Off-Policy Actor-Critic with Shared Experience Replay

Simon Schmitt [1] **Matteo Hessel** [1] **Karen Simonyan** [1]

Abstract

We investigate the combination of actor-critic reinforcement learning algorithms with a uniform large-scale experience replay and propose solutions for two ensuing challenges: (a) efficient actor-critic learning with experience replay (b) the stability of off-policy learning where agents learn from other agents behaviour. To this end we analyze the bias-variance tradeoffs in *V-trace*, a form of importance sampling for actor-critic methods. Based on our analysis, we then argue for mixing experience sampled from replay with on-policy experience, and propose a new trust region scheme that scales effectively to data distributions where V-trace becomes unstable. We provide extensive empirical validation of the proposed solutions on DMLab-30 and further show the benefits of this setup in two training regimes for Atari: (1) a single agent is trained up until 200M environment frames per game (2) a population of agents is trained up until 200M environment frames each and may share experience. We demonstrate state-of-the-art data efficiency among model-free agents in both regimes.

1. Introduction

Value-based and actor-critic policy gradient methods are the two leading model-free techniques of constructing general and scalable reinforcement learning agents (Sutton et al., 2018). Both have been combined with non-linear function approximation (Tesauro, 1995; Williams, 1992), and have achieved remarkable successes on multiple challenging domains; yet, these algorithms still require large amounts of data to determine good policies for any new environment. To improve data efficiency, experience replay agents store experience in a memory buffer (replay) (Lin, 1992), and reuse it multiple times to perform reinforcement learning updates (Riedmiller, 2005). Replay is often combined with the value-based Q-learning (Mnih et al., 2015), as it is an off-policy algorithm by construction, and can perform well even if the sampling distribution from replay is not aligned with the latest agent's policy. Combining experience replay with actor-critic algorithms can be harder due to their on-policy nature. Hence, most established actor-critic algorithms with replay such as (Wang et al., 2017; Gruslys et al., 2018; Haarnoja et al., 2018) employ and maintain Q-functions to learn from the replayed off-policy experience.

Table 1. Comparison of model-free state-of-the-art agents on 57 Atari games in the standard regime: Here *no experience* is shared between agents (differently from Figure 1 and Table 2). One agent is trained per game for up until 200M environment steps per game. We observe that our proposed LASER (LArge Scale Experience Replay) agent achieves a substantially higher score than all model-free prior art.

AGENTS (TRAINED IN STANDARD REGIME)	ATARI-57 MEDIAN AT 200M
LASER (OURS)	431%
META-GRADIENT (XU ET AL., 2018)	288%
RAINBOW (HESSEL ET AL., 2017)	223%
IQN (DABNEY ET AL., 2018)	218%
REACTOR (GRUSLYS ET AL., 2018)	187%
DQN (MNIH ET AL., 2013)	79%

In this paper, we demonstrate that off-policy actor-critic learning with experience replay can be achieved without surrogate Q-function approximators using V-trace by employing the following approaches:

- Off-policy replay experience needs to be mixed with a proportion of on-policy experience. We experimentally observe severe degradation in performance if this is missed (Figure 2) and theoretically that the V-trace policy gradient is otherwise not guaranteed to converge to a locally optimal solution.
- A trust region scheme (Conn et al., 2000; Schulman et al., 2015; 2017) can mitigate bias and enable efficient learning in a strongly off-policy regime, where distinct

[1]DeepMind. Correspondence to: Simon Schmitt <suschmitt@google.com>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

agents share experience through a commonly shared replay module. Sharing experience permits the agents to benefit from parallel exploration (Kretchmar, 2002) (Figures 1 and 3). Here we use a form of rejection sampling to reject transitions unsuitable for importance sampling.

Our paper is structured as follows: In Section 2 we revisit pure importance sampling for actor-critic agents (Degris et al., 2012) and V-trace, which is notable for allowing to trade off bias and variance in its estimates. We recall that variance reduction is necessary (Appendix, Figure 1 left) but is biased in V-trace. We derive proposition 2 stating that off-policy V-trace is not guaranteed to converge to a locally optimal solution – not even in an idealized scenario when provided with the optimal value function. Through theoretical analysis (Section 3) and experimental validation (Figure 2) we determine that mixing on-policy experience into experience replay alleviates the problem. Furthermore we propose a trust region scheme (Conn et al., 2000; Schulman et al., 2015; 2017) in Section 4 that enables efficient learning even in a strongly off-policy regime, where distinct agents share the experience replay module and learn from each others experience. We define the trust region in policy space and prove that the resulting estimator is correct (i.e. estimates the return more accurately).

As a result, we present state-of-the-art data efficiency in Section 5 in terms of median human normalized performance across 57 Atari games (Bellemare et al., 2013), as well as improved learning efficiency on DMLab30 (Beattie et al., 2016) (Tables 1 and 2).

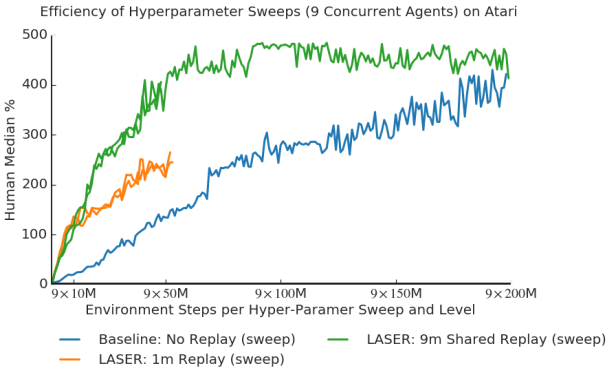

Figure 1. Sharing experience between agents leads to more efficient hyper-parameter sweeps on 57 Atari games. Note that the only previous agent "R2D2" that achieved a score beyond 400% required more than $1 \times 3,000$ (compared to our 9×60) million environment steps (see Kapturowski et al. (2019), page 14, Figure 9). We present the pointwise best agent from hyper-parameter sweeps with and without experience replay (shared and not shared). Each sweep contains 9 agents with different learning rate and entropy cost combinations. Replay experiment were repeated twice and ran for 50M steps. To report scores at 200M we ran the baseline and one shared experience replay agent for 200M steps.

2. The Issue with Importance Sampling: Bias and Variance in V-trace

V-trace importance sampling is a popular off-policy correction for actor-critic agents (Espeholt et al., 2018). In this section we revisit how V-trace controls the (potentially infinite) variance that arises from naive importance sampling. We note that this comes at the cost of a biased estimate (see Proposition 1) and creates a failure mode (see Proposition 2) which makes the policy gradient biased. We propose solutions for said issues in Sections 3 and 4.

2.1. Reinforcement Learning

We follow the notation of Sutton et al. (2018) where an *agent* interacts with its *environment*, to collect *rewards*. On each discrete time-step t, the agent selects an action a_t; it receives in return a reward r_t and an *observation* o_{t+1}, encoding a partial view of the environment's *state* s_{t+1}. In the fully observable case, the RL problem is formalized as a Markov Decision Process (Bellman, 1957): a tuple $(\mathcal{S}, \mathcal{A}, p, \gamma)$, where $\mathcal{S}, \mathcal{A}$ denotes finite sets of states and actions, p models rewards and state transitions (so that $r_t, s_{t+1} \sim p(s_t, a_t)$), and γ is a fixed discount factor. A *policy* is a mapping $\pi(a|s)$ from states to action probabilities. The agent seeks an *optimal* policy π^* that maximizes the *value*, defined as the expectation of the cumulative discounted *returns* $G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$.

Off-policy learning is the problem of finding, or evaluating, a policy π from data generated by a different policy μ. This arises in several settings. Experience replay (Lin, 1992) mixes data from multiple iterations of policy improvement. In large-scale distributed RL, decoupling acting from learning (Nair et al., 2015; Horgan et al., 2018; Espeholt et al., 2018) causes the experience to lag behind the latest agent policy. Finally, it is often useful to learn multiple general value functions (Sutton et al., 2011; Mankowitz et al., 2018; Lample & Chaplot, 2016; Mirowski et al., 2017; Jaderberg et al., 2017b) or *options* (Sutton et al., 1999; Bacon et al., 2017) from a single stream of experience.

2.2. Naive Importance Sampling

On-policy n-step bootstraps give more accurate value estimates in expectation with increasing n (Sutton et al., 2018). Unfortunately n must be chosen suitably as the estimates variance increases with n too. It is desirable to obtain benefits akin to n-step returns in the off-policy case. To this end multi-step importance sampling (Kahn, 1955) can be used. This however adds another source of (potentially infinite (Sutton et al., 2018)) variance to the estimate.

Table 2. Comparison of model-free agents that were trained with population training schemes; ranging from simple sweeps to population based training (Jaderberg et al., 2017a). We present our proposed LASER (LArge Scale Experience Replay) agent in comparison to state-of-the-art agents that were also trained for 9×200M (per Atari game) or 10×10B (jointly on all DMLab-30 games combined) environment steps. Results are obtained from Hessel et al. (2019). As a baseline we consider an implementation of a pixel control agent from Hessel et al. (2019). Extending our approach to the concurrent work by Song et al. (2020) remains for future work.

Agents (trained jointly in populations)	Atari Median	DMLab-30 Median	DMLab-30 Mean-Capped
PopArt-IMPALA+PixelControl (baseline)	-	85.5%	77.6%
LASER: Experience Replay	233% at 9×50M	95.4%	79.6%
LASER: Shared Experience Replay	370% at 9×50M 448% at 9×200M	97.2%	81.7%

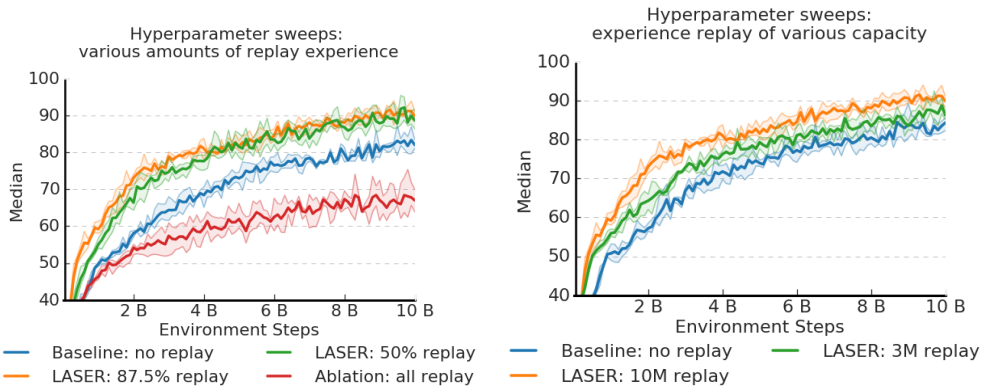

Figure 2. **Left**: Learning entirely off-policy from experience replay fails (red), while combining on-policy data with experience replay leads to improved data efficiency: We present sweeps on DMLab-30 with experience replays of 10M capacity. A ratio of 87.5% implies that there are 7 replayed transitions in the batch for each online transition. Furthermore we consider an agent identical to "LASER 87.5% replay" which however draws all samples from replay. Its batch thus does not contain any online data and we observe a significant performance decrease (see Proposition 2 and 3). The shading represents the point-wise best and worst replica among 3 repetitions. The solid line is the mean. **Right**: The effect of capacity in experience replay with 87.5% replay data per batch on sweeps on DMLab-30. Data-efficiency improves with larger capacity.

Importance sampling can estimate the expected return V^π from trajectories sampled from $\mu \neq \pi$, as long as μ is non-zero whereever π is. We employ a previously estimated value function V as a bootstrap to estimate expected returns. Following (Degris et al., 2012), a multi-step formulation of the expected return is

$$V^\pi(s_t) = \mathbf{E}_\mu \left[V(s_t) + \sum_{k=0}^{K-1} \gamma^k \Big(\prod_{i=0}^{k} \frac{\pi_{t+i}}{\mu_{t+i}} \Big) \delta_{t+k} V \right] \quad (1)$$

where $\mathbf{E}_\mu$ denotes the expectation under policy μ up to an episode termination, $\delta_t V = r_t + \gamma V(s_{t+1}) - V(s_t)$ is the temporal difference error in consecutive states s_{t+1}, s_t, and $\pi_t = \pi_t(a_t|s_t)$. Importance sampling estimates can have high variance. Tree Backup (Precup et al., 2000), and Q(λ) (Sutton et al., 2014) address this, but reduce the number of steps before bootstrapping even when this is undesirable (as in the on-policy case). RETRACE (Munos et al., 2016) makes use of full returns in the on-policy case, but it introduces a zero-mean random variable at each step, adding variance to empirical estimates in both on- and off-policy cases.

2.3. Bias-Variance Analysis & Failure Mode of V-trace Importance Sampling

V-trace (Espeholt et al., 2018) reduces the variance of importance sampling by trading off variance for a biased estimate of the return – resulting in a failure mode (see Proposition 2). It uses clipped importance sampling ratios to approximate V^π by

$$V^{\tilde{\pi}}(s_t) = V(s_t) + \sum_{k=0}^{K-1} \gamma^k \Big(\prod_{i=0}^{k-1} c_i \Big) \rho_t \delta_{t+k} V$$

where V is a learned state value estimate used to bootstrap, and $\rho_t = \min[\pi_t/\mu_t, \bar{\rho}]$, $c_t = \min[\pi_t/\mu_t, \bar{c}]$ are the clipped importance ratios. Note that, differently from RETRACE, V-trace fully recovers the Monte Carlo return when on policy. It similarly reweights the policy gradi-

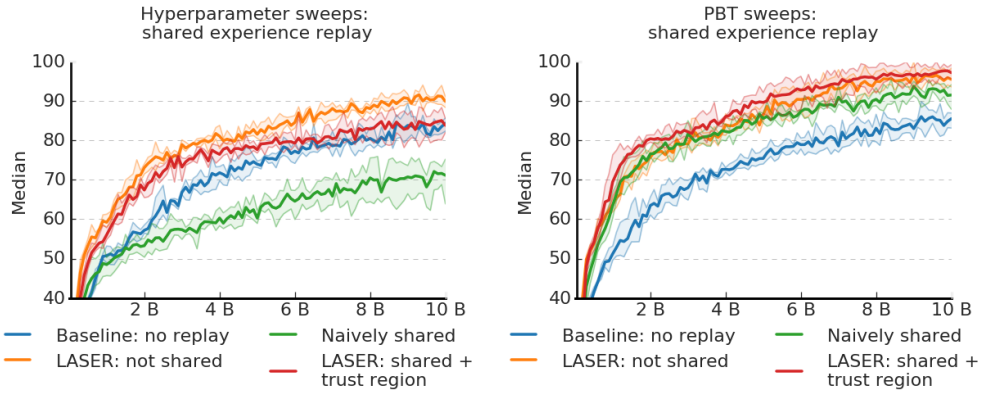

Figure 3. **Left:** Naively sharing experience between distinct agents in a hyper-parameter sweep fails (green) and is worse than the no-replay baseline (blue). The proposed trust region estimator mitigates the issue (red). **Right:** Combining population based training with trust region estimation improves performance further. All replay experiments use a capacity of 10 million observations and 87.5% replay data per batch.

ent as: $\nabla V^{\tilde{\pi}}(s_t) \overset{\text{def}}{=} \mathbf{E}_{\mu}\left[\rho_t \nabla(\log \pi_t)(r_t + \gamma V^{\tilde{\pi}}(s_{t+1}))\right]$ Note that $\nabla V^{\tilde{\pi}}(s_t)$ recovers the naively importance sampled policy gradient for $\bar{\rho} \to \infty$. In the literature, it is common to subtract a baseline from the action-value estimate $r_t + \gamma V^{\tilde{\pi}}(s_{t+1})$ to reduce variance (Williams, 1992), omitted here for simplicity. The constants $\bar{\rho} \geq \bar{c} \geq 1$ (typically chosen $\bar{\rho} = \bar{c} = 1$) define the level of clipping, and improve stability by ensuring a bounded variance. For any given $\bar{\rho}$, the bias introduced by V-trace in the value and policy gradient estimates increases with the difference between π and μ. We analyze this in the following propositions.

Proposition 1. *The V-trace value estimate $V^{\tilde{\pi}}$ is biased: It does not match the expected return of π but the return of a related* implied policy $\tilde{\pi}$ *defined by equation 2 that depends on the behaviour policy μ:*

$$\tilde{\pi}_{\mu}(a|x) = \frac{\min\left[\bar{\rho}\mu(a|x), \pi(a|x)\right]}{\sum_{b \in A} \min\left[\bar{\rho}\mu(b|x), \pi(b|x)\right]} \tag{2}$$

Proof. See Espeholt et al. (2018). □

Note that the biased policy $\tilde{\pi}_{\mu}$ can be very different from π. Hence the V-trace value estimate $V^{\tilde{\pi}}$ may be very different from V^{π} as well. As an illustrative example, consider two policies over a set of two actions, e.g. "left" and "right" represented as a tuple of probabilities. Let us investigate $\mu = (\phi, 1-\phi)$ and $\pi = (1-\phi, \phi)$ defined for any suitably small $\phi \leq 1$. Observe that π and μ share no trajectories (state-action sequences) in the limit as $\phi \to 0$ and they get more focused on one action. A practical example of this could be two policies, one almost always taking a left turn and one always taking the right. Given sufficient data of either policy it is possible to estimate the value of the other e.g. with naive importance sampling. However observe that V-trace with $\bar{\rho} = 1$ will always estimate a biased value - even given infinite data. Observe that $\min\left[\mu(a|x), \pi(a|x)\right] = \min\left[\phi, 1-\phi\right]$ for both actions. Thus $\tilde{\pi}_{\mu}$ is uniform rather than resembling π the policy. The V-trace estimate $V^{\tilde{\pi}}$ would thus compute the average value of "left" and "right" – poorly representing the true V^{π}.

Proposition 2. *The V-trace policy gradient is biased: given the the optimal value function V^* the V-trace policy gradient does not converge to a locally optimal π^* for all off-policy behaviour distributions μ.*

Proof. See Appendix Section 3.

3. Mixing On- and Off-Policy Experience (Mitigation I)

In Proposition 2 we presented a failure mode in V-trace where the variance reduction biases the value estimate and policy gradient. In this section we show that mixing replay data with a suitable α-fraction of on-policy data can prevent the otherwise severe degradation in performance (see Figure 2).

3.1. The Problem

Observe that V-trace computes biased Q-estimates $Q^{\omega} \neq Q$ resulting in a wrong local policy gradient: $\nabla \mathbf{E}_{\pi(a|s)}\left[Q^{\omega}(s,a)\right] \neq \nabla \mathbf{E}_{\pi(a|s)}\left[Q(s,a)\right]$. In equation 4 in the appendix we show that $Q^{\omega}(s,a) = Q(s,a)\omega(s,a)$ where $\omega(s,a) = \min\left[1, \bar{\rho}\frac{\mu(a|s)}{\pi(a|s)}\right] \leq 1$.

The question of how biased the resulting policy will be depends on whether the distortion changes the argmax of the Q-function. Little distortions that do not change the argmax will result in the same local fixpoint of the policy improvement. The policy will continue to select the optimal

action and it will not be biased at this state. The policy will however be biased if the Q-function is distorted too much. For example considering an optimal Q and $\omega(s,a)$ that swaps the argmax for the 2nd largest value, the regret will then be the difference between the maximum and the 2nd largest value. Intuitively speaking the more distorted the Q^ω, the larger will be the regret compared to the optimal policy.

More precisely, the regret of learning a policy that maximizes the distorted Q^ω at state s is: $R(s) = Q(s,a^*) - Q(s,a_{\text{actual}}) = \max_b Q(s,b) - Q(s,a_{\text{actual}})$ where $a^* = \text{argmax}_b(Q,b)$ is the optimal action according to the real Q and $a_{\text{actual}} = \text{argmax}[Q^\omega(s,a)] = \text{argmax}[Q(s,a)\omega(s,a)]$, is the optimal action according to the distorted Q^ω. For generality, we denote A^* as the set of best actions - covering the case with multiple with identical optimal Q-values.

3.2. Mitigating the Problem

Proposition 3 provides a mitigation: Clearly the V-trace policy gradient will converge to the same solution as the true on-policy gradient if the argmax of the Q-function is preserved at all states in a tabular setting. We show that this can be achieved by mixing a sufficient proportion α of on-policy experience into the computation.

We show in equation 7 in the appendix that choosing α such that

$$\frac{\alpha}{1-\alpha} > \max_{b \notin A^*} \left[\frac{Q^\omega(s,b) - Q^\omega(s,a^*)}{Q(s,a^*) - Q(s,b)} \right] \frac{d^\mu(s)}{d^\pi(s)}$$

for $Q^\omega(s,a) = Q(s,a)\omega(s,a)$ will result in a policy that correctly chooses the best action at state s. Note that $\frac{\alpha}{1-\alpha} \to \infty$ as $\alpha \to 1$.

Intuitively: the larger the action value gap of the real Q-function $Q(s,a^*) - Q(s,b)$ the lower the right hand side and the less on-policy data is required. If $\max_b[(Q(s,b)\omega(s,b) - Q(s,a^*)\omega(s,a^*)]$ is negative, then α may be as small as zero hence enabling even pure off-policy learning. Finally note that the right hand side decreases due to $d^\mu(s)/d^\pi(s)$ if π visits the state s more often than μ. All of those conditions can be computed and checked if an accurate Q-function and state distribution is accessible. How to use imperfect Q-function estimates to adaptively choose such an α remain a question for future research.

We provide experimental evidence for these results with function approximators in the 3-dimensional simulated environment DMLab-30 with various $\alpha \geq 1/8$ in Section 5.3 and Figure 2. We observe that $\alpha = 1/8$ is sufficient to facilitate stable learning. Furthermore it results in better data-efficiency than pure on-policy learning as it utilizes off-policy replay experience.

Proposition 3. *Mixing on-policy data into the V-trace policy gradient with the ratio α reduces the bias by providing a regularization to the implied state-action values. In the general function approximation case it changes the off-policy V-trace policy gradient from $\sum_s d^\mu(s)\mathbf{E}_\pi[(Q(s,a)\nabla \log \pi(a|s)]$ to $\sum_s \mathbf{E}_\pi[Q^\alpha(s,a)\nabla \log \pi(a|s)]$ where $Q^\alpha = Qd^\pi(s)\alpha + Q^\omega d^\mu(s)(1-\alpha)$ is a regularized state-action estimate and d^π, d^μ are the state distributions for π and μ. Note that there exists $\alpha \leq 1$ such that Q^α has the same argmax (i.e. best action) as Q.*

Proof. See Appendix Section 3.

Mixing online data with replay data has also been argued for by (Zhang & Sutton, 2017), as a heuristic way of reducing the sensitivity of reinforcement learning algorithms to the size of the replay memory. Proposition 3 grounds this in the theoretical properties of V-trace.

4. Trust Region Scheme for Off-Policy V-trace

Recall that in off-policy learning we have the choice between naive importance sampling (high-variance), V-trace (limited variance + bias) and no-correction (large bias). In this section we introduce an approach to limit the bias in V-trace importance sampling by rejecting high-bias transitions. In Section 5.4 and Figure 3 we apply this to off-policy experience from concurrently learning agents, thus enriching the agents replay with relevant (low variance + low bias) transitions from other agents behaviour.

To this end we introduce a *behaviour relevance function* that classifies behaviour as relevant. We then define a trust-region estimator that computes expectations (such as expected returns, or the policy gradient) only on relevant transitions. In proposition 4 and 5 we show that this trust region estimator indeed computes new state-value estimates that improve over the current value function. While our analysis and proof is general we propose a suitable behaviour relevance function in Section 4.3 that employs the Kullback Leibler divergence between target policy π and implied policy $\tilde{\pi}_\mu$: $\text{KL}(\pi(\cdot|s)||\tilde{\pi}_\mu(\cdot|s))$. We provide experimental validation in Figure 3.

4.1. Behaviour Relevance Functions

In off-policy learning we often consider a family of behaviour policies either indexed by training iteration t: $M_T = \{\mu_t | t < T\}$ for experience replay, or by a different agent k: $M_K = \{\mu_k | k \in K\}$ when training multiple agents. In the classic experience replay case we then sample a time t and locate the transition τ that was generated earlier via μ_t. This extends naturally to the multiple agent case where we sample an agent index k and then obtain a transition for such agent or tuples of (k,t). Without loss of

generality we simplify this notation and index sampled behaviour policies by a random variable $z \sim Z$ that represents the selection process. While online reinforcement learning algorithms process transitions $\tau \sim \pi$, off-policy algorithms process $\tau \sim \mu_z$ for $z \sim Z$. In this notation, given equation (1) and a bootstrap V, the expectation of importance sampled off-policy returns at state s_t is described by:

$$V^\pi_{\text{mix}}(s_t) = \mathbf{E}_z\Big[\mathbf{E}_{\mu_z|z}\big[G^{\pi,\mu_z}(s_t)\big]\Big] \tag{3}$$

where $G^{\pi,\mu}(s_t) = V(s_t) + \sum_{k=0}^{\infty}\gamma^k\Big(\prod_{i=0}^{k}\frac{\pi_{t+i}}{\mu_{t+i}}\Big)\delta_{t+k}V$ is a single importance sampled return. Note that the on-policy return $G^{\pi,\pi}(s_t) = V(s_t) + \sum_{k=0}^{\infty}\gamma^k r_{t+k}$.

Above $\mathbf{E}_{\mu_z|z}$ represents the expectation of sampling from a given μ_z. The conditioning on z is a notational reminder that this expectation does not sample z or μ_z but experience from μ_z. For any sampled z we obtain a μ_z and observe that the inner expectation wrt. experience of μ_z in equation (3) recovers the expected on-policy return in expectation:

$$\begin{aligned}
&\mathbf{E}_{\mu_z|z}\left[G^{\pi,\mu_z}(s_t)\right] \\
&= \mathbf{E}_{\mu_z|z}\left[V(s_t) + \sum_{k=0}^{\infty}\gamma^k\Big(\prod_{i=0}^{k}\frac{\pi_{t+i}}{\mu_{z,t+i}}\Big)\delta_{t+k}V\right] \\
&= \mathbf{E}_{\pi}\left[V(s_t) + \sum_{k=0}^{\infty}\gamma^k\Big(\prod_{i=0}^{k}\frac{\mu_{z,t+i}}{\mu_{z,t+i}}\Big)\delta_{t+k}V\right] \\
&= \mathbf{E}_{\pi}\left[V(s_t) + \sum_{k=0}^{\infty}\gamma^k r_{t+k}\right] = \mathbf{E}_{\pi}\left[G^{\pi,\pi}(s_t)\right] \\
&= V^\pi(s_t)
\end{aligned}$$

Thus $V^\pi_{\text{mix}}(s_t) = \mathbf{E}_z\left[\mathbf{E}_\pi\left[G^{\pi,\pi}(s_t)\right]\right] = \mathbf{E}_\pi\left[G^{\pi,\pi}(s_t)\right] = V^\pi(s_t)$. This holds provided that μ_z is non-zero wherever π is. This fairly standard assumption leads us straight to the core of the problem: it may be that some behaviours μ_z are ill-suited for estimating the inner expectation. However, standard importance sampling applied to very off-policy experience divides by small μ resulting in high or even infinite variance. Similarly, V-trace attempts to compute an estimate of the return following π resulting in limited variance at the cost of a biased estimate in turn.

The key idea of our proposed solution is to compute the return estimate for π at each state only from a subset of suitable behaviours μ_z: $M_{\beta,\pi}(s) = \{\mu_z | z \in Z \text{ and } \beta(\pi,\mu,s) < b\}$ as determined by a *behaviour relevance function*

$$\beta(\pi,\mu,s) : (M_Z, M_Z, S) \to \mathbb{R}$$

and a threshold b. The behaviour relevance function decides if experience from a behaviour is suitable to compute an expected return for π. It can be chosen to control properties of V^π_{mix} by restricting the expectation on subsets of Z. In particular it can be used to control the variance of an importance sampled estimator: Observe that the inner expectation $\mathbf{E}_{\mu_z}\left[G^{\pi,\mu}(s_t)\middle|z\right]$ in equation (3) already matches the expected return V^π. Thus we can condition the expectation on arbitrary subsets of Z without changing the expected value of V^π_{mix}. This allows us to reject high variance $G^{\pi,\mu}$ without introducing a bias in V^π_{mix}. The same technique can be applied to V-trace where we can reject return estimates with high bias.

4.2. Derivation of Trust Region Estimators

Using a behaviour relevance function $\beta(s)$ we can define a trust region estimator for regular importance sampling (IS) and V-trace and show their correctness.

We define the trust region estimator as the conditional expectation

$$V^\pi_{\text{trusted}}(s_t) = \mathbf{E}_z\Big[\mathbf{E}_{\mu_z|z}\big[G^{\pi,\mu_z,\beta}(s_t)\big]\Big|\mu_z \in M_{\beta,\pi}(s_t)\Big]$$

with λ-returns G, chosen as $G^{\pi,\mu_z}_{\text{IS}}(s_t) = V(s_t) + \sum_{k=0}^{\infty}\gamma^k\Big(\prod_{i=0}^{k}\lambda_{\pi,\mu_z}(s_{t+i})\frac{\pi_{t+i}}{\mu_{z,t+i}}\Big)\delta_{t+k}V$ for importance sampling and for V-trace as

$$\begin{aligned}
G^{\pi,\mu_z}_{\text{Vtrace}}(s_t) =& V(s_t) + \sum_{k=0}^{\infty}\gamma^k\Big(\prod_{i=0}^{k-1}\lambda_{\pi,\mu_z}(s_{t+i})c_{z,t+i}\Big) \\
&\lambda_{\pi,\mu_z}(s_{t+k})\rho_{z,t+k}\delta_{t+k}V
\end{aligned}$$

where $\lambda_{\pi,\mu}(s_t)$ is designed to constraint Monte-Carlo bootstraps to relevant behaviour: $\lambda_{\pi,\mu}(s_t) = \mathbb{1}_{\beta(\pi,\mu,s_t)<b}$ and $\rho_{z,t+k} = \min\left[\frac{\pi_{t+i}}{\mu_{z,t+i}}, \bar{\rho}\right]$ and $c_{z,t+k}$ are behaviour dependent clipped importance rations. Thus both $G^{\pi,\mu_z}_{\text{IS}}$ and $G^{\pi,\mu_z}_{\text{Vtrace}}$ are a multi-step return estimators with adaptive length. Note that only estimators with length ≥ 1 are used in V^π_{trusted}. Due to Minkowski's inequality the trust region estimator thus shows at least the same contraction as a 1-step bootstrap, but can be faster due to its adaptive nature:

Proposition 4. *Let $G^{\pi,\mu_z}_{\text{IS}}$ be a set of importance sampling estimators as defined in section 4.2. Note that they all have the same fix point V^π and contract with at least γ. Then the contraction properties carry over to V^π_{trusted}. In particular $\left|V^\pi_{\text{trusted}} - V^\pi\right|_\infty \leq \gamma\left|V - V^\pi\right|_\infty$.*

Proof. See Appendix Section 3.

Proposition 5. *Let $G^{\pi,\mu_z}_{\text{Vtrace}}$ be a set of V-trace estimators (see section 4.2) with corresponding fixed points V^z (see equation 2) to which they contract at a speed of an algorithm and behaviour specific η_z. Then V^π_{trusted} moves towards $V^\beta = \mathbf{E}_{z|\mu_z\in M_{\beta,\pi}(s_t)}\left[V^z\right]$ shrinking the distance as follows $\left|V^\pi_{\text{trusted}} - V^\beta\right|_\infty < \max_{\mu_z\in M_{\beta,\pi}(s_t)}\left|\eta_z(V - V^z)\right|_\infty \leq$*

$\eta_{\max} \max_{\mu_z \in M_{\beta,\pi}(s_t)} |(V - V^z)|_\infty$ *with* $\eta_{\max} = \max_{\mu_z \in M_{\beta,\pi}(s_t)} \eta_z$.

Proof. See Appendix Section 3.

Note how the choice of β and thus $M_{\beta,\pi}$ enables us to discard ill-suited $G^{\pi,\mu_z}_{\text{Vtrace}}$ from the estimation of V^π_{trusted}. Recall that V-trace fixed points V_z are biased. Thus β allows us to selectively create the V-trace target $V^\beta = \mathbf{E}_{z|\mu_z \in M_{\beta,\pi}(s_t)}[V^z]$ and control its bias and the shrinkage $\max_{\mu_z \in M_{\beta,\pi}(s_t)} |\eta_z(V(s) - V^z(s))|_\infty$ (see Proposition 5). Similarly it can control cases where we can not use the exact importance sampled estimator.

4.3. Implementation Details

In Proposition 5 we have seen that the quality of the trust region V-trace return estimator depends on β. A suitable choice of β can move the return estimate V^β closer to V^π and improve the shrinkage by reducing $\max_{\mu_z \in M_{\beta,\pi}(s_t)} |\eta_z(V(s) - V^z(s))|_\infty$. Hence, we employ a behaviour relevance function β_{KL} that rejects high bias transitions by estimating the Kulback-Leibler divergence between the target policy π and the implied policy $\tilde{\pi}_{\mu_z}$ for a sampled behaviour μ_z. Recall from Proposition 1 that $\tilde{\pi}_{\mu_z}$ determines the fixed point of the V-trace estimator for behaviour μ_z and thus determines the bias in V^z.

$$\beta_{\text{KL}}(\pi, \mu, s) = \text{KL}\left(\pi(\cdot|s)||\tilde{\pi}_\mu(\cdot|s)\right)$$

Note that the behaviour probabilities μ_z can be evaluated and saved to the replay when the agent executes the behaviour, similarly the target policy π is represented by the agents neural network. Using both and equation 2, $\tilde{\pi}_\mu$ can be computed. For large or infinite action spaces a Monte Carlo estimate of the KL divergence can be computed.

For simplicity we use the same behaviour relevance functions for the policy and value estimate - while those could theoretically be different. As described above we stop the Monte-Carlo bootstraps at undesirable state-behaviour pairs. This censoring procedure is computed from state dependent $\beta(\pi, \mu, s)$ and ensures that the choice of bootstrapping does not depend on the sampled actions. Rejection by an action-based criteria such as small $\pi(a|s)/\mu(a|s)$ would introduce an additional bias which we avoid by choosing β_{KL}.

5. Experiments

We present experiments to support the following claims:

- Our proposed algorithm obtains state-of-the-art performance on Atari both in the shared regime (see Figure 1) and in the classic single agent regime with 200M environment steps per game (see Table 1).
- Section 5.2: Uniform experience replay obtains comparable results as prioritized experience replay, while being simpler to implement and tune.
- Section 5.3: Using fresh experience before inserting it in experience replay is strictly required as learning purely off-policy from experience replay leads to severe degradation in performance (see Figure 3 left) – in line with Proposition 3.
- Section 5.4: Sharing experience without trust region performs poorly as suggested by Proposition 2. Off-Policy Trust-Region V-trace solves this issue.
- Section 5.5: Sharing experience can take advantage of parallel exploration, while also saving memory through sharing a single experience replay.

5.1. Experimental Setup & Methodology

We use the V-trace distributed reinforcement learning agent (Espeholt et al., 2018) as our baseline. In our experiments we consider two experimental platforms: Atari and DeepMind Lab. On Atari we consider the common single task training regime, where a different agent is trained, from scratch, on each of the tasks. Following (Xu et al., 2018) we use a discount of 0.995. Motivated by recent work by (Kaiser et al., 2019), we use the IMPALA deep network and increased the number of channels $4\times$. We use 96% replay data per batch. Differently from (Espeholt et al., 2018), we do not use gradient clipping by norm (Pascanu et al., 2012). Updates are computed on mini-batches of 32 (regular) and 128 (replay) trajectories, each corresponding to 19 steps in the environment. In the context of DeepMind Lab, we consider the multi-task suite DMLab-30 (Espeholt et al., 2018), as the visuals and the dynamics are more consistent across tasks. Furthermore the multi-task regime is particularly suitable for the investigation of strongly off-policy data distributions arising from sharing the replay across agents, as concurrently learning agents can easily be stuck in different policy plateaus, generating substantially different data (Schaul et al., 2019). As in (Espeholt et al., 2018), in the multi-task setting each agent trains simultaneously on a uniform mixture of all tasks rather than individually on each game. The score of an agent is thus the median across all 30 tasks. Following (Hessel et al., 2019), we augment our agent with multi-task Pop-Art normalization and PixelControl. We use a PreCo LSTM (Amos et al., 2018) instead of the vanilla one (Hochreiter & Schmidhuber, 1997). Updates are computed on mini-batches of multiple trajectories chosen as above, each corresponding to 79 steps in the environment. In early experiments we found that computing the entropy cost only on the online data provided slightly better results, hence we have done so throughout our experiments.

In all our experiments, experience sampled from memory

is mixed with online data within each mini-batch – following Proposition 3. Episodes are removed in a first in first out order, so that replay always holds the most recent experience. Unless explicitly stated otherwise we consider hyper-parameter sweeps, some of which share experience via replay. In this setting multiple agents start from-scratch, run concurrently at identical speed, and add their new experience into a common replay buffer. All agents will then draw uniform samples from the replay buffer. On DMLab-30 we consider both regular hyper-parameter sweeps and sweeps with population based training (PBT) (Jaderberg et al., 2017a). Sweeps contain 10 agents with hyper-parameters sampled similar as Espeholt et al. (2018) but fixed RMSProp $\epsilon = 0.1$. On Atari sweeps contain 9 agents with different constant learning rate and entropy cost combinations $\{3{\cdot}10^{-4}, 6{\cdot}10^{-4}, 1.2{\cdot}10^{-3}\} \times \{5{\cdot}10^{-3}, 1{\cdot}10^{-2}, 2{\cdot}10^{-2}\}$ (distributed by factors $\{1/2, 1, 2\}$ around the initial parameters reported in Espeholt et al. (2018)). Although our focus is on efficient hyper-parameter sweeps given crude initial parameters, we also present a single-agent LASER experiment using the same tuned schedule as Espeholt et al. (2018), a 87.5% replay ratio and a 15M replay. We store entire episodes in the replay buffer and replay each episode from the beginning, using the most recent network parameters to recompute the LSTM states along the way: this is critical when sharing experience between different agents, which may have arbitrarily different state representations.

5.2. Uniform and Prioritized Experience Replay

Prioritized experience replay has the potential to provide more efficient learning compared to uniform experience replay (Schaul et al., 2015; Horgan et al., 2018). However, it also introduces a number of new hyper-parameters and design choices: the most critical are the priority metric. Uniform replay is instead almost parameter-free and can be easily shared between multiple agents. Experiments provided in Figure 1 in the appendix showed little benefit of actor critic prioritized replay on DMLab-30. Furthermore priorities are typically computed from the agent specific metrics such as the TD-error, which are ill-defined when replay is shared among multiple agents. Hence we used uniform replay for our further investigations.

5.3. Mixing On- and Off-policy Experience

Figure 2 (left) shows that performance degrades severely when online data is not present in the batch. This experimentally validates Propositions 2 and 3 that highlight difficulties of learning purely off-policy. Furthermore Figure 2 (right) shows that best results are obtained with experience replay of 10M capacity and 87.5% ratio. A ratio of $87.5\% = 7/8$ corresponds to 7 replay samples for each online sample. We have considered ratios of $1/2, 3/4$, and $7/8$ and observed stable training for all of them. Observe that among those values, larger ratios are more data-efficient as they take advantage of more replayed experience per training step.

5.4. Shared Experience Replay with Off-Policy Trust Region V-trace

In line with proposition 2 we observe in Figure 3 (left) that hyper-parameter sweeps without trust-region are even surpassed by the baseline without experience replay. State-of-the-art results are obtained in Figure 3 (right) when experience is shared with trust-region in a PBT sweep. Observe that this indicates parallel exploration benefits and saves memory at the same time: in our sweep of 10 replay agents the difference between $10 \times 10\text{M}$ (separate replays) and 10M (shared replay) is 10-fold. This effect would be even more pronounced with larger sweeps. As discussed in section 2.3, the bias in V-trace occurs due to the clipping of importance ratios. A potential solution of reducing the bias would be to increase the $\bar{\rho}$ threshold to clip less aggressively and accept increased variance. Figure 1 in the appendix shows that this is not a solution.

5.5. Evaluation on Atari

We apply our proposed agent to Atari which has been a long established suite to evaluate reinforcement learning algorithms (Bellemare et al., 2013).

In Figure 1 we investigate the benefits of sharing experience replay within a hyper-parameter sweep on Atari. A sweep of 9 LASER agents sharing their replay achieves 423% in 9×60M environment steps. Given 9×200M steps it achieves 448%. Shared experience replay obtains better performance than not shared experience replay. This confirms the efficient use of parallel exploration (Kretchmar, 2002).

In Table 1 we consider the classic regime. Here we train a single LASER agent with experience replay for 200M steps per game (no sweep). Observe that it achieves 431% compared to the previous state of the art of 288% (Xu et al., 2018) within 200M steps. The fastest prior agent to reach a score of 400% is presented by Kapturowski et al. (2019) requiring more than 3,000M steps.

6. Conclusion

We have presented LASER – an off-policy actor-critic agent which employs a large experience replay. It achieves state-of-the-art data efficiency in two regimes: (1) where it is trained for 200M steps on it's own and (2) where it may share experience off-policy with concurrently training agents.

To facilitate this algorithm we have proposed two approaches: a) mixing replayed experience and on-policy data

and b) a trust region scheme. We have shown theoretically and demonstrated through a series of experiments that they enable learning in strongly off-policy settings, which present a challenge for conventional importance sampling schemes.

Acknowledgments

We would like to thank Aidan Clark, Will Dabney, Andrei Kashin, Remi Munos, Frank Perbet, Hubert Soyer, Jeff Stanway and Zhe Wang.

References

Amos, B., Dinh, L., Cabi, S., Rothörl, T., Colmenarejo, S. G., Muldal, A., Erez, T., Tassa, Y., de Freitas, N., and Denil, M. Learning awareness models. In *ICLR*, 2018.

Bacon, P., Harb, J., and Precup, D. The option-critic architecture. *AAAI*, 2017.

Beattie, C., Leibo, J. Z., Teplyashin, D., Ward, T., Wainwright, M., Küttler, H., Lefrancq, A., Green, S., Valdés, V., Sadik, A., Schrittwieser, J., Anderson, K., York, S., Cant, M., Cain, A., Bolton, A., Gaffney, S., King, H., Hassabis, D., Legg, S., and Petersen, S. Deepmind lab. *Arxiv*, abs/1612.03801, 2016.

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The Arcade Learning Environment: An evaluation platform for general agents. *JAIR*, 2013.

Bellman, R. A markovian decision process. *Journal of Mathematics and Mechanics*, 1957.

Conn, A. R., Gould, N. I. M., and Toint, Ph. L. *Trust-Region Methods*. SIAM, Philadelphia, PA, USA, 2000.

Dabney, W., Ostrovski, G., Silver, D., and Munos, R. Implicit quantile networks for distributional reinforcement learning. *Arxiv*, abs/1806.06923, 2018.

Degris, T., White, M., and Sutton, R. S. Off-policy actor-critic. In *ICML*, 2012.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., Legg, S., and Kavukcuoglu, K. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *ICML*, 2018.

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

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *ICML*, 2018.

Hessel, M., Modayil, J., van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M. G., and Silver, D. Rainbow: Combining improvements in deep reinforcement learning. *Arxiv*, abs/1710.02298, 2017.

Hessel, M., Soyer, H., Espeholt, L., Czarnecki, W., Schmitt, S., and van Hasselt, H. Multi-task deep reinforcement learning with popart. In *AAAI*, 2019.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735.

Horgan, D., Quan, J., Budden, D., Barth-Maron, G., Hessel, M., van Hasselt, H., and Silver, D. Distributed prioritized experience replay. In *ICLR*, 2018.

Jaderberg, M., Dalibard, V., Osindero, S., Czarnecki, W. M., Donahue, J., Razavi, A., Vinyals, O., Green, T., Dunning, I., Simonyan, K., Fernando, C., and Kavukcuoglu, K. Population based training of neural networks. *Arxiv*, abs/1711.09846, 2017a.

Jaderberg, M., Mnih, V., Czarnecki, W. M., Schaul, T., Leibo, J. Z., Silver, D., and Kavukcuoglu, K. Reinforcement learning with unsupervised auxiliary tasks. *ICLR*, 2017b.

Kahn, H. Use of different monte carlo sampling techniques. *Santa Monica, Calif.*, 1955.

Kaiser, L., Babaeizadeh, M., Milos, P., Osinski, B., Campbell, R. H., Czechowski, K., Erhan, D., Finn, C., Kozakowski, P., Levine, S., Sepassi, R., Tucker, G., and Michalewski, H. Model-based reinforcement learning for atari. *Arxiv*, abs/1903.00374, 2019.

Kapturowski, S., Ostrovski, G., Quan, J., Munos, R., and Dabney, W. Recurrent experience replay in distributed reinforcement learning. In *ICLR*, 2019.

Kretchmar, R. Parallel reinforcement learning. *SCI2002. The 6th World Conference on Systemics, Cybernetics, and Informatics. Orlando, FL*, 2002.

Lample, G. and Chaplot, D. S. Playing FPS games with deep reinforcement learning. In *AAAI*, 2016.

Lin, L.-J. *Reinforcement Learning for Robots Using Neural Networks*. PhD thesis, Pittsburgh, PA, USA, 1992. UMI Order No. GAX93-22750.

Mankowitz, D. J., Zídek, A., Barreto, A., Horgan, D., Hessel, M., Quan, J., Oh, J., van Hasselt, H., Silver, D., and Schaul, T. Unicorn: Continual learning with a universal, off-policy agent. *Arxiv*, abs/1802.08294, 2018.

Mirowski, P., Pascanu, R., Viola, F., Soyer, H., Ballard, A. J., Banino, A., Denil, M., Goroshin, R., Sifre, L., Kavukcuoglu, K., Kumaran, D., and Hadsell, R. Learning to navigate in complex environments. In *ICLR*, 2017.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. A. Playing atari with deep reinforcement learning. *Arxiv*, abs/1312.5602, 2013.

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. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533, 2015.

Munos, R., Stepleton, T., Harutyunyan, A., and Bellemare, M. Safe and efficient off-policy reinforcement learning. In *NIPS*. 2016.

Nair, A., Srinivasan, P., Blackwell, S., Alcicek, C., Fearon, R., De Maria, A., Panneershelvam, V., Suleyman, M., Beattie, C., Petersen, S., Legg, S., Mnih, V., Kavukcuoglu, K., and Silver, D. Massively parallel methods for deep reinforcement learning. *Arxiv*, abs/1507.04296, 2015.

Pascanu, R., Mikolov, T., and Bengio, Y. Understanding the exploding gradient problem. *Arxiv*, abs/1211.5063, 2012.

Precup, D., Sutton, R. S., and Singh, S. P. Eligibility traces for off-policy policy evaluation. In *ICML*, 2000.

Riedmiller, M. Neural fitted q iteration – first experiences with a data efficient neural reinforcement learning method. In *ECML*, 2005.

Schaul, T., Quan, J., Antonoglou, I., and Silver, D. Prioritized experience replay. In *ICLR*, 2015.

Schaul, T., Borsa, D., Modayil, J., and Pascanu, R. Ray interference: a source of plateaus in deep reinforcement learning. 2019.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In Bach, F. and Blei, D. (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1889–1897, Lille, France, 07–09 Jul 2015. PMLR.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *Arxiv*, abs/1707.06347, 2017.

Song, H. F., Abdolmaleki, A., Springenberg, J. T., Clark, A., Soyer, H., Rae, J. W., Noury, S., Ahuja, A., Liu, S., Tirumala, D., Heess, N., Belov, D., Riedmiller, M., and Botvinick, M. M. V-mpo: On-policy maximum a posteriori policy optimization for discrete and continuous control. In *ICLR*, 2020.

Sutton, R. S., Precup, D., and Singh, S. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artif. Intell.*, 112(1-2):181–211, August 1999. ISSN 0004-3702. doi: 10.1016/S0004-3702(99)00052-1.

Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., and Precup, D. Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. *Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, 2011.

Sutton, R. S., Mahmood, A. R., Precup, D., and van Hasselt, H. A new q(λ) with interim forward view and Monte Carlo equivalence. In *ICML*, 2014.

Sutton, R. S., Barto, A. G., and Bach, F. *Reinforcement Learning: An Introduction*. MIT press, 2018.

Tesauro, G. Temporal difference learning and td-gammon. *Commun. ACM*, 38(3):58–68, March 1995. ISSN 0001-0782. doi: 10.1145/203330.203343.

Wang, Z., Bapst, V., Heess, N., Mnih, V., Munos, R., Kavukcuoglu, K., and de Freitas, N. Sample efficient actor-critic with experience replay. In *ICLR*, 2017.

Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *ML*, 1992.

Xu, Z., van Hasselt, H. P., and Silver, D. Meta-gradient reinforcement learning. In *NIPS*. 2018.

Zhang, S. and Sutton, R. S. A deeper look at experience replay. *Arxiv*, abs/1712.01275, 2017.