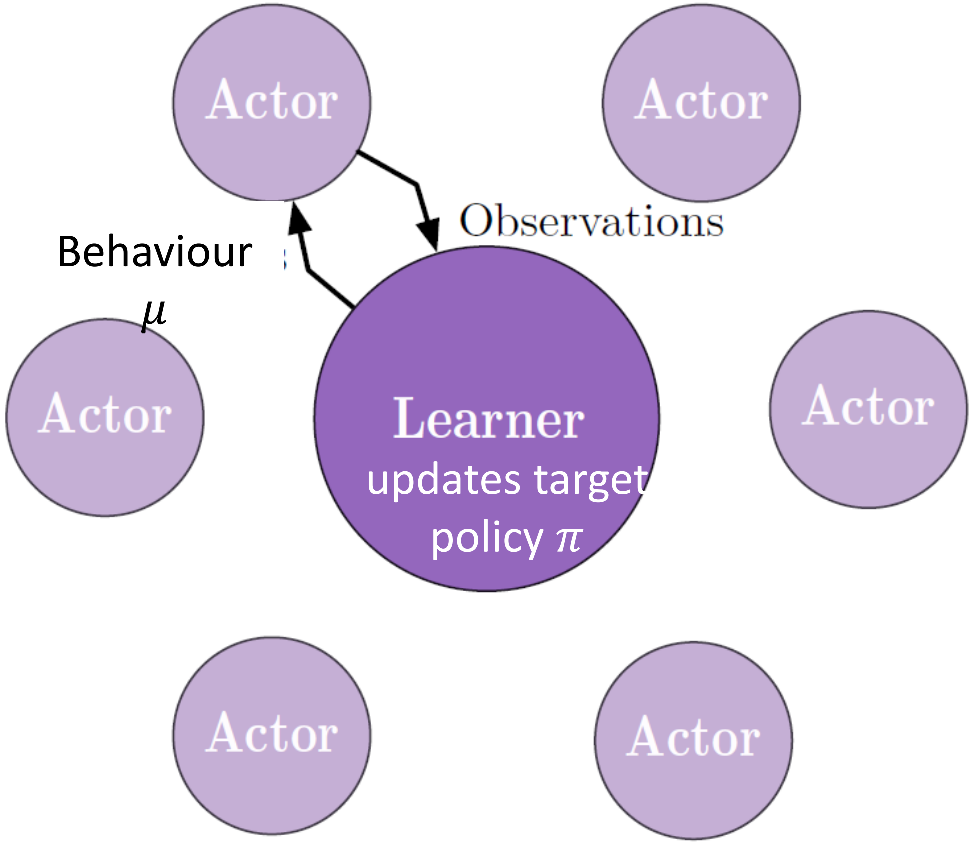
Off-Policy Correction and Batch Learning

Deep Reinforcement Seminar FS 2020, 03.03.2020

Xiang Li



On-policy algorithm $:=$ algorithm requiring $\mu = \pi$.

Why do we want an off-policy algorithm?

- Can choose a better μ
- Sample efficient

Actor-Critic Algorithm

- **Goal**: find policy $\pi: S\ x\ A\ \rightarrow [0,1]$ such that V^{π} is large
- **Algorithm**:
  - Repeat for $t = 1, \ldots$
    - Sample trajectories $\{(s_i, a_i, r_i, s_i'): i \in I\} \subset S \times A \times \mathbb{R}\ \times S$, where $\forall\ i \in I : \boldsymbol{a_i} \sim \boldsymbol{\mu}(\boldsymbol{s_i})$
    - Improve the policy π
    - Estimate the value V^{π}

Actor-Critic Algorithm: Details

„Policy Improvement“

Improve the policy $\boldsymbol{\pi_w}$

Find w such that $V^{\pi_w}(s)$ is large

- Using gradient ascent:
  - $w \leftarrow w + \eta \nabla_w V_\theta^{\pi_w}(s)$

„Policy Evaluation“

Estimate $\boldsymbol{V^{\pi_w}}$

Find θ such that $V^{\pi} \approx V_\theta^{\pi}$

- i.e. $\min_\theta \, [V_\theta^{\pi}(s) - V^{\pi}(s)]^2$
- Using gradient descent:
  - $\theta \leftarrow \theta + \eta' \, [V_\theta^{\pi}(s) - y] \, \nabla_\theta V_\theta^{\pi}(s)$
  - y is an estimate of $V^{\pi}(s)$,

    e.g. $\mathrm{y} = r + \gamma V_\theta(s_{next})$

Policy Evaluation: How to estimate $V^{\pi}(s_0)$?

- **Given**: $s_0, a_0, r_0, s_1, \ldots, a_{n-1}, r_{n-1}, s_n;\ a_i \sim \mu(s_i)$
- Approach 1: $y \coloneqq r_0 + \gamma V(s_1)$ (Abbreviate $V \coloneqq V_{\theta}^{\pi}$)
- Approach 2: $y \coloneqq r_0 + \gamma\, r_1 + \cdots + \gamma^{n-1} r_{n-1} + \gamma^n V(s_{n+1})$

$$= V(s_0) + \sum_{k=0}^{n-1} \gamma^k \big(r_k + \gamma V(s_{k+1}) - V(s_k)\big)$$

- Approach 3: $\mathrm{y} \coloneqq V(s_0) + \sum_{k=0}^{n-1} \gamma^k \left(r_k + \gamma V(s_{k+1}) - V(s_k)\right) \underbrace{\prod_{j=0}^{k} \min\left(1, \frac{\pi(s_j, a_j)}{\mu(s_j, a_j)}\right)}_{\text{Weights}}$

Policy Improvement: How to estimate $\nabla_w V^{\pi_w}(s_0)$?

- $\nabla_w V^{\pi_w}(s_0) = E_{\pi_w}\left[Q^{\pi_w}(s,a)\nabla_w \ln\left(\pi_w(s,a)\right)\right]$

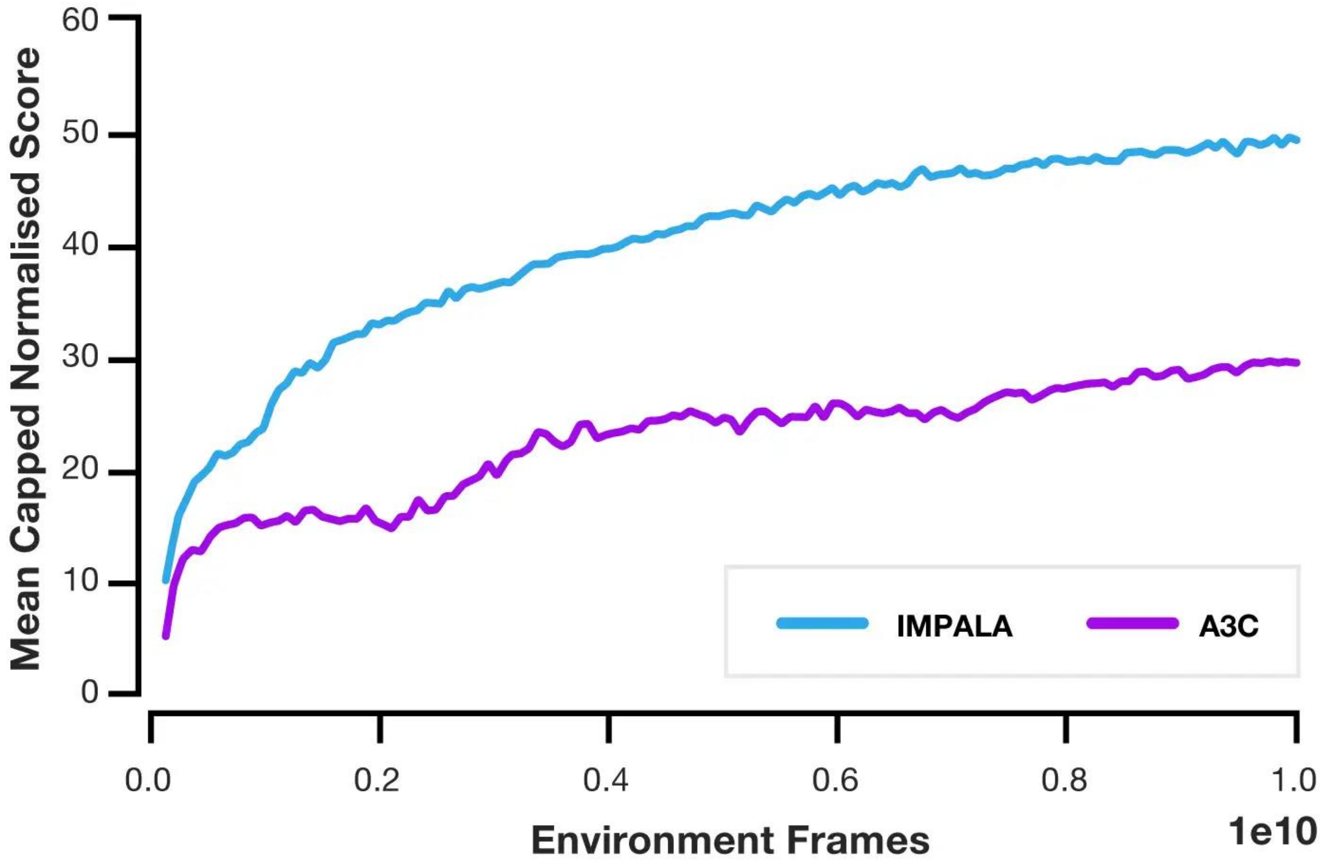
$$= E_\mu\left[Q^{\pi_w}(s,a)\nabla_w \ln\left(\pi_w(s,a)\right)\frac{\pi_w(s,a)}{\mu(s,a)}\right]$$

Effect of Off-Policy Correction

Performance on 5 DeepMind Lab tasks

	Task 1	Task 2	Task 3	Task 4	Task 5
Without Replay					
V-trace	46.8	32.9	**31.3**	**229.2**	**43.8**
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**
No-correction	35.0	21.1	2.8	85.0	11.2

Performance of Impala



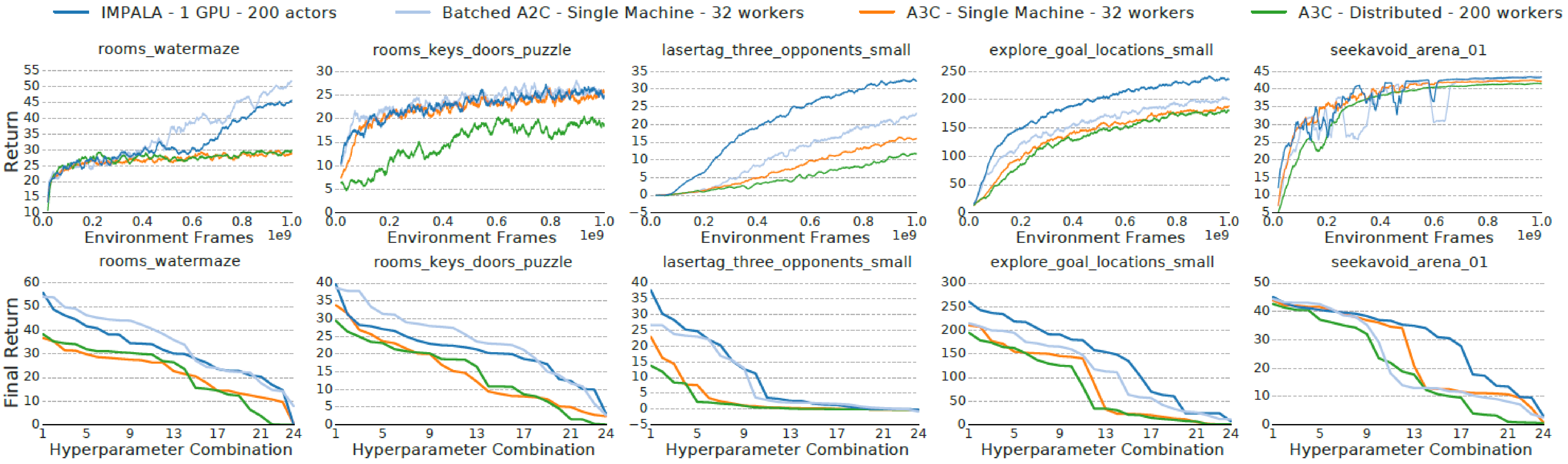


Figure 4. **Top Row:** Single task training on 5 DeepMind Lab tasks. Each curve is the mean of the best 3 runs based on final return. IMPALA achieves better performance than A3C. **Bottom Row:** Stability across hyperparameter combinations sorted by the final performance across different hyperparameter combinations. IMPALA is consistently more stable than A3C.

On/Off-policy & Offline/Online learning

- Task: find a target policy π using data D generated by behavioural policy μ
  - $\mathrm{D} \equiv \{(s_i, a_i, r_i, s_i'): i \in I\} \subset S \times A \times \mathbb{R} \times S$, where $\forall\, i \in I : \boldsymbol{a_i} \sim \boldsymbol{\mu}(\boldsymbol{s_i})$

- On-policy Algorithm $\coloneqq$ an algorithm working well only for $\mu = \pi$
- Off-policy Algorithm $\coloneqq$ an algorithm working well for all μ

- Online learning $\coloneqq$ able to choose a behavioural policy and interact with the environment
- Offline/batch learning $\coloneqq$ no interaction possible. μ generally not known.

The End