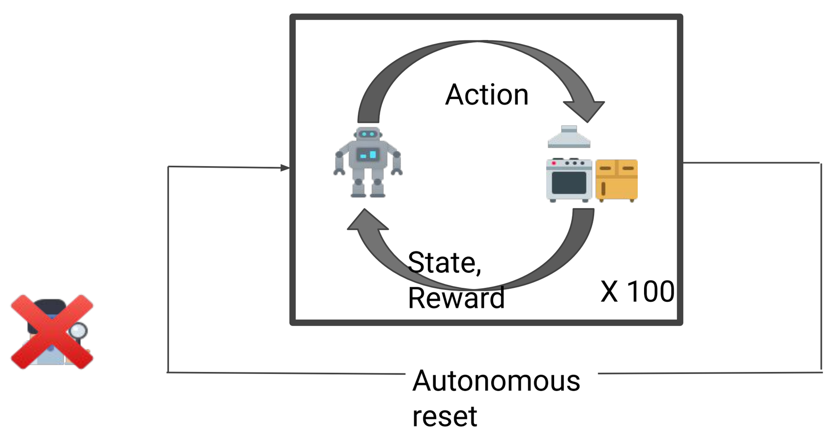
Model-Based Autonomous Reinforcement Learning

Sergio Charles with Archit Sharma, Leo Dong & Chelsea Finn
Department of Computer Science, Stanford University

Background & Related Work

Question: Embodied agents, e.g. humans and robots, function in a continual, non-episodic world. Why does the research community still develop RL algorithms in episodic settings?

- To build autonomous embodied agents, it is essential to learn continually without human interventions.
- Episodic learning requires humans to intervene after every episode, impeding autonomy & scale of learning systems.



Autonomous Resets: MEDAL

In addition to learning a forward policy π_f to solve the task, learn a backward policy π_b to stay close to the states in the demonstration distribution.

Forward policy objective

$$\max_{\pi_f} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$$

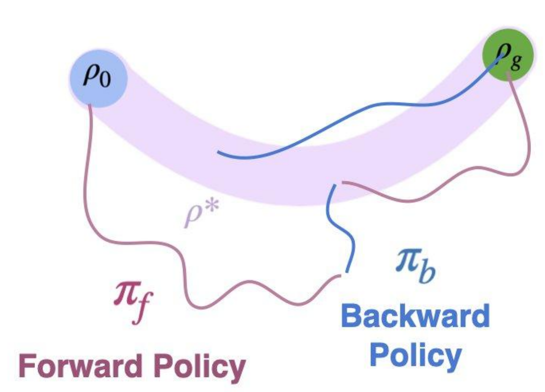
Backward policy objective

$$\min_{\pi_b} \mathcal{D}_{\mathrm{JS}}(\rho^b(s) || \rho^*(s))$$

- To match ρ^b and ρ^*, use a small set of demonstration to learn a state-space classifier $C: S \to [0,1]$.
- Generate states with the backward policy π_b by imitating demonstration states, hence solving the min-max problem:

$$\min_{\pi_b} \max_{C} \mathbb{E}_{s \sim \rho^*}[\log C(s)] + \mathbb{E}_{s \sim \rho^b}[\log(1 - C(s))]$$

Matching Expert Distributions for Autonomous Learning



Motivating a Model-Based Approach

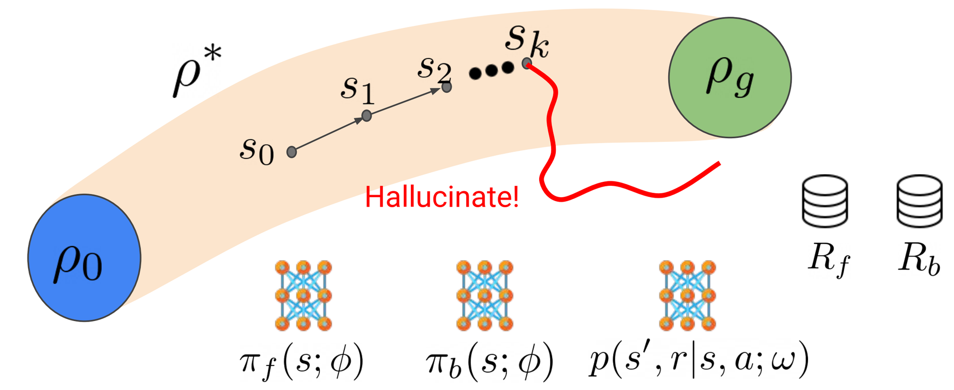
- Existing RL algorithms in the non-episodic setting focus on model-free methods (FBRL, R3L, VaPRL, MEDAL).
- Model-free methods learn a Q-value function $Q(s, a; \theta)$, e.g., via soft-actor critic, to optimize the forward and backward controllers: $\pi_f(a|s; \phi_f)$ and $\pi_b(a|s; \phi_b)$.
- Data sharing across forward & backward policies is non-trivial:
  - Methods like hindsight relabeling for goal-conditioned RL does not work because the two policies are too different.
  - Highly sample inefficient.

Objective

- Learn a **unified dynamics model** $p(s', r|s, a; \omega)$ to efficiently construct a forward policy $\pi_f(a|s; \phi_f)$ and backward policy $\pi_b(a|s; \phi_b)$ around the demonstration state distribution ρ^*.
- Use data collected by π_f and π_b to train the same dynamics model for sample efficiency.

MBARL Algorithm

Approach: Leverage online dynamics and policy learning by hallucinating data with a global dynamics model $p(s', r|s, a; \omega)$, combing MBPO and MEDAL.

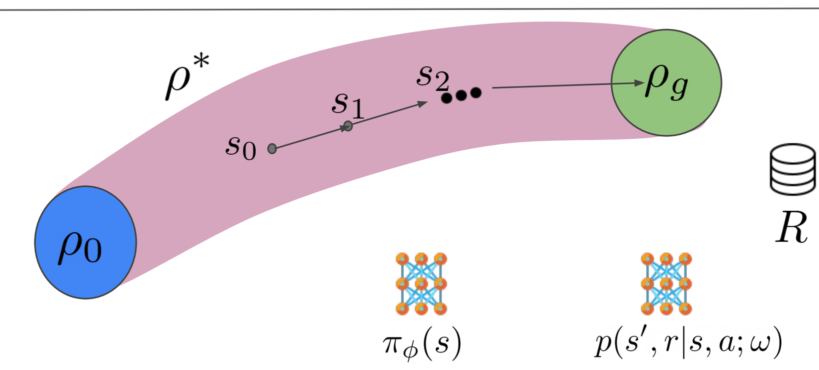


Algorithm 1 Model-based Autonomous Reinforcement Learning (MBARL)

require: forward demos D_f
optional: backward demos D_b
initialize:
  $R_f, \pi_f(a|s; \phi_f), Q^{\pi_f}(s, a; \theta_f)$ ▷forward policy
  $R_b, \pi_f(a|s; \phi_b), Q^{\pi_b}(s, a; \theta_b)$ ▷backward policy
  $p(s', r|s, a; \omega) \sim \mathcal{N}(f^{\mu}_{\omega}, f^{\sigma}_{\omega})$ ▷Gaussian dynamics model
  $C(s; \psi)$ ▷state-space classifier
  $R_f \leftarrow R_f \cup D_f, R_b \leftarrow R_b \cup D_b$ ▷forward backward replay buffers
$s \sim \rho_0$ ▷sample initial state
while not done **do**
▷ Run forward policy for fixed number of steps until goal is reached, otherwise run backward policy
  if forward **then**
    $a \sim \pi_f(\cdot|s; \phi_f)$
    $s' \sim p(\cdot|s, a), r \leftarrow r(s, a)$
    $R_f \leftarrow R_f \cup \{(s, a, s', r)\}$
    update π_f, Q^{π_f}
    update p_ω with (s,a,s') via maximum likelihood
  else
    $a \sim \pi_b(\cdot|s; \phi_b)$
    $s' \sim p(\cdot|s, a), r \leftarrow r(s, a)$
    $R_b \leftarrow R_b \cup \{(s, a, s', r)\}$
    update π_b, Q^{π_b}
    update p_ω with (s,a,s') via maximum likelihood

    ▷ Train discriminator every K steps
    if train-discriminator **then**
      sample positive states $S_p \sim D_f$
      sample negative states $S_n \sim D_b$
      update C on $S_p \cup S_n$
      ▷ Hallucination step using learned dynamics model
      for M model rollouts **do**
        for stage in (forward, backward) **do**
          policy $\pi = \pi_f$ if stage=forward else π_b
          replay buffer $R = R_f$ if stage=forward else R_b
          sample s_t uniformly from R
          perform k-step model rollout starting from s_t using policy π
          add sampled trajectory τ to R
        end for
      end for

Experiments: Offline Model-Based Approach



Goal: Given a set of oracle demonstrations, learn a dynamics model $p(s', r|s, a; \omega)$ such that we learn a good policy.

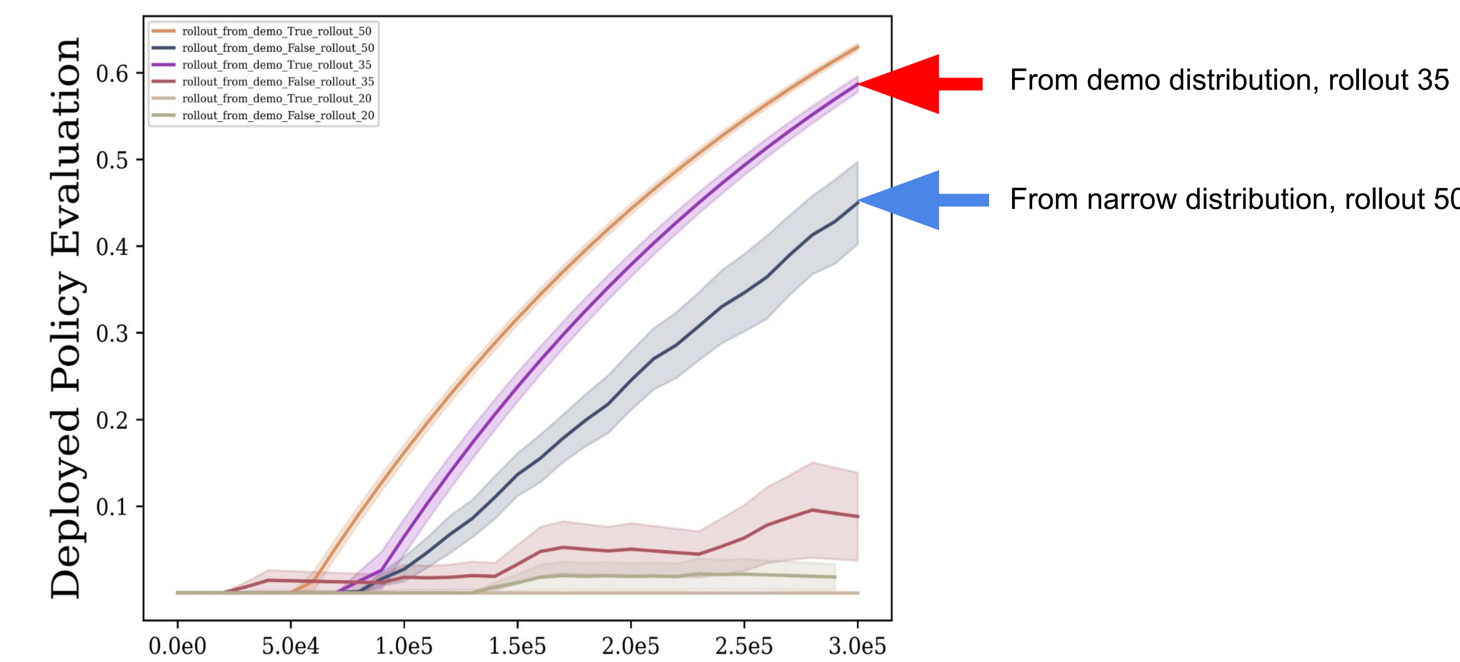
Approach: Offline learning by hallucinating data with dynamics model.

- (*) Collect uniformly sampled data in pointmass to train dynamics model
- **Model sizes**: {1k, 20k, 50k} training set
- **Rollout lengths:** {20, 35, 50}
- **Initial state:**
  - Narrow initial state distribution
  - Demonstration distribution

Results & Analysis

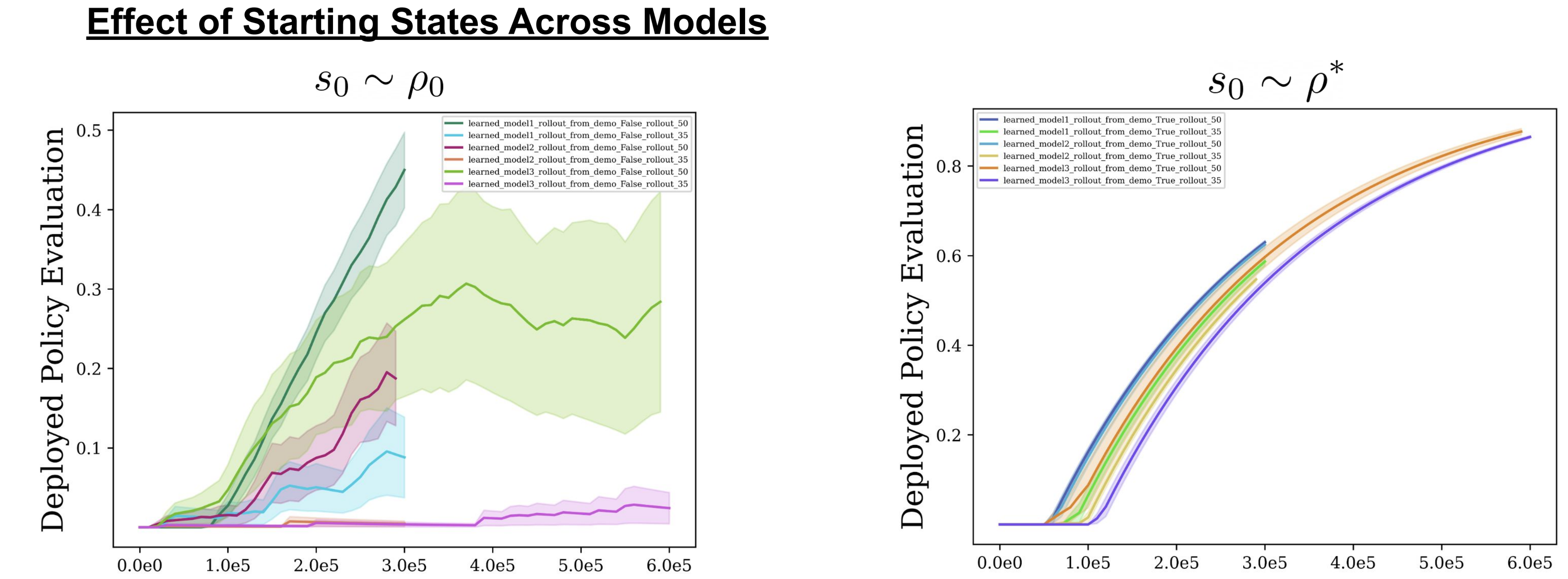
Effect of Rollout Length versus Initial State

Model 1 (50k)



- While longer rollout lengths certainly improve evaluation performance, we find that the way we choose to sample initial states for model hallucination is crucial.
- Sampling initial states from demonstrations significantly boosts performance.

Effect of Starting States Across Models



- When sampling from a narrow initial state distribution, the order of performance is as expected, i.e. model 1 performs the best and longer rollout lengths always outperform their shorter counterparts.
- ***Unexpected:*** when hallucinating from demonstration states, there is little to no difference in model performance regardless of training dataset size.

Decoupling Exploration & Exploitation

- Exploration policy $\pi_{\exp}$ matches the demonstration state distribution:

$$D_{\mathrm{KL}}(\rho^{\pi_{\exp}}(s) || \rho^*(s))$$

- with high entropy for *good coverage*:

$$\text{maximize } H(\rho^{\pi_{\exp}}(s))$$
$$\text{subject to } D_{\mathrm{KL}}(\rho^{\pi_{\exp}}(s) || \rho^*(s)) < \varepsilon$$

References

[1] Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. *When to trust your model: Model-based policy optimization*, 2019.

[2] Archit Sharma, Rehaan Ahmad, and Chelsea Finn. *A state-distribution matching approach to non-episodic reinforcement learning*, 2022.