SIMPLIFYING MODEL-BASED RL: LEARNING REPRESENTATIONS, LATENT-SPACE MODELS, AND POLICIES WITH ONE OBJECTIVE

Raj Ghugare[1,2] **Homanga Bharadhwaj**[2] **Benjamin Eysenbach**[2]
Sergey Levine[3] **Ruslan Salakhutdinov**[2]

[1]VNIT Nagpur [2]Carnegie Mellon University [3]UC Berkeley

raj19@students.vnit.ac.in, hbharadh@cs.cmu.edu, beysenba@cs.cmu.edu

0 Preliminaries - Reinforcement learning

Find policy $\pi(a_t|s_t)$ that maximises:

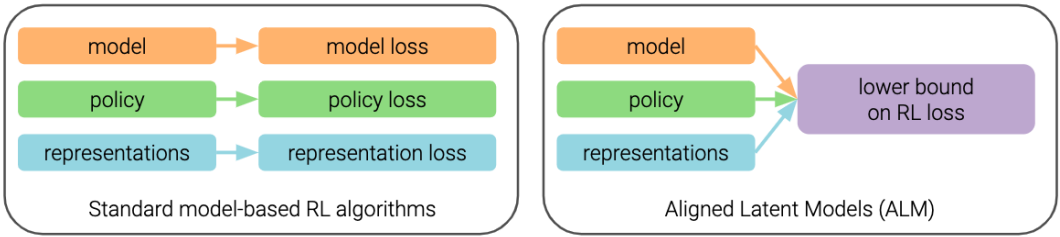
$$\max_{\pi} \mathbb{E}_{\underbrace{s_{t+1} \sim p(\cdot \mid s_t, a_t)}_{\text{environment}}, \underbrace{a_t \sim \pi(\cdot \mid s_t)}_{\text{policy}}} \left[\underbrace{(1-\gamma) \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)}_{R(\tau)} \right] \tag{1}$$

- States: s_t
- Actions: a_t
- Reward function: $r(s_t, a_t) \geq 0$

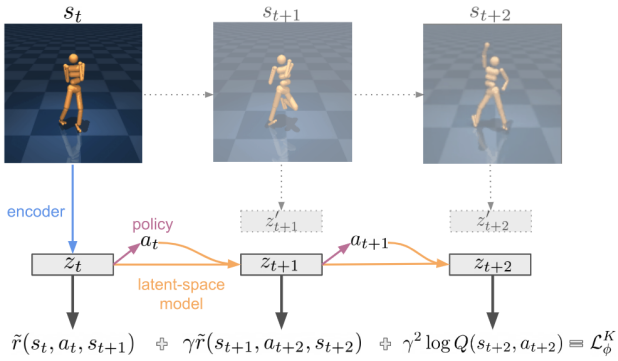
Find policy $\pi(a_t|s_t)$ that maximises:

$$\max_{\pi} \mathbb{E}_{\underbrace{s_{t+1} \sim p_\theta(\cdot \mid s_t, a_t)}_{\text{dynamics}}, \underbrace{a_t \sim \pi(\cdot \mid s_t)}_{\text{policy}}} \left[\underbrace{(1-\gamma) \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)}_{R(\tau)} \right] \tag{2}$$

- States: s_t
- Actions: a_t
- Reward function: $r(s_t, a_t) \geq 0$


model
model loss
policy
policy loss
representations
representation loss
Standard model-based RL algorithms
model
policy
representations
lower bound
on RL loss
Aligned Latent Models (ALM)

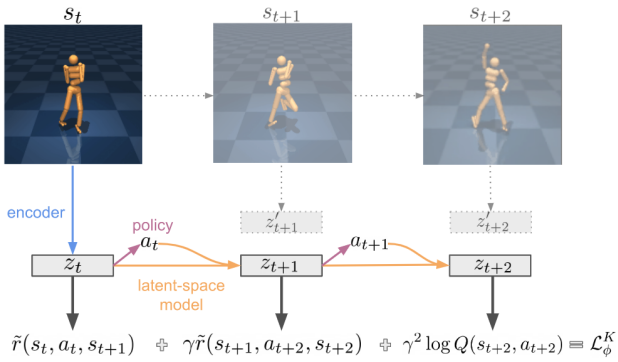
0 Method overview



- Encoder:

$$e_\phi(z_t \mid s_t) \qquad (3)$$

0 Method overview



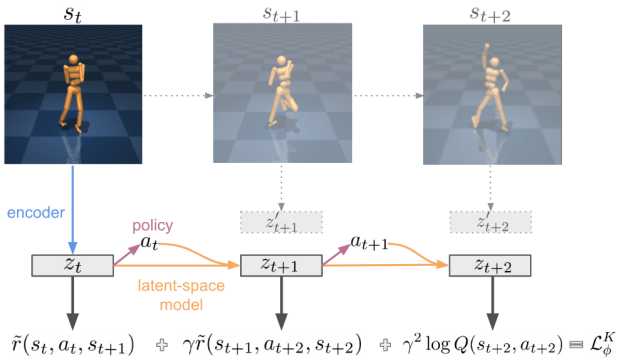
- Encoder:

$$e_\phi(z_t \mid s_t) \quad (3)$$

- Latent space dynamics:

$$m_\phi(z_t \mid z_{t-1}, a_{t-1}) \quad (4)$$

0 Method overview



- Encoder:

$$e_\phi(z_t \mid s_t) \quad (3)$$

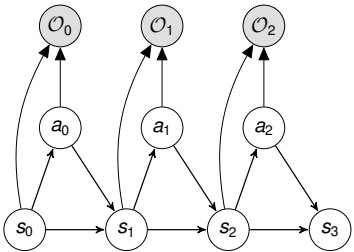
- Latent space dynamics:

$$m_\phi(z_t \mid z_{t-1}, a_{t-1}) \quad (4)$$

- Latent space policy:

$$\pi_\phi(a_t \mid z_t) \quad (5)$$

- Optimality variable $\mathcal{O}_t \in \{0, 1\}$, $(1 = \text{optimal})$[a]

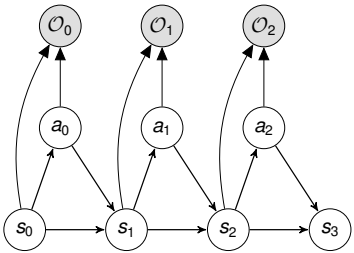


[a] [3] Toussaint. "Robot Trajectory Optimization Using Approximate Inference". 2009.

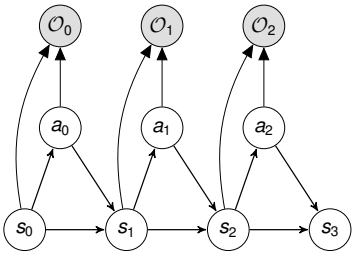
- Optimality variable $\mathcal{O}_t \in \{0, 1\}$, $(1 = \text{optimal})$[a]
- Likelihood:

$$p(\mathcal{O}_{0:\infty} = 1 \mid \tau) \propto R(\tau) \tag{6}$$

where $\tau \triangleq (s_0, a_0, z_0, s_1, a_1, z_1, \cdots)$



[a] [3] Toussaint. "Robot Trajectory Optimization Using Approximate Inference". 2009.



- Optimality variable $\mathcal{O}_t \in \{0, 1\}$, $(1 = \text{optimal})$[a]
- Likelihood:

$$p(\mathcal{O}_{0:\infty} = 1 \mid \tau) \propto R(\tau) \tag{6}$$

where $\tau \triangleq (s_0, a_0, z_0, s_1, a_1, z_1, \cdots)$

- Prior:

$$p_\phi(\tau) \triangleq p_0(s_0) \prod_{t=0}^{\infty} \underbrace{p(s_{t+1} \mid s_t, a_t)}_{\text{environment}} \underbrace{\pi_\phi(a_t \mid z_t)}_{\text{policy}} \underbrace{e_\phi(z_t \mid s_t)}_{\text{encoder}}$$

[a] [3] Toussaint. "Robot Trajectory Optimization Using Approximate Inference". 2009.

0 Distribution over trajectories

$$p_\phi(\tau) \triangleq p_0(s_0) \prod_{t=0}^{\infty} \underbrace{p(s_{t+1} \mid s_t, a_t)}_{\text{environment}} \underbrace{\pi_\phi(a_t \mid z_t)}_{\text{policy}} \underbrace{e_\phi(z_t \mid s_t)}_{\text{encoder}} \tag{7}$$

- Drawing samples from $p_\phi(\tau)$ requires interacting with the environment...

0 Distribution over trajectories

$$p_\phi(\tau) \triangleq p_0(s_0) \prod_{t=0}^{\infty} \underbrace{p(s_{t+1} \mid s_t, a_t)}_{\text{environment}} \underbrace{\pi_\phi(a_t \mid z_t)}_{\text{policy}} \underbrace{e_\phi(z_t \mid s_t)}_{\text{encoder}} \tag{7}$$

- Drawing samples from $p_\phi(\tau)$ requires interacting with the environment...
- Estimate $\mathbb{E}_{p(\tau)}[R(\tau)]$ with trajectories sampled from another distribution $q(\tau)$

0 Distribution over trajectories

$$p_\phi(\tau) \triangleq p_0(s_0) \prod_{t=0}^{\infty} \underbrace{p(s_{t+1} \mid s_t, a_t)}_{\text{environment}} \underbrace{\pi_\phi(a_t \mid z_t)}_{\text{policy}} \underbrace{e_\phi(z_t \mid s_t)}_{\text{encoder}} \tag{7}$$

- Drawing samples from $p_\phi(\tau)$ requires interacting with the environment...
- Estimate $\mathbb{E}_{p(\tau)}[R(\tau)]$ with trajectories sampled from another distribution $q(\tau)$
- Lower bound expected return using $q(\tau)$

$$\log \mathbb{E}_{p(\tau)}[R(\tau)] \geq \mathbb{E}_{q(\tau)}[\log R(\tau) + \log p(\tau) - \log q(\tau)] \tag{8}$$

0 Variational distribution

$$q_\phi^K(\tau) = p_0(s_0) \underbrace{e_\phi(z_0 \mid s_0)}_{\text{encoder}} \pi_\phi(a_0 \mid z_0) \prod_{t=1}^{K} \underbrace{p(s_t \mid s_{t-1}, a_{t-1})}_{\text{environment}} \underbrace{m_\phi(z_t \mid z_{t-1}, a_{t-1})}_{\text{latent space dynamics}} \underbrace{\pi_\phi(a_t \mid z_t)}_{\text{policy}}$$

$$\cdot \prod_{t=K+1}^{\infty} \underbrace{p(s_t \mid s_{t-1}, a_{t-1})}_{\text{environment}} \underbrace{m_\phi(z_t \mid z_{t-1}, a_{t-1})}_{\text{latent space dynamics}} \underbrace{\pi_\phi(a_t \mid z_t)}_{\text{policy}}$$

- Latent representations z_t are independent of states s_t for $t > 0$

0 Variational distribution

$$q_\phi^K(\tau) = p_0(s_0) \underbrace{e_\phi(z_0 \mid s_0)}_{\text{encoder}} \pi_\phi(a_0 \mid z_0) \prod_{t=1}^{K} \underbrace{p(s_t \mid s_{t-1}, a_{t-1})}_{\text{environment}} \underbrace{m_\phi(z_t \mid z_{t-1}, a_{t-1})}_{\text{latent space dynamics}} \underbrace{\pi_\phi(a_t \mid z_t)}_{\text{policy}}$$

$$\cdot \prod_{t=K+1}^{\infty} \underbrace{p(s_t \mid s_{t-1}, a_{t-1})}_{\text{environment}} \underbrace{m_\phi(z_t \mid z_{t-1}, a_{t-1})}_{\text{latent space dynamics}} \underbrace{\pi_\phi(a_t \mid z_t)}_{\text{policy}}$$

- Latent representations z_t are independent of states s_t for $t > 0$
- So can estimate bound for **any policy** using samples from latent-space model

0 Variational distribution

$$q_\phi^K(\tau) = p_0(s_0) \underbrace{e_\phi(z_0 \mid s_0)}_{\text{encoder}} \pi_\phi(a_0 \mid z_0) \prod_{t=1}^{K} \underbrace{p(s_t \mid s_{t-1}, a_{t-1})}_{\text{environment}} \underbrace{m_\phi(z_t \mid z_{t-1}, a_{t-1})}_{\text{latent space dynamics}} \underbrace{\pi_\phi(a_t \mid z_t)}_{\text{policy}}$$

$$\cdot \prod_{t=K+1}^{\infty} \underbrace{p(s_t \mid s_{t-1}, a_{t-1})}_{\text{environment}} \underbrace{m_\phi(z_t \mid z_{t-1}, a_{t-1})}_{\text{latent space dynamics}} \underbrace{\pi_\phi(a_t \mid z_t)}_{\text{policy}}$$

- Latent representations z_t are independent of states s_t for $t > 0$
- So can estimate bound for **any policy** using samples from latent-space model
  - Does not need access to high dimensional states!

0 Objective for model-based RL

$$\mathcal{L}_\phi^K \triangleq \mathbb{E}_{q_\phi^K(\tau)}\left[\left(\sum_{t=0}^{K-1}\gamma^t \tilde{r}\left(s_t, a_t, s_{t+1}\right)\right) + \gamma^K \log Q\left(s_K, a_K\right)\right]$$

where $\tilde{r}\left(s_t, a_t, s_{t+1}\right) = \underbrace{(1-\gamma)\log r\left(s_t, a_t\right)}_{\text{extrinsic term}} + \underbrace{\log e_\phi\left(z_{t+1} \mid s_{t+1}\right) - \log m_\phi\left(z_{t+1} \mid z_t, a_t\right)}_{\text{intrinsic}}$

- Bound holds for any:

0 Objective for model-based RL

$$\mathcal{L}_\phi^K \triangleq \mathbb{E}_{q_\phi^K(\tau)}\left[\left(\sum_{t=0}^{K-1}\gamma^t \tilde{r}\left(s_t, a_t, s_{t+1}\right)\right) + \gamma^K \log Q\left(s_K, a_K\right)\right]$$

$$\text{where} \quad \tilde{r}\left(s_t, a_t, s_{t+1}\right) = \underbrace{(1-\gamma)\log r\left(s_t, a_t\right)}_{\text{extrinsic term}} + \underbrace{\log e_\phi\left(z_{t+1} \mid s_{t+1}\right) - \log m_\phi\left(z_{t+1} \mid z_t, a_t\right)}_{\text{intrinsic}}$$

- Bound holds for any:
  - representation $e_\phi(z_t \mid s_t)$

0 Objective for model-based RL

$$\mathcal{L}_\phi^K \triangleq \mathbb{E}_{q_\phi^K(\tau)}\left[\left(\sum_{t=0}^{K-1}\gamma^t \tilde{r}\left(s_t, a_t, s_{t+1}\right)\right) + \gamma^K \log Q\left(s_K, a_K\right)\right]$$

$$\text{where} \quad \tilde{r}\left(s_t, a_t, s_{t+1}\right) = \underbrace{(1-\gamma)\log r\left(s_t, a_t\right)}_{\text{extrinsic term}} + \underbrace{\log e_\phi\left(z_{t+1} \mid s_{t+1}\right) - \log m_\phi\left(z_{t+1} \mid z_t, a_t\right)}_{\text{intrinsic}}$$

- Bound holds for any:
  - representation $e_\phi(z_t \mid s_t)$
  - latent-space model $m_\phi(z_{t+1} \mid z_t, a_t)$

0 Objective for model-based RL

$$\mathcal{L}_\phi^K \triangleq \mathbb{E}_{q_\phi^K(\tau)}\left[\left(\sum_{t=0}^{K-1}\gamma^t \tilde{r}\left(s_t, a_t, s_{t+1}\right)\right) + \gamma^K \log Q\left(s_K, a_K\right)\right]$$

$$\text{where} \quad \tilde{r}\left(s_t, a_t, s_{t+1}\right) = \underbrace{(1-\gamma)\log r\left(s_t, a_t\right)}_{\text{extrinsic term}} + \underbrace{\log e_\phi\left(z_{t+1} \mid s_{t+1}\right) - \log m_\phi\left(z_{t+1} \mid z_t, a_t\right)}_{\text{intrinsic}}$$

- Bound holds for any:
  - representation $e_\phi(z_t \mid s_t)$
  - latent-space model $m_\phi(z_{t+1} \mid z_t, a_t)$
  - policy $\pi_\phi(a_t \mid z_t)$

0 Practical algorithm - Aligned Latent Models

Algorithm 1 The ALM objective can be optimized with any RL algorithm. We present an implementation based on DDPG Lillicrap et al. (2015).

Initialize the encoder $e_\phi(z_t \mid s_t)$, model $m_\phi(z_{t+1} \mid z_t, a_t)$, policy $\pi_\phi(a_t \mid z_t)$, classifier $C_\theta(z_{t+1}, a_t, z_t)$, reward $r_\theta(z_t, a_t)$, Q-function $Q_\theta(z_t, a_t)$, replay buffer $\mathcal{B}$
for $n = 1, \cdots, N$ do **do**
    Select action $a_n = \pi_\phi(a_n \mid e_\phi(s_n)) + \mathcal{N}$ using the current policy and exploration noise $\mathcal{N}$.
    Execute action a_n and observe reward r_n and next state s_{n+1}.
    Store transition (s_n, a_n, r_n, s_{n+1}) in $\mathcal{B}$; sample length-K sequence $\{s_i, a_i, s_{i+1}\}_{i=t}^{t+K-1} \sim \mathcal{B}$
    Compute lower bound using off-policy actions: $\mathcal{L}^K_{e_\phi, m_\phi}(\{s_i, a_i, s_{i+1}\}_{i=t}^{t+K-1})$ ▷ 7
    Update encoder and model by gradient ascent on off-policy lower bound: $\mathcal{L}^K_{e_\phi, m_\phi}$
    Compute lower bound using on-policy actions: $\mathcal{L}^K_{\pi_\phi}((s_t))$ ▷ 8
    Update policy by gradient ascent on on-policy lower bound: $\mathcal{L}^K_{\pi_\phi}$
    Update classifier, Q-function and reward by gradient descent on: $\mathcal{L}_{C_\theta}, \mathcal{L}_{Q_\theta}, \mathcal{L}_{r_\theta}$ ▷ 11, 9, 10

0 Practical algorithm - Latent-space learning phase

$$\mathcal{L}^{K}_{e_\phi, m_\phi}\left(\{s_i, a_i, s_{i+1}\}_{i=t}^{t+K-1}\right) = \mathbb{E}_{\underbrace{e_\phi(z_{i=t} \mid s_t)}_{\text{encoder}}, \underbrace{m_\phi(z_{i>t} \mid z_{t:i-1}, a_{i-1})}_{\text{latent-space dynamics}}}\Bigg[\gamma^K \underbrace{Q_\theta\left(z_K, \pi\left(z_K\right)\right)}_{\textit{learned},\textit{latent}} + \sum_{i=t}^{t+K-1} \gamma^i \left(\underbrace{r_\theta\left(z_i, a_i\right)}_{\textit{learned},\textit{latent}} - \underbrace{\mathrm{KL}\left(e_{\phi_{\text{targ}}}\left(z_{i+1} \mid s_{i+1}\right) \| m_\phi\left(z_{i+1} \mid z_i, a_i\right)\right)}_{\text{latent-space consistency}} \right)\Bigg]$$

0 Practical algorithm - Planning phase

- Remeber the original objective $\mathcal{L}_\phi^K \triangleq \mathbb{E}_{q_\phi^K(\tau)}\left[\left(\sum_{t=0}^{K-1}\gamma^t \tilde{r}\left(s_t, a_t, s_{t+1}\right)\right) + \gamma^K \log Q\left(s_K, a_K\right)\right]$

  where $\tilde{r}\left(s_t, a_t, s_{t+1}\right) = \underbrace{(1-\gamma)\log r\left(s_t, a_t\right)}_{\text{extrinsic term}} + \underbrace{\log e_\phi\left(z_{t+1} \mid s_{t+1}\right) - \log m_\phi\left(z_{t+1} \mid z_t, a_t\right)}_{\text{intrinsic}}$

0 Practical algorithm - Planning phase

- Remeber the original objective $\mathcal{L}_\phi^K \triangleq \mathbb{E}_{q_\phi^K(\tau)}\left[\left(\sum_{t=0}^{K-1}\gamma^t \tilde{r}\left(s_t, a_t, s_{t+1}\right)\right) + \gamma^K \log Q\left(s_K, a_K\right)\right]$

  where $\tilde{r}\left(s_t, a_t, s_{t+1}\right) = \underbrace{(1-\gamma)\log r\left(s_t, a_t\right)}_{\text{extrinsic term}} + \underbrace{\log e_\phi\left(z_{t+1} \mid s_{t+1}\right) - \log m_\phi\left(z_{t+1} \mid z_t, a_t\right)}_{\text{intrinsic}}$

- Do not have access to s_{t+1} for on-policy actions, so can't calculate $e_\phi(z_{t+1} \mid s_{t+1})$

0 Practical algorithm - Planning phase

- Remeber the original objective $\mathcal{L}_\phi^K \triangleq \mathbb{E}_{q_\phi^K(\tau)}\left[\left(\sum_{t=0}^{K-1}\gamma^t \tilde{r}\left(s_t, a_t, s_{t+1}\right)\right) + \gamma^K \log Q\left(s_K, a_K\right)\right]$

  where $\tilde{r}\left(s_t, a_t, s_{t+1}\right) = \underbrace{(1-\gamma)\log r\left(s_t, a_t\right)}_{\text{extrinsic term}} + \underbrace{\log e_\phi\left(z_{t+1} \mid s_{t+1}\right) - \log m_\phi\left(z_{t+1} \mid z_t, a_t\right)}_{\text{intrinsic}}$
- Do not have access to s_{t+1} for on-policy actions, so can't calculate $e_\phi(z_{t+1} \mid s_{t+1})$
- Learn classifier $C_\theta(z_{t+1}, a_t, z_t)$ to differentiate z_{t+1} from $e_\phi(z_{t+1} \mid s_{t+1})$ and $m_\phi(z_{t+1} \mid s_t, a_t)$

  $\mathcal{L}_{\pi_\phi}^K\left(s_t\right) = \mathbb{E}_{q_\phi\left(z_{t:K}, a_{t:K} \mid s_t\right)}\left[\sum_{i=t}^{t+K-1}\gamma^t\left(r_\theta\left(z_i, a_i\right) + c \cdot \log \frac{C_\theta\left(z_{i+1}, a_i, z_i\right)}{1 - C_\theta\left(z_{i+1}, a_i, z_i\right)}\right) + \gamma^K Q_\theta\left(z_K, \pi\left(z_K\right)\right)\right]$

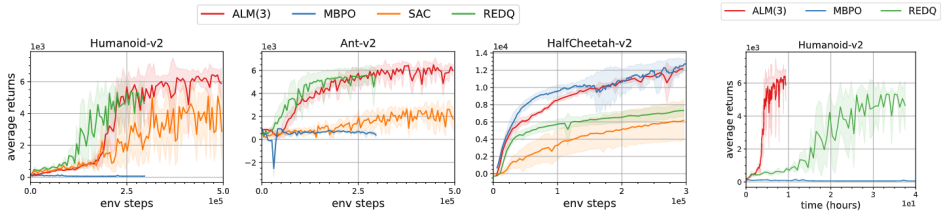
0 Experiments - Is the ALM objective useful?

Table 1: On the model-based benchmark from Wang et al. (2019), ALM outperforms model-based and model-free methods on $4/5$ tasks, often by a wide margin. We report mean and std. dev. across 5 random seeds. T-Humanoid-v2 and T-Ant-v2 refer to the respective truncated environments.

	T-Humanoid-v2	T-Ant-v2	HalfCheetah-v2	Walker2d-v2	Hopper-v2
ALM(3)	**5306** ± **437**	**4887** ± **1027**	**10789** ± **366**	**3006** ± **1183**	2546 ± 1074
SAC-SVG(2) (Amos et al., 2020)	501 ± 37	4473 ± 893	8752 ± 1785	448 ± 1139	**2852** ± **361**
SAC-SVG(3)	472 ± 85	3833 ± 1418	9220 ± 1431	878 ± 1533	2024 ± 1981
SVG(1) (Heess et al., 2015)	811.8 ± 241.5	185 ± 141	336 ± 387	252 ± 48	435 ± 163
SLBO (Luo et al., 2018)	1377 ± 150	200 ±40	1097 ± 166	207 ± 108	805 ± 142
TD3 (Fujimoto et al., 2018)	147 ± 0.7	870 ± 283	3016 ± 969	-516 ± 812	1817 ± 994
SAC (Haarnoja et al., 2018)	1470 ± 794	548 ± 146	3460 ± 1326	166 ± 1318	788 ± 977

- Policy return after $2e5$ environment steps

0 Experiments - Can ALM achieve good performance without ensembles



(a) Sample efficiency comparison

(b) Wall-clock comparison

Figure 3: **Good performance without ensembles.** Our method (ALM) can (*Left*) match the sample complexity of ensembling-based methods (MBPO, REDQ) while (*Right*) requiring less compute. ALM performs updates $\sim 10\times$ faster than MBPO. See Appendix Fig. 7 for results on other environments.

0 Experiments - Why does ALM work - Anaylzing the Q values

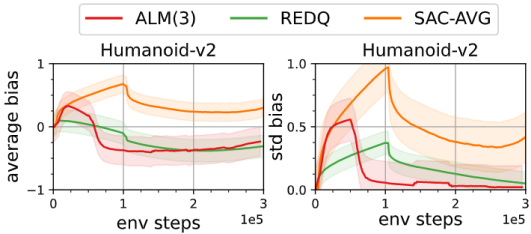
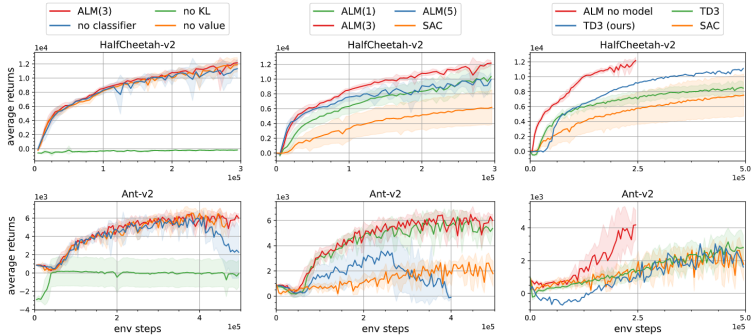


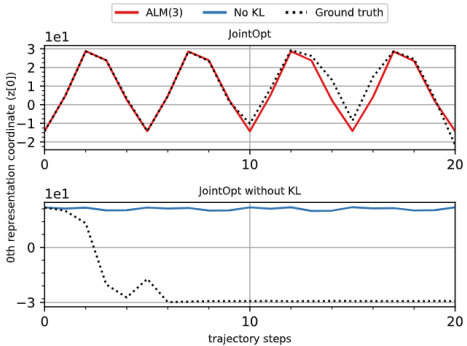
Figure 5: **Analyzing Q-values.** See text for details.

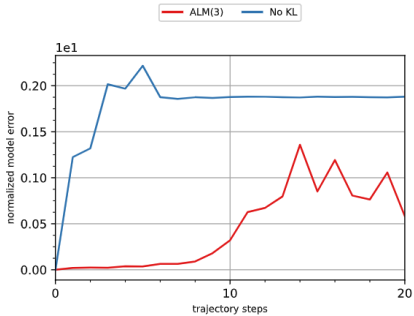
0 Experiments - Why does ALM work - Ablation experiments



(a) Terms in the objective. (b) Sequence length. (c) Model free RL.

0 Experiments - Why does ALM work - Analysing the learned representations





0 Summary

- One objective for learning representations, latent-space dynamics, and policy

[1] [2] Janner et al. “When to Trust Your Model: Model-Based Policy Optimization”. 2019.

[2] [1] Chen et al. “Randomized Ensembled Double Q-Learning: Learning Fast Without a Model”. 2021.

0 Summary

- One objective for learning representations, latent-space dynamics, and policy
- Similar sample efficieny to MBPO[1] and REDQ[2]

[1] [2] Janner et al. “When to Trust Your Model: Model-Based Policy Optimization”. 2019.

[2] [1] Chen et al. “Randomized Ensembled Double Q-Learning: Learning Fast Without a Model”. 2021.

0 Summary

- One objective for learning representations, latent-space dynamics, and policy
- Similar sample efficieny to MBPO[1] and REDQ[2]
  - No ensembles leads to ~6x less wall-clock time

[1] [2] Janner et al. “When to Trust Your Model: Model-Based Policy Optimization”. 2019.

[2] [1] Chen et al. “Randomized Ensembled Double Q-Learning: Learning Fast Without a Model”. 2021.

0 References

[1] Xinyue Chen, Che Wang, Zijian Zhou, and Keith Ross. “Randomized Ensembled Double Q-Learning: Learning Fast Without a Model”. In: *International Conference on Learning Representations*. arXiv, Mar. 2021. arXiv: 2101.05982 [cs].

[2] Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. “When to Trust Your Model: Model-Based Policy Optimization”. In: *Advances in Neural Information Processing Systems*. Vol. 32. 2019.

[3] Marc Toussaint. “Robot Trajectory Optimization Using Approximate Inference”. In: *International Conference on Machine Learning*. 2009.

0 Difference between theory and experiments

- They omit $\log r(s_t, a_t)$ from Equations 7 and 8

0 Difference between theory and experiments

- They omit $\log r(s_t, a_t)$ from Equations 7 and 8
  - because it decreased performance

0 Difference between theory and experiments

- They omit $\log r(s_t, a_t)$ from Equations 7 and 8
  - because it decreased performance
- They scale the KL term in policy objective (Equation 7)

0 Limitations

- More moving parts than model-free algorithms

My thoughts

0 Limitations

- More moving parts than model-free algorithms
  - Target network for e_ϕ

My thoughts

0 Limitations

- More moving parts than model-free algorithms
  - Target network for e_ϕ
  - Target network for Q_θ

My thoughts

0 Limitations

- More moving parts than model-free algorithms
  - Target network for e_ϕ
  - Target network for Q_θ
  - Classifer C_θ for estimating $\log e_\phi(z_{t+1} \mid s_{t+1}) - \log m_\phi(z_{t+1} \mid s_t, a_t)$

My thoughts

0 Limitations

- More moving parts than model-free algorithms
  - Target network for e_ϕ
  - Target network for Q_θ
  - Classifer C_θ for estimating $\log e_\phi(z_{t+1} \mid s_{t+1}) - \log m_\phi(z_{t+1} \mid s_t, a_t)$
- Gap between theoretical analysis and practical implementation (function approximation etc)

My thoughts

0 Limitations

- More moving parts than model-free algorithms
  - Target network for e_ϕ
  - Target network for Q_θ
  - Classifer C_θ for estimating $\log e_\phi(z_{t+1} \mid s_{t+1}) - \log m_\phi(z_{t+1} \mid s_t, a_t)$
- Gap between theoretical analysis and practical implementation (function approximation etc)

My thoughts

- Fig 4 - Still good performance when removing the latent-space dynamics????

0 Limitations

- More moving parts than model-free algorithms
  - Target network for e_ϕ
  - Target network for Q_θ
  - Classifer C_θ for estimating $\log e_\phi(z_{t+1} \mid s_{t+1}) - \log m_\phi(z_{t+1} \mid s_t, a_t)$
- Gap between theoretical analysis and practical implementation (function approximation etc)

My thoughts

- Fig 4 - Still good performance when removing the latent-space dynamics????
  - I'm a little confused about this as learning C_θ in Eq 7 requires access to m_θ?

0 Limitations

- More moving parts than model-free algorithms
  - Target network for e_ϕ
  - Target network for Q_θ
  - Classifer C_θ for estimating $\log e_\phi(z_{t+1} \mid s_{t+1}) - \log m_\phi(z_{t+1} \mid s_t, a_t)$
- Gap between theoretical analysis and practical implementation (function approximation etc)

My thoughts

- Fig 4 - Still good performance when removing the latent-space dynamics????
  - I'm a little confused about this as learning C_θ in Eq 7 requires access to m_θ?
- This method encodes exploration via expectation over latent trajectories $\{z_t\}$?

0 Limitations

- More moving parts than model-free algorithms
  - Target network for e_ϕ
  - Target network for Q_θ
  - Classifer C_θ for estimating $\log e_\phi(z_{t+1} \mid s_{t+1}) - \log m_\phi(z_{t+1} \mid s_t, a_t)$
- Gap between theoretical analysis and practical implementation (function approximation etc)

My thoughts

- Fig 4 - Still good performance when removing the latent-space dynamics????
  - I'm a little confused about this as learning C_θ in Eq 7 requires access to m_θ?
- This method encodes exploration via expectation over latent trajectories $\{z_t\}$?
  - But the distribution over z_t does not represent epistemic uncertainty...