Off-Policy Evaluation of Candidate Generators in Two-Stage Recommender Systems

Peiyao Wang
Amazon.com
Seattle, Washington, USA
peiyaow@amazon.com

Zhan Shi
Amazon.com
Vancouver, British Columbia, Canada
shizhan@amazon.com

Amina Shabbeer
Amazon.com
San Francisco, California, USA
ashabb@amazon.com

Ben London
Amazon.com
Seattle, Washington, USA
blondon@amazon.com

Abstract

We study offline evaluation of two-stage recommender systems, focusing on the first stage, candidate generation. Traditionally, candidate generators have been evaluated in terms of standard information retrieval metrics, using curated or heuristically labeled data, which does not always reflect their true impact to user experience or business metrics. We instead take a holistic view, measuring their effectiveness with respect to the downstream recommendation task, using data logged from past user interactions with the system. Using the contextual bandit formalism, we frame this evaluation task as *off-policy evaluation* (OPE) with a new action set induced by a new candidate generator. To the best of our knowledge, ours is the first study to examine evaluation of candidate generators through the lens of OPE. We propose two importance-weighting methods to measure the impact of a new candidate generator using data collected from a downstream task. We analyze the asymptotic properties of these methods and derive expressions for their respective biases and variances. This analysis illuminates a procedure to optimize the estimators so as to reduce bias. Finally, we present empirical results that demonstrate the estimators' efficacy on synthetic and benchmark data. We find that our proposed methods achieve lower bias with comparable or reduced variance relative to baseline approaches that do not account for the new action set.

CCS Concepts

• **Information systems → Evaluation of retrieval results**.

Keywords

Recommendation, Off-Policy Evaluation, Candidate Generation

ACM Reference Format:
Peiyao Wang, Zhan Shi, Amina Shabbeer, and Ben London. 2025. Off-Policy Evaluation of Candidate Generators in Two-Stage Recommender Systems. In *Proceedings of the Nineteenth ACM Conference on Recommender Systems (RecSys '25), September 22–26, 2025, Prague, Czech Republic.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3705328.3748057

1 Introduction

Modern recommender systems often employ a *two-stage* architecture, comprised of a *candidate generator* and a *policy* [13]. The role of the candidate generator is to winnow an extremely large catalog of items down to a much smaller set of potentially relevant candidates, which are then given to the policy to select the final recommendation(s). This division of work is often motivated by efficiency and scaling, since recommending content from a large catalog poses computational challenges. Accordingly, the candidate generator prioritizes efficiency and recall, while the policy focuses on precision. The two-stage architecture supports a wide range of applications, including online advertising [2], video recommendations [4, 5, 49], news [27, 47] and social media [7].

The candidate generator can have a significant impact on overall system performance (such as user engagement), since it defines the *action space* over which the downstream policy operates. It is therefore critical to evaluate any changes to the candidate generator *before* deploying said changes to a production environment. Reliable offline evaluation is essential, not only to guide model training and selection, but also to justify online evaluation of new models via A/B testing.

Traditionally, candidate generators have been evaluated offline using an information retrieval (IR) methodology, wherein metrics such as precision, recall and NDCG are evaluated against certain benchmark, "ground truth" datasets—which are often heuristically derived from user interactions or hand-labelled data [50]. While intuitive and easy to implement, this approach suffers from several well-known limitations: (i) IR metrics may not reflect business objectives; (ii) this evaluation ignores the downstream policy that directly influences user outcomes; and (iii) using interaction data generated by previously deployed models to validate new models can create a feedback loop, introducing bias and blind spots.

Meanwhile, the contextual bandit framework has emerged as a powerful formalism for recommender systems; and through this view, *off-policy evaluation* (OPE) provides a principled methodology for evaluating recommender systems offline. OPE involves estimating the performance of a new (*target*) policy using data collected under a different (*logging*) policy (which is typically a randomized variant of the current production system). These methods usually assume that both policies operate in the same action space, and have *common support* on this set. Unfortunately, these assumptions break down with a two-stage architecture—specifically, when estimating the impact of changes to the candidate generation stage. While most OPE research has focused on evaluating the final ranking or decision policy, no prior work has studied the effect of upstream components, such as candidate generators.

Our work fills this gap and addresses the limitations of traditional IR evaluation. We frame the evaluation problem as OPE of the downstream recommendation policy under a change to its action set, which is induced by a candidate generator that differs from the one used to log the data. Unlike the IR approach, OPE lets us directly measure impact to the downstream recommendation task, in terms of business metrics, while accounting for selection bias. To estimate expected performance (a.k.a. *reward*), we propose two importance-weighting methods, which we refer to as the *intersection method* and the *proxy method*, respectively. The intersection method restricts the target policy to the intersection of action sets, which provides a simple and intuitive starting point—as well as a surprisingly strong baseline. The more sophisticated proxy method is based on the idea that actions can be grouped into equivalence classes, within which old actions can serve as proxies for new actions. To this end, we leverage *marginalized propensities* over the equivalence classes to effectively share rewards observed on old actions with the other actions in their respective equivalence classes. This reduces bias from unseen new actions by incorporating their probability mass into observed existing actions from the same equivalence classes. We pair each of these importance weighting methods with reward modeling to further reduce bias on unseen actions, and reduce overall variance.

The contributions of this paper are as follows:

- We formulate offline evaluation of candidate generators as OPE with a changed action set.
- We propose two importance-weighting methods, intersection and proxy, to address this scenario and derive corresponding estimators for each.
- We analyze the bias and variance of the proxy method, which provides theoretical insights into their inherent trade-off, as well as guidance for how to optimize the parameters of the proxy-based estimators.
- We present empirical results on both synthetic and real-world data that demonstrate the superiority of our methods in evaluating a new candidate generator.

1.1 Related Work

Off-policy evaluation (and learning) has been studied extensively in the literature on contextual bandits [1, 6, 11, 20–23, 44, 46], reinforcement learning [8, 15, 26, 42, 43] and counterfactual learning [3, 16–19, 33, 36, 38–40]. Many of these works analyzed the bias, variance and uniform convergence of reward estimators, but they focus on a setting in which the logging and target policies have *common support* on the action set. While this assumption is reasonable when the action set is fixed, in our setting, the changing action set makes common support impossible to satisfy.

Lack of common support—or *support deficiency*, as it is sometimes called—often arises in off-policy (not necessarily offline) learning, when the logging (a.k.a. behavioral) policy does not sufficiently explore actions available to the target policy. To compensate, one can constrain the target policy to stay close to the behavioral policy [25, 29, 34], or constrain the target action space to that of the logging policy [29]. Alternatively, one can simply impute reward for actions not taken by the behavioral policy [10, 29]. While these methods make off-policy learning "safe," their impact on off-policy evaluation has not been explored. London and Joachims [24] analyzed an evaluation setting very similar to ours—in which the action set changes, causing support deficiency—and showed that the bias of certain model-based estimators depends crucially on the reward model's ability to predict rewards for new actions. They did not, however, propose a new estimator. We build on London and Joachims's [24] analysis and propose new estimators. Felicioni et al. [9] investigated evaluation with support deficiency and proposed an estimator that is similar to our proxy method, but does not incorporate reward modeling. Our work can be viewed as extending theirs by incorporating reward modeling—which reduces bias on new actions as well as variance.

One particular setting in which common support becomes challenging is when the action set is very large—which often arises in applications such as slate optimization or ranking, in which the action set combinatorial in nature. Even if common support is achieved, the probability of selecting any given action becomes very small, resulting in large variance in importance-weighting estimators. Techniques like weight clipping [14] and self-normalization [40] are common remedies to the variance issue, but they come at a cost of introducing bias. Alternatively, researchers often introduce simplifying assumptions about the environment's reward function [37, 41, 45]. Recent work [28, 31, 32, 35] has adopted an approach based on action grouping, leveraging implicit structure in the action space (e.g., latent factors, clusters, or representative "main" actions). The corresponding importance-weighting estimators marginalize the propensities over action groupings, thereby reducing variance. Our approach is similar in that we also impose structure on the action space—in terms of equivalence classes—and marginalize propensities over these equivalence classes. However, while the primary focus of prior work has been on variance reduction under fixed, large action sets, we tackle a fundamentally different problem with two-stage recommender systems. The fact that the action set changes between logging and evaluation creates challenges that go beyond large action spaces, and offers a novel perspective on the marginalization approach. It is also worth noting that our theoretical analysis is arguably simpler and more intuitive than previous analyses of marginalized importance-weighting estimators, so our proof techniques may shed new light on prior work.

2 Preliminaries

A deployed two-stage recommender system can be formalized in the following *contextual bandit* framework, involving a sequence of i.i.d. interactions with users. In each interaction, a *context*, x—which encapsulates the current user's state and any other relevant variables—is sampled from the environment. Given the context, the candidate generator, $\mathcal{A}$, determines a set of available *actions* (i.e., items), $\mathcal{A}(x)$, from which the policy, π, can select. As its selection may be stochastic, we can think of the policy as inducing a conditional distribution on $\mathcal{A}(x)$. Upon sampling an action, $a \sim \pi(\cdot \mid \mathcal{A}, x)$, the environment generates a stochastic *reward*, such as engagement with the recommended item. We can equivalently think of the reward as a function of the context and action, $r(x, a)$, that is sampled, with each context, from the environment. Our metric of interest is the *expected reward* for the deployed two-stage recommender:

$$R(\mathcal{A}, \pi) \triangleq \mathop{\mathbb{E}}_{(x,r)\sim\mathbb{D}} \mathop{\mathbb{E}}_{a\sim\pi(\,\cdot\,|\,\mathcal{A},x)} \big[r(x, a)\big]. \tag{1}$$

2.1 Problem Setup

Suppose we have previously deployed a candidate generator, $\mathcal{A}_0$, and a policy, π, which we will henceforth refer to as the *target policy* (or *production policy*). Additionally, suppose we have deployed another policy, π_0, for the purpose of randomized data collection. We will refer to π_0 as the *logging policy.* In many applications, the logging policy is distinct from the target policy (which serves the majority of users) because we usually wish to limit randomness in the user experience. The logging policy may be of any parametric or nonparametric form; all we require from π_0 is that it is randomized and has *full support* on $\mathcal{A}_0$—meaning, for any x and $a \in \mathcal{A}_0(x)$, we have $\pi_0(a \,|\, \mathcal{A}_0, x) > 0$. We let the logging policy run for n rounds of interaction and collect a dataset,

$$S \triangleq \big(x_i, a_i, \pi_0(a_i \,|\, \mathcal{A}_0, x_i), r(x_i, a_i)\big)_{i=1}^n,$$

where $(x_i, r_i) \sim \mathbb{D}$ and $a_i \sim \pi_0(\,\cdot\,|\,\mathcal{A}_0, x_i)$. For ease of exposition, we may write the distribution of S as $(\mathbb{D} \times \pi_0)^n$.

Now, suppose we want to test how a different candidate generator, $\mathcal{A}_1$, will perform with π. We assume that $\mathcal{A}_1$ generates actions that are not generated by $\mathcal{A}_0$ for certain contexts; that is, more formally, $\exists x,\ \mathcal{A}_1(x) \setminus \mathcal{A}_0(x) \neq \emptyset$. For actions generated by $\mathcal{A}_0$ that are not generated by $\mathcal{A}_1$, we can equivalently say that they have zero probability under π. To reduce clutter, we will use the notation $\mathcal{A}_{0\cup 1}(x) \triangleq \mathcal{A}_0(x) \cup \mathcal{A}_1(x)$ to denote the union of old and new actions, $\mathcal{A}_{0\cap 1}(x) \triangleq \mathcal{A}_0(x) \cap \mathcal{A}_1(x)$ to denote the intersection of old and new actions, and $\mathcal{A}_{1\setminus 0}(x) \triangleq \mathcal{A}_1(x) \setminus \mathcal{A}_0(x)$ to denote the *strictly new* actions.

Recalling the definition of expected reward (Equation 1), our goal is to estimate $R(\mathcal{A}_1, \pi)$ using the logged data, S. Formally, since the data was collected by a policy that is different from the target policy, this is a problem of *off-policy estimation.* There is much literature in this subject, and many of the existing methods use some form of *importance weighting* to obtain unbiased estimates [12]. However, it is usually assumed that the action set available to the target policy will be the same as the one used by the logging policy—which is not the case here.[1] This creates a fundamental challenge: we cannot observe reward for any strictly new action, $a \in \mathcal{A}_{1\setminus 0}(x)$, during data collection. Without additional assumptions, any importance-weighted reward estimate will be biased.

Throughout this work, we will use

$$\mathrm{Bias}(\hat{R}, \mathcal{A}, \pi) \triangleq \mathop{\mathbb{E}}_{S\sim(\mathbb{D}\times\pi_0)^n} \big[\hat{R}(\mathcal{A}, \pi, S)\big] - R(\mathcal{A}, \pi),$$

to denote the bias of a reward estimator, $\hat{R}$, given a candidate generator, $\mathcal{A}$, and target policy, π.

[1] Another way to view this setting is that there was only ever candidate generator $\mathcal{A}_1$, which subsumes $\mathcal{A}_0$, and the logging policy simply had *deficient support* [9, 29].

2.2 Estimating Expected Reward

Before introducing our proposed methods, we review some standard existing techniques. Arguably, the most basic reward estimator is the *inverse propensity scoring* (IPS) estimator [12],

$$\hat{R}_{\text{IPS}}(\mathcal{A}_1, \pi, S) \triangleq \frac{1}{n}\sum_{i=1}^n \frac{\pi(a_i \,|\, \mathcal{A}_1, x_i)}{\pi_0(a_i \,|\, \mathcal{A}_0, x_i)}\, r(x_i, a_i), \tag{2}$$

which re-weights the observed rewards by an importance weight involving the ratio of action probabilities under the target and logging policies. Normally, when $\mathcal{A}_0$ and $\mathcal{A}_1$ coincide and the logging policy has full support, this importance weight corrects the sampling distribution induced by the logging policy to that of the target policy, thereby yielding an unbiased estimate of expected reward. However, here we have specifically modified the notation to account for the fact that the target policy will use a different action set, as dictated by the new candidate generator, $\mathcal{A}_1$. It is straightforward to show that this causes bias equal to

$$\mathrm{Bias}(\hat{R}_{\text{IPS}}, \mathcal{A}_1, \pi) = \mathop{\mathbb{E}}_{(x,r)\sim\mathbb{D}} \mathop{\mathbb{E}}_{a\sim\pi(\,\cdot\,|\,\mathcal{A}_1,x)} \big[-\mathbb{1}\{a \in \mathcal{A}_{1\setminus 0}(x)\}\, r(x, a)\big].$$

Essentially, the estimator can only account for old actions, but misses the rewards for strictly new actions.

Clearly, importance weighting alone cannot produce useful reward estimates, since it cannot account for the impact of new actions. To compensate, we could adopt a model-based approach using the so-called *direct method* (DM) estimator,

$$\hat{R}_{\text{DM}}(\mathcal{A}_1, \pi, S) \triangleq \frac{1}{n}\sum_{i=1}^n \mathop{\mathbb{E}}_{a\sim\pi(\,\cdot\,|\,\mathcal{A}_1,x_i)} \big[h(x_i, a)\big].$$

This estimator employs a reward model, h, to predict the reward for each available action, which is then weighted by the probability of selecting said action under the target policy. It is well known that the bias of this estimator is

$$\mathrm{Bias}(\hat{R}_{\text{DM}}, \mathcal{A}_1, \pi) = \mathop{\mathbb{E}}_{(x,r)\sim\mathbb{D}} \mathop{\mathbb{E}}_{a\sim\pi(\,\cdot\,|\,\mathcal{A}_1,x)} \big[h(x, a) - r(x, a)\big]; \tag{3}$$

that is, it is determined by the prediction error of the reward model under the target policy's action distribution, regardless of whether the actions are old or new.

By combining IPS with DM, we get the classic *doubly-robust* (DR) estimator [6],

$$\begin{aligned}\hat{R}_{\text{DR}}(\mathcal{A}_1, \pi, S) &\triangleq \hat{R}_{\text{DM}}(\mathcal{A}_1, \pi, S) \\ &\quad + \frac{1}{n}\sum_{i=1}^n \frac{\pi(a_i \,|\, \mathcal{A}_1, x_i)}{\pi_0(a_i \,|\, \mathcal{A}_0, x_i)}\, \delta(x_i, a_i), \qquad (4)\\ \text{where } \delta(x, a) &\triangleq r(x, a) - h(x, a). \qquad (5)\end{aligned}$$

While this reduces the bias in our setting even further, it does not completely eliminate it. Indeed, London and Joachims [24] showed that the bias is

$$\begin{aligned}&\mathrm{Bias}(\hat{R}_{\text{DR}}, \mathcal{A}_1, \pi) \\ &= \mathop{\mathbb{E}}_{(x,r)\sim\mathbb{D}} \mathop{\mathbb{E}}_{a\sim\pi(\,\cdot\,|\,\mathcal{A}_1,x)} \big[\mathbb{1}\{a \in \mathcal{A}_{1\setminus 0}(x)\} \big(h(x, a) - r(x, a)\big)\big].\end{aligned}$$

Since DR combines IPS and DM, it makes sense that its bias is a combination of their respective biases. Thanks to importance weighting, the bias is limited to strictly new actions; and thanks to DM, an accurate reward predictor can reduce the bias further.

3 Estimation via Intersection

Before presenting our proposed proxy-based methods, it is instructive to consider an intuitive baseline that addresses a limitation in the standard IPS estimator in our setting. Specifically, the IPS estimator ignores the probability mass that the target policy assigns to strictly new actions, $\mathcal{A}_{1\setminus 0}$. As a result, the more the target policy emphasizes $\mathcal{A}_{1\setminus 0}$, the greater the potential bias in the IPS estimates. While the DR estimator partially addresses this by incorporating reward prediction for $\mathcal{A}_{1\setminus 0}$, further improvements are possible with importance weighting alone.

Intuitively, if the rewards for strictly new actions are similar to those of the "carry-over" new actions—i.e., actions that are not strictly new, at the intersection of $\mathcal{A}_0$ and $\mathcal{A}_1$—then redistributing the probability mass from $\mathcal{A}_{1\setminus 0}$ to $\mathcal{A}_{0\cap 1}$ should allow the importance weighting to capture more of their reward. We can accomplish this by forcing the target policy to focus on these actions at the intersection; effectively, redistributing probability mass via normalization. Note that this constraint would only be applied to the target policy during evaluation, but the deployed policy would have access to all actions from $\mathcal{A}_1$.

We therefore define two new estimators,

$$\hat{R}_{\text{IIPS}}(\mathcal{A}_1, \pi, S) \triangleq \frac{1}{n}\sum_{i=1}^{n} \frac{\pi(a_i \mid \mathcal{A}_{0\cap 1}, x_i)}{\pi_0(a_i \mid \mathcal{A}_0, x_i)}\, r(x_i, a_i), \tag{6}$$

$$\hat{R}_{\text{IDR}}(\mathcal{A}_1, \pi, S) \triangleq \hat{R}_{\text{DM}}(\mathcal{A}_1, \pi, S) + \frac{1}{n}\sum_{i=1}^{n} \frac{\pi(a_i \mid \mathcal{A}_{0\cap 1}, x_i)}{\pi_0(a_i \mid \mathcal{A}_0, x_i)}\, \delta(x_i, a_i), \tag{7}$$

using the definition of δ from Equation 5. We refer to $\hat{R}_{\text{IIPS}}$ as the *intersection inverse propensity scoring* (IIPS) estimator, and $\hat{R}_{\text{IDR}}$ as the *intersection doubly-robust* (IDR) estimator. Our experiments (Section 6) show that IIPS consistently outperforms the standard IPS estimator, and IDR further improves by reducing variance.

4 Estimation via Proxy Actions

Our second class of estimators is based on the intuition that, if each new action is similar to at least one of the old actions, then we should be able to estimate the reward for a new action using old actions as *proxy*. We can rely on importance weighting to estimate reward for the old actions, then predict the difference in reward using a model. This idea is conceptually similar to DR, in that it combines importance weighting with reward prediction. We also draw inspiration from some recently proposed estimators for large or combinatorial action sets in which actions are effectively projected onto a low-dimensional representation (e.g., clusters, groups or "main" actions) and importance weights are marginalized over this representation [28, 31, 32, 35].

Before defining the estimator, we begin with some notation and auxiliary definitions. By convention, we assume that a policy outputs zero probability mass for any action not contained in the given action set (that is, not output by the candidate generator); i.e., $\pi(a \mid \mathcal{A}, x) = 0$ for any action $a \notin \mathcal{A}(x)$.

Recall our guiding intuition, that each new action performs similarly to some representative proxies in the old actions. We therefore define a *proxy operator*, which maps any action (in any context) to similar actions, some of which must come from the old action set.

Definition 1. For a context, $x \in \mathcal{X}$, let $\mathcal{M}(x) \subset \mathcal{P}(\mathcal{A}_{0\cup 1})$ denote a set of *equivalence classes* defined by an *equivalence relation*, $\sim_x$, on the set all actions, $\mathcal{A}_{0\cup 1}$, where $\mathcal{P}$ indicates the power set. We say that an equivalence relation is *valid* if every equivalence class contains at least one old action; i.e., $\forall m \in \mathcal{M}(x)$, $m \cap \mathcal{A}_0(x) \neq \emptyset$.[2] Given an equivalence relation, a *proxy operator*, $\phi(a, x) = \{a' \mid a' \sim_x a\}$, is a contextual mapping from an action to its equivalence class.

To simplify notation, we refer to $m \cap \mathcal{A}$ as a candidate generator induced by equivalence class m, where $(m \cap \mathcal{A})(x) \triangleq m \cap \mathcal{A}(x)$. We also define some auxiliary propensities that *marginalize* over the proxy mappings. Let

$$\bar{\pi}(a \mid \mathcal{A}, x) \triangleq \sum_{a' \in \mathcal{A}(x)} \pi(a' \mid \mathcal{A}, x)\mathbb{1}\{\phi(x, a) = \phi(x, a')\},$$

denote the *marginalized propensities*, where ϕ is the proxy operator.

With the above definitions, we are now ready to define the proxy-based estimators, which we refer to as *proxy IPS* (PIPS) and *proxy DR* (PDR), respectively:

$$\hat{R}_{\text{PIPS}}(\mathcal{A}_1, \pi, S) \triangleq \frac{1}{n}\sum_{i=1}^{n} \frac{\bar{\pi}(a_i \mid \mathcal{A}_1, x_i)}{\bar{\pi}_0(a_i \mid \mathcal{A}_0, x_i)}\, r(x_i, a_i) \tag{8}$$

$$\hat{R}_{\text{PDR}}(\mathcal{A}_1, \pi, S) \triangleq \hat{R}_{\text{DM}}(\mathcal{A}_1, \pi, S) + \frac{1}{n}\sum_{i=1}^{n} \frac{\bar{\pi}(a_i \mid \mathcal{A}_1, x_i)}{\bar{\pi}_0(a_i \mid \mathcal{A}_0, x_i)}\, \delta(x_i, a_i), \tag{9}$$

where δ is defined in Equation 5. The key difference between PDR and DR is that the importance weight is defined in terms of the marginalized propensities. PDR can be viewed as an instance of the *OffCEM* estimator [32], but with a constraint that each action cluster contains some old actions. Alternatively, PDR can be viewed as the *OPCB* estimator [35], but with "main actions" being limited to certain old actions.

4.1 Analysis

We now characterize the bias and variance of the above estimator, starting with a more general statement for any allowable mapping (per Definition 1). We then consider a simple proxy mapping, which helps ground the theory in a specific example.

The bias will involve the following quantity:

$$\begin{aligned}\Delta(x, a, a') &\triangleq \big(\underbrace{r(x, a) - r(x, a')}_{\Delta_r(x,a,a')}\big) - \big(\underbrace{h(x, a) - h(x, a')}_{\Delta_h(x,a,a')}\big)\\ &= \delta(x, a) - \delta(x, a')\end{aligned}$$

where $\Delta_r(x, a, a')$ and $\Delta_h(x, a, a')$ denote the differences in actual and predicted rewards, respectively, between two actions.

4.1.1 General Proxy Mappings. Following is a general characterization of the bias for an arbitrary proxy operator that satisfies Definition 1. The proof is provided in Appendix A.1.

Theorem 1. *Let ϕ be any proxy operator, with proxies $\mathcal{M}$, satisfying Definition 1. For any $x \in \mathcal{X}$ and $m \in \mathcal{M}(x)$, with some abuse of*

[2]It is straightforward to ensure validity—e.g., by defining equivalences between old actions, then assigning strictly new actions to these equivalence classes.

notation, let

$$\bar{\pi}(m \mid \mathcal{A}, x) \triangleq \sum_{a \in \mathcal{A}(x)} \pi(a \mid \mathcal{A}, x) \mathbb{1}\{\phi(x, a) = m\}$$

denote the marginal probability of selecting an action that gets mapped to proxy m. Further, let

$$\pi(a \mid m \cap \mathcal{A}, x) \triangleq \frac{\pi(a \mid \mathcal{A}, x)}{\bar{\pi}(m \mid \mathcal{A}, x)} \tag{10}$$

denote the conditional probability of selecting an action from m, where $\pi(a \mid m \cap \mathcal{A}, x) \triangleq 0$ *if* $a \notin m$ *or* $\pi(a \mid \mathcal{A}, x) = 0$. *If* π_0 *has full support on* $\mathcal{A}_0$*, then PDR estimator has bias*

$$\begin{aligned}
&\mathrm{Bias}(\hat{R}_{PDR}, \mathcal{A}_1, \pi) \\
&= \mathop{\mathbb{E}}_{(x,r)\sim\mathbb{D}} \mathop{\mathbb{E}}_{m\sim\bar{\pi}(\,\cdot\,\mid \mathcal{A}_1, x)} \mathop{\mathbb{E}}_{\substack{a\sim\pi_0(\,\cdot\,\mid m\cap\mathcal{A}_0, x)\\ a'\sim\pi(\,\cdot\,\mid m\cap\mathcal{A}_1, x)}} \left[\delta(x,a) - \delta(x,a')\right] \qquad (11)\\
&= \mathop{\mathbb{E}}_{(x,r)\sim\mathbb{D}} \mathop{\mathbb{E}}_{m\sim\bar{\pi}(\,\cdot\,\mid \mathcal{A}_1, x)} \mathop{\mathbb{E}}_{\substack{a\sim\pi_0(\,\cdot\,\mid m\cap\mathcal{A}_0, x)\\ a'\sim\pi(\,\cdot\,\mid m\cap\mathcal{A}_1, x)}} \left[\Delta(x,a,a')\right]. \qquad (12)
\end{aligned}$$

To understand Theorem 1, it helps to think of the generative process that computes the bias. First, a context and reward function are sampled from the environment, $(x, r) \sim \mathbb{D}$. Then, a proxy, $m \sim \bar{\pi}(\,\cdot \mid \mathcal{A}_1, x)$, is sampled according to the marginalized propensities under target policy's distribution over new actions. Finally, we sample two actions: one old action, a, from $m \cap \mathcal{A}_0$, using the distribution induced by the logging policy; and one new action, a', from $m \cap \mathcal{A}_1$, using the distribution induced by the target policy. Then, we compute either $\delta(x, a) - \delta(x, a')$ (Equation 11) or $\Delta(x, a, a')$ (Equation 12), which are equivalent.

In Equation 11, the bias decreases as h becomes a better reward predictor; and if h is *perfect*, then there is no bias, which agrees with intuition. Yet having accurate reward prediction is an overly strong condition; according to Equation 12, it actually suffices to predict how the reward of a new action differs from that of an old action from the same class. The latter observation suggests an alternative method for training h via pairwise differences.

We now provide an expression for the variance of PDR (proven in Appendix A.2), which illuminates how the choice of equivalence classes and reward model affect variance.

Theorem 2. *Under the assumptions of Theorem 1, the variance of the PDR estimator can be upper-bounded as*

$$n \,\mathrm{Var}(\hat{R}_{PDR}, \mathcal{A}_1, \pi) \le \underbrace{\mathop{\mathbb{E}}_{(x,r)\sim\mathbb{D}} \mathop{\mathbb{E}}_{a\sim\pi_0(\,\cdot\,\mid \mathcal{A}_0, x)} \left[\left(\frac{\bar{\pi}(a \mid \mathcal{A}_1, x)}{\bar{\pi}_0(a \mid \mathcal{A}_0, x)}\right)^2 \delta(x,a)^2\right]}_{\textit{penalty for importance-weighting}} + \underbrace{\mathop{\mathbb{V}}_{x\sim\mathbb{D}} \mathop{\mathbb{E}}_{a\sim\pi(\,\cdot\,\mid \mathcal{A}_1, x)} \left[h(x,a)\right]}_{\textit{variance of reward prediction}}.$$

Theorem 2 tells us that the variance of PDR estimator can be upper-bounded by the sum of two terms. The first term comes from the variance of importance weighting, and highlights the effect of marginalization over equivalence classes. Large, coarse-grained classes create more equivalences between actions—especially between old and new actions—which effectively reduces the magnitudes of the marginalized importance weights, thereby reducing variance. However, variance alone does not tell the whole story; according to Theorem 1, finer-grained classes may lead to lower bias. Therefore, there is a bias-variance trade-off when determining the optimal granularity of equivalence classes, and this is corroborated by our experimental results. The second term comes from the reward regressor; more expressive reward models can lead to a higher variance. Compared to prior variance analyses for similar estimators (under a fixed action space) [32, 35], our expression provides an upper bound that reduces to two terms. Arguably, this provides a more readable statement, and a more straightforward intuition for the factors that contribute to variance.

Remark 1. It is worth emphasizing that our bias and variance analyses make no assumptions about the proxy mapping other than the validity of the underlying equivalence classes—that is, each class must contain at least one old action—a property which is always satisfiable by construction. No further assumptions are made about how equivalence classes are defined, or how similar the proxy actions are, or the degree of overlap between the old and new actions.

4.1.2 A Simple Proxy Mapping. We now consider a specialization of Definition 1 in which any old action is its own proxy, and any new action is mapped to the most "similar" old action. (This implies that the classes are the old actions, $\mathcal{M}(x) = \mathcal{A}_0(x)$.) A proof is provided in Appendix A.3.

Corollary 1. *For a context,* $x \in \mathcal{X}$*, and action,* $a \in \mathcal{A}_{0\cup 1}(x)$*, let*

$$\phi(x, a) \triangleq \begin{cases} a & \textit{if } a \in \mathcal{A}_0(x), \\ \arg\max_{a' \in \mathcal{A}_0(x)} s(a, a') & \textit{otherwise,} \end{cases} \tag{13}$$

where $s(a, a') \in \mathbb{R}$ *is an arbitrary similarity metric, and the argmax uses arbitrary tie-breaking. If* π_0 *has full support on* $\mathcal{A}_0$*, then PDR, with the proxy operator defined in Equation 13, has bias*

$$\begin{aligned}
&\mathrm{Bias}(\hat{R}_{PDR}, \mathcal{A}_1, \pi) \\
&= \mathop{\mathbb{E}}_{(x,r)\sim\mathbb{D}} \mathop{\mathbb{E}}_{a\sim\pi(\,\cdot\,\mid \mathcal{A}_1, x)} \left[\mathbb{1}\{a \in \mathcal{A}_{1\setminus 0}(x)\} \left(\delta(x, \phi(x,a)) - \delta(x,a)\right)\right] \\
&= \mathop{\mathbb{E}}_{(x,r)\sim\mathbb{D}} \mathop{\mathbb{E}}_{a\sim\pi(\,\cdot\,\mid \mathcal{A}_1, x)} \left[\mathbb{1}\{a \in \mathcal{A}_{1\setminus 0}(x)\} \,\Delta(x, \phi(x,a), a)\right].
\end{aligned}$$

We first note that the bias is really only a function of the strictly new actions, $\mathcal{A}_{1\setminus 0}(x)$. Therefore, the smaller this set is, the lower the bias; and if there are no new actions, then the bias vanishes, which concurs with intuition. Like Equation 11, the bias decreases as h becomes more accurate; or, alternatively, if we can predict how the reward of a new action differs from its (old) proxy.

5 Optimizing the Proxy Method

According to our theoretical analysis in the previous section, the choice of reward model, h, and proxy operator, ϕ, can have a big impact on the bias (Theorem 1) and variance (Theorem 2) of the proxy method. We now describe a procedure to optimize these hyper-parameters.

5.1 Reward Model h

The simplest way to train the reward model is to regress on the observed rewards, using the context and action as covariates. Assuming the reward is continuous (i.e., not binary or ordinal), one would typically aim to minimize the squared error loss: $(r_i - h(x_i, a_i))^2$.

Since the loss function operates on a single action, we refer to this as the *pointwise* formulation. According to Equation 11, minimizing this loss function effectively reduces bias of PDR.

Our bias analysis also motivates an alternative procedure. Recall the term $\Delta(x,a,a') = \Delta_r(x,a,a') - \Delta_h(x,a,a')$ in the bias expression (Equation 12), wherein a and a' are actions from the same equivalence class, and Δ_r and Δ_h represent the differences in the actual and predicted rewards, respectively. The more accurately we can predict Δ_r for actions in the same equivalence class, the more we reduce bias. This motivates a two-step *pairwise* procedure to optimize h, stemming from a decomposition of the reward into terms driven by proxy and residual effects,

$$h(x,a) \;\triangleq\; g(x,\phi(x,a)) \;+\; f(x,a),$$

where we shall interpret g and f to denote the proxy and residual effects, respectively.

Step 1: optimizing the residual effect f. We propose to optimize a pairwise objective,

$$\min_f \sum_{x,r_i,r_j,a_i,a_j:\,\phi(x,a_i)=\phi(x,a_j)} \Big((r_i - r_j) - (f(x,a_i) - f(x,a_j))\Big)^2,$$

where the loss is summed over pairs of observations in the same context, with two actions coming from the same equivalence class. This effectively minimizes $\Delta(x,a,a')$ in Equation 12, thereby reducing bias. In practice, it is challenging or infeasible to observe rewards from two actions in the same context especially in a standard bandit setting, wherein only one action is logged per context. Further, it may be challenging to observe rewards for actions from the same equivalence class. Hence, we can relax the context constraint and pair actions based on a subset of relaxed context features. We conduct an experiment in Section 6.2 to demonstrate the efficacy of this pairwise procedure.

Step 2: optimizing the proxy effect g. After obtaining $\hat{f}$ to model the residual effect, we optimize the following objective to account for the proxy effect:

$$\min_g \sum_{i=1}^{n} \Big((r_i - \hat{h}(x_i,a_i)) - g(x_i,\phi(x_i,a_i))\Big)^2,$$

where the set $\phi(x_i,a_i)$ is mapped to some feature representation of the equivalence class to serve as inputs to the model. In our experiments, we achieve this by computing the centroids for each equivalence class. In the case of the simple proxy mapping introduced in Section 4.1.2, this falls back to using the action itself as the proxy. Optimizing g has an effect of reducing variance, because it stabilizes the reward prediction within the same equivalence class.

5.2 Proxy Operator ϕ

According to Definition 1, a proxy operator ϕ defines a contextual mapping from any action in $\mathcal{A}_{0\cup 1}$ to its equivalence class. Importantly, for ϕ to be *valid*, each equivalence class must contain at least one old action from $\mathcal{A}_0$. This requirement naturally leads to a clustering solution, wherein we cluster the actions in $\mathcal{A}_0$, then assign actions in $\mathcal{A}_1$ to their nearest clusters. To motivate this procedure, recall that the bias of the estimator (given in Equation 12) is a function of $\Delta(x,a,a')$, where a and a' belong to the same equivalence class. Suppose that the true and predicted rewards are Lipschitz functions of action features (arguably, a natural and common assumption), with Lipschitz constants λ_r and λ_h, respectively. Then, using shorthand $\|a-a'\|$ to denote the distance between actions in feature space, it is straightforward to show that

$$\Delta(x,a,a') \;\le\; (\lambda_r + \lambda_h)\,\|a-a'\|\,.$$

Thus, by minimizing intra-class distances—which is precisely what clustering does—we thereby reduce bias.

While there are multiple applicable clustering algorithms in the literature [48], we adopt K-means clustering due to its simplicity, efficiency and broad applicability whenever actions are represented by continuous features.. For each context, x, we apply K-means clustering to $\mathcal{A}_0(x)$, thereby partitioning the old actions into distinct equivalence classes (i.e., clusters). Then, for ever new action in $\mathcal{A}_1(x)$, its equivalence class is determined by the closest cluster from the previous step.

The parameter K effectively controls the granularity of the equivalence classes. The granularity is coarser for smaller K, resulting in a greater number of equivalences. This has the effect of stablilizing the importance weights (which reduces variance), as well as providing more pairwise training examples for optimizing f. On the other hand, choosing K too small can introduce bias by grouping dissimilar actions together, potentially obscuring important distinctions between actions with different reward distributions. Thus, choice of K represents another bias-variance trade-off. We investigate this trade-off empirically in our experiments.

6 Experiments

We empirically evaluate our proposed methods on synthetic and real-world data. Recall that the intersection method yields two estimators, IIPS and IDR (Equations 6 and 7), and the proxy method also yields two estimators, PIPS and PDR (Equations 8 and 9). We compare these to two natural baselines that do not adjust for changing action sets: IPS (Equation 2) and DR (Equation 4).

6.1 Experiments on Synthetic Data

In this section, we describe our experiments on synthetic data, designed to simulate a simple recommendation environment. This allows us to explore how the various facets of our problem setting—such as action set overlap and reward model misspecification—affect our proposed estimators. We also investigate the effect of equivalence class granularity on the proxy method.

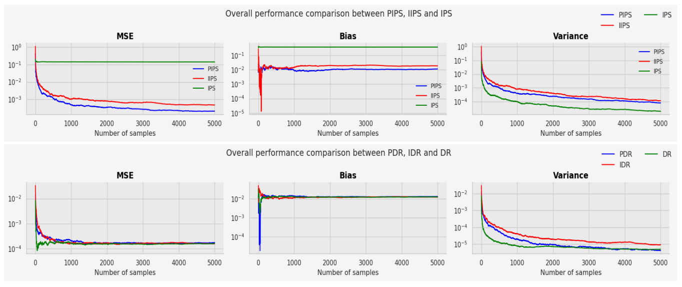

Figure 1: Synthetic data: MSE, absolute bias and variance.

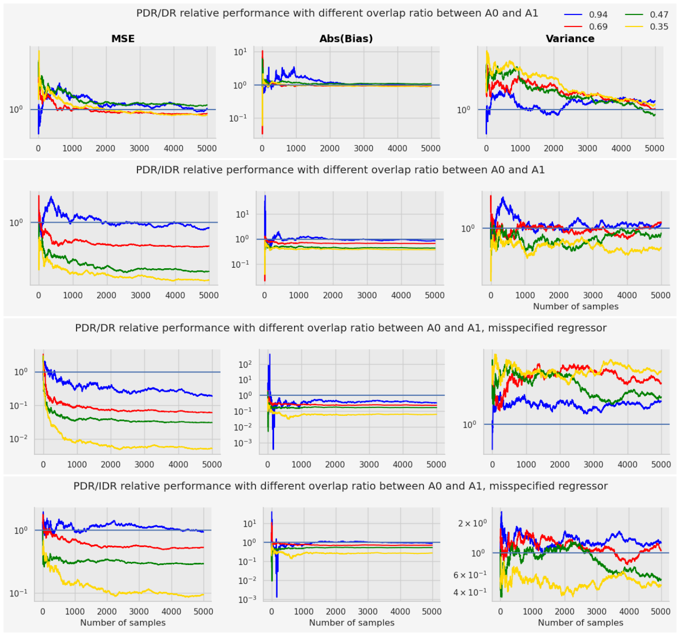

Figure 2: Synthetic data: Comparing PDR to DR/IDR with varying overlap ratios between $\mathcal{A}_0$ and $\mathcal{A}_1$.

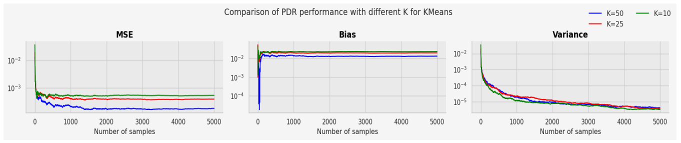

Figure 3: Synthetic data: Varying the granularity of equivalence classes used in PDR via the number of clusters K

6.1.1 Synthetic Environment. In our synthetic experiments, we consider 1000 actions, each represented by a 10-dimensional action vector v_a, and contexts represented by a 3-dimensional vector v_c. We set the *true* expected reward for a context-action pair as:

$$r(x,a) \triangleq \alpha^T v_a + \beta^T v_a^2 + \gamma^T v_c + \zeta^T (v_a \otimes v_c),$$

where parameters α, β, γ, and ζ are constants and the expected rewards are designed to be bounded between 0 and 1. Observed rewards are sampled from a normal distribution centered at $r(x,a)$ with standard deviation of 0.1.

To define the candidate generators and policies, we use an action *scoring function* that is essentially a noisy reward predictor.[3] For each context-action pair, we define the scoring function as $\tilde{r} \triangleq r(x,a) + \text{Uni}(0,\eta)$, where $\text{Uni}(0,\eta)$ is uniformly distributed noise in $[0,\eta]$. We allow different noise parameters for logging and target, and we denote these by η_0 and η_1, respectively. Each candidate generator selects the 50 highest-scoring actions according to $\tilde{r}$; then, the policy samples an action from this set using a softmax distribution over the scores. Note that adjusting the noise parameters not only allows us to control the *quality* of a candidate generator (and its associated policy), it also allows us to control the *overlap* between the logging and target action sets. Higher magnitude noise creates greater variation in ranking, leading to less overlap.

[3]This reward predictor is solely used to define the candidate generators and policies, and should not be confused with the reward model used in the estimators, which is trained from logged data.

6.1.2 Results. Figure 1 plots the MSE, absolute value of bias and variance of each estimator as a function of the data size. In these results, the logging scoring function uses noise parameter $\eta_0 = 0.5$, while the target scoring function uses noise parameter $\eta_1 = 0.2$, yielding an average overlap of approximately 30.6% between the candidate sets $\mathcal{A}_0(x)$ and $\mathcal{A}_1(x)$. The proxy-based methods use $K = |\mathcal{A}_0(x)| = 50$—effectively recreating the simple proxy mapping from Corollary 1, in which each old action is its own class.

Among the importance-weighting variants (i.e., without reward modeling), PIPS notably outperforms both IIPS and standard IPS. Compared to IIPS, PIPS achieves lower variance by marginalizing propensities over equivalence classes, while simultaneously improving bias through better reward estimation for strictly new actions (based on observed proxies). Compared to standard IPS, where actions favored by the target policy but never taken by the logging policy contributes nothing to the IPS estimate. Consequently, the variance of the estimator may appear reduced—not because the estimate is more stable in a meaningful sense, but because it systematically omits parts of the action space. This omission induces bias while artificially lowering variance, since unobserved actions do not contribute any stochasticity to the estimate.

Focusing now on the model-based estimators, all methods exhibit comparable MSE, as the reward model significantly reduces bias across the board. Consequently, although the relative variance relationships observed above among importance-weighting methods is maintained, the differences in variance become much less pronounced. However, as we will subsequently show, this performance parity is likely due to having a sufficiently accurate reward regressor; notably, PDR shows greater robustness to model misspecification.

In Figure 2, we vary the overlap between $\mathcal{A}_0$ and $\mathcal{A}_1$ by adjusting the noise parameters used in the logging and target scoring functions. To simplify, we set both noise parameters to the same value, $\eta_0 = \eta_1 = \eta$, so that varying η alone controls the overlap ratio. In these comparisons, we plot the *ratio* of the evaluation metrics between the given estimators. For example, in the PDR/DR MSE plot, a ratio of 0.8 indicates that the MSE of PDR is 80% of the MSE of DR. Values below 1 indicate superior performance of the numerator estimator PDR and vice-versa.

Varying action set overlap. The first and second subplots of Figure 2 compare the performance of PDR against DR and IDR, respectively, under varying overlap ratios between $\mathcal{A}_0$ and $\mathcal{A}_1$. Intuitively, as overlap decreases, and the strictly new action set grows, the bias incurred from strictly new actions increases. PDR addresses this bias via marginalized importance weighting, whereas DR and IDR do not. Since squared bias is the dominant component of MSE (as can be observed in Figure 1), it is critically important to address this bias. Indeed, in both comparisons and all overlap ratios, we find that PDR demonstrates superior performance as the overlap ratio decreases, and this decrease is primarily driven by bias reduction.

Model misspecification. The third and fourth subplots of Figure 2 examine the robustness of the three model-based methods against model misspecification. To simulate model misspecification, we intentionally bias the regressors via scaling their outputs by a factor of 0.7 after training. As before, we plot the ratio of metrics, comparing PDR against DR and IDR, respectively. DR demonstrates vulnerability to the biased regressor, with PDR increasingly outperforming DR in terms of both MSE and bias as the overlap ratio decreases. In contrast, the comparison between PDR and IDR is much closer, indicating that both methods are robust to model misspecification. Nonetheless, PDR still performs better than IDR in this setting.

Equivalence class granularity. Figure 3 shows the effect of varying the number of equivalence classes used for PDR, which is determined by the number of clusters used in K-Means. Increasing K creates finer-grained equivalence classes. At one extreme, when $K = |\mathcal{A}_0(x)| = 50$, we obtain the simple proxy mapping in which each old action is its own proxy. Reducing K stabilizes the marginalized importance weights and lowers variance, but can introduce bias by creating more heterogeneous equivalence classes. In synthetic experiments, this bias-variance trade-off is evident: larger K reduces bias but increases variance. However, real-world results show the trade-off is not always monotonic, and the optimal K is problem-dependent.

6.2 Experiments on Real-World Data

This section describes our experiments on a from an e-commerce platform. The results showcase the superiority of the PDR estimator in a real-world setting. We also conduct an ablation study of the estimator's two primary parameters: equivalence class granularity and reward model training.

6.2.1 Data and Experiment Setup. The Open Bandit Dataset (OBD) [30] contains logged bandit feedback from a large-scale fashion e-commerce platform, comprising six datasets across three campaigns ('ALL', 'men', 'women'). Each campaign was randomly assigned either a uniform random policy or a Bernoulli TS policy. We focus on the 'ALL' campaign, which includes 80 possible actions per context. Contexts are represented by 20-dimensional vectors, and each action is represented by a 4-dimensional action vector. The reward is binary: 0 or 1. We set the uniform random policy as the logging policy and the Bernoulli TS policy as the target policy. To simulate reward estimation after adding new merchandise to the candidate pool, we remove the last 40 actions from the original logging pool. All events involving these actions are discarded, and logging propensities are re-normalized across the remaining 40 merchandise items, resulting in $\pi_0(a|x) = 1/40$. Using this adjusted log, we estimate rewards for the target policy, which selects from the full set of 80 merchandise items. We train a logistic regression model to capture the binary reward presented in OBD. Input features are preprocessed using min-max normalization to address distributional differences among feature dimensions.

6.2.2 Results. As illustrated in Table 1, the results align with our synthetic experiments. PDR achieves the lowest MSE, highlighting the advantage of proxy-based estimators. Below, we analyze the proxy method's performance across different levels of equivalence class granularity, different overlap ratio between $\mathcal{A}_0$ and $\mathcal{A}_1$ and discuss the potential advantage of pairwise regression.

Table 1: Open Bandit: Comparison between DR, IDR and PDR

Estimator	MSE	Squared bias	Variance
DR	1.72×10^{-6}	6.66×10^{-7}	1.05×10^{-6}
IDR	5.57×10^{-7}	5.04×10^{-7}	5.35×10^{-8}
PDR	$\mathbf{4.93 \times 10^{-7}}$	$\mathbf{4.58 \times 10^{-7}}$	$\mathbf{3.44 \times 10^{-8}}$

Table 2: Open Bandit: Comparison between PDR using different number of equivalence classes

Num. proxies (K)	MSE	Squared bias	Variance
5	8.91×10^{-7}	8.70×10^{-7}	$\mathbf{2.08 \times 10^{-8}}$
20	$\mathbf{5.03 \times 10^{-7}}$	$\mathbf{4.57 \times 10^{-7}}$	4.61×10^{-8}
40	6.72×10^{-7}	6.05×10^{-7}	6.64×10^{-8}

Table 3: Open Bandit: Bias comparison between PIPS using different number of equivalence classes

Num. proxies (K)	bias: old actions	bias: new actions
5	7.35×10^{-4}	1.63×10^{-3}
20	1.39×10^{-4}	$\mathbf{1.44 \times 10^{-3}}$
40	$\mathbf{2.47 \times 10^{-5}}$	1.59×10^{-3}

Equivalence class granularity. Table 2 presents PDR performance for various cluster granularities, controlled by the number of clusters in K-Means. Coarser clusters (smaller K) effectively reduce variance, while moderate granularity provides optimal bias reduction. To better understand how granularity impacts bias, we decompose the bias into two parts: bias associated with old actions and bias from new actions. To do so, we decompose the target policy into two components, $p(a \in \mathcal{A}_{0\cap 1})\pi(a \mid \mathcal{A}_{0\cap 1}, x)$ and $p(a \in \mathcal{A}_{1\setminus 0})\pi(a \mid \mathcal{A}_{1\setminus 0}, x)$, and examine each separately.

We focus on PIPS rather than PDR to isolate the effect of marginalized propensities from the regressor. Table 3 illustrates bias decomposition on old versus new actions using PIPS across varying granularities. Bias on old actions generally increases with coarser clustering due to using marginalized propensities instead of the "raw" propensities. However, bias on new actions dominates bias on old actions in magnitude, achieving its minimum at moderate granularity. Extremely fine clusters lead to "overfitting," treating new actions identically to their nearest old action, whereas overly coarse clusters lead to "underfitting."

Varying action set overlap. In Table 4, we compare DR, IDR, and PDR with a variable number of actions available to the logging policy, which effectively determines the amount of overlap with the target policy's action set. Modifying the protocol described in Section 6.2.1, we randomly keep 30%, 50%, 70%, and 90% of the original 80 actions in the logging action set. Similar to the synthetic data results, PDR outperforms both DR and IDR, with DR showing the weakest performance due to deficient support. The advantage of PDR over IDR becomes more pronounced as the overlap ratio decreases, as PDR effectively adjusts for deficient support through proxy mapping, whereas IDR relies on conditioning onto

Table 4: Open Bandit: MSE comparison between DR, IDR and PDR with different action set overlap ratios

Overlap ratios	MSE: DR	MSE: IDR	MSE: PDR
30%	2.98×10^{-6}	4.03×10^{-7}	$\mathbf{6.12 \times 10^{-8}}$
50%	2.52×10^{-6}	5.54×10^{-7}	$\mathbf{5.41 \times 10^{-7}}$
70%	8.46×10^{-7}	3.11×10^{-7}	$\mathbf{1.32 \times 10^{-7}}$
90%	5.02×10^{-7}	4.31×10^{-7}	$\mathbf{3.21 \times 10^{-7}}$

Table 5: Open Bandit: Comparison between PDR using pairwise vs pointwise regressor

Regressor	MSE	Squared bias	Variance
Pointwise	5.10×10^{-7}	4.75×10^{-7}	$\mathbf{3.50 \times 10^{-8}}$
Pairwise	$\mathbf{2.12 \times 10^{-7}}$	$\mathbf{1.77 \times 10^{-7}}$	3.52×10^{-8}

the intersection of the action sets. This study further highlights the robustness of PDR in scenarios with limited overlap between logging and target policies.

Pairwise regression. In OBD, we observe only one action per context. Hence, for pairwise training, we must relax the context specificity constraint and pair actions based solely on their equivalence classes, which are defined independently of context here. Table 5 compares PDR estimators leveraging either pointwise or pairwise regression to train the reward model. The pairwise regressor achieves lower MSE thanks to its reduced bias, which concurs with our bias decomposition (Theorem 1). The pointwise regressor exhibits slightly lower variance; however, the difference is likely statistically insignificant.

7 Conclusion

We introduced a framework for off-policy evaluation of candidate generators in two-stage recommender systems, framing the problem as off-policy evaluation with a changing action set. We proposed two importance weighting methods—intersection-based and proxy-based—with the latter leveraging equivalence classes and marginalized propensities to handle unseen actions. Our theoretical analysis highlights the inherent bias-variance trade-offs, and our experiments on synthetic and real-world data demonstrate that the proposed proxy-based estimators achieve lower bias and comparable or reduced variance relative to the baselines. This work therefore provides a practical path toward more principled offline evaluation of candidate generators in two-stage recommender systems.

That said, while the methods are promising and broadly applicable, it is important to acknowledge their limitations. Specifically, the quality of the PDR estimator depends on the quality of the equivalence classes and reward model. If the equivalence classes exhibit too much variation in their Δ values—which, recall, are functions of the true and predicted rewards—then the proxy-based reward estimates will suffer (as supported by our analysis in Section 4.1). Thus, construction (and possibly joint optimization) of the equivalence classes and reward model remains an important direction for future work.

References

[1] Aman Agarwal, Soumya Sankar Basu, Tobias Schnabel, and Thorsten Joachims. 2017. Effective Evaluation Using Logged Bandit Feedback from Multiple Loggers. *Knowledge Discovery and Data Mining* (2017).

[2] Deepak K. Agarwal and Maxim Gurevich. 2012. Fast top-k retrieval for model based recommendation. In *Web Search and Data Mining*. 483–492.

[3] Léon Bottou, J. Peters, Joaquin Quiñonero Candela, Denis Xavier Charles, David Maxwell Chickering, Elon Portugaly, Dipankar Ray, Patrice Y. Simard, and Edward Snelson. 2013. Counterfactual Reasoning and Learning Systems: The Example of Computational Advertising. *Journal of Machine Learning Research* 14 (2013).

[4] Paul Covington, Jay K. Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *ACM Conference on Recommender Systems*.

[5] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, and Dasarathi Sampath. 2010. The YouTube video recommendation system. In *ACM Conference on Recommender Systems*.

[6] Miroslav Dudík, John Langford, and Lihong Li. 2011. Doubly Robust Policy Evaluation and Learning. In *International Conference on Machine Learning*.

[7] Chantat Eksombatchai, Pranav Jindal, Zitao Liu, Yuchen Liu, Rahul Sharma, Charles Sugnet, Mark Ulrich, and Jure Leskovec. 2018. Pixie: A system for recommending 3+ billion items to 200+ million users in real-time. In *World Wide Web Conference*.

[8] Mehrdad Farajtabar, Yinlam Chow, and Mohammad Ghavamzadeh. 2018. More Robust Doubly Robust Off-policy Evaluation. In *International Conference on Machine Learning*.

[9] Nicolò Felicioni, Maurizio Ferrari Dacrema, Marcello Restelli, and Paolo Cremonesi. 2022. Off-Policy Evaluation with Deficient Support Using Side Information. In *Neural Information Processing Systems*.

[10] Scott Fujimoto, David Meger, and Doina Precup. 2019. Off-Policy Deep Reinforcement Learning without Exploration. In *ICML*.

[11] Alexandre Gilotte, Clément Calauzènes, Thomas Nedelec, Alexandre Abraham, and Simon Dollé. 2018. Offline A/B Testing for Recommender Systems. In *Web Search and Data Mining*.

[12] Daniel G. Horvitz and D. J. Thompson. 1952. A generalization of sampling without replacement from a finite universe. *J. Amer. Statist. Assoc.* 47, 260 (1952), 663–685.

[13] Jiri Hron, Karl Krauth, Michael I. Jordan, and Niki Kilbertus. 2021. On component interactions in two-stage recommender systems. In *Neural Information Processing Systems*.

[14] Edward L. Ionides. 2008. Truncated Importance Sampling. *Journal of Computational and Graphical Statistics* 17, 2 (2008), 295–311.

[15] Nan Jiang and Lihong Li. 2016. Doubly Robust Off-policy Value Evaluation for Reinforcement Learning. In *International Conference on Machine Learning*.

[16] Thorsten Joachims, Adith Swaminathan, and Maarten de Rijke. 2018. Deep Learning with Logged Bandit Feedback. In *International Conference on Learning Representations*.

[17] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased Learning-to-Rank with Biased Feedback. In *Web Search and Data Mining*.

[18] Nathan Kallus. 2018. Balanced Policy Evaluation and Learning. In *Neural Information Processing Systems*.

[19] Nathan Kallus and Angela Zhou. 2018. Policy Evaluation and Optimization with Continuous Treatments. In *Artificial Intelligence and Statistics*.

[20] John Langford, Alexander L. Strehl, and Jennifer Wortman Vaughan. 2008. Exploration scavenging. In *International Conference on Machine Learning*.

[21] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. 2011. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Web Search and Data Mining*.

[22] Lihong Li, Rémi Munos, and Csaba Szepesvari. 2015. Toward Minimax Off-policy Value Estimation. In *Artificial Intelligence and Statistics*.

[23] Anqi Liu, Hao Liu, Anima Anandkumar, and Yisong Yue. 2019. Triply Robust Off-Policy Evaluation. *CoRR* abs/1911.05811 (2019).

[24] Ben London and Thorsten Joachims. 2020. Offline Policy Evaluation with New Arms. In *NeurIPS Workshop on Offline Reinforcement Learning*.

[25] Ben London and Ted Sandler. 2019. Bayesian Counterfactual Risk Minimization. In *International Conference on Machine Learning*.

[26] Ashique Rupam Mahmood, H. V. Hasselt, and Richard S. Sutton. 2014. Weighted importance sampling for off-policy learning with linear function approximation. In *Neural Information Processing Systems*.

[27] Shumpei Okura, Yukihiro Tagami, Shingo Ono, and Akira Tajima. 2017. Embedding-based news recommendation for millions of users. In *Knowledge Discovery and Data Mining*.

[28] Jie Peng, Hao Zou, Jiashuo Liu, Shaoming Li, Yibao Jiang, Jian Pei, and Peng Cui. 2023. Offline policy evaluation in large action spaces via outcome-oriented action grouping. In *Web Conference*.

[29] Noveen Sachdeva, Yi-Hsun Su, and Thorsten Joachims. 2020. Off-policy Bandits with Deficient Support. In *Knowledge Discovery and Data Mining*.

[30] Yuta Saito, Shunsuke Aihara, Megumi Matsutani, and Yusuke Narita. 2020. Large-scale Open Dataset, Pipeline, and Benchmark for Bandit Algorithms. *ArXiv* abs/2008.07146 (2020).

[31] Yuta Saito and Thorsten Joachims. 2022. Off-Policy Evaluation for Large Action Spaces via Embeddings. In *International Conference on Machine Learning*.

[32] Yuta Saito, Qingyang Ren, and Thorsten Joachims. 2023. Off-Policy Evaluation for Large Action Spaces via Conjunct Effect Modeling. In *International Conference on Machine Learning*.

[33] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as Treatments: Debiasing Learning and Evaluation. In *International Conference on Machine Learning*.

[34] John Schulman, Sergey Levine, P. Abbeel, Michael I. Jordan, and Philipp Moritz. 2015. Trust Region Policy Optimization. In *International Conference on Machine Learning*.

[35] Tatsuhiro Shimizu, Koichi Tanaka, Ren Kishimoto, Haruka Kiyohara, Masahiro Nomura, and Yuta Saito. 2024. Effective Off-Policy Evaluation and Learning in Contextual Combinatorial Bandits. In *ACM Conference on Recommender Systems (RecSys)*.

[36] Alexander L. Strehl, John Langford, Lihong Li, and Sham M. Kakade. 2010. Learning from Logged Implicit Exploration Data. In *Neural Information Processing Systems*.

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

[38] Yi-Hsun Su, Lequn Wang, Michele Santacatterina, and Thorsten Joachims. 2019. CAB: Continuous Adaptive Blending for Policy Evaluation and Learning. In *International Conference on Machine Learning*.

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

[40] Adith Swaminathan and Thorsten Joachims. 2015. The Self-Normalized Estimator for Counterfactual Learning. In *Neural Information Processing Systems*.

[41] Adith Swaminathan, Akshay Krishnamurthy, Alekh Agarwal, Miroslav Dudík, John Langford, Damien Jose, and Imed Zitouni. 2017. Off-policy evaluation for slate recommendation. In *Neural Information Processing Systems*.

[42] Philip S. Thomas and Emma Brunskill. 2016. Data-Efficient Off-Policy Policy Evaluation for Reinforcement Learning. In *International Conference on Machine Learning*.

[43] Philip S. Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. 2015. High-Confidence Off-Policy Evaluation. In *AAAI*.

[44] Nikos A. Vlassis, Aurélien F. Bibaut, Maria Dimakopoulou, and Tony Jebara. 2019. On the Design of Estimators for Bandit Off-Policy Evaluation. In *International Conference on Machine Learning*.

[45] Nikos A. Vlassis, Ashok Chandrashekar, Fernando Amat Gil, and Nathan Kallus. 2021. Control variates for slate off-policy evaluation, In Neural Information Processing Systems. *Neural Information Processing Systems*.

[46] Yu-Xiang Wang, Alekh Agarwal, and Miroslav Dudík. 2017. Optimal and Adaptive Off-policy Evaluation in Contextual Bandits. In *International Conference on Machine Learning*.

[47] Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, and M. Zhou. 2020. Mind: A large-scale dataset for news recommendation. In *Proceedings of the 58th annual meeting of the association for computational linguistics*.

[48] Dongkuan Xu and Ying jie Tian. 2015. A Comprehensive Survey of Clustering Algorithms. *Annals of Data Science* 2, 2 (2015), 165–193.

[49] Xinyang Yi, Ji Yang, Lichan Hong, Derek Zhiyuan Cheng, Lukasz Heldt, Aditee Kumthekar, Zhe Zhao, Li Wei, and Ed H. Chi. 2019. Sampling-bias-corrected neural modeling for large corpus item recommendations. In *ACM Conference on Recommender Systems*.

[50] Eva Zangerle and Christine Bauer. 2022. Evaluating recommender systems: survey and framework. *Comput. Surveys* 55, 8 (2022), 1–38.

A Deferred Proofs

This appendix contains proofs deferred from the main manuscript.

A.1 Proof of Theorem 1

Via linearity of expectation, we can decompose the bias of PDR into the expected value of the importance-weighed term plus the bias of DM:

$$\begin{aligned}\mathrm{Bias}(\hat{R}_{\mathrm{PDR}}, \mathcal{A}_1, \pi) &= \mathop{\mathbb{E}}_{S\sim(\mathbb{D}\times\pi_0)^n}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{\bar{\pi}(a_i \mid \mathcal{A}_1, x_i)}{\bar{\pi}_0(a_i \mid \mathcal{A}_0, x_i)}\delta(x_i, a_i)\right] + \mathop{\mathbb{E}}_{S\sim(\mathbb{D}\times\pi_0)^n}\left[\hat{R}_{\mathrm{DM}}(\mathcal{A}_1, \pi, S) - R(\mathcal{A}_1, \pi)\right] \\ &= \mathop{\mathbb{E}}_{S\sim(\mathbb{D}\times\pi_0)^n}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{\bar{\pi}(a_i \mid \mathcal{A}_1, x_i)}{\bar{\pi}_0(a_i \mid \mathcal{A}_0, x_i)}\delta(x_i, a_i)\right] + \mathrm{Bias}(\hat{R}_{\mathrm{DM}}, \mathcal{A}_1, \pi).\end{aligned}$$

The bias of DM is given by Equation 3, and we can simplify it using the notation in Equation 5:

$$\mathrm{Bias}(\hat{R}_{\mathrm{DM}}, \mathcal{A}_1, \pi) = \mathop{\mathbb{E}}_{(x,r)\sim\mathbb{D}}\ \mathop{\mathbb{E}}_{a\sim\pi(\,\cdot\,\mid \mathcal{A}_1,x)}\big[-\delta(x,a)\big].$$

For the importance-weighted term, we can apply linearity of expectation to push the expectations inside the average, then rename the variables (because they are i.i.d.):

$$\mathop{\mathbb{E}}_{S\sim(\mathbb{D}\times\pi_0)^n}\left[\frac{1}{n}\sum_{i=1}^{n}\frac{\bar{\pi}(a_i \mid \mathcal{A}_1, x_i)}{\bar{\pi}_0(a_i \mid \mathcal{A}_0, x_i)}\delta(x_i, a_i)\right] = \mathop{\mathbb{E}}_{(x,r)\sim\mathbb{D}}\ \mathop{\mathbb{E}}_{a\sim\pi_0(\,\cdot\,\mid \mathcal{A}_0,x)}\left[\frac{\bar{\pi}(a \mid \mathcal{A}_1, x)}{\bar{\pi}_0(a \mid \mathcal{A}_0, x)}\delta(x,a)\right].$$

Combining these quantities, we get

$$\mathrm{Bias}(\hat{R}_{\mathrm{PDR}}, \mathcal{A}_1, \pi) = \mathop{\mathbb{E}}_{(x,r)\sim\mathbb{D}}\left[\underbrace{\mathop{\mathbb{E}}_{a\sim\pi_0(\,\cdot\,\mid \mathcal{A}_0,x)}\left[\frac{\bar{\pi}(a \mid \mathcal{A}_1, x)}{\bar{\pi}_0(a \mid \mathcal{A}_0, x)}\delta(x,a)\right]}_{(a)} - \underbrace{\mathop{\mathbb{E}}_{a\sim\pi(\,\cdot\,\mid \mathcal{A}_1,x)}\big[\delta(x,a)\big]}_{(b)}\right]. \tag{14}$$

We then analyze term (a) in Equation 14:

$$\begin{aligned}\mathop{\mathbb{E}}_{a\sim\pi_0(\,\cdot\,\mid \mathcal{A}_0,x)}\left[\frac{\bar{\pi}(a \mid \mathcal{A}_1, x)}{\bar{\pi}_0(a \mid \mathcal{A}_0, x)}\delta(x,a)\right] &= \sum_{a\in\mathcal{A}_0}\pi_0(a \mid \mathcal{A}_0, x)\frac{\bar{\pi}(a \mid \mathcal{A}_1, x)}{\bar{\pi}_0(a \mid \mathcal{A}_0, x)}\delta(x,a) \\ &= \sum_{m\in\mathcal{M}(x)}\ \sum_{a\in m\cap\mathcal{A}_0(x)}\pi_0(a \mid \mathcal{A}_0, x)\frac{\bar{\pi}(a \mid \mathcal{A}_1, x)}{\bar{\pi}_0(a \mid \mathcal{A}_0, x)}\delta(x,a) &(15)\\ &= \sum_{m\in\mathcal{M}(x)}\ \sum_{a\in m\cap\mathcal{A}_0(x)}\pi_0(a \mid \mathcal{A}_0, x)\frac{\bar{\pi}(m \mid \mathcal{A}_1, x)}{\bar{\pi}_0(m \mid \mathcal{A}_0, x)}\delta(x,a) &(16)\\ &= \sum_{m\in\mathcal{M}(x)}\ \sum_{a\in m\cap\mathcal{A}_0(x)}\pi_0(a \mid m\cap\mathcal{A}_0(x), x)\bar{\pi}(m \mid \mathcal{A}_1, x)\delta(x,a) &(17)\\ &= \mathop{\mathbb{E}}_{m\sim\bar{\pi}(\,\cdot\,\mid \mathcal{A}_1,x)}\ \mathop{\mathbb{E}}_{a\sim\pi_0(\,\cdot\,\mid m\cap\mathcal{A}_0(x),x)}\big[\delta(x,a)\big]. &(18)\end{aligned}$$

Equation 15 uses the fact that ϕ partitions $\mathcal{A}_0(x)$, such that every $a \in \mathcal{A}_0(x)$ belongs to exactly one $m \in \mathcal{M}(x)$. Equation 16 uses the equivalences $\bar{\pi}(a \mid \mathcal{A}_1, x) = \bar{\pi}(m \mid \mathcal{A}_1, x)$ and $\bar{\pi}_0(a \mid \mathcal{A}_0, x) = \bar{\pi}_0(m \mid \mathcal{A}_0, x)$ when $m = \phi(x,a)$. Finally, Equation 17 applies Equation 10. Rearranging terms, we end up with Equation 18, expressed as a double expectation.

Applying the same reasoning to term (b) in Equation 14:

$$\begin{aligned}\mathop{\mathbb{E}}_{a\sim\pi(\,\cdot\,\mid \mathcal{A}_1,x)}\big[\delta(x,a)\big] &= \sum_{a\in\mathcal{A}_1}\pi(a \mid \mathcal{A}_1, x)\delta(x,a) \\ &= \sum_{m\in\mathcal{M}(x)}\ \sum_{a\in m\cap\mathcal{A}_1(x)}\frac{\bar{\pi}(m \mid \mathcal{A}_1, x)}{\bar{\pi}(m \mid \mathcal{A}_1, x)}\pi(a \mid \mathcal{A}_1, x)\delta(x,a) \\ &= \sum_{m\in\mathcal{M}(x)}\ \sum_{a\in m\cap\mathcal{A}_1(x)}\bar{\pi}(m \mid \mathcal{A}_1, x)\pi(a \mid m\cap\mathcal{A}_1(x), x)\delta(x,a) \\ &= \mathop{\mathbb{E}}_{m\sim\bar{\pi}(\,\cdot\,\mid \mathcal{A}_1,x)}\ \mathop{\mathbb{E}}_{a\sim\pi(\,\cdot\,\mid m\cap\mathcal{A}_1(x),x)}\big[\delta(x,a)\big]. &(19)\end{aligned}$$

To complete the proof, we simply plug Equations 18 and 19 into Equation 14 and apply linearity of expectation to combine the expectations.

A.2 Proof of Theorem 2

First, to simplify presentation, we introduce some notation,

$$\hat{\delta}(x,a) \triangleq \frac{\bar{\pi}(a \mid \mathcal{A}_1, x)}{\bar{\pi}_0(a \mid \mathcal{A}_0, x)}\delta(x,a),$$

to denote the marginal importance-weighting of the reward prediction error, δ. We then leverage the i.i.d. assumption to express the variance of the estimator as the variance of a single interaction:

$$\begin{aligned} n\ \mathrm{Var}(\hat{R}_{\mathrm{PDR}}, \mathcal{A}_1, \pi) &= \frac{1}{n}\sum_{i=1}^{n}\left\{\underset{\substack{(x_i,r_i)\sim\mathbb{D} \\ a_i\sim\pi_0(\,\cdot\,\mid \mathcal{A}_0,x_i)}}{\mathbb{V}}\left[\hat{\delta}(x_i,a_i) + \underset{a'\sim\pi(\,\cdot\,\mid\mathcal{A}_1,x_i)}{\mathbb{E}} h(x_i,a')\right]\right\} \\ &= \underset{\substack{(x,r)\sim\mathbb{D} \\ a\sim\pi_0(\,\cdot\,\mid \mathcal{A}_0,x)}}{\mathbb{V}}\left[\hat{\delta}(x,a) + \underset{a'\sim\pi(\,\cdot\,\mid\mathcal{A}_1,x)}{\mathbb{E}} h(x,a')\right]. \end{aligned} \tag{20}$$

Then, we apply the law of total variance to decompose this variance:

$$(20) = \underset{(x,r)\sim\mathbb{D}}{\mathbb{E}}\ \underset{a\sim\pi_0(\,\cdot\,\mid\mathcal{A}_0,x)}{\mathbb{V}}\left[\hat{\delta}(x,a) + \underset{a'\sim\pi(\,\cdot\,\mid\mathcal{A}_1,x)}{\mathbb{E}}\left[h(x,a')\right]\right] \tag{21}$$

$$+ \underset{(x,r)\sim\mathbb{D}}{\mathbb{V}}\left[\underset{a\sim\pi_0(\,\cdot\,\mid\mathcal{A}_0,x)}{\mathbb{E}}\left[\hat{\delta}(x,a)\right] + \underset{a'\sim\pi(\,\cdot\,\mid\mathcal{A}_1,x)}{\mathbb{E}}\left[h(x,a')\right]\right]. \tag{22}$$

In Equation 21, we have that the DM term, $\mathbb{E}_{a'\sim\pi(\,\cdot\,\mid\mathcal{A}_1,x)}$, is constant w.r.t. the inner variance over $a \sim \pi_0(\,\cdot\mid\mathcal{A}_0,x)$, and can therefore be ignored. Then, expanding the inner variance using the identity $\mathrm{Var}[\hat{\delta}] = \mathbb{E}[\hat{\delta}^2] - (\mathbb{E}[\hat{\delta}])^2$, we have

$$\begin{aligned} (21) &= \underset{(x,r)\sim\mathbb{D}}{\mathbb{E}}\ \underset{a\sim\pi_0(\,\cdot\,\mid\mathcal{A}_0,x)}{\mathbb{V}}\left[\hat{\delta}(x,a)\right] \\ &= \underset{(x,r)\sim\mathbb{D}}{\mathbb{E}}\ \underset{a\sim\pi_0(\,\cdot\,\mid\mathcal{A}_0,x)}{\mathbb{E}}\left[\hat{\delta}(x,a)^2\right] - \underset{(x,r)\sim\mathbb{D}}{\mathbb{E}}\left(\underset{a\sim\pi_0(\,\cdot\,\mid\mathcal{A}_0,x)}{\mathbb{E}}\left[\hat{\delta}(x,a)\right]\right)^2. \end{aligned} \tag{23}$$

In Equation 22, we have that the inner expectations involve independent random variables, $a \sim \pi_0(\,\cdot\mid\mathcal{A}_0,x)$ and $a' \sim \pi(\,\cdot\mid\mathcal{A}_1,x)$. Thus, we can decompose the outer variance as

$$(22) = \underset{(x,r)\sim\mathbb{D}}{\mathbb{V}}\ \underset{a\sim\pi_0(\,\cdot\,\mid\mathcal{A}_0,x)}{\mathbb{E}}\left[\hat{\delta}(x,a)\right] + \underset{x\sim\mathbb{D}}{\mathbb{V}}\ \underset{a'\sim\pi(\,\cdot\,\mid\mathcal{A}_1,x)}{\mathbb{E}}\left[h(x,a')\right]. \tag{24}$$

Note that the reward, r, is irrelevant to the second term and is thus omitted. We then combine Equations 23 and 24 and obtain

$$\begin{aligned} (21) + (22) &= \underset{(x,r)\sim\mathbb{D}}{\mathbb{E}}\ \underset{a\sim\pi_0(\,\cdot\,\mid\mathcal{A}_0,x)}{\mathbb{E}}\left[\hat{\delta}(x,a)^2\right] - \underset{(x,r)\sim\mathbb{D}}{\mathbb{E}}\left(\underset{a\sim\pi_0(\,\cdot\,\mid\mathcal{A}_0,x)}{\mathbb{E}}\left[\hat{\delta}(x,a)\right]\right)^2 \\ &\quad + \underset{(x,r)\sim\mathbb{D}}{\mathbb{V}}\ \underset{a\sim\pi_0(\,\cdot\,\mid\mathcal{A}_0,x)}{\mathbb{E}}\left[\hat{\delta}(x,a)\right] + \underset{x\sim\mathbb{D}}{\mathbb{V}}\ \underset{a'\sim\pi(\,\cdot\,\mid\mathcal{A}_1,x)}{\mathbb{E}}\left[h(x,a')\right] \\ &= \underset{(x,r)\sim\mathbb{D}}{\mathbb{E}}\ \underset{a\sim\pi_0(\,\cdot\,\mid\mathcal{A}_0,x)}{\mathbb{E}}\left[\hat{\delta}(x,a)^2\right] - \left(\underset{(x,r)\sim\mathbb{D}}{\mathbb{E}}\ \underset{a\sim\pi_0(\,\cdot\,\mid\mathcal{A}_0,x)}{\mathbb{E}}\left[\hat{\delta}(x,a)\right]\right)^2 + \underset{x\sim\mathbb{D}}{\mathbb{V}}\ \underset{a'\sim\pi(\,\cdot\,\mid\mathcal{A}_1,x)}{\mathbb{E}}\left[h(x,a')\right] \\ &\leq \underset{(x,r)\sim\mathbb{D}}{\mathbb{E}}\ \underset{a\sim\pi_0(\,\cdot\,\mid\mathcal{A}_0,x)}{\mathbb{E}}\left[\hat{\delta}(x,a)^2\right] + \underset{x\sim\mathbb{D}}{\mathbb{V}}\ \underset{a'\sim\pi(\,\cdot\,\mid\mathcal{A}_1,x)}{\mathbb{E}}\left[h(x,a')\right]. \end{aligned}$$

In the final step, we upper bound the variance by removing the middle squared term due to its non-negativity, which completes the proof.

A.3 Proof of Corollary 1

We will proceed by splitting the expectation in Equation 11 into two terms, so that we can analyze each in isolation. We then re-combine them to obtain the final expressions.

Starting with the lefthand expectation:

$$\begin{aligned}
\mathop{\mathbb{E}}_{m\sim\bar{\pi}(\,\cdot\,|\,\mathcal{A}_1,x)}\;\mathop{\mathbb{E}}_{a\sim\pi_0(\,\cdot\,|\,m\cap\mathcal{A}_0(x),x)}\big[\delta(x,a)\big] &= \sum_{m\in\mathcal{M}(x)}\;\sum_{a\in m\cap\mathcal{A}_0(x)}\bar{\pi}(m\,|\,\mathcal{A}_1,x)\pi_0(a\,|\,m\cap\mathcal{A}_0(x),x)\delta(x,a)\\
&= \sum_{a'\in\mathcal{A}_0(x)}\;\sum_{a\in\phi(a',x)}\bar{\pi}(a'\,|\,\mathcal{A}_1,x)\pi_0(a\,|\,\phi(a',x)\cap\mathcal{A}_0,x)\delta(x,a)\\
&= \sum_{a'\in\mathcal{A}_0(x)}\bar{\pi}(a'\,|\,\mathcal{A}_1,x)\pi_0(a'\,|\,\{a'\},x)\delta(x,a')\\
&= \sum_{a'\in\mathcal{A}_0(x)}\bar{\pi}(a'\,|\,\mathcal{A}_1,x)\delta(x,a')\\
&= \sum_{a'\in\mathcal{A}_0(x)}\;\sum_{a\in\mathcal{A}_1(x)}\pi(a\,|\,\mathcal{A}_1,x)\mathbb{1}\{a'=\phi(x,a)\}\delta(x,a')\\
&= \sum_{a\in\mathcal{A}_1(x)}\pi(a\,|\,\mathcal{A}_1,x)\delta(x,\phi(x,a))\\
&= \mathop{\mathbb{E}}_{a\sim\pi(\,\cdot\,|\,\mathcal{A}_1,x)}\big[\delta(x,\phi(x,a))\big].
\end{aligned}$$

Turning now to the righthand expectation, we can simply reverse Equation 19:

$$\mathop{\mathbb{E}}_{m\sim\bar{\pi}(\,\cdot\,|\,\mathcal{A}_1,x)}\;\mathop{\mathbb{E}}_{a'\sim\pi(\,\cdot\,|\,m\cap\mathcal{A}_1(x),x)}\big[\delta(x,a')\big] = \mathop{\mathbb{E}}_{a'\sim\pi(\,\cdot\,|\,\mathcal{A}_1,x)}\big[\delta(x,a')\big].$$

Finally, substituting the above identities into Equation 11, we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{m\sim\pi(\,\cdot\,|\,\mathcal{A}_1,x)}\;\mathop{\mathbb{E}}_{\substack{a\sim\pi_0(\,\cdot\,|\,m\cap\mathcal{A}_0(x),x)\\ a'\sim\pi(\,\cdot\,|\,m\cap\mathcal{A}_1(x),x)}}\Big[\delta(x,a)-\delta(x,a')\Big] &= \mathop{\mathbb{E}}_{a\sim\pi(\,\cdot\,|\,\mathcal{A}_1,x)}\Big[\delta(x,\phi(x,a))-\delta(x,a)\Big]\\
&= \mathop{\mathbb{E}}_{a\sim\pi(\,\cdot\,|\,\mathcal{A}_1,x)}\Big[\mathbb{1}\{a\in\mathcal{A}_{1\setminus 0}(x)\}\Big(\delta(x,\phi(x,a))-\delta(x,a)\Big)\Big].
\end{aligned}$$

The last equality holds because, for any $a\in\mathcal{A}_{0\cap 1}(x)$—that is, an old action that is also a member of $\mathcal{A}_1(x)$—its proxy is itself; hence, $\delta(x,\phi(x,a))-\delta(x,a)=0$, leaving only the actions in $\mathcal{A}_{1\setminus 0}(x)$ to influence the bias.