Open Bandit Dataset and Pipeline: Towards Realistic and Reproducible Off-Policy Evaluation

Yuta Saito
Hanjuku-kaso, Co., Ltd.
saito@hanjuku-kaso.com

Shunsuke Aihara
ZOZO Technologies, Inc.
shunsuke.aihara@zozo.com

Megumi Matsutani
ZOZO Technologies, Inc.
megumi.matsutani@zozo.com

Yusuke Narita
Yale University
yusuke.narita@yale.edu

ABSTRACT

Off-policy evaluation (OPE) aims to estimate the performance of hypothetical policies using data generated by a different policy. Because of its huge potential impact, there has been growing research interest in OPE. There is, however, no real-world public dataset that enables the evaluation of OPE, making its experimental studies unrealistic and irreproducible. With the goal of enabling realistic and reproducible OPE research, we publicize the *Open Bandit Dataset* collected on a large-scale fashion e-commerce platform, ZOZOTOWN. Our dataset is unique in that it contains a set of *multiple* logged bandit feedback datasets collected by running different policies on the same platform. This enables realistic and reproducible experimental comparisons of different OPE estimators for the first time. We also develop Python software called the *Open Bandit Pipeline* to streamline and standardize the implementations of bandit algorithms and OPE. Our open data and pipeline will contribute to the fair and transparent OPE research and help the community identify fruitful research directions. Finally, we provide extensive benchmark experiments of existing OPE estimators using our data and pipeline. Our experiments open up essential challenges and new avenues for future OPE research. Our pipeline and example data are available at **https://github.com/st-tech/zr-obp**. The extended version including detailed experimental results is available at **https://arxiv.org/abs/2008.07146**.

ACM Reference Format:
Yuta Saito, Shunsuke Aihara, Megumi Matsutani, and Yusuke Narita. 2018. Open Bandit Dataset and Pipeline: Towards Realistic and Reproducible Off-Policy Evaluation . In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Interactive bandit and reinforcement learning systems (e.g., personalized medicine, ad/recommendation/search platforms) produce log data valuable for evaluating and redesigning the system. For example, the logs of a news recommendation system record which news article was presented and whether the user read it, giving the system designer a chance to make its recommendations more relevant. Exploiting log bandit data is, however, more difficult than conventional supervised machine learning: the result is only observed for the action chosen by the system, but not for all the other actions that the system could have taken. The logs are also biased in that they overrepresent the actions favored by the system. A potential solution to this problem is an A/B test that compares the performance of counterfactual systems in an online environment. However, A/B testing counterfactual systems is often difficult because deploying a new policy is time- and money-consuming and entails risks of failure. This leads us to the problem of *off-policy evaluation* (OPE), which aims to estimate the performance of a counterfactual (or evaluation) policy using only log data collected by a past (or behavior) policy. OPE allows us to compare the performance of candidate counterfactual policies without implementing A/B tests and contributes to safe policy improvements. Its applications range from contextual bandits [2, 17, 18, 24, 28, 30–32, 37, 40] and reinforcement learning in the web industry [7, 13–15, 20, 33–35, 41] to other social domains such as healthcare [23] and education [21].

Issues with current experimental procedures. Although the research community has produced theoretical breakthroughs, the experimental evaluation of OPE remains primitive. Specifically, it lacks a public benchmark dataset for comparing the performance of different methods. Researchers often validate their methods using synthetic simulation environments [15, 20, 37, 39, 41]. A version of the synthetic approach is to modify multiclass classification datasets and treat supervised machine learning methods as bandit policies to evaluate the estimation accuracy of OPE estimators [5, 7, 38, 40]. An obvious problem with these studies is that they are **unrealistic** because there is no guarantee that their simulation environment is similar to real-world settings. To solve this issue, Gilotte et al. [8], Gruson et al. [10], Narita et al. [24, 25] use proprietary real-world datasets. Because these datasets are not public, however, the results are **irreproducible**, and it remains challenging to compare their methods with new ideas in a fair manner. This is in contrast to other domains of machine learning, where large-scale open datasets, such as the ImageNet dataset [4], have been pivotal in driving objective progress [6, 9, 11, 12].

Contributions. Our goal is to implement and evaluate OPE in **realistic and reproducible** ways. To this end, we release the *Open*

Woodstock '18, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

Bandit Dataset, a set of logged bandit feedback datasets collected on a fashion e-commerce platform, ZOZOTOWN.[1] ZOZOTOWN is the largest fashion e-commerce platform in Japan, with an annual gross merchandise value of over 3 billion US dollars. When the platform produced the data, it used Bernoulli Thompson Sampling (Bernoulli TS) [36] and uniform random (Random) policies to recommend fashion items to users. The dataset includes a set of *multiple* logged bandit feedback datasets collected during an A/B test of these bandit policies. Having multiple log datasets is essential because it enables data-driven evaluation of the estimation accuracy of OPE methods.

In addition to the dataset, we also implement the *Open Bandit Pipeline*, an open-source Python software including a series of modules for implementing dataset preprocessing, policy learning methods, and OPE estimators. Our software provides a complete, standardized experimental procedure for OPE research, ensuring that performance comparisons are fair, transparent, and reproducible. It also enables fast and accurate OPE implementation through a single unified interface, simplifying the practical use of OPE.

Using our dataset and pipeline, we perform an extensive benchmark experiment on existing estimators. Specifically, we implement this by using the log data of one of the policies (e.g., Bernoulli TS) to estimate the policy value of the other policy (e.g., Random) with each OPE estimator. We then assess the accuracy of the estimator by comparing its estimation with the policy value obtained from the data in an *on-policy* manner. This type of data-driven evaluation of OPE is possible with our dataset, because it contains multiple different logged bandit feedback datasets. With our unique real-world dataset, our benchmark experiment is the first empirical study comparing a variety of OPE estimators in a realistic and reproducible manner.

In the benchmark experiment, we obtain the following observations:

- The estimation performances of all OPE estimators drop significantly when they are applied to estimate the future (or out-sample) performance of a new policy.
- The estimation performances of OPE estimators heavily depend on experimental settings and hyperparameters.

These empirical findings lead to the following future research directions: (i) improving out-of-distribution estimation performance and (ii) developing methods to identify appropriate OPE estimators for various settings.

We summarize our key contributions as follows:

- (**Dataset Release**) We build and release the *Open Bandit Dataset*, a set of *multiple* logged bandit feedback datasets to assist realistic and reproducible research about OPE.
- (**Pipeline Implementation**) We implement the *Open Bandit Pipeline*, an open-source Python software that helps practitioners implement OPE to evaluate their bandit systems and researchers compare different OPE estimators in a standardized manner.
- (**Benchmark Experiment**) We perform comprehensive benchmark experiments on existing OPE methods and indicate critical challenges in future research.

[1] https://corp.zozo.com/en/service/

2 OFF-POLICY EVALUATION

2.1 Setup

We consider a general contextual bandit setting. Let $r \in [0, r_{\max}]$ denote a reward variable (e.g., whether a fashion item as an action results in a click). We let $x \in \mathcal{X}$ be a context vector (e.g., the user's demographic profile) that the decision maker observes when picking an action. Rewards and contexts are sampled from unknown probability distributions $p(r \mid x, a)$ and $p(x)$, respectively. Let $\mathcal{A}$ be a finite set of actions. We call a function $\pi : \mathcal{X} \to \Delta(\mathcal{A})$ a *policy*. It maps each context $x \in \mathcal{X}$ into a distribution over actions, where $\pi(a \mid x)$ is the probability of taking action a given context x.

Our setup allows for many popular multi-armed bandit algorithms and off-policy learning methods, as the following examples illustrate.

Example 1 (Random A/B testing). *We always choose each action uniformly at random, i.e., $\pi_{\mathrm{Uniform}}(a \mid x) = |\mathcal{A}|^{-1}$ always holds for any given $a \in \mathcal{A}$ and $x \in \mathcal{X}$.*

Example 2 (Bernoulli Thompson Sampling). *We sample the potential reward $\tilde{r}(a)$ from the beta distribution $Beta(S_{ta}+\alpha, F_{ta}+\beta)$ for each action in $\mathcal{A}$, where $S_{ta} := \sum_{t'=1}^{t-1} r_{t'}$, $F_{ta} := (t-1) - S_{ta}$. (α, β) are the parameters of the prior Beta distribution. We then choose the action with the highest sampled potential reward, $a :\in \operatorname{argmax}_{a' \in \mathcal{A}} \tilde{r}(a')$ (ties are broken arbitrarily). As a result, this algorithm chooses actions with the following probabilities:*

$$\pi_{\mathrm{BernoulliTS}}(a \mid x) = \Pr\{a \in \operatorname*{argmax}_{a' \in \mathcal{A}} \tilde{r}(a')\}$$

for any given $a \in \mathcal{A}$ and $x \in \mathcal{X}$. When implementing the data collection experiment on the ZOZOTOWN platform, we modified TS to adjust to our top-3 recommendation setting shown in Figure 1. The modified TS selects three actions with the three highest sampled rewards which create a non-repetitive set of item recommendations for each comming user.

Example 3 (IPW Learner). *When $\mathcal{D}$ is given, we can train a deterministic policy $\pi_{\mathrm{det}} : \mathcal{X} \to \mathcal{A}$ by maximizing the IPW estimator as follows:*

$$\begin{aligned}
\pi_{\mathrm{det}}(x) &\in \operatorname*{argmax}_{\pi \in \Pi} \hat{V}_{\mathrm{IPW}}(\pi; \mathcal{D}) \\
&= \operatorname*{argmax}_{\pi \in \Pi} \mathbb{E}_{\mathcal{D}} \left[\frac{\mathbb{I}\{\pi(x_t) = a_t\}}{\pi_b(a_t \mid x_t)} r_t \right] \\
&= \operatorname*{argmin}_{\pi \in \Pi} \mathbb{E}_{\mathcal{D}} \left[\frac{r_t}{\pi_b(a_t \mid x_t)} \mathbb{I}\{\pi(x_t) \neq a_t\} \right]
\end{aligned}$$

, which is equivalent to the cost-sensitive classification problem that can be solved with an arbitrary machine learning classifier.

Let $\mathcal{D} := \{(x_t, a_t, r_t)\}_{t=1}^{T}$ be the historical logged bandit feedback with T rounds of observations. a_t is a discrete variable indicating which action in $\mathcal{A}$ is chosen in round t. r_t and x_t denote the reward and the context observed in round t, respectively. We assume that a logged bandit feedback is generated by a *behavior policy* π_b as follows:

$$\{(x_t, a_t, r_t)\}_{t=1}^{T} \sim \prod_{t=1}^{T} p(x_t)\pi_b(a_t \mid x_t)p(r_t \mid x_t, a_t),$$

where each context–action–reward triplet is sampled independently from the product distribution.

For a function $f(x, a, r)$, let

$$\mathbb{E}_{\mathcal{D}}[f] := |\mathcal{D}|^{-1} \sum_{(x_t,a_t,r_t)\in\mathcal{D}} f(x_t, a_t, r_t)$$

denote its empirical expectation over $\mathcal{D}$. Then, for a function $g(x, a)$, we let $g(x, \pi) := \mathbb{E}_{a\sim\pi(a|x)}[g(x, a) \mid x]$. We also use $q(x, a) := \mathbb{E}_{r\sim p(r|x,a)}[r \mid x, a]$ to denote the mean reward function.

2.2 Estimation Target

We are interested in using historical logged bandit data to estimate the following *policy value* of any given *evaluation policy* π_e, which might be different from π_b:

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

Estimating $V(\pi_e)$ before implementing π_e in an online environment is valuable because π_e may perform poorly and damage user satisfaction. Additionally, it is possible to select a promising policy by comparing the performance of candidate policies estimated by OPE without A/B tests.

2.3 Existing Estimators

Here, we summarize several existing OPE methods.

Direct Method (DM).. DM [1] first estimates the mean reward function using a supervised machine learning model, such as random forest or ridge regression. It then plugs it in to estimate the policy value as

$$\hat{V}_{\mathrm{DM}}(\pi_e; \mathcal{D}, \hat{q}) := \mathbb{E}_{\mathcal{D}}[\hat{q}(x_t, \pi_e)],$$

where $\hat{q}(x, a)$ is a reward estimator. If $\hat{q}(x, a)$ is a good approximation of the mean reward function, DM estimates the policy value accurately. If $\hat{q}(x, a)$ fails to approximate the mean reward function well, however, the final estimator is no longer consistent.

Inverse Probability Weighting (IPW).. To alleviate the issue with DM, researchers often use IPW [26, 28]. IPW re-weights the observed rewards by the importance weight as

$$\hat{V}_{\mathrm{IPW}}(\pi_e; \mathcal{D}) := \mathbb{E}_{\mathcal{D}}[w(x_t, a_t)r_t],$$

where $w(x, a) := \pi_e(a \mid x)/\pi_b(a \mid x)$. When the behavior policy is known, IPW is unbiased and consistent for the policy value. However, it can have a large variance, especially when the evaluation policy deviates significantly from the behavior policy.

Doubly Robust (DR).. DR [5] combines DM and IPW as

$$\hat{V}_{\mathrm{DR}}(\pi_e; \mathcal{D}, \hat{q}) := \mathbb{E}_{\mathcal{D}}[\hat{q}(x_t, \pi_e) + w(x_t, a_t)(r_t - \hat{q}(x_t, a_t))].$$

DR mimics IPW to use a weighted version of rewards, but it also uses $\hat{q}$ as a control variate to decrease the variance. It preserves the consistency of IPW if either the importance weight or the reward estimator is consistent (a property called *double robustness*).

Self-Normalized Estimators. Self-Normalized Inverse Probability Weighting (SNIPW) is an approach to address the variance issue with the original IPW. It estimates the policy value by dividing the sum of weighted rewards by the sum of importance weights as:

$$\hat{V}_{\mathrm{SNIPW}}(\pi_e; \mathcal{D}) := \frac{\mathbb{E}_{\mathcal{D}}[w(x_t, a_t)r_t]}{\mathbb{E}_{\mathcal{D}}[w(x_t, a_t)]}.$$

SNIPW is more stable than IPW, because policy value estimated by SNIPW is bounded in the support of rewards and its conditional variance given action and context is bounded by the conditional variance of the rewards [15]. IPW does not have these properties. We can define Self-Normalized Doubly Robust (SNDR) in a similar manner as follows.

$$\hat{V}_{\mathrm{SNDR}}(\pi_e; \mathcal{D}) := \mathbb{E}_{\mathcal{D}}\left[\hat{q}(x_t, \pi_e) + \frac{w(x_t, a_t)(r_t - \hat{q}(x_t, a_t))}{\mathbb{E}_{\mathcal{D}}[w(x_t, a_t)]}\right].$$

Switch Estimators. The DR estimator can still be subject to the variance issue, particularly when the importance weights are large due to weak overlap. Switch-DR aims to reduce the effect of the variance issue by using DM where importance weights are large as:

$$\begin{aligned}&\hat{V}_{\mathrm{SwitchDR}}(\pi_e; \mathcal{D}, \hat{q}, \tau)\\&:= \mathbb{E}_{\mathcal{D}}\left[\hat{q}(x_t, \pi_e) + w(x_t, a_t)(r_t - \hat{q}(x_t, a_t))\mathbb{I}\{w(x_t, a_t) \le \tau\}\right],\end{aligned}$$

where $\mathbb{I}\{\cdot\}$ is the indicator function and $\tau \ge 0$ is a hyperparameter. Switch-DR interpolates between DM and DR. When $\tau = 0$, it coincides with DM, while $\tau \to \infty$ yields DR.

More Robust Doubly Robust (MRDR).. MRDR uses a specialized reward estimator ($\hat{q}_{\mathrm{MRDR}}$) that minimizes the variance of the resulting policy value estimator [7]. This estimator estimates the policy value as:

$$\hat{V}_{\mathrm{MRDR}}(\pi_e; \mathcal{D}, \hat{q}_{\mathrm{MRDR}}) := \hat{V}_{\mathrm{DR}}(\pi_e; \mathcal{D}, \hat{q}_{\mathrm{MRDR}}),$$

where $\hat{q}_{\mathrm{MRDR}}$ is derived by minimizing the (empirical) variance objective:

$$\hat{q}_{\mathrm{MRDR}} \in \operatorname*{argmin}_{\hat{q}\in Q} \mathbb{V}_{\mathcal{D}}(\hat{V}_{\mathrm{DR}}(\pi_e; \mathcal{D}, \hat{q})),$$

where Q is a function class for the reward estimator. When Q is well-specified, then $\hat{q}_{\mathrm{MRDR}} = q$. Here, even if Q is misspecified, the derived reward estimator is expected to behave well since the target function is the resulting variance.

Doubly Robust with Optimistic Shrinkage (DRos). Su et al. [29] proposes DRos based on a new weight function $w_o : \mathcal{X} \times \mathcal{A} \to \mathbb{R}_+$ that directly minimizes sharp bounds on the MSE of the resulting estimator. DRos is defined as

$$\hat{V}_{\mathrm{DRos}}(\pi_e; \mathcal{D}, \hat{q}, \lambda) := \mathbb{E}_{\mathcal{D}}[\hat{q}(x_t, \pi_e) + w_o(x_t, a_t; \lambda)(r_t - \hat{q}(x_t, a_t))],$$

where $\lambda \ge 0$ is a pre-defined hyperparameter and the new weight is

$$w_o(x, a; \lambda) := \frac{\lambda}{w^2(x, a) + \lambda} w(x, a).$$

When $\lambda = 0$, $w_o(x, a; \lambda) = 0$ leading to the standard DM. On the other hand, as $\lambda \to \infty$, $w_o(x, a; \lambda) = w(x, a)$ leading to the original DR.

Table 1: Statistics of the Open Bandit Dataset

Campaigns	Data Collection Policies	#Data	#Items	Average Age	CTR ($V(\pi)$) ±95% CI	Relative-CTR
ALL	**Random**	1,374,327	80	37.93	0.35% ±0.010	1.00
	Bernoulli TS	12,168,084			0.50% ±0.004	1.43
Men's	**Random**	452,949	34	37.68	0.51% ±0.021	1.48
	Bernoulli TS	4,077,727			0.67% ±0.008	1.94
Women's	**Random**	864,585	46	37.99	0.48% ±0.014	1.39
	Bernoulli TS	7,765,497			0.64% ±0.056	1.84

Note: Bernoulli TS stands for Bernoulli Thompson Sampling. **#Data** is the total number of user impressions observed during the 7-day experiment. **#Items** is the total number of items having a non-zero probability of being recommended by each policy. **Average Age** is the average age of users in each campaign. **CTR** is the percentage of a click being observed in log data, and this is the performance of the data collection policies for each campaign. The 95% confidence interval (CI) of CTR is calculated based on a normal approximation of the Bernoulli sampling. **Relative-CTR** is the CTR relative to that of the Random policy for the "ALL" campaign.

Figure 1: Fashion items as actions displayed in ZOZOTOWN recommendation interface. Three fashion items are simultaneously presented to a user in each recommendation.

3 OPEN-SOURCE DATASET AND PIPELINE

Motivated by the paucity of real-world datasets and implementations enabling the data-driven evaluation of OPE, we release the following open-source dataset and software.

Open Bandit Dataset. Our open-source dataset is a set of *multiple* logged bandit feedback datasets provided by ZOZO, Inc., the largest Japanese fashion e-commerce company. The company uses multi-armed bandit algorithms to recommend fashion items to users in their large-scale fashion e-commerce platform called ZOZOTOWN. We present examples of displayed fashion items in Figure 1. We collected the data in a 7-day experiment in late November 2019 on three "campaigns," corresponding to "ALL", "Men's", and "Women's" items, respectively. Each campaign randomly uses either the Random policy or the Bernoulli TS policy for each user impression.[2] These policies select three of the candidate fashion items for each user. Let $\mathcal{I}$ be a set of *items*, and let $\mathcal{K}$ be a set of *positions*. Figure 1 shows that $|\mathcal{K}| = 3$ for our data. We assume that the reward (click indicator) depends only on the item and its position, which is a general assumption on the click generative model in the web industry [19]. Under this assumption, we can apply the OPE setup and estimators in Section 2 to our dataset. We provide some statistics of the dataset in Table 1. The dataset is large and contains many millions of recommendation instances. It also includes the probabilities that item a is displayed at position k by the data collection policies which are used to calculate the importance weight.[3] We share the full version of our dataset at **https://research.zozo.com/data.html**.

Open Bandit Pipeline. To facilitate the use of OPE in practice and standardize its experimental procedure, we also build a Python package called the *Open Bandit Pipeline*.[4] Our pipeline contains the following main modules:

- The **dataset** module provides a data loader for the Open Bandit Dataset and tools to generate synthetic bandit datasets. It also implements a class to handle multiclass classification datasets as bandit feedback in OPE experiments.
- The **policy** module implements several online bandit algorithms and off-policy learning methods such as the one maximizing the IPW objective with only logged bandit data. This module also implements interfaces for implementing new policies so that practitioners can easily evaluate their own policies using OPE.
- The **ope** module implements several existing OPE estimators including the basic ones such as DM, IPW, and DR and some advanced ones such as Switch [40], More Robust Doubly Robust (MRDR) [7], and DR with Optimistic Shrinkage (DRos) [29]. This module also implements interfaces for implementing new estimators so that researchers can test their own estimation methods with our pipeline.

We also provide thorough documentation of the pipeline so that anyone can follow its usage. This pipeline allows researchers to focus on building their OPE estimator and to easily compare it with other methods in realistic and reproducible ways. Every core function of the packages is tested[5] and thus are well maintained. The package

[2] Note that we pre-trained Bernoulli TS for over a month before the data collection process, and the policy well converges to a fixed one. Therefore, our dataset fits the standard OPE formulation, that assumes fixed behavior and evaluation policies.

[3] We computed the true action choice probabilities by Monte Carlo simulations based on the policy parameters (e.g., parameters of the beta distribution used by Bernoulli TS) used during the data collection process.

[4] Our pipeline is available at **https://github.com/st-tech/zr-obp**.

[5] https://github.com/st-tech/zr-obp/tree/master/tests

currently has five core contributors[6]. The active development and maintenance will continue in a long period.

To our knowledge, our open-source dataset is the first to include logged bandit datasets collected by running *multiple* different policies and the exact policy implementations used in real production, enabling "***realistic and reproducible evaluation of OPE***" for the first time.

4 RELATED RESOURCES

In this section, we summarize existing related resources and clarify the differences between ours and previous ones.

4.1 Related Datasets

Our dataset is closely related to those of Lefortier et al. [16] and Li et al. [17]. Lefortier et al. [16] introduces a large-scale logged bandit feedback data (Criteo data) from a leading company in display advertising, Criteo. The data contain context vectors of user impressions, advertisements (ads) as actions, and click indicators as rewards. It also provides the ex-ante probability of each ad being selected by the behavior policy. Therefore, this dataset can be used to compare different off-policy *learning* methods, which aim to learn a new policy using only historical logged bandit data. In contrast, Li et al. [17] introduces a dataset (Yahoo! data) collected on a news recommendation interface of the Yahoo! Today Module. The data contain context vectors of user impressions, presented news as actions, and click indicators as rewards. The data were collected by running a uniform random policy on the news recommendation platform, allowing researchers to evaluate their own bandit algorithms.

However, the Criteo and Yahoo! datasets have several limitations, which we overcome as follows:

- They include only a single logged bandit feedback dataset collected by running a single policy. Moreover, the previous datasets do not provide the implementation to replicate the policies used during data collection. As a result, these datasets cannot be used for the evaluation and comparison of different OPE estimators, although they can be used to evaluate off-policy learning methods.
  → In contrast, we provide the code to replicate the data collection policies (i.e., Bernoulli TS and Random) in our pipeline, which allows researchers to rerun the same policies on the log data. Moreover, our open dataset consists of a set of *multiple* different logged bandit feedback datasets generated by running two different policies on the same platform. It enables the evaluation and comparison of different OPE estimators, as we show in Section 5. This is the first large-scale bandit dataset, enabling realistic and data-driven evaluation of OPE.
- The previous datasets do not provide a pipeline implementation for handling their data. Researchers have to reimplement the experimental environment by themselves before implementing their own OPE methods. This may lead to inconsistent experimental conditions across different studies, potentially causing reproducibility issues.
  → We implement the Open Bandit Pipeline to simplify and standardize the experimental processing of bandit algorithms and OPE. This tool thus contributes to the reproducible and transparent use of our dataset.

Table 2 summarizes key differences between our dataset and existing ones.

4.2 Related Packages

There are several existing Python packages related to the Open Bandit Pipeline. For example, the *contextualbandits* package[7] contains implementations of several contextual bandit algorithms [3]. It aims to provide an easy procedure to compare bandit algorithms to reproduce research papers that do not provide easily available implementations. In addition, *RecoGym*[8] focuses on providing simulation bandit environments imitating the e-commerce recommendation setting [27]. This package also implements an online bandit algorithm based on epsilon greedy and an off-policy learning method based on IPW.

However, the following features differentiate our pipeline from the previous ones:

- The previous packages focus on implementing and comparing online bandit algorithms or off-policy learning methods. However, they **cannot** be used to implement several advanced OPE estimators and the evaluation of OPE procedure.
  → Our package implements a wide variety of OPE estimators, including advanced ones such as Switch, MRDR, and DRos. Our package also provides flexible interfaces for implementing new OPE estimators. Consequently, researchers can easily compare their own estimators with other methods in a fair, standardized manner using our package.
- The previous packages cannot handle real-world bandit datasets.
  → Our package comes with the Open Bandit Dataset and includes the **dataset module**. This enables the evaluation of bandit algorithms and OPE estimators using real-world data. This function of our package contributes to realistic experiments on these topics.

Table 3 summarizes key differences between our pipeline and existing ones.

5 BENCHMARK EXPERIMENTS

We perform benchmark experiments of OPE estimators using the Open Bandit Dataset and Pipeline. We first describe an experimental protocol to evaluate OPE estimators and use it to compare a wide variety of existing estimators. We then discuss our initial findings in the experiments and indicate future research directions. We share the code for running the benchmark experiments at **https://github.com/st-tech/zr-obp/tree/master/benchmark**.

5.1 Experimental Protocol

We can empirically evaluate OPE estimators' performances by using two sources of logged bandit feedback collected by running two different policies. In the protocol, we regard one policy as behavior

[6] https://github.com/st-tech/zr-obp/graphs/contributors

[7] https://github.com/david-cortes/contextualbandits

[8] https://github.com/criteo-research/reco-gym

Table 2: Comparison of currently available large-scale bandit datasets

	Criteo Data [16]	**Yahoo! Data** [17]	**Open Bandit Dataset** (ours)
Domain	Display Advertising	News Recommendation	Fashion E-Commerce
Dataset Size	>103M	>40M	>26M (will increase)
#Data Collection Policies	1	1	**2 (will increase)**
Uniform Random Data	✗	✔	✔
Data Collection Policy Code	✗	✗	✔
Evaluation of Bandit Algorithms	✔	✔	✔
Evaluation of OPE	✗	✗	✔
Pipeline Implementation	✗	✗	✔

Note: **Dataset Size** is the total number of samples included in the whole dataset. **#Data Collection Policies** is the number of policies that were used to collect the data. **Uniform Random Data** indicates whether the dataset contains a subset of data generated by the uniform random policy. **Data Collection Policy Code** indicates whether the code to replicate data collection policies is publicized. **Evaluation of Bandit Algorithms** indicates whether it is possible to use the data to evaluate bandit algorithms. **Evaluation of OPE** indicates whether it is possible to use the dataset to evaluate OPE estimators. **Pipeline Implementation** indicates whether a pipeline tool to handle the dataset is available.

Table 3: Comparison of currently available packages of bandit algorithms and OPE

	contextualbandits [3]	**RecoGym** [27]	**Open Bandit Pipeline** (ours)
Synthetic Data Generator	✗	✔	✔
Classification Data Handler	✗	✗	✔
Support for Real-World Data	✗	✗	✔
Bandit Algorithms	✔	✔	✔
Basic OPE Estimators	✔	✗	✔
Advanced OPE Estimators	✗	✗	✔
Evaluation of OPE	✗	✗	✔

Note: **Synthetic Data Generator** indicates whether it is possible to create synthetic bandit data with the package. **Classification Data Handler** indicates whether it is possible to handle multiclass classification datas as bandit feedback with the package. **Support for Real-World Data** indicates whether it is possible to handle real-world bandit data with the package. **Bandit Algorithms** indicates whether the package includes implementations of online and offline bandit algorithms. **Basic OPE Estimators** indicates whether the package includes implementations of *basic* OPE estimators such as DM, IPW, and DR described in Section 2.3. **Advanced OPE Estimators** indicates whether the package includes implementations of *advanced* OPE estimators such as Switch and More Robust Doubly Robust. **Evaluation of OPE** indicates whether it is possible to evaluate the accuracy of OPE estimators with the package.

policy π_b and the other one as evaluation policy π_e. We denote log data generated by π_b and π_e as $\mathcal{D}^{(b)} := \{(x_t^{(b)}, a_t^{(b)}, r_t^{(b)})\}_{t=1}^{T^{(b)}}$ and $\mathcal{D}^{(e)} := \{(x_t^{(e)}, a_t^{(e)}, r_t^{(e)})\}_{t=1}^{T^{(e)}}$, respectively. Then, by applying the following protocol to several different OPE estimators, we can compare their estimation performances:

(1) Define the evaluation and test sets as
- (*in*-sample) $\mathcal{D}_{\mathrm{ev}} := \mathcal{D}^{(b)}_{1:T^{(b)}}$, $\mathcal{D}_{\mathrm{te}} := \mathcal{D}^{(e)}_{1:T^{(e)}}$
- (*out*-sample) $\mathcal{D}_{\mathrm{ev}} := \mathcal{D}^{(b)}_{1:\tilde{t}}$, $\mathcal{D}_{\mathrm{te}} := \mathcal{D}^{(e)}_{\tilde{t}+1:T^{(e)}}$

where $\mathcal{D}_{a:b} := \{(x_t, a_t, r_t)\}_{t=a}^{b}$.

(2) Estimate the policy value of π_e using $\mathcal{D}_{\mathrm{ev}}$ by $\hat{V}$. We can represent a policy value estimated by $\hat{V}$ as $\hat{V}(\pi_e; \mathcal{D}_{\mathrm{ev}})$.

(3) Estimate $V(\pi_e)$ by the *on-policy estimation* and regard it as the policy value of π_e, i.e.,[9]

$$V_{\mathrm{on}}(\pi_e) := \mathbb{E}_{\mathcal{D}_{\mathrm{te}}}[r_t^{(e)}].$$

(4) Compare the off-policy estimate $\hat{V}(\pi_e; \mathcal{D}_{\mathrm{ev}})$ with its ground-truth $V_{\mathrm{on}}(\pi_e)$. We can evaluate the estimation accuracy of $\hat{V}$ using the following *relative estimation error* (relative-EE):

$$\textit{relative-EE}(\hat{V}; \mathcal{D}_{\mathrm{ev}}) := \left| \frac{\hat{V}(\pi_e; \mathcal{D}_{\mathrm{ev}}) - V_{\mathrm{on}}(\pi_e)}{V_{\mathrm{on}}(\pi_e)} \right|.$$

(5) To estimate the standard deviation of relative-EE, repeat the above process several times with different bootstrap samples of the logged bandit data.

We call the problem setting **without** the sample splitting by time series as the *in*-sample case. In contrast, we call the setting **with** the

[9] Note that Table 1 presents $V_{\mathrm{on}}(\pi_e)$ for each pair of behavior policies and campaigns, and the small confidence intervals ensure that the on-policy estimation is accurate.

Table 4: Comparison of relative-estimation errors of OPE estimators (ALL Campaign)

	Random → Bernoulli TS		Bernoulli TS → Random	
OPE Estimators	*in*-sample	*out*-sample	*in*-sample	*out*-sample
DM	**0.23433** ±0.02131	**0.25730** ±0.02191	**0.34522** ±0.01020	**0.29422** ±0.01199
IPW	**0.05146** ±0.03418	0.09169 ±0.04086	**0.02341** ±0.02146	0.08255 ±0.03798
SNIPW	**0.05141** ±0.03374	**0.08899** ±0.04106	0.05233 ±0.02614	0.13374 ±0.04416
DR	0.05269 ±0.03460	0.09064 ±0.04105	0.06446 ±0.03001	0.14907 ±0.05097
SNDR	0.05269 ±0.03398	**0.09013** ±0.04122	0.04938 ±0.02645	0.12306 ±0.04481
Switch-DR ($\tau = 5$)	0.15350 ±0.02274	0.16918 ±0.02231	0.26811 ±0.00780	0.21945 ±0.00944
Switch-DR ($\tau = 10$)	0.09932 ±0.02459	0.12051 ±0.02203	0.21596 ±0.00907	0.16532 ±0.01127
Switch-DR ($\tau = 50$)	0.05269 ±0.03460	0.09064 ±0.04105	0.09769 ±0.01515	**0.04019** ±0.01349
Switch-DR ($\tau = 100$)	0.05269 ±0.03460	0.09064 ±0.04105	0.05938 ±0.01597	**0.01310** ±0.00988
Switch-DR ($\tau = 500$)	0.05269 ±0.03460	0.09064 ±0.04105	**0.02123** ±0.01386	0.06564 ±0.02132
Switch-DR ($\tau = 1000$)	0.05269 ±0.03460	0.09064 ±0.04105	0.02840 ±0.01929	0.05347 ±0.03330
DRos ($\lambda = 5$)	0.19135 ±0.01964	0.21240 ±0.01938	0.30395 ±0.00726	0.25216 ±0.00929
DRos ($\lambda = 10$)	0.17400 ±0.01993	0.19500 ±0.01885	0.28735 ±0.00706	0.23627 ±0.00899
DRos ($\lambda = 50$)	0.12867 ±0.02124	0.15155 ±0.01911	0.23876 ±0.00707	0.18855 ±0.00907
DRos ($\lambda = 100$)	0.11055 ±0.02241	0.13561 ±0.02080	0.21550 ±0.00744	0.16474 ±0.00942
DRos ($\lambda = 500$)	0.07715 ±0.02736	0.10915 ±0.02944	0.16055 ±0.00942	0.10601 ±0.01048
DRos ($\lambda = 1000$)	0.06739 ±0.02988	0.10187 ±0.03358	0.13717 ±0.01064	0.08034 ±0.01093
MRDR	0.05458 ±0.03386	0.09232 ±0.04169	0.02511 ±0.01735	0.08768 ±0.03821

Note: The averaged relative-estimation errors and their unbiased standard deviations estimated over 30 different bootstrapped iterations are reported. We describe the method to estimate the standard deviations in Appendix A. $\pi_b \rightarrow \pi_e$ represents the OPE situation where the estimators aim to estimate the policy value of π_e using logged bandit data collected by π_b. The **red** and **green** fonts represent the best and second-best estimators, respectively. The **blue** fonts represent the worst estimator for each setting.

sample splitting as the *out*-sample case, where OPE estimators aim to estimate the policy value of an evaluation policy in the future data. The standard OPE assumes the *in*-sample case where there are no distributional change in the environment over time. However, in practice, we aim to estimate the performance of an evaluation policy in the future, which may introduce the distributional change between the data used to conduct OPE and the environment that defines the policy value of the evaluation policy. We thus test the performance of OPE estimators in the *out*-sample case in addition to the *in*-sample case.

5.2 Estimators Compared

We use our protocol and compare the following OPE estimators: DM, IPW, Self-Normalized Inverse Probability Weighting (SNIPW), DR, Self-Normalized Doubly Robust (SNDR), Switch Doubly Robust (Switch-DR), DRos, and MRDR. These above estimators are defined in Section 2.3. We test different hyperparameter values for Switch-DR and DRos. These above estimators have not yet been compared in a large, realistic setting.

For estimators except for DM, we use the true action choice probability $\pi_b(a|x)$ contained in the Open Bandit Dataset. For estimators except for IPW and SNIPW, we need to obtain a reward estimator $\hat{q}$. We do this by using logistic regression (implemented in *scikit-learn*) and training it using 30% of $\mathcal{D}_{\text{ev}}$. We then use the rest of the data to estimate the policy value of an evaluation policy.

5.3 Results and Discussion

The results of the benchmark experiments on the “ALL" campaign are given in Table 4. (We report experimental results on the other campaigns in Appendix A.) We describe **Random → Bernoulli TS** to represent the OPE situation where we use Bernoulli TS as π_e and Random as π_b. Similarly, we use **Bernoulli TS → Random** to represent the situation where we use Random as π_e and Bernoulli TS as π_b.

Performance comparisons. First, DM fails to estimate the policy values in all settings due to the bias of the reward estimator. We observe that the reward estimator does not improve upon a naive estimation using the mean CTR for every estimation in the binary cross-entropy measure. (We present the performance of the reward estimator in Appendix A.) The problem with DM leads us to expect that the other estimators may perform better because they do not rely on the correct specification of the reward estimator. We confirm this expectation in Table 4, where one can see that the others drastically outperform DM. Among the other estimators, IPW, SNIPW, and MRDR exhibit stable estimation performances across different settings, and thus we can use these estimators safely. In **Bernoulli TS → Random**, Switch-DR performs the best with a proper hyperparameter configuration. Its performance, however, largely depends on the choice of hyperparameters, as we discuss later in detail. Note here that the performances of Switch-DR with some large hyperparameters are the same as that of DR. This is a natural observation, as their definitions are the same when the importance weights of all samples are lower than a given hyperparameter. In summary, we observe that simple estimators such as IPW and SNIPW perform better in many cases than Switch-DR and DRos even though these advanced methods performed well on synthetic experiments in previous studies. This suggests that evaluating the performance of OPE methods with only synthetic or classification datasets may produce impractical conclusions about the estimators' empirical properties. In contrast, our dataset enables researchers to produce more practical conclusions about OPE methods.

Out-sample generalization of OPE.. Next, we compare the estimation accuracy of each estimator between the *in*-sample and *out*-sample situations. Table 4 shows that estimators' performances drop significantly in almost all situations when they attempt to generalize their OPE results to the out-sample or future data. The result suggests that the current OPE methods may fail to evaluate the performance of a new policy in the future environment, as they implicitly assume that the data generating distribution does not change over time. Moreover, this kind of realistic out-of-distribution generalization check of OPE cannot be conducted with synthetic or multi-class classification datasets. We thus expect that the Open Bandit Dataset promotes the future research about the robustness of OPE methods to distributional changes.

Table 5: Comparison of OPE performance with different reward estimators (ALL Campaign/Bernoulli TS → Random/in-sample)

OPE Estimators	LR	RF
DM	0.34522	0.30391
DR	0.06446	0.05775
SNDR	0.04938	0.04658
Switch-DR ($\tau = 100$)	0.05938	0.05499
DRos ($\lambda = 100$)	0.21550	0.19111
MRDR	0.02511	0.03000

Note: The averaged relative-estimation errors over 30 different bootstrapped iterations are reported. LR stands for logistic regression and RF stands for random forest. The results on the other campaigns are in Appendix A.

Performance changes across different settings. Finally, we compare the estimation accuracy of each estimator under different experimental conditions and with different hyperparameters. We observe in Table 4 that the estimators' performance can change significantly depending on the experimental conditions. In particular, we tested several values for the hyperparameter τ of Switch-DR. We observe that its estimation performance largely depends on the choice of τ. It is obvious that Switch-DR is significantly better with large values of τ on our data. We also investigate the effect of the choice of the machine learning method to construct $\hat{q}$ in Table 5. Specifically, we additionally test the estimators' performance when random forest is used. The table shows that using random forest to construct $\hat{q}$ provides a more accurate OPE on our dataset. These observations suggest that practitioners have to choose an appropriate OPE estimator or tune estimators' hyperparameters carefully for their specific application. It is thus necessary to develop a reliable method to choose and tune OPE estimators in a data-driven manner. Specifically, in many cases, we have to tune estimators' hyperparameters, including the reward estimator, without the ground-truth policy value of the evaluation policy.

6 CONCLUSION AND FUTURE WORK

To enable realistic and reproducible evaluation of off-policy evaluation, we publicized the Open Bandit Dataset–a set of benchmark logged bandit datasets collected on a large-scale fashion e-commerce platform. The dataset comes with the Open Bandit Pipeline, Python software that makes it easy to evaluate and compare different OPE estimators. We expect them to facilitate understanding of the empirical properties of OPE techniques and address experimental inconsistencies in the literature. In addition to building the data and pipeline, we performed extensive benchmark experiments on OPE. Our experiments highlight that the current OPE methods are inaccurate for estimating the out-of-distribution performance of a new policy. It is also evident that it is necessary to develop a data-driven method to select an appropriate estimator for each given environment.

As future work, we aim to constantly improve the Open Bandit Dataset and Pipeline to include more data and functions. For example, we will add additional log data generated by contextual policies on the platform (whereas the current open data contain only log data generated by context-free policies). Moreover, we assume that the reward of an item at a position does not depend on other simultaneously presented items. This assumption might not hold, as an item's attractiveness can have a significant effect on the expected reward of another item in the same recommendation list [19]. To address more realistic situations, we are implementing some OPE estimators for the slate action setting [22, 32] in the pipeline. Comparing the standard OPE estimators (such as those in Section 2.3) and those for the slate action setting is an interesting future research direction.

REFERENCES

[1] Alina Beygelzimer and John Langford. 2009. The offset tree for learning with partial labels. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. 129–138.
[2] Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, Denis X Charles, D Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. 2013. Counterfactual Reasoning and Learning Systems: The Example of Computational Advertising. *Journal of Machine Learning Research* 14, 1 (2013), 3207–3260.
[3] David Cortes. 2018. Adapting multi-armed bandits policies to contextual bandits scenarios. *arXiv preprint arXiv:1811.04383* (2018).
[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 248–255.
[5] Miroslav Dudík, Dumitru Erhan, John Langford, and Lihong Li. 2014. Doubly Robust Policy Evaluation and Optimization. *Statist. Sci.* 29 (2014), 485–511.
[6] Vijay Prakash Dwivedi, Chaitanya K Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. 2020. Benchmarking graph neural networks. *arXiv preprint arXiv:2003.00982* (2020).

[7] Mehrdad Farajtabar, Yinlam Chow, and Mohammad Ghavamzadeh. 2018. More Robust Doubly Robust Off-policy Evaluation. In *Proceedings of the 35th International Conference on Machine Learning*. 1447–1456.

[8] Alexandre Gilotte, Clément Calauzènes, Thomas Nedelec, Alexandre Abraham, and Simon Dollé. 2018. Offline a/b testing for recommender systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 198–206.

[9] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 580–587.

[10] Alois Gruson, Praveen Chandar, Christophe Charbuillet, James McInerney, Samantha Hansen, Damien Tardieu, and Ben Carterette. 2019. Offline Evaluation to Make Decisions About Playlist Recommendation Algorithms. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 420–428.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[12] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open Graph Benchmark: Datasets for Machine Learning on Graphs. *arXiv preprint arXiv:2005.00687* (2020).

[13] Alex Irpan, Kanishka Rao, Konstantinos Bousmalis, Chris Harris, Julian Ibarz, and Sergey Levine. 2019. Off-Policy Evaluation via Off-Policy Classification. In *Advances in Neural Information Processing Systems*.

[14] Nan Jiang and Lihong Li. 2016. Doubly Robust Off-policy Value Evaluation for Reinforcement Learning. In *Proceedings of the 33rd International Conference on Machine Learning*. 652–661.

[15] Nathan Kallus and Masatoshi Uehara. 2019. Intrinsically efficient, stable, and bounded off-policy evaluation for reinforcement learning. In *Advances in Neural Information Processing Systems*.

[16] Damien Lefortier, Adith Swaminathan, Xiaotao Gu, Thorsten Joachims, and Maarten de Rijke. 2016. Large-scale validation of counterfactual learning methods: A test-bed. *arXiv preprint arXiv:1612.00367* (2016).

[17] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A Contextual-bandit Approach to Personalized News Article Recommendation, In Proceedings of the 19th International Conference on World Wide Web. *WWW*, 661–670.

[18] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. 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. *WSDM*, 297–306.

[19] Shuai Li, Yasin Abbasi-Yadkori, Branislav Kveton, S Muthukrishnan, Vishwa Vinay, and Zheng Wen. 2018. Offline evaluation of ranking policies with click models. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1685–1694.

[20] Yao Liu, Omer Gottesman, Aniruddh Raghu, Matthieu Komorowski, Aldo A Faisal, Finale Doshi-Velez, and Emma Brunskill. 2018. Representation balancing mdps for off-policy policy evaluation. In *Advances in Neural Information Processing Systems*. 2644–2653.

[21] Travis Mandel, Yun-En Liu, Sergey Levine, Emma Brunskill, and Zoran Popovic. 2014. Offline policy evaluation across representations with applications to educational games. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. 1077–1084.

[22] James McInerney, Brian Brost, Praveen Chandar, Rishabh Mehrotra, and Benjamin Carterette. 2020. Counterfactual Evaluation of Slate Recommendations with Sequential Reward Interactions. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1779–1788.

[23] Susan A Murphy, Mark J van der Laan, James M Robins, and Conduct Problems Prevention Research Group. 2001. Marginal mean models for dynamic regimes. *J. Amer. Statist. Assoc.* 96, 456 (2001), 1410–1423.

[24] Yusuke Narita, Shota Yasui, and Kohei Yata. 2019. Efficient counterfactual learning from bandit feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 4634–4641.

[25] Yusuke Narita, Shota Yasui, and Kohei Yata. 2020. Off-policy Bandit and Reinforcement Learning. *arXiv preprint arXiv:2002.08536* (2020).

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

[27] David Rohde, Stephen Bonner, Travis Dunlop, Flavian Vasile, and Alexandros Karatzoglou. 2018. RecoGym: A Reinforcement Learning Environment for the problem of Product Recommendation in Online Advertising. *arXiv preprint arXiv:1808.00720* (2018).

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

[29] Yi Su, Maria Dimakopoulou, Akshay Krishnamurthy, and Miroslav Dudík. 2020. Doubly robust off-policy evaluation with shrinkage. In *Proceedings of the 37th International Conference on Machine Learning*.

[30] Adith Swaminathan and Thorsten Joachims. 2015. Batch Learning from Logged Bandit Feedback through Counterfactual Risk Minimization. *Journal of Machine Learning Research* 16 (2015), 1731–1755.

[31] Adith Swaminathan and Thorsten Joachims. 2015. The Self-normalized Estimator for Counterfactual Learning. In *Advances in Neural Information Processing Systems*. 3231–3239.

[32] Adith Swaminathan, Akshay Krishnamurthy, Alekh Agarwal, Miro Dudik, John Langford, Damien Jose, and Imed Zitouni. 2017. Off-policy Evaluation for Slate Recommendation, In Advances in Neural Information Processing Systems. *NIPS*, 3635–3645.

[33] Philip Thomas and Emma Brunskill. 2016. Data-efficient Off-policy Policy Evaluation for Reinforcement Learning, In Proceedings of the 33rd International Conference on Machine Learning. *ICML*, 2139–2148.

[34] Philip Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. 2015. High Confidence Policy Improvement, In Proceedings of the 32th International Conference on Machine Learning. *ICML*, 2380–2388.

[35] Philip S Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. 2015. High-Confidence Off-Policy Evaluation. *AAAI* (2015), 3000–3006.

[36] William R Thompson. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 25, 3/4 (1933), 285–294.

[37] Masatoshi Uehara, Masahiro Kato, and Shota Yasui. 2020. Off-Policy Evaluation and Learning for External Validity under a Covariate Shift. *Advances in Neural Information Processing Systems* (2020).

[38] Nikos Vlassis, Aurelien Bibaut, Maria Dimakopoulou, and Tony Jebara. 2019. On the Design of Estimators for Bandit Off-Policy Evaluation. In *International Conference on Machine Learning*. 6468–6476.

[39] Cameron Voloshin, Hoang M Le, Nan Jiang, and Yisong Yue. 2019. Empirical Study of Off-Policy Policy Evaluation for Reinforcement Learning. *arXiv preprint arXiv:1911.06854* (2019).

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

[41] Tengyang Xie, Yifei Ma, and Yu-Xiang Wang. 2019. Towards Optimal Off-Policy Evaluation for Reinforcement Learning with Marginalized Importance Sampling. In *Advances in Neural Information Processing Systems*. 9665–9675.

Table 6: Estimation performances of reward estimators

			Random → Bernoulli TS		Bernoulli TS → Random	
Models	**Campaigns**	**Metrics**	*in*-sample	*out*-sample	*in*-sample	*out*-sample
LR	ALL	AUC	0.56380 ±0.00579	0.53103 ±0.00696	0.57139 ±0.00176	0.51900 ±0.00706
		RCE	0.00217 ±0.00133	-0.00853 ±0.00272	0.00588 ±0.00026	-0.01162 ±0.00271
	Men's	AUC	0.58068 ±0.00751	0.54411 ±0.01025	0.57569 ±0.00264	0.56528 ±0.00272
		RCE	-0.00019 ±0.00316	-0.01767 ±0.00600	0.00588 ±0.00038	0.00329 ±0.00084
	Women's	AUC	0.55245 ±0.00588	0.51900 ±0.00706	0.54642 ±0.00157	0.53387 ±0.00249
		RCE	-0.00100 ±0.00196	-0.01162 ±0.00271	0.00307 ±0.00018	0.00140 ±0.00031
RF	ALL	AUC	0.65427 ±0.00699	0.58240 ±0.00881	0.59691 ±0.00214	0.57850 ±0.00268
		RCE	0.02168 ±0.00146	0.00546 ±0.00162	0.00889 ±0.00019	0.00702 ±0.00025
	Men's	AUC	0.64695 ±0.00794	0.55191 ±0.01247	0.59077 ±0.00193	0.56889 ±0.00184
		RCE	0.02122 ±0.00179	-0.00495 ±0.00337	0.00857 ±0.00028	0.00480 ±0.00035
	Women's	AUC	0.62770 ±0.00741	0.53735 ±0.00740	0.56364 ±0.00162	0.54376 ±0.00233
		RCE	0.01574 ±0.00121	-0.00264 ±0.00150	0.00401 ±0.00013	0.00224 ±0.00022

Note: This table presents the area under the ROC curve (AUC) and relative cross-entropy (RCE) of the reward estimator on a validation set for each campaign. The averaged results and their unbiased standard deviations estimated using 30 different bootstrapped samples are reported. LR stands for logistic regression and RF stands for random forest. $\pi_b \rightarrow \pi_e$ represents the OPE situation where the estimators aim to estimate the policy value of π_e using logged bandit data collected by π_b, meaning that $\hat{q}$ is trained on data collected by π_b.

A ADDITIONAL EXPERIMENTAL SETTINGS AND RESULTS

Table 6 reports the estimation accuracy of logistic regression and random forest as a reward estimator. Note that, as their hyperparameters, we use $C = 1000$ for logistic regression and n_estimators = 100, max_depth = 5, min_samples_leaf = 10 for random forest. In addition, we use action-related feature vectors to represent action variables to train reward estimators. Table 7- 8 show the results of the benchmark experiments on Men's and Women's campaigns. All the experiments were conducted on MacBook Pro (2.4 GHz Intel Core i9, 64 GB), and it takes about 1 week to complete the benchmark on the ALL campaign when we use Bernoulli TS as π_b and random forest as a reward estimator (, which takes the longest time among all possible experimental settings).

A.1 Estimation Performance of Reward Estimators

We evaluate the performance of the reward estimators by using the following evaluation metrics.

Relative Cross Entropy (RCE).. RCE is defined as the improvement of an estimation performance relative to the naive estimation, which uses the mean CTR for every prediction. We calculate this metric using a size n of validation samples $\{(x_t, y_t)\}_{t=1}^{n}$ as:

$$RCE\ (\hat{q}) := 1 - \frac{\sum_{t=1}^{n} y_t \log(\hat{q}(x_t)) + (1-y_t)\log(1-\hat{q}(x_t))}{\sum_{t=1}^{n} y_t \log(\hat{q}_{\mathrm{naive}}) + (1-y_t)\log(1-\hat{q}_{\mathrm{naive}})}$$

where $\hat{q}_{\mathrm{naive}} := n^{-1}\sum_{t=1}^{n} y_t$ is the naive estimation. A larger value of RCE means better performance of a predictor.

Area Under the ROC Curve (AUC).. AUC is defined as the probability that positive samples are ranked higher than negative items by a classifier under consideration.

$$AUC\ (\hat{q}) := \frac{1}{n^{\mathrm{pos}} n^{\mathrm{neg}}} \sum_{t=1}^{n^{\mathrm{pos}}} \sum_{j=1}^{n^{\mathrm{neg}}} \mathbb{I}\{\hat{q}(x_t^{\mathrm{pos}}) > \hat{q}(x_j^{\mathrm{neg}})\}$$

where $\mathbb{I}\{\cdot\}$ is the indicator function. $\{x_t^{\mathrm{pos}}\}_{t=1}^{n^{\mathrm{pos}}}$ and $\{x_j^{\mathrm{neg}}\}_{j=1}^{n^{\mathrm{neg}}}$ are sets of positive and negative samples in the validation set, respectively. A larger value of AUC means better performance of a predictor.

A.2 Estimating Mean and Standard Deviation of Performance Measures

To estimate means and standard deviations of relative-EE in the benchmark experiment, we first construct an empirical cumulative distribution function $\hat{F}_{\mathcal{D}_{\mathrm{ev}}}$ of the evaluation set of the logged bandit feedback ($\mathcal{D}_{\mathrm{ev}}$). Then, we draw bootstrap samples $\mathcal{D}_{\mathrm{ev}}^{(1,*)}, \ldots, \mathcal{D}_{\mathrm{ev}}^{(B,*)}$ from $\hat{F}_{\mathcal{D}_{\mathrm{ev}}}$ and compute the relative-EE of a given estimator $\hat{V}$ with each set. Finally, we estimate the mean and its standard deviation (Std) of the $\hat{V}$'s relative-EE by

$$\mathrm{Mean}(\textit{relative-EE}(\hat{V}; \mathcal{D}_{\mathrm{ev}})) := \frac{1}{B}\sum_{b=1}^{B} \textit{relative-EE}(\hat{V}; \mathcal{D}_{\mathrm{ev}}^{(b,*)}),$$

$$\mathrm{Std}(\textit{relative-EE}(\hat{V}; \mathcal{D}_{\mathrm{ev}}))$$
$$:= \sqrt{\frac{1}{B-1}\sum_{b=1}^{B}\left(\textit{relative-EE}(\hat{V}; \mathcal{D}_{\mathrm{ev}}^{(b,*)}) - \mathrm{Mean}(\textit{relative-EE}(\hat{V}; \mathcal{D}_{\mathrm{ev}}))\right)^2},$$

where we use $B = 30$ for all experiments.

Table 7: Comparison of relative-estimation errors of OPE estimators (Men's Campaign)

OPE Estimators	Random → Bernoulli TS		Bernoulli TS → Random	
	in-sample	*out*-sample	*in*-sample	*out*-sample
DM	**0.24311** ±0.03128	**0.29088** ±0.03440	**0.24332** ±0.01661	0.12275 ±0.01791
IPW	0.11060 ±0.04173	0.19521 ±0.04533	**0.02908** ±0.02413	0.08407 ±0.02471
SNIPW	**0.09343** ±0.04170	**0.17499** ±0.04611	0.07301 ±0.03406	0.19564 ±0.04117
DR	0.09727 ±0.04091	0.18073 ±0.04519	0.14994 ±0.05710	**0.28765** ±0.07703
SNDR	0.09447 ±0.04139	0.17794 ±0.04629	0.11218 ±0.04287	0.23546 ±0.05585
Switch-DR ($\tau = 5$)	0.23820 ±0.01950	0.27584 ±0.02035	0.17478 ±0.01145	0.06573 ±0.01204
Switch-DR ($\tau = 10$)	0.16504 ±0.02665	0.20912 ±0.03873	0.17381 ±0.01215	0.05575 ±0.01489
Switch-DR ($\tau = 50$)	0.22290 ±0.04091	0.18073 ±0.04519	0.13706 ±0.02529	0.02666 ±0.01919
Switch-DR ($\tau = 100$)	0.09727 ±0.04091	0.18073 ±0.04519	0.11114 ±0.02864	**0.02139** ±0.01596
Switch-DR ($\tau = 500$)	0.09727 ±0.04091	0.18073 ±0.04519	0.05424 ±0.03006	0.05825 ±0.02440
Switch-DR ($\tau = 1000$)	0.09727 ±0.04091	0.18073 ±0.04519	0.05199 ±0.02997	0.06140 ±0.02461
DRos ($\lambda = 5$)	0.22303 ±0.02110	0.26581 ±0.02070	0.21428 ±0.01219	0.09445 ±0.01382
DRos ($\lambda = 10$)	0.21329 ±0.02029	0.25640 ±0.02045	0.20239 ±0.01157	0.08366 ±0.01301
DRos ($\lambda = 50$)	0.17230 ±0.02335	0.22410 ±0.02548	0.17536 ±0.01109	0.05879 ±0.01254
DRos ($\lambda = 100$)	0.15069 ±0.02707	0.20992 ±0.02990	0.16542 ±0.01183	0.04906 ±0.01350
DRos ($\lambda = 500$)	0.11407 ±0.03594	0.18917 ±0.03980	0.14470 ±0.01597	0.02957 ±0.01567
DRos ($\lambda = 1000$)	0.10636 ±0.03816	0.18523 ±0.04222	0.13638 ±0.01835	**0.02306** ±0.01598
MRDR	**0.09173** ±0.04145	**0.17754** ±0.04673	**0.04385** ±0.03299	0.07649 ±0.02900

Note: The averaged relative-estimation errors and their unbiased standard deviations estimated over 30 different bootstrapped iterations are reported. $\pi_b \rightarrow \pi_e$ represents the OPE situation where the estimators aim to estimate the policy value of π_e using logged bandit data collected by π_b. The **red** and **green** fonts represent the best and the second best estimators. The **blue** fonts represent the worst estimator for each setting.

Table 8: Comparison of relative-estimation errors of OPE estimators (Women's Campaign)

OPE Estimators	Random → Bernoulli TS		Bernoulli TS → Random	
	in-sample	*out*-sample	*in*-sample	*out*-sample
DM	**0.21719** ±0.03274	**0.25428** ±0.02940	**0.31762** ±0.01011	**0.21892** ±0.01346
IPW	0.02827 ±0.02418	**0.03957** ±0.02779	0.03992 ±0.01997	0.09295 ±0.02527
SNIPW	**0.02827** ±0.02383	0.04221 ±0.02976	0.07564 ±0.02578	0.11461 ±0.02646
DR	0.02835 ±0.02420	**0.04200** ±0.02952	0.09244 ±0.03063	0.12652 ±0.02904
SNDR	0.02833 ±0.02415	0.04280 ±0.02973	0.07659 ±0.02582	0.11809 ±0.02661
Switch-DR ($\tau = 5$)	0.15483 ±0.02355	0.20191 ±0.02660	0.24993 ±0.00614	0.16243 ±0.00919
Switch-DR ($\tau = 10$)	0.05966 ±0.03183	0.10547 ±0.03843	0.21151 ±0.00827	0.12292 ±0.00950
Switch-DR ($\tau = 50$)	0.02835 ±0.02420	**0.04200** ±0.02952	0.12182 ±0.01416	**0.02639** ±0.01515
Switch-DR ($\tau = 100$)	0.02835 ±0.02420	**0.04200** ±0.02952	0.08990 ±0.01381	**0.01129** ±0.00921
Switch-DR ($\tau = 500$)	0.02835 ±0.02420	**0.04200** ±0.02952	**0.01838** ±0.01793	0.05898 ±0.02007
Switch-DR ($\tau = 1000$)	0.02835 ±0.02420	**0.04200** ±0.02952	**0.01644** ±0.01352	0.07120 ±0.02171
DRos ($\lambda = 5$)	0.17694 ±0.02694	0.21672 ±0.02729	0.28591 ±0.00635	0.19300 ±0.00982
DRos ($\lambda = 10$)	0.15834 ±0.02583	0.19949 ±0.02692	0.27144 ±0.00606	0.17989 ±0.00930
DRos ($\lambda = 50$)	0.09811 ±0.02576	0.13920 ±0.02857	0.23040 ±0.00625	0.14109 ±0.00843
DRos ($\lambda = 100$)	0.07023 ±0.02786	0.10826 ±0.03132	0.21119 ±0.00679	0.12227 ±0.00852
DRos ($\lambda = 500$)	0.03415 ±0.02303	0.05588 ±0.03474	0.16675 ±0.00864	0.07698 ±0.00994
DRos ($\lambda = 1000$)	0.02948 ±0.02380	0.04770 ±0.03301	0.14829 ±0.00957	0.05800 ±0.01082
MRDR	**0.02809** ±0.02388	0.04354 ±0.03060	0.02800 ±0.01758	0.08990 ±0.01898

Note: The averaged relative-estimation errors and their unbiased standard deviations estimated over 30 different bootstrapped iterations are reported. $\pi_b \rightarrow \pi_e$ represents the OPE situation where the estimators aim to estimate the policy value of π_e using logged bandit data collected by π_b. The **red** and **green** fonts represent the best and the second best estimators. The **blue** fonts represent the worst estimator for each setting.