Joint Policy-Value Learning for Recommendation

Olivier Jeunen
Adrem Data Lab, University of Antwerp
olivier.jeunen@uantwerp.be

David Rohde, Flavian Vasile, Martin Bompaire
Criteo AI Lab
{d.rohde,f.vasile,m.bompaire}@criteo.com

ABSTRACT

Conventional approaches to recommendation often do not explicitly take into account information on previously shown recommendations and their recorded responses. One reason is that, since we do not know the outcome of actions the system did not take, learning directly from such logs is not a straightforward task. Several methods for off-policy or counterfactual learning have been proposed in recent years, but their efficacy for the recommendation task remains understudied. Due to the limitations of offline datasets and the lack of access of most academic researchers to online experiments, this is a non-trivial task. Simulation environments can provide a reproducible solution to this problem.

In this work, we conduct the first broad empirical study of counterfactual learning methods for recommendation, in a simulated environment. We consider various different policy-based methods that make use of the Inverse Propensity Score (IPS) to perform Counterfactual Risk Minimisation (CRM), as well as value-based methods based on Maximum Likelihood Estimation (MLE). We highlight how existing off-policy learning methods fail due to stochastic and sparse rewards, and show how a logarithmic variant of the traditional IPS estimator can solve these issues, whilst convexifying the objective and thus facilitating its optimisation. Additionally, under certain assumptions the value- and policy-based methods have an identical parameterisation, allowing us to propose a new model that combines both the MLE and CRM objectives. Extensive experiments show that this "Dual Bandit" approach achieves state-of-the-art performance in a wide range of scenarios, for varying logging policies, action spaces and training sample sizes.

ACM Reference Format:
Olivier Jeunen, David Rohde, Flavian Vasile and Martin Bompaire. 2020. Joint Policy-Value Learning for Recommendation. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '20), August 23–27, 2020, Virtual Event, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3394486.3403175

1 INTRODUCTION

Traditional approaches to recommendation are often based on some form of collaborative filtering on the user-item matrix containing *organic* user-item interactions [24, 38, 42]. These are generally user-item-timestamp triplets, indicating item purchases, clicks, views, et cetera. From rating- to next-item-prediction, such methods have known widespread success [41]. Generally, they are oblivious to whether actual recommendations were being shown to users in the data they learn from. In a parallel research direction, computational advertising applications often frame recommendation as an optimal *decision-making* problem, where the learning step aims to build an explicit reward model for all (user, recommendation)-pairs and the inference step chooses the best recommendation for the user given the learnt model. Existing work on modelling the probability of clicking on recommendations falls into this class, and publications on the subject traditionally originate from advertising research labs (see [28] for an overview). These approaches focus on *bandit* feedback: interactions between users and recommendations being shown. Therefore, this data is conditioned on the policy describing the existing recommender system.

For both existing frameworks, the majority of new recommendation algorithms presented in academic papers are evaluated on an offline dataset of logged user-item interactions, with results reported for some offline ranking metric. Recent work has shown repeatedly that offline evaluation results tend to diverge from online performance [8, 14, 35]. Additionally, existing offline evaluation results are often even contradictory over different runs and datasets, or extremely hard to reproduce in a robust manner [5, 32]. From a practical or industrial point of view, a need arises for offline evaluation methods that are robust, reproducible and closely related with the actual online objectives of the deployed recommender system.

The reinforcement learning literature has long dealt with similar issues. Based on logged data from a certain *policy* (recommender), we want to predict what the performance would have been if another policy had been deployed. Counterfactual estimators, often based on importance sampling [30], are at the heart of this type of evaluation. Recent work has shown that they can accurately reflect online performance in recommendation use-cases [9, 10].

We believe that the main reason why the value of bandit feedback is not further explored in most recommendation research, lies in the datasets we use: the vast majority of available offline datasets simply do not include information about the recommendations that were shown, and whether the user interacted with them. Naturally, we cannot learn from what we do not know. Some datasets containing logs of historical recommendations and their outcomes do exist, mostly for the specific task of click-through-rate (CTR) prediction. Nevertheless, they either do not include propensity scores produced through adding randomisation at recommendation time, which is often a requirement for unbiased learning and evaluation [2, 9], or the variance induced by the propensities prohibits effective learning and evaluation [20]. Recently, several simulation environments have been proposed for the recommendation setting, allowing online experiments such as A/B-tests to be simulated [12, 34]. They enable us to explore the use of bandit feedback for learning and evaluation of recommender systems in a reproducible manner, and have opened up promising new research directions. The value of such simulation frameworks has steadily gained more attention in recent research [16, 29, 36].

KDD '20, August 23–27, 2020, Virtual Event, CA, USA

Moving on from evaluation, recent work [2, 15, 17, 27, 47, 48] explores the problem of learning a new and optimal recommendation policy based on a dataset consisting of logged bandit feedback. However, real-world recommender systems tend to differ from the assumptions made in the existing work that studies the use of bandit feedback for the problem off-policy learning or Counterfactual Risk Minimization (CRM), in terms of the stochasticity and sparsity of rewards. Furthermore, the size of the action space in real-world scenarios makes the choice of problem formulations that result in convex objectives very attractive (due to the existence of mature fast large-scale optimisation algorithms, such as L-BFGS [19]). The technical contributions we present in this paper are the following:

(1) *Analysis of the Convex Policy Lower-bound for Stochastic Rewards.* We show where existing off-policy learning approaches fall short, especially due to the inherent stochasticity and sparsity of the reward process. We introduce a logarithmic variant of the traditional Inverse Propensity Scoring (IPS) estimator that allows us to map the objective to a convex problem, yielding a weighted multinomial log-likelihood that lower bounds the original. Recent related work has introduced a similar objective as a Policy Improvement Lower-bound (PIL) and a variance reduction technique [27]; we focus on its practicality in stochastic environments and empirically show how it can improve performance. As the logarithmic transformation convexifies the objective, it facilitates its optimisation.

(2) *Joint Policy-Value Optimisation.* By presenting value- and policy-based methods in a common framework, we show that they have an identical parameterisation under certain assumptions. This allows us to propose *Dual Bandit (DB)*, a hybrid learning objective that combines both Maximum Likelihood Estimation (MLE) and Counterfactual Risk Minimisation (CRM) without introducing additional parameters, effectively unifying the value- and policy-based families. We show how it effectively alleviates well-known problems such as propensity overfitting. Moreover, standard off-policy learning methods do not take into account negative evidence (i.e. non-clicked recommendations), which is solved through the value-based cross-entropy term. Finally, we show that the DB approach achieves state-of-the-art performance in a wide range of recommendation settings, for varying logging policies, training sample and action space sizes. The most interesting and realistic results deal with large action spaces, finite samples, and limited randomisation, which is exactly where the Dual Bandit shows its superiority.

(3) *Reproducible Simulation Study.* The empirical performance of counterfactual learning methods has mainly been studied in multi-class [17, 27, 47] and multi-label [48] classification environments where the bandit setting is simulated. Recent work that focuses on the recommendation use-case adopts a supervised-to-bandit conversion on existing datasets and custom simulated datasets that assume deterministic rewards [26], or shows empirical success through live experiments [4]. We conduct the first broad simulation study of both value-based methods based on MLE, and policy-based methods that rely on IPS to perform CRM in stochastic recommendation environments. In order to aid in the reproducibility of the research presented in our work, we adopt the RecoGym environment in our experiments [34]. All source code is available at https://github.com/olivierjeunen/dual-bandit-kdd-2020.

2 BACKGROUND AND RELATED WORK

In what follows, we present an overview of the methods we consider in our comparison. As we focus on so-called *bandit feedback*, these methods make use of action-reward pairs: recommendations that were shown and whether they were interacted with. We discern two broad families: value- and policy-based methods. The first aims to model the reward a certain action will yield, relying on classical supervised learning approaches [11]. The latter directly models the actions that should be taken in order to maximise the total cumulative reward a policy will collect. This line of research is more closely related to the reinforcement learning (RL) field [45]. Value-based models and their variations are often referred to as Q-learning in the RL community.

We assume to have access to a dataset of logged bandit feedback $\mathcal{D}$, consisting of N tuples $(\mathbf{x}_i, a_i, p_i, c_i)$. This data has been collected under some stochastic logging policy π_0 that describes a probability distribution over actions (*recommendations*), conditioned on some context. Here, $\mathbf{x}_i \in \mathbb{R}^n$ describes the user state or context vector. Although this vector can be of arbitrary dimension, we will assume it to be a vector of length n containing counts of historical *organic* interactions with items for fair comparison and simplicity. $a_i \in \{1, 2, \ldots, n\}$ is a scalar identifier representing the action that was taken (i.e. the item that was shown when the system was presented with context $\mathbf{x}_i$), we denote the corresponding one-hot encoded vector as $\mathbf{a}_i$. The probability with which that action was taken by the logging policy is denoted by $p_i \equiv \pi_0(a_i|\mathbf{x}_i) \in [0, 1]$. The observed reward (whether the user interacted with the presented recommendation) is represented as $c_i \in \{0, 1\}$.

2.1 Value-based Approaches

The most straightforward method is to first perform statistical inference through Maximum Likelihood Estimation (MLE) and then do decision making in a separate step, bypassing the empirical/counterfactual risk minimisation principles. If the reward is a binary variable, then a logistic regression model is appropriate, as shown in Equation 1. This formulation can be naturally extended to include more advanced likelihood-based models such as deep neural networks.

$$P(C = 1|\mathbf{x}, a, \boldsymbol{\beta}) = \sigma((\mathbf{x} \otimes \mathbf{a})^\intercal \operatorname{vec}(\boldsymbol{\beta}) + b) = \sigma(\mathbf{x}^\intercal \boldsymbol{\beta}_{\cdot,a} + b) \quad (1)$$

Here, $\boldsymbol{\beta} \in \mathbb{R}^{n \times n}$ are the model parameters where $\boldsymbol{\beta}_{\cdot,a}$ are those corresponding to predictions for action a. $\sigma(\cdot)$ is the logistic sigmoid, $\otimes$ is the Kronecker product and $\operatorname{vec}(\cdot)$ vectorises the matrix into a column vector.[1] The intercept or bias-term is denoted by b. As it is a single constant scalar for all context-action pairs, it does not have any impact on the ranking of competing actions or the actual decision making process. Nevertheless, it positively impacts the quality of the fitted model [11]. We ensure fair comparison with the other approaches: all consist of exactly n^2 parameters that impact the resulting decision rule (excluding hyper-parameters, that is). Optimising the binary cross-entropy or negative log-likelihood of this model with respect to a historical dataset yields the objective shown in Equation 2.

[1] We implement the rightmost formula as it is equivalent to, but computationally significantly less expensive than explicitly computing the Kronecker product.

Family	Method	$P(C\|x,a)$	$P(A\|x)$	Neg.	IPS	SVP	IML	Equivariant	Ref.
Value learning	Likelihood	✓		✓					[11]
	IPS Likelihood	✓		✓	✓				[43]
Policy learning	Contextual Bandit		✓		✓				[2]
	POEM		✓		✓	✓			[47]
	BanditNet		✓	~	✓			✓	[17]
	PIL-IML		✓		✓		✓		[27]
Joint learning	Dual Bandit	✓	✓	✓	✓				This work.

Table 1: An overview of the methods we discuss in this paper, and how they relate to one another in terms of the target they optimise and whether they exploit negative samples, make use of Inverse Propensity Scoring (IPS), Sample Variance Penalisation (SVP), an Imitation Learning (IML) term, or whether the approach is invariant to translations of the reward.

$$\boldsymbol{\beta}^* = \arg\max_{\boldsymbol{\beta}} \sum_{i=1}^{N} c_i \ln\left(\sigma(\mathbf{x}_i^\intercal \boldsymbol{\beta}_{\cdot,a_i} + b)\right) + (1-c_i)\ln\left(1-\sigma(\mathbf{x}_i^\intercal \boldsymbol{\beta}_{\cdot,a_i} + b)\right) \tag{2}$$

Once the model parameters have been fitted, we can obtain a greedy decision for a given context $\mathbf{x}$ by performing the action a^* with the highest probability of leading to a positive reward. Naturally, this requires n model evaluations followed by an argmax operation:

$$a^* = \arg\max_a P(C=1|\mathbf{x}, a, \boldsymbol{\beta}) = \arg\max_a \mathbf{x}^\intercal \boldsymbol{\beta}_{\cdot,a}. \tag{3}$$

We often fit simpler (for example, linear) models to capture more complex relationships. When doing this, the model *underfits* the data as it is unable to capture the true relationship. When a standard MLE approach is used, the error due to the underfitting will be minimised around common occurrences of $(\mathbf{x}, a)$. If the distribution of $(\mathbf{x}, a)$-pairs in the historical training data differs from those in the test set, this leads to a phenomenon widely known as *covariate shift* [39]. As we generally wish to learn a new policy that improves upon the logging policy, it will take different actions by definition, and covariate shift is inevitable.

One general solution to this issue is to make use of importance sampling [30], and reweight samples to adjust for the difference in the distribution of past actions (as per the logging policy) and future actions (which we will evaluate uniformly in this case, as that is exactly what we do with the argmax-operation over actions in Eq. 3). Practically, this is achieved by reweighting samples $(\mathbf{x}_i, a_i, c_i)$ in Equation 2 by the inverse propensity score of the logging policy during maximum likelihood estimation: $w_i = \frac{1}{\pi_0(a_i|\mathbf{x}_i)}$.

Value-based methods estimate the likely reward for each action. Due to the logging policy, the quality of this estimate can vary dramatically. The estimation uncertainty will be low for actions performed often by the logging policy, but poor otherwise. As the action is selected by selecting the maximum value, the policy may be disrupted badly by a single erroneously optimistic estimate; a phenomenon known as optimiser's curse [40]. In recommender systems, the action space can be large and the size of the training sample and amount of randomisation are often limited. Because of this, these problems are very tangible in a real-world setting.

2.2 Policy-based Approaches

Value-based methods model the probability of a click (*value*), given a context-action pair. Policy-based methods, on the other hand, bypass this step and directly map a context to a decision rule.

Contextual bandits are one example of such a method, directly modelling the probability of an action, conditioned on a context vector. This is shown in Equation 4, where $\boldsymbol{\theta} \in \mathbb{R}^{n\times n}$ are the model parameters.

$$P(A=a|\mathbf{x}, \boldsymbol{\theta}) = \pi_{\boldsymbol{\theta}}(a|\mathbf{x}) = \frac{\exp(\mathbf{x}^\intercal \boldsymbol{\theta}_{\cdot,a})}{\sum_{j=1}^{n} \exp(\mathbf{x}^\intercal \boldsymbol{\theta}_{\cdot,a_j})} = [\text{softmax}(\mathbf{x}^\intercal \boldsymbol{\theta})]_a \tag{4}$$

Consistent with the work of Bottou et al. [2] and Swaminathan and Joachims [47], this formulation focuses on learning policies that are parametrised as exponential models (i.e. going through a softmax). The goal at hand is to learn a policy that chooses the optimal action given a context $\mathbf{x}$, i.e. the policy that maximises the reward we would have gotten when $\pi_{\boldsymbol{\theta}}$ was deployed instead of the logging policy π_0. In our setting, this reward can be interpreted as the absolute number of clicks. Equation 5 formalises this counterfactual objective, which can be optimised directly. Essentially, contextual bandits try to replay decisions that worked in the training sample. From a learned stochastic policy $\pi_{\boldsymbol{\theta}}$, a deterministic decision rule can easily be deduced as shown in Equation 6.

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{N} c_i \frac{\pi_{\boldsymbol{\theta}}(a_i|\mathbf{x_i})}{\pi_0(a_i|\mathbf{x_i})} \tag{5}$$

$$a^* = \arg\max_a P(A=a|\mathbf{x}, \boldsymbol{\theta}) = \arg\max_a \mathbf{x}^\intercal \boldsymbol{\theta}_{\cdot,a} \tag{6}$$

Note that if we let $\boldsymbol{\beta} \equiv \boldsymbol{\theta}$, the decision rule in Equation 3 is identical to the decision rule in Equation 6. Although the optimisation problems given in Equations 1 and 4 are quite different: maximum likelihood ignores the IPS score, and the contextual bandit ignores the non-clicks. The likelihood-based approach attempts to model the click for every action, whereas the contextual bandit simply attempts to identify the best action. This common parameterisation will motivate the "Dual Bandit" method we present later in this work, a principled approach to jointly optimise these two objectives.

Inverse Propensity Scoring (IPS) is a powerful technique that allows for counterfactual optimisation. When the target policy $\pi_{\boldsymbol{\theta}}$ and the logging policy π_0 diverge, however, IPS-based estimators tend to have very high variance and be unreliable for policy

learning. Several extensions to the classical estimator have been proposed, trading variance for bias. Clipping the propensity ratios to a maximum value or self-normalising them are common practices. The most notable recent extensions to this formulation are POEM [47], BanditNet [17] and PIL-IML [27]. They respectively include additional terms for sample variance penalisation (SVP), self-normalisation (SNIPS) or imitation learning (IML). Conceptually, they all restrict the newly learned policy to not stray too far from the logging policy, as the uncertainty on rare actions brings along high variance on the performance. Originally introduced by Swaminathan and Joachims, we refer to these approaches as performing some form of Counterfactual Risk Minimisation (CRM).

Recent related work has linked IPS techniques to policy gradient methods such as REINFORCE [50]. These approaches aim to maximise the expected cumulative reward over a certain time horizon, and were extended to handle specific cases such as top-K recommendation [4] and two-stage recommendation pipelines [26]. For the binary, immediate reward use-case we tackle in this paper, maximising the expected reward via REINFORCE yields the same result (that is the same policy) as optimising the IPS objective given in Eq. 5. Ma et al. draw further connections between their objective and natural policy gradients; we refer the interested reader to Appendix E in their work [27]. Table 1 shows an overview of the discussed methods.

Several additional counterfactual estimators have been proposed in the literature, such as the Direct Method (DM) and Doubly Robust (DR) [6], More Robust Doubly Robust (MRDR) [7], Continuous Adaptive Blending (CAB) [44] and others [49, 52]. These works either focus on evaluation instead of learning (as they are non-differentiable), or they include an additional regression model that estimates the reward $\delta(\mathbf{x}, a)$ for a given context-action pair. As such, they are out of scope for this study. Additionally, contextual and multi-arm bandit approaches with online model updates have been thoroughly studied. Some notable approaches are [3, 22, 23]. In contrast with these methods, the models we discuss in this work are entirely *off-policy* and lack any interactive component.

As we will discuss in depth in the following section, recommender system logs are stochastic in nature. Probabilistic models propose a way of naturally handling uncertainty. When sophisticated priors are used, Bayesian methods can perform well in modelling complex relationships with small samples. For the related recommendation task of rating prediction, Bayesian approaches have recently been shown to robustly obtain state-of-the-art performance [33, 37]. In the case of top-N recommendation from organic user-item interaction data, recent variational methods consistently attain impressive results as well [5, 24, 38]. Sakhi et al. propose a probabilistic approach to combine both organic and bandit signals in order to improve the estimation of recommendation quality in a Bayesian latent factor model [36]. Further exploring such models and their applicability in off-policy recommendation settings is a promising avenue for future research.

3 LEARNING FOR RECOMMENDATION

The performance of most methods discussed in this work has been empirically validated for multi-class [17, 27, 47] or multi-label [48] classification environments, which simulate a "Batch-Learning from Bandit-Feedback" (BLBF) context. The policy performs an action (guesses a class or label), and observes the reward (the guess is either correct or incorrect). Other work has adopted these supervised-to-bandit conversions to mimic the recommendation task as well [26]. Assuming we have more than one item in the catalogue that is of interest to the user, the recommendation use-case is indeed most closely aligned with the multi-label setup. Several key differences remain, which we tackle throughout the rest of this section:

(1) *Stochastic vs Deterministic Rewards.* Previous work has always assumed deterministic rewards: if a model makes decision a when presented with context $\mathbf{x}$, the observed reward c will always be exactly the same. This assumption does not hold in real-world recommendation settings: users may click (or conversely, refuse to click) on shown recommendations for any number of reasons. As it is intractable for all factors on which the reward c is dependent to be included in the context-vector $\mathbf{x}$, the resulting model will always be misspecified, and rewards stochastic as a result.

(2) *Sparse Rewards and Low Treatment Effects.* CTR measurements in real-world systems can be notoriously low, skewing the estimates. When the training log $\mathcal{D}$ contains tuples for the same context-action pairs with different observed rewards, however, the information embedded in the clicks is often more valuable than the non-clicks, although the latter will make up the vast majority of the logged samples. Furthermore, the difference between the empirical *best* and *second best* arms might not always be large. As such, solely focusing on the empirically best action is not an optimal strategy.

(3) *Large Action Spaces.* Recommender systems typically deal with vast item catalogues. Previous experimental validation of these methods has focused on setups where the number of *actions* (classes, labels, documents, …) is at most a few dozen. Their practicality in very large action spaces remains understudied.

3.1 Logarithmic IPS for Stochastic Rewards

Rationally, we could assume the true reward to be drawn from some Bernoulli-distribution with parameter q relative to the relevance of the taken action; but relevant actions will not always lead to clicks, and a click does not necessarily imply that the performed action was the most relevant option. Estimating this parameter q is an indubitably harder task than the deterministic case, requiring larger training samples and leaving us vulnerable for common pitfalls such as the optimiser's curse [40].

The counterfactual objective function in Equation 5 describes an empirical IPS-estimate of the reward that $\pi_{\boldsymbol{\theta}}$ would incur based on data collected under π_0, given a historical dataset $\mathcal{D}$. We denote this estimator as $\hat{R}_{\text{IPS}}(\pi_{\boldsymbol{\theta}}, \mathcal{D})$, and drop parentheses and arguments when they are clear from context.

$$\hat{R}_{\text{IPS}}(\pi_{\boldsymbol{\theta}}, \mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} c_i \frac{\pi_{\boldsymbol{\theta}}(a_i|\mathbf{x}_i)}{\pi_0(a_i|\mathbf{x}_i)} \tag{7}$$

This objective leads to a "*winner takes it all*" scenario, where the optimal policy puts all its mass on the actions that obtained the highest empirical reward in the finite training sample. We argue that this can lead to sub-optimal policies in the stochastic scenario, as positive samples for the empirically "*second best*" actions are simply ignored. This behaviour has recently been studied in top-K recommendation scenarios [4]; we focus on top-1 recommendation.

In supervised learning, it is common practice to optimise a surrogate loss function instead of a direct metric (hard classification error, for example). Through Jensen's inequality, we can derive a lower bound of the traditional IPS estimator ($\hat{R}_{\text{IPS}}$) that uses a logarithmic transform on the likelihood.[2] Intuitively, this is equivalent to optimising the log-likelihood of a multinomial logistic regression model where each observation has been weighted by $w_i = \frac{c_i}{\pi_0(a_i|\mathbf{x}_i)}$. We refer to this logarithmic estimator as $\hat{R}_{\ln(\text{IPS})}$ and present it in Equation 8. Recent related work introduced this equivalently as a policy improvement lower-bound [25, 27]. These works primarily focus on model misspecification and multiple iterations of logging and learning, whereas we study the impact of the transformation in stochastic environments.[3]

$$\hat{R}_{\ln(\text{IPS})}(\pi_{\boldsymbol{\theta}}, \mathcal{D}) = \frac{1}{N}\sum_{i=1}^{N} c_i \frac{\ln(\pi_{\boldsymbol{\theta}}(a_i|\mathbf{x}_i))}{\pi_0(a_i|\mathbf{x}_i)} \tag{8}$$

In combination with the exponential parameterisation in Equation 4, this logarithmic objective leads to a *proportional* allocation of probability mass. It can be readily plugged into existing policy learning methods described in Section 2. As it forces the model to include positive samples for all actions instead of the empirically best action only ($\lim_{P\to 0} \ln(P) = -\infty$), we expect and empirically observe that this leads to (1) less overfitting, and (2) more robust performance. Furthermore, computations in the log-space improve the numerical stability of the optimisation procedure. Negative log-likelihood or cross-entropy (as also used in Equation 2 and widely adopted in classification and deep learning) is commonly used in these settings, as it transforms the objective to be convex, and thus easy to optimise at large scale.

In policy gradient methods such as REINFORCE, a common way to optimise the objective consists of performing a gradient step involving the logarithm of the policy [46]. Also called the "log-trick" or "log derivative trick", this technique is by nature very different from ours. Indeed, the "log derivative trick" is a way to compute an unbiased estimate of the gradient of the expected reward under the new policy. Hence, using this trick does not change the optimised objective. In our case, the logarithmic transform *changes* the objective and moves the global optimum. Figure 1 visualises how this transforms the objective when the reward is stochastic, or the model misspecified.

3.2 Joint Policy-Value Optimisation

Policy-based approaches can suffer from so-called propensity overfitting, where the learning objective is trivially optimised by missing the observed data [48]. To see this, suppose we would transform Eq. 5 to a minimisation of non-clicks instead of a maximisation of clicks. The result of this transformation would be that putting 0 probability mass on all observed $\pi_0(a|\mathbf{x})$ trivially minimises the objective, although it would clearly not lead to better generalisation capabilities. Indeed, learning to avoid actions does not imply learning which actions to take. These types of issues are generally avoided through the use of a SNIPS estimator, which includes a multiplicative control variate in the IPS estimator that heavily penalises this behaviour. BanditNet optimises a Lagrangian form of this estimator and achieved promising results in a deterministic multi-class bandit setting [17]. By introducing translations to the binary reward, it opens up opportunities for learning from *negative* samples; as can be seen from Eq. 5, probabilities for 0-reward actions ($c_i = 0$) have no direct impact on the objective. However, as the optimal Lagrange multiplier γ could be 0, this is not a guarantee; which is exactly what we observed in our empirical results.

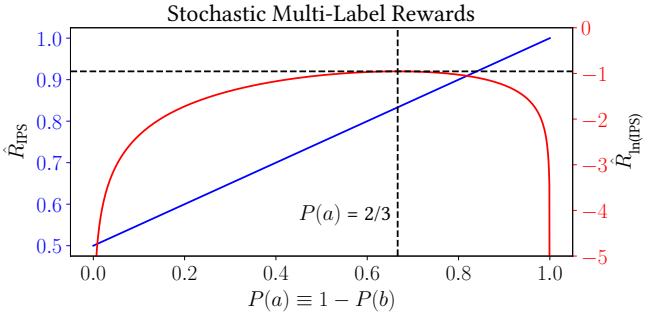

Figure 1: $\hat{R}_{\ln(\text{IPS})}$ penalises the learned policy for *missing* actions that led to positive rewards in the training sample. Maximising this empirical estimator for a toy example training sample with 2 clicks on action a and 1 click on action b leads to more proportional allocation compared to $\hat{R}_{\text{IPS}}$ (assuming a uniform π_0). For deterministic multi-class or multi-label settings, both estimators share optima.

Another way of limiting the issue of propensity overfitting is by introducing an IML term to the objective, as done in PIL-IML [27]. By penalising the objective with the Kullback-Leibler divergence between the learned and logging policies, it effectively favours those policies that *imitate* π_0. In most of their experimental setup, π_0 is a value-based linear model. Therefore, π_0 holds information on the non-clicked training samples and learning to imitate π_0 indirectly transfers this signal to the learned policy. It is clear to see that the quality of the policy learned through such an optimisation procedure is highly dependent on the quality of π_0, and that it might have adverse effects for highly skewed logging policies (e.g. an ϵ-greedy based approach).

As pointed out in Section 2, value- and policy-based methods can be formulated such that they have an identical parameterisation (Equations 6 and 3). This allows us to present a combined MLE-CRM objective that optimises a weighted average of the two. Due to the softmax-formulation from CRM, the model will be less prone to over-estimate under-explored actions. The logistic likelihood term can be seen as a regularisation term that ensures the dot-product between $\mathbf{x}$ and $\boldsymbol{\theta}_{\cdot,a}$ is low for un-clicked $(\mathbf{x}, a)$ samples in the training data, information that standard policy-based methods fail to exploit. We call this joint objective "Dual Bandit", as it provides a principled way to combine the best of the value- and policy-based worlds, whilst alleviating their individual weaknesses. Equation 9 shows the novel objective, where $0 \leq \alpha \leq 1$ is a hyperparameter that controls the influence of the log-likelihood on the final estimate. As such, both the contextual bandit and likelihood approaches can be seen as special cases of the dual bandit model, for $\alpha = 0$ and $\alpha = 1$ respectively. The contextual bandit aims to learn a probability

[2] A derivation can be found in the reproducibility appendix.
[3] See [27] and its appendices to link the logarithmic estimator to existing variance reduction techniques such as capped IPS, SVP, SNIPS, and others.

distribution over n items, and its objective can be interpreted as the expected number of clicks under the new policy. The likelihood approach aims to learn whether an action led to a click, where the objective reflects whether the training sample supports the model parameters. Due to this disparity, α will act as a rescaling factor on top of a blending parameter. Keeping this in mind, higher values of α don't map linearly to higher importance of the likelihood approach compared with the contextual bandit.

$$\begin{aligned}\boldsymbol{\theta}^* &= \arg\max_{\boldsymbol{\theta}}(1-\alpha)\left(\sum_{i=1}^{N} c_i \frac{P(A=a_i|\mathbf{x}_i,\boldsymbol{\theta})}{\pi_0(a_i|\mathbf{x}_i)}\right) \\ &+\alpha\left(\sum_{i=1}^{N} c_i \ln\left(P(C=1|\mathbf{x}_i,a_i,\boldsymbol{\theta})\right) + (1-c_i)\ln\left(1-P(C=1|\mathbf{x}_i,a_i,\boldsymbol{\theta})\right)\right) \\ &= \arg\max_{\boldsymbol{\theta}}(1-\alpha)\left(\sum_{i=1}^{N} c_i \frac{[\text{softmax}(\mathbf{x}_i^\intercal\boldsymbol{\theta})]_{a_i}}{\pi_0(a_i|\mathbf{x}_i)}\right) \\ &+\alpha\left(\sum_{i=1}^{N} c_i \ln\left(\sigma(\mathbf{x}_i^\intercal\boldsymbol{\theta}_{\cdot,a_i}+b)\right) + (1-c_i)\ln\left(\sigma(\mathbf{x}_i^\intercal\boldsymbol{\theta}_{\cdot,a_i}+b)\right)\right)\end{aligned} \tag{9}$$

As minimising $\pi_{\boldsymbol{\theta}}(a|\mathbf{x})$ for all seen $(\mathbf{x}, a)$-pairs will negatively impact the cross-entropy until it dominates the objective, this solves propensity overfitting as well. Unlike the NormPOEM-objective introduced by Swaminathan and Joachims [48], our objective decomposes into a sum over the observed samples. Because of this, it is perfectly suitable for optimisation through stochastic methods such as SGD and, as a consequence, applicable to large-scale problems and training deep neural networks. In this sense it is similar to BanditNet [17], but the Dual Bandit formulation additionally guarantees the inclusion of un-clicked samples in the model. Naturally, the IPS estimator in the first term can be replaced by the logarithmic IPS estimator introduced in Equation 8, or any other counterfactual estimator provided they are continuous and differentiable. SVP or IML terms could be naturally included in the objective, but our experiments show that the Dual Bandit is already highly competitive without them.

4 EXPERIMENTAL RESULTS

In what follows, we experimentally validate the efficacy of counterfactual learning approaches presented in previous and this work, with a focus on the task of recommendation. In order to evaluate these approaches effectively, we need a dataset containing logged feedback for context-action pairs, along with the logging propensity for the action performed. Related work has evaluated counterfactual learning methods on multi-class, multi-label or LTR tasks [13, 17, 47, 48], synthetically generating bandit feedback samples for a certain logging policy and existing datasets. What makes the recommendation task fundamentally different from the aforementioned settings, is that access to the true labels (i.e. how likely a user is to click on a given recommendation) becomes impractical, and effective offline evaluation thus much less straightforward. Recent work that focuses on the recommendation use-case adopts a supervised-to-bandit conversion on existing datasets, and custom simulated datasets that assume deterministic rewards [26], or shows empirical success through live experiments [4]. In order to aid in the reproducibility of the research presented in our work, we adopt the RecoGym simulation environment in our experiments [34]. RecoGym provides functionality to generate offline logs under a given logging policy (for training and/or evaluation), and allows for the opportunity to simulate online experiments such as A/B-tests. More information regarding the specific setup of our experiments can be found in the reproducibility appendix. All the methods discussed throughout this work are optimised with the full-batch L-BFGS algorithm, in order to avoid the choice of optimiser to be a confounding factor. Although some of the objectives presented in this work are non-convex and non-smooth, the choice of L-BFGS is theoretically well-supported [21, 51] and has been empirically shown to yield good performance in previous work [19, 48]. We aim to answer the following research questions:

RQ1 How does the logarithmic IPS (or PIL) estimator $\hat{R}_{\ln(\text{IPS})}$ impact existing off-policy learning methods?
RQ2 How do the various methods presented in this paper compare in terms of performance in a recommendation setting?
RQ3 How sensitive is the performance of the learned models with respect to the quality of the initial logging policy π_0?
RQ4 How do the number of items n and the number of available samples N influence performance?

4.1 Logging policies

We now describe the logging policies that we employ to generate logged bandit feedback samples that serve as the training datasets to our methods.

Uniform. The uniform logging policy chooses its actions uniformly at random. Thus, every item's probability of being recommended is $\pi_{\text{uniform}}(a|\mathbf{x}) = \frac{1}{n}$, independent of the context. As a consequence, IPS reweighting does not have any impact, because all the weights would be identical. Data logged uniformly at random contains no biases, and learning from it is a considerably easier task than otherwise. Nevertheless, real-world data will usually not be logged under this type of policy, as complete randomisation can significantly impair user satisfaction. It is an idealised and unrealistic setting, but it provides interesting insights nonetheless.

Popularity-based. A simple yet effective baseline policy is to sample actions with probabilities proportionate to the occurrence frequency of the item in the user's historical *organic* interactions. In a toy setting with 3 products and a user state of $[3, 1, 0]$, this means we sample item 1 with probability $\frac{3}{4}$, item 2 with probability $\frac{1}{4}$, and we don't sample item 3. In general, for a user history $\mathbf{x}$, $\pi_{\text{pop}}(a|\mathbf{x}) = \frac{\mathbf{x}_a}{\sum_{i=1}^{n}\mathbf{x}_i}$. This policy does not have full support over the item catalogue, violating the assumptions that guarantee importance sampling to yield an unbiased estimate [30]. As a consequence, learning an effective policy from data logged under such a policy can become problematic (especially for value-based methods). This can be mitigated by adopting an ϵ-greedy scheme: with probability ϵ, take an action uniformly at random; with probability $1-\epsilon$, sample from the original probability distribution. Clearly, when $\epsilon = 0$, this reverts to the original policy. Although learning from data logged under a policy that does not have full support over the item catalogue loses some theoretical guarantees, we believe it to be closer to a realistic environment. In many real-world use-cases, various items may be non-recommendable, due to recency, stock,

licensing, business rules, et cetera. Furthermore, as real-world item catalogues are usually vast, it is realistic to randomise over a smaller set of candidate items. To reflect these real-world constraints, we include both variants in our experiments.

4.2 Discussion

We simulate A/B-tests for varying amounts of training data logged under the different policies presented in the previous subsection, and report the approaches' attained CTR measurements in Figure 2. A more in-depth overview of our experimental setup can be found in the reproducibility appendix. The logging policy is denoted as π_0, an oracle policy that always performs the action with the highest probability of leading to a click is shown as π^*. While such a skyline is unattainable, it is interesting to analyse the *regret* of the competing methods. We do not show the horizontal skyline on the row corresponding to $n = 100$ as it skews the y-axis limit of the plots, but it occurs around 2.44%. All reported results are averaged out over multiple runs, and show the 95% confidence interval.

As discussed in Section 2, the "CB" objective is equivalent to a policy gradient method with a single-step horizon. "Log-CB" corresponds to the PIL-IML objective [27], without an accompanying IML term. As we don't suffer from propensity overfitting in this setup, and most logging policies are severely skewed, our hyperparameter tuning procedure showed the optimal weight for this term to be 0. This was also the case for the Lagrange multiplier λ from BanditNet [17], reverting it back to the CB formulation. "Log-POEM" is POEM with the logarithmic estimator, "DB" and "Log-DB" are the Dual Bandit objective for the traditional and lower-bound estimators respectively. Every column represents a different logging policy. From left to right, these plots can be interpreted as going from the "most" to "least" realistic settings. Every row represents a differently sized action space, going up to 100. Larger action spaces are very relevant to the recommendation use-case, and we wish to study the generalisability of our results to them in future work. From a scalability perspective, this would require several adaptations to be made to the model formalisation in Section 2.

Effects of the Convex Policy Lower-bound (RQ1). We observe that the logarithmically transformed estimator positively influences results for both the popularity-based and the uniform logging policy. Moreover, we observed much more stable learning behaviour over various runs when using $\hat{R}_{\ln(\text{IPS})}$. As we increase the number of users in the training data, the performance of methods trained using $\hat{R}_{\ln(\text{IPS})}$ consistently improves. Methods using $\hat{R}_{\text{IPS}}$ showed far less consistent behaviour, as well as more variance across runs. This is consistent with the findings from Ma et al., as they primarily use the logarithm as a variance reduction technique. Aside from that, the logged estimator forces the model to take positive samples for *all* actions into account, leading to less overfitting on solely the *best* empirical actions.

While still improving over the logging policy and showing consistent behaviour, $\hat{R}_{\ln(\text{IPS})}$ hurts performance for the ϵ-greedy logging policy. Because it does not allow a single clicked sample to be missed, it is all the more sensitive to clicks on rare actions that might actually be suboptimal recommendations. Recent work on addressing click noise due to trust bias might provide a way of handling noisy training data in policy learning too [1].

Our experiments deal with the setting where the logging propensities $\pi_0(a|x)$ are known and exact. As a consequence, there isn't always a need to be conservative, which is what the convex lower bound does. Related work addresses settings where the logging propensities are unknown, learning an approximate $\hat{\pi_0}$ alongside their new policy [4]. In combination with highly stochastic rewards and small treatment effects, distinguishing the empirically optimal action becomes even more troublesome and noisy (see Fig. 1, for larger sample sizes, small variations on $\hat{\pi_0}$ may entirely flip the optimum). We expect the conservative logarithmic estimator to prove its worth even further in these settings.

Performance comparison (RQ2-4). Key observations from these results are the following: (1) In virtually all settings, the Dual Bandit approach achieves the best performance. Gains are the most tangible in cases where the logging policy does not randomise over the entire action space, as this is where classical value-based methods tend to fail. Policy-based methods can still improve upon the logging policy in these cases, but their performance does not seem to greatly improve with the size of the training sample. The Dual Bandit exhibits significant benefits over solely using value- or policy-based approaches. This suggests these families are complementary, and capture different relationships from the data. (2) As the logging policy randomises more uniformly, the performance of competing methods tends to converge. A logging policy with support over the entire action space positively influences the performance of the value-based approaches, but the Dual Bandit either improves or reproduces their performance (for $\alpha = 1$). (3) In many cases, SVP as used in POEM has a positive impact on the traditional contextual bandit approach, but the regularisation strength λ is not straightforward to tune. The parameter is essentially unbounded and highly dependent on the variance in the data. An SVP term could straightforwardly be added to the DB objective. IPS reweighting for MLE disturbs the stability of the method, providing mixed results. In the majority of the cases, it causes a significant drop in performance. (4) Most of the compared methods gain significantly in performance when the size of the training sample increases. For sufficiently large enough samples, we expect all methods to slowly converge to the optimum. The most interesting and realistic results deal with large action spaces, finite samples, and limited randomisation, which is exactly where the Dual Bandit shows its superiority.

5 CONCLUSION

In this work, we have motivated the use of methods that exploit bandit feedback for recommendation tasks. Due to the limitations of offline datasets, we introduced simulation environments as an alternative and reproducible evaluation approach. We have presented an overview of the state-of-the-art in counterfactual learning, reviewing commonalities and key differences in existing algorithms. In doing so, we highlighted the fact that value- and policy-based approaches can be formulated with an identical parameterisation. This insight enabled us to propose a new combined MLE-CRM objective, aiming to unify both families. Various experiments underline the superiority of this Dual Bandit approach, excelling the most in the presence of finite samples and limited randomisation. Additionally, we discussed specific properties of the recommendation task such

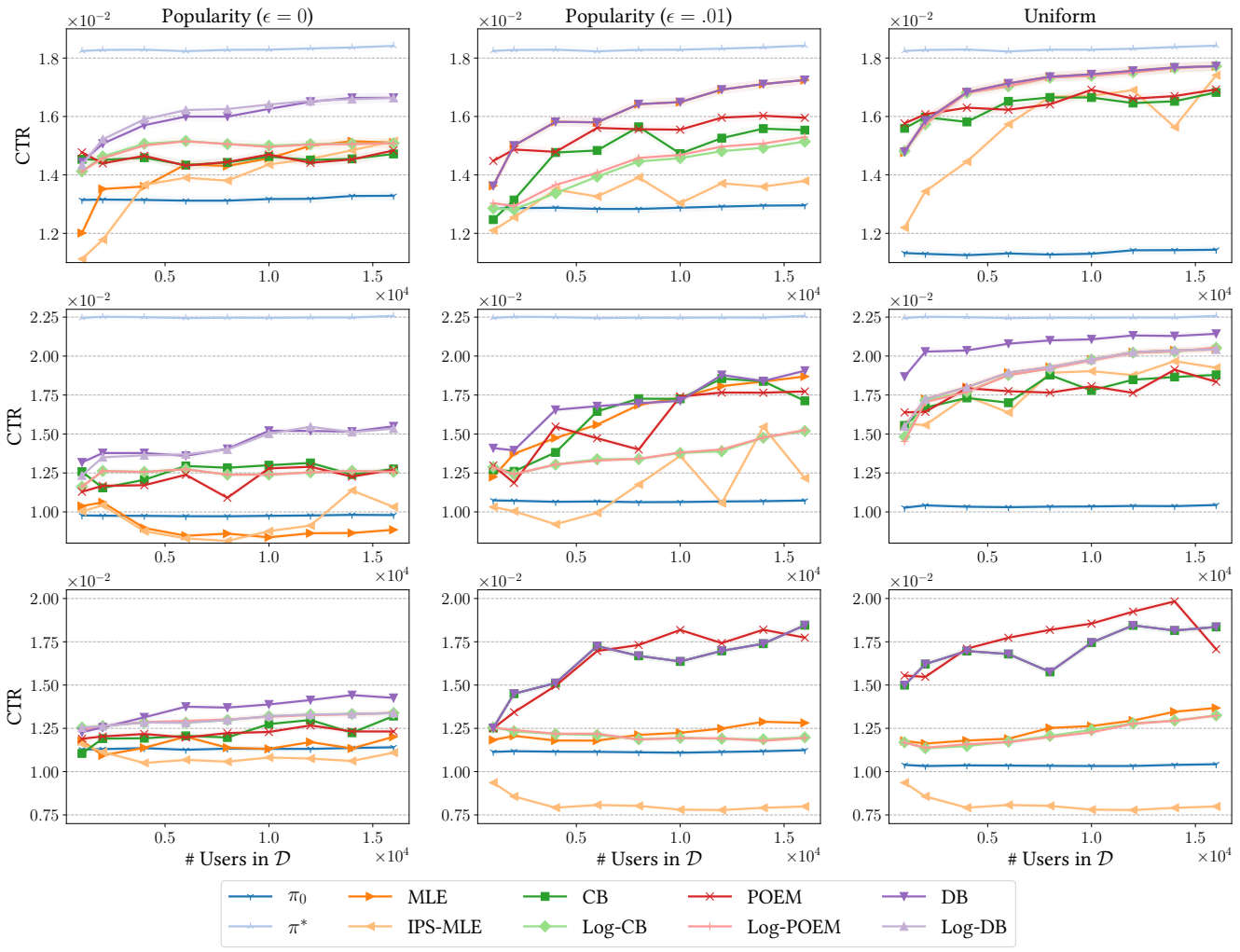

Figure 2: Experimental results from a range of A/B-tests for different settings in the RecoGym environment. Every column corresponds to a different logging policy, rows correspond to action spaces with $n \in \{10, 25, 100\}$. The size of the training sample is increased over the x-axis, the y-axis shows the average attained CTR over 5 runs, along with the 95% confidence interval.

as stochastic and sparse rewards, small treatment effects, and large action spaces. To effectively deal with some of these issues, we introduced a logarithmic variant of the conventional IPS estimator and empirically show how it can further improve performance in the right environments but hurt in the wrong ones, connecting it to analogous findings in related work. Our findings represent the first general empirical study of the use of counterfactual techniques in a bandit-feedback recommendation scenario.

We believe that our work opens up many interesting directions for future research. First, we wish to include additional recent an advanced policy learning and evaluation methods in our comparison, such as DM, DR, MRDR, CAB and others. We specifically wish to further explore the relation between DR and our Dual Bandit approach. Second, the limitations of value-based approaches can be handled in other ways than we presented. Probabilistic models and Bayesian approaches that incorporate priors are a way forward in this aspect, as they can naturally handle the uncertainty that arises in recommendation scenarios. All the models discussed in this work require $O(n^2)$ parameters. By using latent embeddings for the user-state, we could significantly reduce the parameter space to $O(kn)$. This would enable much larger action spaces to be considered, and is very relevant to the recommendation use-case. As the quality of the learned models is then also dependent on the quality of the embeddings, we leave this for future work. Finally, we can extend the value-based methods we compare in this work to higher-order models with non-linearities such as deep neural networks, and extend our analysis to include recent advances in counterfactual learning for top-K recommendations [4], two-stage recommendation pipelines [26], or multiple iterations of logging and learning, touching on the exploration-exploitation trade-off.

ACKNOWLEDGMENTS

We thank Dmytro Mykhaylov and Alexandre Gilotte for insightful comments on early drafts of the manuscript. Olivier Jeunen received funding from the Flemish Government (AI Research Program).

REFERENCES

[1] A. Agarwal, X. Wang, C. Li, M. Bendersky, and M. Najork. 2019. Addressing Trust Bias for Unbiased Learning-to-Rank. In *Proc. of the 2019 World Wide Web Conference (WWW '19)*. ACM, 4–14.

[2] L. Bottou, J. Peters, J. Quiñonero-Candela, D. Charles, D. Chickering, E. Portugaly, D. Ray, P. Simard, and E. Snelson. 2013. Counterfactual reasoning and learning systems: The example of computational advertising. *The Journal of Machine Learning Research* 14, 1 (2013), 3207–3260.

[3] O. Chapelle and L. Li. 2011. An Empirical Evaluation of Thompson Sampling. In *Proc. of the 24th International Conference on Neural Information Processing Systems (NIPS'11)*. 2249–2257.

[4] M. Chen, A. Beutel, P. Covington, S. Jain, F. Belletti, and E. H. Chi. 2019. Top-K Off-Policy Correction for a REINFORCE Recommender System. In *Proc. of the 12th ACM International Conference on Web Search and Data Mining (WSDM '19)*. ACM, 456–464.

[5] M. F. Dacrema, P. Cremonesi, and D. Jannach. 2019. Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches. In *Proc. of the 13th ACM Conference on Recommender Systems (RecSys '19)*. ACM, 101–109.

[6] M. Dudík, J. Langford, and L. Li. 2011. Doubly Robust Policy Evaluation and Learning. In *Proc. of the 28th International Conference on International Conference on Machine Learning (ICML'11)*. 1097–1104.

[7] M. Farajtabar, Y. Chow, and M. Ghavamzadeh. 2018. More Robust Doubly Robust Off-policy Evaluation. In *Proc. of the 35th International Conference on Machine Learning (ICML'18, Vol. 80)*. PMLR, 1447–1456.

[8] F. Garcin, B. Faltings, O. Donatsch, A. Alazzawi, C. Bruttin, and A. Huber. 2014. Offline and Online Evaluation of News Recommender Systems at Swissinfo.Ch. In *Proc. of the 8th ACM Conference on Recommender Systems (RecSys '14)*. 169–176.

[9] A. Gilotte, C. Calauzènes, T. Nedelec, A. Abraham, and S. Dollé. 2018. Offline A/B Testing for Recommender Systems. In *Proc. of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, 198–206.

[10] A. Gruson, P. Chandar, C. Charbuillet, J. McInerney, S. Hansen, D. Tardieu, and B. Carterette. 2019. Offline Evaluation to Make Decisions About Playlist Recommendation Algorithms. In *Proceedings of the 12th ACM International Conference on Web Search and Data Mining (WSDM '19)*. ACM, 420–428.

[11] D. Hosmer Jr., S. Lemeshow, and R. Sturdivant. 2013. *Applied logistic regression*. Vol. 398. John Wiley & Sons.

[12] E. Ie, C. Hsu, M. Mladenov, V. Jain, S. Narvekar, J. Wang, R. Wu, and C. Boutilier. 2019. RecSim: A Configurable Simulation Platform for Recommender Systems. arXiv:1909.04847 [cs.LG]

[13] R. Jagerman, H. Oosterhuis, and M. de Rijke. 2019. To Model or to Intervene: A Comparison of Counterfactual and Online Learning to Rank from User Interactions. In *Proc. of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'19)*. ACM, 15–24.

[14] O. Jeunen. 2019. Revisiting Offline Evaluation for Implicit-feedback Recommender Systems. In *Proc. of the 13th ACM Conference on Recommender Systems (RecSys '19)*. ACM, 596–600.

[15] O. Jeunen, D. Mykhaylov, D. Rohde, F. Vasile, A. Gilotte, and M. Bompaire. 2019. Learning from Bandit Feedback: An Overview of the State-of-the-art. arXiv:1909.08471 [cs.IR]

[16] O. Jeunen, D. Rohde, and F. Vasile. 2019. On the Value of Bandit Feedback for Offline Recommender System Evaluation. arXiv:1907.12384 [cs.IR]

[17] T. Joachims, A. Swaminathan, and M. de Rijke. 2018. Deep Learning with Logged Bandit Feedback. In *Proc. of the 6th International Conference on Learning Representations (ICLR '18)*.

[18] Y. Koren, R. Bell, and C. Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (Aug. 2009), 30–37.

[19] Q. Le, J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, and A. Ng. 2011. On optimization methods for deep learning. In *Proc. of the 28th International Conference on International Conference on Machine Learning (ICML '11)*. 265–272.

[20] D. Lefortier, A. Swaminathan, X. Gu, T. Joachims, and M. de Rijke. 2016. Large-scale validation of counterfactual learning methods: A test-bed. *arXiv preprint arXiv:1612.00367* (2016).

[21] A. S. Lewis and M. L. Overton. 2013. Nonsmooth optimization via quasi-Newton methods. *Mathematical Programming* 141, 1-2 (2013), 135–163.

[22] L. Li, W. Chu, J. Langford, and R. E. Schapire. 2010. A Contextual-Bandit Approach to Personalized News Article Recommendation. In *Proc. of the 19th International Conference on World Wide Web (WWW '10)*. ACM, 661–670.

[23] S. Li, A. Karatzoglou, and C. Gentile. 2016. Collaborative Filtering Bandits. In *Proc. of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '16)*. ACM, 539–548.

[24] D. Liang, R. G. Krishnan, M. D Hoffman, and T. Jebara. 2018. Variational autoencoders for collaborative filtering. In *Proc. of the 2018 World Wide Web Conference (WWW '18)*. ACM, 689–698.

[25] B. London and T. Sandler. 2019. Bayesian Counterfactual Risk Minimization. In *Proc. of the 36th International Conference on Machine Learning (ICML '19, Vol. 97)*. PMLR, 4125–4133.

[26] J. Ma, Z. Zhao, X. Yi, J. Yang, M. Chen, J. Tang, L. Hong, and E. H. Chi. 2020. Off-Policy Learning in Two-Stage Recommender Systems. In *Proc. of the 2020 World Wide Web Conference (WWW '20)*. ACM.

[27] Y. Ma, Y. Wang, and B. Narayanaswamy. 2019. Imitation-Regularized Offline Learning. In *Proc. of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS) (AIStats '19, Vol. 89)*. PMLR, 2956–2965.

[28] H. B. McMahan, G. Holt, D. Sculley, M. Young, D. Ebner, J. Grady, L. Nie, T. Phillips, E. Davydov, D. Golovin, et al. 2013. Ad click prediction: a view from the trenches. In *Proc. of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1222–1230.

[29] D. Mykhaylov, D. Rohde, F. Vasile, M. Bompaire, and O. Jeunen. 2019. Three Methods for Training on Bandit Feedback. arXiv:1904.10799 [cs.IR]

[30] A. B. Owen. 2013. *Monte Carlo theory, methods and examples.*

[31] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*. 8026–8037.

[32] S. Rendle. 2019. Evaluation Metrics for Item Recommendation under Sampling. arXiv:1912.02263 [cs.IR]

[33] S. Rendle, L. Zhang, and Y. Koren. 2019. On the Difficulty of Evaluating Baselines: A Study on Recommender Systems. arXiv:1905.01395 [cs.IR]

[34] D. Rohde, S. Bonner, T. Dunlop, F. Vasile, and A. Karatzoglou. 2018. RecoGym: A Reinforcement Learning Environment for the problem of Product Recommendation in Online Advertising. *ArXiv e-prints* (Aug. 2018). arXiv:1808.00720 [cs.IR]

[35] M. Rossetti, F. Stella, and M. Zanker. 2016. Contrasting Offline and Online Results when Evaluating Recommendation Algorithms. In *Proc. of the 10th ACM Conference on Recommender Systems (RecSys '16)*. ACM, 31–34.

[36] O. Sakhi, S. Bonner, D. Rohde, and F. Vasile. 2020. BLOB : A Probabilistic Model for Recommendation that Combines Organic and Bandit Signals. In *Proc. of the 26th ACM Conference on Knowledge Discovery & Data Mining (KDD '20)*. ACM.

[37] R. Salakhutdinov and A. Mnih. 2008. Bayesian Probabilistic Matrix Factorization Using Markov Chain Monte Carlo. In *Proc. of the 25th International Conference on Machine Learning (ICML '08)*. ACM, 880–887.

[38] I. Shenbin, A. Alekseev, E. Tutubalina, V. Malykh, and S. I. Nikolenko. 2020. RecVAE: A New Variational Autoencoder for Top-N Recommendations with Implicit Feedback. In *Proc. of the 13th International Conference on Web Search and Data Mining (WSDM '20)*. ACM, 528–536.

[39] H. Shimodaira. 2000. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference* 90, 2 (2000), 227 – 244.

[40] J. E. Smith and R. L. Winkler. 2006. The Optimizer's Curse: Skepticism and Postdecision Surprise in Decision Analysis. *Management Science* 52, 3 (2006), 311–322.

[41] H. Steck. 2013. Evaluation of Recommendations: Rating-prediction and Ranking. In *Proc. of the 7th ACM Conference on Recommender Systems (RecSys '13)*. ACM, 213–220.

[42] H. Steck. 2019. Embarrassingly Shallow Autoencoders for Sparse Data. In *The World Wide Web Conference (WWW '19)*. ACM, 3251–3257.

[43] A. Storkey. 2009. When training and test sets are different: characterizing learning transfer. *Dataset shift in machine learning* (2009), 3–28.

[44] Y. Su, L. Wang, M. Santacatterina, and T. Joachims. 2019. CAB: Continuous Adaptive Blending for Policy Evaluation and Learning. In *International Conference on Machine Learning (ICML'19)*. 6005–6014.

[45] R. S. Sutton and A. G. Barto. 1998. *Introduction to reinforcement learning*. Vol. 135.

[46] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. 1999. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems (NIPS'99)*. 1057–1063.

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

[48] A. Swaminathan and T. Joachims. 2015. The Self-Normalized Estimator for Counterfactual Learning. In *Advances in Neural Information Processing Systems*. 3231–3239.

[49] N. Vlassis, A. Bibaut, M. Dimakopoulou, and T. Jebara. 2019. On the Design of Estimators for Bandit Off-Policy Evaluation. In *Proc. of the 36th International Conference on Machine Learning (ICML'19, Vol. 97)*. PMLR, 6468–6476.

[50] R. J. Williams. 1992. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning* 8, 3–4 (May 1992), 229–256.

[51] J. Yu, S.V.N. Vishwanathan, S. Günter, and N. Schraudolph. 2010. A quasi-Newton approach to nonsmooth convex optimization problems in machine learning. *Journal of Machine Learning Research* 11, Mar (2010), 1145–1200.

[52] H. Zou, K. Kuang, B. Chen, P. Chen, and P. Cui. 2019. Focused Context Balancing for Robust Offline Policy Evaluation. In *Proc. of the 25th ACM Conference on Knowledge Discovery & Data Mining (KDD '19)*. ACM, 696–704.

A REPRODUCIBILITY APPENDIX

In what follows, we describe our experimental set-up in further detail. We make use of publicly available simulators to aid in the reproducibility of our work. Implementations of all methods are written in Python3.7, using PyTorch [31].[4]

A.1 Derivation of $\hat{R}_{\text{IPS}}$ lower bound

We derive the lower bound for the empirical IPS estimator as follows:

$$\begin{aligned}\hat{R}_{\text{IPS}}(\pi_{\boldsymbol{\theta}}, \mathcal{D}) &= \sum_{i=1}^{n} \frac{c_i}{\pi_0(a_i|\mathbf{x}_i)} \pi_{\boldsymbol{\theta}}(a_i|\mathbf{x}_i) \\ &= \frac{1}{\sum_{i=1}^{n} \frac{c_i}{\pi_0(a_i|\mathbf{x}_i)}} \sum_{i=1}^{n} \frac{c_i}{\pi_0(a_i|\mathbf{x}_i)} \sum_{i=1}^{n} \frac{c_i}{\pi_0(a_i|\mathbf{x}_i)} \pi_{\boldsymbol{\theta}}(a_i|\mathbf{x}_i) \\ &= \sum_{i=1}^{n} \frac{c_i}{\pi_0(a_i|\mathbf{x}_i)} \sum_{i=1}^{n} \frac{c_i}{\pi_0(a_i|\mathbf{x}_i)} \frac{1}{\sum_{i=1}^{n} \frac{c_i}{\pi_0(a_i|\mathbf{x}_i)}} \pi_{\boldsymbol{\theta}}(a_i|\mathbf{x}_i)\end{aligned}$$

Subsequently taking the logarithm and applying Jensen's inequality (that says $\log(E(x)) \geq E(\log(x))$, we have that:

$$\begin{aligned}\log(\hat{R}_{\text{IPS}}(\pi_{\boldsymbol{\theta}}, \mathcal{D})) \geq \log\Big(\sum_{i=1}^{n} \frac{c_i}{\pi_0(a_i|\mathbf{x}_i)}\Big) \\ + \sum_{i=1}^{n} \frac{c_i}{\pi_0(a_i|\mathbf{x}_i)} \frac{1}{\sum_{i=1}^{n} \frac{c_i}{\pi_0(a_i|\mathbf{x}_i)}} \log(\pi_{\boldsymbol{\theta}}(a_i|\mathbf{x}_i))\end{aligned}$$

A.2 The RecoGym Environment

We make use of the RecoGym simulation environment in our experiments [34].[5] An OpenAI Gym-inspired framework, it provides a standard, robust and reproducible way of evaluating recommendation approaches through simulation. It provides functionality to generate offline logs under a given logging policy (for training and/or counterfactual evaluation), and additionally allows for the opportunity to simulate online experiments such as A/B-tests [16]. Below, we provide an overview of the actual simulation framework behind RecoGym. Note that this is not our contribution, and all merit should go to the authors of the original paper [34]. We merely aim to provide a comprehensive overview of its inner workings for the interested reader, as the original paper only presents it from a higher level perspective.

Users and items are modelled via an underlying latent-factor model, as is widely accepted in the literature [18]. When the environment is set up, the system generates an embedding consisting of k latent factors for every item. These embeddings are represented in a real-valued matrix $\Gamma \in \mathbb{R}^{n \times k}$ and drawn from a multivariate Gaussian distribution centred around 0 with unit variance: $\Gamma \sim \mathcal{N}(0, 1)$. A notion of item popularity is modelled as an additive bias per item: $\mu \in \mathbb{R}^n$, normally distributed with a configurable variance: σ_μ^2. Γ and μ directly impact how users organically interact with these items. The RecoGym authors argue that a given user's bandit behaviour for an item would be different, but similar to the organic behaviour between that user and item. The rationale being that users' natural browsing behaviour and reactions to recommendations are related, but not an exact one-to-one mapping. As such, bandit embeddings and popularities are obtained by performing a transformation f on the organic parameters: $B, \mu' = f(\Gamma, \mu)$. We will not go further in-depth on how this transformation f is performed, and refer interested readers to the RecoGym source code for further information.

Parameter	k	σ_μ	σ_u	n
Value	5	3	0.1	{10,25,100}

Table 2: Parameter configurations for the RecoGym environment we used for our experiments.

Users are described by vectors that conceptually reside in the same latent space as the items. That is, a user embedding $\omega \in \mathbb{R}^k$ is sampled from a multivariate Gaussian distribution with configurable variance: $\omega \sim \mathcal{N}(0, \sigma_u^2)$. Users are simulated sequentially and independently. The behaviour of a single user is modelled as a Markov chain, being either in an "organic" or "bandit" state. The organic state implies the user is currently browsing the item catalog, and generates organic user-item interactions. The bandit state on the other hand requires interventions from the agent, and generates the labeled training data we use for learning throughout this work. We use the default values for the state transition probabilities.

The next item a user views organically is sampled from a categorical distribution, where the individual probabilities for every item are proportional to how similar the user and the respective item's latent organic embeddings are. Formally, given u we draw

$$i \sim \text{Categorical}(\rho), \text{ where } \rho_i \propto \exp(\omega \Gamma_{i,\cdot}^\intercal + \mu_i).$$

When an action a is performed by an agent, the probability of it actually leading to a click is Bernoulli-distributed, with the probability of success being dependent on the similarity between the user and the recommendation's bandit embedding:

$$c \sim \text{Bernoulli}(p), \text{ where } p \propto \sigma(\omega B_{a,\cdot}^\intercal + \mu_a').$$

Naturally, consistently taking the action a^* with the highest probability of leading to a click leads to an optimal recommendation policy:

$$a^* = \arg\max_a \omega B_{a,\cdot}^\intercal + \mu_a'.$$

This is exactly the skyline policy π^* shown in Figure 2. Naturally, none of these parameters are accessible to the learning algorithms we consider in our work: all they observe and learn from are the $(\mathbf{x}, a, p, c)$-samples as introduced in Section 2. Furthermore, the relation between the user history of organic interactions $\mathbf{x}$ and the probability of a click for a given action is non-linear, whereas all the approaches we study model them as such.

A.3 Experimental Setup

Here, we list the details for the experimental setup used throughout Section 4. We vary the number of items $n \in \{10, 25, 100\}$. As we model users in a bag-of-word fashion and our models consist of n^2 parameters, this prohibits us to increase n significantly. Modelling users with latent embeddings is a solution for this, but falls outside the scope of this work. Our experimental observations are general and translate to larger action spaces, with the exception that effect

[4] Source code available at https://github.com/olivierjeunen/dual-bandit-kdd-2020.
[5] https://github.com/criteo-research/reco-gym

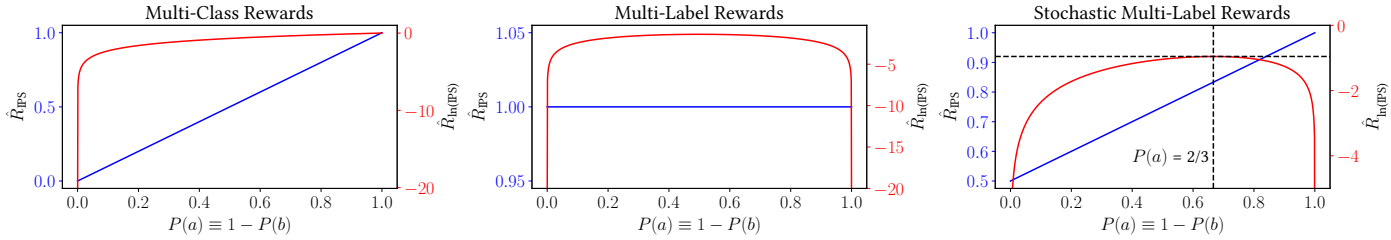

Figure 3: Behaviour of the objective functions corresponding to the $\hat{R}_{\text{IPS}}$ (blue, leftmost y-axis) and $\hat{R}_{\ln(\text{IPS})}$ (red, rightmost y-axis) estimators for the toy examples presented in Section 3.1. We see that both estimators agree on an optimal policy in the case of deterministic rewards, but this is no longer true in the case of a stochastic reward process.

sizes typically diminish. The number of unique users in the training sample varies in {1 000, 2 000, 4 000, 6 000, 8 000, 10 000 12 000, 14 000, 16 000}. These settings lead to anywhere 80 000 and 1 280 000 bandit feedback samples. The empirical CTR for the logging policy in the training data varies from 1.1% to 1.4%, yielding a wide spread between roughly 10 and 1 800 positive samples per item on average. Every contending method was trained until convergence and subsequently tested in a simulated A/B-test with 10 000 users. This process was repeated with 5 different random seeds, aggregating results to provide a robust CTR estimate with a tight confidence interval. We report the 95% confidence interval with error bars on the plot. Hyper-parameters were optimised through a grid-search where the training set consisted of 5 000 users, and validated through simulated A/B-tests with 10 000 users as outlined above, albeit with different random seeds as to ensure the training, validation and test users to be disjoint. Grid searches were analogously repeated and results averaged out over 5 runs to reduce the inherent noise of the simulation. Note that results from the grid-search with only 5 000 users for training might not yield the optimal hyper-parameters for smaller or larger training samples. Furthermore, optimising hyper-parameters through online experiments might not accurately reflect a real-world situation, but it ensures a fair comparison among algorithms. We varied the SVP-strength for POEM $\lambda \in \{.0, 0.05, .1, .25, .5, 1.0, 1.5, 2.0\}$ and DB's CRM-MLE balance $\alpha \in \{.0, .8, .85, .925, .95, .975, 1 - 1e2, 1 - 1e3, 1 - 1e4, 1.0\}$, and report results only for the optimal values. However, we observed much more stable results for varying α than for varying λ. $\alpha = 0$ corresponds to the (Log-)CB objective, $\alpha = 1$ to MLE. Analogously, $\lambda = 0$ corresponds to (Log-)CB, whereas $\lambda \to \infty$ goes to the logging policy. As we mentioned in Section 4, none of the weights ϵ for the IML-term used in PIL-IML improved performance. We varied the weight $\epsilon \in \{.0, .05, .1, .15, .20, .25, .50, 1.0\}$, and observed a steady decrease in attained CTR. The Lagrange multiplier γ for BanditNet was varied over {.0, .125, .25, .5, .75, .875, 1.0}, and was equally unable to improve results. Note that for $\gamma = 1.0$, the model only learns from the non-clicks and ignores the clicks. In this setting, propensity overfitting occurs. We observed that adding an additional IML term to BanditNet was then indeed able to prevent this, but the quality of the learned models remained subpar to those optimised for the original objectives.

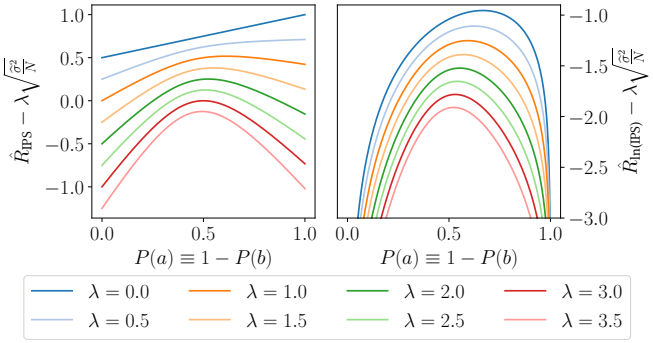

Figure 4: Impact of the SVP term on both the $\hat{R}_{\text{IPS}}$ (left) and $\hat{R}_{\ln(\text{IPS})}$ (right) estimators, for varying strengths λ and the stochastic example setup in Figure 1. The logarithmic estimator penalises degenerate policies, and SVP encourages imitation of the logging policy.

A.4 Behaviour of Convex Policy Lower-Bound

Section 3.1 and Figure 1 discuss how the logarithmic IPS lower bound penalises policies that miss a single positive action in the training sample, leading to an allocation of probability mass that is proportional to the observed reward per action. Figure 3 shows how this only impacts the optimum of the objective function when rewards are stochastic. In the deterministic multi-class example, only class a is correct. In the deterministic multi-label example, labels a and b are both equally correct and $\hat{R}_{\text{IPS}}$ becomes indifferent. In the stochastic multi-label case, actions a and b are not equally likely to lead to a positive reward, and the optimum for $\hat{R}_{\ln(\text{IPS})}$ diverges from that of $\hat{R}_{\text{IPS}}$.

Figure 4 shows the impact of adding an additional SVP term to the optimisation objectives with varying strengths λ, for the stochastic multi-label setup. λ is the SVP strength, $\widehat{\sigma}^2$ an empirical estimate of the variance. Naturally, as $\lambda \to \infty$, the objective is increasingly dominated by the SVP term and the optimal policy moves towards the logging policy (the uniform distribution in this toy example). Intuitively, we can see that adding an SVP term to the original estimator $\hat{R}_{\text{IPS}}$ has similar effects as using the logarithmic variant: extreme values (0 and 1) are increasingly penalised, making the objective better behaved. Adding the SVP term to the logarithmic estimator $\hat{R}_{\ln(\text{IPS})}$ has less dramatic effects and does not change the shape of the objective function as much, as it was concave to begin with. It shifts the optimum towards imitating the logging policy ($P(a) = \pi_0(a)$).