Proceedings of Machine Learning Research vol 144:1–17, 2022

Example-Based Offline Reinforcement Learning without Rewards

Kyle Hatch[*] [1] KHATCH@STANFORD.EDU
Tianhe Yu[*] [1] TIANHEYU@CS.STANFORD.EDU
Rafael Rafailov [1] RAFAILOV@STANFORD.EDU
Chelsea Finn [1] CBFINN@CS.STANFORD.EDU
[1]*Department of Computer Science, Stanford University*

Abstract

Offline reinforcement learning (RL) methods, which tackle the problem of learning a policy from a static dataset, have shown promise in deploying RL in real-world scenarios. Offline RL allows the re-use and accumulation of large datasets while mitigating safety concerns that arise in online exploration. However, prior offline RL methods require human-defined reward labels to learn from offline datasets. Reward specification remains a major challenge for deep RL algorithms and also poses an issue for offline RL in the real world since designing reward functions could take considerable manual effort and also potentially requires installing extra hardware such as visual sensors on robots to detect the completion of a task. In contrast, in many settings, it is easier for users to provide examples of a completed task such as images than specifying a complex reward function. Based on this observation, we propose an algorithm that can learn behaviors from offline datasets without reward labels, instead using a small number of example images. Our method learns a conservative classifier that directly learns a Q-function from the offline dataset and the successful examples while penalizing the Q-values to prevent distributional shift. Through extensive empirical results, we find that our method outperforms prior imitation learning algorithms and inverse RL methods by 53% that directly learn rewards in vision-based robot manipulation domains

1. Introduction

While recent advances in deep reinforcement learning (RL) have shown promises in enabling robots to handle high dimensional image inputs in simulated domains (Kostrikov et al., 2020; Srinivas et al., 2020; Laskin et al., 2020; Hafner et al., 2020), these methods suffer from high sample complexity and potential safety concerns due to the need for online exploration in the environment and also require reward specification that requires manual effort and remains the major challenge in RL (Amodei et al., 2016; Everitt and Hutter, 2019; Rajeswaran et al., 2018). Offline RL (Lange et al., 2012; Levine et al., 2020), i.e. policy learning from a pre-collected dataset, has emerged as an effective alternative to the aforementioned online RL approaches. Since offline RL methods only leverage previous historical data, issues on safety and high sample complexity that arise due to online exploration are mitigated. Moreover, in offline RL, the agent is also allowed to reuse diverse prior data for better generalization. However, prior offline RL methods still require rewards of each transition in the offline dataset, which could make generating offline datasets impractical in many real-world scenarios. To address the challenge of reward specification in offline RL and make RL more applicable to real-world problems, we aim to develop a method that can effectively

[*] denotes equal contribution.

learn a policy from offline data while eschewing manually designed reward functions with minimal additional supervision.

To circumvent the challenge of reward specification, a large number of prior works have focused on learning from visual demonstrations given by either humans or robots, which can be a more natural way to teach robots to complete tasks than manually specifying the reward functions. Prior methods either use imitation learning or inverse RL to learn a policy that matches the expert behavior given by the demonstration of a fully trajectory (Pomerleau, 1988; Ross et al., 2011; Ho and Ermon, 2016; Finn et al., 2016; Fu et al., 2018a). Nevertheless, to combat the covariate shift that could lead the policy to drift away from the expert distribution, these methods require online interaction and are often sample inefficient. Recent advances in reward learning from examples (Fu et al., 2018b; Eysenbach et al., 2020, 2021) remove the requirement of having access to full trajectories of the expert demonstration. These approaches learn a reward function from successful examples and then optimize the learned reward with standard online RL algorithms. While these works mitigate the reward design challenges, they still require costly online interaction in order to learn both the reward function and the policy, limiting the applicability to real-world tasks. In our work, we aim to address addresses the challenges of online sample complexity and of reward specification simultaneously by developing an algorithm that learns visuomotor skills from offline datasets without explicit reward relabels nor access to the full optimal trajectories.

Our work considers a setting where we have access to both a large pre-recorded *unlabeled* offline dataset and a handful of successful examples given by users. To achieve effective reward learning and policy optimization in this unlabeled offline setting, we are faced with the well-known challenge of distribution shift between the learned policy and the behavior policy as identified in prior works (Fujimoto et al., 2018; Kumar et al., 2019, 2020). The learned policy could produce out-of-distribution actions and the critics would output arbitrary values on those unseen actions, leading to overestimation and divergence. Besides distribution shift between the learned policy and the behavior policy, reward learning introduces an additional error term between the learned rewards and the ground-truth rewards, which leads to even more erroneous value backups and is challenging to correct without online samples. To mitigate the the negative effect on offline value backups caused by both thedistributional shift between learn policy and the behavior policy and reward learning error, we propose a new offline example-based RL algorithm that that implicitly learns a reward via training a Q-function through recursive classification (Eysenbach et al., 2021) from offline data and successful outcomes. We present the Q-function as the likelihood ratio of a classifier that corresponds to the success of the task in the future and penalize the classifier prediction such that overestimation of the classifier on out-of-distribution actions would be alleviated. By training the conservative Q-function via recursive classification that predicts whether the task will be completed in the future, we bypass the separate reward learning phase and handle the distribution shift of both the policy and the reward function jointly via regularizing the Q-values. Therefore, the proposed method is able to learn the optimal behavior inferred from the successful examples in the fully offline setting without incurring excessive distributional shift.

The main contribution of this work is a new algorithm, Conservative Example-Based Offline RL (CEBORL) that is able to acquire visuomotor skills from a unlabeled offline dataset and a few hundred of images of successful outcomes. We characterize that EBORL optimizes a lower bound of the true Q-function given by the Bayes-optimal classifier in the non-function approximation setting. Through empirical evaluations, we find that EBORL outperforms prior imitation learning and example-based reward learning methods in simulated vision-based experiments by 53%.

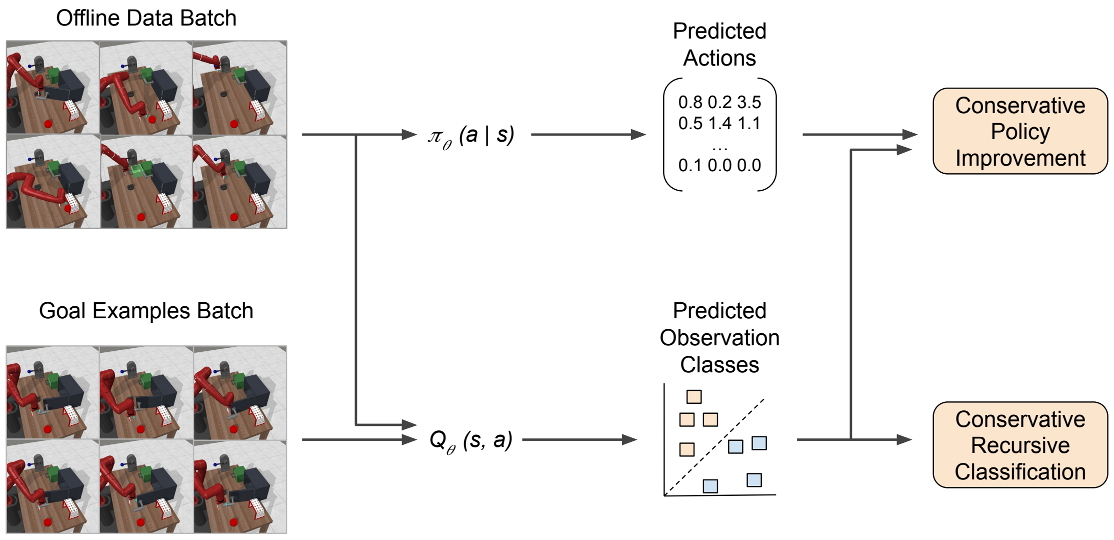

Figure 1: CEBORL Training Pipeline Overview. Recursive classification is used to train a critic network directly from offline data and goal examples. The critic network is then used to train the policy network as in SAC (Haarnoja et al., 2018). Learning of a conservative policy is enforced by adding a penalty to the Q-values of out-of-distribution actions.

2. Related Work

Reward learning. To overcome the challenge of hand-engineering reward functions, prior methods either use supervised learning or adversarial training to learn a policy that matches the expert behavior given by the demonstration (imitation learning) (Pomerleau, 1988; Ross et al., 2011; Ho and Ermon, 2016; Spencer et al., 2021) or learns a reward function from demonstration and optimize the policy with the learned reward through trial and error (inverse RL) (Ng and Russell, 2000; Abbeel and Ng, 2004; Ratliff et al., 2006; Ziebart et al., 2008; Finn et al., 2016; Fu et al., 2018a). Recent works have generalized the case of inverse RL beyond the requirement of full demonstrations to the setting where only a handful of user-specified outcomes, goals, or human videos are needed (Fu et al., 2018c; Singh et al., 2019; Kalashnikov et al., 2021; Xie et al., 2018; Eysenbach et al., 2021; Chen et al., 2021). These reward learning approaches have shown successes in real-world robotic manipulation tasks from high-dimensional image inputs (Finn et al., 2016; Singh et al., 2019; Zhu et al., 2020; Chen et al., 2021). Nevertheless, to combat covariate shift that could lead the policy to drift away from the expert distribution, these methods usually require significant online interaction. Unlike these works that study online settings, we consider learning visuomotor skills from offline datasets.

Offline RL. Offline RL (Ernst et al., 2005; Riedmiller, 2005; Lange et al., 2012; Levine et al., 2020) studies the problem of learning a policy from a static dataset without online data collection in the environment, which has shown promising results in robotic manipulation (Kalashnikov et al., 2018; Mandlekar et al., 2020; Rafailov et al., 2021; Singh et al., 2020; Julian et al., 2020; Kalashnikov et al., 2021). Prior offline RL methods focus on the challenge of policy distribution shift, by developing a variety of techniques such as regularization between the learned policy and the behavior policy of the dataset using direct policy constraints (Fujimoto et al., 2018; Liu et al., 2020; Jaques et al., 2019; Wu et al., 2019; Zhou et al., 2020; Kumar et al., 2019; Siegel et al., 2020; Peng et al., 2019; Fujimoto and Gu, 2021; Ghasemipour et al., 2021), learning conservative Q-functions (Kumar et al., 2020; Kostrikov et al., 2021; Yu et al., 2021; Sinha and Garg, 2021), and penalizing out-of-distribution states generated by learned dynamics models (Kidambi et al., 2020; Yu et al., 2020b; Matsushima et al., 2020; Argenson and Dulac-Arnold, 2020; Swazinna et al., 2020; Rafailov et al., 2021; Lee et al., 2021; Yu et al., 2021). While prior works combat overestimation caused by distribution shift, they typically require reward annotations of each datapoint in the large offline dataset. Unlike these methods, our approach considers the unlabeled offline setting and learns the reward using a minimal amount of user-specified success examples. While a few previous works have studied this setting, they either require access to the full demonstrations (Mandlekar et al., 2020; Zolna et al., 2020), do not address the issue of distributional shift (Ebert et al., 2018; Cabi et al., 2019), or require additional human supervision for reward annotation (Cabi et al., 2019). Unlike these approaches, our method addresses distributional shift induced by both the learned policy and the reward function in a principled way and only requires a few user-provided successful examples, resulting in an effective and practical offline reward learning scheme.

3. Preliminaries

Offline RL. Offline RL considers the standard Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, T, r, \gamma, \mu_0)$, where $\mathcal{S}$ and $\mathcal{A}$ represent the state and action spaces, r denotes the reward function, $T(\mathbf{s}'|\mathbf{s}, \mathbf{a})$ is dynamics transition function, $\gamma \in [0, 1]$ is the discount factor, and $\mu_0(\mathbf{s})$ is an initial state distribution. In offline RL, the agent cannot interact with the environment and instead has access to a fixed dataset $\mathcal{D} = \{(\mathbf{s}, \mathbf{a}, r, \mathbf{s}')\}$, which consists of transition tuples from trajectories collected using a behavior policy π_β. In other words, the dataset $\mathcal{D}$ is sampled from $d^{\pi_\beta}(\mathbf{s}, \mathbf{a}) := d^{\pi_\beta}(\mathbf{s})\pi_\beta(\mathbf{a}|\mathbf{s})$ where $d^{\pi_\beta}(\mathbf{s}) := (1-\gamma)\sum_{t=0}^{\infty} \gamma^t \mathcal{P}(\mathbf{s}_t = \mathbf{s}|\pi)$, where $\mathcal{P}(s_t = \mathbf{s}|\pi_\beta)$ is the probability of reaching state $\mathbf{s}$ at time t by executing π_β in $\mathcal{M}$. We define $\overline{\mathcal{M}}$ as the empirical MDP induced by the dataset $\mathcal{D}$ and $d(\mathbf{s}, \mathbf{a})$ as the sampled-based version of $d^{\pi_\beta}(\mathbf{s}, \mathbf{a})$. Therefore, the goal of offline RL is to learn a policy π that maximizes the return, or long term cumulative rewards in $\overline{\mathcal{M}}$: $\max_\pi J(\overline{\mathcal{M}}, \pi) := \frac{1}{1-\gamma}\mathbb{E}_{(\mathbf{s},\mathbf{a})\sim d(\mathbf{s},\mathbf{a})}[r(\mathbf{s}, \mathbf{a})]$.

Example-based RL. Example-based online RL (Eysenbach et al., 2021) offers a way to learn a policy without access to reward labels but instead with a set of success examples. Concretely, the agent is in a controlled Markov process $\mathcal{M}_E = (\mathcal{S}, \mathcal{A}, T, \gamma, \mu_0)$, i.e. MDP without rewards, and is given a set of *success examples*, $\mathcal{S}^* \sim p_U(\mathbf{s}_t|\mathbf{a}_t)$, where p_U is a user-specified distribution. A binary random variable $\mathbf{e}_t \in \{0, 1\}$ indicates the success at time t and $p(\mathbf{e}_t|\mathbf{s}_t)$ represents the probability of success at state $\mathbf{s}_t$. We further define the (discounted) probability of success at a future timestep under a policy $\pi(\cdot|\mathbf{s})$ as $p^\pi(\mathbf{e}_{t+}|\mathbf{s}_t, \mathbf{a}_t) := \mathbb{E}_{p^\pi(\mathbf{s}_{t+}|\mathbf{s}_t,\mathbf{a}_t)} p(\mathbf{e}_{t+}|\mathbf{s}_{t+})$ where $p^\pi(\mathbf{s}_{t+}|\mathbf{s}_t, \mathbf{a}_t) := (1-\gamma)\sum_{\Delta=0}^{\infty} p^\pi(\mathbf{s}_{t+\Delta} = \mathbf{s}_{t+}|\mathbf{s}_t, \mathbf{a}_t)$ and $\mathbf{s}_{t+\Delta}$. The objective of example-based RL is to optimize the policy that maximizes the likelihood of solving a task as follows:

$\arg\max_\pi p^\pi(\mathbf{e}_{t+}) = \mathbb{E}_{\mathbf{s}_0\sim\mu_0(\mathbf{s}_0),\mathbf{a}_0\sim\pi(\mathbf{a}_0|\mathbf{s}_0)} p^\pi(\mathbf{e}_{t+} = 1|\mathbf{s}_0, \mathbf{a}_0)$. With the access to $\mathcal{S}^*$, example-based RL solves the above MLE objective by using a classifier $C^\pi_\theta(\mathbf{s}_t, \mathbf{a}_t)$ to estimate the future success probability $p^\pi(\mathbf{e}_{t+}|\mathbf{s}_t, \mathbf{a}_t)$ where θ denotes the parameters of the classifier. $C^\pi_\theta(\mathbf{s}_t, \mathbf{a}_t)$ is trained using "positive" state-action pairs sampled from the conditional distribution $p^\pi(\mathbf{s}_t, \mathbf{a}_t|\mathbf{e}_{t+} = 1)$ and "negatives" sampled from the marginal state-action distribution in the off-policy replay buffer $p(\mathbf{s}_t, \mathbf{a}_t)$. Example-based RL uses a weight of $p(\mathbf{e}_{t+} = 1)$ for the "positives" and a weight of 1 for the "negatives". In this case, assuming the classifier is perfectly trained, the likelihood ratio of the Bayes-optimal solution is equal to the probability of seeing future success of the task:

$$\frac{C^\pi_\theta(\mathbf{s}_t, \mathbf{a}_t)}{1 - C^\pi_\theta(\mathbf{s}_t, \mathbf{a}_t)} = p^\pi(\mathbf{e}_{t+} = 1|\mathbf{s}_t, \mathbf{a}_t). \tag{1}$$

To learn the classifier C^π_θ, example-based RL optimizes θ using maximum likelihood:

$$\mathcal{L}^\pi(\theta) = p(\mathbf{e}_{t+} = 1)\mathbb{E}_{p^\pi(\mathbf{s}_t,\mathbf{a}_t|\mathbf{e}_{t+}=1)}\left[\log C^\pi_\theta(\mathbf{s}_t, \mathbf{a}_t)\right] + \mathbb{E}_{p(\mathbf{s}_t,\mathbf{a}_t)}\left[\log(1 - C^\pi_\theta(\mathbf{s}_t, \mathbf{a}_t))\right]. \tag{2}$$

Since the agent cannot directly sample from $p^\pi(\mathbf{s}_t, \mathbf{a}_t|\mathbf{e}_{t+} = 1)$, example-based RL uses recursive classification methods that resembles temporal difference learning, yielding the following objective:

$$\mathcal{L}^\pi(\theta) = (1-\gamma)\mathbb{E}_{\substack{\mathbf{s}_t\sim\mathcal{S}^* \\ \mathbf{a}_t\sim\pi(\cdot|\mathbf{s}_t)}}\left[\log C^\pi_\theta(\mathbf{s}_t, \mathbf{a}_t)\right] + \mathbb{E}_{p(\mathbf{s}_t,\mathbf{a}_t)}\left[\gamma\omega\log(C^\pi_\theta(\mathbf{s}_t, \mathbf{a}_t)) + \log(1 - C^\pi_\theta(\mathbf{s}_t, \mathbf{a}_t))\right],$$

where

$$\omega = \mathbb{E}_{\mathbf{a}_{t+1}\sim\pi(\mathbf{a}_{t+1}|\mathbf{s}_{t+1})}\left[\log\frac{C^\pi_\theta(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})}{1 - C^\pi_\theta(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})}\right] \tag{3}$$

represents the classifier's predicted probability ratio at the next time step. As shown in Lemma 4.1 in Eysenbach et al. (2021), optimizing Eq.**??** is equivalent to performing standard empirical Bellman backup for offline Q-learning with $r(\mathbf{s}_t, \mathbf{a}_t) = (1-\gamma)p(\mathbf{e}_t = 1|\mathbf{s}_t)$ and $Q^\pi_\theta(\mathbf{s}_t, \mathbf{a}_t) = \frac{C^\pi_\theta(\mathbf{s}_t,\mathbf{a}_t)}{1-C^\pi_\theta(\mathbf{s}_t,\mathbf{a}_t)}$.

While example-based RL enables learning without rewards using successful examples, it requires online interaction, which is sample inefficient and potentially unsafe. In our work, we extend it to the fully offline setting, which we will discuss in the next section.

4. Offline RL with Success Examples

The goal of our work is to devise an offline reinforcement learning algorithm that can enable effective policy learning from examples of success rather than carefully designed reward functions. While prior works in example-based RL achieve promising results, they require online data collection in order to ensure the learned classifier ratio is equal to the probability of the task being solved in the future in theory and are also sample inefficient and potentially dangerous in practical applications such as robotic control and autonomous driving. Therefore, to tackle such an issue, we consider the problem of offline RL with a large unlabeled offline dataset and a small number of examples of success. We define our problem setting as *example-based offline RL* and present our algorithm CEBORL that tackles this problem setting, which are discussed in detail in the following subsections.

4.1. Example-Based Offline RL

We formalize our problem setting in a MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, T, \gamma, \mu_0)$ without the reward function. As in the standard offline RL setting, the agent can only learn the policy from a static dataset $\mathcal{D} = \{(\mathbf{s}, \mathbf{a}, \mathbf{s}')\}$, which consists of transition tuples *without the reward relabels*. Additionally, the agent has access to a small set of *success examples* $\mathcal{S}^*$. Adapted from online example-based RL, the objective of offline example-based RL is maximizing the likelihood of solving a task w.r.t. the learned policy π as follows:

$$\arg\max_{\pi} p^{\pi}(\mathbf{e}_{t+}) = \mathbb{E}_{\mathbf{s}_1 \sim \mathcal{D}, \mathbf{a}_1 \sim \pi(\mathbf{a}|\mathbf{s})} \left[p^{\pi}(\mathbf{e}_{t+} = 1 | \mathbf{s}_1, \mathbf{a}_1) \right]. \tag{4}$$

To optimize Eq. 4, similar to online example-based RL, we train $C_{\theta}^{\pi}(\mathbf{s}_t, \mathbf{a}_t)$ with the difference being that the "negatives" are sampled from the marginal distribution, $d^{\pi_\beta}(\mathbf{s}_t, \mathbf{a}_t)$. In this case, the Bayes-optimal solution is

$$C_{\theta}^{\pi}(\mathbf{s}_t, \mathbf{a}_t) = \frac{p^{\pi}(\mathbf{s}_t, \mathbf{a}_t | \mathbf{e}_{t+} = 1) p(\mathbf{e}_{t+} = 1)}{p^{\pi}(\mathbf{s}_t, \mathbf{a}_t | \mathbf{e}_{t+} = 1) p(\mathbf{e}_{t+} = 1) + d^{\pi_\beta}(\mathbf{s}_t, \mathbf{a}_t)}, \tag{5}$$

assuming the classifier is perfectly trained. In this case, the classifier's probability ratio to be equal to the probability of seeing future success of the task $p^{\pi}(\mathbf{e}_{t+} = 1 | \mathbf{s}_t, \mathbf{a}_t)$ multiplied with the density ratio between the policy π and the behavior policy π_β:

$$\begin{aligned}
\frac{C_{\theta}^{\pi}(\mathbf{s}_t, \mathbf{a}_t)}{1 - C_{\theta}^{\pi}(\mathbf{s}_t, \mathbf{a}_t)} &= \frac{p^{\pi}(\mathbf{s}_t, \mathbf{a}_t | \mathbf{e}_{t+} = 1) p(\mathbf{e}_{t+} = 1)}{d^{\beta}(\mathbf{s}_t, \mathbf{a}_t)} \\
&= \frac{p^{\pi}(\mathbf{s}_t, \mathbf{a}_t | \mathbf{e}_{t+} = 1) p(\mathbf{e}_{t+} = 1)}{p^{\pi}(\mathbf{s}_t, \mathbf{a}_t)} \frac{p^{\pi}(\mathbf{s}_t, \mathbf{a}_t)}{d^{\beta}(\mathbf{s}_t, \mathbf{a}_t)} \qquad (6) \\
&= p^{\pi}(\mathbf{e}_{t+} = 1 | \mathbf{s}_t, \mathbf{a}_t) \frac{\pi(\mathbf{a}_t | \mathbf{s}_t)}{\pi_\beta(\mathbf{a}_t | \mathbf{s}_t)} \qquad (7)
\end{aligned}$$

where the last equality follows from Bayes' Theorem and decomposing $p^{\pi}(\mathbf{s}_t, \mathbf{a}_t) = \pi(\mathbf{a}_t | \mathbf{s}_t) d^{\beta}(\mathbf{s}_t)$ in the offline setting. Therefore, in the offline setting where we use the offline dataset as the negatives for classifier training, the ratio of the classifier prediction does not exactly correspond to the $p^{\pi}(\mathbf{e}_{t+} = 1 | \mathbf{s}_t, \mathbf{a}_t)$ and is subject to the density ratio between π and π_β respectively. This implies that if the distributional shift between the π and π_β is large, then $\frac{C_{\theta}^{\pi}(\mathbf{s}_t, \mathbf{a}_t)}{1 - C_{\theta}^{\pi}(\mathbf{s}_t, \mathbf{a}_t)}$ will deviate from $p^{\pi}(\mathbf{e}_{t+} = 1 | \mathbf{s}_t, \mathbf{a}_t)$, limiting the efficacy of the algorithm.

Similar to the online setting, we optimize the parameter θ using maximum likelihood estimation to learn the classifier C_{θ}^{π}:

$$\mathcal{L}^{\pi}(\theta) = p(\mathbf{e}_{t+} = 1) \mathbb{E}_{p^{\pi}(\mathbf{s}_t, \mathbf{a}_t | \mathbf{e}_{t+} = 1)} \left[\log C_{\theta}^{\pi}(\mathbf{s}_t, \mathbf{a}_t) \right] + \mathbb{E}_{d(\mathbf{s}_t, \mathbf{a}_t)} \left[\log(1 - C_{\theta}^{\pi}(\mathbf{s}_t, \mathbf{a}_t)) \right], \tag{8}$$

which can be optimized tractably using recursive classification by replacing $p(\mathbf{s}_t, \mathbf{a}_t)$ with $d(\mathbf{s}_t, \mathbf{a}_t)$ in Eq. **??**. Therefore, we bypass the direct reward learning step by learning a Q-function with success examples instead. However, one potential challenge of this approach is that the distributional shift between the learned policy π and the behavior policy π_β would cause π to produce out-of-distribution actions $\mathbf{a}_{t+1}$ in Eq. **??** that lead to erroneous classifier prediction and thus over-estimated Q-values $\frac{C_{\theta}^{\pi}(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})}{1 - C_{\theta}^{\pi}(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})}$ given by the classifier prediction ratio. Moreover, according to Eq. 7, such distributional shift could also produce large density ratios between π and π_β, hence leading to inaccurate estimate of the true probability of future success. To resolve such an issue, we propose to perform conservative example-based Bellman backup in the following subsection.

4.2. Combating Distributional Shift with Conservative Exampled-Based Offline RL

As discussed in Section 4.1, distributional shift between the learned policy π and the behavior policy π_β could cause the classifier $C_\theta^\pi(\mathbf{s}_t, \mathbf{a}_t)$ to produce incorrect predictions. Erroneously predicted probability of success on unseen actions produced by the learned policy π would lead to over-estimation. To mitigate such vulnerability, we present our approach, CEBORL. We utilize the conservative Q-learning (CQL) (Kumar et al., 2020) algorithm that additionally penalizes the Q-values on out-of-distribution actions. In addition to performing standard Bellman backup updates for Q-learning, CQL minimizes the Q-values at states in the dataset for actions not observed in the dataset while maximizing the Q-values on state-action pairs that lie within the dataset. In our example-based offline RL setting, we use a variant of the CQL objective as follows:

$$\begin{aligned}\max_\theta p(\mathbf{e}_{t+}=1)\mathbb{E}_{p^\pi(\mathbf{s}_t,\mathbf{a}_t|\mathbf{e}_{t+}=1)}\left[\log C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)\right] + \mathbb{E}_{d(\mathbf{s}_t,\mathbf{a}_t)}\left[\log(1-C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t))\right] \\ -\beta\left(\mathbb{E}_{\mathbf{s}_t\sim\mathcal{D},\mathbf{a}_t\sim\pi(\cdot|\mathbf{s}_t)}\left[C_\theta(\mathbf{s}_t,\mathbf{a}_t)\right] - \mathbb{E}_{\mathbf{s}_t,\mathbf{a}_t\sim\mathcal{D}}\left[C_\theta(\mathbf{s}_t,\mathbf{a}_t)\right]\right),\end{aligned} \tag{9}$$

where β is a positive coeffcient that controls the amount of conservatism and the second line is essentially the maximum likelihood estimation objective defined in Eq. 8. The regularization as shown in the first line of Eq.9 pushes down probability predicted by the future success classifier on states sampled from the offline unlabeled dataset $\mathcal{D}$ and actions sampled from the learned policy that is likely to be unseen in the offline dataset while pushing up the predicted probability of the classifier on state-action pairs within the dataset.

We now characterize that the Bayes-optimal solution C_θ^π learned by CEBORL can produce the estimated Q-value $\frac{C_\theta^\pi}{1-C_\theta^\pi}$ that is lower-bounded by the probability of seeing future success of the task $p^\pi(\mathbf{e}_{t+}=1|\mathbf{s}_t,\mathbf{a}_t)$ under expectation of π in setting without function approximation. Now, we state our main result below. Proofs can be found in Appendix A.

Proposition 1 (CEBORL learns lower-bounded classifiers; Informal) *In the setting without function approximation, assume that $C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)$ is of full support. If β is sufficiently large, optimizing Eq. 9 yields* $\mathbb{E}_{\mathbf{a}_t\sim\pi(\cdot|\mathbf{s}_t)}\left[\frac{C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)}{1-C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)}\right] \le \mathbb{E}_{\mathbf{a}_t\sim\pi(\cdot|\mathbf{s}_t)}\left[p^\pi(\mathbf{e}_{t+}=1|\mathbf{s}_t,\mathbf{a}_t)\right] \forall \mathbf{s}_t\in\mathcal{D}$.

Therefore, CEBORL effectively learns the Bayes optimal solution of the classifier $C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)$ is lower-bounded by the probability of the target task being solved in the future under the current policy under expectation of the learned policy π in the tabular setting. Unlike the Bayes optimal solution of example-based RL without conservative constraint in Eq. 7 that can be arbitrarily bad due to the multiplier of policy density ratio $\frac{\pi(\mathbf{a}_t|\mathbf{s}_t)}{\pi_\beta(\mathbf{a}_t|\mathbf{s}_t)}$, CEBORL gets rid of the policy density ratio and ensures the lower-bound guarantee similar to prior works (Kumar et al., 2020). Following Theorem 3.6 in (Kumar et al., 2020), the policy π learned the lower-bounded Bayes optimal solution that corresponds to the Q-value in our setting enjoys the safe policy improvement guarantee that the policy return of π improves over π_β with high probability. Note that in our setting, we do assume the offline dataset $\mathcal{D}$ contains a reasonable number of negatives in order to learn a well-behaved classifier, i.e. the data quality of $\mathcal{D}$ shouldn't be close the expert level, which is a reasonable assumption in practice. We also show that in practice, CEBORL achieves superior performance than vanilla example-based offline RL in the function approximation setting shown in Section 5.

Algorithm 1 CEBORL: Conservative Example-Based Offline RL

```
Require: Offline dataset $\mathcal{D}$, success examples $\mathcal{S}^*$, initialized policy and classifier $\pi_\phi$ and $C^\pi_\theta$.
1: for $i = 1, 2, 3, \cdots$, do
2:     Sample a batch of transitions $(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}) \sim \mathcal{D}, \mathbf{a}_{t+1} \sim \pi_\phi(\mathbf{a}_{t+1}|\mathbf{s}_{t+1})$.
3:     Sample a batch of success examples $\mathbf{s}_t \sim \mathcal{S}^*, \mathbf{a}_t \sim \pi(\mathbf{a}_t|\mathbf{s}_t)$.
4:     Compute $\omega$ using Eq. 3.
5:     Perform conservative policy evaluation of $\pi^i_\phi$ by repeatedly optimizing Eq. 10 to obtain $\hat{C}^{\pi^i_\phi}_\theta$ using
       samples from $\mathcal{D}$ and $\mathcal{S}^*$.
6:     Conservatively improve the policy by solving Eq. 11 to obtain $\pi^{i+1}_\phi$.
7: end for
```

Practical Implementations. In practice, we optimize Equation 9 via recursive classification as follows:

$$\min_\theta \underbrace{\beta \left(\mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\mu(\cdot|\mathbf{s})}\left[C_\theta(\mathbf{s},\mathbf{a})\right] - \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}\left[C_\theta(\mathbf{s},\mathbf{a})\right]\right)}_{:=\text{(a)}} + \underbrace{(1-\gamma)\mathbb{E}_{\mathbf{s}\sim\mathcal{S}^*,\mathbf{a}\sim\pi(\mathbf{a}_t|\mathbf{s}_t)}\left[\text{CE}(C_\theta(\mathbf{s}_t,\mathbf{a}_t); y=1)\right]}_{:=\text{(b)}}$$
$$\underbrace{+ (1+\gamma\omega)\mathbb{E}_{\mathbf{s}_t,\mathbf{a}_t,\mathbf{s}_{t+1}\sim\mathcal{D}}\left[\text{CE}\left(C_\theta(\mathbf{s}_t,\mathbf{a}_t); y = \frac{\gamma\omega}{\gamma\omega+1}\right)\right]}_{:=\text{(c)}}, \tag{10}$$

where ω is defined in Eq. 3, $\mu(\cdot|\mathbf{s})$ in term (a) is a wide action distribution such as the uniform distribution over all possible actions as used in Kumar et al. (2020) and term (b) and (c) are directly derived from Eq.**??** with CE denotes the cross-entropy loss. Intuitively, the second and third line of Eq.10 is essentially performing exampled-based Bellman update as discussed in Section 4.1. After performing the **conservative policy evaluation** procedure in Eq.**??**, CEBORL performs the **policy improvement** as follows:

$$\pi \leftarrow \arg\max_{\pi'} \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\pi'(\cdot|\mathbf{s})}\left[C^\pi_\theta(\mathbf{s},\mathbf{a})\right]. \tag{11}$$

We present the pseudocode of CEBORL in Algorithm 1. More practical implementation details can be found in Appendix B.2.

5. Experiments

We conducted experiments to answer the following questions: (1) Can CEBORL learn to complete tasks from large, unstructured offline datasets without reward labels? (2) How does CEBORL compare to existing offline reward learning methods in high dimensional, image based domains? (3) How important are the reward learning and conservatism aspects of CEBORL to its overall performance?

In order the answer these questions, we conducted experiments on a large, multimodal dataset of metaworld (Yu et al., 2020a) environments with image observations. We compare our method against ORIL (Zolna et al., 2020), which is an existing offline reward learning method. Although ORIL was originally used in the imitation learning setting, where it learns a reward discriminator from whole trajectories, we used it in the example based learning setting, where it learns to discriminate between individual example vs non-example observations. To determine the effect of conservatism in CEBORL, we compare it with RCE (Eysenbach et al., 2021), which uses a similar

	Dial Turn	Door Open	Drawer Open	Lever Pull	Plate Slide	All
CEBORL (Ours)	0.75	0.93	0.0	0.56	0.93	0.63
RCE	0.28	0.37	0.0	0.01	0.0	0.13
ORIL	0.02	0.01	0.03	0.05	0.41	0.1
BC	0.11	0.0	0.01	0.01	0.0	0.03
d-CQL	0.0	0.0	0.0	0.74	0.99	0.35

Table 1: We show multimodal Meta-World results here. Numbers are in success rates. We compare CEBORL to several baseline methods: RCE, ORIL, behavioral cloning (BC), and d-CQL. RCE is an example based RL algorithm similar to our method, but that is not designed for use in offline settings. ORIL is a method in which a reward classifier is first trained from goal examples using the offline data and then used to label the offline data. A standard offline RL algorithm is then trained on the relabeled data. In our implementation, we used CQL as the offline RL algorithm to match our method. d-CQL is the normal CQL algorithm except that instead of using the actual reward labels, the average negative pixel distance between each image and a batch of goal images is used as the reward. Each method was trained for 500,000 thousand steps on the multimodal dataset. A policy checkpoint was saved every 5,000 training steps. Each method was trained on each task with three random seeds. Success rates for each task are computed by rolling out the trained policies of the final five policy checkpoints of each random seed for ten episodes each and then computing the number of episodes in which the task is solved.

reward learning approach to our method but is meant for online settings and does not penalize out of distribution actions during training. To examine the effects of reward example based learning on our method, we compare against a baseline that we call d-CQL. d-CQL is simply CQL (Kumar et al., 2020) except that instead of using the true rewards to train on, it uses the mean negative pixel distance between a batch of goal examples and the observations as the reward. d-CQL is intended to examine how well a method that penalizes out of distribution actions but that uses a naive reward learning scheme can perform. Additionally, we compare our method with behavioral cloning (BC).

5.1. Multimodal Metaworld Environment

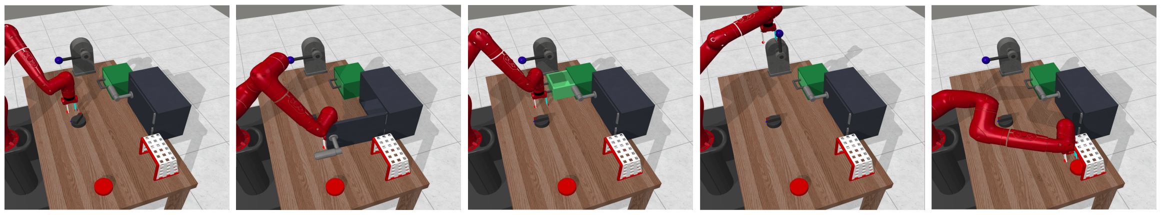

Figure 2: Multimodal Metaworld Environment. Five different Metaworld robotic manipulation tasks (from left to right: `sawyer-dial-turn`, `sawyer-door-open`, `sawyer-drawer-open`, `sawyer-lever-pull`, `sawyer-plate-slide`) are combined into a single environment by arranging each task onto a single table. A successful example is shown for each of the tasks.

To determine the ability of our method to learn on large unstructured offline data without reward labels, we constructed a multimodal environment based on the Metaworld framework. We combined five different Metaworld robotic manipulation tasks (`sawyer-dial-turn`, `sawyer-door-open`, `sawyer-drawer-open`, `sawyer-lever-pull`, `sawyer-plate-slide`) into a single environment by arranging each of the tasks' target objects onto a single table. On this environment, we collected an unstructured, multimodal offline dataset. The dataset contains trajectories from all five of the different tasks without any indicators in the observations that show which task was being performed when the trajectory was collected. We visualize all of the tasks in Figure 2.

Data was collected by training an SAC (Haarnoja et al., 2018) agent on each of the tasks and evaluating its policy at multiple training checkpoints. 1,500 trajectories of length 50 were collected for each task, amounting to 75,000 total timesteps. Within the data of a given task, $25\% - 35\%$ of the trajectories are successful, and so only $5\% - 7\%$ of the trajectories in the overall dataset are successful completions of a given task.

CEBORL achieves the best performance of any of the methods on three of the five tasks, and achieves the best overall performance by a large margin. This demonstrates that our method is able to learn to distinguish between the task shown in the goal examples it is provided with and the data from other tasks. It can then learn to perform the task in the offline data that matches the goal examples. ORIL achieves surprisingly poor performance on all of the tasks except for the plate slide task, perhaps indicating the advantage of directly training a Q-network from goal examples instead of training a separate reward discriminator network. BC achieves poor performance on all of the tasks, indicating that it is unable to determine which task contained within the multimodal dataset it is supposed to perform. RCE achieves non-trivial performance on some of the tasks, but is significantly outperformed by CEBORL on each task, demonstrating the importance of using conservatism in our method. d-CQL achieves high performance on two of the Plate Slide and Lever Pull tasks, but fails to achieve nonzero success rates on the other three tasks. This suggests that a combination of conservatism and naive reward learning can be sufficient for some tasks, but overall it is outperformed by more sophisticated reward learning methods.

6. Conclusion

In this paper, we present a new algorithm CEBORL that tackles the problem of example-based offline RL where we do not have access to the reward labels of the offline dataset and instead leverage a set of success examples. CEBORL builds upon prior works on learning a classifier whose Bayes optimal solution corresponds to the probability of the agent solving the task in the future and training the classifier using recursive classification. CEBORL addresses the issue of distributional shift between the learned policy and the behavior policy that leads to overestimation in the probability of future success via penalizing the classifier prediction on out-of-distribution actions. We characterize that CEBORL ensures the learned classifier produces a Bayes optimal solution that is a lower-bound of the probability of future success in expectation. In the empirical evaluations, CEBORL outperforms the vanilla example-based RL approach as well as prior offline imitation learning and goal reaching methods by a significant margin in both state-based and vision-based robotic manipulation domains with diverse offline datasets. Despite the advantages of CEBORL, CEBORL has a few challenges. For example, CEBORL requires a decent amount of failures in the offline dataset to ensure the classifier can be trained well with sufficient negative examples. CEBORL is also limited to either low-dimensional states or images as the success examples. We leave extensions of CEBORL to other forms of success examples such as languages as future work.

References

Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1, 2004.

Dario Amodei, Chris Olah, J. Steinhardt, Paul F. Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *ArXiv*, abs/1606.06565, 2016.

Arthur Argenson and Gabriel Dulac-Arnold. Model-based offline planning. *arXiv preprint arXiv:2008.05556*, 2020.

Serkan Cabi, Sergio Gómez Colmenarejo, Alexander Novikov, Ksenia Konyushkova, Scott Reed, Rae Jeong, Konrad Zolna, Yusuf Aytar, David Budden, Mel Vecerik, et al. Scaling data-driven robotics with reward sketching and batch reinforcement learning. *arXiv preprint arXiv:1909.12200*, 2019.

Annie S Chen, Suraj Nair, and Chelsea Finn. Learning generalizable robotic reward functions from" in-the-wild" human videos. *arXiv preprint arXiv:2103.16817*, 2021.

Frederik Ebert, Chelsea Finn, Sudeep Dasari, Annie Xie, Alex Lee, and Sergey Levine. Visual foresight: Model-based deep reinforcement learning for vision-based robotic control. *arXiv preprint arXiv:1812.00568*, 2018.

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, 2005.

Tom Everitt and Marcus Hutter. Reward tampering problems and solutions in reinforcement learning: A causal influence diagram perspective. *ArXiv*, abs/1908.04734, 2019.

Benjamin Eysenbach, Ruslan Salakhutdinov, and Sergey Levine. C-learning: Learning to achieve goals via recursive classification. *arXiv preprint arXiv:2011.08909*, 2020.

Benjamin Eysenbach, Sergey Levine, and Ruslan Salakhutdinov. Replacing rewards with examples: Example-based policy search via recursive classification, 2021.

Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International conference on machine learning*, pages 49–58. PMLR, 2016.

Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. *International Conference on Learning Representations*, 2018a.

Justin Fu, Avi Singh, Dibya Ghosh, Larry Yang, and Sergey Levine. Variational inverse control with events: A general framework for data-driven reward definition. *Conference on Neural Information Processing Systems*, 2018b.

Justin Fu, Avi Singh, Dibya Ghosh, Larry Yang, and Sergey Levine. Variational inverse control with events: A general framework for data-driven reward definition. *arXiv preprint arXiv:1805.11686*, 2018c.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *arXiv preprint arXiv:2106.06860*, 2021.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. *arXiv preprint arXiv:1812.02900*, 2018.

Seyed Kamyar Seyed Ghasemipour, Dale Schuurmans, and Shixiang Shane Gu. Emaq: Expected-max q-learning operator for simple yet effective offline and online rl. In *International Conference on Machine Learning*, pages 3682–3691. PMLR, 2021.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *International Conference on Learning Representations*, 2020.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Conference on Neural Information Processing Systems*, 2016.

Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. *arXiv preprint arXiv:1907.00456*, 2019.

Ryan Julian, Benjamin Swanson, Gaurav S Sukhatme, Sergey Levine, Chelsea Finn, and Karol Hausman. Efficient adaptation for end-to-end vision-based robotic manipulation. *arXiv preprint arXiv:2004.10190*, 2020.

Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pages 651–673. PMLR, 2018.

Dmitry Kalashnikov, Jacob Varley, Yevgen Chebotar, Benjamin Swanson, Rico Jonschkowski, Chelsea Finn, Sergey Levine, and Karol Hausman. Mt-opt: Continuous multi-task robotic reinforcement learning at scale. *arXiv preprint arXiv:2104.08212*, 2021.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *arXiv preprint arXiv:2005.05951*, 2020.

Ilya Kostrikov, Denis Yarats, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *arXiv preprint arXiv:2004.13649*, 2020.

Ilya Kostrikov, Jonathan Tompson, Rob Fergus, and Ofir Nachum. Offline reinforcement learning with fisher divergence critic regularization. *arXiv preprint arXiv:2103.08050*, 2021.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, pages 11761–11771, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779*, 2020.

Sascha Lange, Thomas Gabel, and Martin A. Riedmiller. Batch reinforcement learning. In *Reinforcement Learning*, volume 12. Springer, 2012.

Michael Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. *arXiv preprint arXiv:2004.14990*, 2020.

Byung-Jun Lee, Jongmin Lee, and Kee-Eung Kim. Representation balancing offline model-based reinforcement learning. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=QpNz8r_Ri2Y`.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Yao Liu, Adith Swaminathan, Alekh Agarwal, and Emma Brunskill. Provably good batch reinforcement learning without great exploration. *arXiv preprint arXiv:2007.08202*, 2020.

Ajay Mandlekar, Fabio Ramos, Byron Boots, Silvio Savarese, Li Fei-Fei, Animesh Garg, and Dieter Fox. Iris: Implicit reinforcement without interaction at scale for learning control from offline robot manipulation data. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4414–4420. IEEE, 2020.

Tatsuya Matsushima, Hiroki Furuta, Yutaka Matsuo, Ofir Nachum, and Shixiang Gu. Deployment-efficient reinforcement learning via model-based offline optimization. *arXiv preprint arXiv:2006.03647*, 2020.

Andrew Y. Ng and Stuart J. Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, 2000.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Dean A Pomerleau. Alvinn: an autonomous land vehicle in a neural network. In *Proceedings of the 1st International Conference on Neural Information Processing Systems*, pages 305–313, 1988.

Rafael Rafailov, Tianhe Yu, A. Rajeswaran, and Chelsea Finn. Offline reinforcement learning from images with latent space models. *Learning for Decision Making and Control (L4DC)*, 2021.

Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations. In *Proceedings of Robotics: Science and Systems (RSS)*, 2018.

Nathan D. Ratliff, J. Andrew Bagnell, and Martin A. Zinkevich. Maximum margin planning. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, 2006.

Martin Riedmiller. Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method. In *European Conference on Machine Learning*, pages 317–328. Springer, 2005.

Stephane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. *AISTATS*, 2011.

Noah Y Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, Thomas Lampe, Roland Hafner, and Martin Riedmiller. Keep doing what worked: Behavioral modelling priors for offline reinforcement learning. *arXiv preprint arXiv:2002.08396*, 2020.

Avi Singh, Larry Yang, Kristian Hartikainen, Chelsea Finn, and Sergey Levine. End-to-end robotic reinforcement learning without reward engineering. *arXiv preprint arXiv:1904.07854*, 2019.

Avi Singh, Albert Yu, Jonathan Yang, Jesse Zhang, Aviral Kumar, and Sergey Levine. Cog: Connecting new skills to past experience with offline reinforcement learning. *arXiv preprint arXiv:2010.14500*, 2020.

Samarth Sinha and Animesh Garg. S4rl: Surprisingly simple self-supervision for offline reinforcement learning. *arXiv preprint arXiv:2103.06326*, 2021.

Jonathan Spencer, Sanjiban Choudhury, Arun Venkatraman, Brian Ziebart, and J. Andrew Bagnell. Feedback in imitation learning: The three regimes of covariate shift. *ArXiv Preprint*, 2021.

Aravind Srinivas, Michael Laskin, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. *arXiv preprint arXiv:2004.04136*, 2020.

Phillip Swazinna, Steffen Udluft, and Thomas Runkler. Overcoming model bias for robust offline deep reinforcement learning. *arXiv preprint arXiv:2008.05533*, 2020.

Ziyu Wang, Alexander Novikov, Konrad Żołna, Jost Tobias Springenberg, Scott Reed, Bobak Shahriari, Noah Siegel, Josh Merel, Caglar Gulcehre, Nicolas Heess, et al. Critic regularized regression. *arXiv preprint arXiv:2006.15134*, 2020.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Annie Xie, Avi Singh, Sergey Levine, and Chelsea Finn. Few-shot goal inference for visuomotor learning and planning. In *Conference on Robot Learning*, pages 40–52. PMLR, 2018.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, pages 1094–1100. PMLR, 2020a.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *arXiv preprint arXiv:2005.13239*, 2020b.

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization. *arXiv preprint arXiv:2102.08363*, 2021.

Wenxuan Zhou, Sujay Bajracharya, and David Held. Plas: Latent action space for offline reinforcement learning. *arXiv preprint arXiv:2011.07213*, 2020.

Henry Zhu, Justin Yu, Abhishek Gupta, Dhruv Shah, Kristian Hartikainen, Avi Singh, Vikash Kumar, and Sergey Levine. The ingredients of real-world robotic reinforcement learning. *arXiv preprint arXiv:2004.12570*, 2020.

Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.

Konrad Zolna, Alexander Novikov, Ksenia Konyushkova, Caglar Gulcehre, Ziyu Wang, Yusuf Aytar, Misha Denil, Nando de Freitas, and Scott Reed. Offline learning from demonstrations and unlabeled experience. *arXiv preprint arXiv:2011.13885*, 2020.

Appendix A. Proof of Proposition 1

Proposition 2 (Formal version of Proposition 1) *In the tabular setting, assume $C_\theta^\pi(\mathbf{s}_t, \mathbf{a}_t)$ is full support and $\frac{p^\pi(\mathbf{e}_{t+}=1|\mathbf{s}_t,\mathbf{a}_t)}{C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)} \leq B$ for some constant $B > 0$. If $\beta \geq B$, optimizing Eq. 9 yields $\mathbb{E}_{\mathbf{a}_t\sim\pi(\cdot|\mathbf{s}_t)}\left[\frac{C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)}{1-C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)}\right] \leq \mathbb{E}_{\mathbf{a}_t\sim\pi(\cdot|\mathbf{s}_t)}\left[p^\pi(\mathbf{e}_{t+}=1|\mathbf{s}_t,\mathbf{a}_t)\right] \forall \mathbf{s}_t \in \mathcal{D}$.*

Proof In the tabular setting, we can rewrite Eq. 9 as follows:

$$\max_\theta p(\mathbf{e}_{t+}=1)\sum_{\mathbf{s},\mathbf{a}} p^\pi(\mathbf{s}_t,\mathbf{a}_t|\mathbf{e}_{t+}=1)\left[\log C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)\right] + \sum_{\mathbf{s},\mathbf{a}} d^\beta(\mathbf{s}_t,\mathbf{a}_t)\left[\log(1-C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t))\right]$$
$$-\beta\left(\sum_{\mathbf{s},\mathbf{a}} d^\beta(\mathbf{s})\pi(\mathbf{a}_t|\mathbf{s}_t)\left[C_\theta(\mathbf{s}_t,\mathbf{a}_t)\right] - \sum_{\mathbf{s},\mathbf{a}} d^\beta(\mathbf{s},\mathbf{a})\left[C_\theta(\mathbf{s}_t,\mathbf{a}_t)\right]\right), \tag{12}$$

To optimize Eq. 10, since Eq. 12 is convex w.r.t. C, we take the derivative of Eq. 12 and set it to 0. Solving this pointwise gives as follows:

$$\frac{p(\mathbf{e}_{t+}=1)p^\pi(\mathbf{s}_t,\mathbf{a}_t|\mathbf{e}_{t+}=1)}{C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)} - \frac{d^\beta(\mathbf{s}_t,\mathbf{a}_t)}{1-C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)} - \beta\left[d^\beta(\mathbf{s})\pi(\mathbf{a}_t|\mathbf{s}_t) - d^\beta(\mathbf{s},\mathbf{a})\right] = 0 \tag{13}$$

$$\Longrightarrow \frac{p(\mathbf{e}_{t+}=1)p^\pi(\mathbf{s}_t,\mathbf{a}_t|\mathbf{e}_{t+}=1)}{d^\beta(\mathbf{s},\mathbf{a})C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)} - \frac{1}{1-C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)} = \beta\left[\frac{\pi(\mathbf{a}_t|\mathbf{s}_t)}{\pi_\beta(\mathbf{a}|\mathbf{s})} - 1\right] \tag{14}$$

$$\Longrightarrow \frac{C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)}{1-C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)} = \frac{p(\mathbf{e}_{t+}=1)p^\pi(\mathbf{s}_t,\mathbf{a}_t|\mathbf{e}_{t+}=1)}{d^\beta(\mathbf{s},\mathbf{a})} - \beta\left[\frac{\pi(\mathbf{a}_t|\mathbf{s}_t)}{\pi_\beta(\mathbf{a}|\mathbf{s})} - 1\right]C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t) \tag{15}$$

$$\Longrightarrow \frac{C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)}{1-C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)} = p^\pi(\mathbf{e}_{t+}=1|\mathbf{s}_t,\mathbf{a}_t)\frac{\pi(\mathbf{a}_t|\mathbf{s}_t)}{\pi_\beta(\mathbf{a}_t|\mathbf{s}_t)} - \beta\left[\frac{\pi(\mathbf{a}_t|\mathbf{s}_t)}{\pi_\beta(\mathbf{a}|\mathbf{s})} - 1\right]C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t) \tag{16}$$

$$\Longrightarrow \frac{C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)}{1-C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)} = p^\pi(\mathbf{e}_{t+}=1|\mathbf{s}_t,\mathbf{a}_t) - \left[\frac{\pi(\mathbf{a}_t|\mathbf{s}_t)}{\pi_\beta(\mathbf{a}|\mathbf{s})} - 1\right](\beta C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t) - p^\pi(\mathbf{e}_{t+}=1|\mathbf{s}_t,\mathbf{a}_t)), \tag{17}$$

where Eq. 16 follows from Eq. 7. Therefore, since $\beta \geq \frac{p^\pi(\mathbf{e}_{t+}=1|\mathbf{s}_t,\mathbf{a}_t)}{C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)}$, we denote $\delta = \beta C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t) - p^\pi(\mathbf{e}_{t+}=1|\mathbf{s}_t,\mathbf{a}_t) \geq 0$. We have

$$\mathbb{E}_{\mathbf{a}_t\sim\pi(\cdot|\mathbf{s}_t)}\left[\frac{C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)}{1-C_\theta^\pi(\mathbf{s}_t,\mathbf{a}_t)}\right] = \mathbb{E}_{\mathbf{a}_t\sim\pi(\cdot|\mathbf{s}_t)}\left[p^\pi(\mathbf{e}_{t+}=1|\mathbf{s}_t,\mathbf{a}_t)\right] - \sum_{\mathbf{a}}\left[\pi(\mathbf{a}_t|\mathbf{s}_t)\left(\frac{\pi(\mathbf{a}_t|\mathbf{s}_t)}{\pi_\beta(\mathbf{a}|\mathbf{s})} - 1\right)\right]\delta \tag{18}$$

$$\leq \mathbb{E}_{\mathbf{a}_t\sim\pi(\cdot|\mathbf{s}_t)}\left[p^\pi(\mathbf{e}_{t+}=1|\mathbf{s}_t,\mathbf{a}_t)\right] \tag{19}$$

where Eq. 18 follows from the fact that $\sum_{\mathbf{a}}\left[\pi(\mathbf{a}_t|\mathbf{s}_t)\left(\frac{\pi(\mathbf{a}_t|\mathbf{s}_t)}{\pi_\beta(\mathbf{a}|\mathbf{s})} - 1\right)\right] \geq 0$ as shown in Kumar et al. (2020). ■

Appendix B. Experimental Details

B.1. Details of Multimodal Metaworld Environment

We use 64x64 images, an action repeat of 4, and an episode length of 200. The dataset we gathered contains a total of 1,500 trajectories per task, with 7,500 trajectories total. 33% of the Dial Turn trajectories successfully complete the task, 31% of the Door Open trajectories successfully complete the task, 28% of the Drawer Open trajectories successfully complete the task, 35% of the Lever Pull trajectories successfully complete the task, and 33% of the Plate Slide trajectories successfully complete the task.

B.2. Hyperparameters for Reinforcement Learning

Our method integrates aspects of the Conservative Q-Learning (CQL) algorithm and example-based RL. The hyperparameters we used for our experiments are shown below:

- **Number of goal examples**: 200, same is in (Eysenbach et al., 2021)
- **Discount factor**: 0.99
- **Q-function learning rate**: $3e-4$
- **Policy learning rate**: $3e-4$
- **Number of Q-functions**: 2, where $min(Q_1, Q_2)$ is used for the Q-function backup and policy update
- **Batch size:** 256
- α **in CQL:** 0.1
- **Ratio of policy to Q-function updates:** 1:1
- **Number of samples used for estimating logsumexp:** 16
- **Target network update rate:** 0.005
- **Automatic entropy tuning:** False

We implemented our ORIL baseline using CQL as the offline reinforcement learning algorithm instead of Critic-Regulazed Regression (CRR) (Wang et al., 2020) as in the original paper to have a more direct comparison to our method. Additional hyperparameters for ORIL are shown below:

- **Reward classifier learning rate:** 3e-4
- **Number of reward classifier training steps:** 1000
- **Reward classifier batch size:** 256