

# Preventing Reward Hacking with Occupancy Measure Regularization

*Shivam Singhal  
Cassidy Laidlaw  
Anca Dragan*



Electrical Engineering and Computer Sciences  
University of California, Berkeley

Technical Report No. UCB/EECS-2024-125

<http://www2.eecs.berkeley.edu/Pubs/TechRpts/2024/EECS-2024-125.html>

May 17, 2024

Copyright © 2024, by the author(s).  
All rights reserved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

### Acknowledgement

I would like to give a heartfelt thank you to Cassidy Laidlaw for all of his amazing mentorship. He has always made me feel welcomed into the research community, and he has always valued my ideas and believed in me—I truly couldn't have asked for a better person to work with. I would also like to thank Smitha Milli for inspiring me to partake in research and for introducing me to the important field of AI safety as a CHAI intern. Additionally, I would like to acknowledge Professor Anca Dragan for all of her insightful guidance and my labmates at the InterACT lab for their friendship. Last but definitely not least, I would like to thank my family for all of their constant love and support. Words cannot describe how much they mean to me, and without them, I wouldn't be here.

---

# Preventing Reward Hacking with Occupancy Measure Regularization

Shivam Singhal

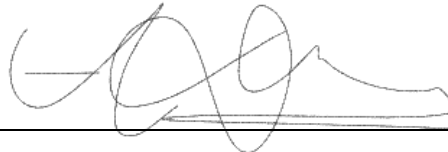
---

## Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences,  
University of California at Berkeley, in partial satisfaction of the requirements for the  
degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

### Committee:



---

Professor Anca Dragan  
Research Advisor

---

5/13/24

(Date)

\*\*\*\*\*



---

Professor Sergey Levine  
Second Reader

---

5/13/24

(Date)

---

# Preventing Reward Hacking with Occupancy Measure Regularization

---

Shivam Singhal<sup>\*1</sup> Cassidy Laidlaw<sup>\*1</sup> Anca Dragan<sup>1</sup>

## Abstract

Reward hacking occurs when an agent performs very well with respect to a “proxy” reward function (which may be hand-specified or learned), but poorly with respect to the unknown true reward. Since ensuring good alignment between the proxy and true reward is extremely difficult, one approach to prevent reward hacking is optimizing the proxy conservatively. Prior work has particularly focused on enforcing the learned policy to behave similarly to a “safe” policy by penalizing the KL divergence between their action distributions (AD). However, AD regularization doesn’t always work well since a small change in action distribution at a single state can lead to potentially calamitous outcomes, while large changes might not be indicative of any dangerous activity. Our insight is that when reward hacking, the agent visits drastically different states from those reached by the safe policy, causing large deviations in state *occupancy measure* (OM). Thus, we propose regularizing based on the OM divergence between policies instead of AD divergence to prevent reward hacking. We theoretically establish that OM regularization can more effectively avoid large drops in true reward. Then, we empirically demonstrate in a variety of realistic environments that OM divergence is superior to AD divergence for preventing reward hacking by regularizing towards a safe policy. Furthermore, we show that occupancy measure divergence can also regularize learned policies *away* from reward hacking behavior. Our code and data are available at <https://github.com/cassidylaidlaw/orpo>.

---

<sup>\*</sup>Equal contribution <sup>1</sup>Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA, USA. Correspondence to: Cassidy Laidlaw <cassidy.laidlaw@berkeley.edu>, Shivam Singhal <shivamsinghal@berkeley.edu>.

## 1. Introduction

A major challenge for the designers of goal-oriented AI systems is specifying a reward function that robustly captures their goals and values. Manually designing reward functions is difficult due to the ambiguities and complex variables underlying real-world scenarios (Ibarz et al., 2018). An alternative is to learn reward functions from human data (Sadigh et al., 2017; Jeon et al., 2020), but these often fail to generalize outside the distribution of behavior seen during training (McKinney et al., 2023; Tien et al., 2023). Thus, a learned or hand-specified reward function is often just a *proxy* for the true reward underlying the system designer’s intent. Misalignment between the two objectives can lead to *reward hacking*: a learned policy performs well according to the proxy reward function, but not according to the true reward function (Russell et al., 2010; Amodei et al., 2016; Pan et al., 2022; Skalse et al., 2022). A reward hacking policy’s behavior is often undesirable and can be especially catastrophic when deployed in safety-critical scenarios, such as autonomous driving (Krakovna et al., 2019; Turner et al., 2019; Knox et al., 2022). Unfortunately, reward hacking is a common phenomenon (Krakovna, 2018), which has problematic implications in the real world (Lum & Isaac, 2016; Corbett-Davies et al., 2017; Obermeyer et al., 2019; Milli et al., 2021; Pierson et al., 2021; Franchi et al., 2023; Kleinberg et al., 2023).

One method to prevent reward hacking is to avoid fully optimizing the proxy reward function by using constraints or regularization. In particular, prior work has regularized the chosen actions of a learning policy to be similar to those of a known safe policy (Yang et al., 2021). A *safe policy* is any policy that has reasonable (although potentially quite suboptimal) performance and does not reward hack; safe policies can be hard-coded or learned from human data. For example, RLHF for LLMs generally optimizes the learned reward in addition to a term that penalizes divergence from the pre-trained language model’s output (Glaese et al., 2022; Ouyang et al., 2022). Intuitively, this kind of regularization pushes the learned policy away from “unusual” behaviors for which the reward function may be misaligned.

The goal of optimizing a policy with regularization is to achieve higher true reward than the safe policy while avoiding reward hacking. To do so effectively, we must choose a

regularization regime that is simultaneously strong enough to prevent the learned policy from reward hacking, while also being sufficiently lenient to ensure the learned policy outperforms the safe policy. We argue that in many cases, regularizing based on the *action distributions* (AD) of policies makes it impossible to achieve this goal. This is because small shifts in action distribution can lead to large differences in outcomes, but large shifts in action distributions may not cause any difference in outcome. As an example, imagine an autonomous car driving alongside a steep cliff on a coastal highway. Suppose we have access to a safe policy that drives slowly and avoids falling off the cliff. However, the car is optimizing a proxy reward function that prioritizes quickly reaching the destination, but not necessarily staying on the road. If we try to regularize the car’s action distributions to the safe policy, we will need to apply heavy regularization, since only slightly increasing the probability of some unsafe action (e.g., making a sharp right turn) can lead to disaster. Since heavy regularization will prevent even minor deviations in action distribution, it would be near-impossible to improve upon the safe policy.

If action distribution divergences are poor regularizers for reward hacking, what can we do instead? In our car example, while a single catastrophic action doesn’t change the action distribution much, it does drastically change the *distribution over states* visited by the car. The learned policy will have a high probability of reaching states where the car is off the cliff and crashed, while the safe policy never reaches such states. Our proposal follows naturally from this observation: to avoid reward hacking, regularize based on divergence from the safe policy’s *occupancy measure*, rather than action distribution. A policy’s occupancy measure (OM) is the distribution of states or state-action pairs seen by a policy when it interacts with its environment. Unlike action distribution-based metrics, occupancy measures take into account the states that the agent reaches. While algorithms based on occupancy measures have been widely used for imitation learning (Ho & Ermon, 2016), offline RL (Lee et al., 2022), and efficient exploration (Hazan et al., 2019), using OM divergence to prevent reward hacking remains unexplored.

We show that OM-based regularization is superior to AD regularization for preventing reward hacking in both theory and practice. Theoretically, we show that there is a bound on the difference in return of two policies under *any* reward function based on the divergence between their occupancy measures. Thus, constraining the OM divergence from a safe policy can prevent the large drop in true reward associated with reward hacking, even when the true reward function is unknown. In contrast, only much weaker guarantees can be established for AD divergence.

Empirically, we derive an algorithm called **Occupancy-**

**Regularized Policy Optimization** (ORPO) that can be easily incorporated into deep RL algorithms like Proximal Policy Optimization (PPO) (Schulman et al., 2017). ORPO approximates the occupancy measure divergence between policies using a discriminator network. We use ORPO to optimize policies trained with misaligned proxy reward functions in multiple reward hacking benchmark environments (Pan et al., 2022) and compare it to AD regularization. The results of our experiments demonstrate that training with occupancy measure regularization leads to better performance under the unseen true reward function in all of the environments. In contrast, we find that it is difficult to tune AD regularization in some environments to both prevent reward hacking and allow meaningful improvement over the safe policy. To explain why this is the case, we show that, when compared with AD divergence, OM divergence from the safe policy is a much more accurate predictor of whether the learned policy is reward hacking. This validates our theoretical explanation that OM divergence is more indicative of the drop in the unknown true reward associated with reward hacking.

When a safe policy is unavailable, an alternative method to prevent reward hacking is to encourage a learned policy to have behavior that is as *different* from a reward hacking policy as possible. Our experiments show that optimizing for the proxy reward plus OM divergence from a reward hacking policy is also effective at avoiding reward hacking.

Our main contributions can be summarized as follows:

1. We show theoretically that occupancy measure regularization is superior to action distribution regularization for preventing reward hacking because constraining OM divergence effectively prevents large drops in the unknown true reward function.
2. We present the ORPO algorithm to implement OM regularization and show that it outperforms AD regularization in realistic environments.
3. We demonstrate that OM regularization can also be effectively used to regularize away from reward hacking.

## 2. Related work

While there have been separate lines of work investigating reward hacking and exploring the use of occupancy measures for other applications, to the best of our knowledge, we are the first to specifically study applying occupancy measures to the problem of reward hacking.

**Reward hacking:** Some prior works establish theoretical models of reward hacking as a special case of Goodhart’s Law (Goodhart, 1984; Leike et al., 2018; Krakovna, 2019; Skalse et al., 2022; Ngo et al., 2023). Krakovna (2018) provide a list of many examples of reward hacking. Pan et al. (2022) categorize types of reward misspecification and

relate optimization power to reward hacking.

**Safe reinforcement learning:** Regularizing policies to be similar to an offline policy based on their action distribution KL divergences was first proposed by Stiennon et al. (2020) and has since been widely employed in the context of optimizing LLMs using reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022; Bai et al., 2022; Glaese et al., 2022). KL regularization for RLHF has been further studied by Vieillard et al. (2021), Gao et al. (2022), and Korbak et al. (2022). Some alternative approaches to avoid reward hacking include quantizers (Taylor, 2016), “mild” optimization (Taylor et al., 2020), and impact regularization (Turner et al., 2020). While constrained RL can prevent the misbehavior of agents that optimize flawed reward functions (Dalal et al., 2018; Chow et al., 2019; Zhang et al., 2020; Roy et al., 2022), it simply shifts the difficulty of designing a reward function to specifying a set of constraints and weights. Other proposals to address the reward specification problem attempt to infer the true reward function based on the given proxy reward function, environment context, and/or feedback from humans (Hadfield-Menell et al., 2017; Reddy et al., 2020; Lee et al., 2021).

**Applications of occupancy measures:** Many offline RL algorithms use occupancy measure-based regularization to ensure that the learned policy remains within the training data distribution (Lee et al., 2022; Mandal et al., 2023; He, 2023; Cheng et al., 2022; Rashidinejad et al., 2023; Xie et al., 2023). Various types of distributional regularization are used in model-based RL as well since learned models may not generalize out-of-distribution (Yang et al., 2022). GAIL (Ho & Ermon, 2016) is an algorithm for robust imitation learning that aims to match the imitator’s occupancy measure to that of the demonstrator. Kang et al. (2018) combines GAIL with a reward function to efficiently explore using human data. Another line of work aims to find a policy with the highest-entropy occupancy measure for the purpose of exploring the state space (Hazan et al., 2019; Lee et al., 2020; Nedergaard & Cook, 2023).

**Our contribution:** Some of these previous works leverage occupancy measures and derive algorithms that are similar to our proposed ORPO algorithm. However, unlike previous work, we use OM-based regularization to *prevent reward hacking*, which to our knowledge is a novel application. We view our contribution as demonstrating that occupancy measure regularization is superior to action distribution regularization for this purpose. We do not explore the myriad ways that OM regularization could be incorporated into RL to prevent reward hacking. Instead, we focus our experiments on the simple and general ORPO algorithm. We leave to future work further investigation of alternate methods for preventing reward hacking with OM-based regularization.

### 3. Action Distribution vs. Occupancy Measure Regularization

We begin by theoretically and conceptually motivating why occupancy measure regularization should be superior to action distribution regularization for preventing reward hacking. We present our theoretical analysis in the setting of an infinite-horizon Markov decision process (MDP). An agent takes actions  $a \in \mathcal{A}$  to transition between states  $s \in \mathcal{S}$  over a series of timesteps  $t = 0, 1, 2, \dots$ . The first state  $s_0$  is sampled from an initial distribution  $\mu_0(s)$ , and when an agent takes action  $a_t$  in  $s_t$  at time  $t$ , the next state  $s_{t+1}$  is reached at timestep  $t + 1$  with transition probability  $p(s_{t+1} | s_t, a_t)$ . The agent aims to optimize a reward function  $R : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ , and rewards are accumulated over time with discount factor  $\gamma \in [0, 1)$ . A policy  $\pi$  maps each state  $s$  to a distribution over actions to take at that state  $\pi(a | s)$ . We define the (normalized) *return* of a policy  $\pi$  under a reward function  $R$  as

$$J(\pi, R) = (1 - \gamma) \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right]$$

where  $\mathbb{E}_\pi$  refers to the expectation under the distribution of states and actions induced by running the policy  $\pi$  in the environment. The normalizing factor  $1 - \gamma$  guarantees that  $J(\pi, R) \in [0, 1]$  always.

We define the *state-action occupancy measure*  $\mu_\pi$  of a policy  $\pi$  as the expected discounted number of times the agent will be in a particular state and take a specific action:

$$\mu_\pi(s, a) = (1 - \gamma) \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t \mathbb{1}\{s_t = s \wedge a_t = a\} \right].$$

Intuitively, the occupancy measure represents the distribution of states and actions visited by the policy over time.

The standard approach to solving an MDP is to find a policy  $\pi$  that maximizes its return:

$$\text{maximize } J(\pi, R). \quad (1)$$

However, as we discussed in section 1, an AI system designer most likely does not have access to a reward function that perfectly encapsulates their preferences. Instead, the designer might optimize  $\pi$  using a learned or hand-specified *proxy* reward function  $\tilde{R}$  which is misaligned with the *true* reward function  $R$ . Blindly maximizing the proxy reward function could lead to reward hacking.

**The drawbacks of action distribution regularization:** One approach to preventing reward hacking is to optimize the policy’s return with respect to the proxy  $\tilde{R}$  plus a regularization term that penalizes the KL divergence of the policy’s action distribution (AD) from a *safe policy*  $\pi_{\text{safe}}$ . This is equivalent to solving the following constrained optimization problem:

$$\begin{aligned} & \text{maximize } J(\pi, \tilde{R}) \quad \text{s.t.} \\ & (1 - \gamma) \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t D_{\text{KL}}(\pi(\cdot | s_t) \| \pi_{\text{safe}}(\cdot | s_t)) \right] \leq \epsilon. \end{aligned} \quad (2)$$



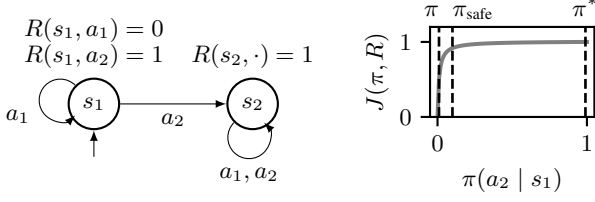


Figure 1. The MDP on the left, similar to that used in the proof of Proposition 3.1, demonstrates one drawback of using divergence between policies’ action distributions for regularization. The agent stays in state  $s_1$ , where it receives no reward, until it takes action  $a_2$ , after which it remains in state  $s_2$  forever and receives 1 reward per timestep. The plot on the right shows the return  $J(\pi, R)$  for a policy  $\pi$  when  $\gamma = 0.99$  as a function of the policy’s action distribution at  $s_1$ . While  $\pi$  and  $\pi_{\text{safe}}$  (shown on the plot as dotted lines) are close in action distribution space, they achieve very different returns. Meanwhile, the optimal policy  $\pi^*$  is far from  $\pi_{\text{safe}}$  in action distribution space. Propositions 3.2 and A.2 show that occupancy measure divergences do not have these drawbacks.

Intuitively, the aim of the AD constraint in (2) is to prevent the unusual behavior associated with reward hacking policies by constraining  $\pi$  to take similar actions to  $\pi_{\text{safe}}$ .

While AD regularization is simple and easy to implement, this method also has serious drawbacks. In particular, the following proposition shows that in some cases small changes in action distribution from a safe policy can induce large drops in true reward, but large changes in AD are necessary to improve on the safe policy.

**Proposition 3.1.** Fix  $c_1 > 0$  and  $\delta > 0$  arbitrarily small, and  $c_2 \geq 0$  arbitrarily large. Then there is an MDP, true reward function  $R$ , and safe policy  $\pi_{\text{safe}}$  where both of the following hold:

1. There is a policy  $\pi$  where the action distribution KL divergence satisfies

$$(1-\gamma) \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t D_{KL}(\pi(\cdot | s_t) \| \pi_{\text{safe}}(\cdot | s_t)) \right] \leq c_1$$

$$\text{but } J(\pi_{\text{safe}}, R) - J(\pi, R) \geq 1 - \delta.$$

2. Any optimal policy  $\pi^* \in \arg \max_{\pi} J(\pi, R)$  satisfies

$$(1-\gamma) \mathbb{E}_{\pi^*} \left[ \sum_{t=0}^{\infty} \gamma^t D_{KL}(\pi^*(\cdot | s_t) \| \pi_{\text{safe}}(\cdot | s_t)) \right] \geq c_2.$$

All proofs are given in Appendix A. The first part of Proposition 3.1 states that in the worst case, a policy with AD divergence from the safe policy below some arbitrarily small threshold  $c_1$  can induce a drop in return under the true reward function  $R$  that is almost as large as the entire possible range of returns. Thus, we must set the AD divergence constraint very small (i.e.,  $\epsilon \ll c_1$ ) to prevent reward hacking. The second part of Proposition 3.1 shows that in the same MDP, it is necessary to change the action distribution by an arbitrarily large divergence  $c_2$  to improve on the safe policy and reach an optimal policy. Thus, if we set  $\epsilon \ll c_1$  to prevent reward hacking, it will not allow for the large changes

to the action distribution that are necessary to improve over  $\pi_{\text{safe}}$ . See Figure 1 for a graphical illustration of the results in Proposition 3.1.

While the MDP discussed in Proposition 3.1 represents a particularly bad case for AD regularization, we argue that realistic environments often have the same issues. In many safety-critical environments, even slightly increasing the probability of taking an unsafe action can greatly reduce true reward, as posited in part 1 of the proposition. Furthermore, safe policies are often non-robust out-of-distribution (OOD), e.g., a policy learned from human data might take unusual actions in states outside the distribution of those normally visited. Thus, taking just a single unusual action could lead to an OOD state in which the safe policy is no longer a meaningful regularization target; this also means small AD divergence can lead to large drops in reward.

**The benefits of occupancy measure regularization:** Due to the drawbacks of action distribution regularization, we propose preventing reward hacking by regularizing the divergence between the occupancy measures of the learned and safe policies:

$$\text{maximize } J(\pi, \tilde{R}) \quad \text{s.t.} \quad \|\mu_{\pi} - \mu_{\pi_{\text{safe}}}\|_1 \leq \epsilon. \quad (3)$$

In (3), we use the total variation (TV) between the occupancy measures, defined as

$$\|\mu_{\pi} - \mu_{\pi_{\text{safe}}}\|_1 = \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} |\mu_{\pi}(s, a) - \mu_{\pi_{\text{safe}}}(s, a)|.$$

Why should using the occupancy measure divergence to regularize perform better than using the divergence between action distributions? Ideally, unlike action distribution divergence, there should be a closer relationship between the rewards of two policies and their occupancy measure divergence. In fact, it is possible to show that the difference in returns between two policies for *any* reward function can be bounded by their occupancy measures divergence:

**Proposition 3.2.** For any MDP, reward function  $R$ , and pair of policies  $\pi, \pi_{\text{safe}}$ , we have

$$|J(\pi_{\text{safe}}, R) - J(\pi, R)| \leq \|\mu_{\pi} - \mu_{\pi_{\text{safe}}}\|_1. \quad (4)$$

Results equivalent to Proposition 3.2 have been shown by Xu et al. (2020) among others, but this result has not been applied before in the context of reward hacking. For completeness we give a proof with our notation in Appendix A.2.

Proposition 3.2 suggests that OM regularization can effectively prevent the large drops in true reward associated with reward hacking, even when the true reward is unknown. Suppose the returns of all reward hacking policies  $\pi_{\text{hacking}}$  satisfy  $J_R(\pi_{\text{hacking}}) < J_R(\pi_{\text{safe}}) - C$ , i.e., reward hacking policies have true reward that is smaller than that of the safe policy by more than  $C$ . Then, setting the OM divergence constraint

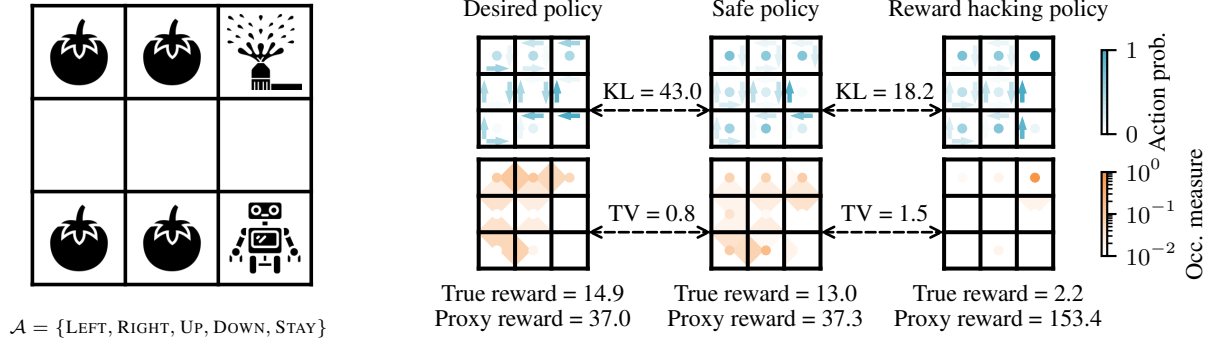


Figure 2. This adaption of the tomato-watering AI Safety Gridworld (Leike et al., 2017) provides an intuitive example of why OM divergence is superior to AD divergence for regularizing to a safe policy. The robot agent can move up, down, left, right, or stay in place. The true reward function  $R$  only rewards watering tomatoes, while the proxy reward function  $\tilde{R}$  also highly rewards reaching the sprinkler. **AD divergence** The top row shows three policies for this environment: a desired policy that achieves the highest true reward, a safe policy that achieves lower true reward, and a reward hacking policy that exploits the sprinkler state to achieve high proxy reward but low true reward. The AD KL divergences between the policies, shown over the arrows connecting them, suggest that the reward hacking policy is actually closer to the safe policy than the desired policy is. Thus, if we regularize to the safe policy using action distribution KL divergence, we would be more likely to find a policy that hacks the proxy reward, rather than one like the left policy, which we prefer. **OM divergence** The bottom row shows the occupancy measures for each policy in the top row, and the arrows between the columns show the total variation distance  $\|\mu - \mu\|_1$ . The desired policy on the left is closer to the safe policy than the reward hacking policy is in OM divergence. This is because both the desired and safe policies spend most of their time actually watering tomatoes, while the reward hacking policy mainly visits the sprinkler state. Thus, if we trained a policy regularized with occupancy measure divergence in this environment, we could hope to find a policy like the desired one on the left and avoid a reward hacking policy like the one on the right.

$\epsilon = C$  in (3) will prevent reward hacking, since any policy within the constraint must satisfy  $J_R(\pi) \geq J_R(\pi_{\text{safe}}) - C$  by Proposition 3.2.  $C$  is often large in practice since reward hacking induces a large drop in true reward. Thus, we can use a large constraint bound  $\epsilon$  in (3) that allows improvement over the safe policy while still preventing reward hacking.

Although it is possible to prove a similar bound to (4) using action distribution divergence, it has a  $\frac{1}{1-\gamma}$  prefactor (Xu et al., 2020), meaning that a constraint on AD divergence must be set  $1 - \gamma$  times the equivalent OM constraint to obtain an equivalent guarantee about the true reward. Thus, in environments with high discount factors—i.e., most realistic environments—the constraint must be set to a value too small to allow meaningful improvement over  $\pi_{\text{safe}}$ .

**An illustrative example:** See Figure 2 for an example of why OM regularization outperforms AD regularization. While in this example it is particularly obvious that OM regularization should work better, we find in Section 5 that OM outperforms AD in more realistic environments too.

**Why does AD regularization work for LLMs?** Despite the drawbacks of action distribution regularization in theory, it has performed well in practice when used as part of RLHF for large language models (LLMs) (Stiennon et al., 2020; Gao et al., 2022). In Appendix A.4, we show that for current implementations of RLHF for LLMs, action distribution and OM-based regularization are actually equivalent. Thus, RLHF for LLMs is essentially already using occupancy measure regularization. However, this is only true under certain

strict assumptions which are satisfied almost exclusively in the current RLHF-for-LLMs paradigm. For more general environments, there can be significant differences between action distribution and OM-based regularization, as clearly demonstrated by our experiments.

#### 4. Occupancy-regularized policy optimization (ORPO)

In the previous sections, we showed theoretical evidence that regularizing RL by constraining OM divergence is superior to constraining AD divergence. We now introduce an algorithm, Occupancy-Regularized Policy Optimization (ORPO), to feasibly approximate the occupancy measure divergence between the learned and safe policies for the purpose of regularizing deep RL agents.

While our theory uses the TV distance between occupancy measures, we find that the KL divergence is more stable to calculate in practice. Since Pinsker’s inequality and the Bretagnolle-Huber inequality show that TV distance is upper-bounded in terms of KL divergence, our theoretical guarantees remain valid in the case of OM KL (Canonne, 2022). Our objective from (3) can be reformulated with the KL divergence in place of the TV distance and a Lagrangian relaxation in place of the hard constraint:

$$\text{maximize } J(\pi, \tilde{R}) - \lambda D_{\text{KL}}(\mu_\pi \parallel \mu_{\pi_{\text{safe}}}). \quad (5)$$

We optimize (5) using a gradient-based method. The gradient of the first term is estimated using PPO, a popular



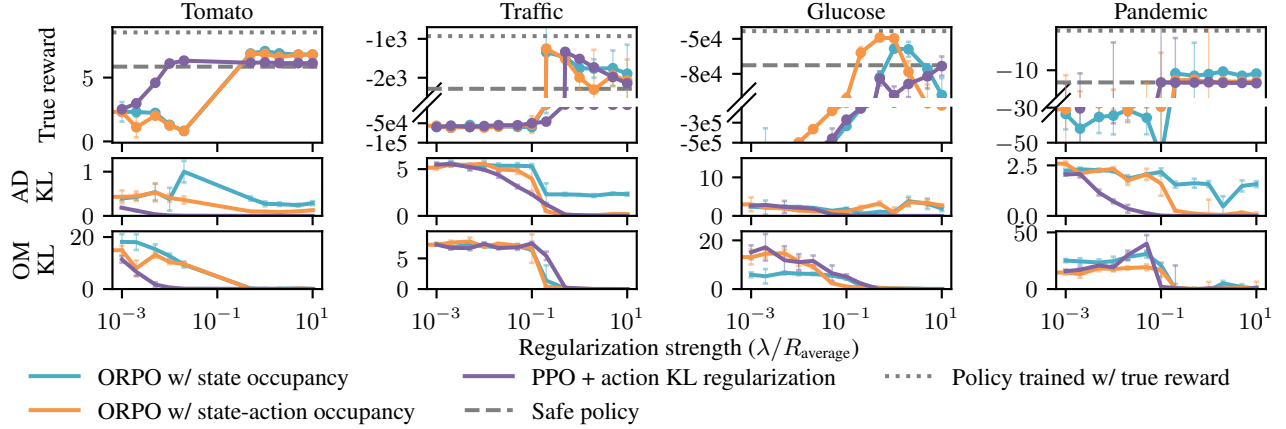


Figure 3. The top row of plots in the figure shows the true rewards of policies trained with three types of regularization to  $\pi_{\text{safe}}$  for several values of  $\lambda/R_{\text{average}}$ : action distribution, state-only OM, and state-action OM. The bottom two rows of plots in the plot show the KL divergence between the action distributions and occupancy measures of the learned and safe policies for each coefficient. For all plots and tables, we give the median across 5 seeds with error bars indicating the standard deviation. We find that reward hacking consistently occurs in each environment without regularization or with very small regularization coefficients  $\lambda$ . As  $\lambda$  is increased to moderate values, the learned policy stops reward hacking and often improves upon the safe policy. At high regularization coefficients, the learned policy approaches the safe policy.

policy gradient method (Schulman et al., 2017). However, calculating the occupancy measure divergence for the second term is intractable to do in closed form since it requires the enumeration of *all* possible state-action pairs, an impossible task in deep RL. Thus, we approximate the KL divergence between the occupancy measures of policies by training a *discriminator network*, a technique that has previously been used for generative adversarial networks (GANs) (Goodfellow et al., 2014) and GAIL (Ho & Ermon, 2016).

The discriminator network  $d : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$  assigns a score  $d(s, a) \in \mathbb{R}$  to any state-action pair  $(s, a) \in \mathcal{S} \times \mathcal{A}$ , and it is trained on a mixture of data from both the learned policy  $\pi$  and safe policy  $\pi_{\text{safe}}$ . The objective used to train  $d$  incentivizes low scores for state-action pairs from  $\pi_{\text{safe}}$  and high scores for state-action pairs from  $\pi$ :

$$d = \arg \min_d \sum_{t=0}^{\infty} \left( \mathbb{E}_{\pi} [\gamma^t \log(1 + e^{-d(s_t, a_t)})] + \mathbb{E}_{\pi_{\text{safe}}} [\gamma^t \log(1 + e^{d(s_t, a_t)})] \right). \quad (6)$$

Huszár (2017) proves that if the loss function in (6) is minimized, then the expected discriminator scores for state-action pairs drawn from the learned policy distribution will approximately equal the KL divergence between the occupancy measures of the two policies:

$$D_{\text{KL}}(\mu_{\pi}(s, a) \parallel \mu_{\pi_{\text{safe}}}(s, a)) \approx (1-\gamma) \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t d(s_t, a_t) \right].$$

Applying the definitions of the learned policy’s returns and the KL divergence between the policies’ occupancy measures, we can now rewrite our ORPO objective:

$$\text{maximize } \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t \left( \tilde{R}(s_t, a_t) - \lambda d(s_t, a_t) \right) \right]. \quad (7)$$

Note that (7) is identical to the normal RL objective with a reward function  $R'(s, a) = \tilde{R}(s, a) - \lambda d(s, a)$ . Thus, once the discriminator has been trained, we add the discriminator scores to the proxy reward function and use the combined values to update  $\pi$  with PPO. The training process for ORPO consists of iterating between two phases: one in which data from both the safe and learned policies is used to train the discriminator to minimize (6), and one in which data from the learned policy is used to train the PPO agent with the augmented reward function in (7). After a policy gradient step, the process repeats.

#### Regularization with state-only occupancy measure:

While we have thus far considered the state-action occupancy measure of a policy  $\mu_{\pi}(s, a)$ , it sometimes makes more sense to regularize based on the state-only occupancy measure  $\mu_{\pi}(s) = (1-\gamma) \mathbb{E}_{\pi} [\sum_{t=0}^{\infty} \gamma^t \mathbb{1}\{s_t = s\}]$ . In particular the reward function  $R(s)$  in many environments is a function of only the state. In these cases, it is simple to establish similar guarantees to Proposition 3.2 based on just the state occupancy measure, and therefore, the state occupancy measure might be more effective in regularizing the behavior of the agent. We can implement this within ORPO by only providing the state as input to the discriminator rather than a state-action pair. Intuitively, regularizing with state OM divergence in environments where the reward function only depends on the state avoids over-applying regularization when it is unnecessary. See Appendix A.5 for more details.

**Regularizing away from reward hacking policies:** While there is a natural safe policy for many environments, it may not always be possible to define one. In such cases, we can potentially regularize *away* from reward hacking behav-

Method	Environment			
	Tomato	Traffic ( $\times 10^3$ )	Glucose ( $\times 10^3$ )	Pandemic
Action dist. regularization ( $\lambda^*$ )	6.19 $\pm$ 0.03	-1.33 $\pm$ 0.05	-73.38 $\pm$ 8.26	-12.20 $\pm$ 0.06
State occupancy regularization ( $\lambda^*$ )	<b>7.07</b> $\pm$ 0.11	-1.47 $\pm$ 0.18	-58.39 $\pm$ 3.36	<b>-10.24</b> $\pm$ 0.54
State-action occupancy regularization ( $\lambda^*$ )	6.80 $\pm$ 0.05	<b>-1.25</b> $\pm$ 0.06	<b>-48.88</b> $\pm$ 0.48	-11.73 $\pm$ 0.19
Action dist. regularization ( $\lambda_{\text{drop}}$ )	4.59 $\pm$ 0.17	-55.10 $\pm$ 2.37	-459.92 $\pm$ 102.08	-23.10 $\pm$ 5.04
State occupancy regularization ( $\lambda_{\text{drop}}$ )	<b>6.89</b> $\pm$ 0.12	-1.34 $\pm$ 22.63	<b>-158.74</b> $\pm$ 25.74	<b>-10.60</b> $\pm$ 0.78
State-action occupancy regularization ( $\lambda_{\text{drop}}$ )	6.84 $\pm$ 0.17	<b>-1.25</b> $\pm$ 0.06	-181.65 $\pm$ 6.69	-11.88 $\pm$ 0.72
$\pi_{\text{safe}}$	5.86 $\pm$ 0.00	-2.28 $\pm$ 0.00	-72.64 $\pm$ 0.00	-12.26 $\pm$ 0.00
No regularization	2.35 $\pm$ 0.14	-57.38 $\pm$ 3.53	-599.02 $\pm$ 1.58	-29.57 $\pm$ 6.86
Early stopping (best case)	6.82 $\pm$ 0.17	-2.24 $\pm$ 0.13	-78.26 $\pm$ 22.90	-9.18 $\pm$ 3.86
Training with true reward	8.54 $\pm$ 0.12	-0.93 $\pm$ 0.11	-43.41 $\pm$ 0.81	-2.65 $\pm$ 0.83

Table 1. The top three rows of the table give the median true reward when using the optimal coefficient  $\lambda^*$  for each type of regularization. The middle three rows show the true reward attained when using the coefficient  $\lambda_{\text{drop}}$  which decreases AD or OM divergence the most compared to a slightly smaller coefficient. The bottom four rows show the true rewards for the baselines: the safe policy  $\pi_{\text{safe}}$ , a policy trained on the proxy reward without regularization (exhibiting reward hacking), a policy trained with the proxy reward with early stopping when the highest true reward is achieved, and a policy trained on the true reward. The latter two baselines are impossible in practice because when true reward is unknown but are given as additional comparisons. The median and standard deviation across 5 random seeds are reported.

ior. That is, suppose training without any regularization in some environment results in a policy  $\pi_{\text{hacking}}$  that exhibits reward hacking. Then, we can train a second policy with the following objective:

$$\text{maximize } J(\pi, \tilde{R}) + \lambda D_{\text{KL}}(\mu_{\pi} \parallel \mu_{\pi_{\text{hacking}}}).$$

Unlike in (3), we encourage the occupancy measure of our new policy  $\pi$  to be as far as possible from  $\pi_{\text{hacking}}$ . This will hopefully prevent  $\pi$  from exhibiting the same reward hacking behavior as  $\pi_{\text{hacking}}$ . It is trivial to modify ORPO to optimize this objective; we simply need to flip the sign of the discriminator term in (7).

## 5. Experiments

We now use ORPO to compare the empirical performance of occupancy measure and action distribution regularization in four environments: a tomato-watering gridworld similar to that in Figure 2; Flow, an autonomous vehicle control environment introduced by Wu et al. (2022); SimGlucose, a blood glucose monitoring system developed by Fox et al. (2020), and a COVID-19 pandemic simulator created by Kompella et al. (2020). We chose the first for illustrative purposes, and the following three because they are reward hacking benchmark environments from Pan et al. (2022).

**Tomato gridworld:** Like in Figure 2, the tomato environment contains a sprinkler state where the agent perceives all tomatoes as being watered and thus receives high proxy reward but no true reward. We train a safe policy using the true reward function, and then add a 10% chance of taking a random action to ensure there is room to improve upon it.

**Flow traffic simulator:** The traffic environment simulates a group of human-controlled and RL-controlled vehicles

on an on-ramp attempting to merge into traffic on a highway. The true reward prioritizes a small mean commute time, while the proxy reward is the average velocity of all cars. When reward hacking, the RL controlled vehicle on the on-ramp stops indefinitely and lets cars continue forward at high speeds on the highway, which maximizes the proxy reward but increases the commute times of cars on the on-ramp infinitely. As the safe policy for the traffic environment we used the Intelligent Driver Model (IDM), a standard approximation of human driving behavior (Treiber et al., 2000). In practice, safe policies are often learned via imitation learning, so to simulate this we generate data from the IDM controller and train a behavioral cloning (BC) policy using the generated data.

**SimGlucose:** The SimGlucose blood glucose monitoring environment is an extension of the FDA-approved glucose monitoring simulator proposed by Man et al. (2014) for Type 1 Diabetes patients. The RL agent controls the insulin administered to a simulated patient in order to maintain healthy glucose levels. The true reward is a standard measure of health risk for the patient, but the proxy reward is misaligned and prioritizes the monetary cost of the treatment. Optimizing for a cost-based proxy has caused major disparities in access to healthcare on the basis of race (Obermeyer et al., 2019). As the safe baseline policy, we train a BC policy based on data generated by a PID controller with parameters tuned by the original designers of the simulator (Steil, 2013).

**COVID-19 simulator:** The pandemic environment simulates a population’s infection dynamics using the SEIR model (Mwalili et al., 2020). The RL agent chooses the level of lockdown restrictions placed on the population by observing the results of testing. The proxy reward func-

tion omits the political cost associated with certain decisions. Our safe policy is trained via BC on a combination of hand-specified and real-world strategies employed by governments during the pandemic, which were also used by Kompella et al. (2020) as baselines.

**Regularizing towards a safe policy:** We train RL policies in each environment with action distribution regularization and OM regularization to the environment’s safe policy, varying the regularization coefficient  $\lambda$  across a wide range. Since the scale of the reward functions varies between environments, we normalize  $\lambda$  by the typical per-timestep reward for each environment, which we denote  $R_{\text{average}}$ . In the environments that we studied,  $\lambda/R_{\text{average}}$  values between  $10^{-3}$  and  $10^1$  seemed to work best. See Appendix D for all experimental details.

The results of our experiments are shown in Table 1 and Figure 3. In each environment, we find that OM regularization with the optimal coefficient ( $\lambda^*$ ) outperforms action distribution regularization. OM regularization consistently allows improvement over the performance of  $\pi_{\text{safe}}$  while preventing reward hacking; meanwhile, action distribution regularization fails to improve significantly on the safe policy in the glucose and pandemic environments.

We were able to determine the optimal regularization coefficients  $\lambda^*$  by considering the policies’ performance on the true reward. However, in practice, comparing the results of each type of regularization with the optimal coefficients is unrealistic, since system designers must choose a regularization regime without access to the unknown true reward function. Observing changes in the policies’ divergences as  $\lambda$  is varied can help designers choose the right coefficient. In particular, we find that the optimal regularization coefficient is often the coefficient at which the regularized divergence drops the most compared to a slightly smaller coefficient. To demonstrate this, we compare the true rewards of policies for  $\lambda$  values chosen based on this heuristic—which we denote as  $\lambda_{\text{drop}}$ —in the middle three rows of Table 1. Regularizing based on OM divergence with  $\lambda_{\text{drop}}$  achieves true reward close to those obtained with  $\lambda^*$ , despite being chosen without access to the true reward function.

We find that both state-only and state-action occupancy measure regularization achieve similar true reward. Generally, state-only occupancy measures perform better in environments whose true reward functions depend primarily on the state, reflecting the intuition of our theory in Appendix A.5. In practice, we recommend experimenting with both OM regularizers.

In addition to comparing OM and AD regularization, we also test *early stopping*, which has been proposed by Karowski et al. (2023) as another method for preventing reward hacking. While they introduce a specific criterion for

Environment	AUROC for predicting reward hacking	
	Occ. measure KL	Action dist. KL
Tomato	<b>1.00</b>	0.89
Traffic	<b>1.00</b>	0.98
Glucose	<b>0.99</b>	0.79
Pandemic	<b>0.94</b>	0.82

Table 2. We find that, compared to AD divergence, OM divergence is a much better predictor of whether reward hacking is occurring during training according to its area under the ROC curve (AUROC). This validates that OM divergence is a more successful regularizer because it more accurately identifies when reward hacking is happening. See Figure 4 for full AUROC curves.

deciding when to stop training, we consider the best possible case: we train policies on the proxy reward function and then evaluate the policy from the iteration with the highest true reward. We find that OM regularization is superior to early stopping in all environments except for the pandemic simulator. Furthermore, this best-case approach is infeasible in practice since the true reward is unknown, so a more realistic early stopping method would only perform more poorly. Our results thus suggest that policy regularization is usually more effective at preventing reward hacking than early stopping.

**Explaining the superior performance of OM regularization:** In Section 3, we hypothesized that OM regularization is superior to action distribution regularization because there is a stronger relationship between OM divergence and the difference in returns of two policies under any reward function; therefore, OM divergence should better measure when there is a large difference in true rewards between the safe and learned policies, indicating reward hacking. We empirically test this hypothesis by comparing how well both action distribution and OM divergence predict if reward hacking is occurring during RL training. In particular, we divide all of our training runs into ten segments, and for each segment record (i) whether the agent is reward hacking, (ii) the action distribution divergence from  $\pi_{\text{safe}}$ , and (iii) the OM divergence from  $\pi_{\text{safe}}$ . For (i), we define reward hacking as achieving higher proxy reward but lower true reward than  $\pi_{\text{safe}}$ . Then, we calculate how accurately each type of divergence can predict whether reward hacking is occurring across all training run segments. The results of this experiment are shown in Table 2. We find that in all environments, OM divergence is a better classifier of reward hacking behavior, validating our hypothesis as to why it is a better regularizer for preventing reward hacking.

**Regularizing away from reward hacking behavior:** We experiment with regularizing away from reward hacking policies using both action distribution and OM regularization. We obtain a  $\pi_{\text{hacking}}$  for each environment by training on the proxy reward without regularization, and we regularize away from  $\pi_{\text{hacking}}$  using a range of values of  $\lambda$ .

Regularization	Environment			
	Tomato	Traffic ( $\times 10^3$ )	Glucose ( $\times 10^3$ )	Pandemic
AD	$1.98 \pm 1.49$	$-58.23 \pm 2.95$	<b><math>-10.37 \pm 0.20</math></b>	<b><math>-8.35 \pm 1.94</math></b>
State OM	$5.32 \pm 0.22$	$-1.10 \pm 0.04$	$-186.14 \pm 17.98$	$-14.28 \pm 0.40$
State-Act. OM	<b><math>5.59 \pm 0.32</math></b>	<b><math>-1.07 \pm 0.01</math></b>	$-93.15 \pm 29.54$	$-14.23 \pm 5.02$
No regularization ( $\pi_{\text{hacking}}$ )	$2.35 \pm 0.14$	$-57.38 \pm 3.53$	$-599.02 \pm 1.58$	$-29.57 \pm 6.86$

Table 3. The true rewards achieved by regularizing away from reward hacking policies in the four environments. OM regularization prevents reward hacking in all four environments, while AD regularization fails to improve on  $\pi_{\text{hacking}}$  in the tomato and traffic environments.

The results are presented in Table 3. We find that OM KL regularization consistently avoids reward hacking and, in some cases, even outperforms the safe policies. On the other hand, the AD regularized policies’ true reward is dangerously close to that of  $\pi_{\text{hacking}}$  in some of the environments, indicating that it is unable to consistently prevent reward hacking.

## 6. Conclusion

We have presented theoretical and empirical evidence that occupancy measure regularization can more effectively prevent reward hacking than action distribution regularization when training with a misaligned proxy reward function. Our results are a step towards a better understanding of methods for preventing reward hacking. However, many open problems remain, including determining the best way of integrating occupancy measure regularization into the training process, deriving better approximators of OM divergence, and intelligently choosing regularization coefficients. As AI systems’ objectives become more complex and they are used in increasingly important societal roles, reward hacking will become both more common and more consequential. Thus, we hope that our results contribute to the goal of ensuring that future AI systems are safe and beneficial.

## Broader Impacts

Reward hacking in the real world has already led to significant disparities on the basis of race, gender, and other distinguishing factors in the realms of healthcare (Obermeyer et al., 2019; Pierson et al., 2021), policing (Lum & Isaac, 2016; Corbett-Davies et al., 2017; Franchi et al., 2023), and online platforms like recommender systems (Milli et al., 2021; Kleinberg et al., 2023). As AI agents trained with RL become more powerful, it is likely these systems will exhibit reward hacking behaviors that could further exacerbate bias and can potentially cause substantial harm, especially in safety critical scenarios. Thus, the aim of our work is to prevent reward hacking by improving regularization methods. We do not believe that there are any noteworthy negative

consequences of developing such methods.

## References

- Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., and Mané, D. Concrete Problems in AI Safety, July 2016. URL <http://arxiv.org/abs/1606.06565>. arXiv:1606.06565 [cs].
- Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., Das-Sarma, N., Drain, D., Fort, S., Ganguli, D., Henighan, T., Joseph, N., Kadavath, S., Kernion, J., Conerly, T., El-Showk, S., Elhage, N., Hatfield-Dodds, Z., Hernandez, D., Hume, T., Johnston, S., Kravec, S., Lovitt, L., Nanda, N., Olsson, C., Amodei, D., Brown, T., Clark, J., McCandlish, S., Olah, C., Mann, B., and Kaplan, J. Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback, April 2022. URL <http://arxiv.org/abs/2204.05862>. arXiv:2204.05862 [cs].
- Canonne, C. L. A short note on an inequality between KL and TV, February 2022. URL <http://arxiv.org/abs/2202.07198>. arXiv:2202.07198 [math, stat].
- Cheng, C.-A., Xie, T., Jiang, N., and Agarwal, A. Adversarially Trained Actor Critic for Offline Reinforcement Learning, July 2022. URL <http://arxiv.org/abs/2202.02446>. arXiv:2202.02446 [cs].
- Chow, Y., Nachum, O., Faust, A., Duenez-Guzman, E., and Ghavamzadeh, M. Lyapunov-based Safe Policy Optimization for Continuous Control, February 2019. URL <http://arxiv.org/abs/1901.10031>. arXiv:1901.10031 [cs, stat].
- Corbett-Davies, S., Pierson, E., Feller, A., Goel, S., and Huq, A. Algorithmic decision making and the cost of fairness, June 2017. URL <http://arxiv.org/abs/1701.08230>. arXiv:1701.08230 [cs, stat].
- Dalal, G., Dvijotham, K., Vecerik, M., Hester, T., Paduraru, C., and Tassa, Y. Safe Exploration in Continuous Action



- Spaces, January 2018. URL <http://arxiv.org/abs/1801.08757>. arXiv:1801.08757 [cs].
- Fox, I., Lee, J., Pop-Busui, R., and Wiens, J. Deep Reinforcement Learning for Closed-Loop Blood Glucose Control, September 2020. URL <http://arxiv.org/abs/2009.09051>. arXiv:2009.09051 [cs, stat].
- Franchi, M., Zamfirescu-Pereira, J. D., Ju, W., and Pierson, E. Detecting disparities in police deployments using dashcam data. In *2023 ACM Conference on Fairness, Accountability, and Transparency*, pp. 534–544, June 2023. doi: 10.1145/3593013.3594020. URL <http://arxiv.org/abs/2305.15210>. arXiv:2305.15210 [cs].
- Gao, L., Schulman, J., and Hilton, J. Scaling Laws for Reward Model Overoptimization, October 2022. URL <http://arxiv.org/abs/2210.10760>. arXiv:2210.10760 [cs, stat].
- Glaese, A., McAleese, N., Trebacz, M., Aslanides, J., Firoiu, V., Ewalds, T., Rauh, M., Weidinger, L., Chadwick, M., Thacker, P., Campbell-Gillingham, L., Uesato, J., Huang, P.-S., Comanescu, R., Yang, F., See, A., Dathathri, S., Greig, R., Chen, C., Fritz, D., Elias, J. S., Green, R., Mokra, S., Fernando, N., Wu, B., Foley, R., Young, S., Gabriel, I., Isaac, W., Mellor, J., Hassabis, D., Kavukcuoglu, K., Hendricks, L. A., and Irving, G. Improving alignment of dialogue agents via targeted human judgements, September 2022. URL <http://arxiv.org/abs/2209.14375>. arXiv:2209.14375 [cs].
- Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative Adversarial Networks, June 2014. URL <http://arxiv.org/abs/1406.2661>. arXiv:1406.2661 [cs, stat].
- Goodhart, C. A. E. Problems of Monetary Management: The UK Experience. In Goodhart, C. A. E. (ed.), *Monetary Theory and Practice: The UK Experience*, pp. 91–121. Macmillan Education UK, London, 1984. ISBN 978-1-349-17295-5. doi: 10.1007/978-1-349-17295-5\_4. URL [https://doi.org/10.1007/978-1-349-17295-5\\_4](https://doi.org/10.1007/978-1-349-17295-5_4).
- Hadfield-Menell, D., Milli, S., Abbeel, P., Russell, S., and Dragan, A. Inverse Reward Design, 2017. URL <http://arxiv.org/abs/1711.02827>. arXiv:1711.02827 [cs].
- Hazan, E., Kakade, S. M., Singh, K., and Van Soest, A. Provably Efficient Maximum Entropy Exploration, January 2019. URL <http://arxiv.org/abs/1812.02690>. arXiv:1812.02690 [cs, stat].
- He, H. A Survey on Offline Model-Based Reinforcement Learning, May 2023. URL <http://arxiv.org/abs/2305.03360>. arXiv:2305.03360 [cs, eess].
- Ho, J. and Ermon, S. Generative Adversarial Imitation Learning. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL [https://papers.nips.cc/paper\\_files/paper/2016/hash/cc7e2b878868cbae992d1fb743995d8f-Abstract.html](https://papers.nips.cc/paper_files/paper/2016/hash/cc7e2b878868cbae992d1fb743995d8f-Abstract.html).
- Huszár, F. Variational Inference using Implicit Distributions, February 2017. URL <http://arxiv.org/abs/1702.08235>. arXiv:1702.08235 [cs, stat].
- Ibarz, B., Leike, J., Pohlen, T., Irving, G., Legg, S., and Amodei, D. Reward learning from human preferences and demonstrations in Atari. 2018. doi: 10.48550/ARXIV.1811.06521. URL <https://arxiv.org/abs/1811.06521>. Publisher: arXiv Version Number: 1.
- Jeon, H. J., Milli, S., and Dragan, A. D. Reward-rational (implicit) choice: A unifying formalism for reward learning, December 2020. URL <http://arxiv.org/abs/2002.04833>. arXiv:2002.04833 [cs].
- Kang, B., Jie, Z., and Feng, J. Policy optimization with demonstrations. In *International Conference on Machine Learning*, 2018. URL <https://api.semanticscholar.org/CorpusID:51875782>.
- Karwowski, J., Hayman, O., Bai, X., Kiendlhofer, K., Griffin, C., and Skalse, J. Goodhart’s Law in Reinforcement Learning, October 2023. URL <http://arxiv.org/abs/2310.09144>. arXiv:2310.09144 [cs].
- Kleinberg, J., Mullainathan, S., and Raghavan, M. The Challenge of Understanding What Users Want: Inconsistent Preferences and Engagement Optimization, October 2023. URL <http://arxiv.org/abs/2202.11776>. arXiv:2202.11776 [cs].
- Knox, W. B., Allievi, A., Banzhaf, H., Schmitt, F., and Stone, P. Reward (Mis)design for Autonomous Driving, March 2022. URL <http://arxiv.org/abs/2104.13906>. arXiv:2104.13906 [cs].
- Kompella, V., Capobianco, R., Jong, S., Browne, J., Fox, S., Meyers, L., Wurman, P., and Stone, P. Reinforcement Learning for Optimization of COVID-19 Mitigation policies, October 2020. URL <http://arxiv.org/abs/2010.10560>. arXiv:2010.10560 [cs].
- Korbak, T., Perez, E., and Buckley, C. L. RL with KL penalties is better viewed as Bayesian inference, October 2022. URL <http://arxiv.org/abs/2205.11275>. arXiv:2205.11275 [cs, stat].



- Krakovna, V. Specification gaming examples in AI, April 2018. URL <https://vkrakovna.wordpress.com/2018/04/02/specification-gaming-examples-in-ai/>.
- Krakovna, V. Classifying specification problems as variants of Goodhart’s Law, August 2019. URL <https://vkrakovna.wordpress.com/2019/08/19/classifying-specification-problems-as-variants-of-goodhart-law/>.
- Krakovna, V., Orseau, L., Kumar, R., Martic, M., and Legg, S. Penalizing side effects using stepwise relative reachability, March 2019. URL <http://arxiv.org/abs/1806.01186>. arXiv:1806.01186 [cs, stat].
- Laidlaw, C., Russell, S., and Dragan, A. Bridging RL Theory and Practice with the Effective Horizon, April 2023. URL <http://arxiv.org/abs/2304.09853>. arXiv:2304.09853 [cs, stat].
- Lee, J., Paduraru, C., Mankowitz, D. J., Heess, N., Precup, D., Kim, K.-E., and Guez, A. COptiDICE: Offline Constrained Reinforcement Learning via Stationary Distribution Correction Estimation, April 2022. URL <http://arxiv.org/abs/2204.08957>. arXiv:2204.08957 [cs].
- Lee, K., Smith, L., and Abbeel, P. PEBBLE: Feedback-Efficient Interactive Reinforcement Learning via Relabeling Experience and Unsupervised Pre-training, June 2021. URL <http://arxiv.org/abs/2106.05091>. arXiv:2106.05091 [cs].
- Lee, L., Eysenbach, B., Parisotto, E., Xing, E., Levine, S., and Salakhutdinov, R. Efficient Exploration via State Marginal Matching, February 2020. URL <http://arxiv.org/abs/1906.05274>. arXiv:1906.05274 [cs, stat].
- Leike, J., Martic, M., Krakovna, V., Ortega, P. A., Everitt, T., Lefrancq, A., Orseau, L., and Legg, S. AI Safety Gridworlds, November 2017. URL <http://arxiv.org/abs/1711.09883>. arXiv:1711.09883 [cs].
- Leike, J., Krueger, D., Everitt, T., Martic, M., Maini, V., and Legg, S. Scalable agent alignment via reward modeling: a research direction, November 2018. URL <http://arxiv.org/abs/1811.07871>. arXiv:1811.07871 [cs, stat].
- Li, A., Misra, D., Kolobov, A., and Cheng, C.-A. Survival Instinct in Offline Reinforcement Learning, November 2023. URL <http://arxiv.org/abs/2306.03286>. arXiv:2306.03286 [cs].
- Liang, E., Liaw, R., Moritz, P., Nishihara, R., Fox, R., Goldberg, K., Gonzalez, J. E., Jordan, M. I., and Stoica, I. RLlib: Abstractions for Distributed Reinforcement Learning, June 2018. URL <http://arxiv.org/abs/1712.09381>. arXiv:1712.09381 [cs].
- Lum, K. and Isaac, W. To Predict and Serve? *Significance*, 13(5):14–19, October 2016. ISSN 1740-9705. doi: 10.1111/j.1740-9713.2016.00960.x. URL <https://doi.org/10.1111/j.1740-9713.2016.00960.x>. eprint: <https://academic.oup.com/jrssig/article-pdf/13/5/14/4910646/sign.13v5.14.pdf>.
- Man, C. D., Micheletto, F., Lv, D., Breton, M., Kovatchev, B., and Cobelli, C. The UVA/PADOVA Type 1 Diabetes Simulator. *Journal of Diabetes Science and Technology*, 8(1):26–34, January 2014. ISSN 1932-2968. doi: 10.1177/1932296813514502. URL <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4454102/>.
- Mandal, D., Triantafyllou, S., and Radanovic, G. Performative Reinforcement Learning, February 2023. URL <http://arxiv.org/abs/2207.00046>. arXiv:2207.00046 [cs].
- McKinney, L., Duan, Y., Krueger, D., and Gleave, A. On The Fragility of Learned Reward Functions, January 2023. URL <http://arxiv.org/abs/2301.03652>. arXiv:2301.03652 [cs].
- Milli, S., Belli, L., and Hardt, M. From Optimizing Engagement to Measuring Value. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pp. 714–722, March 2021. doi: 10.1145/3442188.3445933. URL <http://arxiv.org/abs/2008.12623>. arXiv:2008.12623 [cs, stat].
- Mwalili, S., Kimathi, M., Ojiambo, V., Gathungu, D., and Mbogo, R. SEIR model for COVID-19 dynamics incorporating the environment and social distancing. *BMC Research Notes*, 13:352, July 2020. ISSN 1756-0500. doi: 10.1186/s13104-020-05192-1. URL <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7376536/>.
- Nedergaard, A. and Cook, M. k-Means Maximum Entropy Exploration, November 2023. URL <http://arxiv.org/abs/2205.15623>. arXiv:2205.15623 [cs].
- Ngo, R., Chan, L., and Mindermann, S. The alignment problem from a deep learning perspective, September 2023. URL <http://arxiv.org/abs/2209.00626>. arXiv:2209.00626 [cs].
- Obermeyer, Z., Powers, B., Vogeli, C., and Mullainathan, S. Dissecting racial bias in an algorithm used to manage the health of populations. *Science*, 366(6464):447–453, October 2019. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.aax2342. URL <https://www.science.org/doi/10.1126/science.aax2342>.

- Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., and Lowe, R. Training language models to follow instructions with human feedback. 2022. doi: 10.48550/ARXIV.2203.02155. URL <https://arxiv.org/abs/2203.02155>.
- Pan, A., Bhatia, K., and Steinhardt, J. The Effects of Reward Misspecification: Mapping and Mitigating Misaligned Models, February 2022. URL <http://arxiv.org/abs/2201.03544>. arXiv:2201.03544 [cs, stat].
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. PyTorch: An Imperative Style, High-Performance Deep Learning Library, December 2019. URL <http://arxiv.org/abs/1912.01703>. arXiv:1912.01703 [cs, stat].
- Pierson, E., Cutler, D. M., Leskovec, J., Mullainathan, S., and Obermeyer, Z. An algorithmic approach to reducing unexplained pain disparities in underserved populations. *Nature Medicine*, 27:136 – 140, 2021. URL <https://api.semanticscholar.org/CorpusID:231598658>.
- Rashidinejad, P., Zhu, B., Ma, C., Jiao, J., and Russell, S. Bridging Offline Reinforcement Learning and Imitation Learning: A Tale of Pessimism, July 2023. URL <http://arxiv.org/abs/2103.12021>. arXiv:2103.12021 [cs, math, stat].
- Reddy, S., Dragan, A. D., Levine, S., Legg, S., and Leike, J. Learning Human Objectives by Evaluating Hypothetical Behavior, March 2020. URL <http://arxiv.org/abs/1912.05652>. arXiv:1912.05652 [cs, stat].
- Roy, J., Girgis, R., Romoff, J., Bacon, P.-L., and Pal, C. Direct Behavior Specification via Constrained Reinforcement Learning, June 2022. URL <http://arxiv.org/abs/2112.12228>. arXiv:2112.12228 [cs].
- Russell, S. J., Norvig, P., and Davis, E. *Artificial intelligence: a modern approach*. Prentice Hall series in artificial intelligence. Prentice Hall, Upper Saddle River, 3rd ed edition, 2010. ISBN 978-0-13-604259-4.
- Sadigh, D., Dragan, A., Sastry, S., and Seshia, S. Active Preference-Based Learning of Reward Functions. In *Robotics: Science and Systems XIII*. Robotics: Science and Systems Foundation, July 2017. ISBN 978-0-9923747-3-0. doi: 10.15607/RSS.2017.XIII.053. URL <http://www.roboticsproceedings.org/rss13/p53.pdf>.
- Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal Policy Optimization Algorithms, August 2017. URL <http://arxiv.org/abs/1707.06347>. arXiv:1707.06347 [cs].
- Skalse, J., Howe, N. H. R., Krasheninnikov, D., and Krueger, D. Defining and Characterizing Reward Hacking, September 2022. URL <http://arxiv.org/abs/2209.13085>. arXiv:2209.13085 [cs, stat].
- Steil, G. M. Algorithms for a Closed-Loop Artificial Pancreas: The Case for Proportional-Integral-Derivative Control. *Journal of Diabetes Science and Technology*, 7(6):1621–1631, November 2013. ISSN 1932-2968. URL <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3876341/>.
- Stiennon, N., Ouyang, L., Wu, J., Ziegler, D. M., Lowe, R., Voss, C., Radford, A., Amodei, D., and Christiano, P. Learning to summarize from human feedback, September 2020. URL <http://arxiv.org/abs/2009.01325>. arXiv:2009.01325 [cs].
- Taylor, J. Quantilizers: A Safer Alternative to Maximizers for Limited Optimization. March 2016. URL <https://www.semanticscholar.org/paper/Quantilizers%3A-A-Safer-Alternative-to-Maximizers-for-Taylor/4e8ff3b4069a12a00196d62925bab8add7389742>.
- Taylor, J., Yudkowsky, E., LaVictoire, P., and Critch, A. Alignment for Advanced Machine Learning Systems. In *Ethics of Artificial Intelligence*. Oxford University Press, August 2020. ISBN 978-0-19-090505-7. Google-Books-ID: 1yT3DwAAQBAJ.
- Tien, J., He, J. Z.-Y., Erickson, Z., Dragan, A. D., and Brown, D. S. Causal Confusion and Reward Misidentification in Preference-Based Reward Learning, March 2023. URL <http://arxiv.org/abs/2204.06601>. arXiv:2204.06601 [cs].
- Treiber, M., Hennecke, A., and Helbing, D. Congested Traffic States in Empirical Observations and Microscopic Simulations. *Physical Review E*, 62(2):1805–1824, August 2000. ISSN 1063-651X, 1095-3787. doi: 10.1103/PhysRevE.62.1805. URL <http://arxiv.org/abs/cond-mat/0002177>. arXiv:cond-mat/0002177.
- Turner, A. M., Smith, L., Shah, R., Critch, A., and Tadepalli, P. Optimal Policies Tend To Seek Power. December 2019. URL <https://www.semanticscholar.org/paper/Optimal-Policies-Tend-To-Seek-Power-Turner-Smith/46d4452eb041e33f1e58eab64ec8cf5af534b6ff>.

- Turner, A. M., Ratzlaff, N., and Tadepalli, P. Avoiding Side Effects in Complex Environments, October 2020. URL <http://arxiv.org/abs/2006.06547>. arXiv:2006.06547 [cs].
- Uchendu, I., Xiao, T., Lu, Y., Zhu, B., Yan, M., Simon, J., Bennice, M., Fu, C., Ma, C., Jiao, J., Levine, S., and Hausman, K. Jump-Start Reinforcement Learning, July 2023. URL <http://arxiv.org/abs/2204.02372>. arXiv:2204.02372 [cs].
- Vieillard, N., Kozuno, T., Scherrer, B., Pietquin, O., Munos, R., and Geist, M. Leverage the Average: an Analysis of KL Regularization in RL, January 2021. URL <http://arxiv.org/abs/2003.14089>. arXiv:2003.14089 [cs, stat].
- Wu, C., Kreidieh, A., Parvate, K., Vinitzky, E., and Bayen, A. M. Flow: A Modular Learning Framework for Mixed Autonomy Traffic. *IEEE Transactions on Robotics*, 38 (2):1270–1286, April 2022. ISSN 1552-3098, 1941-0468. doi: 10.1109/TRO.2021.3087314. URL <http://arxiv.org/abs/1710.05465>. arXiv:1710.05465 [cs].
- Xie, T., Cheng, C.-A., Jiang, N., Mineiro, P., and Agarwal, A. Bellman-consistent Pessimism for Offline Reinforcement Learning, October 2023. URL <http://arxiv.org/abs/2106.06926>. arXiv:2106.06926 [cs, stat].
- Xu, T., Li, Z., and Yu, Y. Error Bounds of Imitating Policies and Environments, October 2020. URL <http://arxiv.org/abs/2010.11876>. arXiv:2010.11876 [cs].
- Yang, S., Feng, Y., Zhang, S., and Zhou, M. Regularizing a Model-based Policy Stationary Distribution to Stabilize Offline Reinforcement Learning, June 2022. URL <http://arxiv.org/abs/2206.07166>. arXiv:2206.07166 [cs].
- Yang, T.-Y., Rosca, J., Narasimhan, K., and Ramadge, P. J. Accelerating Safe Reinforcement Learning with Constraint-mismatched Policies, July 2021. URL <http://arxiv.org/abs/2006.11645>. arXiv:2006.11645 [cs, stat].
- Zhang, Y., Vuong, Q., and Ross, K. W. First Order Constrained Optimization in Policy Space, October 2020. URL <http://arxiv.org/abs/2002.06506>. arXiv:2002.06506 [cs, stat].

# Appendix

## A. Proofs

### A.1. Proof of Proposition 3.1

**Proposition 3.1.** Fix  $c_1 > 0$  and  $\delta > 0$  arbitrarily small, and  $c_2 \geq 0$  arbitrarily large. Then there is an MDP, true reward function  $R$ , and safe policy  $\pi_{\text{safe}}$  where both of the following hold:

1. There is a policy  $\pi$  where the action distribution KL divergence satisfies

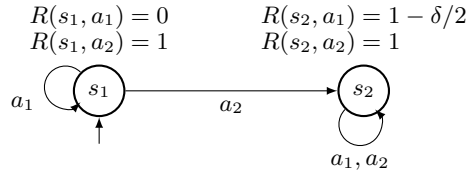
$$(1 - \gamma) \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t D_{\text{KL}}(\pi(\cdot | s_t) \| \pi_{\text{safe}}(\cdot | s_t)) \right] \leq c_1$$

but  $J(\pi_{\text{safe}}, R) - J(\pi, R) \geq 1 - \delta$ .

2. Any optimal policy  $\pi^* \in \arg \max_\pi J(\pi, R)$  satisfies

$$(1 - \gamma) \mathbb{E}_{\pi^*} \left[ \sum_{t=0}^{\infty} \gamma^t D_{\text{KL}}(\pi^*(\cdot | s_t) \| \pi_{\text{safe}}(\cdot | s_t)) \right] \geq c_2.$$

*Proof.* We assume that  $\delta < 1$ , since otherwise letting  $\pi = \pi_{\text{safe}}$  trivially satisfies the first part of the proposition. Consider the following MDP, similar to the one shown in Figure 1:



In this MDP,  $\mathcal{S} = \{s_1, s_2\}$ ,  $\mathcal{A} = \{a_1, a_2\}$ , and the transition probabilities and reward function are defined by

$$\begin{aligned} p(s_1 | s_1, a_1) &= 1 & p(s_2 | s_1, a_2) &= 1 \\ p(s_2 | s_2, a_1) &= 1 & p(s_2 | s_2, a_2) &= 1 \\ R(s_1, a_1) &= 0 & R(s_1, a_2) &= 1 \\ R(s_2, a_1) &= 1 - \delta/2 & R(s_2, a_2) &= 1. \end{aligned}$$

The initial state is always  $s_1$ . Thus, the agent stays in state  $s_1$  and receives no reward until after it takes action  $a_2$ , at which point it transitions to  $s_2$  and receives 1 or  $1 - \delta/2$  reward per timestep. Define for any  $(p, q) \in [0, 1]^2$  a policy  $\pi_{(p,q)}$ :

$$\pi_{(p,q)}(a_2 | s_1) = p \quad \pi_{(p,q)}(a_2 | s_2) = q.$$

We will prove the proposition using

$$\begin{aligned} \gamma &= 1 - \frac{\delta}{2}(1 - e^{-c_1}) \\ \pi_{\text{safe}} &= \pi_{(p,q)} \quad \text{where } p = 2(1 - \gamma)/\delta \quad \text{and } q = \exp\{-c_2/\gamma\} \\ \pi &= \pi_{(0,0)} \\ \pi^* &= \pi_{(1,1)}. \end{aligned}$$

Note the following:

- $\pi^*$  is the unique optimal policy:  $J(\pi^*, R) = 1$  and for any other policy  $\pi$ ,  $J(\pi, R) < 1$ .
- $\gamma \in [0, 1)$ :  $c_1 > 0$  and thus  $1 - e^{-c_1} > 0$ , and  $\delta < 1$ .
- $p \in [0, 1]$ : since  $\gamma > 1 - \delta/2$ , we have  $p < 1$ .

- $q \in [0, 1]$ : since  $c_2 \geq 0$ , we have  $q \leq 1$ .

To start, we need to show that

$$(1 - \gamma) \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t D_{\text{KL}}(\pi(\cdot | s_t) \| \pi_{\text{safe}}(\cdot | s_t)) \right] \leq c_1. \quad (8)$$

Since  $\pi$  always stays at  $s_1$ , we can rewrite the LHS of (8) as

$$\begin{aligned} (1 - \gamma) \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t D_{\text{KL}}(\pi(\cdot | s_t) \| \pi_{\text{safe}}(\cdot | s_t)) \right] &= D_{\text{KL}}(\pi(\cdot | s_1) \| \pi_{\text{safe}}(\cdot | s_1)) \\ &= \pi(a_1 | s_1) \log \frac{\pi(a_1 | s_1)}{\pi_{\text{safe}}(a_1 | s_1)} + \pi(a_2 | s_1) \log \frac{\pi(a_2 | s_1)}{\pi_{\text{safe}}(a_2 | s_1)} \\ &= \log \frac{1}{1 - p} \\ &= \log \frac{1}{e^{-c_1}} \\ &= c_1, \end{aligned}$$

which proves (8).

Next, we need to show that  $J(\pi_{\text{safe}}, R) - J(\pi, R) \geq 1 - \delta$ . Clearly,  $J(\pi, R) = 0$ . We can bound  $J(\pi_{\text{safe}}, R)$  as

$$\begin{aligned} J(\pi_{\text{safe}}, R) &= (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t \left[ \mathbb{P}_\pi(s_t = s_1)p + \mathbb{P}_\pi(s_t = s_2)(q + (1 - q)(1 - \delta/2)) \right] \\ &\geq (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t (1 - \delta/2) \left[ \mathbb{P}_\pi(s_t = s_1)p + \mathbb{P}_\pi(s_t = s_2) \right] \\ &= (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t (1 - \delta/2) \mathbb{P}_\pi(\exists t' \leq t \text{ s.t. } a_{t'} = a_2) \\ &= (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t (1 - \delta/2) \left( 1 - (1 - p)^{t+1} \right) \\ &= (1 - \gamma)(1 - \delta/2)(1 - p) \sum_{t=0}^{\infty} \gamma^t \left( \frac{1}{1 - p} - (1 - p)^t \right) \\ &= (1 - \gamma)(1 - \delta/2)(1 - p) \left( \frac{1}{(1 - p)(1 - \gamma)} - \frac{1}{1 - \gamma(1 - p)} \right) \\ &= (1 - \delta/2) \frac{p}{1 - \gamma(1 - p)}. \end{aligned}$$

Plugging in  $p = 2(1 - \gamma)/\delta$  gives

$$\begin{aligned} J(\pi_{\text{safe}}, R) &\geq (1 - \delta/2) \frac{2(1 - \gamma)/\delta}{1 - \gamma(1 - 2(1 - \gamma)/\delta)} \\ &= \frac{1 - \delta/2}{\gamma + \delta/2} \\ &\stackrel{(i)}{\geq} (1 - \delta/2)(2 - \gamma - \delta/2) \\ &\geq (1 - \delta/2)(1 - \delta/2) \\ &\geq 1 - \delta, \end{aligned}$$

which proves  $J(\pi_{\text{safe}}, R) - J(\pi, R) \geq 1 - \delta$  as desired. (i) uses the fact that  $1/x \geq 2 - x$  for  $x > 0$ .



All that remains to be shown is that

$$(1 - \gamma) \mathbb{E}_{\pi^*} \left[ \sum_{t=0}^{\infty} \gamma^t D_{\text{KL}}(\pi^*(\cdot | s_t) \| \pi_{\text{safe}}(\cdot | s_t)) \right] \geq c_2. \quad (9)$$

We can bound the LHS of (9) based on only the KL divergence at  $s_2$ :

$$(1 - \gamma) \mathbb{E}_{\pi^*} \left[ \sum_{t=0}^{\infty} \gamma^t D_{\text{KL}}(\pi^*(\cdot | s_t) \| \pi_{\text{safe}}(\cdot | s_t)) \right] \geq (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t D_{\text{KL}}(\pi^*(\cdot | s_2) \| \pi_{\text{safe}}(\cdot | s_2)) \mathbb{P}_{\pi^*}(s_t = s_2).$$

Since  $\pi^*$  always takes action  $a_2$ , we know that  $\mathbb{P}_{\pi^*}(s_t = s_2) = \mathbb{1}\{t \geq 1\}$ . Thus, we can continue the bound as

$$\begin{aligned} &\geq (1 - \gamma) \sum_{t=1}^{\infty} \gamma^t D_{\text{KL}}(\pi^*(\cdot | s_2) \| \pi_{\text{safe}}(\cdot | s_2)) \\ &= \gamma D_{\text{KL}}(\pi^*(\cdot | s_2) \| \pi_{\text{safe}}(\cdot | s_2)) \\ &= \gamma \left[ \pi^*(a_1 | s_2) \log \frac{\pi^*(a_1 | s_2)}{\pi_{\text{safe}}(a_1 | s_2)} + \pi^*(a_2 | s_2) \log \frac{\pi^*(a_2 | s_2)}{\pi_{\text{safe}}(a_2 | s_2)} \right] \\ &= \gamma \log \frac{1}{q} \\ &= c_2 \end{aligned}$$

by the definition of  $q$ . This proves (9) and completes the proof.  $\square$

## A.2. Proof of Proposition 3.2

We first prove another useful proposition:

**Proposition A.1.** *The return of a policy  $\pi$  under a reward function  $R$  is given by*

$$J(\pi, R) = \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} \mu_{\pi}(s, a) R(s, a).$$

*Proof.* Applying the definitions of return and occupancy measure, we have

$$\begin{aligned} J(\pi, R) &= (1 - \gamma) \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right] \\ &= (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} R(s, a) \mathbb{P}_{\pi}(s_t = s \wedge a_t = a) \\ &= (1 - \gamma) \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} R(s, a) \sum_{t=0}^{\infty} \gamma^t \mathbb{P}_{\pi}(s_t = s \wedge a_t = a) \\ &= \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} R(s, a) (1 - \gamma) \mathbb{E}_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t \mathbb{1}\{s_t = s \wedge a_t = a\} \right] \\ &= \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} \mu_{\pi}(s, a) R(s, a). \end{aligned}$$

$\square$

According to Proposition A.1, the return of a policy is simply a weighted sum of the reward function, where the weights are given by the occupancy measure. We now prove Proposition 3.2.

**Proposition 3.2.** *For any MDP, reward function  $R$ , and pair of policies  $\pi, \pi_{\text{safe}}$ , we have*

$$|J(\pi_{\text{safe}}, R) - J(\pi, R)| \leq \|\mu_{\pi} - \mu_{\pi_{\text{safe}}}\|_1. \quad (4)$$

*Proof.* Applying Proposition A.1, Hölder’s inequality, and the fact that  $R(s, a) \in [0, 1]$ , we have

$$\begin{aligned}
 & |J(\pi_{\text{safe}}, R) - J(\pi, R)| \\
 &= \left| \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} (\mu_{\pi_{\text{safe}}}(s, a) - \mu_{\pi}(s, a)) R(s, a) \right| \\
 &\leq \left( \max_{(s,a) \in \mathcal{S} \times \mathcal{A}} |R(s, a)| \right) \left( \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} |\mu_{\pi_{\text{safe}}}(s, a) - \mu_{\pi}(s, a)| \right) \\
 &\leq \|\mu_{\pi} - \mu_{\pi_{\text{safe}}}\|_1.
 \end{aligned}$$

□

### A.3. Additional results

The following proposition demonstrates that there is always some reward function for which the bound in (4) is tight up to a factor of two.

**Proposition A.2.** *Fix an MDP and pair of policies  $\pi, \pi_{\text{safe}}$ . Then there is some reward function  $R$  such that*

$$|J(\pi_{\text{safe}}, R) - J(\pi, R)| \geq \frac{1}{2} \|\mu_{\pi} - \mu_{\pi_{\text{safe}}}\|_1.$$

*Proof.* Define two reward functions

$$\begin{aligned}
 R_1(s, a) &= \mathbb{1}\{\mu_{\pi_{\text{safe}}}(s, a) \geq \mu_{\pi}(s, a)\} \\
 R_2(s, a) &= \mathbb{1}\{\mu_{\pi_{\text{safe}}}(s, a) \leq \mu_{\pi}(s, a)\}.
 \end{aligned}$$

Using Proposition A.1, we have

$$\begin{aligned}
 & |J(\pi_{\text{safe}}, R_1) - J(\pi, R_1)| + |J(\pi, R_2) - J(\pi_{\text{safe}}, R_2)| \\
 &\geq J(\pi_{\text{safe}}, R_1) - J(\pi, R_1) + J(\pi, R_2) - J(\pi_{\text{safe}}, R_2) \\
 &= \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} (\mu_{\pi_{\text{safe}}}(s, a) - \mu_{\pi}(s, a)) (R_1(s, a) - R_2(s, a)) \\
 &= \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} (\mu_{\pi_{\text{safe}}}(s, a) - \mu_{\pi}(s, a)) \begin{cases} 1 & \mu_{\pi_{\text{safe}}}(s, a) > \mu_{\pi}(s, a) \\ -1 & \mu_{\pi_{\text{safe}}}(s, a) < \mu_{\pi}(s, a) \\ 0 & \mu_{\pi_{\text{safe}}}(s, a) = \mu_{\pi}(s, a) \end{cases} \\
 &= \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} |\mu_{\pi_{\text{safe}}}(s, a) - \mu_{\pi}(s, a)| \\
 &= \|\mu_{\pi} - \mu_{\pi_{\text{safe}}}\|_1.
 \end{aligned}$$

Since both of the terms on the first line are positive, one must be at least  $\frac{1}{2} \|\mu_{\pi} - \mu_{\pi_{\text{safe}}}\|_1$ , which completes the proof. □

### A.4. Occupancy measure regularization in LLMs

As noted in the main text, in the current paradigm of using RLHF to train LLMs, we can show that action distribution divergence between two policies is equivalent to occupancy measure divergence. In particular, we prove the following proposition.

**Proposition A.3.** *Suppose that an environment satisfies the following conditions:*

- *It is deterministic:  $\mu_0(s_0) = 1$  for exactly one state  $s_0$ , and for all  $s_t, a_t \in \mathcal{S} \times \mathcal{A}$ ,  $p(s_{t+1} | s_t, a_t) = 1$  for exactly one state  $s_{t+1}$ .*

- Exactly one sequence of actions leads to each state: if following  $a_0, \dots, a_{t-1}$  leads to  $s$ , then no other sequence of actions (of any length) can also lead to  $s$ .

Then, for any policies  $\pi, \pi'$ , the action distribution and occupancy measure KL divergences between them are equal:

$$D_{\text{KL}}(\mu_\pi \parallel \mu_{\pi'}) = \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t D_{\text{KL}}(\pi(\cdot \mid s_t) \parallel \pi'(\cdot \mid s_t)) \right].$$

*Proof.* Given the assumptions about the environment, we can rewrite the log-occupancy measure of a state-action pair in terms of the sum of log action probabilities over the unique sequence of actions leading to that state. Suppose  $a_0, \dots, a_{t-1}$  is the unique action sequence leading to  $s$  and that this action sequence visits states  $s_0, \dots, s_{t-1}, s$ . Then

$$\begin{aligned} \log \mu_\pi(s, a) &= \log(1 - \gamma) \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t \mathbf{1}\{s_t = s \wedge a_t = a\} \right] \\ &= \log(1 - \gamma) \mathbb{P}_\pi(s_t = s \wedge a_t = a) \\ &= \log(1 - \gamma) \prod_{i=0}^t \pi(a_i \mid s_i) \\ &= \log(1 - \gamma) + \sum_{i=0}^t \log \pi(a_i \mid s_i). \end{aligned}$$

Using this, we can rewrite the occupancy measure KL divergence as

$$\begin{aligned} D_{\text{KL}}(\mu_\pi \parallel \mu_{\pi'}) &= \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} \mu_\pi(s, a) \log \left( \frac{\mu_\pi(s, a)}{\mu_{\pi'}(s, a)} \right) \\ &= (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t \sum_{a_0, \dots, a_t \in \mathcal{A}^{t+1}} \mathbb{P}_\pi(a_0 \wedge \dots \wedge a_t) \sum_{i=0}^t \left( \log \pi(a_i \mid s_i) - \log \pi'(a_i \mid s_i) \right) \\ &= (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t \sum_{a_0, \dots, a_t \in \mathcal{A}^{t+1}} \left( \prod_{j=0}^t \pi(a_j \mid s_j) \right) \sum_{i=0}^t \left( \log \pi(a_i \mid s_i) - \log \pi'(a_i \mid s_i) \right), \quad (10) \end{aligned}$$

where  $s_i$  is the state reached by taking  $a_0, \dots, a_{i-1}$ .

We will now show inductively that

$$\sum_{a_0, \dots, a_t \in \mathcal{A}^{t+1}} \left( \prod_{j=0}^t \pi(a_j \mid s_j) \right) \sum_{i=0}^t \left( \log \pi(a_i \mid s_i) - \log \pi'(a_i \mid s_i) \right) = \sum_{i=0}^t \sum_{s_i \in \mathcal{S}} \mathbb{P}_\pi(s_i) D_{\text{KL}}(\pi(\cdot \mid s_i) \parallel \pi'(\cdot \mid s_i)). \quad (11)$$

Consider first if  $t = 0$ . Then

$$\begin{aligned} &\sum_{a_0 \in \mathcal{A}} \pi(a_0 \mid s_0) \left( \log \pi(a_0 \mid s_0) - \log \pi'(a_0 \mid s_0) \right) \\ &= D_{\text{KL}}(\pi(\cdot \mid s_0) \parallel \pi'(\cdot \mid s_0)) \\ &= \mathbb{P}_\pi(s_0) D_{\text{KL}}(\pi(\cdot \mid s_0) \parallel \pi'(\cdot \mid s_0)). \end{aligned}$$

Now suppose (11) holds for  $t - 1$ . Then for  $t$  we have

$$\begin{aligned}
 & \sum_{a_0, \dots, a_t \in \mathcal{A}^{t+1}} \left( \prod_{j=0}^t \pi(a_j | s_j) \right) \sum_{i=0}^t \left( \log \pi(a_i | s_i) - \log \pi'(a_i | s_i) \right) \\
 &= \sum_{a_0, \dots, a_{t-1} \in \mathcal{A}^t} \left( \prod_{j=0}^{t-1} \pi(a_j | s_j) \right) \sum_{a_t \in \mathcal{A}} \pi(a_t | s_t) \left[ \log \pi(a_t | s_t) - \log \pi'(a_t | s_t) + \sum_{i=0}^{t-1} \left( \log \pi(a_i | s_i) - \log \pi'(a_i | s_i) \right) \right] \\
 &= \sum_{a_0, \dots, a_{t-1} \in \mathcal{A}^t} \left( \prod_{j=0}^{t-1} \pi(a_j | s_j) \right) \left[ D_{\text{KL}}(\pi(\cdot | s_t) \| \pi'(\cdot | s_t)) + \sum_{a_t \in \mathcal{A}} \pi(a_t | s_t) \sum_{i=0}^{t-1} \left( \log \pi(a_i | s_i) - \log \pi'(a_i | s_i) \right) \right] \\
 &= \sum_{a_0, \dots, a_{t-1} \in \mathcal{A}^t} \left( \prod_{j=0}^{t-1} \pi(a_j | s_j) \right) \left[ D_{\text{KL}}(\pi(\cdot | s_t) \| \pi'(\cdot | s_t)) + \sum_{i=0}^{t-1} \left( \log \pi(a_i | s_i) - \log \pi'(a_i | s_i) \right) \right] \\
 &= \sum_{s_t \in \mathcal{S}} \mathbb{P}_\pi(s_t) D_{\text{KL}}(\pi(\cdot | s_t) \| \pi'(\cdot | s_t)) + \sum_{a_0, \dots, a_{t-1} \in \mathcal{A}^t} \left( \prod_{j=0}^{t-1} \pi(a_j | s_j) \right) \sum_{i=0}^{t-1} \left( \log \pi(a_i | s_i) - \log \pi'(a_i | s_i) \right) \\
 &\stackrel{(i)}{=} \sum_{s_t \in \mathcal{S}} \mathbb{P}_\pi(s_t) D_{\text{KL}}(\pi(\cdot | s_t) \| \pi'(\cdot | s_t)) + \sum_{i=0}^{t-1} \sum_{s_i \in \mathcal{S}} \mathbb{P}_\pi(s_i) D_{\text{KL}}(\pi(\cdot | s_i) \| \pi'(\cdot | s_i)) \\
 &= \sum_{i=0}^t \sum_{s_i \in \mathcal{S}} \mathbb{P}_\pi(s_i) D_{\text{KL}}(\pi(\cdot | s_i) \| \pi'(\cdot | s_i)),
 \end{aligned}$$

where (i) is from the inductive hypothesis.

Now, plugging (11) into (10) gives

$$\begin{aligned}
 & D_{\text{KL}}(\mu_\pi \| \mu_{\pi'}) \\
 &= (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t \sum_{i=0}^t \sum_{s_i \in \mathcal{S}} \mathbb{P}_\pi(s_i) D_{\text{KL}}(\pi(\cdot | s_i) \| \pi'(\cdot | s_i)) \\
 &= (1 - \gamma) \sum_{i=0}^{\infty} \sum_{t=i}^{\infty} \gamma^t \sum_{s_i \in \mathcal{S}} \mathbb{P}_\pi(s_i) D_{\text{KL}}(\pi(\cdot | s_i) \| \pi'(\cdot | s_i)) \\
 &= (1 - \gamma) \sum_{i=0}^{\infty} \sum_{s_i \in \mathcal{S}} \mathbb{P}_\pi(s_i) D_{\text{KL}}(\pi(\cdot | s_i) \| \pi'(\cdot | s_i)) \sum_{t=i}^{\infty} \gamma^t \\
 &= (1 - \gamma) \mathbb{E}_\pi \left[ \sum_{i=0}^{\infty} D_{\text{KL}}(\pi(\cdot | s_i) \| \pi'(\cdot | s_i)) \sum_{t=i}^{\infty} \gamma^t \right] \\
 &= (1 - \gamma) \mathbb{E}_\pi \left[ \frac{\gamma^i}{1 - \gamma} \sum_{i=0}^{\infty} D_{\text{KL}}(\pi(\cdot | s_i) \| \pi'(\cdot | s_i)) \right] \\
 &= \mathbb{E}_\pi \left[ \gamma^i \sum_{i=0}^{\infty} D_{\text{KL}}(\pi(\cdot | s_i) \| \pi'(\cdot | s_i)) \right],
 \end{aligned}$$

which is the desired result.  $\square$

Proposition A.3 applies to two common MDP formulations of generating LLM responses. In the first formulation, each entire LLM response is considered a single action and then the MDP terminates. In this case, the conditions of Proposition A.3 are clearly satisfied. In the second formulation, each word generated is considered a single action, and the state consists of all previously generated words. Clearly this also satisfies the conditions of the proposition. Thus, in either case, AD and OM KL regularization are equivalent when training LLMs via RL.

However, the conditions of Proposition A.3 are unlikely to be met by many other MDPs. Many MDPs are stochastic, violating the first assumption. Even among deterministic MDPs, it is very uncommon that only a single action sequence can lead to each state. None of the environments we experiment with in the main text, and no common RL benchmarks outside of certain text generation or discrete optimization tasks, satisfy this property.

### A.5. State-only occupancy measures

In this section, we prove results for state-only occupancy measures

$$\mu_\pi(s) = (1 - \gamma) \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t \mathbb{1}\{s_t = s\} \right]$$

which are similar to our results for state-action occupancy measures. In particular, suppose that the reward function only depends on the state, i.e.,  $R(s, a) = R(s)$ . Then we can state the following propositions.

**Proposition A.4.** *The return of a policy  $\pi$  under a state-based reward function  $R$  is given by*

$$J(\pi, R) = \sum_{s \in \mathcal{S}} \mu_\pi(s) R(s).$$

*Proof.* We have

$$\begin{aligned} J(\pi, R) &= (1 - \gamma) \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t) \right] \\ &= (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t \sum_{s \in \mathcal{S}} R(s) \mathbb{P}_\pi(s_t = s) \\ &= (1 - \gamma) \sum_{s \in \mathcal{S}} R(s) \sum_{t=0}^{\infty} \gamma^t \mathbb{P}_\pi(s_t = s) \\ &= \sum_{s \in \mathcal{S}} R(s) (1 - \gamma) \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t \mathbb{1}\{s_t = s\} \right] \\ &= \sum_{s \in \mathcal{S}} \mu_\pi(s) R(s). \end{aligned}$$

□

**Proposition A.5.** *For any MDP, state-based reward function  $R$ , and pair of policies  $\pi, \pi_{safe}$ , we have*

$$|J(\pi_{safe}, R) - J(\pi, R)| \leq \|\mu_{\pi_{safe}}^s - \mu_\pi^s\|_1,$$

where  $\|\mu_{\pi_{safe}}^s - \mu_\pi^s\|_1 = \sum_{s \in \mathcal{S}} |\mu_{\pi_{safe}}(s) - \mu_\pi(s)|$ .

*Proof.* The proof proceeds via an analogous application of Hölder’s inequality as in the proof of Proposition 3.2. □

## B. Additional Results

### B.1. AUROC Curves for reward hacking detection

Occupancy measure KL is better at classifying when reward hacking is occurring than action distribution KL. We can see this as the AUROC for the OM-based detectors is closer to one than the AD-based detectors. Curves are shown in Figure 4, and the tabulated AUROC in Table 2.



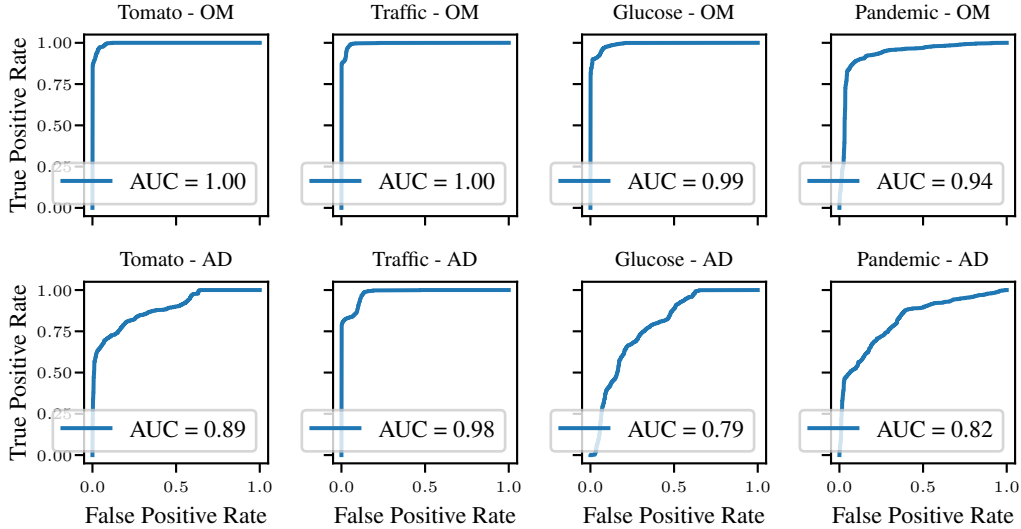


Figure 4. AUROC curves for OM and AD-based reward hacking predictors

## B.2. Detailed Results

**Regularizing towards a safe policy** Here, we provide the median true reward achieved across 5 seeds for each of the coefficients tested in each of the environments for the three regularization methods (AD, state-action OM, and state OM). As described in the main text, the coefficients that were run were determined by multiplying a range of scale-independent coefficients by the average per-timestep rewards in each environments that we calculated after running evaluation runs. The results for the tomato, traffic, glucose, and pandemic environments are in Tables 4, 5, 6, and 7 respectively.

Coefficient	AD KL	State-Action OM KL	State OM KL
0.4	$6.16 \pm 0.03$	$6.84 \pm 0.17$	$6.89 \pm 0.12$
0.08	$6.26 \pm 0.04$	$7.32 \pm 0.25$	$7.62 \pm 0.05$
0.16	$6.21 \pm 0.05$	$7.12 \pm 0.10$	$7.20 \pm 0.11$
0.8	$6.19 \pm 0.03$	$6.86 \pm 0.17$	$7.07 \pm 0.11$
1.6	$6.14 \pm 0.03$	$6.61 \pm 0.28$	$6.90 \pm 0.12$
4.0	$6.13 \pm 0.03$	$6.79 \pm 0.11$	$6.80 \pm 0.14$
8.0	$6.13 \pm 0.01$	$6.80 \pm 0.05$	$6.81 \pm 0.25$
16.0	$6.13 \pm 0.00$	$6.83 \pm 0.22$	$6.94 \pm 0.09$
0.016	$6.33 \pm 0.11$	$0.84 \pm 0.19$	$0.82 \pm 0.20$
0.04	$6.26 \pm 0.04$	$1.81 \pm 0.28$	$1.17 \pm 2.61$
0.008	$6.10 \pm 0.13$	$1.25 \pm 0.28$	$1.30 \pm 0.17$
0.004	$4.59 \pm 0.17$	$2.01 \pm 0.20$	$2.23 \pm 0.05$
0.0016	$2.98 \pm 0.30$	$1.11 \pm 0.80$	$2.31 \pm 0.06$
0.0008	$2.52 \pm 0.16$	$2.31 \pm 0.07$	$2.32 \pm 0.77$

Table 4. Tomato Results

Preventing Reward Hacking with Occupancy Measure Regularization

Coefficient	AD KL	State-Action OM KL	State OM KL
0.00025	-1334.87 ± 46.05	-1514.19 ± 86.37	-1471.37 ± 182.34
5e-05	-49992.47 ± 3616.60	-53387.70 ± 21264.93	-59958.59 ± 1479.34
0.0001	-45722.70 ± 8065.79	-1252.16 ± 62.46	-1343.77 ± 22630.61
0.0005	-1517.03 ± 36.56	-1993.42 ± 311.02	-1755.62 ± 195.70
0.001	-1733.61 ± 59.09	-2304.83 ± 1021.99	-1763.23 ± 236.91
0.0025	-1982.69 ± 60.14	-1940.36 ± 268.60	-1755.88 ± 458.19
0.005	-2145.45 ± 46.40	-2075.82 ± 544.53	-1895.66 ± 744.30
0.01	-2110.00 ± 42.60	-2144.57 ± 499.23	-2115.40 ± 893.93
1e-05	-54839.89 ± 2817.67	-58848.18 ± 2444.56	-57623.83 ± 2803.94
2.5e-05	-55095.29 ± 2365.65	-56859.38 ± 4898.74	-59319.06 ± 1223.88
5e-06	-57242.98 ± 2345.02	-61238.06 ± 1794.17	-59034.62 ± 4842.77
2.5e-06	-59583.55 ± 4325.42	-59594.74 ± 2107.35	-54590.79 ± 2826.26
1e-06	-56204.81 ± 3596.68	-61175.89 ± 2565.85	-61586.81 ± 2435.51
5e-07	-59723.52 ± 2031.10	-56360.16 ± 2290.04	-58656.01 ± 2599.17

Table 5. Traffic Results

Coefficient	AD KL	State-Action OM KL	State OM KL
0.015	-84091.61 ± 6066.60	-48884.78 ± 481.14	-82918.52 ± 5019.81
0.003	-270021.63 ± 35551.66	-101191.91 ± 4503.72	-332322.58 ± 36637.25
0.006	-154530.03 ± 4918.73	-61888.10 ± 4690.05	-158741.15 ± 25737.30
0.03	-98280.34 ± 7488.11	-49597.88 ± 1072.53	-58391.57 ± 3357.52
0.06	-88645.25 ± 11470.58	-78266.23 ± 9095.24	-58968.09 ± 6395.74
0.15	-82117.03 ± 10407.25	-106643.71 ± 17533.81	-75930.59 ± 4071.71
0.3	-73379.85 ± 8256.09	-127284.20 ± 22133.23	-98103.54 ± 14432.68
0.6	-88556.96 ± 4995.05	-118496.45 ± 9588.01	-112541.50 ± 14891.49
0.0006	-590000.85 ± 6702.13	-364253.30 ± 5221.89	-593110.06 ± 4845.44
0.0015	-459923.51 ± 102083.68	-181647.21 ± 6693.00	-511113.29 ± 18895.36
0.0003	-593615.68 ± 5323.58	-497935.91 ± 10001.19	-592025.72 ± 26247.74
0.00015	-592338.62 ± 45872.51	-577059.36 ± 10017.97	-607941.22 ± 9888.13
6e-05	-600567.81 ± 11115.49	-594716.62 ± 2981.95	-589003.48 ± 233319.14
3e-05	-598445.43 ± 35483.72	-583805.34 ± 54751.09	-604122.09 ± 9554.66

Table 6. Glucose Results

Preventing Reward Hacking with Occupancy Measure Regularization

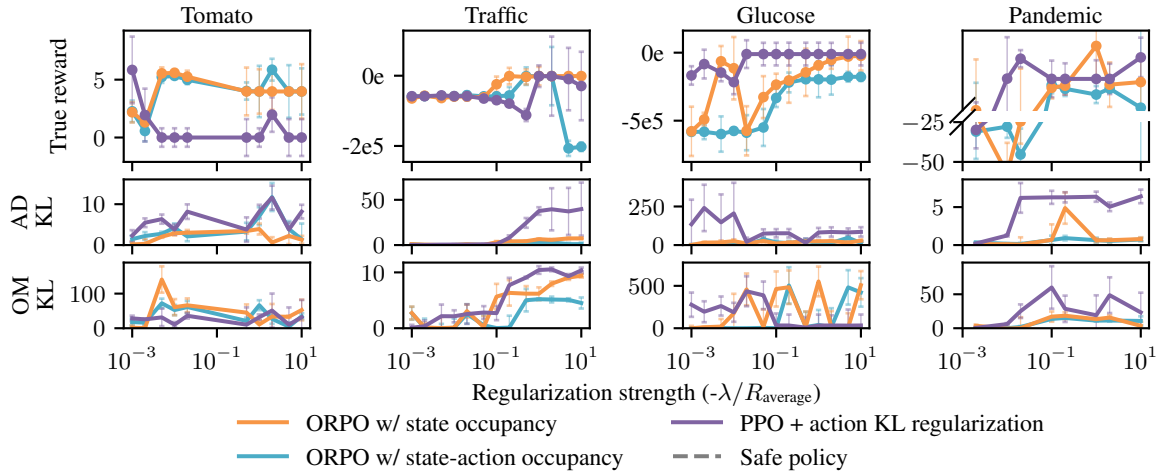


Figure 5. This plot is similar to the one shown in Figure 3, except instead of regularizing towards safe policies, we are regularizing towards reward hacking policies.

Coefficient	AD KL	State-Action OM KL	State OM KL
0.03	-12.28 ± 0.13	-11.73 ± 0.19	-11.03 ± 6.14
0.006	-12.26 ± 10.57	-29.42 ± 22.70	-58.08 ± 42.02
0.012	-12.30 ± 8.29	-11.88 ± 0.72	-10.60 ± 0.78
0.06	-12.20 ± 0.06	-12.23 ± 12.75	-10.71 ± 0.16
0.12	-12.33 ± 0.03	-12.09 ± 0.34	-10.24 ± 0.54
0.3	-12.35 ± 0.04	-12.11 ± 0.22	-11.02 ± 0.51
0.6	-12.40 ± 0.04	-12.11 ± 0.25	-10.61 ± 0.32
1.2	-12.33 ± 0.03	-12.02 ± 0.25	-10.50 ± 0.23
0.0012	-25.17 ± 9.16	-31.77 ± 5.38	-31.28 ± 7.01
0.003	-23.51 ± 6.18	-23.90 ± 11.46	-35.76 ± 9.42
0.0006	-21.85 ± 17.02	-22.40 ± 8.81	-34.56 ± 10.71
0.0003	-23.10 ± 5.04	-27.56 ± 6.71	-35.29 ± 9.23
0.00012	-30.39 ± 19.82	-19.67 ± 5.75	-41.96 ± 11.36
6e-05	-21.23 ± 8.71	-30.96 ± 20.05	-33.59 ± 8.84

Table 7. Pandemic Results

**Regularizing away from a reward hacking policy** Here, we provide the median true reward achieved across 5 seeds for each of the coefficients tested in each of the environments (tomato, traffic, and glucose) when using the three regularization methods (AD, state-action OM, and state OM) to regularize away from reward hacking policies. As described in the main text, the coefficients that were run were determined by multiplying a range of scale-independent coefficients by the average per-timestep rewards in each environments that we calculated after running evaluation runs and negating them. The results for the tomato, traffic, glucose and pandemic environments are in Tables 8, 9, 10, and 11 respectively. A plot of the best coefficients along with the divergence values is shown in Figure 5.

Preventing Reward Hacking with Occupancy Measure Regularization

Coefficient	AD KL	State-Action OM KL	State OM KL
-0.4	0.00 ± 1.59	3.98 ± 2.06	4.02 ± 0.74
-0.16	0.00 ± 1.59	4.37 ± 0.74	4.52 ± 0.63
-0.08	0.00 ± 1.59	4.98 ± 0.57	4.36 ± 0.71
-0.04	0.00 ± 0.80	4.64 ± 0.47	5.35 ± 0.61
-0.0016	1.94 ± 2.28	1.31 ± 0.92	0.56 ± 0.89
-0.0008	5.84 ± 2.88	2.18 ± 0.88	2.26 ± 0.93
-0.016	0.00 ± 0.80	5.25 ± 0.52	4.99 ± 0.37
-0.004	0.00 ± 0.80	5.54 ± 0.52	5.23 ± 0.51
-16.0	1.98 ± 1.49	3.98 ± 1.44	2.00 ± 0.01
-0.008	0.00 ± 0.80	5.59 ± 0.32	5.32 ± 0.22
-0.8	0.00 ± 0.80	3.98 ± 1.92	3.98 ± 2.22
-8.0	0.00 ± 1.59	3.98 ± 2.36	3.98 ± 2.00
-4.0	0.00 ± 1.59	3.98 ± 0.00	3.98 ± 2.27
-1.6	1.98 ± 1.49	3.98 ± 1.59	5.86 ± 0.94

Table 8. Tomato Results

Coefficient	AD KL	State-Action OM KL	State OM KL
-0.00025	-111977.53 ± 17214.26	-3540.67 ± 21315.65	-1217.29 ± 25063.82
-0.0001	-79400.16 ± 8060.00	-1063.95 ± 23432.71	-56205.18 ± 24004.46
-5e-05	-69687.41 ± 9217.50	-24407.41 ± 21935.85	-58568.36 ± 2934.04
-2.5e-05	-65045.02 ± 4714.27	-61487.52 ± 4376.05	-58749.20 ± 8501.17
-1e-06	-58401.46 ± 5709.63	-56992.75 ± 8139.89	-58166.89 ± 8260.66
-5e-07	-58461.39 ± 4383.00	-64768.07 ± 8999.20	-59487.15 ± 7486.11
-1e-05	-58061.92 ± 12651.36	-59513.49 ± 3834.08	-55291.82 ± 7638.28
-2.5e-06	-56013.23 ± 7444.14	-63617.64 ± 6862.90	-57949.39 ± 5170.07
-0.01	-28921.78 ± 97414.16	-204987.75 ± 86439.75	-5167.42 ± 54467.61
-5e-06	-58232.20 ± 2954.65	-59555.21 ± 2851.43	-58309.62 ± 6294.95
-0.0005	-1360.39 ± 30769.19	-1072.47 ± 26376.43	-1100.54 ± 43.69
-0.005	-29328.95 ± 97511.81	-1080.62 ± 11.04	-202327.06 ± 12232.67
-0.0025	-9611.75 ± 88855.63	-1066.31 ± 14.41	-206875.58 ± 21588.90
-0.001	-1265.30 ± 112637.24	-1085.87 ± 9.22	-1146.10 ± 83045.62

Table 9. Traffic Results

Coefficient	AD KL	State-Action OM KL	State OM KL
-0.015	-10431.09 ± 83660.78	-144386.41 ± 174033.93	-195231.31 ± 20347.91
-0.006	-10455.46 ± 84102.84	-204268.44 ± 164070.55	-219111.33 ± 72876.95
-0.003	-10567.31 ± 83248.45	-235054.15 ± 80412.50	-333025.09 ± 66709.74
-0.0015	-10409.22 ± 82987.79	-326774.83 ± 154161.23	-549874.74 ± 135826.64
-6e-05	-84013.55 ± 105714.03	-490927.45 ± 48825.69	-581792.14 ± 54097.78
-3e-05	-167448.46 ± 67923.39	-578315.55 ± 181073.52	-582141.72 ± 18980.46
-0.0006	-10588.67 ± 102136.84	-574612.89 ± 180942.26	-588377.02 ± 129099.43
-0.00015	-146171.68 ± 64183.33	-63295.08 ± 112176.90	-598283.71 ± 129875.42
-0.6	-10339.89 ± 83918.50	-29134.35 ± 136847.01	-186144.49 ± 17975.91
-0.0003	-216420.47 ± 89003.44	-114448.81 ± 229381.16	-575210.64 ± 23361.36
-0.03	-10360.95 ± 84058.53	-93145.02 ± 29539.90	-196008.38 ± 131953.77
-0.3	-10564.38 ± 83308.06	-23144.75 ± 108330.87	-177928.14 ± 30129.81
-0.15	-10366.68 ± 203.84	-28563.54 ± 146819.94	-177062.41 ± 36374.03
-0.06	-10409.75 ± 82863.95	-53480.07 ± 65091.05	-198117.45 ± 110658.28

Table 10. Glucose Results

Coefficient	AD KL	State-Action OM KL	State OM KL
-0.03	-12.92 ± 4.23	-5.47 ± 7.84	-16.46 ± 1.93
-0.012	-12.92 ± 3.99	-20.27 ± 7.43	-15.60 ± 5.19
-0.006	-12.92 ± 3.53	-14.46 ± 1.95	-15.15 ± 1.64
-0.003	-12.92 ± 4.14	-14.93 ± 2.41	-14.30 ± 1.87
-0.00012	-32.88 ± 10.34	-25.61 ± 10.26	-32.31 ± 7.20
-6e-05	-29.86 ± 11.63	-20.02 ± 11.50	-30.95 ± 17.15
-0.0012	-8.42 ± 3.86	-8.45 ± 2.80	-14.53 ± 19.33
-0.0003	-12.86 ± 9.37	-57.43 ± 19.42	-27.83 ± 7.03
-1.2	-12.92 ± 4.72	-5.25 ± 12.84	-14.98 ± 13.08
-0.0006	-8.35 ± 1.94	-24.58 ± 13.98	-45.36 ± 16.31
-0.06	-12.92 ± 3.43	-14.23 ± 5.02	-15.10 ± 1.01
-0.6	-12.92 ± 3.93	-9.13 ± 23.48	-14.28 ± 0.40
-0.3	-8.06 ± 4.47	-13.63 ± 5.47	-19.43 ± 68.02
-0.12	-12.92 ± 3.60	-14.29 ± 9.61	-14.47 ± 1.19

Table 11. Pandemic Results

## C. Environment details

### C.1. Tomato environment

In Figure 6, we have the setup of the tomato environment board we used for training.

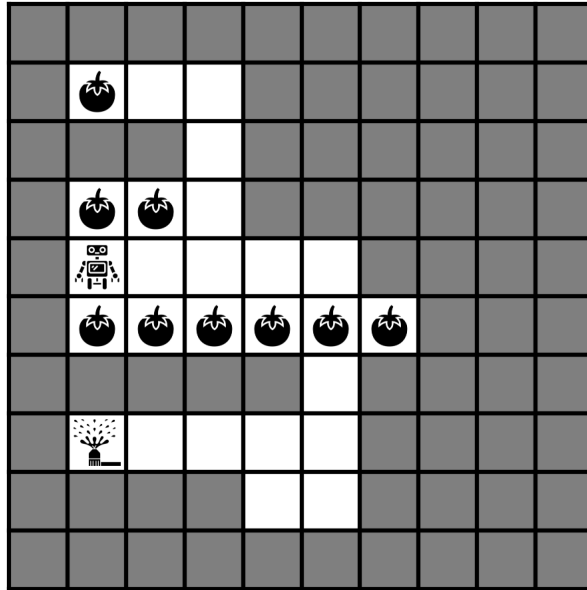


Figure 6. Here, the gray squares represent walls, and the white squares represent open spaces where the agent can travel.

The sprinkler state is down a narrow hallway, and on the other end a tomato is down another narrow hallway. We wanted to try out a scenario where the reward hacking would be relatively difficult for the agent to find to see whether or not our method works for more complex gridworld scenarios.

### C.2. Traffic environment

In Figure 7, we have a simplified rendering of the traffic flow environment merge scenario.

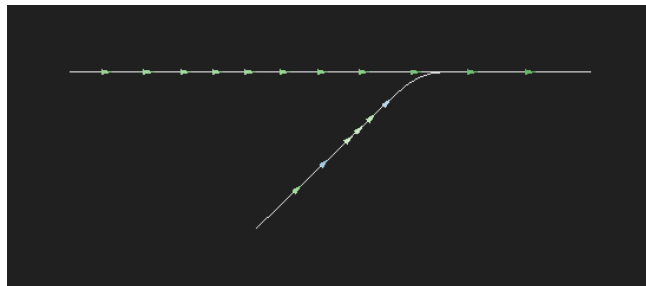


Figure 7. Here, the green cars are controlled by the human driver model IDM controller, and the blue cars are controlled by RL.

Within this particular frame, reward hacking is taking place. As we can see the blue RL vehicle has stopped completely on the on-ramp, resulting in cars to collect behind it. This way, the proxy reward, which is the average velocity of all vehicles in the simulation, is optimized as the cars on the straightway are able to continue speeding along the road without having to wait for merging cars. However, little to no true reward of the average commute time is achieved as the cars on the on-ramp aren't able to continue their commute.

## D. Experiment details

Here, we give some extra details about the architectures and hyperparameters we used for training the ORPO agents. We build ORPO using RLLib (Liang et al., 2018) and PyTorch (Paszke et al., 2019). For all RL experiments we train with 5

random seeds and report the median reward.

**Network architectures** The policy model for both the traffic and tomato environments was a simple fully connected network (FC-net) with a width of 512 and depth of 4. The policy model for the glucose environment is a basic LSTM network with 3 layers, each with widths of 64. We made this choice since the observation of the environment contains continuous historical information about the patient’s blood glucose levels and previously administered insulin. The model sizes were chosen as we found that models with these capacities empowered the agents significantly enough for them to reward hack consistently.

The discriminator model for the tomato and traffic environments was a simple FC-net with a width of 256 and depth of 4. For the glucose environment, we defined multiple configurations for the discriminator due to the continuous nature of its observation space. First, we have an option to allow for the entire history of the patient that is captured in the observation by default to be fed into the discriminator network, in which case the discriminator will be an LSTM network similar to the policy network in order to properly handle the time series data. By default, the last four hours of the patient’s state split into five minute intervals will be fed into the discriminator, but there is also an option to decrease the amount of history being used. If no history is used for the input to the discriminator network, we default to using the same FC-net used for the tomato and traffic environments. We additionally have the option of using the entire observation provided in the glucose environment (the CGM readings of the patient and the amount of insulin delivered) or just the CGM readings.

**ORPO training: tips and tricks** Naively, we can train the discriminator using the entire action and observation given by the environment and still attain impressive performance in comparison to action distribution KL regularization, but upon further experimentation, we found that different settings of the discriminator can help achieve better results with respect to the unknown true reward. In particular, with the continuous glucose environment, we found that not passing into the discriminator the patient’s entire history that is encoded in the observation provided by the environment helped performance. Intuitively, this could make sense since MDPs do not rely on history, and occupancy measures only take into account the last time step.

We also found that only feeding in the observation into the discriminator (so that effectively only the state occupancy measure is being calculated) seemed to further boost the agent’s performance on the hidden true reward as it is primarily affected by the state of the patient. Additionally, we found that selectively passing in different elements of the observation, such as just the CGM readings in the case of the glucose environment, also helped prevent reward hacking better than naively feeding in everything to the discriminator since these values are most important for the reward function.

We found that we can get more stable policies if we train the discriminator on the latest training data batches to avoid a large distribution shift when calculating occupancy measure divergences. In general, setting the KL target parameter to be smaller can also make the training runs more stable because the policies will not change too rapidly over time.

**Policy initialization** Initializing using an imitation learning policy has been shown to effectively speed up the learning process (Laidlaw et al., 2023; Uchendu et al., 2023) and is used in practice for RLHF (Stiennon et al., 2020), so we initialize our policies using the specified  $\pi_{\text{safe}}$  for the more realistic traffic, glucose, and pandemic environments.

**Note about using KL divergence over TV divergence for ORPO** When presenting our theoretical results, we choose the TV distance as it has nice theoretical properties that result in the tight bound we find. It is also preferable for our proofs since its magnitude is bounded. However, as stated at the start of 4, we rely on the KL divergence within our algorithm ORPO since it is more stable to calculate in practice. Furthermore, because Pinsker’s inequality bounds the TV distance in terms of the KL divergence, the nice theoretical properties we find for the TV distance between the occupancy measures of policies also hold for the KL divergence. Huszár (2017) used KL divergence because of its relevance to the Variational Inference literature. Specifically, KL is always differentiable, which can be useful when training structures such as GANs, whereas TV isn’t always differentiable everywhere. In addition, KL divergence’s asymmetry is actually seen as a desirable quality since it allows for the variable overestimation and underestimation of probability in different parts of the distributions.

**Hyperparameters** Some hyperparameters for the traffic environment were tuned by Pan et al. (2022). We chose the hyperparameters listed below in order to ensure that without any regularization, reward hacking will occur. This way, we can actually see if the various regularization methods actually succeed at preventing reward hacking when they are used. More details about our safe policy generation and other parameters required for training can be found within our code repository.



Hyperparameter	Tomato	Traffic	Glucose	Pandemic
Training iterations	500	250	500	260
Batch size	3000	40000	100000	3860
SGD minibatch size	128	16384	1024	64
SGD epochs per iteration	8	5	4	5
Optimizer	Adam	Adam	Adam	Adam
Learning rate	1e-3	5e-5	1e-4	0.0003
Gradient clipping	0.1	None	10	10
Discount rate ( $\gamma$ )	0.99	0.99	0.99	0.99
GAE coefficient ( $\lambda$ )	0.98	0.97	0.98	0.95
Entropy coefficient (start)	0.01	0.01	0.01	0.1
Entropy coefficient (end)	0.01	0.01	0.01	0.01
Entropy schedule horizon	0	0	0	500000
KL target	0.001	0.02	1e-3	0.01
Value function loss clipping	10	10,000	100	20
Value function loss coefficient	0.1	0.5	0.0001	0.5
Share value function layers	F	T	T	T

Table 12. PPO/ORPO hyperparameters.

Hyperparameter	Tomato	Traffic	Glucose	Pandemic
Discriminator reward clipping	1000	10	1e10	0.1
Regularization coefficient ( $\lambda$ )	Varied	Varied	Varied	Varied
Epochs for discriminator training	1	1	1	2

Table 13. ORPO-specific hyperparameters.

The coefficient  $\lambda$  that is used for determining how much regularization to apply was varied throughout the experiments and noted in our result. While our empirical results have been generated using the KL divergence, we have implemented support for the total variation (TV) and Wasserstein distances within our code. After thorough experimentation, we determined that these other divergence metrics are relatively unstable in comparison to the KL-divergence.

## E. Elaborated Related Work

### E.1. Offline RL:

Offline RL doesn’t necessarily consider any reward function, and even with knowledge of the environment’s transition dynamics or infinite amounts of data, offline RL algorithms can still perform horribly without any ground truth reward signal, resulting in catastrophic outcomes (He, 2023). Several previous offline RL theoretical results have only provided performance guarantees in the case of when the dataset actually reflects the true reward function (Cheng et al., 2022). The limitations that are addressed by offline RL methods are also separate from the problem of reward hacking that ORPO addresses. In particular, in the offline RL setting, we will practically have limited amounts of data available, whereas in our setting, we are challenged by a misspecified or “hackable” reward that can motivate unsafe behavior from the agent.

Offline RL algorithms typically optimize over the empirical transition or reward function found within the provided dataset, which is subject to estimation errors due to the limited amount of data practically available. So far, the approach to account for this estimation error has been to act pessimistically, applying different kinds of reward penalties based on the error (Rashidinejad et al., 2023); however, this pessimism can result in suboptimal policies that do not explore enough (Xie et al., 2023). Occupancy measures have been used previously in the offline RL literature; however, ORPO is unique in its emphasis on preventing *reward hacking*. For instance, algaeDICE, OptiDICE, and other methods from the DICE family have a dual objective of estimating the ratio between the occupancy measures of both the expected optimal policy and the policy under which the dataset was collected and optimizing the learned policy so that the ratio previously calculated is minimized (Lee et al., 2022). These methods do not actually calculate occupancy measures of a particular policy; instead, they use a duality

trick to approximate the corrections that will be needed to turn the offline dataset distribution into the expected optimal policy’s distribution. On the other hand, we actually calculate the occupancy measures using a discriminator and incorporate them into our algorithm as a crucial value for regularizing the learned policy to a provided safe policy, rather than trying to approximate some desired policy and adjusting the safe policy’s distribution to match that.

A recent study has shown that offline RL possesses an inherent “survival instinct” due to its pessimistic approach towards optimization and its limited access to data that renders it robust to some kinds of reward misspecifications (Li et al., 2023). However, there are several assumptions at play here, particularly regarding the type and quality of the dataset and the underlying reward function. We rely on no such assumptions, other than the fact that the safe policy is reasonably attainable and doesn’t include reward hacking activity. Thus, while offline RL does account for training and test time data distribution mismatches by remaining close to the distribution of the provided rollouts, which can sometimes prove to be robust towards misspecified reward functions, it is ultimately solving a different issue and is severely limited in its ability to prevent reward hacking.

### E.2. Entropy Maximization

Occupancy measures have also been used previously for a related subset of methods that focus on maximum entropy exploration. These methods optimize for a lower bound of the policy-induced steady-state distribution’s entropy that can then be used to define intrinsic rewards (Hazan et al., 2019; Nedergaard & Cook, 2023). Our method is fundamentally different as we are trying to regularize the behavior of the agent so that it is not only safe but also an improvement with respect to the provided safe policy, *not* artificially construct rewards for under-specified environments. Other similar works, such as state marginal matching, assume that the system designer has some knowledge about the target distribution to which the learned policy’s state distribution must be aligned (Lee et al., 2020), whereas our algorithm requires no extra input other than a reasonably specifiable policy that doesn’t exhibit reward hacking behaviors. While these methods provide an effective way to reconcile with the exploration-exploitation trade-off, they fail to guarantee the safety of the agent as it is still reasonable to expect that even with this principled approach towards exploration, the agent can find ways to hack the specified goal, since there is nothing preventing the discovery of these dangerous states.