Appendices

A Imitation Learning

Imitation learning (IL) algorithms [6] study how to learn a policy by mimicking expert experience demonstrations. Imitation learning has been combined with reinforcement learning, either by learning from demonstrations [43] [9] [44], or using deep reinforcement learning extensions [7] [45], or using variants policy gradient methods [46] [8]. Although this family of methods has proven its efficiency, it is still insufficient in the face of fully offline data sets. They either require interaction with the environment or need high-quality data, and these requirements are difficult to meet under offline settings, which makes the use of imitation learning from offline data impractical [17]. How to deal with the impact of noise is also an urgent area in imitation learning [47] [48]. Existing methods always have additional requirements on the quality of expert data. Gao et al. introduced an algorithm that learns from imperfect data, but it is not suitable for continuous control tasks. We borrow from the idea of imitation learning and introduce a generative model into our model, which gives our model the potential of rapid learning.

B Missing Proofs

Definition 1. We define estimation gap for policy π in state s as $\delta_{\text{MDP}}(s) = V^{\pi}(s) - V^{\pi}_{\mathcal{D}}(s)$.

Theorem 1. Given any policy π and state s, the error term $\delta_{\text{MDP}}(s)$ satisfies the following Bellman-like equation:

$$\begin{aligned}\delta_{\text{MDP}}(s) = &\sum_a \pi(a|s) \sum_{s',r} \left[p(s',r|s,a) - p_{\mathcal{D}}(s',r|s,a)\right] \left(r(s,a,s') + \gamma V^{\pi}_{\mathcal{D}}(s')\right) \\ &+ \gamma \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a) \delta_{\text{MDP}}(s')\end{aligned} \tag{11}$$

Proof. Through the definition of the V function, it can be proved by expanding this equation.

$$\begin{aligned}\delta_{\text{MDP}}(s) =& V^{\pi}(s) - V^{\pi}_{\mathcal{D}}(s) \\ =& \mathbb{E}[r(s,a,s') + V^{\pi}(s')] - V^{\pi}_{\mathcal{D}}(s) \\ =& \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r(s,a,s') + \gamma V^{\pi}(s')] \\ & - \sum_a \pi(a|s) \sum_{s',r} p_{\mathcal{D}}(s',r|s,a)[r(s,a,s') + \gamma V^{\pi}_{\mathcal{D}}(s')] \\ =& \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r(s,a,s') + \gamma(V^{\pi}_{\mathcal{D}}(s') + \delta_{\text{MDP}}(s'))] \\ & - \sum_a \pi(a|s) \sum_{s',r} p_{\mathcal{D}}(s',r|s,a)[r(s,a,s') + \gamma V^{\pi}_{\mathcal{D}}(s')] \\ =& \sum_a \pi(a|s) \sum_{s',r} \left[p(s',r|s,a) - p_{\mathcal{D}}(s',r|s,a)\right] r(s,a,s') \\ & + \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a) \gamma \delta_{\text{MDP}}(s') \\ & + \sum_a \pi(a|s) \sum_{s',r} \left[p(s',r|s,a) - p_{\mathcal{D}}(s',r|s,a)\right] V^{\pi}_{\mathcal{D}}(s') \\ =& \sum_a \pi(a|s) \sum_{s',r} \left[p(s',r|s,a) - p_{\mathcal{D}}(s',r|s,a)\right] \left(r(s,a,s') + \gamma V^{\pi}_{\mathcal{D}}(s')\right) \\ & + \gamma \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a) \delta_{\text{MDP}}(s')\end{aligned} \tag{12}$$

□

C Visualization of data distribution

Figure 2 shows the visualization of data generated by the halfcheetah-v2 environment, where the state space is 17-dim and action space is 6-dim. We concat the trajectory as a vector, and we reduce the trajectory with a dimension of 23k to a two-dimensional plane.

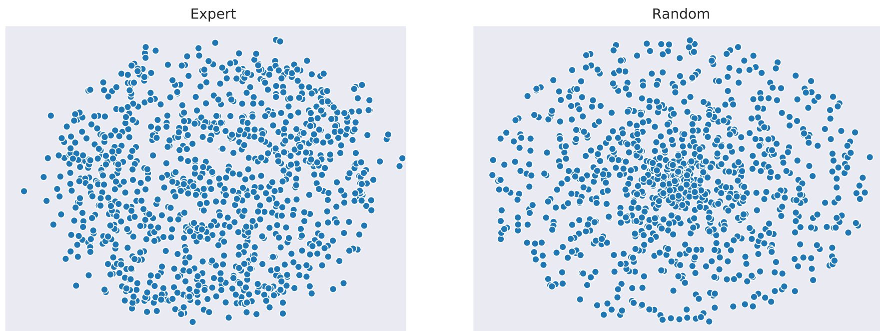

Figure 2: Visualization of data generated by the halfcheetah-v2 environment. Left: expert data. Right: random data.

D Algorithm

Algorithm 1 Pessimistic Offline Policy Optimization (POPO)

Require

- Data set $\mathcal{D}$, the size of mini-batch N, target network update rate η, motion correction coefficient ψ
- distortion risk measure β, random initialized networks and corresponding target networks, parameterized by $\theta_1' \leftarrow \theta_1, \theta_2' \leftarrow \theta_2, \phi' \leftarrow \phi$, VAE $G = \{E(\cdot, \cdot; \omega_1), D(\cdot, \cdot; \omega_2)\}$.

for iteration $= 1, 2, ...$ **do**

Sample mini-batch data (s, a, r, s') from data set $\mathcal{D}$.

\# Update VAE

$\mu, \sigma = E(s, a; \omega_1)$, $\hat{a} = D(s, z, ; \omega_2)$, $z \sim \mathcal{N}(\mu, \sigma)$

$\omega \leftarrow \arg\min_\omega \sum(\hat{a} - a)^2 + D_{\mathrm{KL}}(\mathcal{N}(\mu, \sigma) || \mathcal{N}(0, 1))$.

\# Update Z.

Set Z loss $\mathcal{L}(\cdots; \theta)$ (Equation 7).

$\theta \leftarrow \arg\min_\theta \mathcal{L}(\cdots; \theta)$.

\# Update actor

$\phi \leftarrow \arg\max_\phi Q_\beta(s, \hat{a} + \nu(s, \hat{a}; \phi); \theta)$

\# Update target networks

$\theta_i' \leftarrow \eta\theta_i + (1 - \eta)\theta_i'$, $\phi' \leftarrow \eta\phi + (1 - \eta)\phi'$

end for

E Experiments

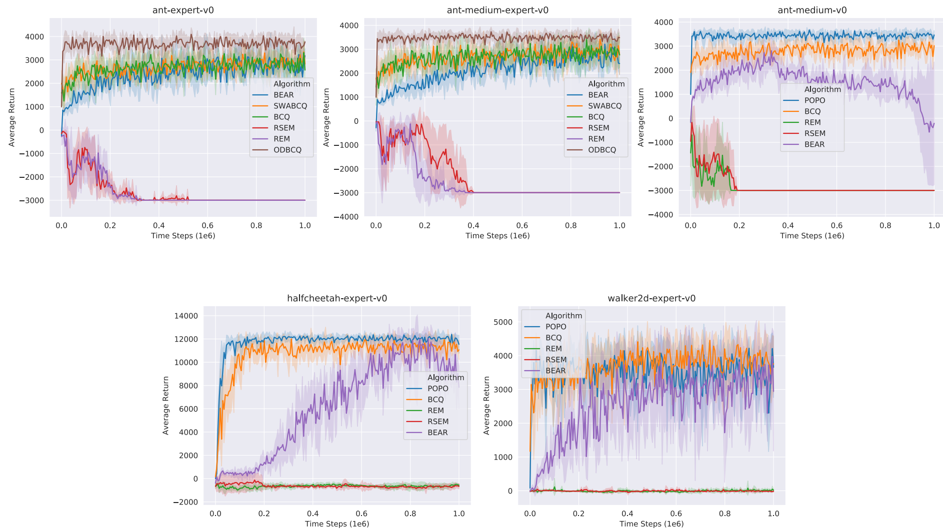

Figure 3: Performance curves for OpenAI gym continuous control tasks in MuJoCo suite. The shaded region represent a standard deviation of the average evaluation over five seeds.