Off-Policy Deep Reinforcement Learning by Bootstrapping the Covariate Shift

Carles Gelada, Marc G. Bellemare
Google Brain
cgel@google.com, bellemare@google.com

Abstract

In this paper we revisit the method of off-policy corrections for reinforcement learning (COP-TD) pioneered by Hallak et al. (2017). Under this method, online updates to the value function are reweighted to avoid divergence issues typical of off-policy learning. While Hallak et al.'s solution is appealing, it cannot easily be transferred to nonlinear function approximation. First, it requires a projection step onto the probability simplex; second, even though the operator describing the expected behavior of the off-policy learning algorithm is convergent, it is not known to be a contraction mapping, and hence, may be more unstable in practice. We address these two issues by introducing a discount factor into COP-TD. We analyze the behavior of discounted COP-TD and find it better behaved from a theoretical perspective. We also propose an alternative soft normalization penalty that can be minimized online and obviates the need for an explicit projection step. We complement our analysis with an empirical evaluation of the two techniques in an off-policy setting on the game Pong from the Atari domain where we find discounted COP-TD to be better behaved in practice than the soft normalization penalty. Finally, we perform a more extensive evaluation of discounted COP-TD in 5 games of the Atari domain, where we find performance gains for our approach.

Introduction

Central to reinforcement learning is the idea that an agent should learn from experience. While many algorithms learn in a purely online fashion, sample-efficient methods typically make use of past data, viewed either as a fixed dataset, or stored in a replay memory (Lin 1993, Mnih et al. 2015). Because this past data may not be generated according to the policy currently under evaluation, the agent is said to be learning *off-policy* (Sutton and Barto 2018).

By now it is well-documented that off-policy learning may carry a significant cost when combined to function approximation. Early results have shown that estimating the value function off-policy, using Bellman updates, may diverge (Baird 1995), (Tsitsiklis and Van Roy 1997). More recently, value divergence was perhaps the most significant issue dealt with in the design of the DQN agent (Mnih et al. 2015), and remains a source of concern in deep reinforcement learning (van Hasselt, Guez, and Silver 2016).

Copyright © 2019, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Further, under off-policy learning, the quality of the Bellman fixed point suffers as studied by Kolter (2011) and Munos (2003). The value function error can be unboundedly large even if the value function can be perfectly approximated. Hence, even in the case where convergence to the fixed point with off-policy data occurs, solutions can be of poor quality. Thus, the existing TD learning algorithms with convergence guarantees under off-policy data (Maei et al. 2009), (Sutton et al. 2009) can still suffer from off-policy issues.

This paper studies the covariate shift method for dealing with the off-policy problem. The covariate shift method, studied by Hallak and Mannor (2017) and Sutton, Mahmood, and White (2016), reweights online updates according to the ratio of the target and behavior stationary distributions. Under optimal conditions, the covariate shift method recovers convergent behavior with linear approximation, breaking what Sutton and Barto (2018) call the "deadly triad" of reinforcement learning. We argue the method is particularly appealing in the context of replay memories, where the reweighting can be replaced by a reprioritization scheme similar to that of Schaul et al. (2016).

We improve on Hallak and Mannor's COP-TD algorithm, which has provable guarantees but is difficult to implement in a deep reinforcement learning setting. First, we introduce a discount factor into their update rule to obtain a more stable algorithm. Second, we develop an alternative normalization scheme that can be combined with deep networks, avoiding the projection step necessary in the original algorithm. We perform an empirical study of the two methods and their variants on the game Pong from the Arcade Learning Environment and find that our improvements give rise to significant benefits for off-policy learning.

Background

We study the standard RL setting in which an agent interacts with an environment by observing a state, selecting an action, receiving a reward, and observing the next state. We model this process with a Markov Decision Process $\langle \mathcal{S}, \mathcal{A}, R, P, \gamma \rangle$. Here, $\mathcal{S}$, $\mathcal{A}$ denote the state and action spaces and P is the transition function. Throughout we will assume that $\mathcal{S}$ and $\mathcal{A}$ are finite and write $n := |\mathcal{S}|$. A policy π maps a state to a distribution over actions, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and γ is the discount factor.

We are interested in the *policy evaluation* problem, where we seek to learn the value function V^π of a policy from samples. The value function is the expected sum of discounted rewards from a state when following policy π:

$$V^\pi(s) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) | s_0 = s\right],$$

where each action is drawn from the policy π, i.e. $a_t \sim \pi(\cdot \mid s_t)$, and states are drawn from the transition function: $s_{t+1} \sim P(\cdot \mid s_t, a_t)$. We combine the policy π and transition function P into a state-to-state transition function P_π, whose entries are

$$P_\pi(s' \mid s) := \sum_{a \in \mathcal{A}} \pi(a \mid s) P(s' \mid s, a).$$

Let $r_\pi(s) := \mathbb{E}_{a \sim \pi} R(s, a)$ be the expected reward under π. One of the key properties of the value function V^π is that it satisfies the *Bellman equation*:

$$V^\pi(s) = r_\pi(s) + \gamma \mathop{\mathbb{E}}_{s' \sim P_\pi} V^\pi(s').$$

In vector notation (Puterman 1994), this becomes

$$V^\pi = r_\pi + \gamma P_\pi V^\pi,$$

where $V^\pi \in \mathbb{R}^n$, $r^\pi \in \mathbb{R}^n$, and $P_\pi \in \mathbb{R}^{n \times n}$. The value function is in fact the fixed point of the *Bellman operator* $\mathcal{T}_\pi : \mathbb{R}^n \to \mathbb{R}^n$, defined as

$$\mathcal{T}_\pi V := r_\pi + \gamma P_\pi V.$$

The Bellman operator describes a single step of dynamic programming (Bellman 1957) or *bootstrap*; the process $V^{k+1} := \mathcal{T}_\pi V^k$ converges to V^π. More interestingly for us, the operator also describes the expected behavior of learning rules such as temporal-difference learning (Sutton 1988) and consequently their learning dynamics (Tsitsiklis and Van Roy 1997). In the sequel, whenever we analyze the behavior of operators it is with this relationship in mind.

In this paper we consider the process of learning V^π from samples drawn from P and a *behavior policy* μ. Following standard usage, we call this process *off-policy learning* (Sutton and Barto 2018).

Let $d \in \mathbb{R}^n$. We write D_d for the corresponding diagonal matrix. For a matrix $A \in \mathbb{R}^{n \times n}$, the A-weighted squared seminorm of a vector $x \in \mathbb{R}^n$ is $\|x\|_A^2 := \|Ax\|^2 = x^\top A^\top A x$. We specialize this notation to vectors in $\mathbb{R}^n$ as $\|x\|_d^2 := \sum_{i=1}^n d(i) x(i)^2$. We write e for the vector of all ones and $\Delta(\mathcal{S})$ for the simplex over states ($d \in \Delta(\mathcal{S}) \implies d^\top e = 1, d \geq 0$). Finally, recall that $d \in \Delta(\mathcal{S})$ is the stationary distribution of a transition function P if and only if

$$d = dP.$$

This distribution is unique when P defines a Markov chain with a single recurrent class (called a unichain, Meyn and Tweedie 2012). Throughout we will write d_π and d_μ for the stationary distributions of P_π and P_μ, respectively.

Off-Policy Learning with Linear Approximation

In most practical applications the size of the state space precludes so-called tabular representations, which learn the value of each state separately. Instead, one must approximate the value function. One common scheme is *linear function approximation*, which uses a mapping from states to features $\phi : \mathcal{S} \to \mathbb{R}^k$. The approximate value function at s is then the inner product of a feature vector with a vector of weights $\theta \in \mathbb{R}^k$:

$$\hat{V}(s) = \phi(s)^\top \theta. \tag{1}$$

If $\Phi \in \mathbb{R}^{n \times k}$ denotes the matrix of row-feature vectors, (1) becomes, in vector notation:

$$\hat{V} = \Phi\theta.$$

The *semi-gradient update rule* for TD learning (Sutton and Barto 2018) learns an approximation of V^π from sample transitions. Given a starting state $s \in \mathcal{S}$, a successor state $s' \sim P_\pi(\cdot \mid s)$, and a step-size parameter $\alpha > 0$, this update is

$$\theta \leftarrow \theta + \alpha \left[r_\pi(s) + \gamma \phi(s')^\top \theta - \phi(s)^\top \theta\right] \phi(s). \tag{2}$$

While (2) does not correspond to a proper gradient descent procedure (see e.g. Barnard 1993), it can be shown to converge, as we shall now see.

The expected behavior of the semi-gradient update rule is described by the *projected Bellman operator*, denoted $\Pi_d \mathcal{T}_\pi$ for some distribution $d \in \Delta(\mathcal{S})$ (Tsitsiklis and Van Roy 1997). The projected Bellman operator is the combination of the usual Bellman operator with a projection Π_d in norm $\|\cdot\|_d$ onto the span of Φ. Typically, the learning rule (2) is studied in an online setting, where samples correspond to an agent sequentially experiencing the environment. In the simplest case where an agent follows a single behavior policy μ, this corresponds to $d = d_\mu$.

The stationary point of (2), if it exists, is the solution of the *projected Bellman equation*

$$\hat{V}^\pi = \Pi_d \mathcal{T}_\pi \hat{V}^\pi.$$

When the projection is performed under the stationary distribution d_π, (2) converges to this fixed point provided α is taken to satisfy the Robbins-Monro conditions and other mild assumptions(see Tsitsiklis and Van Roy (1997)). Taking $d \neq d_\pi$, however, may lead to divergence of the weight vector in (2). A sign of the importance of this issue can be seen in Sutton and Barto's choice to dub "deadly triad" the combination of off-policy learning, function approximation, and bootstrapping.

A prerequisite to guarantee the convergence of (2) to $\hat{V}^\pi$ is that

$$\hat{V}^{k+1} := \Pi_d \mathcal{T}_\pi \hat{V}^k \tag{3}$$

should also converge for any initial condition $\hat{V}^0 \in \mathbb{R}^n$. Tsitsiklis and Van Roy proved convergence when $d = d_\pi$ by showing that the projected Bellman operator is a contraction in d_π-weighted norm. That is, for any $V, V' \in \mathbb{R}^n$,

$$\left\|\Pi_{d_\pi} \mathcal{T}_\pi V - \Pi_{d_\pi} \mathcal{T}_\pi V'\right\|_{d_\pi} \leq \gamma \left\|V - V'\right\|_{d_\pi},$$

from which an application of Banach's fixed point theorem allows us to conclude that $\hat{V}^k \to \hat{V}^\pi$. More formally, the

result follows from noting that the induced operator norm of $\Pi_{d_\pi} P_\pi \leq 1$. The lack of a similar result when $d \neq d_\pi$ explains the divergence, and is by now well-documented in the literature (Baird 1995).

Independent of the convergence issues raised by off-policy learning, the fixed point of the Bellman equation with linear function approximation is also affected. A metric for the quality of a value function $\hat{V}(s)$ is $\mathbb{E}_{s\sim d_\pi}[(\hat{V}(s) - V^\pi(s))^2] = \left\|\hat{V} - V^\pi\right\|_{d_\pi}$, the expected value prediction error sampling states from the stationary distribution of the policy evaluated. Under some conditions, we can bound the quality of the fixed point under off-policy data as a constant factor times the optimal prediction error $\|\Pi_{d_\pi} V^\pi - V^\pi\|_{d_\pi}$.

Theorem 1. *[Based on Munos 2003] Let $d \in \Delta(\mathcal{S})$ be some arbitrary distribution. Suppose that $\|\Pi_d P_{d_\pi}\|_{d_\pi} < 1/\gamma$ and there is a fixed point $\hat{V}_d^\pi$ to the projected Bellman equation $V := \Pi_d \mathcal{T}_\pi V$. Then its approximation error in d_π-weighted norm is at most*

$$\left\|\hat{V}_d^\pi - V^\pi\right\|_{d_\pi} \leq \frac{\|\Pi_d V^\pi - V^\pi\|_{d_\pi}}{1 - \gamma \|\Pi_d P_\pi\|_{d_\pi}}.$$

Furthermore, this error is minimized when $d = d_\pi$.

Theorem 1 is interesting because it suggests that d_π is also the optimal in the sense that it yields the smallest approximation bound. Kolter (2011) showed that when Theorem 1 does not apply (because $\|\Pi_d P_\pi\|_{d_\pi} \geq 1/\gamma$) it is possible to construct examples where the fixed point error is unbounded, even if $\|\Pi_d V^\pi - V^\pi\|_{d_\pi} = 0$ (i.e. cases where a perfect solution exists). Thus, no general bound on the quality of the off-policy fixed point exists.

Not only do we expect that improvements in off-policy learning should lead to more stable learning behavior, but also to the improved quality of the value functions which, in the control setting, should translate to increases in performance. In this paper we will study the covariant shift approach to off-policy learning, where updates in (2) are reweighting so as to induce a projection under d_π.

The Covariate Shift Approach

Suppose that the stationary distributions d_π and d_μ are known, and that states are updated according to a distribution $s \sim d_\mu$. We use importance sampling (e.g. Precup, Sutton, and Singh 2000) to define the update rule

$$\theta \leftarrow \theta + \alpha \frac{d_\pi(s)}{d_\mu(s)} \left[r(s,a) + \gamma \phi(s')^\top \theta - \phi(s)^\top \theta\right] \phi(s)^\top,$$

where as before $a \sim \mu(\cdot \mid s), s' \sim P(\cdot \mid s, a)$, is equivalent to applying the semi-gradient update rule (2) *under the sampling distribution* d_π. Further multiplying the update term by $\frac{\pi(a \mid s)}{\mu(a \mid s)}$, we recover (in expectation) the semi-gradient update rule for learning V^π, under the sampling distribution d_π (Hallak and Mannor 2017). Thus, provided we reweighted updates correctly, we obtain a provably convergent off-policy algorithm.

The *COP-TD* learning rule proposed by Hallak and Mannor learns the ratio $\frac{d_\pi}{d_\mu}$ from samples. Although much of the original work is concerned with the combined learning dynamics of the value and the ratio, we will focus on the process by which this ratio is learned.

Similar to temporal difference learning, COP-TD estimates $\frac{d_\pi}{d_\mu}$ by bootstrapping from a previous prediction. Given a step-size $\alpha > 0$, a ratio vector $c \in \mathbb{R}^n$ and a sample transition (s, a, s') where $s \sim d_\mu$, $a \sim \mu(\cdot \mid s)$, and $s' \sim P(\cdot \mid s, a)$, COP-TD performs the following update:

$$c(s') \leftarrow c(s') + \alpha \left[\frac{\pi(a \mid s)}{\mu(a \mid s)} c(s) - c(s')\right]. \tag{4}$$

Note that this update rule learns "in reverse" compared to TD learning. The expected behavior of the update rule is captured by the *COP operator* Y:

$$(Yc)(s') := \mathop{\mathbb{E}}_{s\sim d_\mu, a\sim\mu} \left[\frac{\pi(a \mid s)}{\mu(a \mid s)} c(s) \mid s'\right].$$

In vector notation, this operator is:

$$Yc = D_{d_\mu}^{-1} P_\pi^\top D_{d_\mu} c. \tag{5}$$

Any multiple of $\frac{d_\pi}{d_\mu}$ is a fixed point of Y: $Y\beta\frac{d_\pi}{d_\mu} = \beta\frac{d_\pi}{d_\mu}$, for $\beta \in \mathbb{R}$. Hallak and Mannor, under the assumption that the transition matrix P_π has a full set of real eigenvectors, give a partial proof that the iterates $c^{k+1} := Yc^k$ converge to such a fixed point. Our first result is to provide an alternative proof of convergence that does not require this assumption.

Theorem 2. *Suppose that P_π defines an ergodic Markov chain on the state space $\mathcal{S}$, and let $c^0 \in \Delta$. Then the process $c^{k+1} = Yc^k$ converges to $C\frac{d_\pi}{d_\mu}$, where $C \in \mathbb{R}$ is a positive scalar.*

Corollary 1. *Suppose that the conditions of Theorem 2 are met. Define the* normalized COP operator

$$(\bar{Y}c)(s') := \frac{\tilde{c}(s')}{\sum_s \tilde{c}(s)} \qquad \tilde{c} := Yc.$$

Then the unique fixed point of the operator $\bar{Y}$ is the ratio $\frac{d_\pi}{d_\mu}$, to which the process $c^{k+1} := \bar{Y}c^k$ converges.

COP-TD with Linear Function Approximation

The covariate shift method is called-for when the value function is approximated. Under these circumstances, one might expect that we also need to learn an approximate ratio $\hat{c}$. Hallak and Mannor consider the linear approximation

$$\hat{c}(s) = \phi(s)^\top w,$$

where $w \in \mathbb{R}^k$.[1] This gives rise to a semi-gradient update rule similar to (2) but implementing (4):

$$\tilde{w} \leftarrow w + \alpha \left[\frac{\pi(a \mid s)}{\mu(a \mid s)} \phi(s)^\top w - \phi(s')^\top w\right] \phi(s')$$

[1] In practice, we may avoid negative $\hat{c}$'s by clipping them at 0.

and also followed by a projection step on the d_μ-weighted simplex Δ_{Φ,d_μ} defined by the set $W_{\Phi,d_\mu} := \{u \in \mathbb{R}^k : \sum_{s\in\mathcal{S}} d_\mu(s)\phi(s)^\top u = 1, \phi(s)^\top u \geq 0\}$:

$$w \leftarrow \operatorname*{arg\,min}_{u\in W_{\Phi,d_\mu}} \|u - \tilde{w}\|.$$

The projection step ensures that the approximate ratio $\hat{c}$ corresponds to some distribution ration $\frac{d}{d_\mu}$ for $d \in \mathcal{S}$. The combined process is summarized by the normalized COP operator: $\hat{c}^{k+1} := \Pi_{\Delta_{\Phi,d_\mu}} \Pi_d Y \hat{c}^k$, whose repeated application converges to some approximate ratio.

One interesting fact is that the semi-gradient update rule, which corresponds to a d-weighted projection, is by itself insufficient to guarantee the good behavior of the algorithm.

Lemma 1. *Let Y be a symmetric COP-TD operator and Π be the projection onto Φ in L_2 norm. If $\frac{d_\pi}{d_\mu}$ is not in the span of Φ, then $c = 0$ is the only solution to*

$$\Pi Y c = c.$$

Lemma 1 argues that the normalization step is not only a convenience but is in fact necessary for the process to converge to anything meaningful. This is further validated by numerical experiments with general P_π, d_μ and d where we obseve that the repeated application of operator $\Pi_d Y$ either converges to 0 or diverges.

A Practical COP-TD

In this paper we are concerned with the application of COP-TD to practical scenarios, where approximating $\frac{d_\pi}{d_\mu}$ is a must. As the following observations suggest, however, there are a number of limitations to COP-TD.

Lack of contraction factor. The operator Y is not in general a contraction mapping. Hence, while the process $c^{k+1} := Yc^k$ converges, it may do so at a slow rate, with greater variations in the sample-based case, and more importantly may be unstable when combined with function approximation.

Hard-to-satisfy projection step. In the approximate case, we saw that it is necessary to combine the COP operator to a projection onto the d_μ-weighted simplex. Although it is possible to approximate this projection step in an online, sample-based manner for linear function approximation (Hallak and Mannor recommend constraining the weights to the simplex generated by a sufficiently large enough sample), no counterpart exists for more general classes of function approximations, making COP-TD hard to combine with neural networks.

In what follows we address these two issues in turn.

The Discounted COP Learning Rule

While repeated applications of the operator converge to $\frac{d_\pi}{d_\mu}$, the operator is not in general a contraction mapping, and its convergence profile is tied to the (usually unknown) mixing time of the Markov chain described by P_π. Our main contribution is the *$\hat{\gamma}$-discounted COP-TD* learning rule, which recovers COP-TD for $\hat{\gamma} = 1$.

Definition 1. *Let $c \in \mathbb{R}^n$. For a step-size $\alpha > 0$, discount factor $\hat{\gamma} \in [0, 1]$, and sample (s, a, s') drawn respectively from d_μ, μ, and P, the $\hat{\gamma}$-discounted COP-TD learning rule is*

$$c(s') \leftarrow c(s') + \alpha \left[\hat{\gamma}\frac{\pi(a \mid s)}{\mu(a \mid s)}c(s) + (1 - \hat{\gamma}) - c(s')\right]. \quad (6)$$

The corresponding operator is

$$Y_{\hat{\gamma}} c := \hat{\gamma} Y c + (1 - \hat{\gamma})e.$$

By inspection, it is clear that $Y_1 = Y$. However, as we will see, the discounted COP-TD learning rule has several desirable properties compared to its undiscounted counterpart. We begin by characterizing the discounted COP operator.

Definition 2. *For a given $\hat{\gamma} \in [0, 1]$, we define the* discounted reset transition function $\hat{P}_\pi$ *as:*

$$\hat{P}_\pi := \hat{\gamma} P_\pi + (1 - \hat{\gamma}) e d_\mu^\top,$$

where $ed_\mu^\top$ is the matrix whose columns are all d_μ.

The discounted reset transition function can be understood as a process which either transitions as usual with probability $\hat{\gamma}$, or resets to the stationary distribution d_μ with the remainder probability. This is analogous to the perspective of the discount factor as a probability of terminating (White 2017), and is related to the constraint that arises in the dual formulation of the value function (Wang et al. 2008).

We denote by $\hat{d}_\pi$ the stationary distribution satisfying $\hat{d}_\pi = \hat{d}_\pi \hat{P}_\pi$. As an aside, the inclusion of the reset guarantees the ergodicity of the Markov chain defined by $\hat{P}_\pi$.

Proposition 1. *The stationary distribution $\hat{d}_\pi$ is given by*

$$\hat{d}_\pi = (1 - \hat{\gamma})(I - \hat{\gamma} P_\pi^\top)^{-1} d_\mu,$$

where the sum

$$(I - \hat{\gamma} P_\pi^\top)^{-1} := \sum_{t=0}^{\infty} (\hat{\gamma} P_\pi^\top)^t.$$

is convergent for $\hat{\gamma} < 1$.

Put another way, $\hat{d}_\pi$ describes an exponentially weighted sum of k-step deviations from the behavior policy's stationary distribution d_μ, where the k^{th} deviation corresponds to applying transition $P_\pi^\top$ k times.

Lemma 2. *For $\hat{\gamma} < 1$, the ratio $\frac{\hat{d}_\pi}{d_\mu}$ is the unique fixed point of the operator $Y_{\hat{\gamma}}$.*

Theorem 3. *Let $c^0 \in \Delta$. For $\hat{\gamma} < 1$ the process $c^{k+1} := Y_{\hat{\gamma}} c^k$ converges to $\frac{\hat{d}_\pi}{d_\mu}$, where $\hat{d}_\pi$ is the stationary distribution of the transition function $\hat{P}_\pi$ corresponding to the given $\hat{\gamma}$.*

One of the most appealing properties of the discounted operator (for $\hat{\gamma} < 1$) is that it neither requires normalization, or even positive initial values to guarantee convergence. As we shall see, this greatly simplifies the learning process.

Discounted COP with Linear Function Approximation

The appeal of the COP-TD learning rule is that it can be applied online. The same remains true for our discounted COP learning rule (6). Naturally, when combined with function approximation the same issue of norm arises: can our learning process itself be guaranteed to converge? The answer is yes, provided the discount factor is taken to be small enough.

To begin, let us assume sample transitions are drawn as $s \sim d_\mu, a \sim \mu(\cdot \mid s), s' \sim P(\cdot \mid s, a)$, as before. Because d_μ is the stationary distribution, $s' \sim d_\mu$ also. The process we study is therefore described by the projected discounted COP operator $\Pi_{d_\mu} Y_{\hat{\gamma}}$.

Lemma 3. *The induced operator norm of the COP operator Y^n is upper bounded by a constant $\sqrt{K_{\pi,\mu,n}}$, in the sense that*

$$\|Y^n\|_{d_\mu}^2 \leq K_{\pi,\mu,n} := \sup_{s' \in \mathcal{S}} \sum_{s \in \mathcal{S}} \frac{d_\mu(s)}{d_\mu(s')} P_\pi^n(s' \mid s).$$

Further, the series can be bounded by a constant,

$$K_{\pi,\mu,n} \leq K_{\pi,\mu} := \left\| \frac{d_\mu(s)}{d_\pi(s)} \right\|_\infty \left\| \frac{d_\pi(s)}{d_\mu(s)} \right\|_\infty .$$

The term $K_{\pi,\mu,n}$ is a concentration coefficient similar to those studied by Munos (2003). Intuitively, it measures the discrepancy in stationary distributions between two states that are "close" according to π, in the sense that s' is reachable from s in n steps. When $\mu = \pi$, the sum simplifies to $d_\pi(s')$ and this term is 1.

We can make use of the concentration coefficient to provide a safe value of $\hat{\gamma}$ below which the discounted COP learning rule is convergent. Although most of our work concerns 1-step updates, we provide a slightly more general result on n-step methods here, based on known contraction results (Sutton and Barto 2018) and the existing multi-step extension of COP-TD (Hallak and Mannor 2017).

Theorem 4. *Consider the n-step discounted COP operator $Y_{\hat{\gamma}}^n$. Then for any $c \in \mathbb{R}^n$,*

$$\left\| Y_{\hat{\gamma}}^n c - \frac{\hat{d}_\pi}{d_\mu} \right\|_{d_\mu} \leq \hat{\gamma}^n \sqrt{K_{\pi,\mu,n}} \left\| c - \frac{\hat{d}_\pi}{d_\mu} \right\|_{d_\mu}$$

and in particular for $\hat{\gamma} < (K_{\pi,\mu,n})^{-1/2n}$, $Y_{\hat{\gamma}}^n$ is a contraction mapping. Since $K_{\pi,\mu,n}$ is a bounded series, the exponential factor is guaranteed to dominate. As a result, there exists a value of $\hat{\gamma} < 1$ for which the projected n-step discounted COP operator $\Pi_{d_\mu} Y_{\hat{\gamma}}^n$ is a contraction mapping.

Theorem 4 shows that we can avoid the usual divergence issues with the learning rule (6) by taking a sufficiently small $\hat{\gamma}$. While these results are not altogether surprising (they mirror the case of value function approximation), we emphasize that there is no equivalent guarantee in the undiscounted case.

More generally, we are unlikely to be in the worst-case scenario achieving the concentration coefficient $K_{\pi,\mu,n}$ and, as our empirical evaluation will show, divergence does not seem to be a problem even with large $\hat{\gamma}$. Yet, one may wonder whether it is relevant at all to learn an approximation to $\frac{d_\pi}{d_\mu}$. Using Theorem 1 we argue that since the bound is continuous in the learning distribution d we can expect improved performance even when the covariate shift is approximated for $\hat{\gamma} < 1$ or where a prediction error due to function approximation occurs.

Taken as a whole, our results suggest that incorporating the discount factor $\hat{\gamma}$ should improve the behavior of the COP-TD algorithm in practice.

Soft Ratio Normalization

Suppose we are given a function $c : \mathcal{S} \to \mathbb{R}$ differentiable w.r.t. its parameters for which we would like that

$$\sum_{s \in \mathcal{S}} d_\mu(s) c(s) = 1.$$

A common approach in deep reinforcement learning settings is to treat this as an additional loss to be minimized. In this section we also follow this approach, and consider minimizing the *normalization loss*

$$\mathcal{L}(c) := \tfrac{1}{2} \big(\sum_{s \in \mathcal{S}} d_\mu(s) c(s) - 1 \big)^2. \tag{7}$$

The gradient of this loss is

$$\nabla \mathcal{L}(c) = \big(\sum_{s \in \mathcal{S}} d_\mu(s) c(s) - 1 \big) \sum_{s \in \mathcal{S}} d_\mu(s) \nabla c(s). \tag{8}$$

We seek an unbiased estimate of this gradient. However, we cannot recover such an estimate with a single sample $s \sim d_\mu$, in a classic case of the double-sampling problem (Baird 1995). In particular, it is not hard to see that

$$\mathop{\mathbb{E}}_{s \sim d_\mu} [(c(s) - 1) \nabla c(s)] \neq \nabla \mathcal{L}(c).$$

However, we can obtain such an estimate by considering $m \geq 2$ samples $s_1, \ldots, s_m$ drawn from d_μ. The quantity

$$\hat{\nabla}(c) := \big(\tfrac{1}{m-1} \sum_{i=2}^{m} c(s_i) - 1 \big) \nabla c(s_1)$$

is an unbiased estimate of the loss gradient (8). In fact, as the following theorem states, we can do better by allowing each sample to play both roles in the estimate, and averaging the results.

Theorem 5. *Consider a differentiable function $c : \mathcal{S} \to \mathbb{R}$ and the loss function (7). Given $s_1, \ldots, s_m$ independent samples drawn from d_μ,*

$$\tfrac{1}{m} \sum_{i=1}^{m} \big(\tfrac{1}{m-1} \sum_{j \neq i} c(s_j) - 1 \big) \nabla c(s_i)$$

is an unbiased estimate of $\nabla \mathcal{L}(c)$.

In our experimental section we will see that the normalization loss plays an important role in making COP-TD practical.

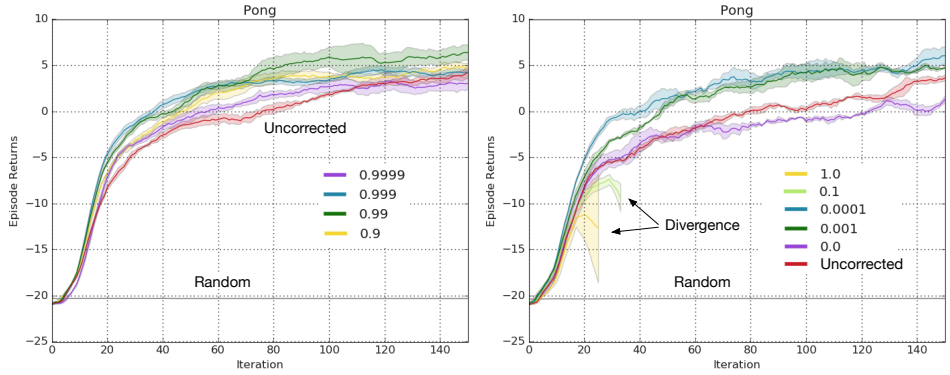

Figure 1: $\eta = 0.002$ with 5 seeds per run for 150 iterations. **Left.** Comparing discount factors in Pong. Using a discount factor gives a significant performance improvement. **Right.** Comparing normalization weights in Pong. Using normalization helps learning, but a large normalization weight causes divergence in the c values.

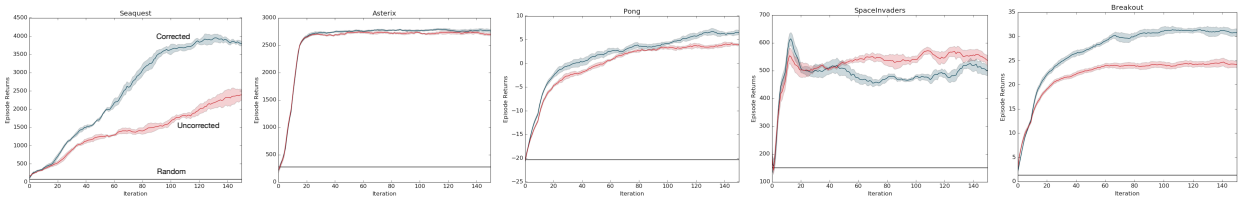

Figure 2: $\eta = 0.02$ with 3 seeds for 150 iterations. Performance of discounted COP-TD with a small target update period of 1000 and $\hat{\gamma} = 0.99$ on 5 Atari 2600 games.

Experimental Results

In this section we provide empirical evidence demonstrating that our method yields useful benefits in an off-policy, deep reinforcement learning setting. In our experiments we use the Arcade Learning Environment (ALE) (Bellemare et al. 2013), an RL interface to Atari 2600 games. We consider the single-GPU agent setup pioneered by Mnih et al. (2015). In this setup, the agent uses a replay memory (implemented as a windowed buffer) to store past experience, which it trains on continuously. As a result, much of the agent's learning carries an off-policy flavor.

We focus on a fixed behavior policy, specifically the uniformly random policy. We are interested in learning as good of a control policy as we can. That is, at each step the target policy is the greedy policy with respect to the predicted Q-values. While the theory we developed here applies to the policy evaluation case, we believe this setup to be a more practical and more stringent test of the idea. We emphasize that on the ALE, the uniformly random policy generates data that is significantly different from any learned policy; as a result, our experiments exhibit a high degree of off-policyness. To the best of our knowledge, we are the first to consider such a drastic setting.

Implementation

Our baseline is the C51 distributional reinforcement learning agent (Bellemare, Dabney, and Munos 2017), and we use published hyperparameters unless otherwise noted. We augment the C51 network by adding an extra head, the ratio model $c(s)$, to the final convolutional layer, whose role is to predict the ratio $\frac{d_\pi}{d_\mu}$. The ratio model consists of a two-layer fully-connected network with a ReLU hidden layer of size 512. Whenever a correction term is used as a sampling priority or to compute a bootstrapping target, we clip negative outputs to 0. In what follows $\bar{\theta}$ denotes the parameters of the target network, which includes the ratio model.

In initial experiments we found that multiplicatively reweighting the loss function using covariate shifts hurt performance, likely due to larger gradient variance. Instead, to reweight sample transitions we use a prioritized replay memory (Schaul et al. 2016) where priorities correspond to the approximate ratios of our model, which in expectation recovers the reweighting. These adjusted sampling priorities result in large portions of the dataset being mostly ignored (i.e. those unlikely under policy π); hence, the effective size of the data set is reduced and we risk overfitting. In our experiment we mitigated this effect by taking a larger replay memory size (10 million frames) than usual.

Identical to C51, the target policy $\pi_{\bar{\theta}}$ is the ϵ-greedy policy with respect to the expected value of the distribution output of the target network. We set $\epsilon = 0.1$. The ratio model is trained by adding the squared loss

$$\eta\left(\hat{\gamma} c_{\bar{\theta}}(s) \frac{\pi_{\bar{\theta}}(a|s)}{\mu(a|s)} + (1-\hat{\gamma}) - c_\theta(s')\right)^2 \qquad (9)$$

to the usual distributional loss of the agent, where $\eta > 0$

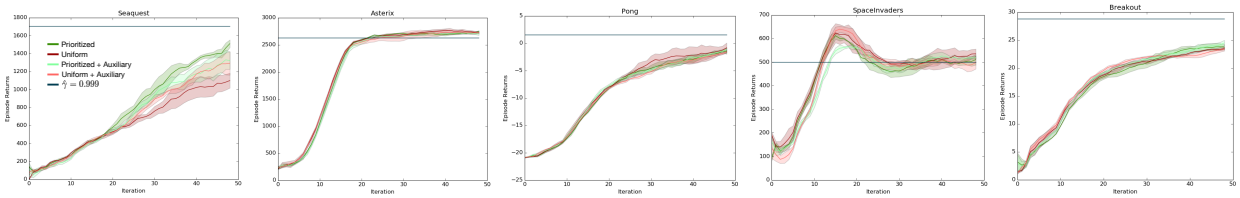

Figure 3: $\eta = 0.002$ with 3 seeds for 50 iterations. 4-way performance comparison using the discounted COP-TD loss as an auxiliary task and TD error prioritization as in (Schaul et al. 2016), blue line corresponds to the corrected agent with $\hat{\gamma} = 0.999$ at iteration 50.

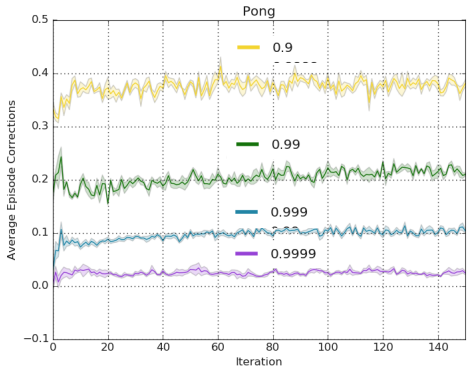

Figure 4: From same runs shown in Figure 1, left. Average predicted ratio in evaluation episode for a set of $\hat{\gamma}$ in the game of Pong.

is a hyperparameter trading off the two losses. In experiments where we also normalize the ratio model, a third loss (with corresponding weight hyperparameter) is also added. Preliminary experiments showed that learning the ratio with prioritized sampling led to stability issues, hence we train the ratio model by sampling transitions uniformly from the replay memory. Each training step samples two independent transition batches, prioritized and uniform for the value function and covariate shift respectively.

Since the training is done "backwards in time", no valid transition exists that would update the correction of an initial state s_0. This is similar to how there is no valid transition that updates the value of the terminal state in an episodic MDP. However, the distribution of any initial state s_0 is policy-independent, and so its ratio is 1. As a result, we modify the loss (9) for initial states by replacing the bootstrap target with 1. A more detailed analysis of our method in the episodic case is provided in the supplementary material.

Discounting and Normalization

We first study the effect of using the discounted COP update rule and/or normalization in the context of the game of Pong. In Pong, the random agent achieves an average score close to -21, the minimum (Bellemare et al. 2013). Figure 1, left, compares the learning curves of various values of the discount factor $\hat{\gamma}$, the agent with no corrections and the random baseline using a ratio loss weight $\eta = 0.02$. For discount factors not too large (all except $\hat{\gamma} = 0.9999$) better performance compared to the uncorrected baseline is achieved. Using normalization instead (Figure 1, right) also improves performance. However, for high values of the normalization weight, we observed unstable and sometimes divergent behavior. The runs which can be seen to stop mid plot had diverged in their c outputs (Figure **??**, appendix). We speculate that the reason for the divergence is higher variance in the loss function, and that a smaller step size might reduce such instabilities. Since using a discount factor proved more stable, has a slight performance advantage and is better understood theoretically than normalization, we will center the rest of the empirical evaluation around it.

In Figure 2 we report results for five Atari games chosen on the basis that a random agent playing these games explores the state space at least sufficiently to provide useful data for off-policy learning. We run C51 with discounted corrections with $\hat{\gamma} = 0.99$ and $\eta = 0.02$. We observe performance improvements in Seaquest, Breakout and Pong, no noticeable difference in Asterix and a small loss in Space Invaders.

Auxiliary tasks and Prioritization

One might wonder if the performance benefits observed are really due to sampling from a more on-policy distribution. Auxiliary tasks in the form of extra prediction losses have successfully been used to learn better representations and aid the learning of the value function (Jaderberg et al. 2016), (Aytar et al. 2018). To validate that the gains originate from correcting the off-policy data distribution as opposed to better representations, we show a modification of the previous experiment where the covariate shift was learned but not used. We also compare how our proposed prioritization scheme compares to the one originally proposed by (Schaul et al. 2016) who used a function of the TD error to set the priorities. A four-way comparison of auxiliary tasks and TD error prioritization is shown in Figure 3 where $\eta = 0.002$. We note that neither using the covariate shift prediction as an auxiliary task nor using TD error based prioritized sampling seemed to make any difference in all the games except SpaceInvaders. Interestingly, the covariate shift auxiliary task in SpaceInvaders helped when uniform sampling was used but hurt under prioritization.

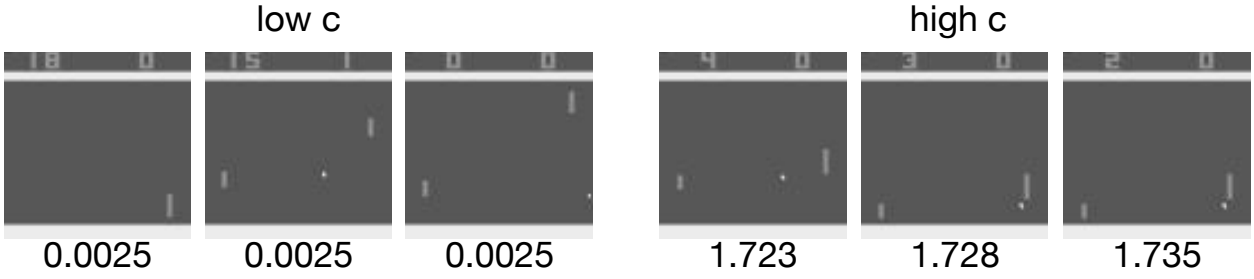

Figure 5: Sample states (frames) encountered under the random policy, predicted either as relatively less likely under π (low c) or relatively more likely under π (high c). The experiment clipped the corrections at 0.0025 which was later found to be unnecessary.

The Effect of the Discount Factor

To better understand the effect of $\hat{\gamma}$ in the learned ratios Figure 4 shows the average predicted ratio over evaluation episodes (where the ϵ-greedy policy is used instead of the uniform random policy) in the game of Pong for the same set of runs shown in Figure 1, left. Too large a discount ($\hat{\gamma} = 0.9999$) causes a decrease in performance, (see Figure 1, left) and one might expect that divergence of the ratios would be the cause. Surprisingly, we observe that $\hat{\gamma} = 0.9999$ show no signs of divergence. We emphasize is that the average episode ratio decays monotonically with $\hat{\gamma}$, hinting that there is a tendency for the ratios to collapse to 0 which overcomes any potential divergence issues.

Qualitative Evaluating the Learned Ratios

As an additional experiment, we qualitatively assessed the ratio $c(s)$ learned by our deep network. We generated 100,000 sample states by executing the random behavior policy on Pong. From these, we selected the top and bottom 50 states according to the ratio (c value) predicted by an agent trained under the regime of the previous section for 50 million frames. Recall that that $c > 1$ means the network believes the state is more likely under π than μ, while when $c < 1$ the converse is true.

Figure 5 shows the outcome of this experiment for the top 3 states in terms of c-value, and 3 low-c states; additional frames are provided in the supplemental. While our results remain qualitative, we see a clear trend in the selected images. States that are assigned low c correspond to those in which the opponent is about to score a point (2nd and 3rd images). The network also assigns a low c to a state in which the opponent has scored a high number of points (18 out of a possible total of 21) compared to the agent's (0 out of 21). This is indeed an unlikely state under π: if the trained agent ties the computer opponent, on average, then we expect its score to roughly match that of the opponent.

By contrast, states that are likely under c are those for in which the agent successfully returns the ball. These are naturally unlikely situations under μ, which plays randomly, but likely under the more successful policy π, which has learned to avoid the negative reward associated with failing to return the ball.

From this qualitative evidence we conclude that our model learns to clearly distinguish likely and unlikely sample transitions. We believe these results are particularly significant given the relative scarcity of off-policy methods of this kind in deep reinforcement learning.

Conclusion

In this paper we revisited Hallak and Mannor's COP-TD algorithm and extended it to be applicable to the deep reinforcement learning setting. While these results on the Atari 2600 suite of games remain preliminary, they demonstrate the practicality of learning the covariate shift in complex settings. We believe our results further open the door to increased sample efficiency in deep reinforcement learning.

We emphasize that the instabilities observed when learning the covariate shift under prioritized sampling point to the importance of the data distribution used to learn the ratios. Which distribution is optimal will be the focus of future work. The covariate shift method is a "backward" off-policy method, in the sense that it corrects a mismatch between distributions based on past transitions. It would be interesting to combine our method to "forward" off-policy methods such as Retrace (Munos et al. 2016), which have also yielded good results on the Atari 2600 (Gruslys et al. 2018). Then, it would be interesting to understand whether overfitting does occur due to a smaller effective replay size, and how this can be addressed. Finally, an exciting avenue would be extending the method to the more general case where multiple policies have generated off-policy data, which would allow COP-TD to be applied in the standard control setting.

Acknowledgements

We would like to thank Dale Schuurmans, James Martens, Ivo Danihelka, Danilo J. Rezende for insightful discussion. We also thank Jacob Buckman, Saurabh Kumar, Robert Dadashi and Nicolas Le Roux for reviewing and improving the draft.

References

Aytar, Y.; Pfaff, T.; Budden, D.; Paine, T. L.; Wang, Z.; and de Freitas, N. 2018. Playing hard exploration games by

watching Youtube. *CoRR* abs/1805.11592.

Baird, L. 1995. Residual algorithms: Reinforcement learning with function approximation. In *Proceedings of the twelfth international conference on machine learning (ICML 1995)*, 30–37.

Barnard, E. 1993. Temporal-difference methods and Markov models. *IEEE Transactions on Systems, Man, and Cybernetics*.

Bellemare, M. G.; Naddaf, Y.; Veness, J.; and Bowling, M. 2013. The Arcade Learning Environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research* 47:253–279.

Bellemare, M. G.; Dabney, W.; and Munos, R. 2017. A distributional perspective on reinforcement learning. In *Proceedings of the International Conference on Machine Learning*.

Bellman, R. E. 1957. *Dynamic programming*. Princeton, NJ: Princeton University Press.

Gruslys, A.; Dabney, W.; Azar, M. G.; Piot, B.; Bellemare, M.; and Munos, R. 2018. The reactor: A fast and sample-efficient actor-critic agent for reinforcement learning. In *International Conference on Learning Representations*.

Hallak, A., and Mannor, S. 2017. Consistent on-line off-policy evaluation.

Jaderberg, M.; Mnih, V.; Czarnecki, W.; Schaul, T.; Leibo, J. Z.; Silver, D.; and Kavukcuoglu, K. 2016. Reinforcement learning with unsupervised auxiliary tasks. *CoRR* abs/1611.05397.

Kolter, J. Z. 2011. The fixed points of off-policy TD. In *NIPS*.

Lin, L. 1993. Scaling up reinforcement learning for robot control. In *Machine Learning: Proceedings of the Tenth International Conference*, 182–189.

Maei, H. R.; Szepesvári, C.; Bhatnagar, S.; Precup, D.; Silver, D.; and Sutton, R. S. 2009. Convergent temporal-difference learning with arbitrary smooth function approximation. In *NIPS*.

Meyn, S. P., and Tweedie, R. L. 2012. *Markov chains and stochastic stability*.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518(7540):529–533.

Munos, R.; Stepleton, T.; Harutyunyan, A.; and Bellemare, M. G. 2016. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems*.

Munos, R. 2003. Error bounds for approximate policy iteration. In *Proceedings of the International Conference on Machine Learning*.

Precup, D.; Sutton, R. S.; and Singh, S. P. 2000. Eligibility traces for off-policy policy evaluation. In *Proceedings of the International Conference on Machine Learning*.

Puterman, M. L. 1994. *Markov Decision Processes: Discrete stochastic dynamic programming*. John Wiley & Sons, Inc.

Schaul, T.; Quan, J.; Antonoglou, I.; and Silver, D. 2016. Prioritized experience replay. In *International Conference on Learning Representations*.

Sutton, R. S., and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT Press, 2nd edition.

Sutton, R. S.; Maei, H. R.; Precup, D.; Bhatnagar, S.; Silver, D.; Szepesvári, C.; and Wiewiora, E. 2009. Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *ICML*.

Sutton, R. S.; Mahmood, A. R.; and White, M. 2016. An emphatic approach to the problem of off-policy temporal-difference learning. *Journal of Machine Learning Research*.

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

Tsitsiklis, J. N., and Van Roy, B. 1997. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control* 42(5):674–690.

van Hasselt, H.; Guez, A.; and Silver, D. 2016. Deep reinforcement learning with double Q-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Wang, T.; Lizotte, D.; Bowling, M.; and Schuurmans, D. 2008. Dual representations for dynamic programming. *Journal of Machine Learning Research* 1–29.

White, M. 2017. Unifying task specification in reinforcement learning.