Supplementary Material: Off-Policy Actor-Critic with Shared Experience Replay

1. Additional Experiments

1.1. Reduced Clipping in V-trace does not Enable Shared Experience Replay

Increasing the clipping constant $\bar{\rho}$ in V-trace reduces bias in favour of increased variance. We investigate if reducing bias in this manner enables sharing experience replay between multiple agents in a hyper-parameter sweep. Figure 1 (left) shows that this is not a solution, thus motivating our trust region scheme.

1.2. Prioritized and Uniform Experience Replay, LSTM States

With prioritized experience replay each transition τ is sampled with probability $P(\tau) \propto p_\tau^\alpha$, for a suitable unnormalized priority score p_τ and a global tunable parameter α. It is common (Schaul et al., 2015; Horgan et al., 2018; Hessel et al., 2017) to then weight updates computed from that sample by $1/P(\tau)^\beta$ for $0 < \beta \leq 1$, where $\beta = 1$ fully corrects for the bias introduced in the state distribution. In one step temporal difference methods, typical priorities are based on the immediate TD-error, and are typically recomputed after a transition is sampled from replay. This means low priorities might stay low and get stale – even if the transition suddenly becomes relevant. To alleviate this issue, the sampling distribution is mixed with a uniform, as controlled by a third hyper parameter ϵ.

The performance of agents with prioritized experience replay can be quite sensitive to the hyper-parameters α, β, and ϵ.

A critical practical consideration is how to implement random access for recurrent memory agents such as agents using an LSTM. Prioritized agents sample a presumably interesting transition from the past. This transition may be at any position within the episode. To infer the correct recurrent memory-state at this environment-state all earlier environment-states within that episode would need to be replayed. A prioritized agent with a random access pattern would thus require costly LSTM refreshes for each sampled transition. If LSTM states are not recomputed representational missmatch (Kapturowski et al., 2019) occurs.

Sharing experience between multiple agents amplifies the issue of LSTM state representation missmatch. Here each agent has its own network parameters and the state representations between agents may be arbitrarily different.

As a mitigation Kapturowski et al. (2019) use a burn-in window or to initialize with a constant starting state. We note that those solutions can only partially mitigate the fundamental issue and that counter examples such as arbitrarily long T-Mazes (Tolman, 1948; Olton, 1979) can be constructed easily.

We thus advocate for uniform sampling. In our implementation we uniformly sample an episode. Then we replay each episode from the beginning, using the most recent network parameters to recompute the LSTM states along the way: this is particularly critical when sharing experience between different agents, which may have arbitrarily different state representations.

This solution is exact and cost-efficient as it only requires one additional forward pass for each learning step (forward + backward pass).

An even more cost efficient approach would be to not refresh LSTM states at all. Naturally this comes at the cost of representational missmatch. However it would allow for an affordable implementation of prioritized experience replay. We investigate this in Figure 1 (right) and observe that it is not viable. We compare a baseline V-trace agent with no experience replay, one with uniform experience replay, and two different prioritized replay agents. We do not refresh LSTM states for any of the agents.

The uniform replay agent is more data efficient then the baseline, and also saturates at a higher level of performance. The best prioritized replay agent uses full importance sampling corrections ($\beta = 1$). However it performs no higher than with uniform replay. We therefore we used uniform replay with full state correction for all our investigations in the paper.

1.3. Evaluation Protocol

For evaluation, we average episode returns within buckets of 1M (Atari) and 10M (DMLab) environment steps for each agent instance, and normalize scores on each game by using the scores of a human expert and a random agent

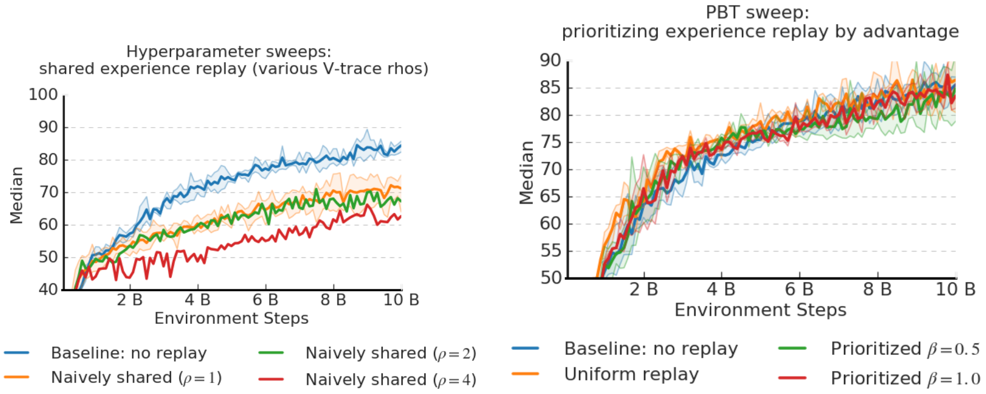

Figure 1. **Left** Increasing the V-trace clipping constant $\bar{\rho}$ does not enable shared experience replay. In fact sharing experience replay in this particular way is worse than pure online learning. This motivates the use of our proposed trust region scheme. On a side note, increased clipping thresholds resulting in worse performance verifies the importance of variance reduction through clipping. **Right:** Median human normalized performance across 30 tasks for the best agent in a sweep, averaged across 2 replicas. All replay experiments use 50% replay ratio and a capacity of 3 million observations. We investigate if uncorrected LSTM states can be used in combination with different replay modes. We consider uniform sampling and prioritization via the critic's loss, and include both full ($\beta = 1$) and partial ($\beta = 0.5$) importance corrections

(van Hasselt et al., 2016). In the multi-task setting, we then define the performance of each agent as the median normalized score of all levels that the agent trains on. Given the use of population based training, we need to perform the comparisons between algorithms at the level of sweeps. We do so by selecting the best performing agent instance within each sweep at any time. Note that for the multi-task setting, our approach of first averaging across many episodes, then taking the median across games, on DMLab further downsampling to 100M env steps, and only finally selecting the maximum within the sweep, results in substantially lower variance than if we were to compute the maximum before the median and smoothing.

All DMLab-30 sweeps are repeated $3\times$ with the exception of $\rho = 2$ and $\rho = 4$ in Figure 1. We then plot a shaded area between the point-wise best and worst replica and a solid line for the mean. Atari sweeps having 57 games are summarized and plotted by the median of the human-normalized scores.

2. Algorithm Pseudocode

We present algorithm pseudocode for LASER with trust region (Algorithm 1). For clarity we present a version without LSTM and focus on the single agent case. The multi-agent case is a simple extension where all agents save to the same replay database and also sample from the same replay. Also each agent starts with different network parameters and hyper-parameters. The LSTM state recomputation can be achieved with *Replayer Threads* (nearly identical to Actor Threads) that sample entire epsiodes from replay, step through them while reevaluating the LSTM state and slice the experience into trajectories of length T. Similar to regular LSTM Actor Threads from Espeholt et al. (2018) the Replayer Threads send each trajectory together with an LSTM state to the learning thread via a queue. The Learner Thread initializes the LSTM with the transmitted state when the LSTM is unrolled over the trajectory.

3. Propositions

We have stated five propositions in our paper for which we provide proofs below.

3.1. Definitions

We recall the definitions for the importance sampling and the V-trace return estimators:

$$G^{\pi,\mu_z}_{\text{IS}}(s_t) = V(s_t) + \sum_{k=0}^{\infty} \gamma^k \Big(\prod_{i=0}^{k} \lambda_{\pi,\mu_z}(s_{t+i}) \frac{\pi_{t+i}}{\mu_{z,t+i}} \Big) \delta_{t+k} V \tag{1}$$

$$G^{\pi,\mu_z}_{\text{Vtrace}}(s_t) = V(s_t) + \sum_{k=0}^{\infty} \gamma^k \Big(\prod_{i=0}^{k-1} \lambda_{\pi,\mu_z}(s_{t+i}) c_{z,t+i} \Big) \lambda_{\pi,\mu_z}(s_{t+k}) \rho_{z,t+k} \delta_{t+k} V \tag{2}$$

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 3 that depends on the behaviour policy μ:*

$$\tilde{\pi}_\mu(a|x) = \frac{\min\left[\bar{\rho}\mu(a|x), \pi(a|x)\right]}{\sum_{b\in A} \min\left[\bar{\rho}\mu(b|x), \pi(b|x)\right]} \tag{3}$$

Algorithm 1 Single Agent LASER with Trust Region

```
Initialize parameter vectors $\theta$. Initialize $\pi_1 = \pi_\theta$.
Actor Thread:
while training is ongoing do
    Sample trajectory unroll $u = \{\tau_t\}_{t\in\{1,\dots,T\}}$ of length $T$ by acting in the environment using the latest $\pi_k$ where $\tau_t = (s_t, a_t, r_t, \mu_t = \pi_k(s_t|\cdot))$.
    Enqueue $u$ into Lerner Queue, wait if full.
    Add $u$ into Replay Database.
    Remove oldest episode if database has reached desired capacity limit.
end while
Learner Thread:
Given: Batch size $B$, online fraction $\alpha$.
for training iteration $k$ do
    Form training batch $U = \{u_b\}_{b\in\{1,\dots,B\}}$ of $B$ trajectories of length $T$, by dequeuing $B\alpha$ trajectories from Lerner Queue and sampling $B(1-\alpha)$ trajectories from Replay Database.
    Evaluate the target policy $\pi_k$ on the sampled transitions $s_{b,t}$ in $U$: i.e. $\pi_k(s_{b,t}|\cdot)$.
    Compute behaviour relevance mask $M$ with $M_{b,t} = \mathrm{KL}(\pi_k(s_{b,t}|\cdot)||\mu_{b,t}) < b$ where $\mu_{b,t}, s_{b,t}$ are obtained from $U_{b,t}$.
    Compute trust-region V-trace return $V_{t,b}$ (see section 3.1) where $\lambda_{\pi,\mu}(s_{b,t}) = M_{b,t}$.
    Let $[L_V(\theta)]_{t,b} = \frac{1}{2}(V_{t,b} - V_\theta(s_{t,b}))^2$.
    Let $A_{t,b} = V_{t,b} - V_\theta(s_{t,b})$ and $[L_P(\theta)]_{t,b} = \rho_{t,b}\log[\pi_\theta(s_{t,b}|a_{t,b})]A_{t,b}$, where $\rho$ is the clipped v-trace importance sampling ratio.
    Perform gradient update to $\theta$ using $\nabla_\theta \sum_{t,b}[L_V(\theta) + L_P(\theta)]_{t,b}M_{t,b}$, denote the resulting $\pi_\theta$ as $\pi_{k+1}$.
end for
```

Proof. See Espeholt et al. (2018). □

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. Proof by contradiction:

Consider a tabular counter example with a single (locally) optimal policy at s_t given by $\pi^*(s_t) = \mathrm{argmax}_\pi \left[\sum_{a\in A}\pi(a|s_t)Q^*(a,s_t)\right]$ that always selects the action $\mathrm{argmax}_a Q^*(a,s_t)$.

Even in this ideal tabular setting V-trace policy gradient estimates a different $\tilde{\pi}^*$ rather than the optimal π^* as follows

$$\begin{aligned}
\nabla V^{*,\pi}(s_t) &= \mathbf{E}_\mu\left[\rho_t(r_t+\gamma V^*(s_{t+1})\nabla\log\pi(a_t|s_t)\right] \\
&= \mathbf{E}_\mu\left[\rho_t Q^*(s_t,a_t)\nabla\log\pi(a_t|s_t)\right] \\
&= \mathbf{E}_\mu\left[\min\left[\frac{\pi(a_t|s_t)}{\mu(a_t|s_t)},\bar{\rho}\right] Q^*(s_t,a_t)\nabla\log\pi(a_t|s_t)\right] \\
&= \mathbf{E}_\mu[\\
&\frac{\pi(a_t|s_t)}{\mu(a_t|s_t)}\min\left[1,\bar{\rho}\frac{\mu(a_t|s_t)}{\pi(a_t|s_t)}\right] Q^*(s_t,a_t)\nabla\log\pi(a_t|s_t)\Big] \\
&= \mathbf{E}_\pi\left[\min\left[1,\bar{\rho}\frac{\mu(a_t|s_t)}{\pi(a_t|s_t)}\right] Q^*(s_t,a_t)\nabla\log\pi(a_t|s_t)\right] \\
&= \mathbf{E}_\pi\left[\omega(s_t,a_t)Q^*(s_t,a_t)\nabla\log\pi(a_t|s_t)\right] \\
&= \mathbf{E}_\pi\left[Q^{*,\omega}(s_t,a_t)\nabla\log\pi(a_t|s_t)\right]
\end{aligned} \tag{4}$$

Observe how the optimal Q-function Q^* is scaled by $\omega(s_t,a_t) = \min\left[1,\bar{\rho}\frac{\mu(a_t|s_t)}{\pi(a_t|s_t)}\right] \leq 1$ resulting in *implied state-action values* $Q^{*,\omega}$. This penalizes actions where $\mu(a_t|s_t)\bar{\rho} < \pi(a_t|s_t)$ and makes V-trace greedy w.r.t. to the remaining ones. Thus μ can be chosen adversarially to corrupt the optimal state action value. Note that $\bar{\rho}$ is a constant typically chosen to be 1.

To prove the lemma consider a counter example such as an MDP with two actions and $Q^* = (2,5)$ and $\mu = (0.9, 0.1)$ and initial $\pi = (0.5, 0.5)$. Here the second action with expected return 5 is clearly favourable. Abusing notation $\mu/\pi = (1.8, 0.2)$. Thus $Q^{\tilde{\pi},\omega} = (2*1, 5*0.2) = (2,1)$. Therefore $\tilde{\pi}^* = (1,0)$ wrongly selects the first action. □

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 $\sum_s d^\mu(s)\mathbf{E}_\pi\left[(Q(s,a)\nabla\log\pi(a|s)\right]$ to $\sum_s \mathbf{E}_\pi\left[Q^\alpha(s,a)\nabla\log\pi(a|s)\right]$ 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. Note that the on-policy policy gradient is given by

$$\nabla J_{\mathrm{on}}(\pi) = \sum_s d^\pi(s)\mathbf{E}_\pi\left[Q(s,a)\nabla\log\pi(a|s)\right]$$

Similarly the off-policy V-trace gradient is given by

$$\nabla J_{\mathrm{off}}(\pi) = \sum_s d^\mu(s)\mathbf{E}_\pi\left[\omega(s,a)Q(s,a)\nabla\log\pi(a|s)\right]$$

with the V-trace distortion factor $\omega(s_t, a_t) = \min\left[1, \bar{\rho}\frac{\mu(a_t|s_t)}{\pi(a_t|s_t)}\right] \leq 1$ that can de-emphasize action values and $Q^\omega(s,a) = \omega(s,a)Q(s,a)$.

The α-interpolation of both gradients can be transformed as follows:

$$\begin{aligned}
&\nabla\big(\alpha J_{\text{on}} + (1-\alpha)J_{\text{off}}\big)(\pi) \\
&\quad = \alpha \sum_s d^\pi(s)\mathbf{E}_\pi\left[Q(s,a)\nabla\log\pi(a|s)\right] \\
&\quad + (1-\alpha)\sum_s d^\mu(s)\mathbf{E}_\pi\left[\omega(s,a)Q(s,a)\nabla\log\pi(a|s)\right] \\
&\quad = \sum_s d^\pi(s)\mathbf{E}_\pi\left[Q(s,a)\alpha\nabla\log\pi(a|s)\right] \\
&\quad + \sum_s d^\mu(s)\mathbf{E}_\pi\left[Q^\omega(s,a)(1-\alpha)\nabla\log\pi(a|s)\right] \\
&\quad = \sum_s \mathbf{E}_\pi\left[Q(s,a)d^\pi(s)\alpha\nabla\log\pi(a|s)\right] \\
&\quad + \sum_s \mathbf{E}_\pi\left[Q^\omega(s,a)d^\mu(s)(1-\alpha)\nabla\log\pi(a|s)\right] \\
&\quad = \sum_s \mathbf{E}_\pi\left[Q(s,a)d^\pi(s)\alpha\nabla\log\pi(a|s)\right] \\
&\quad + \mathbf{E}_\pi\left[Q^\omega(s,a)d^\mu(s)(1-\alpha)\nabla\log\pi(a|s)\right] \\
&\quad = \sum_s \mathbf{E}_\pi\left[\big(Q(s,a)d^\pi(s)\alpha\right] \\
&\quad + \mathbf{E}_\pi\left[Q^\omega(s,a)d^\mu(s)(1-\alpha)\big)\nabla\log\pi(a|s)\right] \\
&\quad = \sum_s \mathbf{E}_\pi\left[Q^\alpha(s,a)\nabla\log\pi(a|s)\right]
\end{aligned} \tag{5}$$

for $Q^\alpha(s,a) = Q(s,a)d^\pi(s)\alpha + Q^\omega(s,a)d^\mu(s)(1-\alpha)$ □

Interpretation of Proposition 3 As discussed in section 3 of the paper the V-trace policy gradient will have the correct local fixpoint at state s if the argmax of the state-value function is preserved despite the distortion: i.e. if $\text{argmax}_a[Q(s,a)] = \text{argmax}_a[Q^\omega(s,a)]$. Respectively when mixing in an $\alpha \in [0,1)$ share of online data the fixpoint will be preserved if

$$\text{argmax}_a[Q(s,a)] = \text{argmax}_a[Q^\alpha(s,a)] \tag{6}$$

Let $a^* = \text{argmax}_b(Q,b)$ be any best action and A^* be set of best actions. Then equation 6 is equivalent to:

$$Q^\alpha(s,a^*) > Q^\alpha(s,b)\,\forall b \notin A^*$$

Using the definition of Q^α this can be rewritten as:

$$\begin{aligned}
&Q(s,a^*)d^\pi(s)\alpha + Q^\omega(s,a^*)d^\mu(s)(1-\alpha) \\
&\quad > Q(s,b)d^\pi(s)\alpha + Q^\omega(s,b)d^\mu(s)(1-\alpha)\,\forall b \notin A^*
\end{aligned}$$

Which can be rearranged to:

$$\begin{aligned}
&[Q(s,a^*)d^\pi(s) - Q(s,b)d^\pi(s)]\alpha \\
&\quad > [Q^\omega(s,b)d^\mu(s) - Q^\omega(s,a^*)d^\mu(s)](1-\alpha)\,\forall b \notin A^*
\end{aligned}$$

By definition $Q(s,a^*)d^\pi(s) - Q(s,b)d^\pi(s) > 0\,\forall b \notin A^*$, hence:

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

It follows that the policy gradient will have the same local fixpoint if

$$\frac{\alpha}{1-\alpha} > \max_{b \notin A^*}\left[\frac{Q^\omega(s,b) - Q^\omega(s,a^*)}{Q(s,a^*) - Q(s,b)}\right]\frac{d^\mu(s)}{d^\pi(s)} \tag{7}$$

Note that $\frac{\alpha}{1-\alpha} \to \infty$ as $\alpha \to 1$. Mixing-in more online data thus increases the left hand side. Also 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. Furthermore the larger the action value gap in the real Q-function $Q(s,a^*) - Q(s,b)$ the lower the right hand side. Finally the denominator will be negative if $\max_{b \notin A^*}[Q^\omega(s,b)] < Q^\omega(s,a^*)$ thus enabling correct learning even in the pure off-policy case with $\alpha = 0$.

Note that 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.

Proposition 4. *Let $G_{\text{IS}}^{\pi,\mu_z}$ be a set of importance sampling estimators as defined in section 3.1. 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_{trusted}^π. In particular $|V_{\text{trusted}}^\pi - V^\pi|_\infty \leq \gamma\,|V - V^\pi|_\infty$.*

Proof. Let us consider the set of importance sampling estimators as defined in section 3.1 and note that they all contract to the same fixed point V^π with at least $\left|\mathbf{E}_{\mu_z|z}\left[G_{\text{IS}}^{\pi,\mu_z}(s)\right] - V^\pi(s)\right|_\infty \leq \gamma\,|V(s) - V^\pi(s)|_\infty$ for any state s.

By Minkowski's inequality the contraction properties of importance sampled Monte-Carlo bootstraps carry over to

V^{π}_{trusted} which is a $p(z|\mu_z \in M_{\beta,\pi}(s_t))$ weighted average:

$$
\begin{aligned}
&\left|V^{\pi}_{\text{trusted}}(s) - V^{\pi}(s)\right|_{\infty} \\
&= \left|\mathbf{E}_z\left[\mathbf{E}_{\mu_z|z}\left[G^{\pi,\mu_z}_{\text{IS}}(s)\right] \middle| \mu_z \in M_{\beta,\pi}(s_t)\right] - V^{\pi}(s)\right|_{\infty} \\
&= \left|\mathbf{E}_z\left[\mathbf{E}_{\mu_z|z}\left[G^{\pi,\mu_z}_{\text{IS}}(s)\right] - V^{\pi}(s) \middle| \mu_z \in M_{\beta,\pi}(s_t)\right]\right|_{\infty} \\
&\leq \mathbf{E}_z\left[\left|\mathbf{E}_{\mu_z|z}\left[G^{\pi,\mu_z}_{\text{IS}}(s)\right] - V^{\pi}(s_t)\right|_{\infty} \middle| \mu_z \in M_{\beta,\pi}(s_t)\right] \\
&< \mathbf{E}_z\left[\gamma\left|V(s) - V^{\pi}(s)\right|_{\infty} \middle| \mu_z \in M_{\beta,\pi}(s_t)\right] \\
&= \gamma\left|V(s) - V^{\pi}(s)\right|_{\infty}
\end{aligned}
\tag{8}
$$

□

Proposition 5. *Let $G^{\pi,\mu_z}_{\text{Vtrace}}$ be a set of V-trace estimators (see section 3.1) with corresponding fixed points V^z (see equation 3) 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 $\left|V^{\pi}_{\text{trusted}} - V^{\beta}\right|_{\infty} < \max_{\mu_z \in M_{\beta,\pi}(s_t)}\left|\eta_z(V - V^z)\right|_{\infty} \leq \eta_{\max}\max_{\mu_z \in M_{\beta,\pi}(s_t)}\left|(V - V^z)\right|_{\infty}$ with $\eta_{\max} = \max_{\mu_z \in M_{\beta,\pi}(s_t)}\eta_z$.*

Proof. Recall the contraction properties of a V-trace importance sampled Monte-Carlo bootstraps $G^{\pi,\mu_z}_{\text{Vtrace}}$ being

$$\left|\mathbf{E}_{\mu_z|z}\left[G^{\pi,\mu_z}_{\text{Vtrace}}(s)\right] - V^z(s)\right|_{\infty} < \eta_z\left|V(s) - V^z(s)\right|_{\infty}$$

for an algorithm and behaviour specific $\eta_z < 1$ for a z dependent fixed point V^z and for any bootstrap V. We then show that V^{π}_{trusted} moves towards the weighted average of fixed points $V^{\beta} = \mathbf{E}_{z|\mu_z \in M_{\beta,\pi}(s_t)}\left[V^z\right]$, since

$$\left|V^{\pi}_{\text{trusted}}(s) - V^{\beta}(s)\right|_{\infty} < \eta_{\max}\max_{\mu_z \in M_{\beta,\pi}(s_t)}\left|V(s) - V^z(s)\right|_{\infty}$$

holds for any bootstrap function V as we show below.

$$
\begin{aligned}
&\left|V^{\pi}_{\text{trusted}}(s) - V^{\beta}(s)\right|_{\infty} \\
&= \left|\mathbf{E}_z\left[\mathbf{E}_{\mu_z|z}\left[G^{\pi,\mu_z}_{\text{Vtrace}}(s)\right] - V^z(s) \middle| \mu_z \in M_{\beta,\pi}(s_t)\right]\right|_{\infty} \\
&\leq \mathbf{E}_z\left[\left|\mathbf{E}_{\mu_z|z}\left[G^{\pi,\mu_z}_{\text{Vtrace}}(s)\right] - V^z(s)\right|_{\infty} \middle| \mu_z \in M_{\beta,\pi}(s_t)\right] \\
&< \mathbf{E}_z\left[\left|\eta_z(V(s) - V^z(s))\right|_{\infty} \middle| \mu_z \in M_{\beta,\pi}(s_t)\right] \\
&\leq \max_{\mu_z \in M_{\beta,\pi}(s_t)}\left|\eta_z(V(s) - V^z(s))\right|_{\infty} \\
&\leq \eta_{\max}\max_{\mu_z \in M_{\beta,\pi}(s_t)}\left|V(s) - V^z(s)\right|_{\infty}
\end{aligned}
\tag{9}
$$

□

4. Detailed Atari Results

We display the Atari per-level performance of various agents at 50M and 200M environment steps in Table 1. The scores correspond to the agents presented in Figure 1 of the paper. The LASER scores are computed by averaging the last 100 episode returns before 50M or respectively 200M environment frames have been experienced. Following the procedure defined by Mnih et al. (2015) we initialize the environment with a random number of no-op actions (up to 37 in our case). Again following Mnih et al. (2015) episodes are terminated after 30 minutes of gameplay. Note that Xu et al. (2018) have not published per-level scores. Rainbow scores are obtained from Hessel et al. (2017).

References

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Volodymir Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *ICML*, 2018.

Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Daniel Horgan, Bilal Piot, Mohammad Gheshlaghi Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. *Arxiv*, abs/1710.02298, 2017.

Dan Horgan, John Quan, David Budden, Gabriel Barth-Maron, Matteo Hessel, Hado van Hasselt, and David Silver. Distributed prioritized experience replay. In *ICLR*, 2018.

Steven Kapturowski, Georg Ostrovski, John Quan, Remi Munos, and Will Dabney. Recurrent experience replay in distributed reinforcement learning. In *ICLR*, 2019.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

David S. Olton. Mazes, maps, and memory. *American Psychologist*, 1979.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *ICLR*, 2015.

Edward C. Tolman. Cognitive maps in rats and men. *The Psychological Review*, 1948.

Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double Q-learning. In *AAAI*, 2016.

Zhongwen Xu, Hado P van Hasselt, and David Silver. Meta-gradient reinforcement learning. In *NIPS*. 2018.

Table 1. Per level performance of various agents corresponding to Figure 1 of the paper.

Game	LASER Shared (sweep at 9×50M)	LASER Shared (sweep at 9×200M)	LASER (no sweep at 200M)	Rainbow (no sweep at 200M)
alien	18635.3	18277.3	**35565.9**	9491.7
amidar	1838.3	2695	1829.2	**5131.2**
assault	26027.1	**40603.2**	21560.4	14198.5
asterix	**496735.0**	240770	240090	428200
asteroids	232651	**257420.1**	213025	2712.8
atlantis	**889934.0**	866584	841200	826660
bank_heist	1333.1	**1712.8**	569.4	1358
battle_zone	66900	**131880.0**	64953.3	62010
beam_rider	80830.5	**125795.2**	90881.6	16850.2
berzerk	46651.6	**64513.1**	25579.5	2545.6
bowling	42.4	47.4	**48.3**	30
boxing	99.8	99.4	**100.0**	99.6
breakout	**852.5**	850.3	747.9	417.5
centipede	208008	**409702.8**	292792	8167.3
chopper_command	24814	727333	**761699.0**	16654
crazy_climber	160494	88818	167820	**168788.5**
defender	355447	**369397.0**	336953	55105
demon_attack	133557	**138000.6**	133530	111185
double_dunk	0.1	**23.5**	14	-0.3
enduro	0	0	0	**2125.9**
fishing_derby	45.4	**62.6**	45.2	31.3
freeway	**34.0**	**34.0**	0	**34.0**
frostbite	5297.4	2230.8	5083.5	**9590.5**
gopher	86222.2	39721.2	**114820.7**	70354.6
gravitar	1360.5	**2812.0**	1106.2	1419.3
hero	30159.2	36510.6	31628.7	**55887.4**
ice_hockey	20.2	**38.7**	17.4	1.1
jamesbond	21663	**60402.5**	37999.8	19809
kangaroo	13932	14187	14308	**14637.5**
krull	**9559.3**	5743.6	9387.5	8741.5
kung_fu_master	65032	81792	**607443.0**	52181
montezuma_revenge	1	1	0.3	**384.0**
ms_pacman	6089.3	**6890.7**	6565.5	5380.4
name_this_game	25998.9	**27910.7**	26219.5	13136
phoenix	458355	**628711.6**	519304	108529
pitfall	-0.2	-0.2	-0.6	**0.0**
pong	**21.0**	**21.0**	**21.0**	20.9
private_eye	100	100	96.3	**4234.0**
qbert	20283.8	24600.8	21449.6	**33817.5**
riverraid	24138.1	35491.5	**40362.7**	22920.8
road_runner	52942	**63762.0**	45289	62041
robotank	63.6	**67.8**	62.1	61.4
seaquest	1802.2	**557213.3**	2890.3	15898.9
skiing	**-8904.8**	-8980.1	-29968.4	-12957.8
solaris	2222.4	3017.6	2273.5	**3560.3**
space_invaders	36071.4	**53124.3**	51037.4	18789
star_gunner	331327	**602540.0**	321528	127029
surround	**9.8**	**9.8**	8.4	9.7
tennis	0	0	**12.2**	0
time_pilot	77899	**113603.0**	105316	12926
tutankham	251.8	268.5	**278.9**	241
up_n_down	341988	**368586.5**	345727	125755
venture	0	0	0	**5.5**
video_pinball	513121	397451	511835	**533936.5**
wizard_of_wor	22280	**45335.0**	29059.3	17862.5
yars_revenge	145055	144370	**166292.3**	102557
zaxxon	50486	**106862.0**	41118	22209.5