Distributionally Robust Policy Gradient for Offline Contextual Bandits

Zhouhao Yang[*]
Shanghai Jiao Tong University
yangzhouhao@sjtu.edu.cn

Yihong Guo[*]
Johns Hopkins University
yguo80@jhu.edu

Pan Xu
Duke University
pan.xu@duke.edu

Anqi Liu
Johns Hopkins University
aliu@cs.jhu.edu

Anima Anandkumar
California Institute of Technology
anima@caltech.edu

Abstract

Learning an optimal policy from offline data is notoriously challenging, which requires the evaluation of the learning policy using data pre-collected from a static logging policy. We study the policy optimization problem in offline contextual bandits using policy gradient methods. We employ a distributionally robust policy gradient method, DROPO, to account for the distributional shift between the static logging policy and the learning policy in policy gradient. Our approach conservatively estimates the conditional reward distributional and updates the policy accordingly. We show that our algorithm converges to a stationary point with rate $O(1/T)$, where T is the number of time steps. We conduct experiments on real-world datasets under various scenarios of logging policies to compare our proposed algorithm with baseline methods in offline contextual bandits. We also propose a variant of our algorithm, DROPO-exp, to further improve the performance when a limited amount of online interaction is allowed. Our results demonstrate the effectiveness and robustness of the proposed algorithms, especially under heavily biased offline data.

[*]Zhouhao Yang and Yihong Guo contributed equally to this work.

Proceedings of the 26[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2023, Valencia, Spain. PMLR: Volume 206.

1 INTRODUCTION

Contextual bandits are ubiquitous in sequential decision problems, ranging from recommendation systems (Li et al., 2011) to clinical trials (Lu et al., 2021). In contextual bandits, a learning agent interacts with the environment to select the action that gives the largest reward based on the context observed for the given items. The agent usually aims to find an optimal policy that maximizes its cumulative reward along a time horizon (Radlinski et al., 2008; Cesa-Bianchi et al., 2013). Due to only observing the reward for the selected action at each time step, the agent will need to explore different actions that could lead to higher long-term accumulative rewards despite the myopic action suggested by the short-term empirical reward prediction. This leads to a long line of research in efficient and effective exploration in online contextual bandits (Li et al., 2010, 2011; Agarwal et al., 2014; Agrawal and Goyal, 2013; Lale et al., 2019; Xu et al., 2022).

However, exploration is not always feasible or affordable, especially in applications where collecting new data is expensive, slow, or even high-risk (Levine et al., 2020). For example, switching to a new policy could damage users' experience in news recommendation systems (Li et al., 2011); in policy optimization for the robotic control, a new policy can lead to the crash of the device (Liu et al., 2020); the treatment exploration on patients has very domain-specific safety constraints in health care. Therefore, learning from an offline dataset becomes more imperative in contextual bandit problems in these and many other domains. In particular, offline contextual bandits require the agent to be able to evaluate, improve, and optimize its policy through only a pre-collected dataset. The policy used for the data collection is usually referred to as the behavior policy or the logging policy (Precup, 2000; Thomas and Brunskill, 2016).

Policies are often evaluated in contextual bandits by their expected reward (a.k.a, the value function). However, this becomes much more challenging in the offline setting be-

cause we cannot roll out new data using the evaluation policy, which causes the distributional shift issue between the logging policy and the evaluation policy. To overcome this issue, a fruitful line of research has been focused on the problem of off-policy evaluation (OPE), including Inverse Propensity Score (IPS) (Horvitz and Thompson, 1952; Swaminathan and Joachims, 2015b), Direct Method (DM) (Beygelzimer and Langford, 2009), and Doubly Robust (DR) estimators (Robins and Rotnitzky, 1995; Dudík et al., 2011; Wang et al., 2017; Su et al., 2020) in the policy evaluation. However, the performance of these OPE methods is highly dependent on the setup and the specific dataset (Voloshin et al., 2019). More recent and effective OPE methods are focused on distributionally robust learning (Liu et al., 2019; Si et al., 2020b; Kallus et al., 2022), which overcomes the distributional shift and improves the robustness of the evaluation by conservatively estimating the reward mapping.

Given the abundant literature on OPE, a natural question arises: *how can we optimize the policy in offline contextual bandits?* This leads to the off-policy optimization (OPO) problem. A typical method for policy optimization is policy gradient (Sutton et al., 1999), where the policy is parameterized as a distribution $\pi_{\boldsymbol{\theta}}(\cdot)$ over the action space and the parameter $\boldsymbol{\theta}$ is optimized via gradient ascent type of algorithms. Similar to OPE, the main challenge in OPO is that the computation of policy gradient entails an accurate evaluation of the reward function, whereas we only have a biased one due to the distributional shift. As a result, we may end up updating the policy with inaccurate gradient estimates and fail to converge to the optimal policy.

In this work, we propose the distributionally robust offline policy optimization (DROPO) algorithm for offline contextual bandits. Specifically, we view the distributional shift problem in offline contextual bandits from a covariate shift perspective, where the training and test data follow different distributions, but the conditional distribution of the output given an input remains unchanged (Sugiyama et al., 2007). This reformulation into a covariate shift problem is straightforward and natural for offline contextual bandits, where the reward distribution conditional on any input context-action pair is fixed but the offline dataset and evaluation dataset differ (Chen et al., 2016; Liu et al., 2019). We perform a distributionally robust regression to learn the reward distribution and subsequently update the policy with the policy gradient (Sutton et al., 1999).

The most related work to ours is by Si et al. (2020b), who formulates the distributional shift problem into a perturbation of the joint distribution of context, action, and reward. They then search for an optimal target policy robust to the worst-case joint distribution within this perturbation. However, the joint distributional shift assumption is a general one, inducing a potentially large uncertainty set. This could lead to over-conservativeness in practice. In later sections, we show that our DROPO algorithm outperforms their algorithm both theoretically and empirically.

In summary, our contributions are as follows.

- We propose the distributionally robust offline policy optimization (DROPO) algorithm for offline contextual bandits, which consists of two important components: first, we perform robust regression on the current parameterized policy by solving a covariate shift problem (Chen et al., 2016; Liu et al., 2019); then, the policy is updated with the policy gradient method.
- We provide theoretical analysis for the convergence of our method under standard assumptions. In particular, we show that the generalization error of our policy gradient estimator consists of an $O(1/N^{1/2})$ term and other terms coming from the offline dataset and model approximation error, where N is the logging dataset size. Moreover, we prove that the policy parameter converges to a stationary point with an $O(1/T)$ rate up to the aforementioned statistical error of the policy gradient estimator, where T stands for the total time step.
- We conduct experiments on real-world machine learning datasets including *Optdigits*, *MNIST*, *CIFAR-10* and *Adult*. We compare our algorithm DROPO with baseline algorithms for offline contextual bandits. We show that our method is much more robust than baseline methods and outperforms them in terms of regret minimization in different levels of distributional shift. The comparison is more pronounced under large distributional shifts. Furthermore, we propose a variant of DROPO when limited online exploration is permitted (named as DROPO-exp). We observe advantages in training with limited online data, especially in the large distributional shift case, where the performance is significantly improved with only a small amount of online exploration data.

2 RELATED WORK

OPE in Contextual Bandits There has been growing literature on OPE in contextual bandits (Li et al., 2010; Krause and Ong, 2011; Yue et al., 2012; Tang et al., 2013; Bottou et al., 2013) and reinforcement learning (Precup, 2000; Lei et al., 2014; Thomas et al., 2015; Thomas and Brunskill, 2016; Tennenholtz et al., 2020). Generally, three kinds of methods address the OPE problem, which are also used as the baseline model for comparison in our experiments. The first one is *Direct Method* (DM) (Beygelzimer and Langford, 2009), which learns a mapping from the context-action pairs to the reward over the observed dataset. It has low variance but high bias due to the distributional shift between the logging policy and the evaluation policy. The second one is *Inverse Propensity Score* (IPS) (Horvitz and Thompson, 1952; Swaminathan et al., 2017), which uses importance weighting to correct the mismatch between the logging and evaluation policy. It is an unbiased estimator

when the logging policy is known but it has a large variance, especially when the distributional shift is large. The third method *Doubly Robust* (DR) (Robins and Rotnitzky, 1995; Bang and Robins, 2005; Dudík et al., 2011; Wang et al., 2017; Su et al., 2019) is a combination of them and leverages the strength of DM (low variance) and IPS (no bias when logging policy is known). However, DR is still sensitive to the IPS. Many variants are developed to account for extreme values in the weights (Wang et al., 2017; Su et al., 2020; Mahmood et al., 2014; Farajtabar et al., 2018). A recent benchmark experiment on OPE shows that the performance of these OPE methods is highly dependent on the setup and the specific dataset (Voloshin et al., 2019; Fu et al., 2021). In our paper, we use DM, IPS and DR as our baseline models. In our approach, we first perform OPE for our current policy and then estimate the policy gradient based on the OPE result for updating the policy.

OPO in Contextual Bandits There are a few ways to do the off-policy optimization in contextual bandits. Similar to OPE, we can estimate the missing bandit feedback by supervised learning methods (Zadrozny et al., 2003; Beygelzimer and Langford, 2009). Propensity scores (Bottou et al., 2013; Swaminathan and Joachims, 2015a) and doubly robust (Zhao et al., 2015) can also be used to derive unbiased estimators from the exploration logs. Recently, pessimistic learning is also adopted when the offline dataset has insufficient coverage over the state and action spaces (Nguyen-Tang et al., 2022; Li et al., 2022). OPO is also closely related to causal inference (Zhang et al., 2012; Kitagawa and Tetenov, 2018) and causal perspectives have been incorporated in OPO as well (Kallus and Zhou, 2018). Recently, it becomes more important to study real-world problems in high-dimensional data (Joachims et al., 2018) and application systems (Chernozhukov et al., 2019; Pan et al., 2019).

Policy gradient is a direct way to optimize the objective of bandits and reinforcement learning. The policy is assumed to be parameterized by a set of parameter Θ. Typically, we can estimate the gradient with Monte Carlo simulations (Sutton et al., 1999; Kakade, 2001; Schulman et al., 2015), but it requires the on-policy sampling. In the off-policy setting (Silver et al., 2014; Chen et al., 2019; Kallus and Uehara, 2020), many works consider cases where it is possible to sample abundant data from the environment in an online fashion. In our work, we mainly discuss the pure offline setting, where we only have the historical data or very limited online data collection. We apply a distributionally robust method to estimate the policy gradients for optimizing the policy.

Distributionally Robust Learning Distributionally robust learning (DRL) tries to minimize the worst-case expected loss on target data (Chen et al., 2016; Gao et al., 2017; Mohajerin Esfahani and Kuhn, 2018; Chen and Paschalidis, 2018; Shafieezadeh-Abadeh et al., 2019; Gao and Kleywegt, 2022). It usually involves a minimax game with the target distribution playing an adversarial role and constrained by the source training data. The constraints are usually the perturbed sets of the distribution in terms of KL-divergence (Si et al., 2020b; Kallus et al., 2022), Wasserstein distance (Kuhn et al., 2019), total variation distance (Dixit et al., 2022) or f-divergence (Kallus et al., 2022). This idea of 'distributional robustness' has also been applied to the offline bandit evaluation and learning problems (Smirnova et al., 2019; Liu et al., 2019; Si et al., 2020b,a; Sakhi et al., 2020; Kallus et al., 2022). The most related work to ours is by Si et al. (2020b); Kallus et al. (2022). They estimate the robust value function of a target policy based on a perturbation of the joint distribution of context, action, and reward. Similar ideas have also been studied in the contextual pricing policy evaluation problem (Biggs et al., 2021), where the goal is to find the minimum variance loss function within a class of unbiased loss functions, while the demand estimation is perfect or chosen by an adversary.

Different from them, our method explicitly models the covariate shift between the logging policy and the target policy and focuses on the estimation of biased reward. In our approach, the adversarial player is the shared conditional reward distribution instead of the joint distribution (Si et al., 2020a). The conditional reward distribution given the context and action is usually invariant regardless of the different distributions of actions in the off-policy optimization. This further leads to a nice closed form of the solution of the robust regression problem (Chen et al., 2016; Liu et al., 2019). In contrast, the uncertainty set that constrains the joint distribution of context, action and reward in Si et al. (2020b) may be too general to capture the specific type of distributional shift between the logging policy and the evaluation policy, resulting in poor practical performance. Moreover, there is no convergence guarantee for their algorithm which is also backed by our experimental results.

There are a line of work formulating DRL under covariate shift in classification problems (Liu and Ziebart, 2014; Wang et al., 2015; Fathony et al., 2018; Rezaei et al., 2020, 2021). Our work is built on the robust regression under covariate shift (Chen et al., 2016) such that our reward estimation is robust to the worst-case possible data-generating conditionals in the logging policy data.

3 PROBLEM SETUP

In contextual bandits, at the t-th round, for each context vector $x_t \in \mathcal{X}$, the agent needs to choose an action $a_t \in \mathcal{A}(x_t)$ and then observes a reward $r(x_t, a_t)$. The agent's strategy of choosing actions is called a policy, which is usually represented by a mapping $\pi(a|x)$ from $\mathcal{X} \times \mathcal{A}$ to $\mathbb{R}$. The objective of policy optimization is to find a policy π that maximizes the value function defined as follows.

$$V^\pi = \mathbb{E}_{x\sim P_x, a\sim\pi(\cdot|x)}[r(x,a)],$$

where P_x is the distribution of the context that is fixed in this paper, and $r(x, a)$ could be noisy given any context-action pair. We use $\boldsymbol{\theta} \in \mathbb{R}^d$ to parameterize the policy such that $\int_a \pi_{\boldsymbol{\theta}}(a|x)\mathrm{d}a = 1$ for all $x \in \mathcal{X}$. Then the gradient of the value function is given by

$$\nabla_{\boldsymbol{\theta}} V^{\pi_{\boldsymbol{\theta}}} = \mathbb{E}_{x\sim P_x(\cdot), a\sim\pi_{\boldsymbol{\theta}}(\cdot|x)}[\nabla \log \pi_{\boldsymbol{\theta}}(a|x) r(x, a)]. \tag{3.1}$$

In the online setting, we can generate a dataset $\mathcal{D}_{\pi_{\boldsymbol{\theta}},N} = \{(x_i, a_i, r(x_i, a_i))\}_{i=1}^N$ by following the target policy $\pi_{\boldsymbol{\theta}}(\cdot|x)$. Then the online gradient estimator is given by

$$g_{\text{online}}(\boldsymbol{\theta}) = \sum_{i\in\mathcal{D}_{\pi_{\boldsymbol{\theta}},N}} \frac{\nabla \log \pi_{\boldsymbol{\theta}}(a_i|x_i) r(x_i, a_i)}{|\mathcal{D}_{\pi_{\boldsymbol{\theta}},N}|}, \tag{3.2}$$

which is an unbiased Monte Carlo estimator for $\nabla_{\boldsymbol{\theta}} V^{\pi_{\boldsymbol{\theta}}}$. Based on the gradient estimator in (3.2), policy gradient updates the policy parameter by $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta g_{\text{online}}(\boldsymbol{\theta}_t)$.

In this paper, we study the problem of offline policy optimization, where during the learning process we only have access to an offline dataset $\mathcal{D}_{\log} = \{(x_i, a_i, r(x_i, a_i))\}$ generated by a specific logging policy $\beta(a|x)$, $x \in \mathcal{X}, a \in \mathcal{A}$. In other words, we are not allowed to interact with the environment. Therefore, we can only construct the gradient estimator based on the logging dataset $\mathcal{D}_{\log}$, which may follow a different distribution from $\mathcal{D}_{\pi_{\boldsymbol{\theta}},N}$.

Off-policy problems as covariate shift When the goal is to estimate reward, the distributional shift between the logging policy that generates the offline data and the potential evaluation policy is a covariate shift. As the conditional reward distribution is usually invariant. That is why the most common way of dealing with this distribution mismatch between the datasets is to use importance weights between policies, which yields the IPS estimator:

$$g_{\text{IPS}}(\boldsymbol{\theta}) = \sum_{i\in\mathcal{D}_{\log}} \frac{\nabla \log \pi_{\boldsymbol{\theta}}(a_i|x_i) r(x_i, a_i)}{|\mathcal{D}_{\log}|} \frac{\pi_{\boldsymbol{\theta}}(a_i|x_i)}{\beta(a_i|x_i)},$$

where $\beta(\cdot|x)$ is the logging policy generating the dataset. The IPS estimator is known as an unbiased estimator for $\nabla_{\boldsymbol{\theta}} V^{\pi_{\boldsymbol{\theta}}}$, which is suggested as follows:

$$\begin{aligned}&\mathbb{E}_{x\sim P_x, a\sim\beta(\cdot|x)}[g_{\text{IPS}}(\boldsymbol{\theta})] \\ &= \mathbb{E}_{x\sim P_x, a\sim\pi(\cdot|x)}[\nabla \log \pi_{\boldsymbol{\theta}}(a_i|x_i) r(x_i, a_i)],\end{aligned}$$

which, however, causes instability and high variance in practice because the importance weight $\frac{\pi(a|x)}{\beta(a|x)}$ could be extremely large at data points that have good coverage in the target policy π but are seldomly chosen by the source policy β. In this work, we aim to avoid directly using importance weights in the gradient estimator by learning a robust reward distribution estimator.

4 DISTRIBUTIONALLY ROBUST POLICY OPTIMIZATION

In this section, we present our main algorithms (Algorithm 1 and Algorithm 2) for learning distributional robust policy gradient and a convergence analysis. We start with the distributionally robust policy gradient estimator.

4.1 Distributionally Robust Policy Gradient Estimator

We now write the policy gradient estimator in the previous section in a more general form. Given the context distribution P_x and evaluation policy $\pi_{\boldsymbol{\theta}}(\cdot|x)$, we can simulate context-action pairs $\mathcal{S}_{\pi_{\boldsymbol{\theta}}} = \{(x_i, a_i)\}$ and use the following gradient estimator to approximate $\nabla_{\boldsymbol{\theta}} V^{\pi_{\boldsymbol{\theta}}}$:

$$\hat{g}(\boldsymbol{\theta}) = \frac{1}{|\mathcal{S}_{\pi_{\boldsymbol{\theta}}}|} \sum_{i\in\mathcal{S}_{\pi_{\boldsymbol{\theta}}}} \nabla \log \pi_{\boldsymbol{\theta}}(a_i|x_i) \hat{r}_{\mathcal{D}_{\log}}(x, a), \tag{4.1}$$

where $\hat{r}_{\mathcal{D}_{\log}}(x, a)$ is a reward estimator. If we choose $\hat{r}_{\mathcal{D}_{\log}}(x, a)$ as the true reward for action context pair (x, a), we recover the online estimator in (3.2). Similarly, if we use the reward from the logging dataset with an importance weight, we recover the IPS estimator. Based on the reward estimator and the policy gradient estimator in (4.1), we can update the policy parameter accordingly.

In this paper, we propose a distributionally robust policy gradient estimator that utilizes a distributionally robust reward estimator $\hat{r}_{\mathcal{D}_{\log}}(x, a)$. Here Direct Method is employed to learn the reward estimator, considering the fact that when we know the target policy $\pi_{\boldsymbol{\theta}}$ and the context-action pair that we want to query the reward for, the most straightforward approach is to learn a reward mapping $r_{\mathcal{D}_{\log}} : \mathcal{X} \times \mathcal{A} \mapsto \mathbb{R}$ based on the logging dataset $\mathcal{D}_{\log}$. We perform distributionally robust learning to estimate a reward distribution that is robust to the worst-case data-generating distribution compatible with the logging dataset. We then utilize the mean of the reward distribution as the reward estimator in (4.1).

4.2 Distributionally Robust Learning for Reward Distribution Estimation

To find the robust reward distribution, we formulate the learning problem as finding the solution of the following min-max problem introduced by Liu et al. (2019).

$$\min_{f(\cdot|x,a)} \max_{h(\cdot|x,a)\in\Sigma_{\mathcal{D}_{\log}}} \mathbb{E}_{a\sim\pi}[\mathcal{L}(f(r|x, a), h(r|x, a))], \tag{4.2}$$

where f is the reward distribution we aim to learn, h is a distribution from the constraint set $\Sigma_{\mathcal{D}_{\log}}$ that imposes consistency with the empirical dataset $\mathcal{D}_{\log}$, and $\mathcal{L}$ is the loss function. Note that h is chosen as the worst possible distribution in $\Sigma_{\mathcal{D}_{\log}}$ that could adversarially affect the learning distribution f. We omit the notation x in the expectation for

simplicity as the marginal covariate distribution does not change in our setting. Therefore, the resulting reward estimator f is robust to the worst-case possible data-generating distribution h that satisfies the constraints in $\Sigma_{\mathcal{D}_{\log}}$. In the sequel, we elaborate on the details of $\mathcal{L}$ and $\Sigma_{\mathcal{D}_{\log}}$.

Loss function The loss function $\mathcal{L}$ is chosen as the relative log-loss between two distributions as follows:

$$\begin{aligned}&\mathcal{L}(f(r|x,a), h(r|x,a))\\ &= \mathbb{E}_{r\sim h(\cdot|x,a)}[-\log f(r|x,a) + \log f_0(r|x,a)],\end{aligned}$$

where f_0 is a base conditional distribution for the reward that serves as a prior. $\mathcal{L}$ represents the difference between the cross entropy of a reward distribution predictor f and a baseline conditional distribution f_0 with respect to h on the evaluation data distribution. This loss function essentially accounts for the amount of expected "surprise" in modeling true data distribution $P_x\pi(a|x)h(r|x,a)$ that comes from $P_x\pi(a|x)f(r|x,a)$ rather than $P_x\pi(a|x)f_0(r|x,a)$. Since the logging data is not always informative in minimizing the loss function on the evaluation data, we set an upper bound for f's loss using f_0. The choice of f_0 is usually based on prior knowledge. In practice, we assume f_0 to be a Gaussian distribution, where we choose $\frac{r_{\max}+r_{\min}}{2}$ as the mean of f_0.

Uncertainty Set We constrain the adversary reward distribution h into the following uncertainty set $\Sigma_{\mathcal{D}_{\log}}$:

$$\begin{aligned}\Sigma_{\mathcal{D}_{\log}} \triangleq \Bigg\{h : \big|&\mathbb{E}_{a\sim\beta, r\sim h}[[r \quad \phi(x,a)][r \quad \phi(x,a)]^\top]\\ &- \frac{1}{|\mathcal{D}_{log}|}\sum_{i\in\mathcal{D}_{log}}[r_i \quad \phi(x_i,a_i)][r_i \quad \phi(x_i,a_i)]^\top\big| \le \lambda\Bigg\},\end{aligned}$$

where $\phi(x,a)$ is a user-defined feature vector (e.g., the last layer of the neural network), $[r \quad \phi(\boldsymbol{x},a)]$ is the vector augmented with the reward, and λ is a hyperparameter controlling the size of the constraint set. This constraint means that the expectation of a quadratic function in reward r under the adversary distribution h must be consistent with the empirical expectation (the average) of the feature function in logging data. The reason we choose this quadratic function is that it induces an exponential family distribution as the solution of the min-max problem (4.2). More specifically, according to Chen et al. (2016); Liu et al. (2019), when the base distribution f_0 in (4.2) is Gaussian, the solution to the distributionally robust learning problem is also a Gaussian distribution.

Compared with other DRL work like Si et al. (2020b), where a joint distribution over context, action and reward is modeled, our uncertainty set only focuses on modeling the shared conditional reward distribution between logging policy and evaluation policy, which is a more natural and practical choice. In addition, the feature moment-matching uncertainty set enables an analytical solution to the min-max game, which we will show in Proposition 4.1.

Parametric Formulation The following proposition provides a closed-form solution to problem (4.2).

Proposition 4.1 (Appendix A in Liu et al. (2019))**.** *Assume we choose a Gaussian base distribution in* (4.2)*, i.e., $f_0 \sim \mathcal{N}(\mu_0, \sigma_0^2)$. Let f be parameterized by $\rho = (\rho_r, \boldsymbol{\rho}_x) \in \mathbb{R}^{1+d}$. Then the min-max problem in* (4.2) *has a closed form solution f_ρ, which is a Gaussian distribution $\mathcal{N}(\mu_\rho(x,a), \sigma_\rho^2(x,a))$, where*

$$\sigma_\rho^2(x,a) = \left(2\frac{\beta(a|x)}{\pi(a|x)}\rho_r + \sigma_0^{-2}\right)^{-1}, \tag{4.3}$$

$$\mu_\rho(x,a) = \sigma_\rho^2(x,a)\left(-2\frac{\beta(a|x)}{\pi(a|x)}\boldsymbol{\rho}_{\boldsymbol{x}}\phi(x,a) + \mu_0\sigma_0^{-2}\right). \tag{4.4}$$

Here, ρ can be obtained via maximum conditional log likelihood estimation:

$$\rho = \arg\max_\rho \mathbb{E}_{a\sim\pi, r\sim h}\big[\log f_\rho(x,a)\big]. \tag{4.5}$$

The derivation involves using strong duality to switch the $\min$ problem and the $\max$ problem and applying the Lagrangian multiplier ρ to convert the constrained problem into an unconstrained problem. Note that given the form of the reward estimator f_ρ, (4.5) is convex with respect to ρ. Thus the closed form solution translates our min-max problem into a parameter learning problem, which is much easier to solve than the min-max problem in Si et al. (2020b); Kallus et al. (2022). Moreover, the subsequent algorithm benefits from this closed form solution in clarity and practicality, since the convergence criterion of parameters is rather vague otherwise.

Property of the Prediction This analytic form utilizes the ratio between the two policies β and π to adjust the uncertainty of the reward prediction. Note that this density ratio is the inverse of the importance weight in IPS. Therefore, it does not cause a high variance issue as in IPS. For IPS, if $\beta(a|x) \ll \pi(a|x)$, which means the action is rarely seen in logging dataset but commonly seen in the evaluation, the reward estimator will up-weight its reward with extremely high values, causing a high variance. However, in our approach, if an action is rarely seen in the logging policy, the ratio is close to 0. σ_ρ^2 would consequently be bounded by σ_0^2, and μ_ρ would be close to μ_0. This indicates that the logging data cannot supervise the reward estimation for an action that has scarce support under $\beta(a|x)$. Moreover, if an action is well-covered by the logged data, the prediction would be very confident with μ_ρ, which heavily depends on the parameter $\boldsymbol{\rho}_{\boldsymbol{x}}$ learned from data.

Parameter Learning We take the derivative of $\mathbb{E}_{a\sim\pi, r\sim h}\big[\log f_\rho(x,a)\big]$ with respect to the parameters ρ_r and $\boldsymbol{\rho}_{\boldsymbol{x}}$ based on the offline dataset, which are denoted as $\widetilde{\nabla}_{\rho_r}$ and $\widetilde{\nabla}_{\boldsymbol{\rho}_{\boldsymbol{x}}}$ respectively. We have:

$$\widetilde{\nabla}_{\rho_r} \triangleq \widetilde{\nabla}_{\rho_r}\mathbb{E}_{a\sim\pi, r\sim h}\big[\log f_\rho(x,a)\big]$$

$$= \frac{1}{|\mathcal{D}_{log}|} \sum_{i \in \mathcal{D}_{log}} r_i^2 - \mu_\rho^2(x_i, a_i) - \sigma_\rho^2(x_i, a_i),$$

$$\widetilde{\nabla}_{\boldsymbol{\rho_x}} \triangleq \widetilde{\nabla}_{\boldsymbol{\rho_x}} \mathbb{E}_{a\sim\pi, r\sim h}\big[\log f_\rho(x, a)\big]$$
$$= \frac{1}{|\mathcal{D}_{log}|} \sum_{i \in \mathcal{D}_{log}} (r_i - \mu_\rho(x_i, a_i))\phi(x_i, a_i).$$

We then learn the parameters ρ_r and $\boldsymbol{\rho}_x$ using gradient descent. The final algorithm is displayed in Algorithm 1. In every iteration (taking the t-th iteration for example), we first solve problem (4.5) and attain the estimated reward distribution $f_{\rho_t} \sim \mathcal{N}(\mu_t, \sigma_t^2)$. Then we sample new context-action pairs with the current evaluation policy $\pi_{\boldsymbol{\theta}_t}$ and update the policy parameter $\boldsymbol{\theta}_t$ with gradient ascent. Here, the gradient is calculated based on (A.1), and $\hat{r}_{\mathcal{D}_{\log}}$ is substituted by μ_t. Note that even though the objective function is the expected target likelihood, the gradients are only associated with logging data samples. We refer more details about this result to Chen et al. (2016).

Algorithm 1 Distributionally Robust Offline Policy Optimization (DROPO)

Inputs: Training data $\mathcal{S}$ with training samples $\{x_i, a_i, r_i\}_{i=1}^{|\mathcal{S}|}$, learning rates γ and η, ρ^{rr}, ρ^{rx}
for $t = 0, \ldots, T$ **do**
  for $k = 0, \ldots, K-1$ **do**
    $\rho_{t,k+1}^{rr} = \rho_{t,k}^{rr} - \gamma \widetilde{\nabla}_{\rho_r}$
    $\rho_{t,k+1}^{rx} = \rho_{t,k}^{rx} - \gamma \widetilde{\nabla}_{\boldsymbol{\rho_x}}$
  end for
  $\sigma_t^2 = \big(2\frac{\beta(a|x)}{\pi(a|x)}\rho_{t,K}^{rr} + \sigma_0^{-2}\big)^{-1}$
  $\mu_t = \sigma_t^2\big(-2\frac{\beta(a|x)}{\pi(a|x)}\rho_{t,K}^{rx}\phi(x,a) + \mu_0\sigma_0^{-2}\big)$
  Sample context-action pairs $\mathcal{D}_{\pi_{\boldsymbol{\theta}_t}}$ from $\pi_{\boldsymbol{\theta}_t}$
  Update $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta\hat{g}(\boldsymbol{\theta}_t)$, where $\hat{g}(\boldsymbol{\theta}_t) = \frac{1}{|\mathcal{D}_{\pi_{\boldsymbol{\theta}_t}}|}\sum_{(a_i,x_i)\in\mathcal{D}_{\pi_{\boldsymbol{\theta}_t}}} \nabla \log \pi_{\boldsymbol{\theta}_t}(a_i|x_i)\mu_t(x_i, a_i)$
end for
Output: π_{θ_T}

4.3 Convergence Analysis of the Proposed Algorithm

Now we provide a convergence analysis for Algorithm 1.

First, we lay down the following assumptions, on which our main theorems are based. In assumption 4.2, we bound the log-density of the policy function $\pi_{\boldsymbol{\theta}}$.

Assumption 4.2. *Let $\pi_{\boldsymbol{\theta}}(a|x)$ be the policy of an agent observing context x. There exists a constant $G > 0$ such that the log-density of the policy function satisfies $\|\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|x)\|_2 \leq G$, for all $a \in \mathcal{A}$ and $x \in \mathcal{X}$.*

Assumption 4.2 ensures that the parameterized policy $\pi_{\boldsymbol{\theta}}$ changes smoothly with the updation of $\boldsymbol{\theta}$. It is a common and important condition in policy evaluation and optimization (Reddi et al., 2016; Allen-Zhu and Hazan, 2016; Pan et al., 2019; Xu et al., 2019, 2020).

The next assumption regulates that the density ratio is always bounded in our algorithm.

Assumption 4.3. *Let $\pi_{\boldsymbol{\theta}_t}$ and β be the target and logging policies respectively at iteration t. For all $t \in [T]$, there exists a constant $W > 0$ such that the density ratio satisfies $\pi_{\boldsymbol{\theta}_t}(a|x)/\beta(a|x) \leq W$ for all $a \in \mathcal{A}$ and $x \in \mathcal{X}$.*

The above assumption follows a line of work in distributionally robust learning, including (Liu et al., 2019, 2020). In practice, it can be satisfied as long as the logging policy is not too extreme, namely every action is chosen with the probability of at least $1/W$ in the logging dataset. Moreover, Assumption 4.3 also guarantees the coverage of the offline dataset over actions.

Now we propose our first main theorem: we show that the statistical error of policy gradient estimation under distributionally robust learning is bounded.

Theorem 4.4. *Assume Assumptions 4.2 and 4.3 hold. In Algorithm 1, let $\mathcal{S} = \{(x_i, a_i, r_i)\}_{i=1}^{|\mathcal{S}|}$ be a logging dataset with i.i.d. training samples from logging policy β, $\pi_{\boldsymbol{\theta}_t}$ be the learning policy at iteration t, $\mathcal{F}$ be a reward function class satisfying $\sup_{(x,a)\in\mathcal{S}, r, r'\in\mathcal{F}} |r(x,a) - r'(x,a)| \leq M$ for a constant $M > 0$, and $\hat{\mathfrak{R}}_S(\mathcal{F})$ be the empirical Rademacher complexity on $\mathcal{S}$. We assume that $\rho_r > 0$ is lower bounded by B, the weight estimation for the prediction $\hat{\beta}(a|x)/\hat{\pi}_{\boldsymbol{\theta}_t}(a|x)$ is lower bounded by R for any context-action pair (x, a), base distribution variance is σ_0^2, and λ is the upper bound of all λ_i among the dimensions of $\phi(x)$. With probability at least $1-\delta$, the generalization bound of policy gradient estimation satisfies:*

$$\mathbb{E}_{\pi_{\boldsymbol{\theta}_t}}[\|\hat{g}(\boldsymbol{\theta}_t) - \nabla V^{\pi_{\boldsymbol{\theta}_t}}\|_2^2]$$
$$\leq G^2 W\big[\big(2RB + \sigma_0^{-2}\big)^{-1} + \lambda$$
$$+ 4M\hat{\mathfrak{R}}_S(\mathcal{F}) + 3M^2\sqrt{1/(2|\mathcal{S}|)\log(2/\delta)}\big].$$

The proof of Theorem 4.4 can be found in Appendix A.1.

Remark 4.5. *The above estimation error of policy gradient can be interpreted as follows. On the right hand side, $(2RB + \sigma_0^2)^{-1} + \lambda$ stands for the variance and bias coming from the offline dataset. Suppose that we are training with an unbiased dataset containing enough samples,we will have an accurate reward estimation, and hence also an unbiased policy gradient estimator. $4M\hat{\mathfrak{R}}_S(\mathcal{F})$ reflects the richness of the assumed reward function class $\mathcal{F}$. The last term $3M^2\sqrt{\log(2/\delta)/2|\mathcal{S}|}$ is an $O(1/N^{1/2})$ decreases to zero if we have infinite training samples.*

Based on Theorem 4.4, we demonstrate the convergence of our policy optimization method.

Theorem 4.6. *Under the same conditions as Theorem 4.4, assume that the value function $V^{\pi_{\boldsymbol{\theta}}}$ is always L-smooth for every $\boldsymbol{\theta}$, and that we run at least T iterations for policy gradient learning with step size $\eta = 1/(4L)$. Then the*

output $\pi_{\boldsymbol{\theta}_{\hat{T}}}$ of Algorithm 1 satisfies:

$$\mathbb{E}[\|\nabla V^{\pi_{\boldsymbol{\theta}_{\hat{T}}}}\|_2^2] \leq 4/(T\eta)\big(V^{\pi_{\boldsymbol{\theta}_{T+1}}} - V^{\pi_{\boldsymbol{\theta}_1}}\big) + 3G^2W\big[(2RB + \sigma_0^{-2})^{-1} + \lambda + 4M\hat{\mathfrak{R}}_S(\mathcal{F}) + 3M^2\sqrt{1/(2|\mathcal{S}|)\log(2/\delta)}\big].$$

The proof of Theorem 4.6 can be found in Appendix A.2.

Remarks: Theorem 4.6 shows that the policy optimized in Algorithm 1 converges to a stationary point with a convergence rate of $O(1/T)$ and $O(1/N^{1/2})$, where T, N are the number of time steps and samples respectively. The second to the last term on the right-hand side comes from policy gradient estimation in each iteration. As mentioned in Remarks for Theorem 4.4, if the reward and policy gradient estimates are accurate and unbiased at every iteration, then the policy will converge with an $O(1/T)$ rate. Compared with (Si et al., 2020b), where the estimated policy asymptotically converges to the optimal policy at a canonical statistical $O(1/N^{1/2})$ rate, our method enjoys an additional $O(1/T)$ convergence guarantee over time steps.

4.4 Refinement with Limited Online Exploration

In many real-life applications of bandits learning problems, limited online interaction is allowed to improve performance. In this subsection, we show that with limited access to the online environment, the policy learned by Algorithm 1 can be further refined with a small number of online exploration samples. The proposed method, DROPO-exp, is displayed in Algorithm 2. The idea is that after one episode, we collect an exploration dataset $\mathcal{D}_{exp}$ with the current policy π_{θ_t} and add them to the logging dataset $\mathcal{D}_{\log} = \mathcal{D}_{exp} \cup \mathcal{D}_{\log}$. Then we call Algorithm 1 with the initial policy parameter θ_t. We set the exploration size $|D_{exp}|$ to be much smaller than the size of the logging dataset $|\mathcal{D}_0|$. Note that we do not update our policy using the collected reward during the online interaction and thus the exploration could be done in parallel for efficiency.

After we collect online exploration data and merge that with offline data, we estimate the reward in a similar fashion with the offline setting. Notat that when applying IPS-based policy gradients estimators in this setting, the "logging policies" are different in the reward estimation in offline data and the online-collected exploration data. Specifically, for $\mathcal{D}_{\log}$, the logging policy is the one that generates the offline dataset, while in $\mathcal{D}_{exp}$, the logging policy is the current policy in the policy gradient.

5 EXPERIMENTS

In this section, we conduct experiments to study the empirical performances of the proposed algorithms[1]. We test our algorithm and baseline methods on real-world datasets including *Optdigits* and *Adult* from the UCI machine learning repository (Dua and Graff, 2019), *MNIST* (LeCun et al., 1998), and *CIFAR-10* (Krizhevsky et al., 2009), with details in Table 1. We compare DROPO (Algorithm 1) and DROPO-exp (Algorithm 2) with baseline methods including DM, IPS, DR (Dudík et al., 2011), and SDRPL (Si et al., 2020b). Note that DM and DR also need to learn a reward mapping like in (4.1). Instead of learning reward mapping using the mean-squared loss function, we use the same robust regression in our methods but without considering the covariate shift, which significantly improves the performance of baseline algorithms. For the completeness of our work, we also compare the vanilla implementation of DM and DR using mean-squared loss with our improved versions in Appendix B.3.1.

Algorithm 2 Distributionally Robust Offline Policy Optimization with Limited Online Exploration (DROPO-exp)

Inputs: Initial logging data $\mathcal{D}_0$ with training samples $\{x_i, a_i, r_i\}_{i=1}^{|\mathcal{D}_0|}$, learning rate γ, ρ^{rr}, ρ^{rx}. Exploration data set $\mathcal{D}_{exp} = \mathcal{D}_0$
for $i = 0, \ldots, I$ **do**

Offline Learning

$\boldsymbol{\theta}_{i+1} \leftarrow$ Call DROPO with dataset $\mathcal{D}_i$ and initial policy parameter $\boldsymbol{\theta}_i$ ▷ Algorithm 1

Online Exploration

Collect exploration dataset $\mathcal{D}_{exp}$ from policy $\pi_{\boldsymbol{\theta}_{i+1}}$
Update logging dataset $\mathcal{D}_{i+1} = \mathcal{D}_i \cup \mathcal{D}_{exp}$
end for

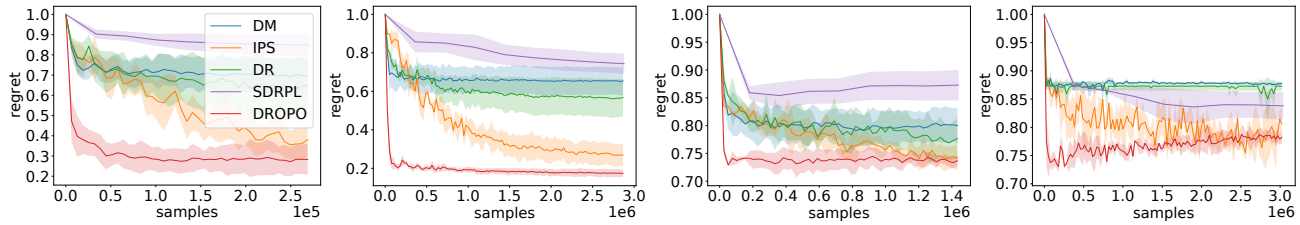
Table 1: Real-world datasets used in our experiments.

	DIMENSION	# OF ARMS	# OF SAMPLES
OPTDIGITS	64	10	5,620
MNIST	784	10	60,000
CIFAR-10	3072	10	60,000
ADULT	92	14	45,221

Logging policies We test our algorithms on various offline scenarios where the logging policy follows different distributions. In particular, inspired by previous work on dealing with label shifts (Lipton et al., 2018), we design the logging policy for collecting data as follows.

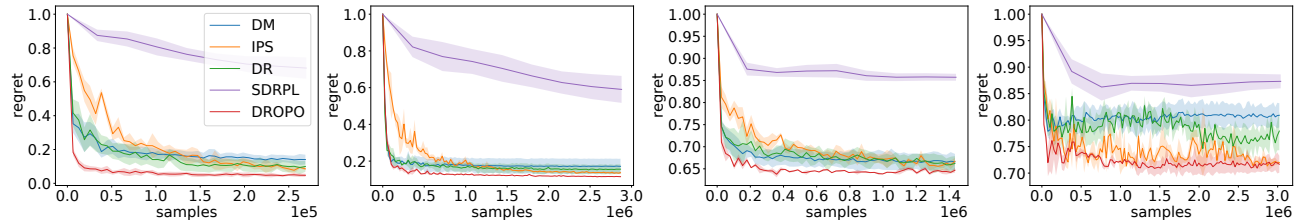
1. *Tweak-1(ρ)*: one class accounts for ρ probability, and the other classes evenly share the remaining $1 - \rho$. Larger ρ means a larger distributional shift of the logging dataset. We consider $\rho = 0.95, 0.99$ respectively.
2. *Dirichlet (α)*: the logging policy complies with Dirichlet distribution with parameter α. Smaller α means a larger shift of the logging dataset. We randomly gener-

[1]All the implementations are available at `https://github.com/guoyihonggyh/Distributionally-Robust-Policy-Gradient-for-Offline-Contextual-Bandits`



(a) *Optdigits* (b) *MNIST* (c) *CIFAR-10* (d) *Adult*

Figure 1: Performance comparison for *Tweak-1(*0.99*)* in the offline setting on four datasets.



(a) *Optdigits* (b) *MNIST* (c) *CIFAR-10* (d) *Adult*

Figure 2: Performance comparison for *Tweak-1(*0.95*)* in the offline setting on four datasets.

ate one logging policy and fix that for every experiment. $\alpha = 1.0, 0.1$ is considered in our paper. However, note that for *Dirichlet(*0.1*)*, the probability for some classes will be so small that no sample coming from those classes is sampled. Therefore, in *Dirichlet(*0.1*)*, we are actually using logging policy $= 0.95 \times$ *Dirichlet*$(0.1) + 0.05 \times$ *uniform*.

Offline dataset generation We first convert a k-class classification task into a k-armed contextual bandit problem. In the classification task, we have $(x, y) \sim D$, where $x \in \mathcal{X}$ is the context vector and $y \in \mathcal{Y}$ is the label. The goal is to find a classifier that minimizes the classification error. We then generate the offline dataset from D based on the logging policy β. Following Beygelzimer and Langford (2009), given a feature vector x, we choose a label (arm) a based on $\beta(a|x)$ and reveal the reward $r(x, a)$. The offline data will be in the form of $(x, a, r(x, a), \beta(a|x))$. During the training, we assume the logging policy is known. We use 60% samples for training and 40% for testing.

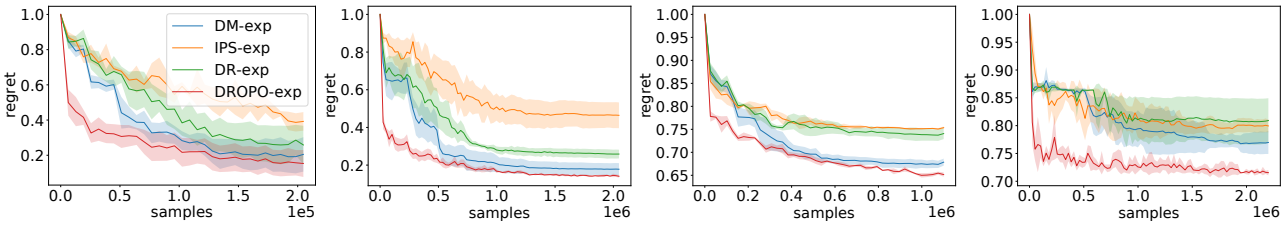
Simulating online exploration To implement Algorithm 2, for each dataset, we use 40% samples for the initial offline training, 20% as the online environment and 40% for testing. In each round, we collect an exploration dataset D_{exp} by sampling with replacement. Note that here the initial offline training size (40%) is smaller than that of the pure offline learning setting (60%).

Evaluation We use the regret on the test dataset as our evaluation metric. For the comparison between offline and online experiments, our experimental result is demonstrated based on the relationship between "samples" and "regret". Here, "samples" means the number of data used for policy optimization, which equals mini-batch$\times$batch-size in both the offline setting and online setting. We repeat the experiments 5 times and plot the average regret with standard deviation. For each experiment, we use the same set of random seeds for data generation.

DROPO outperforms baseline methods under large distributional shifts. Average regret and variance for DROPO and baselines are plotted in Fig. 1 and Fig. 2. We present the results for *Tweak-1(*0.95*)* and *Tweak-1(*0.99*)* on *Optdigits*, *MNIST*, *CIFAR-10* and *Adult*. Results with Dirichlet logging policies can be found in Appendix B.2. In all datasets and scenarios, we can clearly see the advantage of DROPO over baselines, especially in large shift cases (*Tweak-1(*0.99*)*). DROPO has smaller regret on average, showing its effectiveness. It also has a smaller variance during training, which means our method is more robust under various logging policies.

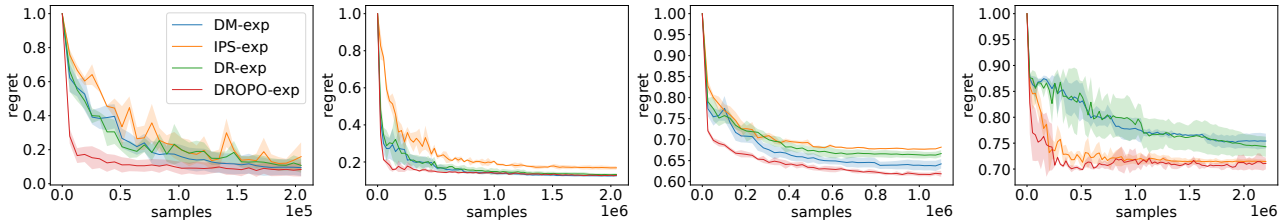
In Fig. 3 and Fig. 4, we plot the results of policy optimization with limited online exploration. We can similarly conclude that DROPO-exp outperforms the baselines in the limited online exploration setting. Due to the page limit, we defer the experimental results for Dirichlet logging policies to Appendix B.2. Since SDRPL (Si et al., 2020b) does not have the online exploration version, we only show the results in pure offline setting. Its performance is significantly worse than other methods. This is due to that SDRPL is overly conservative and unstable to train in practice, which has no convergence guarantees.

Limited online exploration further improves DROPO. We also compare the results between pure offline and limited online exploration settings for *Optdigits* with $|D_{exp}| = 100$ and *MNIST* with $|D_{exp}| = 500$ in Fig. 5. DROPO-exp utilizes a smaller amount of data and achieves better or at



(a) *Optdigits*, $|D_{exp}| = 100$ (b) *MNIST*, $|D_{exp}| = 500$ (c) *CIFAR-10*, $|D_{exp}| = 500$ (d) *Adult*, $|D_{exp}| = 500$

Figure 3: Performance comparison for *Tweak-1*(0.99) in the online setting on four datasets. DROPO-exp significantly outperforms baseline methods.



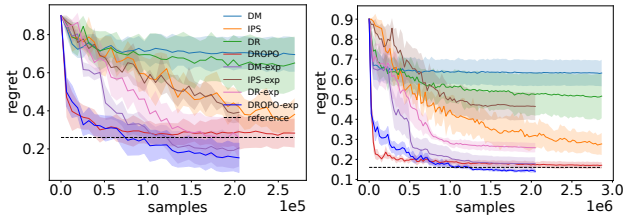
(a) *Optdigits*, $|D_{exp}| = 100$ (b) *MNIST*, $|D_{exp}| = 500$ (c) *CIFAR-10*, $|D_{exp}| = 500$ (d) *Adult*, $|D_{exp}| = 500$

Figure 4: Performance comparison for *Tweak-1*(0.95) in the online setting on four datasets. DROPO-exp outperforms baseline methods on *Optdigits* and *CIFAR-10* in the whole learning process. It also outperforms baseline methods on *MNIST* and *Adult* in the early stage of the learning.

least comparable performance than the pure offline version. Note that DROPO-exp starts with a significantly smaller offline dataset and has fewer total training samples, but eventually achieves a similar or even better regret than DROPO. The total number of exploration data is about 14% of all data when $|D_{exp}|$ = 100 in *Optdigits* and 7% of all data when $|D_{exp}|$ = 500 in *MNIST*. We also apply our online exploration strategy to other baseline models, which again shows a significant improvement compared to the pure offline version. This is because the newly sampled data can greatly improve the quality of the combined dataset used for reward estimation. Besides, we see that online exploration improves the performance more on baseline methods than ours. This is because DROPO is more robust to the data insufficiency in the offline training and the online data is less helpful compared to the baseline methods. We also investigate two other online exploration strategies. We defer the results to Appendix B.3.

6 CONCLUSION AND FUTURE WORK

In this paper, we propose the distributionally robust offline policy optimization (DROPO) algorithm for contextual bandits. We show that our method converges to a stationary point up to the statistical error of the offline dataset and the model approximation error of the reward function with convergence rate $O(1/T)$, where T is the number of steps. We compare our algorithm with baseline methods such as DM, IPS, DR, and SDRPL (Si et al., 2020b) in real-



(a) *Optdigits*, $|D_{exp}| = 100$ (b) *MNIST*, $|D_{exp}| = 500$

Figure 5: Comparison between limited online exploration and pure offline settings on *Optdigits* and *MNIST* under *Tweak-1*(0.99) shift. The horizontal dashed lines refer to the best performance that DROPO can achieve in the pure offline setting. Starting with a significantly smaller dataset (40% of all data) than offline settings (60% of all data), DROPO-exp achieves better results with fewer training samples. See Appendix B.3.2 for more experiments on different exploration sizes.

world dataset experiments. The results show that DROPO gives a more robust policy gradient estimation and thus a lower regret on the test dataset than all baselines. Extensive evaluations show that when the distributional shift is large, the advantage of our DROPO algorithm is more pronounced. We also propose DROPO-exp that can improve the performance when even a small amount of online exploration is allowed. Future work includes the investigation of more effective exploration strategies to utilize the newly collected exploration data. We are also interested in studying the exploration under exploration budgets when limited online exploration is allowed.

Acknowledgements

The authors would like to thank the anonymous reviewers for their invaluable comments. Pan Xu is supported by the Whitehead Scholar award and the startup funding at Duke University. Anqi Liu is supported by the JHU-Amazon AI2AI faculty award. Anima Anandkumar is partially supported by Bren Named Chair Professorship at Caltech.

References

Agarwal, A., Hsu, D., Kale, S., Langford, J., Li, L., and Schapire, R. (2014). Taming the monster: A fast and simple algorithm for contextual bandits. In *International Conference on Machine Learning*, pages 1638–1646. PMLR.

Agrawal, S. and Goyal, N. (2013). Thompson sampling for contextual bandits with linear payoffs. In *International conference on machine learning*, pages 127–135. PMLR.

Allen-Zhu, Z. and Hazan, E. (2016). Variance reduction for faster non-convex optimization. In *International conference on machine learning*, pages 699–707. PMLR.

Bang, H. and Robins, J. M. (2005). Doubly robust estimation in missing data and causal inference models. *Biometrics*, 61(4):962–973.

Beygelzimer, A. and Langford, J. (2009). The offset tree for learning with partial labels. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 129–138.

Biggs, M., Gao, R., and Sun, W. (2021). Loss functions for discrete contextual pricing with observational data. *arXiv preprint arXiv:2111.09933*.

Bottou, L., Peters, J., Quiñonero-Candela, J., Charles, D. X., Chickering, D. M., Portugaly, E., Ray, D., Simard, P., and Snelson, E. (2013). Counterfactual reasoning and learning systems: The example of computational advertising. *Journal of Machine Learning Research*, 14(11).

Cesa-Bianchi, N., Gentile, C., and Zappella, G. (2013). A gang of bandits. *Advances in neural information processing systems*, 26.

Chen, M., Beutel, A., Covington, P., Jain, S., Belletti, F., and Chi, E. H. (2019). Top-k off-policy correction for a reinforce recommender system. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 456–464.

Chen, R. and Paschalidis, I. C. (2018). A robust learning approach for regression models based on distributionally robust optimization. *Journal of Machine Learning Research*, 19(13).

Chen, X., Monfort, M., Liu, A., and Ziebart, B. D. (2016). Robust covariate shift regression. In *Artificial Intelligence and Statistics*, pages 1270–1279. PMLR.

Chernozhukov, V., Demirer, M., Lewis, G., and Syrgkanis, V. (2019). Semi-parametric efficient policy learning with continuous actions. *Advances in Neural Information Processing Systems*, 32.

Dixit, A., Ahmadi, M., and Burdick, J. W. (2022). Distributionally robust model predictive control with total variation distance. *arXiv preprint arXiv:2203.12062*.

Dua, D. and Graff, C. (2019). UCI machine learning repository.

Dudík, M., Langford, J., and Li, L. (2011). Doubly robust policy evaluation and learning. *arXiv preprint arXiv:1103.4601*.

Farajtabar, M., Chow, Y., and Ghavamzadeh, M. (2018). More robust doubly robust off-policy evaluation. In *International Conference on Machine Learning*, pages 1447–1456. PMLR.

Fathony, R., Asif, K., Liu, A., Bashiri, M. A., Xing, W., Behpour, S., Zhang, X., and Ziebart, B. D. (2018). Consistent robust adversarial prediction for general multiclass classification. *arXiv preprint arXiv:1812.07526*.

Fu, J., Norouzi, M., Nachum, O., Tucker, G., Wang, Z., Novikov, A., Yang, M., Zhang, M. R., Chen, Y., Kumar, A., et al. (2021). Benchmarks for deep off-policy evaluation. *arXiv preprint arXiv:2103.16596*.

Gao, R., Chen, X., and Kleywegt, A. J. (2017). Distributional robustness and regularization in statistical learning. *arXiv preprint arXiv:1712.06050*.

Gao, R. and Kleywegt, A. (2022). Distributionally robust stochastic optimization with wasserstein distance. *Mathematics of Operations Research*.

Horvitz, D. G. and Thompson, D. J. (1952). A generalization of sampling without replacement from a finite universe. *Journal of the American statistical Association*, 47(260):663–685.

Joachims, T., Swaminathan, A., and De Rijke, M. (2018). Deep learning with logged bandit feedback. In *International Conference on Learning Representations*.

Kakade, S. M. (2001). A natural policy gradient. *Advances in neural information processing systems*, 14.

Kallus, N., Mao, X., Wang, K., and Zhou, Z. (2022). Doubly robust distributionally robust off-policy evaluation and learning. *arXiv preprint arXiv:2202.09667*.

Kallus, N. and Uehara, M. (2020). Statistically efficient off-policy policy gradients. In *International Conference on Machine Learning*, pages 5089–5100. PMLR.

Kallus, N. and Zhou, A. (2018). Confounding-robust policy improvement. *Advances in neural information processing systems*, 31.

Kitagawa, T. and Tetenov, A. (2018). Who should be treated? empirical welfare maximization methods for treatment choice. *Econometrica*, 86(2):591–616.

Krause, A. and Ong, C. S. (2011). Contextual gaussian process bandit optimization. In *Neural Information Processing Systems (NeurIPS)*.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

Kuhn, D., Esfahani, P. M., Nguyen, V. A., and Shafieezadeh-Abadeh, S. (2019). Wasserstein distributionally robust optimization: Theory and applications in machine learning. In *Operations research & management science in the age of analytics*, pages 130–166. Informs.

Lale, S., Azizzadenesheli, K., Anandkumar, A., and Hassibi, B. (2019). Stochastic linear bandits with hidden low rank structure. *arXiv preprint arXiv:1901.09490*.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lei, H., Tewari, A., and Murphy, S. (2014). An actor-critic contextual bandit algorithm for personalized interventions using mobile devices. *Neural Information Processing Systems (NeurIPS)*.

Levine, S., Kumar, A., Tucker, G., and Fu, J. (2020). Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*.

Li, G., Ma, C., and Srebro, N. (2022). Pessimism for offline linear contextual bandits using ℓ_p confidence sets. *arXiv preprint arXiv:2205.10671*.

Li, L., Chu, W., Langford, J., and Schapire, R. E. (2010). A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670.

Li, L., Chu, W., Langford, J., and Wang, X. (2011). Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 297–306.

Lipton, Z., Wang, Y.-X., and Smola, A. (2018). Detecting and correcting for label shift with black box predictors. In *International conference on machine learning*, pages 3122–3130. PMLR.

Liu, A., Liu, H., Anandkumar, A., and Yue, Y. (2019). Triply robust off-policy evaluation. *arXiv preprint arXiv:1911.05811*.

Liu, A., Shi, G., Chung, S.-J., Anandkumar, A., and Yue, Y. (2020). Robust regression for safe exploration in control. In *Learning for Dynamics and Control*, pages 608–619. PMLR.

Liu, A. and Ziebart, B. (2014). Robust classification under sample selection bias. *Advances in neural information processing systems*, 27.

Lu, Y., Xu, Z., and Tewari, A. (2021). Bandit algorithms for precision medicine. *arXiv preprint arXiv:2108.04782*.

Mahmood, A. R., Van Hasselt, H. P., and Sutton, R. S. (2014). Weighted importance sampling for off-policy learning with linear function approximation. *Advances in Neural Information Processing Systems*, 27.

Mohajerin Esfahani, P. and Kuhn, D. (2018). Data-driven distributionally robust optimization using the wasserstein metric: Performance guarantees and tractable reformulations. *Mathematical Programming*, 171(1):115–166.

Nguyen-Tang, T., Gupta, S., Nguyen, A. T., and Venkatesh, S. (2022). Offline neural contextual bandits: Pessimism, optimization and generalization. In *International Conference on Learning Representations*.

Pan, F., Cai, Q., Tang, P., Zhuang, F., and He, Q. (2019). Policy gradients for contextual recommendations. In *The World Wide Web Conference*, pages 1421–1431.

Precup, D. (2000). Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, page 80.

Radlinski, F., Kleinberg, R., and Joachims, T. (2008). Learning diverse rankings with multi-armed bandits. In *Proceedings of the 25th international conference on Machine learning*, pages 784–791.

Reddi, S. J., Hefny, A., Sra, S., Póczos, B., and Smola, A. (2016). Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, pages 314–323. PMLR.

Rezaei, A., Fathony, R., Memarrast, O., and Ziebart, B. (2020). Fairness for robust log loss classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5511–5518.

Rezaei, A., Liu, A., Memarrast, O., and Ziebart, B. D. (2021). Robust fairness under covariate shift. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9419–9427.

Robins, J. M. and Rotnitzky, A. (1995). Semiparametric efficiency in multivariate regression models with missing data. *Journal of the American Statistical Association*, 90(429):122–129.

Sakhi, O., Faury, L., and Vasile, F. (2020). Improving offline contextual bandits with distributional robustness. *arXiv preprint arXiv:2011.06835*.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015). Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR.

Shafieezadeh-Abadeh, S., Kuhn, D., and Esfahani, P. M. (2019). Regularization via mass transportation. *Journal of Machine Learning Research*, 20(103):1–68.

Si, N., Zhang, F., Zhou, Z., and Blanchet, J. (2020a). Distributional robust batch contextual bandits. *arXiv preprint arXiv:2006.05630*.

Si, N., Zhang, F., Zhou, Z., and Blanchet, J. (2020b). Distributionally robust policy evaluation and learning in offline contextual bandits. In *International Conference on Machine Learning*, pages 8884–8894. PMLR.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. (2014). Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. PMLR.

Smirnova, E., Dohmatob, E., and Mary, J. (2019). Distributionally robust reinforcement learning. *arXiv preprint arXiv:1902.08708*.

Su, Y., Dimakopoulou, M., Krishnamurthy, A., and Dudík, M. (2019). Doubly robust off-policy evaluation with shrinkage. *arXiv preprint arXiv:1907.09623*.

Su, Y., Dimakopoulou, M., Krishnamurthy, A., and Dudík, M. (2020). Doubly robust off-policy evaluation with shrinkage. In *International Conference on Machine Learning*, pages 9167–9176. PMLR.

Sugiyama, M., Krauledat, M., and Müller, K.-R. (2007). Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research*, 8(5).

Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. (1999). Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12.

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

Swaminathan, A. and Joachims, T. (2015b). The self-normalized estimator for counterfactual learning. *advances in neural information processing systems*, 28.

Swaminathan, A., Krishnamurthy, A., Agarwal, A., Dudik, M., Langford, J., Jose, D., and Zitouni, I. (2017). Off-policy evaluation for slate recommendation. *Advances in Neural Information Processing Systems*, 30.

Tang, L., Rosales, R., Singh, A., and Agarwal, D. (2013). Automatic ad format selection via contextual bandits. In *ACM International Conference on Information and Knowledge Management (CIKM)*.

Tennenholtz, G., Shalit, U., and Mannor, S. (2020). Off-policy evaluation in partially observable environments. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10276–10283.

Thomas, P. and Brunskill, E. (2016). Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 2139–2148. PMLR.

Thomas, P., Theocharous, G., and Ghavamzadeh, M. (2015). High-confidence off-policy evaluation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

Voloshin, C., Le, H. M., Jiang, N., and Yue, Y. (2019). Empirical study of off-policy policy evaluation for reinforcement learning. *arXiv preprint arXiv:1911.06854*.

Wang, H., Xing, W., Asif, K., and Ziebart, B. (2015). Adversarial prediction games for multivariate losses. *Advances in Neural Information Processing Systems*, 28.

Wang, Y.-X., Agarwal, A., and Dudık, M. (2017). Optimal and adaptive off-policy evaluation in contextual bandits. In *International Conference on Machine Learning*, pages 3589–3597. PMLR.

Xu, P., Gao, F., and Gu, Q. (2019). An improved convergence analysis of stochastic variance-reduced policy gradient. In *Uncertainty in Artificial Intelligence*, pages 541–551. PMLR.

Xu, P., Gao, F., and Gu, Q. (2020). Sample efficient policy gradient methods with recursive variance reduction. In *International Conference on Learning Representations*.

Xu, P., Zheng, H., Mazumdar, E. V., Azizzadenesheli, K., and Anandkumar, A. (2022). Langevin monte carlo for contextual bandits. In *International Conference on Machine Learning*, pages 24830–24850. PMLR.

Yue, Y., Hong, S. A., and Guestrin, C. (2012). Hierarchical exploration for accelerating contextual bandits. In *International Conference on Machine Learning (ICML)*.

Zadrozny, B., Langford, J., and Abe, N. (2003). Cost-sensitive learning by cost-proportionate example weighting. In *Third IEEE international conference on data mining*, pages 435–442. IEEE.

Zhang, B., Tsiatis, A. A., Davidian, M., Zhang, M., and Laber, E. (2012). Estimating optimal treatment regimes from a classification perspective. *Stat*, 1(1):103–114.

Zhao, Y.-Q., Zeng, D., Laber, E. B., Song, R., Yuan, M., and Kosorok, M. R. (2015). Doubly robust learning for estimating individualized treatment with censored data. *Biometrika*, 102(1):151–168.

A THEORETICAL ANALYSIS OF THE PROPOSED ALGORITHM

In this section, we prove our theoretical results.

A.1 Distributionally Robust Learning on the Reward Function

We first provide the proof of Theorem 4.4. We begin with recalling the policy gradient estimator defined as follows:

$$\hat{g}(\boldsymbol{\theta}) = \frac{1}{|\mathcal{S}_{\pi_{\boldsymbol{\theta}}}|} \sum_{i \in \mathcal{S}_{\pi_{\boldsymbol{\theta}}}} \nabla \log \pi_{\boldsymbol{\theta}}(a_i|x_i) \hat{r}(x_i, a_i), \tag{A.1}$$

where $\hat{r}$ is the reward function learned from the offline dataset with the distributionally robust learning method. (A.1) suggests that the policy gradient estimator $\hat{g}$ follows directly from the estimated reward function $\hat{r}$. Therefore, the statistical error of policy gradient estimation can be reduced to the statistical error of reward estimation. To this end, we present the following key lemma.

Lemma A.1. *(Theorem 1 in Liu et al. (2020)) Assume Assumptions 4.2 and 4.3 hold. In Algorithm 1, let* $\mathcal{S} = \{(x_i, a_i, r_i)\}_{i=1}^{|\mathcal{S}|}$ *be a logging dataset with i.i.d. training samples from logging policy* β*,* $\pi_{\boldsymbol{\theta}_t}$ *be the learning policy at iteration* t*,* $\mathcal{F}$ *be a reward function class satisfying* $\sup_{(x,a)\in\mathcal{S}, r, r' \in \mathcal{F}} |r(x,a) - r'(x,a)| \leq M$ *for a constant* $M > 0$*, and* $\hat{\Re}_S(\mathcal{F})$ *be the empirical Rademacher complexity on* $\mathcal{S}$*. We assume that* $\rho_r > 0$ *is lower bounded by* B*, the weight estimation for the prediction* $\hat{\beta}(a|x)/\hat{\pi}_{\boldsymbol{\theta}_t}(a|x)$ *is lower bounded by R for any context-action pair* (x, a)*, base distribution variance is* σ_0^2*, and* λ *is the upper bound of all* λ_i *among the dimensions of* $\phi(x)$*. With probability at least* $1 - \delta$*, the generalization bound of reward estimation satisfies:*

$$\mathbb{E}_{\pi_{\boldsymbol{\theta}_t}}\left[\left(r(x,a) - \hat{r}(x,a)\right)^2 \right] \leq W \left[\left(2RB + \sigma_0^{-2}\right)^{-1} + \lambda + 4M\hat{\Re}_S(\mathcal{F}) + 3M^2 \sqrt{\frac{\log \frac{2}{\delta}}{2|\mathcal{S}|}} \right].$$

Lemma A.1 shows that the statistical error of reward estimation is bounded by the composition of the bias and the variance brought by the offline dataset and the model approximation error. Note that this bound could be small if the offline dataset is unbiased and the number of training samples is large enough.

Now we are ready to lay down the proof of Theorem 4.4. We restate Theorem 4.4 as follows.

Theorem A.2. *Under the same conditions as in Lemma A.1, the generalization bound of policy gradient estimation satisfies:*

$$\mathbb{E}_{\pi_{\boldsymbol{\theta}_t}}[\|\hat{g}(\boldsymbol{\theta}_t) - \nabla V^{\pi_{\boldsymbol{\theta}_t}}\|_2^2] \leq G^2 W \left[\left(2RB + \sigma_0^{-2}\right)^{-1} + \lambda + 4M\hat{\Re}_S(\mathcal{F}) + 3M^2 \sqrt{\frac{\log \frac{2}{\delta}}{2|\mathcal{S}|}} \right].$$

Proof. By the definition of $\hat{g}(\boldsymbol{\theta})$ and $\nabla V^{\pi_{\boldsymbol{\theta}_t}}$ in (A.1) and (3.1), we have

$$\begin{aligned}
\mathbb{E}_{\pi_{\boldsymbol{\theta}_t}}[\|\hat{g}(\boldsymbol{\theta}_t) - \nabla V^{\pi_{\boldsymbol{\theta}_t}}\|_2^2] &= \mathbb{E}_{\pi_{\boldsymbol{\theta}_t}}[\|\nabla \log \pi_{\boldsymbol{\theta}_t} \hat{r}(x,a) - \nabla \log \pi_{\boldsymbol{\theta}_t} r(x,a)\|_2^2] \\
&= \mathbb{E}_{\pi_{\boldsymbol{\theta}_t}}[\|\nabla \log \pi_{\boldsymbol{\theta}_t}\|_2^2 \left(\hat{r}\left(x,a\right) - r\left(x,a\right)\right)^2] \\
&\leq G^2 W \left[\left(2RB + \sigma_0^{-2}\right)^{-1} + \lambda + 4M\hat{\Re}_S(\mathcal{F}) + 3M^2 \sqrt{\frac{\log \frac{2}{\delta}}{2|\mathcal{S}|}} \right],
\end{aligned}$$

where the inequality holds due to Assumption 4.2 and Lemma A.1. □

A.2 Convergence of Policy Optimization

We devote this section to the proof of Theorem 4.6. In specific, we demonstrate that the policy parameter $\boldsymbol{\theta}$ learned by our algorithm converges to a stationary point such that $\|\nabla V^{\pi_{\boldsymbol{\theta}}}\|_2 \leq \epsilon + C$ for any $\epsilon > 0$, where C is a constant determined by the offline dataset and model approximation error. We first restate Theorem 4.6 as follows.

Theorem A.3. *Under the same conditions as Theorem 4.4, assume that the value function $V^{\pi_{\boldsymbol{\theta}}}$ is always L-smooth for every $\boldsymbol{\theta}$, and that we run at least T iterations for policy gradient learning with step size $\eta = 1/(4L)$. Then the output $\pi_{\boldsymbol{\theta}_{\hat{T}}}$ of Algorithm 1 satisfies:*

$$\mathbb{E}[\|\nabla V^{\pi_{\boldsymbol{\theta}_{\hat{T}}}}\|_2^2] \leq \frac{V^{\pi_{\boldsymbol{\theta}_{T+1}}} - V^{\pi_{\boldsymbol{\theta}_1}}}{T\eta/4} + 3G^2W\left[\left(2RB + \sigma_0^{-2}\right)^{-1} + \lambda + 4M\hat{\mathfrak{R}}_S(\mathcal{F}) + 3M^2\sqrt{\frac{\log\frac{2}{\delta}}{2|\mathcal{S}|}}\right].$$

Proof. In the rest of this proof, we denote $V(\boldsymbol{\theta}) \triangleq V^{\pi_{\boldsymbol{\theta}}}$ to simplify the notation.

Based on the assumption that $V(\boldsymbol{\theta})$ is L-smooth for every $\boldsymbol{\theta}$ and the update rule $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta\hat{g}(\boldsymbol{\theta}_t)$, we have

$$\begin{aligned}
V(\boldsymbol{\theta}_{t+1}) &\geq V(\boldsymbol{\theta}_t) + \langle\nabla V(\boldsymbol{\theta}_t), \boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t\rangle - \frac{L}{2}\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t\|_2^2 \\
&= V(\boldsymbol{\theta}_t) + \eta\langle\nabla V(\boldsymbol{\theta}_t), \hat{g}(\boldsymbol{\theta}_t)\rangle - \frac{L}{2}\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t\|_2^2 \\
&= V(\boldsymbol{\theta}_t) + \eta\|\nabla V(\boldsymbol{\theta}_t)\|_2^2 + \eta\langle\nabla V(\boldsymbol{\theta}_t), \hat{g}(\boldsymbol{\theta}_t) - \nabla V(\boldsymbol{\theta}_t)\rangle - \frac{L}{2}\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t\|_2^2. \qquad \text{(A.2)}
\end{aligned}$$

Due to the fact that $\|\nabla V(\boldsymbol{\theta}_t)\|_2^2 + \|\nabla g(\boldsymbol{\theta}_t) - \nabla V(\boldsymbol{\theta}_t)\|_2^2 \geq 2|\langle\nabla V(\boldsymbol{\theta}_t), \hat{g}(\boldsymbol{\theta}_t) - \nabla V(\boldsymbol{\theta}_t)\rangle|$, we have

$$\begin{aligned}
\text{(A.2)} &\geq V(\boldsymbol{\theta}_t) + \eta\|\nabla V(\boldsymbol{\theta}_t)\|_2^2 - \frac{\eta}{2}\|\nabla V(\boldsymbol{\theta}_t)\|_2^2 - \frac{\eta}{2}\|\hat{g}(\boldsymbol{\theta}_t) - \nabla V(\boldsymbol{\theta}_t)\|_2^2 - \frac{L}{2}\eta^2\|\hat{g}(\boldsymbol{\theta}_t)\|_2^2 \\
&\geq V(\boldsymbol{\theta}_t) + \eta\|\nabla V(\boldsymbol{\theta}_t)\|_2^2 - \frac{\eta}{2}\|\nabla V(\boldsymbol{\theta}_t)\|_2^2 - \frac{\eta}{2}\|\hat{g}(\boldsymbol{\theta}_t) - \nabla V(\boldsymbol{\theta}_t)\|_2^2 \\
&\quad - L\eta^2\|\nabla V(\boldsymbol{\theta}_t)\|_2^2 - L\eta^2\|\hat{g}(\boldsymbol{\theta}_t) - \nabla V(\boldsymbol{\theta}_t)\|_2^2 \qquad \text{(A.3)} \\
&= V(\boldsymbol{\theta}_t) + (\eta/2 - L\eta^2)\|\nabla V(\boldsymbol{\theta}_t)\|_2^2 - (\eta/2 + L\eta^2)\|\hat{g}(\boldsymbol{\theta}_t) - \nabla V(\boldsymbol{\theta}_t)\|_2^2,
\end{aligned}$$

where (A.3) holds due to mean inequality $\|\nabla V(\boldsymbol{\theta}_t)\|_2^2 + \|\hat{g}(\boldsymbol{\theta}_t) - \nabla V(\boldsymbol{\theta}_t)\|_2^2 \geq \frac{1}{2}\|\hat{g}(\boldsymbol{\theta}_t)\|^2$.

Rearranging the above inequality yields

$$(\eta/2 - L\eta^2)\|\nabla V(\boldsymbol{\theta}_t)\|_2^2 \leq V(\boldsymbol{\theta}_{t+1}) - V(\boldsymbol{\theta}_t) + (\eta/2 + L\eta^2)\|\hat{g}(\boldsymbol{\theta}_t) - \nabla V(\boldsymbol{\theta}_t)\|_2^2.$$

Now we take the expectation over $\pi_{\boldsymbol{\theta}_t}$ on both sides of the above inequality and then sum it up over $t = 1, 2, \cdots, T$. We have

$$\sum_{t=1}^{T}\mathbb{E}_{\pi_{\boldsymbol{\theta}_t}}\left[\|\nabla V(\boldsymbol{\theta}_t)\|_2^2\right] \leq \frac{V(\boldsymbol{\theta}_{T+1}) - V(\boldsymbol{\theta}_1)}{\eta/2 - L\eta^2} + \frac{\eta/2 + L\eta^2}{\eta/2 - L\eta^2}\sum_{t=1}^{T}\mathbb{E}_{\pi_{\boldsymbol{\theta}_t}}\left[\|\hat{g}(\boldsymbol{\theta}_t) - \nabla V(\boldsymbol{\theta}_t)\|_2^2\right].$$

Setting step size as $\eta = 1/(4L)$ in the above inequality yields

$$\sum_{t=1}^{T}\mathbb{E}_{\pi_{\boldsymbol{\theta}_t}}\left[\|\nabla V(\boldsymbol{\theta}_t)\|_2^2\right] \leq \frac{V(\boldsymbol{\theta}_{T+1}) - V(\boldsymbol{\theta}_1)}{\eta/4} + 3\sum_{t=1}^{T}\mathbb{E}_{\pi_{\boldsymbol{\theta}_t}}\left[\|\hat{g}(\boldsymbol{\theta}_t) - \nabla V(\boldsymbol{\theta}_t)\|_2^2\right]. \qquad \text{(A.4)}$$

We define $\boldsymbol{\theta}_{\hat{T}}$ such that the index $\hat{T}$ is randomly picked from $\{1, \ldots, T\}$. By Jensen's inequality, we have

$$\mathbb{E}[\|\nabla V(\boldsymbol{\theta}_{\hat{T}})\|_2^2] \leq \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_{\pi_{\boldsymbol{\theta}_t}}\left[\|\nabla V(\boldsymbol{\theta}_t)\|_2^2\right]. \qquad \text{(A.5)}$$

Combining (A.5) with (A.4) and subsequently with Theorem A.3, we immediately have

$$\begin{aligned}
\text{(A.5)} &\leq \frac{V(\boldsymbol{\theta}_{T+1}) - V(\boldsymbol{\theta}_1)}{T\eta/4} + \frac{3}{T}\sum_{t=1}^{T}\mathbb{E}_{\pi_{\boldsymbol{\theta}_t}}\left[\|\hat{g}(\boldsymbol{\theta}_t) - \nabla V(\boldsymbol{\theta}_t)\|_2^2\right] \\
&\leq \frac{V^{\pi_{\boldsymbol{\theta}_{T+1}}} - V^{\pi_{\boldsymbol{\theta}_1}}}{T\eta/4} + 3G^2W\left[\left(2RB + \sigma_0^{-2}\right)^{-1} + \lambda + 4M\hat{\mathfrak{R}}_S(\mathcal{F}) + 3M^2\sqrt{\frac{\log\frac{2}{\delta}}{2|\mathcal{S}|}}\right].
\end{aligned}$$

This completes the proof. □

B ADDITIONAL EXPERIMENTAL DETAILS AND RESULTS

In this section, we present some additional experimental results to further justify the effectiveness of our proposed method and conduct several ablation studies.

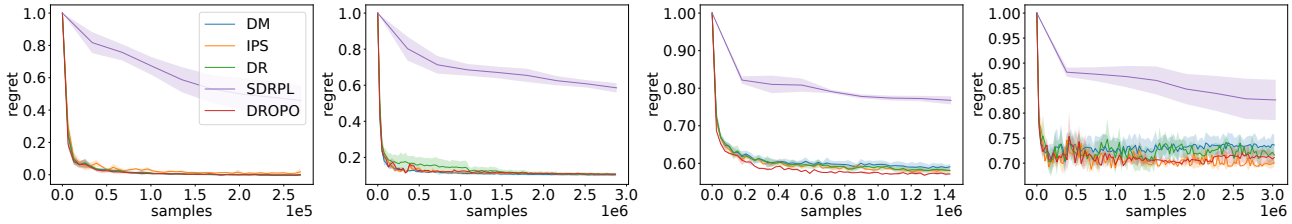
B.1 Hyper-parameter Tuning

We tune the hyper-parameter with grid search on the learning rate of policy gradient, reward estimation's neural network, and learning rate decay of policy gradient. Specifically, the learning rate of the policy gradient is searched in $[0.005, 0.001, 0.0005, 0.0001]$, the learning rate of reward estimation network is searched in $[0.001, 0.005, 0.001, 0.0005]$ and the learning rate decay of policy gradient is searched in $[1.0, 0.9, 0.7, 0.5]$. For optimization of ρ_r and $\boldsymbol{\rho}_x$ in (4.3) and (4.4), we use Stochastic Gradient Descent (SGD), where the learning rate is set as $\frac{10}{10+\sqrt{i-1}}$ at the i-th epoch.

B.2 More Experiments on Dirichlet Distributional Shift

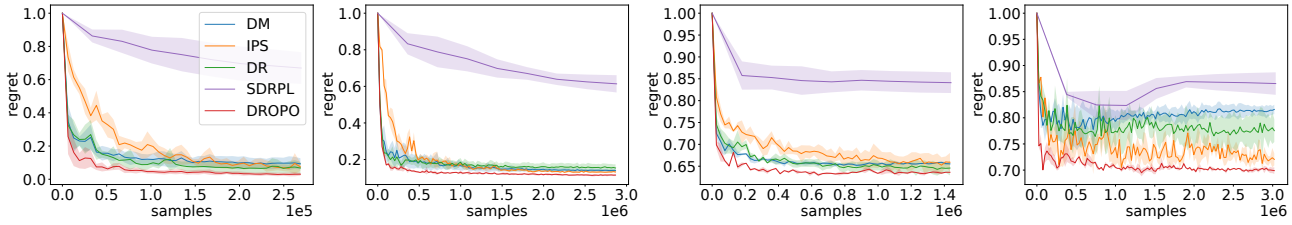
Fig. 6 and Fig.7 show DROPO's results for *Dirichlet*(1.0) and *Dirichlet*(0.1) on the four datasets in the pure offline setting. It can be seen that when the logging policy is *Dirichlet*(0.1), which means the distributional shift is relatively large, DROPO outperforms all the baseline methods. While for *Dirichlet*(1.0) where the logging distribution is much closer to a uniform distribution, all the methods have similar performances. This is consistent with the intuition that a more biased training dataset induces a worse testing performance.

Fig. 6 and Fig. 7 show DROPO-exp's results for *Dirichlet*(1.0) and *Dirichlet*(0.1) on the four datasets when limited online exploration is permitted. When the distributional shift is large, i.e., *Dirichlet*(1.0), DROPO-exp clearly outperforms baseline methods. When the shift is small, we again observe that DROPO-exp does not have an advantage over the baseline methods due to the good coverage of the logging data.



(a) *Optdigits* (b) *MNIST* (c) *CIFAR-10* (d) *Adult*

Figure 6: Performance comparison for *Dirichlet*(1.0) in the offline setting on four datasets. DROPO has similar level of performance on the four datasets because the logging is close to the uniform policy.



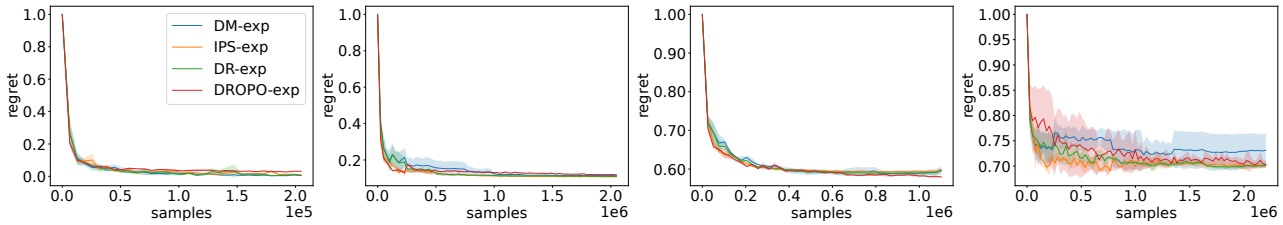
(a) *Optdigits* (b) *MNIST* (c) *CIFAR-10* (d) *Adult*

Figure 7: Performance comparison for *Dirichlet*(0.1) in the offline setting on four datasets. DROPO outperforms the baseline methods.

B.3 Ablation Study

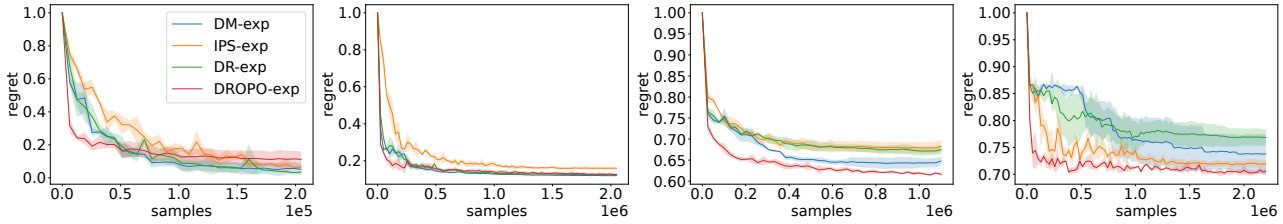
B.3.1 Our Direct Method vs. Vanilla Direct Method

In this section, we explain our choice of DM methods in our experiment. In particular, the DM algorithm implemented in our experiment is a variant of DROPO by setting the weight ratio $\frac{\beta(a|x)}{\pi(a|x)}$ to 1 in Algorithm 1 and Algorithm 2. In this way,



(a) *Optdigits*, $|D_{exp}| = 100$ (b) *MNIST* , $|D_{exp}| = 500$ (c) *CIFAR-10*, $|D_{exp}| = 500$ (d) *Adult*, $|D_{exp}| = 500$

Figure 8: Performance comparison for *Dirichlet*(1.0) in the online setting on four datasets. DROPO-exp has the same level of performance compared to the baseline methods because the *Dirichlet*(1.0) logging policy is close to the uniform policy.



(a) *Optdigits*, $|D_{exp}| = 100$ (b) *MNIST* , $|D_{exp}| = 500$ (c) *CIFAR-10*, $|D_{exp}| = 500$ (d) *Adult*, $|D_{exp}| = 500$

Figure 9: Performance comparison for *Dirichlet*(0.1) in the online setting on four datasets. DROPO-exp has the same level of performance compared to the baseline methods because the *Dirichlet*(0.1) logging policy is close to the uniform policy.

DROPO is reduced to a direct method since we do not consider the covariate shift between the logging policy $\beta(a|x)$ and the current policy $\pi(a|x)$. In contrast, note that in existing literature, people use DM to refer the Vanilla Direct Method, which tries to minimize the mean squared error (MSE) loss on the logging data to obtain a reward mapping $\hat{r}(x, a)$.

$$\arg\min_{\hat{r}} \sum_{(x,a,r(x,a))\in\mathcal{S}} \left(r(x, a) - \hat{r}(x, a)\right)^2 .$$

The reason why we choose the non-shift variant of DROPO as the DM is that it is more straightforward to justify the benefit of considering distributional shift in policy learning by comparing the performance between our DM and DROPO.

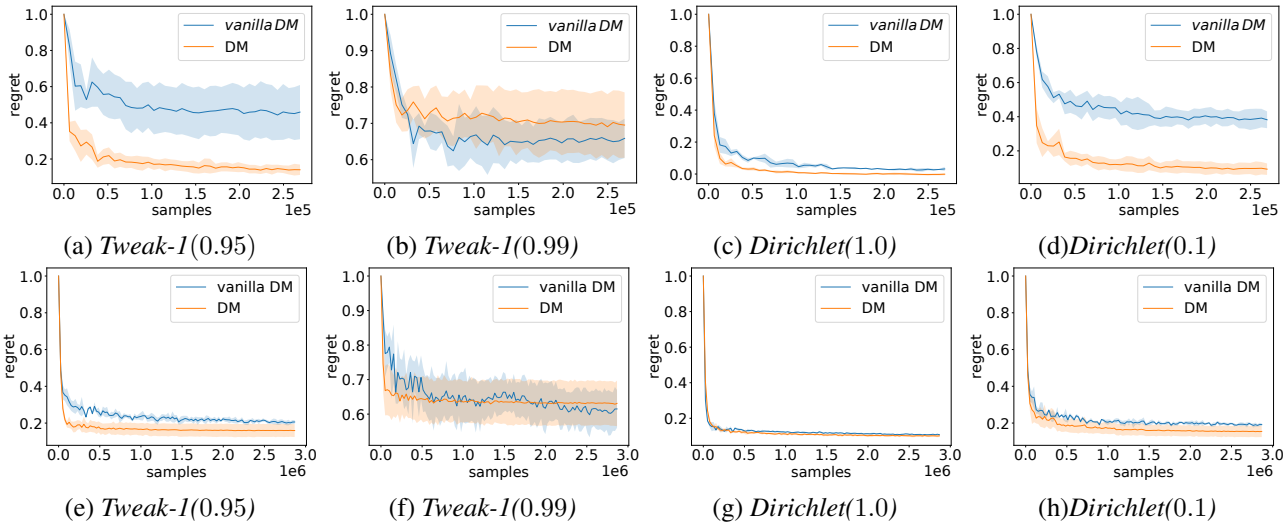
To justify our choice, we compare the results of Vanilla DM and our DM in Fig. 10. The experiment is under the pure offline setting. In general, our DM achieves better results compared to the Vanilla DM in both datasets and has a more stable regret curve. One explanation is that the reward estimation in our DM has a prior Gaussian distribution $f_0(r|x, a) \sim \mathcal{N}\left(\mu_0, \sigma_0^2\right)$, which helps regularize the method. Therefore, our distributionally robust learning framework benefits the learning under distributional shift even without explicitly incorporating the density ratios.

B.3.2 Different Exploration Size

We compare the results between online and offline settings for *Optdigits* and *MNIST* with different exploration sizes in Fig. 11 and Fig. 12. We run the online and offline experiments with the same number of epochs. The experiments under online setting start with a smaller logging dataset and have smaller combined dataset of logging data and exploration data than the offline setting at each epoch. So the regret curve for the online setting are shorter in length.

In *Tweak-1*(0.99) (large shift), DROPO-exp utilizes a smaller amount of data and receives a better or at least comparable performance than DROPO. Also, we see that online exploration improves the performance over the offline version more on baseline methods than ours. This is because DROPO is more robust to the data insufficiency in the offline training and the online data is less helpful compared to the baseline methods. Besides, as we increase the exploration size, DROPO-exp does not benefit a lot, while we can see a great improvement on the baselines methods. For one thing, DROPO is more robust to the data insufficiency. For another thing, with small amount of high quality data, DROPO already achieves good performance and more online exploration data is less helpful. But the baseline methods are still far from the best performance, so they will benefit more from the larger exploration size.

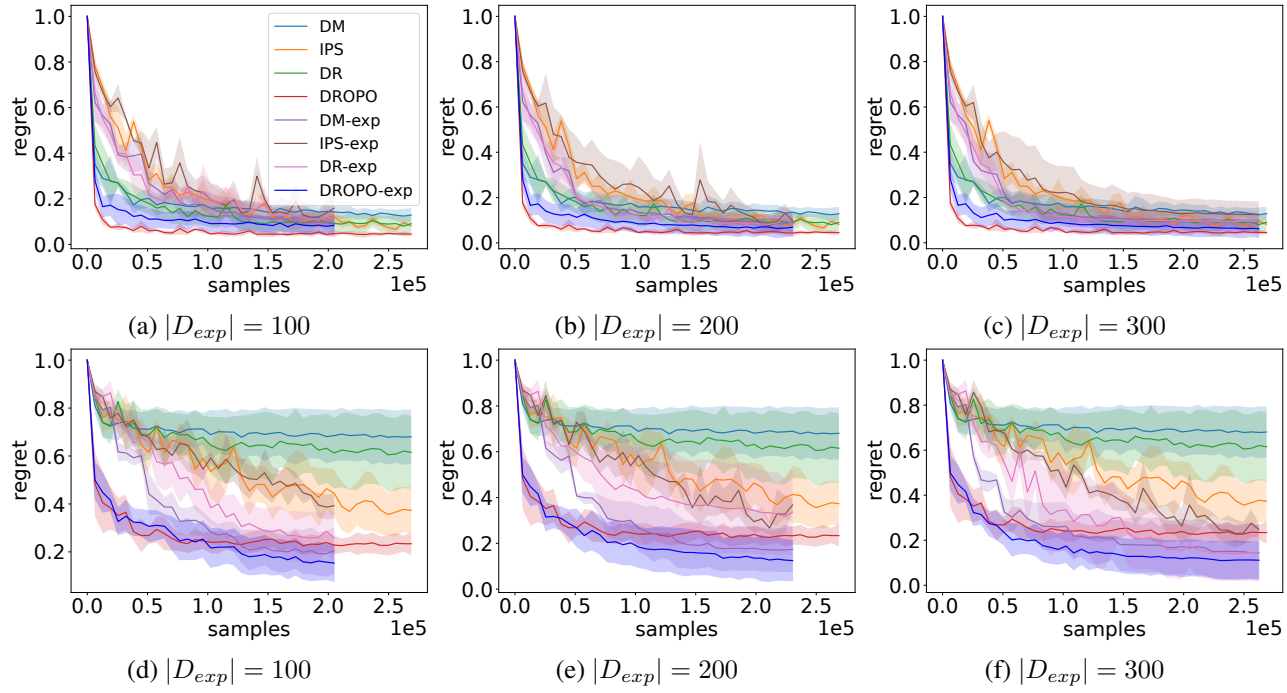
In *Tweak-1*(0.95) (smaller shift), DROPO-exp barely receives a better performance than DROPO. Our conjecture is that



(a) *Tweak-1*(0.95) (b) *Tweak-1*(0.99) (c) *Dirichlet*(1.0) (d)*Dirichlet*(0.1)

(e) *Tweak-1*(0.95) (f) *Tweak-1*(0.99) (g) *Dirichlet*(1.0) (h)*Dirichlet*(0.1)

Figure 10: **Upper row**: Experiments of comparison on DM on *Optdigits* in different distributional shift scenarios. **Bottom row**: Experiments of comparison on DM on *MNIST* in different distributional shift scenarios. In general, our DM outperforms Vanilla DM and has a more stable regret curve, except in *Tweak-1*(0.99) case.



(a) $|D_{exp}| = 100$ (b) $|D_{exp}| = 200$ (c) $|D_{exp}| = 300$

(d) $|D_{exp}| = 100$ (e) $|D_{exp}| = 200$ (f) $|D_{exp}| = 300$

Figure 11: Comparison between online and offline setting under different exploration size $|D_{exp}|$ on *Optdigits*. **Upper row**: *Tweak-1*(0.95). **Bottom row**: *Tweak-1*(0.99). Starting with a significantly smaller dataset, online exploration achieves on par or even better results. The improvement is more obvious in *Tweak-1*(0.99). As we increase the exploration size, our DROPO-exp is not significantly improved while the baseline methods benefit more. DROPO-exp still has the best performance compared to other baseline under online setting. **Different lengths** of the curve is due to that we run all the experiments with the same number of epochs but the epoch size is different across online and offline settings.

the online version starts with a smaller dataset than the offline version. In the large shift case, the smaller newly collected high-quality dataset is more important and informative than the larger logging dataset in our setting. However, in smaller shift case, the situation is potentially the opposite. For future work, it would be interesting to explore more on how to effectively conduct the limited online exploration and select the optimal exploration size with the trade-off between the

performance and the exploration cost.

B.3.3 IPS Gradient in Online Update

We explore whether the IPS gradient on the exploration data will benefit our gradient estimation. Since the latest exploration data is collected from the current policy, which does not have a large shift as in the logging data, we can directly calculate the IPS gradient at iteration t by

$$g_{IPS}(\boldsymbol{\theta}_t) = \frac{1}{|\mathcal{D}_{exp}|} \sum_{(x,a,r)\in\mathcal{D}_{exp}} \frac{\pi_{\boldsymbol{\theta}_t}(a|x)}{\beta_t(a|x)} \nabla \log \pi_{\boldsymbol{\theta}_t}(a|x) r.$$

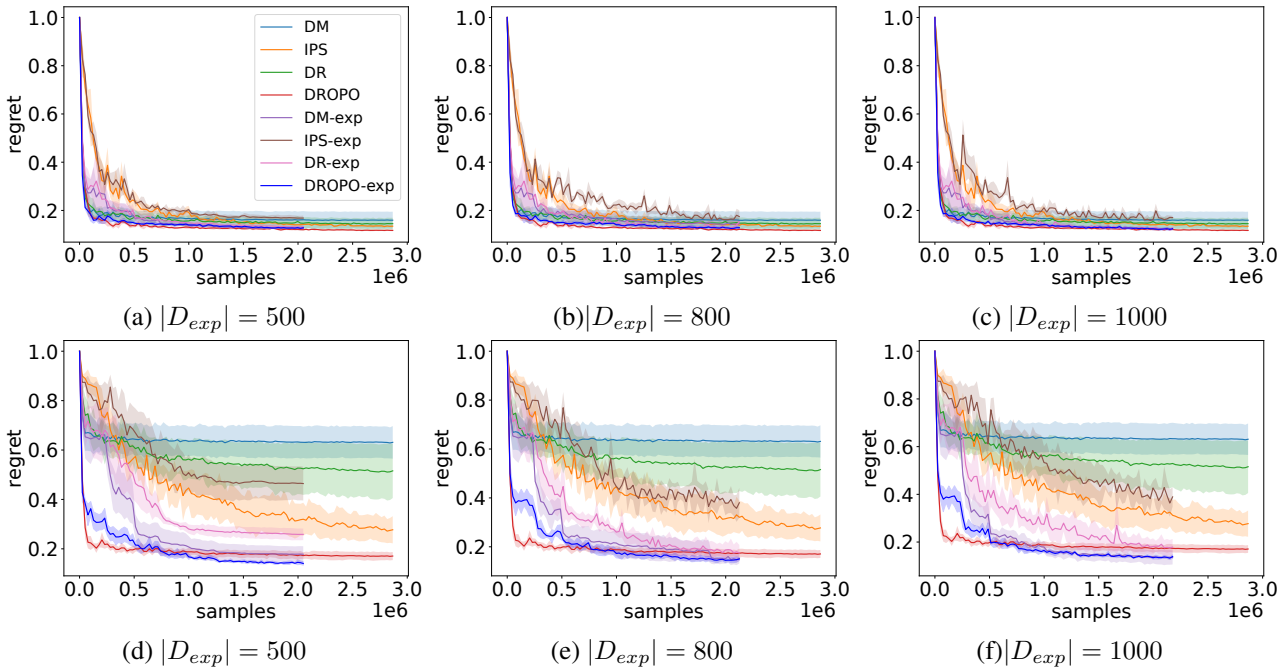
In practice, we use the mini-batch gradient ascent for policy gradient optimization and batch-size $\neq |D_{exp}|$. Moreover, since the exploration data is small, we calculate the IPS gradient on the entire exploration data. At each mini-batch gradient ascent, the gradient is the weighted linear combination of the batch gradient and the IPS gradient, written as

$$\hat{g}(\boldsymbol{\theta}_t) = \tau \frac{1}{|\mathcal{S}_{\pi_{\theta_t},\text{batch}}|} \sum_{i\in\mathcal{S}_{\pi_{\theta_t},\text{batch}}} \nabla \log \pi_{\boldsymbol{\theta}_t}(a_i|x_i) \hat{r}_{\mathcal{D}_{\log}}(x_i,a_i) + (1-\tau) g_{IPS}(\boldsymbol{\theta}_t),$$

where $\tau \in [0,1]$ is a hyper-parameter.

Fig. 13 shows how different τ affects the result. Similarly, we run the online and offline experiments with the same epochs. The experiments under online setting start with a smaller logging dataset and have smaller combined dataset of logging data and exploration data than the offline setting at each epoch. Hence the regret curve for the online setting are shorter in length.

The IPS gradient correction does not help with the performance but sometimes causes unstable training. For example, in Fig. 13 (g), there is a sudden increase on the regret with $\tau = 0.3$ and $\tau = 0.4$. We have a conjecture that IPS gradient has a large variance even when we are using the combination of the batch gradient and the IPS gradient. This makes the training

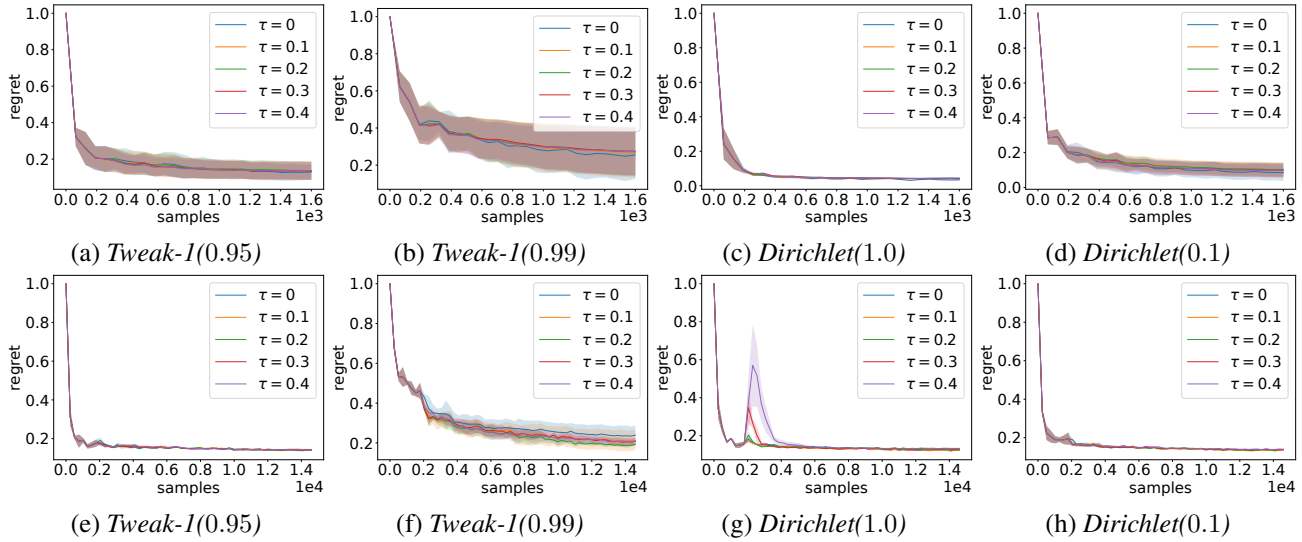


(a) $|D_{exp}| = 500$ (b) $|D_{exp}| = 800$ (c) $|D_{exp}| = 1000$

(d) $|D_{exp}| = 500$ (e) $|D_{exp}| = 800$ (f) $|D_{exp}| = 1000$

Figure 12: Comparison between online and offline settings under different exploration sizes $|D_{exp}|$ on *MNIST*. **Upper row**: *Tweak-1*(0.95). **Bottom row**: *Tweak-1*(0.99). Starting with a significantly smaller dataset, online exploration achieves on-par or even better results. The improvement is more obvious in *Tweak-1*(0.99). As we increase $|D_{exp}|$, our DROPO-exp is not significantly improved but the baseline methods have significant improvement. DROPO-exp still has the best performance compared to other baselines in online settings. **Different lengths** of the curve is due to that we run all the experiments with the same number of epochs but the epoch size is different across online and offline settings.

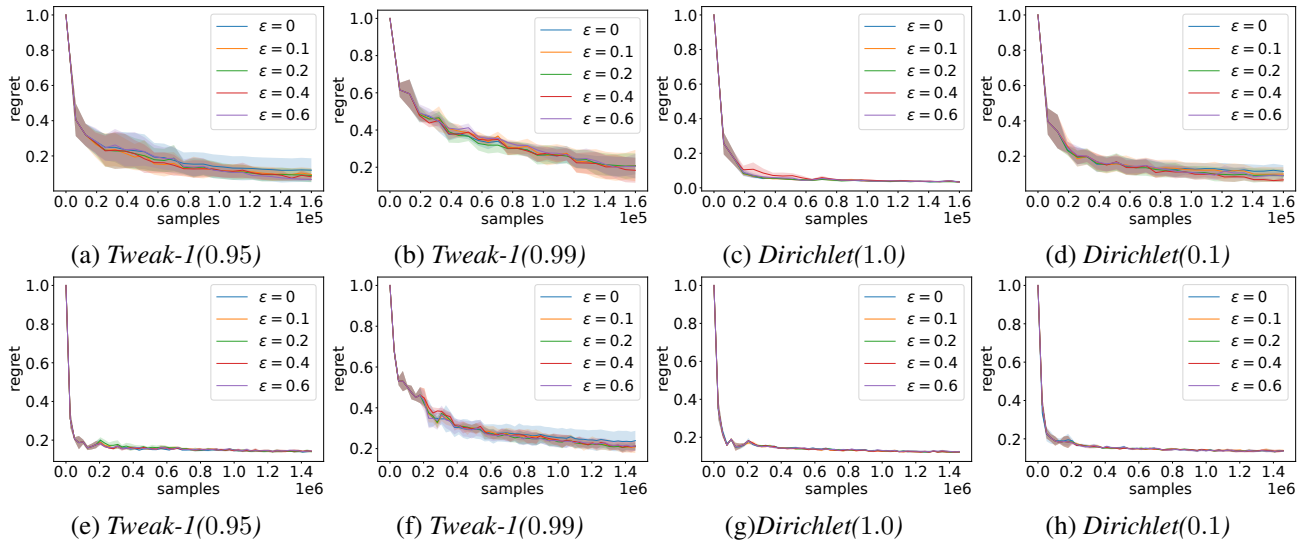
unstable. In our experiment, we need to set a large learning rate decay to mitigate the unstable training caused by a large variance of IPS gradient, otherwise the regret curve will have a large fluctuation. A promising future direction is to study more advanced methods to account for the high variance in online updates for policy learning.



(a) *Tweak-1*(0.95) (b) *Tweak-1*(0.99) (c) *Dirichlet*(1.0) (d) *Dirichlet*(0.1)

(e) *Tweak-1*(0.95) (f) *Tweak-1*(0.99) (g) *Dirichlet*(1.0) (h) *Dirichlet*(0.1)

Figure 13: **Upper row**: Experiments of different τ for IPS gradient on *Optdigits* in different distributional shift scenarios. Limited online exploration is permitted, $|D_{exp}| = 100$. **Bottom row**: Experiments of different τ for IPS gradient on *MNIST* in different distributional shift scenarios. Limited online exploration is permitted, $|D_{exp}| = 500$. The IPS gradient in online update does not improve the performance but induces instability in training.



(a) *Tweak-1*(0.95) (b) *Tweak-1*(0.99) (c) *Dirichlet*(1.0) (d) *Dirichlet*(0.1)

(e) *Tweak-1*(0.95) (f) *Tweak-1*(0.99) (g)*Dirichlet*(1.0) (h) *Dirichlet*(0.1)

Figure 14: **Upper row**: Experiments on different ϵ for uniform sampling on *Optdigits* in different distributional shift scenarios. Limited online exploration is permitted, $|D_{exp}| = 100$. **Bottom row**: Experiments on different ϵ for uniform sampling on *MNIST* in different distributional shift scenarios. Limited online exploration is permitted, $|D_{exp}| = 500$. $\epsilon > 0$ tends to have better result in (a), (b), (d) and (f) as the training progresses to the end and uniform sampling data accumulates.

B.4 Alternative Online Data Collection

With limited online exploration, we want to explore if there is a better way to utilize the online exploration. In large shift scenarios, like the *Tweak-1*(0.99), the probabilities in logging policy for most of the actions are low, which means there exists a low coverage of those actions. This will cause a high bias in the reward estimation and gradient estimation. To

solve the low coverage problem, we propose an $\epsilon - \pi$ exploration policy for online exploration. Instead of sampling the action with the current policy, we sample the action with a uniform policy with probability ϵ, and sample the action with the current policy with probability $1 - \epsilon$.

Fig. 14 shows how different ϵ affects the result. As depicted in Fig. 14 (a), (b), (d) and (f), the $\epsilon > 0$ cases tend to have better a performance compared with the $\epsilon = 0$ case. However, the improvement is insignificant, since we can only observe it at the last few epochs. In our experiment, the budget for the online exploration is limited, and the exploration under the uniform policy is performed only with a small probability. Therefore, we are actually collecting only a very small amount of data with the uniform policy. This offsets the advantages brought by uniform sampling, namely more informative logging data and less biased reward estimation.

Based on the same logging dataset, the combined dataset of exploration data and logging data is similar across different ϵ at the first few epochs, because the number of exploration data is significantly smaller than the logging data. However, as the training progresses and the data sampled with uniform policy accumulates, the difference between the combined dataset will grow larger. This is when the regret curves in Fig. 14 (a), (b), (d) and (f) start to deviate.

Note that, larger ϵ does not always bring a better performance. The data collected from the current policy are more valuable for policy learning, while the data collected using the uniform policy are not as effective in policy gradient because of the distributional shift between the uniform policy and the current policy. Therefore, there exists a trade-off between reward estimation and policy learning as they require different types of training data. How to balance them in offline policy learning with limited online exploration is an interesting future direction.

B.5 Code Repository

The code repository for the experiments can be found in: https://github.com/guoyihonggyh/Distributionally-Robust-Policy-Gradient-for-Offline-Contextual-Bandits