	**13148 ± 438**	13076 ± 357	12321 ± 657

where $\hat{\theta}_k$ is the $k-$th feasible solution, $\eta_k \in \mathbb{R}$ is the learning rate, and $g_k \in \mathbb{R}$ is an iterative direction, a stochastic estimate of the gradient.

Our goal is the following unconstrained optimization to find the degree of optimism/pessimism β that maximizes expected return $J(\theta)$:

$$\text{(P)} \max_{\beta \in \mathbb{R}} J(\theta) \tag{6}$$

There are two versions of approximating g_m: One-sided and Two-sided. One-sided gradient approximations involve measuring $L(\beta_m)$ and $L(\beta_m + \text{perturbation})$, whereas two-sided gradient approximations measures $L(\beta_m \pm \text{perturbation})$. Before episode m begins, we perturb β_m to estimate the gradient. We use one-sided gradient approximation as it is relatively cheap to compute compared to two-sided gradient approximations. Hence, our gradient estimate is:

$$g_m = L(\beta_m + c_m \cdot \Delta_m)\frac{\Delta_m}{c_m} \tag{7}$$

where $\Delta_m \sim \mathcal{N}(0,1)$. In our case, the feedback is our loss function L. Our feedback mechanism is the same as TOP:

$$L_{\text{sb}} = R_{m+1} - R_m \tag{8}$$

One of the challenges in directly using SPSA is that the learning rate η_m must decay over the entire course of the agent's training experience to satisfy convergence properties. This means we would need to specify how many episodes M the agent will experience in its training lifetime. M now becomes a random variable as it is dependent on different environments. To mitigate this problem, we use Exponential Weighted Averaging over the recent five feedback samples.

5 Experiments

The key question we would like to answer is whether learning the degree of optimism/pessimism β help, rather than pre-specifying the arm values when framed as a multi-armed bandit problem. We test this hypothesis by running experiments on three state-based continuous control tasks from the Mujoco framework [17] via OpenAI gym [18]. We use TD3 and TOP-TD3 as baselines. TD3 uses the default hyperparameters configuration. TOP-TD3 uses the arm settings $\{-1, 0\}$, with $\beta = -1$ corresponding to a pessimistic arm and $\beta = 0$ corresponds to average of critics' quantiles. Hyperparameters were kept constant for all environments. Each algorithm was trained for one million steps and repeated for five random seeds. Our results displayed in Table 1 show cTOP outperforming TOP-TD3 and TD3 for Walker2d-v2 and HalfCheetah-v2, while marginally underperforming for Hopper-v2.

6 Conclusion

We empirically demonstrate that learning the degree of optimism β while training the agent is often helpful across tasks. To vary the degree of optimism, previous off-policy RL algorithms either rely on a fixed value of optimism or may have to pre-specify the arm values when framed as a multi-armed bandit problem. We show that cTOP adaptively updates its degree of optimism while training the agent in order to achieve maximum return.

One limitation of our algorithm is stability. While cTOP achieves a better average return for Walker2d-v2 and HalfCheetah-v2, there is a marginal drop in performance for Hopper-v2. We believe this is due to the non-decaying nature of EWMA on the learning rate. This might lead to noisy updates, which could result in divergence from the optimal action-value function. A natural extension will be to use meta-gradients to learn β robustly.

References

[1] Ted Moskovitz, Jack Parker-Holder, Aldo Pacchiano, Michael Arbel, and Michael Jordan. Tactical optimism and pessimism for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, 2021.

[2] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1587–1596. PMLR, 2018.

[3] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications.

[4] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *Proceedings of the 31st International Conference on Machine Learning*, number 1 in Proceedings of Machine Learning Research, pages 387–395, Bejing, China, 2014. PMLR.

[5] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[6] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[7] Ronen I. Brafman and Moshe Tennenholtz. R-max - a general polynomial time algorithm for near-optimal reinforcement learning. *J. Mach. Learn. Res.*, 3:213–231, mar 2003.

[8] Kamil Ciosek, Quan Vuong, Robert Loftin, and Katja Hofmann. Better exploration with optimistic actor critic. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, 2019.

[9] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29, 2016.

[10] Gabriel Barth-Maron, Matthew W. Hoffman, David Budden, Will Dabney, Dan Horgan, Dhruva TB, Alistair Muldal, Nicolas Heess, and Timothy P. Lillicrap. Distributed distributional deterministic policy gradients. In *6th International Conference on Learning Representations, ICLR 2018, April 30 - May 3, 2018, Conference Track Proceedings*, Vancouver, BC, Canada, 2018. OpenReview.net.

[11] Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1):9–44, August 1988.

[12] Christopher John Cornish Hellaby Watkins. *Learning from Delayed Rewards.* PhD thesis, King's College, Cambridge, UK, May 1989.

[13] Richard Bellman. *Dynamic Programming.* Princeton University Press, Princeton, NJ, USA, 1 edition, 1957.

[14] Long-Ji Lin. *Reinforcement Learning for Robots Using Neural Networks.* PhD thesis, USA, 1992.

[15] Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games.* Cambridge University Press, 2006.

[16] James C. Spall. An overview of the simultaneous perturbation method for efficient optimization.

[17] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012.

[18] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *CoRR*, abs/1606.01540, 2016.