# Hyperparameter Optimization Can Even Be Harmful in Off-Policy Learning and How to Deal with It

**Yuta Saito**[1] , **Masahiro Nomura**[2]
[1]Cornell University,
[2]CyberAgent, Inc.
ys552@cornell.edu, nomura_masahiro@cyberagent.co.jp

## Abstract

There has been a growing interest in off-policy evaluation in the literature such as recommender systems and personalized medicine. We have so far seen significant progress in developing estimators aimed at accurately estimating the effectiveness of counterfactual policies based on biased logged data. However, there are many cases where those estimators are used not only to evaluate the value of decision making policies but also to search for the best hyperparameters from a large candidate space. This work explores the latter ***hyperparameter optimization (HPO)*** task for *off-policy learning*. We empirically show that naively applying an unbiased estimator of the generalization performance as a surrogate objective in HPO can cause an unexpected failure, merely pursuing hyperparameters whose generalization performance is greatly overestimated. We then propose simple and computationally efficient corrections to the typical HPO procedure to deal with the aforementioned issues simultaneously. Empirical investigations demonstrate the effectiveness of our proposed HPO algorithm in situations where the typical procedure fails severely.

## 1 Introduction

Interactive decision making systems, such as recommender systems, produce logged data valuable for optimizing future decision making. For example, the logs of an e-commerce recommender system record which product was recommended and whether the users purchased it, giving the system designer a rich logged dataset useful for evaluating and improving the decision making quality. This type of historical data is often called *logged bandit data* and is one of the most ubiquitous forms of data available in many real-life applications [Swaminathan and Joachims, 2015a; Su *et al.*, 2020; Kiyohara *et al.*, 2021; Saito *et al.*, 2024].

*Off-Policy Learning (OPL)* aims to train a new decision making policy using only the logged bandit data. OPL is useful in that it can improve the decision making system continuously in a batch manner without requiring a risky exploration. Owing to the ubiquity of logged bandit data in the real-world, significant attention has been paid to OPL of contextual bandits [Strehl *et al.*, 2010; Swaminathan and Joachims, 2015a; Swaminathan and Joachims, 2015b; Wang *et al.*, 2017; Kallus *et al.*, 2021; Kiyohara *et al.*, 2023; Kiyohara *et al.*, 2024].

The fundamental problem in OPL is that the outcome is only observed for the action chosen by the system in the past. Thus, estimating the generalization performance of a policy is non-trivial because we cannot naively apply the empirical risk as done in typical supervised machine learning (ML). Therefore, a variety of estimators have been developed in the field of off-policy evaluation (OPE), such as Inverse Propensity Score (IPS) [Precup *et al.*, 2000] and Doubly Robust (DR) [Dudík *et al.*, 2014]. Then, a feasible approach to OPL is to maximize one such estimator as a surrogate objective using only the logged data. Hyperparameter optimization (HPO) can also be performed based on one of the estimators on a validation set of the logged data [Paine *et al.*, 2020].

In this study, we investigate how well automatic HPO algorithms work for OPL using only the available logged data. In particular, we empirically find two critical issues in HPO that have yet to be investigated in the literature, but can have a significant adverse impact on the effectiveness of the OPL pipeline. The first issue is ***optimistic bias***, which implies that the hyperparameter values selected by an HPO procedure are often the ones whose performance is greatly overestimated. In HPO, we often use an unbiased estimator as a strategy to optimize the generalization performance (primary objective) using only validation data. The problem is that, when optimizing the validation performance as a surrogate objective, HPO can identify a set of hyperparameters whose validation performance looks good but its generalization performance is detrimental. As a result, the typical HPO procedure often produces a highly sub-optimal solution, even with an unbiased estimator of the generalization performance. The second issue is ***unsafe behavior***, which suggests that the typical HPO procedure can output a solution, which *underperforms* the logging (data collection) policy, even when we set the logging policy as an initial solution. This is problematic because a logging policy is often a baseline policy to improve upon in OPL. If an HPO procedure aggravates the performance of the logging policy, there is no need to implement it in practice.

After formulating the problem in Section 2, Section 3 provides clear empirical evidence of optimistic bias and unsafe behavior. We observe these phenomena even when we use

an unbiased surrogate objective and a popular adaptive HPO algorithm. We also explain these observations theoretically, demonstrating that ignoring the fact that HPO optimizes the validation performance as a surrogate of the generalization performance can lead to a worse regret of HPO algorithms. More specifically, we identify that a heavy-tailed distribution of overestimation bias during HPO can cause an unexpected gap between the generalization and validation regret. These empirical and theoretical observations result in our proposed corrections to the typical HPO procedure, which we describe in Section 4. Finally, Section 5 conducts comprehensive experiments and demonstrates that our simple corrections can deal with the aforementioned issues and improve the typical procedure, particularly for cases where the typical procedure becomes unsafe and underperforms the logging policy.[1]

## 2 Preliminaries

We use $x \in \mathcal{X}$ to denote a context vector and $a \in \mathcal{A}$ to denote a (discrete) action such as a playlist recommendation in a music streaming service. Let $r \in [0, r_{\max}]$ denote a reward variable, which is sampled identically and independently from an unknown conditional distribution $p(r|x,a)$. A decision making policy is modeled as a distribution over the action space, i.e., $\pi : \mathcal{X} \to \Delta(\mathcal{A})$ where $\Delta(\cdot)$ is a probability simplex. We can then represent the probability of action $a$ being taken by policy $\pi$ given context $x$ as $\pi(a|x)$.

### 2.1 Off-Policy Evaluation and Learning

In OPE, we are given logged bandit data $\mathcal{D} := \{(x_i, a_i, r_i)\}_{i=1}^n$ consisting of $n$ independent draws from the *logging policy* $\pi_0$. Using this logged dataset, OPE aims to estimate the *generalization* performance of a given *evaluation policy* $\pi_e$, which is often different from $\pi_0$:

$$V(\pi_e) := \mathbb{E}_{(x,a,r)\sim p(x)\pi_e(a|x)p(r|x,a)}[r]. \tag{1}$$

This is the ground-truth performance of the evaluation policy when deployed in an environment of interest. OPE uses an estimator $\hat{V}$ to estimate $V(\pi_e)$ based only on $\mathcal{D}$ as $V(\pi_e) \approx \hat{V}(\pi_e; \mathcal{D})$. A typical choice of $\hat{V}$ is IPS:

$$\hat{V}_{\mathrm{IPS}}(\pi_e; \mathcal{D}) := \frac{1}{n}\sum_{i=1}^{n} \frac{\pi_e(a_i \,|\, x_i)}{\pi_0(a_i \,|\, x_i)} r_i,$$

where $\pi_e(a_i|x_i)/\pi_0(a_i|x_i)$ is called the importance weight. Under some assumptions for identification such as full support ($\pi_e(a|x) > 0 \to \pi_0(a|x) > 0, \ \forall(x,a)$), IPS provides an unbiased estimate of the generalization policy performance, i.e., $\mathbb{E}_{\mathcal{D}}[\hat{V}_{\mathrm{IPS}}(\pi_e; \mathcal{D})] = V(\pi_e)$. Beyond IPS, significant efforts have been made to enable a more accurate OPE from the logged data [Dudík *et al.*, 2014; Wang *et al.*, 2017; Su *et al.*, 2020; Saito *et al.*, 2023].

In OPL, we aim to learn an optimal decision making policy $\pi^* := \arg\max_{\pi} V(\pi)$ from the logged data. As in supervised ML, we cannot directly use the generalization policy performance. Instead, we use its estimator as a surrogate:

$$\hat{\pi} = \arg\max_{\pi\in\Pi} \ \hat{V}(\pi; \mathcal{D}) - \lambda \cdot \mathcal{R}(\pi),$$

where $\Pi$ is a policy class, which might be a linear class [Swaminathan and Joachims, 2015a] or deep neural nets [Joachims *et al.*, 2018]. $\mathcal{R}(\cdot)$ regularizes the complexity of the policy $\pi$, and $\lambda(\geq 0)$ is a hyperparamter that controls the effect of regularization.

**Algorithm 1** Typical HPO with IPS as a surrogate (**Baseline**)

**Input:** $A$, $\Theta$, $T$, $\mathcal{D}_{tr}$, $\mathcal{D}_{val}$
**Output:** $\hat{\theta}$

$\pi^* \leftarrow \pi_0$ , $\mathcal{S}_0 \leftarrow \emptyset$
**for** $t = 1, 2, \ldots, T$ **do**
  $\theta_t \leftarrow A(\theta \,|\, \mathcal{S}_{t-1})$ *// sample candidate hyperparameters*
  $\pi_t \leftarrow \hat{\pi}(\cdot \,|\, \theta_t, \mathcal{D}_{tr})$ *// train a policy (lower-level)*
  **if** $\hat{V}_{\mathrm{IPS}}(\pi_t; \mathcal{D}_{val}) > \hat{V}_{\mathrm{IPS}}(\pi^*; \mathcal{D}_{val})$ **then**
    $\hat{\theta} \leftarrow \theta_t$, $\pi^* \leftarrow \pi_t$ *// update the solution*
  **end if**
  $\mathcal{S}_t \leftarrow \mathcal{S}_{t-1} \cup \{(\theta_t, \hat{V}_{\mathrm{IPS}}(\pi_t; \mathcal{D}_{val}))\}$ *// store the result*
**end for**

### 2.2 Hyperparameter Optimization

OPE involves many hyperparameters to be properly tuned from those defining the policy class $\Pi$ to the regularization parameter $\lambda$. In a typical HPO procedure for OPL, we first split the original logged bandit data $\mathcal{D}$ into training ($\mathcal{D}_{tr}$) and validation ($\mathcal{D}_{val}$) sets. Then, we wish to solve the following bi-level optimization:

$$\theta^* := \arg\max_{\theta\in\Theta} \ V\big(\hat{\pi}(\cdot; \theta, \mathcal{D}_{tr})\big), \tag{2}$$

where $\Theta$ is a pre-defined hyperparamter search space. $\hat{\pi}(\cdot; \theta, \mathcal{D}_{tr})$ is a policy parameterized by a set of hyperparameters $\theta$. The model parameter of $\hat{\pi}(\cdot; \theta, \mathcal{D}_{tr})$ is trained on the training set $\mathcal{D}_{tr}$ (lower-level optimization). The problem here is that the generalization performance of $\hat{\pi}(\cdot; \theta, \mathcal{D}_{tr})$ is unknown and needs to be estimated. A feasible HPO procedure based on an estimated policy performance is:

$$\hat{\theta}(\mathcal{D}_{val}) := \arg\max_{\theta\in\Theta} \ \hat{V}\big(\hat{\pi}(\cdot; \theta, \mathcal{D}_{tr}); \mathcal{D}_{val}\big), \tag{3}$$

where the generalization performance of $\hat{\pi}(\cdot; \theta, \mathcal{D}_{tr})$ is estimated by an estimator $\hat{V}$ on the validation set $\mathcal{D}_{val}$.[2] A common choice of $\hat{V}$ is an unbiased estimator that satisfies $\mathbb{E}[\hat{V}(\pi; \mathcal{D}_{val})] = V(\pi), \forall \pi \in \Pi$ such as IPS. Then, one can apply grid search, random search [Bergstra and Bengio, 2012], or adaptive methods such as tree-structured Parzen estimator (TPE) [Bergstra *et al.*, 2011] to solve the higher-level optimization in Eq. (3) efficiently. Algorithm 1 describes this typical HPO procedure for OPL, which starts from the logging policy $\pi_0$ as its initial solution and adaptively samples promising hyperparameters via an arbitrary HPO algorithm (denoted here as $A$) [Tang and Wiens, 2021].

[1] Appendix B provides a comprehensive survey of related work. Note that the arXiv version of the paper, available at https://arxiv.org/abs/2404.15084, includes the full text with appendices.

[2] For brevity of notation, we sometimes use $V(\theta)$ and $\hat{V}(\theta; \mathcal{D})$ to denote the generalization and validation performances of the policy induced by $\theta$.

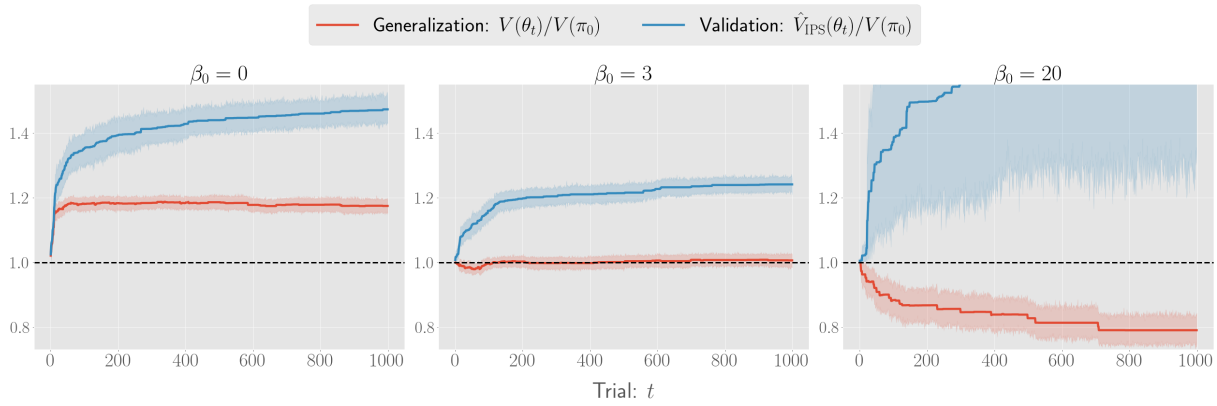


Figure 1: Empirical Evidence of Optimistic Bias and Unsafe Behavior in HPO for OPL (w/ TPE). The results are averaged over 25 runs with different seeds and then normalized by $V(\pi_0)$. The shaded regions indicate 95% confidence intervals.

## 3 Unexpected Failure in HPO for OPL

This section studies the effectiveness of HPO when applied to OPL from both empirical and theoretical perspectives.

### 3.1 Empirical Analysis

First, we conduct a synthetic experiment and provide empirical evidence of surprising failure of HPO in OPL.

**Synthetic Data**

Our empirical analysis is based on *OpenBanditPipeline* (OBP)[3], an open-source toolkit for OPE and OPL, which includes synthetic data generation modules and a range of estimators [Saito *et al.*, 2021]. We synthesize context vectors $x$ by sampling them from a 10-dimensional standard normal distribution. We then set $|\mathcal{A}| = 10$, where each action $a \in \mathcal{A}$ is characterized by a 10-dimensional representation vector $e_a$. The reward function $\mu(x, a) := \mathbb{E}[r \mid x, a]$ is defined as:

$$\mu(x, a) = \sigma\left(x^\top M e_a + \eta_x^\top x + \eta_a^\top e_a\right), \tag{4}$$

where $\sigma(z) := 1/(1 + \exp(-z))$ is the sigmoid function. $M$, $\eta_x$, and $\eta_a$ are parameter matrices or vectors for defining the synthetic reward function. These parameters are sampled from a uniform distribution with range $[-1, 1]$. After generating the synthetic reward function, we sample binary rewards from a Bernoulli distribution with parameter $\mu(x, a)$.

We then define the logging policy $\pi_0$ by applying the softmax function to the reward function $\mu(x, a)$ as follows.

$$\pi_0(a \mid x) = \frac{\exp(\beta_0 \cdot \mu(x, a))}{\sum_{a' \in \mathcal{A}} \exp(\beta_0 \cdot \mu(x, a'))}, \tag{5}$$

where $\beta_0$ is an inverse temperature parameter to control the optimality and entropy of the logging policy. A large positive value of $\beta_0$ leads to a near-deterministic and near-optimal logging policy. When $\beta_0 = 0$, $\pi_0$ is uniform.

**Policy Class and HPO Algorithms**

To train a new policy $\pi$ from only the logged data, we first estimate $\mu(x, a)$ by a supervised ML method, where the resulting estimator is denoted as $\hat{\mu}(x, a; \mathcal{D}_{tr})$. We then form a stochastic policy by applying the softmax rule as:

$$\pi(a \mid x; \theta, \mathcal{D}_{tr}) = \frac{\exp(\beta \cdot \hat{\mu}(x, a; \mathcal{D}_{tr}))}{\sum_{a' \in \mathcal{A}} \exp(\beta \cdot \hat{\mu}(x, a'; \mathcal{D}_{tr}))}, \tag{6}$$

where $\beta$ is an inverse temperature parameter to define a new policy. $\theta$ is a set of hyperparameters, which consists of $\beta$, supervised ML model to construct $\hat{\mu}$, and the hyperparameters of $\hat{\mu}$. The hyperparameter search space $\Theta$ is summarized in Table 1 in Appendix E.

As an HPO algorithm, we use TPE [Bergstra *et al.*, 2011], which is a popular adaptive method in the HPO community [Akiba *et al.*, 2019]. TPE has been shown to work well for HPO of supervised ML, however, whether it also works for OPL has never been thoroughly investigated.

**Observations**

In this synthetic experiment, we set $\beta_0 \in \{0, 3, 20\}$ and $|\mathcal{D}_{tr}| = |\mathcal{D}_{val}| = 1,000$. The number of trials ($T$ in Algorithm 1) for HPO is set to 1,000.

Figure 1 shows the **validation performance** ($\hat{V}_{\mathrm{IPS}}(\pi; \mathcal{D}_{val})$; what HPO algorithm maximizes from the logged data) and the **generalization performance** ($V(\pi)$; the primary objective of OPL) during the HPO procedure. We obtain the following key observations in this experiment.

1. **Optimistic Bias**: For all $\beta_0$, TPE succeeds in maximizing the validation performance, monotonically improving the blue lines. However, there is a substantial gap between validation and generalization, and the validation performance becomes an extremely optimistic proxy of the generalization performance. For example, when $\beta_0 = 3$, TPE does not bring any impact on the generalization performance, although the validation performance is greatly improved. This result suggests that implementing HPO is indeed a waste of time and resources for this setting.
2. **Unsafe Behavior**: When $\beta_0 = 20$ (where $\pi_0$ is already much better than uniform random), TPE outputs a solution that is significantly worse than the logging policy with respect to the generalization performance. This is problematic, as the solution at the final trial seems to provide a substantial improvement over the logging policy with respect to the unbiased validation performance

[3] https://github.com/st-tech/zr-obp

(blue lines). In reality, we have no access to the generalization performance (red lines), making it impossible to detect this performance degradation, possibly deploying an unsafe policy in the field without even noticing it.

These observations suggest that optimizing an unbiased surrogate objective is not an ideal strategy and is even harmful in some cases regarding the optimization of the generalization performance. Note that we obtain similar results when random search (RS) is used as an HPO algorithm and DR is used as an OPE estimator as reported in Appendix E. In particular, comparing RS with TPE in terms of the generalization performance, we find that there are no particular differences between the two algorithms for $\beta_0 = 0, 3$. Even more surprisingly, when $\beta_0 = 20$, TPE is outperformed by RS, even if TPE is better at optimizing the validation performance. These results further suggest that merely optimizing an unbiased surrogate objective is not a suitable approach for optimizing the generalization performance in HPO of OPL.

### 3.2 Theoretical Analysis

Next, we investigate the mechanism causing the somewhat surprising issues observed in the previous section.[4] First, we explain the phenomena from a statistical perspective.

**Proposition 3.1.** *Given that $\hat{V}$ is unbiased, we have the following inequalities.*

$$\mathbb{E}_{\mathcal{D}}\left[\hat{V}\left(\hat{\theta}(\mathcal{D}); \mathcal{D}\right)\right] \geq V\left(\theta^*\right) \geq \mathbb{E}_{\mathcal{D}}\left[V\left(\hat{\theta}(\mathcal{D})\right)\right], \quad (7)$$

*where $\mathbb{E}_{\mathcal{D}}[\cdot]$ takes expectation over every randomness in the logged data $\mathcal{D}$, and $\mathbb{E}_{\mathcal{D}}[\hat{V}(\hat{\theta}(\mathcal{D}); \mathcal{D})] - \mathbb{E}_{\mathcal{D}}[V(\hat{\theta}(\mathcal{D}))]$ is the amount of optimistic bias.*

Note that, in Eq. (2), $V\left(\theta^*\right)$ is defined as the best generalization performance we could achieve with HPO. Thus, the first inequality in Eq. (7) suggests that the *validation* performance of the HPO solution $\hat{\theta}(\mathcal{D}_{val})$ is better than the best achievable generalization performance in expectation, suggesting that the performance estimation of the HPO solution is optimistic in general. In addition, the second inequality in Eq. (7) implies that the *generalization* performance of $\hat{\theta}(\mathcal{D}_{val})$ is worse than the best achievable generalization performance in expectation, even though the validation performance of $\hat{\theta}(\mathcal{D}_{val})$ is likely to be better. As a result, we will often be disappointed with the performance of the HPO solution $\hat{\theta}$ even with an unbiased surrogate (validation) objective. Overall, Proposition 3.1 explains the substantial gap between the blue ($\mathbb{E}[\hat{V}(\hat{\theta}(\mathcal{D}_{val}))]$) and red ($\mathbb{E}[V(\hat{\theta}(\mathcal{D}_{val}))]$) lines observed in Figure 1.

Next, we analyze "regret" to understand *what causes the optimistic bias in Proposition 3.1 and how we can deal with it*. For this, we define two variants of regret, which measure the difference between the validation or generalization performances of the optimal hyperparameter and HPO solution.

$$r_{gen}(T; A, \mathcal{D}) := V\left(\theta^*\right) - V(\hat{\theta}_{T,A}(\mathcal{D})), \quad (8)$$

$$r_{val}(T; A, \mathcal{D}) := \hat{V}_{\mathrm{IPS}}(\hat{\theta}^*; \mathcal{D}) - \hat{V}_{\mathrm{IPS}}(\hat{\theta}_{T,A}(\mathcal{D}); \mathcal{D}), \quad (9)$$

[4]Appendix C provides proofs omitted in the main text.

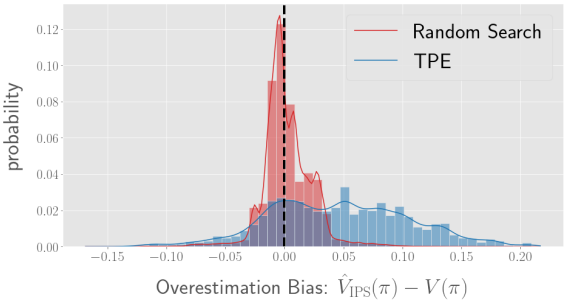


Figure 2: Distributions of Overestimation Bias ($\beta_0 = 3$)

where $\theta^* := \arg\max_{\theta \in \Theta} V(\theta)$ is the optimal hyperparameter with respect to the generalization performance. $\hat{\theta}^* = \arg\max_{\theta \in \Theta} \hat{V}_{\mathrm{IPS}}(\theta; \mathcal{D})$ denotes the optimal hyperparameter with respect to the *validation* performance, and $\hat{\theta}_{T,A}(\mathcal{D})$ is the solution of Algorithm 1 given budget $T$ and algorithm $A$. We also define the overestimation bias for a specific hyperparameter $\theta$ as $\tau(\theta; \mathcal{D}) = \hat{V}_{\mathrm{IPS}}(\theta; \mathcal{D}) - V(\theta)$. Then, the following implies that a **heavy-tailed distribution of overestimation bias during HPO can produce an unexpected gap between the generalization and validation regret**.

**Proposition 3.2.** *Given HPO algorithm $A$, budget $T$, and logged data $\mathcal{D}$, the generalization regret can be written as*

$$r_{gen}(T; A, \mathcal{D}) = r_{val}(T; A, \mathcal{D}) + \Delta\tau(\hat{\theta}_{T,A}(\mathcal{D}), \theta^*; \mathcal{D}) + C, \quad (10)$$

*where $\Delta\tau(\theta_1, \theta_2; \mathcal{D}) := \tau(\theta_1; \mathcal{D}) - \tau(\theta_2; \mathcal{D})$, and $C := \hat{V}_{\mathrm{IPS}}(\theta^*; \mathcal{D}) - \hat{V}_{\mathrm{IPS}}(\hat{\theta}^*; \mathcal{D})$.*

Only the first two terms of the RHS in Eq. (10) depend on the HPO solution $\hat{\theta}_{T,A}(\mathcal{D})$, and are thus critical for analyzing the HPO performance. The first term $r_{val}$ is the validation regret. Under some mild conditions, we can achieve *no-regret* ($r_{val}(T; A, \mathcal{D}) = o(1)$) with optimization methods such as GP-UCB [Srinivas *et al.*, 2010], as we can target the validation performance directly using available data. The second term $\Delta\tau(\hat{\theta}_{T,A}(\mathcal{D}), \theta^*; \mathcal{D})$ is the difference in the extent of overestimation between $\hat{\theta}_{T,A}(\mathcal{D})$ and $\theta^*$. When the extent of overestimation of $\hat{\theta}_{T,A}(\mathcal{D})$ is larger than that of $\theta^*$, $\Delta\tau(\hat{\theta}_{T,A}(\mathcal{D}), \theta^*; \mathcal{D})$ becomes large. Therefore, Proposition 3.2 suggests that **the overestimation bias of $\hat{\theta}_{T,A}(\mathcal{D})$ can exacerbate the generalization regret of HPO algorithms**. More specifically, if an HPO algorithm is likely to sample many hyperparameters whose performance is overestimated ($\hat{V}_{\mathrm{IPS}}(\theta) - V(\theta) > 0$) and the overestimation bias has a heavy-tailed distribution, the second term of Eq. (10) tends to become large, so does the generalization regret $r_{gen}$. Given this regret analysis, we investigate the distributions of overestimation bias observed in the empirical analysis in Figure 2. This figure implies that TPE more frequently samples hyperparameters incurring a large overestimation bias than RS. According to Proposition 3.2, this is why we do not find the advantage of TPE with respect to the generalization performance. RS has a

worse validation regret than TPE, while overestimation bias of RS is not very problematic compared to TPE. As a result, RS performs similarly to or slightly better than TPE in terms of the generalization performance. In this way, the heavy-tailed distribution of overestimation bias makes the generalization regret of HPO algorithms (in particular TPE) worse than its validation regret, resulting in optimistic bias and possibly unsafe behavior.

## 4 How Should We Deal with the Issues?

In this section, we propose two simple corrections, namely (i) **conservative surrogate objective** and (ii) **adaptive imitation regularization**, to deal with the critical issues in HPO. We also describe the resulting HPO procedure, which we call ***Conservative and Imitation-Regularized HPO (CIR-HPO)***.

### 4.1 Conservative Surrogate Objective (CSO)

First, we address the heavy-tailed distribution of overestimation bias ($\hat{V}_{\mathrm{IPS}}(\pi) - V(\pi)$) during HPO, as suggested in Figure 2. Proposition 3.2 implies that the overestimation of the value of hyperparameters sampled during HPO can exacerbate the generalization regret of an HPO algorithm. To deal with this issue, we introduce ***conservative surrogate objective***, which penalizes the validation performance of hyperparameters whose performance has a large uncertainty to avoid the issue of overestimation bias during HPO. Specifically, we propose to use a high probability lower bound of the generalization performance (denoted as $\hat{V}_{-}(\cdot)$) as an alternative surrogate objective, which is given as: $\mathbb{P}\big(V(\pi) \geq \hat{V}_{-}(\pi; \mathcal{D}, \delta)\big) \geq 1-\delta$ where $\delta \in (0, 1)$ specifies a confidence level.

A prevalent strategy to construct $\hat{V}_{-}(\cdot)$ in OPE is to apply a concentration inequality such as Hoeffding and Bernstein [Thomas *et al.*, 2015b; Thomas *et al.*, 2015a]. A problem is that these inequalities are often overly conservative as they make no assumptions about underlying distribution. Thus, we use an alternative strategy to construct $\hat{V}_{-}(\cdot)$ based on the Student's t-distribution as follows.

$$\hat{V}^t_{-}(\pi; \mathcal{D}, \delta) := \hat{V}_{\mathrm{IPS}}(\pi; \mathcal{D}) - t_{1-\delta,\nu} \sqrt{\frac{\mathbb{V}_n(\hat{V}_{\mathrm{IPS}}(\pi; \mathcal{D}))}{n-1}}, \tag{11}$$

where $t_{1-\delta,\nu}$ is the T-value given confidence level $\delta$ and degrees of freedom $\nu$.

The upside of Eq. (11) is that it produces a tighter lower bound than aforementioned concentration inequalities. This is because Eq. (11) introduces the additional assumption that the mean of importance weighted rewards $(\pi/\pi_0)r$ is normally distributed. This assumption is reasonable with growing data sizes. However, $(\pi/\pi_0)r$ often follows a distribution with heavy upper tails, which may make the assumption invalid in a small sample setting. Nonetheless, Appendix E empirically verifies that Eq. (11) is *reasonably tight* compared with other popular concentration inequalities.

### 4.2 Adaptive Imitation Regularization (AIR)

The second technique we propose is ***adaptive imitation regularization***, which tackles the unsafe behavior of the typical procedure. The issue of unsafe behavior suggests that, if logging policy $\pi_0$ is better than uniform random or is near-optimal, Algorithm 1 can produce a solution whose performance is much worse than that of the logging policy. Avoiding this problem is non-trivial, because we do not have access to the generalization performance and do not know the optimality of the logging policy in practice. For example, simply setting $\pi_0$ as an initial solution does not solve the issue at all, as suggested in Section 3.1. An instant idea might be to *imitate* the logging policy to some extent:

$$\pi_t(a|x; \alpha, \theta_t, \mathcal{D}_{tr}) = (1-\alpha)\hat{\pi}(a|x; \theta_t, \mathcal{D}_{tr}) + \alpha\pi_0(a|x), \tag{12}$$

where $\theta_t$ is a set of hyperparameters sampled at the $t$-th trial. $\alpha\,(\in [0, 1])$ is a regularization parameter, which mixes the policy induced by $\theta_t$ and $\pi_0$ to construct a policy to evaluate. A large value of $\alpha$ makes $\pi_t$ closer to the logging policy, possibly avoiding the unsafe behavior. However, if the logging policy is detrimental, we should use a small $\alpha$ so that we can avoid an unnecessary performance degradation. So, a natural question to ask here is: *how should we set the regularization parameter $\alpha$?* Again, this problem is non-trivial, as the optimality of the logging policy is unknown when performing HPO.

To overcome this difficulty in correctly setting $\alpha$, we propose *adaptively tuning this parameter over the course of HPO*. Based on the previous discussion, we should apply a strong regularization if $\pi_0$ performs well, otherwise we should not imitate $\pi_0$. A key idea here is that we can reason about the optimality of the logging policy by comparing it with solutions sampled during HPO, i.e., $\{\hat{\pi}(a|x; \theta_t, \mathcal{D}_{tr})\}_{t=1}^{T}$. If most of the sampled solutions underperform $\pi_0$, we can infer that the logging policy is well-performing. To make a valid comparison between the sampled solutions and the logging policy, we apply a Student's t-test based on the following T-value.

$$T(\pi_1, \pi_2) := \frac{|\Delta\hat{V}_{\mathrm{IPS}}(\pi_1, \pi_2)|}{\sqrt{\hat{\mathbb{V}}_n(\Delta\hat{V}_{\mathrm{IPS}}(\pi_1, \pi_2))/(n-1)}}.$$

where $\Delta\hat{V}_{\mathrm{IPS}}(\pi_1, \pi_2) := \hat{V}_{\mathrm{IPS}}(\pi_1) - \hat{V}_{\mathrm{IPS}}(\pi_2)$ is the performance difference between the two policies estimated by IPS. Given a null hypothesis ($\Delta\hat{V}_{\mathrm{IPS}}(\pi_1, \pi_2) = 0$) and a normality assumption, $T(\pi_1, \pi_2)$ follows a t-distribution with $\nu$ degrees of freedom. We then calculate the optimality score of $\pi_0$ at

**Algorithm 2** Conservative and Imitation-Regularized HPO

**Input:** $A$, $\delta$, $\gamma$, $\Theta$, $T$, $\pi_0$, $\mathcal{D}_{tr}$, $\mathcal{D}_{val}$
**Output:** $\hat{\theta}$

$\mathcal{S}_0 \leftarrow \emptyset$
**for** $t = 1, 2, \ldots, T$ **do**
  $\theta_t \leftarrow A(\theta \,|\, \mathcal{S}_{t-1})$ *// sample candidate hyperparameters*
  $\hat{\pi}_t \leftarrow \hat{\pi}(\cdot \,|\, \theta_t, \mathcal{D}_{tr})$ *// train a policy (lower-level)*
  $\pi_t \leftarrow (1-\alpha_t)\cdot\hat{\pi}_t + \alpha_t \cdot \pi_0$ *// regularization (Eq. (14))*
  **if** $\hat{V}^t_{-}(\pi_t; \mathcal{D}_{val}, \delta) \geq \hat{V}^t_{-}(\pi^*; \mathcal{D}_{val}, \delta)$ **then**
    $\hat{\theta} \leftarrow \theta_t$, $\pi^* \leftarrow \pi_t$ *// update the solution*
  **end if**
  $\mathcal{S}_t \leftarrow \mathcal{S}_{t-1} \cup \{(\theta_t, \hat{V}^t_{-}(\pi_t; \mathcal{D}_{val}, \delta))\}$ *// store the result*
**end for**

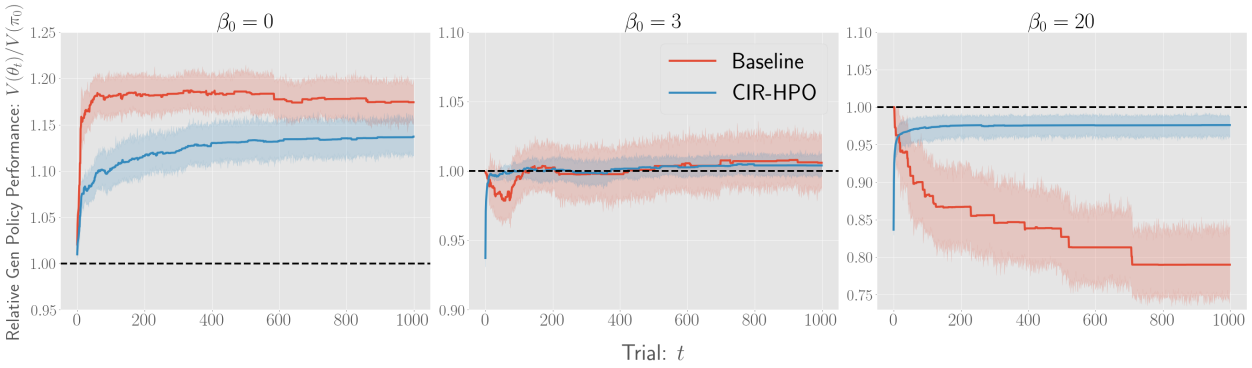


Figure 3: Comparing CIR-HPO (our proposal) and Baseline by their generalization performance. The results are averaged over 25 runs with different seeds and then normalized by $V(\pi_0)$. The shaded regions indicate 95% confidence intervals.

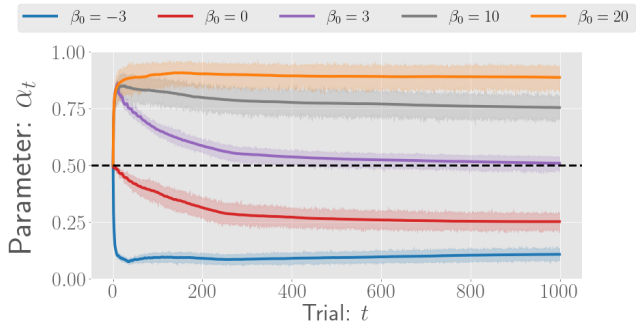


Figure 4: Behavior of adaptive regularization parameter ($\alpha_t$) of CIR-HPO with varying values of $\beta_0 \in \{-3, 0, 3, 10, 20\}$.

the $t$-th trial as follows.

$$s_t = \begin{cases} 1 & (T(\pi_0, \pi_t) \geq t_{1-\delta/2,\nu} \text{ and } \Delta\hat{V}_{\mathrm{IPS}}(\pi_0, \pi_t) \geq 0) \\ -1 & (T(\pi_0, \pi_t) \geq t_{1-\delta/2,\nu} \text{ and } \Delta\hat{V}_{\mathrm{IPS}}(\pi_0, \pi_t) < 0) \\ 0 & (\textit{otherwise, i.e., } T(\pi_0, \pi_t) < t_{1-\delta/2,\nu}) \end{cases} \tag{13}$$

$s_t$ indicates whether $\pi_0$ is better or worse than $\pi_t$ in a significant level. If $\pi_0$ is better than $\pi_t$, then $s_t = 1$. Instead, $s_t = -1$ if $\pi_0$ is tested to be worse. If there is no significant difference between $\pi_0$ and $\pi_t$, the score is zero.

Using the sequence of scores up to the $t$-th trial, i.e., $\{s_{t'}\}_{t'=1}^{t}$, we define *adaptive regularization parameter* as:

$$\alpha_t := \alpha_{init} + (1 - \alpha_{init}) \cdot \left(\frac{t}{T}\right)^{\gamma} \cdot \frac{\sum_{t'=1}^{t} s_{t'}}{t} \tag{14}$$

where $\alpha_{init} \in [0, 1]$ is an initial regularization parameter and $\gamma\,(> 0)$ is a scheduling parameter for adaptive regularization. For example, suppose that $s_t = 1, \forall t = 1, 2, \ldots, T$, meaning that $\pi_0$ is always better than $\pi_t$ in a significant level. Then, following Eq. (14), $\alpha_T = 1$ and the HPO procedure outputs $\pi_0$, because it should be near-optimal. On the other hand, if $s_t = -1, \forall t = 1, 2, \ldots, T$, meaning that $\pi_0$ is always worse than $\pi_t$ in a significant level, then $\alpha_T = 0$ and the HPO procedure does not imitate the logging policy at all, because it should be a bad policy.

### 4.3 The CIR-HPO Algorithm

Algorithm 2 describes the CIR-HPO algorithm, which leverages conservative surrogate objective (lines 6 and 9) and adaptive imitation regularization (line 5). $\delta$ and $\gamma$ are *meta hyperparameters*. $\delta$ controls how conservative we would like to be during HPO, and $\gamma$ controls the scheduling of the adaptive regularization. In Section 5, we show that these configurations have some impact on the behavior of CIR-HPO, but we also demonstrate that the default values ($\delta = 0.1$ and $\gamma = 0.01$) work reasonably well in a range of experiment settings. The other inputs are the same as those of Algorithm 1. Note that our algorithm is easy to implement with a few additional lines of code and there is no additional computational overhead compared to the typical procedure in Algorithm 1.

## 5 Empirical Evaluation

This section empirically compares **Baseline** (Algorithm 1) and **CIR-HPO** (Algorithm 2), employing the same synthetic data and policy class as in Section 3.1. Note that we compare CIR-HPO against only **Baseline** because there is no other method proposed for HPO using logged bandit data (comprehensive summary of related work can be found in Appendix B).

### 5.1 Baseline vs CIR-HPO

Figure 3 compares the performance of **Baseline** and **CIR-HPO** with varying logging policies ($\beta_0 \in \{0, 3, 20\}$). First, when $\beta_0 = 0$ where the logging policy is uniform random, both Baseline and CIR-HPO work reasonably well and succeed in finding a set of hyperparameters that leads to a policy much better than the logging policy. What is notable for this setting is that CIR-HPO is inefficient and slow to converge compared to Baseline due to adaptive imitation regularization, even though it reaches far above the black horizontal line ($V(\pi_0)$). At the initial stage of HPO, we do not know how close the logging policy is to the optimal policy. Therefore, the proposed procedure gradually learns the optimality of the logging policy, potentially leading to a slower convergence if the logging policy is far from optimal (such as uniform random). Next, when $\beta_0 = 3$ where the logging policy is better than uniform random, but is not close to the optimal, both Baseline and CIR-HPO slightly improve the logging policy. However, the confidence intervals indicate that CIR-HPO is

much more stable than Baseline. In particular, Baseline is much more likely to *underperform* the logging policy, even though it outperforms the logging policy on average. Finally, when $\beta_0 = 20$ where the logging policy is near-optimal, Baseline outputs a solution that is substantially worse than the logging policy, even though it starts from the logging policy as its initial solution. In contrast, CIR-HPO learns that the logging policy is near-optimal during HPO and strengthens the imitation regularization adaptively. As a result, it prevents the solution from being significantly worse than the (already near-optimal) logging policy, which is compelling, because we do not know the optimality of the logging policy in advance. Figure 4 illustrates the behavior of adaptive imitation regularization, which suggests that it succeeds in controlling the strength of regularization depending on the optimality of the logging policy.

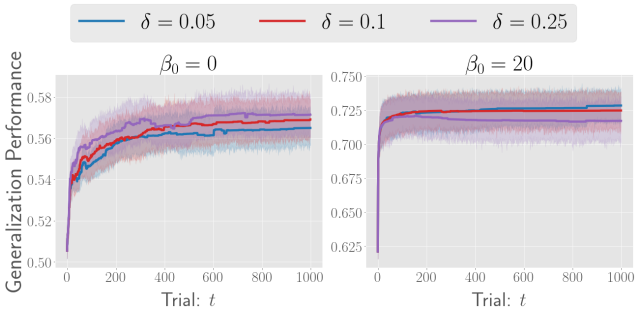


Figure 5: Sensitivity of the generalization performance of CIR-HPO regarding the choice of $\delta$.

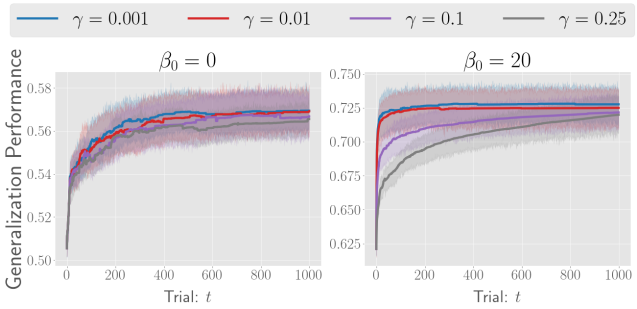


Figure 6: Sensitivity of the generalization performance of CIR-HPO regarding the choice of $\gamma$.

## 5.2 Choice of Meta Hyperparameters

Next, we evaluate the sensitivity of CIR-HPO to the choice of its meta hyperparameters. Figure 5 shows that the effectiveness of CIR-HPO with different values of $\delta$. The result demonstrates that there is no significant difference among the three values, suggesting that we do not have to care too much about which value to use for $\delta$. In addition, Figure 6 evaluates different values of $\gamma$, which controls the scheduling of adaptive imitation regularization. This result implies that, for a sub-optimal logging policy ($\beta_0 = 0$), the choice of $\gamma$ has no significant effect on the behavior of CIR-HPO. For a near-optimal logging policy ($\beta_0 = 20$), however, a smaller $\gamma$ leads to a faster convergence, although all values achieve the same level of performance in the final stage.

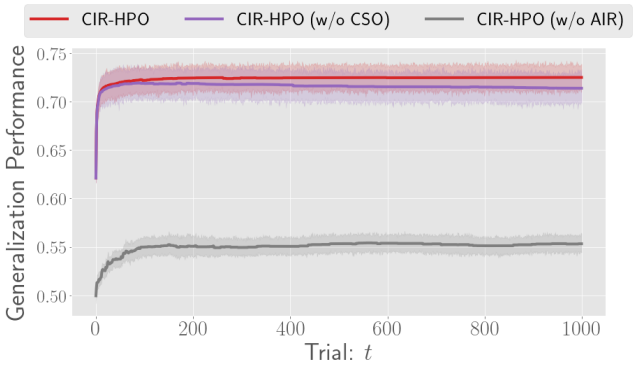


Figure 7: Ablation study of CIR-HPO ($\beta_0 = 20$).

## 5.3 Ablation Study

We also conduct an ablation study to evaluate the contribution of **conservative surrogate objective (CSO)** and **adaptive imitation regularization (AIR)** to the effectiveness of CIR-HPO. To this end, we compare **CIR-HPO** to **CIR-HPO (w/o CSO)** and **CIR-HPO (w/o AIR)** in Figure 7. The result demonstrates that both CSO and AIR clearly contribute to the performance of CIR-HPO, while AIR has a more appealing effect (CSO and AIR provide 1.6% and 23.6% improvements, respectively, in terms of the final generalization performance).

## 5.4 A Real-World Experiment

In addition to the synthetic experiments, we apply CIR-HPO to the Open Bandit Dataset [Saito *et al.*, 2021], a publicly available logged bandit dataset collected on a large-scale fashion e-commerce platform. The results suggest that CIR-HPO leads to a better policy compared to the Baseline procedure in terms of the generalization performance, providing a further argument regarding its real-world applicability. The experiment detail and results can be found in Appendix A.

# 6 Conclusion

This work studies the effectiveness of the typical HPO procedure in the OPL setup from both empirical and theoretical perspectives and found that it can fail and even be harmful. In particular, we investigated two surprising issues, namely optimistic bias and unsafe behavior, and showed that a heavy-tailed distribution of overestimation can cause an unexpected gap between validation and generalization. In response, we made two extremely simple corrections to the typical HPO procedure, resulting in the CIR-HPO algorithm, to deal with the issues. Extensive experiments demonstrated that CIR-HPO can be advantageous, particularly when the conventional procedure collapses and causes a significant and undetectable deterioration in the generalization performance.

# References

[Akiba *et al.*, 2019] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A

next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, pages 2623–2631, New York, NY, USA, 2019. ACM.

[Bergstra and Bengio, 2012] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2), 2012.

[Bergstra *et al.*, 2011] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24, 2011.

[Dudík *et al.*, 2014] Miroslav Dudík, Dumitru Erhan, John Langford, and Lihong Li. Doubly Robust Policy Evaluation and Optimization. *Statistical Science*, 29:485–511, 2014.

[Joachims *et al.*, 2018] Thorsten Joachims, Adith Swaminathan, and Maarten de Rijke. Deep learning with logged bandit feedback. In *International Conference on Learning Representations*, 2018.

[Kallus *et al.*, 2021] Nathan Kallus, Yuta Saito, and Masatoshi Uehara. Optimal off-policy evaluation from multiple logging policies. In *International Conference on Machine Learning*, pages 5247–5256. PMLR, 2021.

[Kiyohara *et al.*, 2021] Haruka Kiyohara, Kosuke Kawakami, and Yuta Saito. Accelerating offline reinforcement learning application in real-time bidding and recommendation: Potential use of simulation. *arXiv preprint arXiv:2109.08331*, 2021.

[Kiyohara *et al.*, 2023] Haruka Kiyohara, Ren Kishimoto, Kosuke Kawakami, Ken Kobayashi, Kazuhide Nakata, and Yuta Saito. Towards assessing and benchmarking risk-return tradeoff of off-policy evaluation. In *The Twelfth International Conference on Learning Representations*, 2023.

[Kiyohara *et al.*, 2024] Haruka Kiyohara, Masahiro Nomura, and Yuta Saito. Off-policy evaluation of slate bandit policies via optimizing abstraction. *arXiv preprint arXiv:2402.02171*, 2024.

[Paine *et al.*, 2020] Tom Le Paine, Cosmin Paduraru, Andrea Michi, Caglar Gulcehre, Konrad Zolna, Alexander Novikov, Ziyu Wang, and Nando de Freitas. Hyperparameter selection for offline reinforcement learning. *arXiv preprint arXiv:2007.09055*, 2020.

[Precup *et al.*, 2000] Doina Precup, Richard S. Sutton, and Satinder Singh. Eligibility Traces for Off-Policy Policy Evaluation. In *Proceedings of the 17th International Conference on Machine Learning*, pages 759–766, 2000.

[Saito *et al.*, 2021] Yuta Saito, Shunsuke Aihara, Megumi Matsutani, and Yusuke Narita. Open bandit dataset and pipeline: Towards realistic and reproducible off-policy evaluation. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

[Saito *et al.*, 2023] Yuta Saito, Ren Qingyang, and Thorsten Joachims. Off-policy evaluation for large action spaces via conjunct effect modeling. In *International Conference on Machine Learning*, pages 29734–29759. PMLR, 2023.

[Saito *et al.*, 2024] Yuta Saito, Jihan Yao, and Thorsten Joachims. Potec: Off-policy learning for large action spaces via two-stage policy decomposition. *arXiv preprint arXiv:2402.06151*, 2024.

[Srinivas *et al.*, 2010] Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pages 1015–1022, 2010.

[Strehl *et al.*, 2010] Alex Strehl, John Langford, Lihong Li, and Sham M Kakade. Learning from Logged Implicit Exploration Data. In *Advances in Neural Information Processing Systems*, volume 23, pages 2217–2225, 2010.

[Su *et al.*, 2020] Yi Su, Maria Dimakopoulou, Akshay Krishnamurthy, and Miroslav Dudík. Doubly robust off-policy evaluation with shrinkage. In *International Conference on Machine Learning*, volume 119, pages 9167–9176. PMLR, 2020.

[Swaminathan and Joachims, 2015a] Adith Swaminathan and Thorsten Joachims. Batch learning from logged bandit feedback through counterfactual risk minimization. *The Journal of Machine Learning Research*, 16(1):1731–1755, 2015.

[Swaminathan and Joachims, 2015b] Adith Swaminathan and Thorsten Joachims. The self-normalized estimator for counterfactual learning. In *Advances in Neural Information Processing Systems*, volume 28, pages 3231–3239, 2015.

[Tang and Wiens, 2021] Shengpu Tang and Jenna Wiens. Model selection for offline reinforcement learning: Practical considerations for healthcare settings. *arXiv preprint arXiv:2107.11003*, 2021.

[Thomas *et al.*, 2015a] Philip Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High-confidence off-policy evaluation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

[Thomas *et al.*, 2015b] Philip Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High confidence policy improvement. In *Proceedings of the 32th International Conference on Machine Learning*, volume 37, pages 2380–2388, 2015.

[Wang *et al.*, 2017] Yu-Xiang Wang, Alekh Agarwal, and Miroslav Dudik. Optimal and Adaptive Off-policy Evaluation in Contextual Bandits. In *Proceedings of the 34th International Conference on Machine Learning*, pages 3589–3597, 2017.