Eligibility traces, V-trace, IMPALA

Milan Straka

November 30, 2020

EUROPEAN UNION
European Structural and Investment Fund
Operational Programme Research,
Development and Education

Charles University in Prague
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics

BY SA
unless otherwise stated

Off-policy Correction Using Control Variates

Let $G_{t:t+n}$ be the estimated n-step return

$$G_{t:t+n} \stackrel{\text{def}}{=} \left(\sum_{k=t}^{t+n-1} \gamma^{k-t} R_{k+1}\right) + \big[\text{episode still running in } t+n\big]\gamma^n V(S_{t+n}),$$

which can be written recursively as

$$G_{t:t+n} \begin{cases} 0 & \text{if episode ended before } t, \\ V(S_t) & \text{if } n = 0, \\ R_{t+1} + \gamma G_{t+1:t+n} & \text{otherwise.} \end{cases}$$

Off-policy Correction Using Control Variates

Note that we can write

$$\begin{aligned}G_{t:t+n} - V(S_t) &= R_{t+1} + \gamma G_{t+1:t+n} - V(S_t) \\ &= R_{t+1} + \gamma\big(G_{t+1:t+n} - V(S_{t+1})\big) + \gamma V(S_{t+1}) - V(S_t),\end{aligned}$$

which yields

$$G_{t:t+n} - V(S_t) = R_{t+1} + \gamma V(S_{t+1}) - V(S_t) + \gamma\big(G_{t+1:t+n} - V(S_{t+1})\big).$$

Denoting the TD error as $\delta_t \overset{\text{def}}{=} R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$, we can therefore write the n-step estimated return as a sum of TD errors:

$$G_{t:t+n} = V(S_t) + \sum_{i=0}^{n-1} \gamma^i \delta_{t+i}.$$

Off-policy Correction Using Control Variates

Now consider applying the IS off-policy correction to $G_{t:t+n}$ using the importance sampling ratio

$$\rho_t \stackrel{\text{def}}{=} \frac{\pi(A_t|S_t)}{b(A_t|S_t)}, \quad \rho_{t:t+n} \stackrel{\text{def}}{=} \prod_{i=0}^{n} \rho_{t+i}.$$

First note that

$$\mathbb{E}_{A_t \sim b}\big[\rho_t\big] = \sum_{A_t} b(A_t|S_t) \frac{\pi(A_t|S_t)}{b(A_t|S_t)} = 1,$$

which can be extended to

$$\mathbb{E}_b\big[\rho_{t:t+n}\big] = 1.$$

Off-policy Correction Using Control Variates

Until now, we used

$$G^{\text{IS}}_{t:t+n} \overset{\text{def}}{=} \rho_{t:t+n-1} G_{t:t+n}.$$

However, such correction has unnecessary variance. Notably, when expanding $G_{t:t+n}$

$$G^{\text{IS}}_{t:t+n} = \rho_{t:t+n-1}\big(R_{t+1} + \gamma G_{t+1:t+n}\big),$$

the R_{t+1} depends only on ρ_t, not on $\rho_{t+1:t+n}$, and given that the expectation of the importance sampling ratio is 1, we can simplify to

$$G^{\text{IS}}_{t:t+n} = \rho_t R_{t+1} + \rho_{t:t+n-1}\gamma G_{t+1:t+n}.$$

Such an estimate can be written recursively as

$$G^{\text{IS}}_{t:t+n} = \rho_t\big(R_{t+1} + \gamma G^{\text{IS}}_{t+1:t+n}\big).$$

Off-policy Correction Using Control Variates

We can reduce the variance even further – when $\rho_t = 0$, we might consider returning the value of $V(S_t)$ instead of 0.

Therefore, we might add another term, the so-called **control variate**, to the estimate

$$G_{t:t+n}^{\mathrm{CV}} \stackrel{\mathrm{def}}{=} \rho_t \big(R_{t+1} + \gamma G_{t+1:t+n}^{\mathrm{CV}}\big) + (1 - \rho_t) V(S_t),$$

which is valid, since the expected value of $1 - \rho_t$ is zero and ρ_t and S_t are independent.

Similarly as before, rewriting to

$$\begin{aligned} G_{t:t+n}^{\mathrm{CV}} - V(S_t) &= \rho_t \big(R_{t+1} + \gamma G_{t+1:t+n}^{\mathrm{CV}}\big) - \rho_t V(S_t) \\ &= \rho_t \big(R_{t+1} + \gamma V(S_{t+1}) - V(S_t) + \gamma (G_{t+1:t+n}^{\mathrm{CV}} - V(S_{t+1}))\big) \end{aligned}$$

results in

$$G_{t:t+n}^{\mathrm{CV}} = V(S_t) + \sum\nolimits_{i=0}^{n-1} \gamma^i \rho_{t:t+i} \delta_{t+i}.$$

Eligibility Traces

Eligibility traces are a mechanism of combining multiple n-step return estimates for various values of n.

First note instead of an n-step return, we can use any average of n-step returns for different values of n, for example $\frac{2}{3}G_{t:t+2} + \frac{1}{3}G_{t:t+4}$.

λ-return

For a given $\lambda \in [0, 1]$, we define **λ-return** as

$$G_t^\lambda \stackrel{\text{def}}{=} (1-\lambda) \sum_{i=1}^\infty \lambda^{i-1} G_{t:t+i}.$$

Alternatively, the λ-return can be written recursively as

$$G_t^\lambda = (1-\lambda) G_{t:t+1} \\ + \lambda(R_{t+1} + \gamma G_{t+1}^\lambda).$$

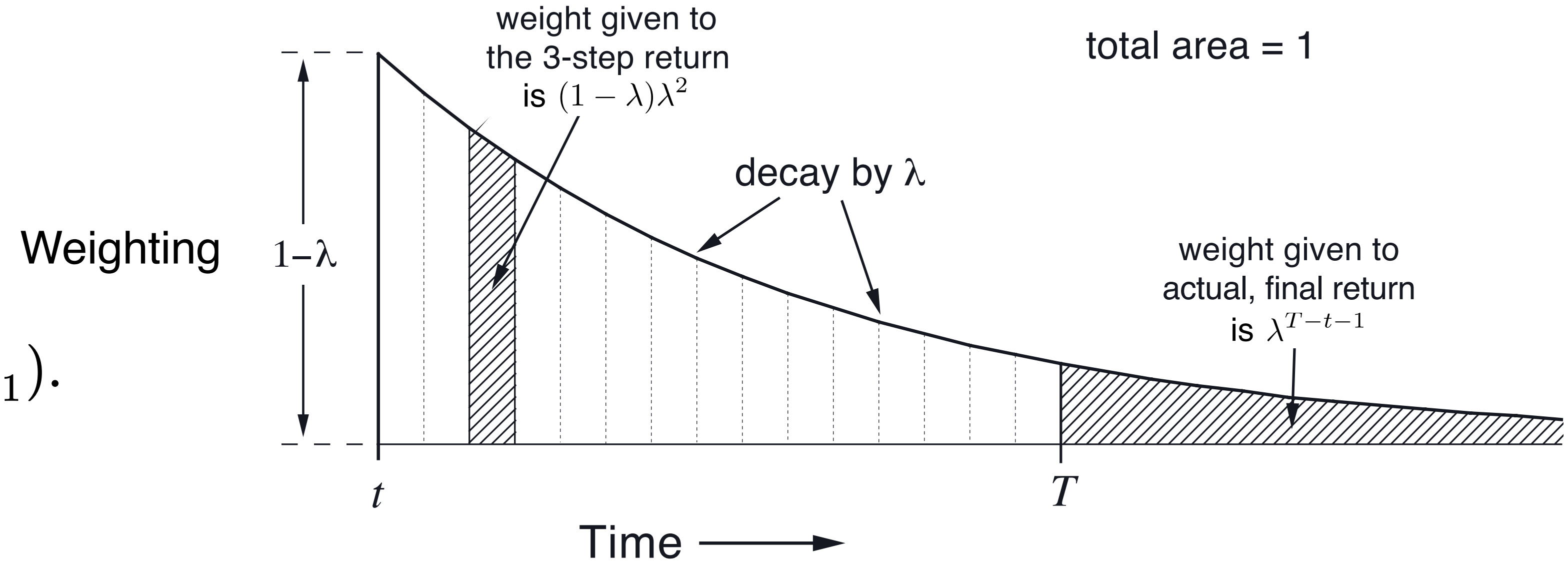


Figure 12.2: Weighting given in the λ-return to each of the n-step returns.

Figure 12.2 of "Reinforcement Learning: An Introduction, Second Edition".

λ-return

In an episodic task with time of termination T, we can rewrite the λ-return to

$$G_t^\lambda = (1-\lambda)\sum_{i=1}^{T-t-1} \lambda^{i-1} G_{t:t+i} + \lambda^{T-t-1} G_t.$$

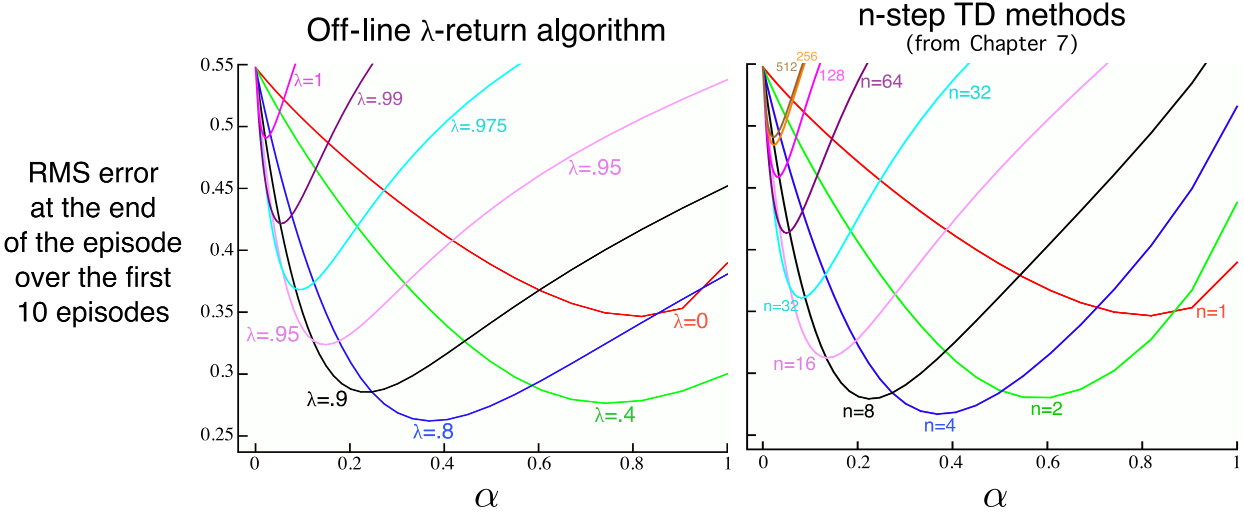


Figure 12.3: 19-state Random walk results (Example 7.1): Performance of the off-line λ-return algorithm alongside that of the n-step TD methods. In both case, intermediate values of the bootstrapping parameter (λ or n) performed best. The results with the off-line λ-return algorithm are slightly better at the best values of α and λ, and at high α.

Figure 12.3 of "Reinforcement Learning: An Introduction, Second Edition".

Truncated λ-return

We might also set a limit on the largest value of n, obtaining **truncated λ-return**

$$G_{t:t+n}^{\lambda} \stackrel{\text{def}}{=} (1-\lambda)\sum_{i=1}^{n-1}\lambda^{i-1}G_{t:t+i} + \lambda^{n-1}G_{t:t+n}.$$

The truncated λ return can be again written recursively as

$$G_{t:t+n}^{\lambda} = (1-\lambda)G_{t:t+1} + \lambda(R_{t+1} + \gamma G_{t+1:t+n}^{\lambda}),\ \ G_{t:t+1}^{\lambda} = G_{t:t+1}.$$

Similarly to before, we can express the truncated λ return as a sum of TD errors

$$\begin{aligned}G_{t:t+n}^{\lambda} - V(S_t) &= (1-\lambda)\big(R_{t+1} + \gamma V(S_{t+1})\big) + \lambda(R_{t+1} + \gamma G_{t+1:t+n}^{\lambda}) - V(S_t)\\ &= R_{t+1} + \gamma V(S_{t+1}) - V(S_t) + \lambda\gamma\big(G_{t+1:t+n}^{\lambda} - V(S_{t+1})\big),\end{aligned}$$

obtaining an analogous estimate $G_{t:t+n}^{\lambda} = V(S_t) + \sum_{i=0}^{n-1}\gamma^i\lambda^i\delta_{t+i}$.

Variable λs

The (truncated) λ-return can be generalized to utilize different λ_i at each step i. Notably, we can generalize the recursive definition

$$G_{t:t+n}^{\lambda} = (1-\lambda)G_{t:t+1} + \lambda(R_{t+1} + \gamma G_{t+1:t+n}^{\lambda})$$

to

$$G_{t:t+n}^{\lambda_i} = (1-\lambda_{t+1})G_{t:t+1} + \lambda_{t+1}(R_{t+1} + \gamma G_{t+1:t+n}^{\lambda_i}),$$

and express this quantity again by a sum of TD errors:

$$G_{t:t+n}^{\lambda_i} = V(S_t) + \sum_{i=0}^{n-1} \gamma^i \left(\prod_{j=1}^{i} \lambda_{t+j}\right) \delta_{t+i}.$$

Off-policy Traces with Control Variates

Finally, we can combine the eligibility traces with off-policy estimation using control variates:

$$G_{t:t+n}^{\lambda,\mathrm{CV}} \stackrel{\text{def}}{=} (1-\lambda)\sum_{i=1}^{n-1} \lambda^{i-1} G_{t:t+i}^{\mathrm{CV}} + \lambda^{n-1} G_{t:t+n}^{\mathrm{CV}}.$$

Recalling that

$$G_{t:t+n}^{\mathrm{CV}} = \rho_t\big(R_{t+1} + \gamma G_{t+1:t+n}^{\mathrm{CV}}\big) + (1-\rho_t)V(S_t),$$

we can rewrite $G_{t:t+n}^{\lambda,\mathrm{CV}}$ recursively as

$$G_{t:t+n}^{\lambda,\mathrm{CV}} = (1-\lambda)G_{t:t+1}^{\mathrm{CV}} + \lambda\Big(\rho_t\big(R_{t+1} + \gamma G_{t+1:t+n}^{\lambda,\mathrm{CV}}\big) + (1-\rho_t)V(S_t)\Big),$$

which we can simplify by expanding $G_{t:t+1}^{\mathrm{CV}} = \rho_t(R_{t+1} + \gamma V(S_{t+1})) + (1-\rho_t)V(S_t)$ to

$$G_{t:t+n}^{\lambda,\mathrm{CV}} - V(S_t) = \rho_t\big(R_{t+1} + \gamma V(S_{t+1}) - V(S_t)\big) + \gamma\lambda\rho_t\big(G_{t+1:t+n}^{\lambda,\mathrm{CV}} - V(S_{t+1})\big).$$

Off-policy Traces with Control Variates

Consequently, analogously as before, we can write the off-policy traces estimate with control variates as

$$G_{t:t+n}^{\lambda,\mathrm{CV}} = V(S_t) + \sum\nolimits_{i=0}^{n-1} \gamma^i \lambda^i \rho_{t:t+i} \delta_{t+i},$$

and by repeating the above derivation we can extend the result also for time-variable λi, obtaining

$$G_{t:t+n}^{\lambda_\mathrm{i},\mathrm{CV}} = V(S_t) + \sum\nolimits_{i=0}^{n-1} \gamma^i \left(\prod_{j=1}^{i} \lambda_{t+j}\right) \rho_{t:t+i} \delta_{t+i}.$$

Return Recapitulation

Recursive definition	Formulation with TD errors
$G_{t:t+n} \overset{\text{def}}{=} R_{t+1} + \gamma G_{t+1:t+n}$	$V(S_t) + \sum_{i=0}^{n-1} \gamma^i \delta_{t+i}$
$G^{\text{IS}}_{t:t+n} \overset{\text{def}}{=} \rho_t \big(R_{t+1} + \gamma G^{\text{IS}}_{t+1:t+n}\big)$	
$G^{\text{CV}}_{t:t+n} \overset{\text{def}}{=} \rho_t \big(R_{t+1} + \gamma G^{\text{CV}}_{t+1:t+n}\big) + (1-\rho_t)V(S_t)$	$V(S_t) + \sum_{i=0}^{n-1} \gamma^i \rho_{t:t+i} \delta_{t+i}$
$G^{\lambda}_{t:t+n} \overset{\text{def}}{=} (1-\lambda)G_{t:t+1} + \lambda(R_{t+1} + \gamma G^{\lambda}_{t+1:t+n})$	$V(S_t) + \sum_{i=0}^{n-1} \gamma^i \lambda^i \delta_{t+i}$
$G^{\lambda_i}_{t:t+n} \overset{\text{def}}{=} (1-\lambda_{t+1})G_{t:t+1} + \lambda_{t+1}(R_{t+1} + \gamma G^{\lambda_i}_{t+1:t+n})$	$V(S_t) + \sum_{i=0}^{n-1} \gamma^i \left(\prod_{j=1}^{i} \lambda_{t+j}\right) \delta_{t+i}$
$G^{\lambda,\text{CV}}_{t:t+n} \overset{\text{def}}{=} (1-\lambda)G^{\text{CV}}_{t:t+1}$ $+ \lambda\big(\rho_t\big(R_{t+1} + \gamma G^{\lambda,\text{CV}}_{t+1:t+n}\big) + (1-\rho_t)V(S_t)\big)$	$V(S_t) + \sum_{i=0}^{n-1} \gamma^i \lambda^i \rho_{t:t+i} \delta_{t+i}$
$G^{\lambda_i,\text{CV}}_{t:t+n} \overset{\text{def}}{=} (1-\lambda_{t+1})G^{\text{CV}}_{t:t+1}$ $+\lambda_{t+1}\big(\rho_t\big(R_{t+1} + \gamma G^{\lambda_i,\text{CV}}_{t+1:t+n}\big) + (1-\rho_t)V(S_t)\big)$	$V(S_t)$ $+ \sum_{i=0}^{n-1} \gamma^i \left(\prod_{j=1}^{i} \lambda_{t+j}\right) \rho_{t:t+i} \delta_{t+i}$

TD(λ)

We have defined the λ-return in the so-called **forward view**.

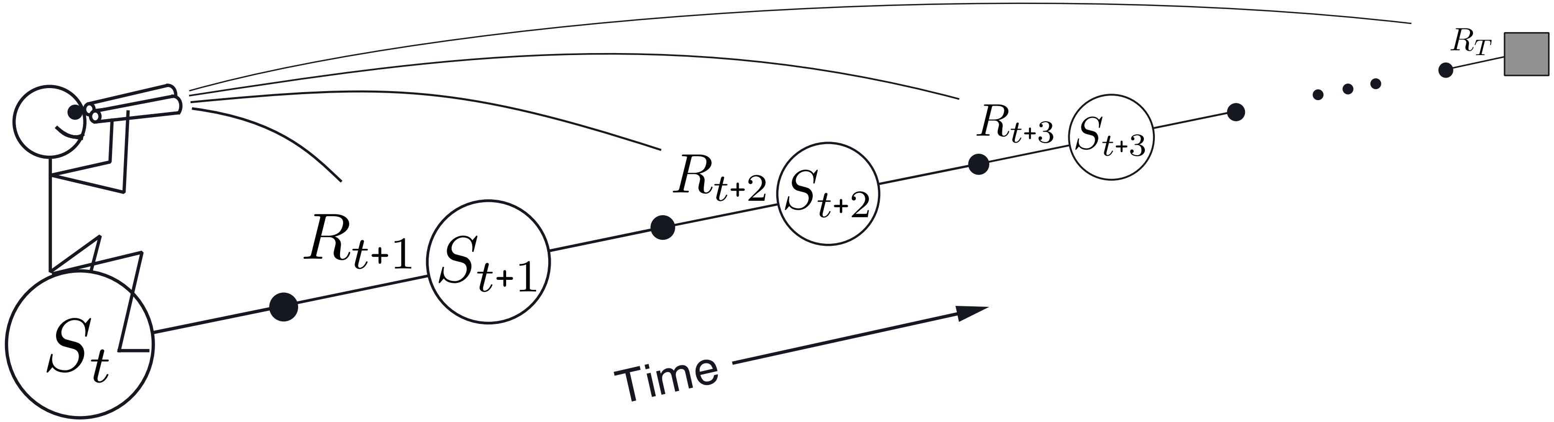


Figure 12.4: The forward view. We decide how to update each state by looking forward to future rewards and states.

Figure 12.4 of "Reinforcement Learning: An Introduction, Second Edition".

TD(λ)

However, to allow on-line updates, we might consider also the **backward view**

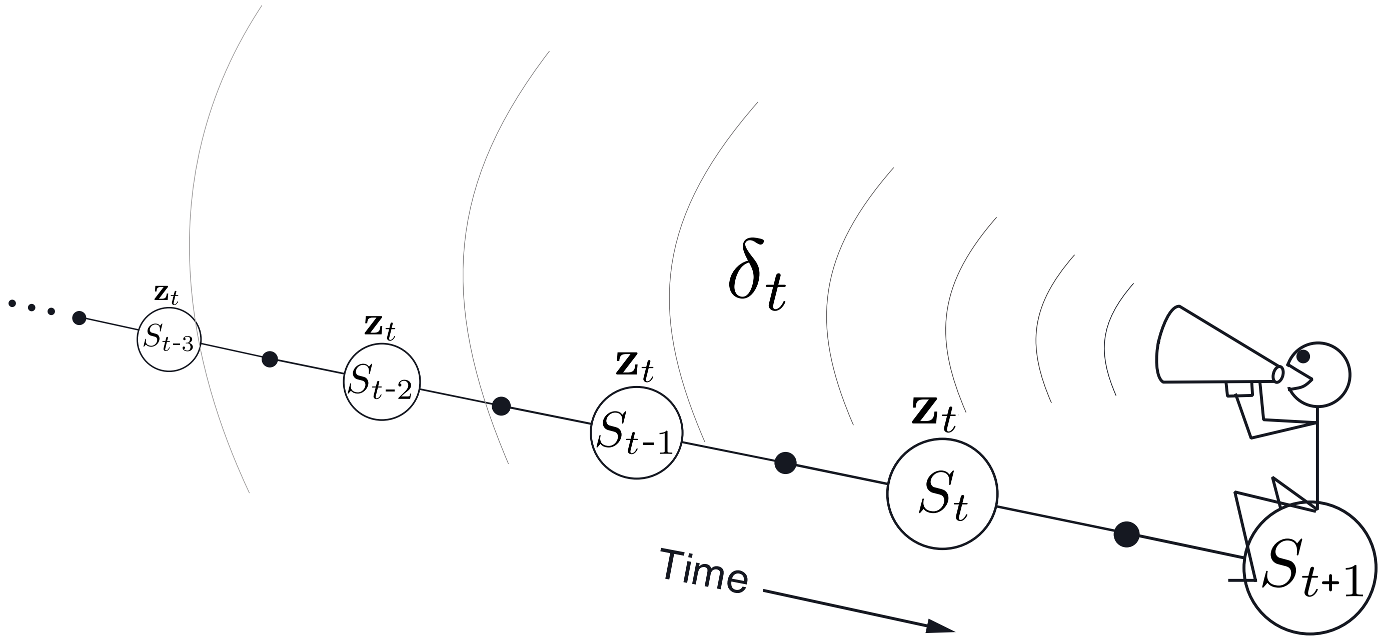


Figure 12.5: The backward or mechanistic view of TD(λ). Each update depends on the current TD error combined with the current eligibility traces of past events.

Figure 12.5 of "Reinforcement Learning: An Introduction, Second Edition".

TD(λ)

TD(λ) is an algorithm implementing on-line policy evaluation utilizing the backward view.

Semi-gradient TD(λ) for estimating $\hat{v} \approx v_\pi$

```
Input: the policy $\pi$ to be evaluated
Input: a differentiable function $\hat{v} : \mathcal{S}^+ \times \mathbb{R}^d \to \mathbb{R}$ such that $\hat{v}(\text{terminal},\cdot) = 0$
Algorithm parameters: step size $\alpha > 0$, trace decay rate $\lambda \in [0, 1]$
Initialize value-function weights $\mathbf{w}$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)

Loop for each episode:
    Initialize $S$
    $\mathbf{z} \leftarrow \mathbf{0}$                                        (a $d$-dimensional vector)
    Loop for each step of episode:
    |   Choose $A \sim \pi(\cdot|S)$
    |   Take action $A$, observe $R, S'$
    |   $\mathbf{z} \leftarrow \gamma\lambda\mathbf{z} + \nabla\hat{v}(S,\mathbf{w})$
    |   $\delta \leftarrow R + \gamma\hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w})$
    |   $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\mathbf{z}$
    |   $S \leftarrow S'$
    until $S'$ is terminal
```

Algorithm 12.2 of "Reinforcement Learning: An Introduction, Second Edition".

V-trace

V-trace is a modified version of n-step return with off-policy correction, defined in the Feb 2018 IMPALA paper as (using the notation from the paper):

$$G_{t:t+n}^{\text{V-trace}} \stackrel{\text{def}}{=} V(S_t) + \sum_{i=0}^{n-1} \gamma^i \left(\prod\nolimits_{j=0}^{i-1} \bar{c}_{t+j} \right) \bar{\rho}_{t+i} \delta_{t+i},$$

where $\bar{\rho}_t$ and $\bar{c}_t$ are the truncated importance sampling ratios for $\bar{\rho} \geq \bar{c}$:

$$\bar{\rho}_t \stackrel{\text{def}}{=} \min\left(\bar{\rho}, \frac{\pi(A_t|S_t)}{b(A_t|S_t)} \right), \quad \bar{c}_t \stackrel{\text{def}}{=} \min\left(\bar{c}, \frac{\pi(A_t|S_t)}{b(A_t|S_t)} \right).$$

Note that if $b = \pi$ and assuming $\bar{c} \geq 1$, v_s reduces to n-step Bellman target.

V-trace

Note that the truncated IS weights $\bar\rho_t$ and $\bar c_t$ play different roles:

- The $\bar\rho_t$ appears defines the fixed point of the update rule. For $\bar\rho = \infty$, the target is the value function v_π, if $\bar\rho < \infty$, the fixed point is somewhere between v_π and v_b. Notice that we do not compute a product of these $\bar\rho_t$ coefficients.

  Concretely, the fixed point of an operator defined by $G_{t:t+n}^{\text{V-trace}}$ corresponds to a value function of the policy

  $$\pi_{\bar\rho}(a|s) \propto \min\big(\bar\rho b(a|s), \pi(a|s)\big).$$

- The $\bar c_t$ impacts the speed of convergence (the contraction rate of the Bellman operator), not the sought policy. Because a product of the $\bar c_t$ ratios is computed, it plays an important role in variance reduction.

However, the paper utilizes $\bar c = 1$ and out of $\bar\rho \in \{1, 10, 100\}$, $\bar\rho = 1$ works empirically the best, so the distinction between $\bar c_t$ and $\bar\rho_t$ is not useful in practice.

V-trace Analysis

Let us define the (untruncated for simplicity; similar results can be proven for a truncated one) V-trace operator $\mathcal{R}$ as:

$$\mathcal{R}V(S_t) \stackrel{\text{def}}{=} V(S_t) + \mathbb{E}_b\left[\sum_{i\geq 0}\gamma^i\left(\prod\nolimits_{j=0}^{i-1}\bar{c}_{t+j}\right)\bar{\rho}_{t+i}\delta_{t+i}\right],$$

where the expectation $\mathbb{E}_b$ is with respect to trajectories generated by behaviour policy b. Assume there exists $\beta \in (0,1]$ such that $\mathbb{E}_b\bar{\rho}_0 \geq \beta$.

It can be proven (see Theorem 1 in Appendix A.1 in the Impala paper if interested) that such an operator is a contraction with a contraction constant

$$\gamma^{-1} - (1-\gamma)\sum_{i\geq 0}\gamma^i\mathbb{E}_b\left[\left(\prod\nolimits_{j=0}^{i-1}\bar{c}_j\right)\bar{\rho}_i\right] \leq 1 - (1-\gamma)\beta < 1,$$

therefore, $\mathcal{R}$ has a unique fixed point.

V-trace Analysis

We now prove that the fixed point of $\mathcal{R}$ is $V^{\pi_{\bar\rho}}$. We have:

$$\begin{aligned}
&\mathbb{E}_b\Big[\bar\rho_t\big(R_{t+1}+\gamma V^{\pi_{\bar\rho}}(S_{t+1}) - V^{\pi_{\bar\rho}}(S_t)\big)\Big|S_t\Big] \\
&= \sum_a b(a|S_t)\min\left(\bar\rho, \frac{\pi(a|S_t)}{b(a|S_t)}\right)\Big[R_{t+1}+\gamma\mathbb{E}_{s'\sim p(S_t,a)}V^{\pi_{\bar\rho}}(s') - V^{\pi_{\bar\rho}}(S_t)\Big] \\
&= \underbrace{\sum_a \pi_{\bar\rho}(a|S_t)\Big[R_{t+1}+\gamma\mathbb{E}_{s'\sim p(S_t,a)}V^{\pi_{\bar\rho}}(s') - V^{\pi_{\bar\rho}}(S_t)\Big]}_{=0}\sum_b \min\big(\bar\rho b(b|S_t), \pi(b|S_t)\big),
\end{aligned}$$

where the tagged part is zero, since it is the Bellman equation for $V^{\pi_{\bar\rho}}$. This shows that $\mathcal{R}V^{\pi_{\bar\rho}} = V^{\pi_{\bar\rho}}$, and therefore $V^{\pi_{\bar\rho}}$ is the unique fixed point of $\mathcal{R}$.

Consequently, in $G_{t:t+n}^{\lambda_\mathrm{i},\mathrm{CV}} = V(S_t) + \sum_{i=0}^{n-1}\gamma^i\left(\prod_{j=1}^i\lambda_{t+j}\right)\rho_{t:t+i}\delta_{t+i}$, only the last ρ_{t+i} from every $\rho_{t:t+i}$ is actually needed for off-policy correction; $\rho_{t:t+i-1}$ can be considered as traces.

IMPALA

Impala (**Imp**ortance Weighted **A**ctor-**L**earner **A**rchitecture) was suggested in Feb 2018 paper and allows massively distributed implementation of an actor-critic-like learning algorithm.

Compared to A3C-based agents, which communicate gradients with respect to the parameters of the policy, IMPALA actors communicate trajectories to the centralized learner.

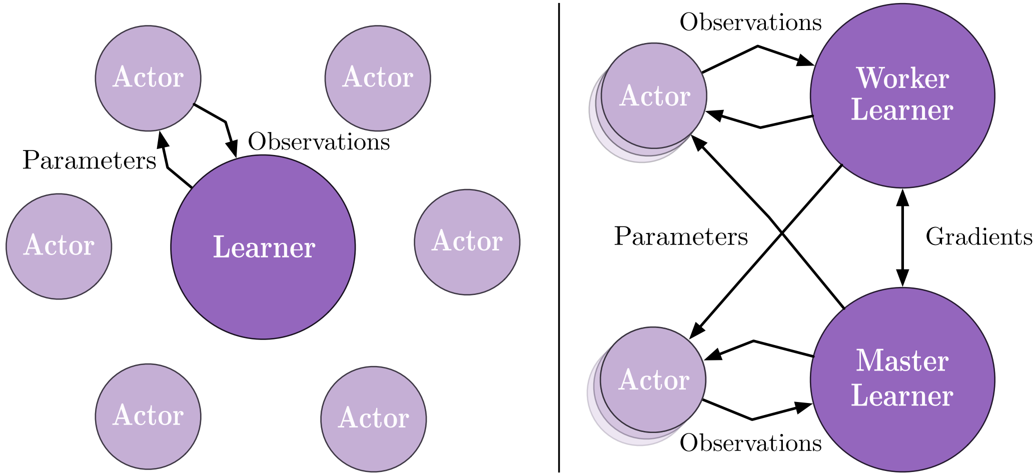


Figure 1 of the paper "IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures" by Lasse Espeholt et al.

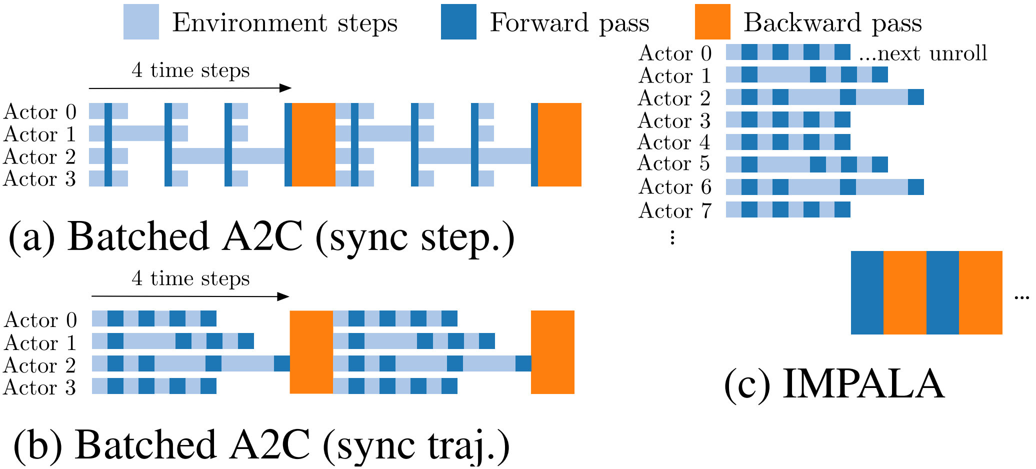


Figure 2 of the paper "IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures" by Lasse Espeholt et al.

If many actors are used, the policy used to generate a trajectory can lag behind the latest policy. Therefore, the V-trace off-policy actor-critic algorithm is employed.

IMPALA

Consider a parametrized functions computing $v(s;\boldsymbol\theta)$ and $\pi(a|s;\boldsymbol\omega)$, we update the critic in the direction of

$$\Big(G_{t:t+n}^{\text{V-trace}} - v(S_t;\boldsymbol\theta)\Big)\nabla_{\boldsymbol\theta} v(S_t;\boldsymbol\theta),$$

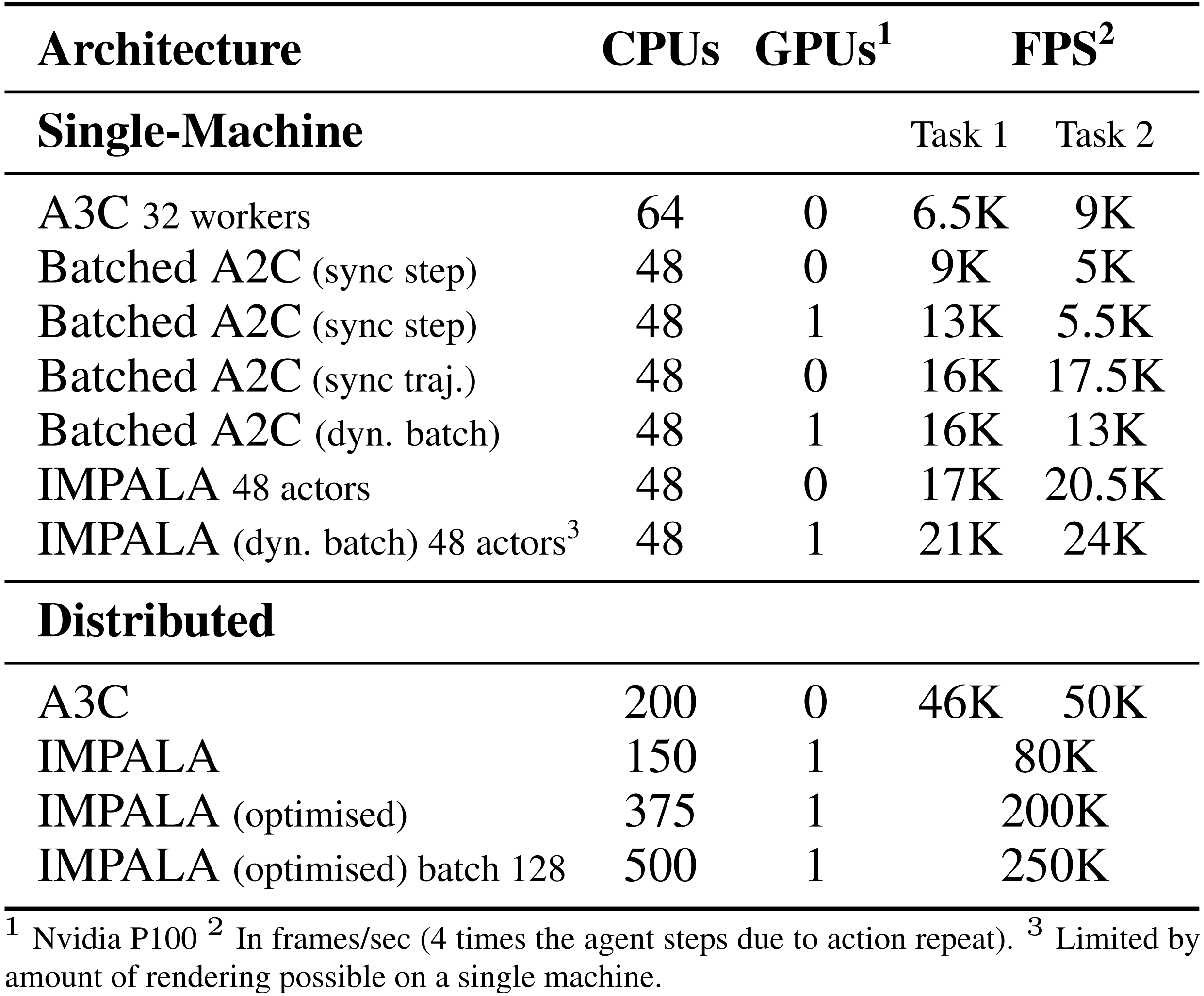
and the actor in the direction of the policy gradient

$$\bar\rho_t \nabla_{\boldsymbol\omega} \log\pi(A_t|S_t;\boldsymbol\omega)\big(R_{t+1} + \gamma G_{t+1:t+n}^{\text{V-trace}} - v(S_t;\boldsymbol\theta)\big),$$

where we estimate $Q^\pi(S_t, A_t)$ as $R_{t+1} + \gamma G_{t+1:t+n}^{\text{V-trace}}$.

Finally, we again add the entropy regularization term $\beta H\big(\pi(\cdot|S_t;\boldsymbol\omega)\big)$ to the loss function.

IMPALA



Architecture	CPUs	GPUs[1]	FPS[2]	
Single-Machine			Task 1	Task 2
A3C 32 workers	64	0	6.5K	9K
Batched A2C (sync step)	48	0	9K	5K
Batched A2C (sync step)	48	1	13K	5.5K
Batched A2C (sync traj.)	48	0	16K	17.5K
Batched A2C (dyn. batch)	48	1	16K	13K
IMPALA 48 actors	48	0	17K	20.5K
IMPALA (dyn. batch) 48 actors[3]	48	1	21K	24K
Distributed				
A3C	200	0	46K	50K
IMPALA	150	1	80K	
IMPALA (optimised)	375	1	200K	
IMPALA (optimised) batch 128	500	1	250K	

[1] Nvidia P100 [2] In frames/sec (4 times the agent steps due to action repeat). [3] Limited by amount of rendering possible on a single machine.

Table 1 of the paper "IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures" by Lasse Espeholt et al.

IMPALA – Population Based Training

For Atari experiments, population based training with a population of 24 agents is used to adapt entropy regularization, learning rate, RMSProp ε and the global gradient norm clipping threshold.

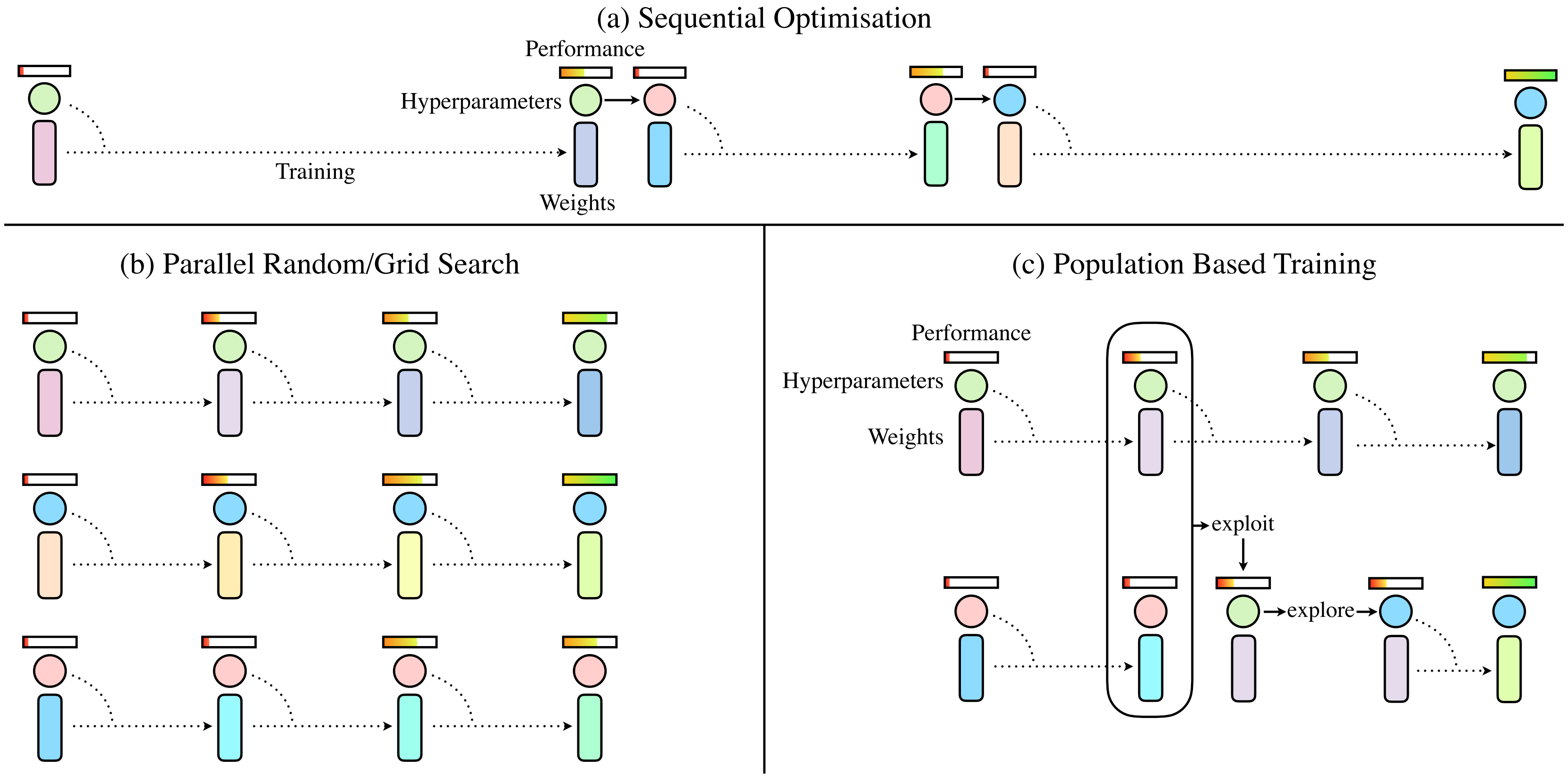


Figure 1 of paper "Population Based Training of Neural Networks" by Max Jaderberg et al.

IMPALA – Population Based Training

For Atari experiments, population based training with a population of 24 agents is used to adapt entropy regularization, learning rate, RMSProp ε and the global gradient norm clipping threshold.

In population based training, several agents are trained in parallel. When an agent is *ready* (after 5000 episodes), then:

- it may be overwritten by parameters and hyperparameters of another randomly chosen agent, if it is sufficiently better (5000 episode mean capped human normalized score returns are 5% better);
- and independently, the hyperparameters may undergo a change (multiplied by either 1.2 or $1/1.2$ with 33% chance).

IMPALA – Architecture

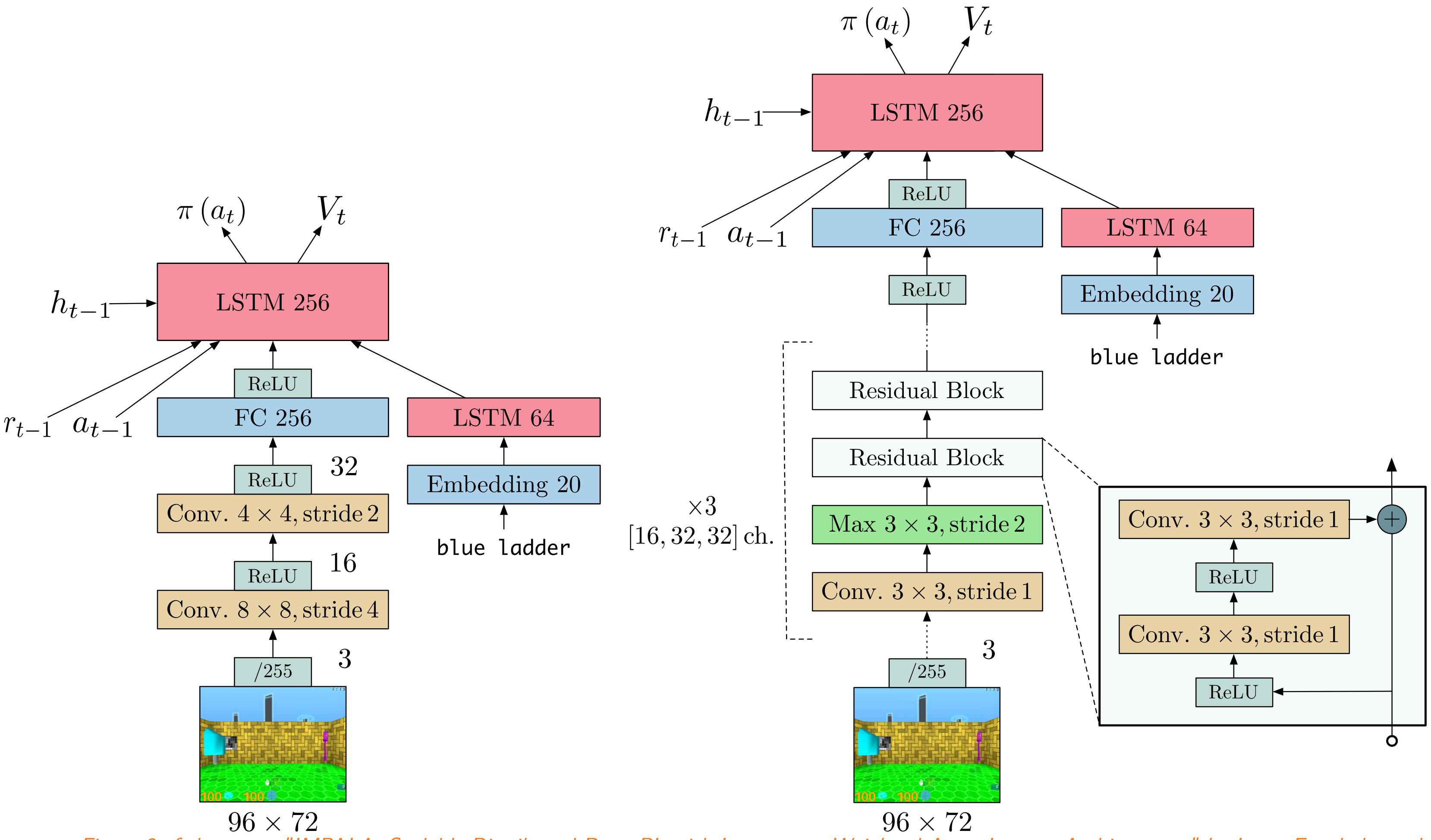


Figure 3 of the paper "IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures" by Lasse Espeholt et al.

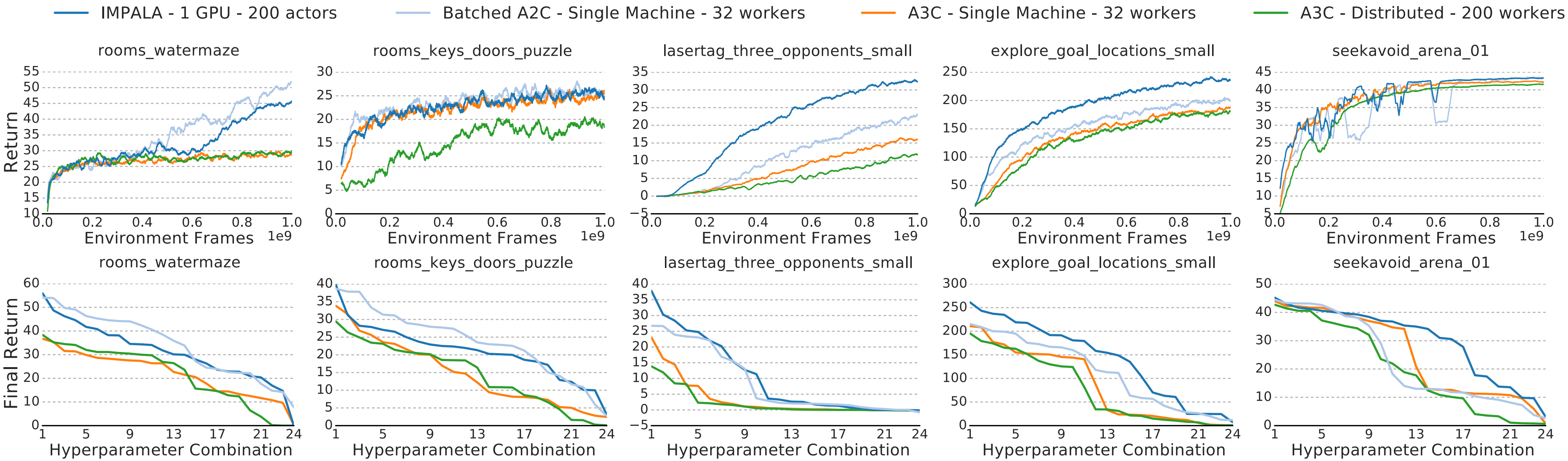
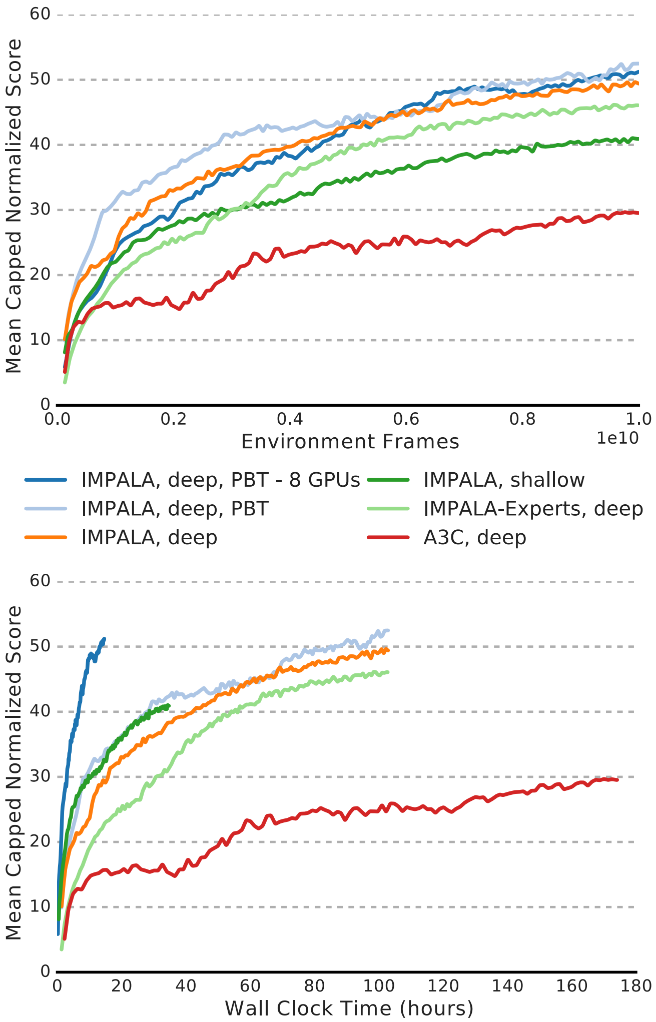


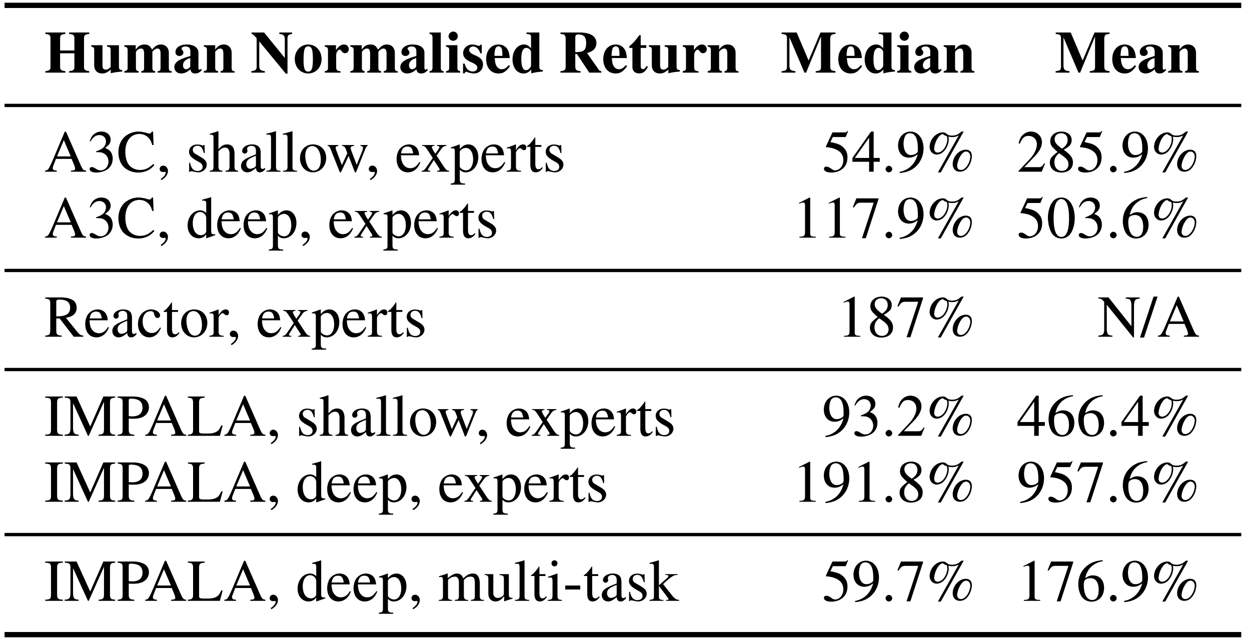
Figure 4 of the paper "IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures" by Lasse Espeholt et al.

IMPALA – Learning Curves



Figures 5, 6 of the paper "IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures" by Lasse Espeholt et al.

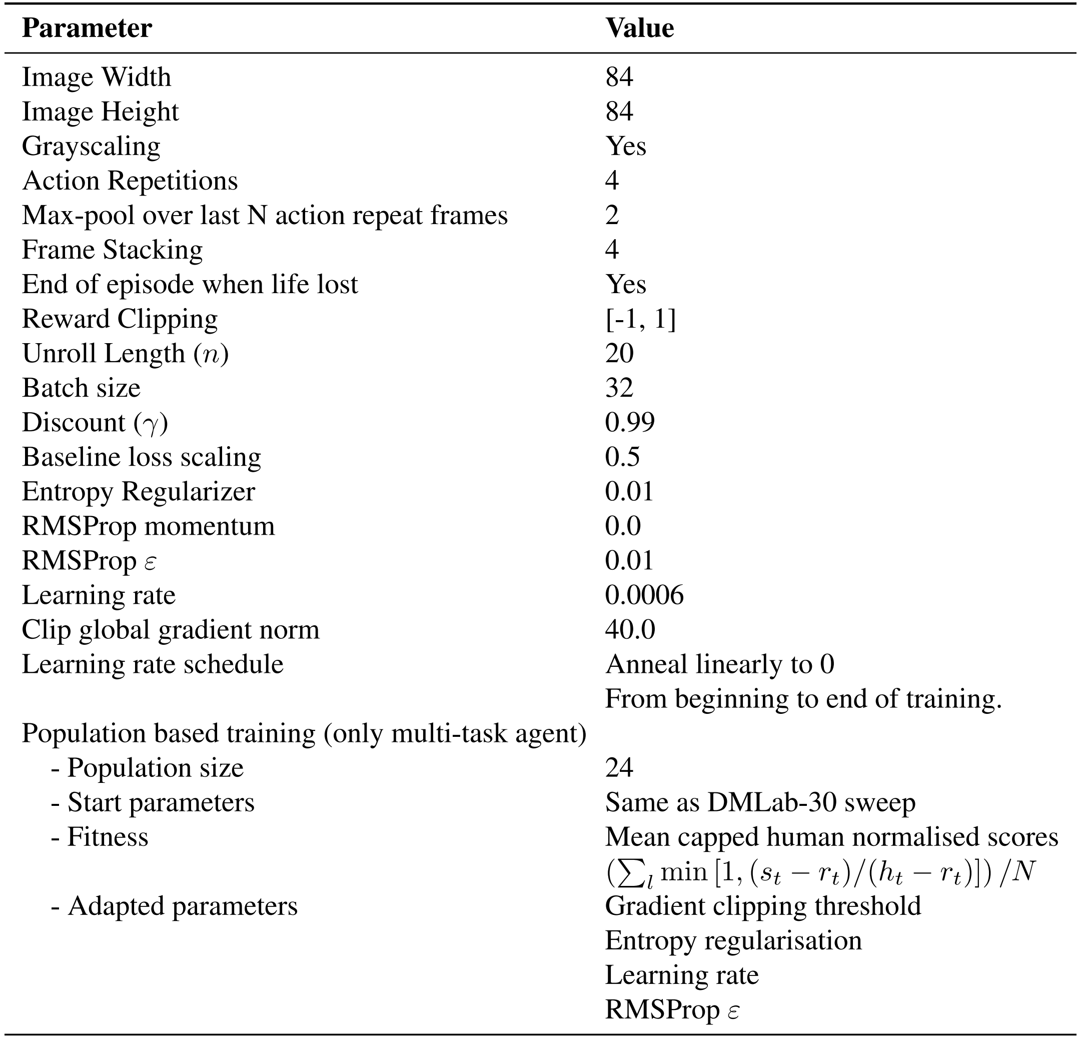
IMPALA – Atari Games



Human Normalised Return	Median	Mean
A3C, shallow, experts	54.9%	285.9%
A3C, deep, experts	117.9%	503.6%
Reactor, experts	187%	N/A
IMPALA, shallow, experts	93.2%	466.4%
IMPALA, deep, experts	191.8%	957.6%
IMPALA, deep, multi-task	59.7%	176.9%

Table 4 of the paper "IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures" by Lasse Espeholt et al.

IMPALA – Atari Hyperparameters

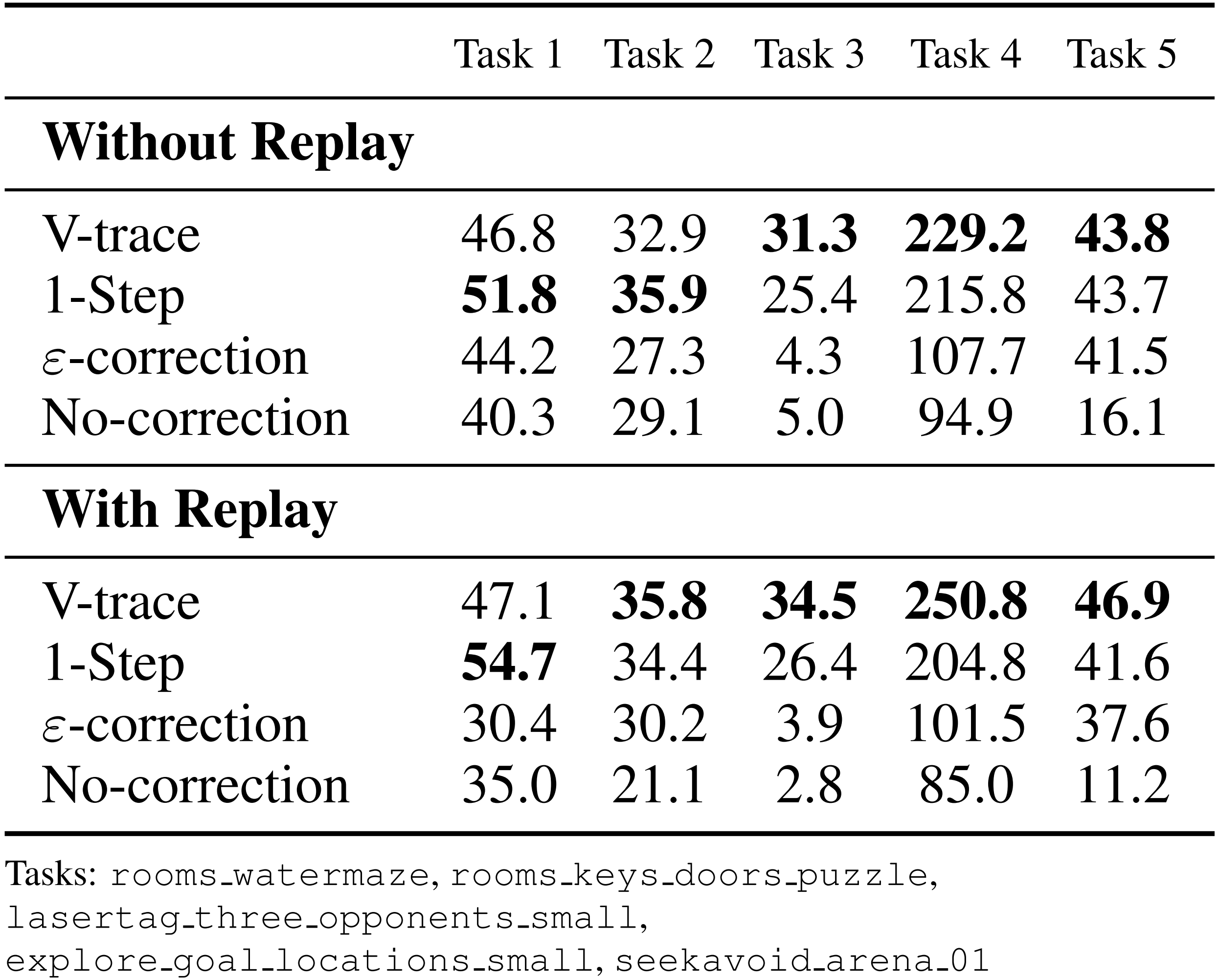


Parameter	Value
Image Width	84
Image Height	84
Grayscaling	Yes
Action Repetitions	4
Max-pool over last N action repeat frames	2
Frame Stacking	4
End of episode when life lost	Yes
Reward Clipping	[-1, 1]
Unroll Length (n)	20
Batch size	32
Discount (γ)	0.99
Baseline loss scaling	0.5
Entropy Regularizer	0.01
RMSProp momentum	0.0
RMSProp ε	0.01
Learning rate	0.0006
Clip global gradient norm	40.0
Learning rate schedule	Anneal linearly to 0 From beginning to end of training.
Population based training (only multi-task agent)	
- Population size	24
- Start parameters	Same as DMLab-30 sweep
- Fitness	Mean capped human normalised scores $\left(\sum_l \min\left[1, (s_t - r_t)/(h_t - r_t)\right]\right)/N$
- Adapted parameters	Gradient clipping threshold Entropy regularisation Learning rate RMSProp ε

Table G1 of the paper "IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures" by Lasse Espeholt et al.

IMPALA – Ablations

- **No-correction**: no off-policy correction;
- **ε-correction**: add a small value $\varepsilon = 10^{-6}$ during gradient calculation to prevent π to be very small and lead to unstabilities during $\log \pi$ computation;
- **1-step**: no off-policy correction in update of the value function, TD errors are multiplied by the corresponding ρ (but no cs).



	Task 1	Task 2	Task 3	Task 4	Task 5
Without Replay					
V-trace	46.8	32.9	**31.3**	**229.2**	**43.8**
1-Step	**51.8**	**35.9**	25.4	215.8	43.7
ε-correction	44.2	27.3	4.3	107.7	41.5
No-correction	40.3	29.1	5.0	94.9	16.1
With Replay					
V-trace	47.1	**35.8**	**34.5**	**250.8**	**46.9**
1-Step	**54.7**	34.4	26.4	204.8	41.6
ε-correction	30.4	30.2	3.9	101.5	37.6
No-correction	35.0	21.1	2.8	85.0	11.2

Tasks: `rooms_watermaze`, `rooms_keys_doors_puzzle`, `lasertag_three_opponents_small`, `explore_goal_locations_small`, `seekavoid_arena_01`

Table 2 of the paper "IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures" by Lasse Espeholt et al.

IMPALA – Ablations

The effect of the policy lag (the number of updates the actor is behind the learned policy) on the performance.

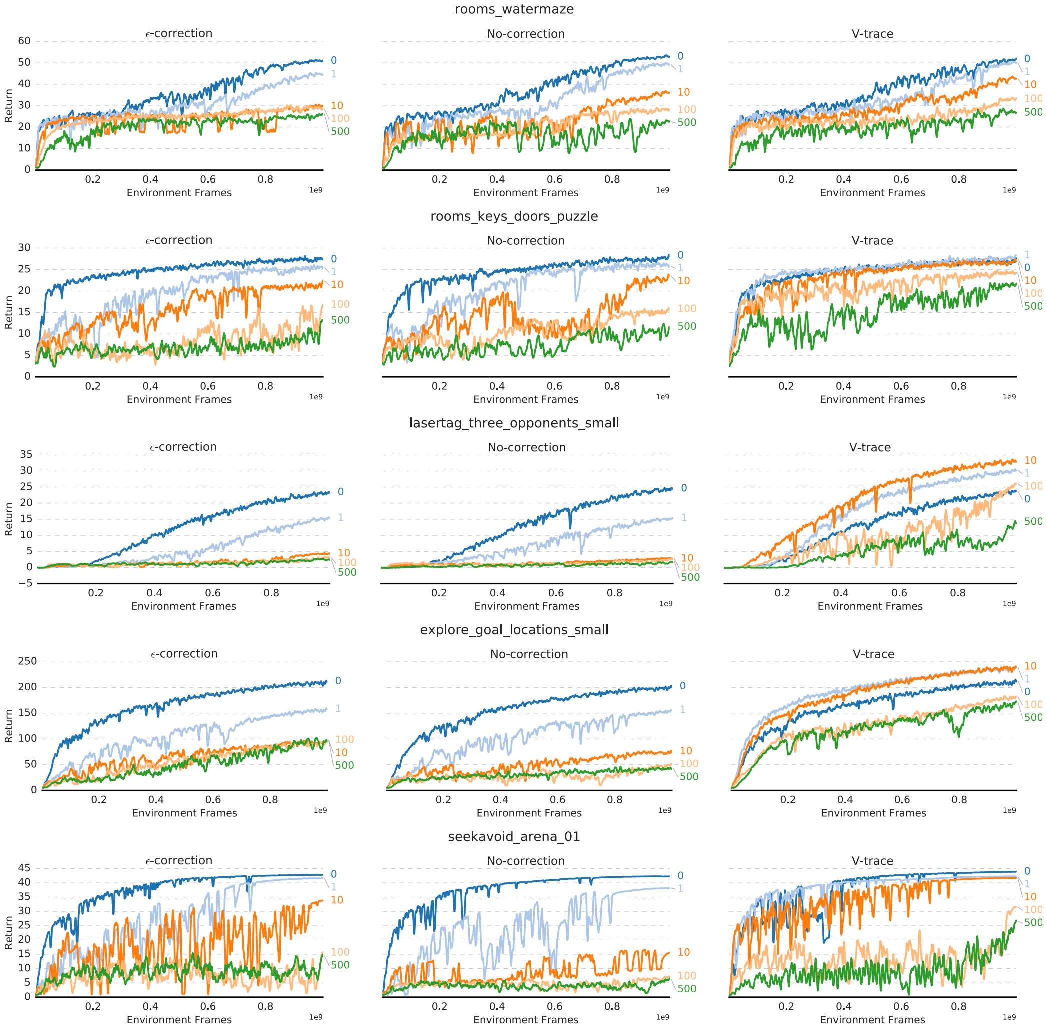


Figure E.1 of the paper "IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures" by Lasse Espeholt et al.