Active Offline Policy Selection

Anonymous Author(s)
Affiliation
Address
email

Abstract

This paper addresses the problem of policy selection in domains with abundant logged data, but with a very restricted interaction budget. Finding a solution to this problem is important because it would enable safe evaluation and deployment of offline reinforcement learning policies in industry, robotics, and healthcare domain among others. Several off-policy evaluation (OPE) techniques have been proposed to assess the value of policies using only logged data. However, there is still a big gap between the evaluation by OPE and the full online evaluation in the real environment. To reduce this gap, we introduce a novel active offline policy selection problem formulation, which relies both on logged data and limited online interactions with the environment. We build upon Bayesian optimization to decide which policies to evaluate in a sequential manner. We rely on the advances in OPE to warm start the evaluation and combine it with online policy evaluations as evaluation progresses. To make this approach scalable to the large number of policies, we introduce a kernel function to model similarity between policies. We use several benchmark environments to show that the proposed approach improves upon state-of-the-art OPE estimates and outperforms common interaction strategies including uniform sampling and independent multi-armed bandits.

1 Introduction

Reinforcement learning (RL) has recently proven to be successful in a range of applications from computer and board games [44, 53, 56] to robotics and chip design [35, 9, 43]. However, many challenges of real-world systems still prevent RL from being applied at scale in practice [14]. One of the limiting factors in real applications is that environment interactions are often expensive. Thus, training and evaluating agents becomes prohibitively slow and costly. Offline RL attempts to address this problem by training agents on a dataset without any additional environment interactions [2, 21]. This enables researchers to train policies with multiple algorithms and different hyperparameter settings [48] on a single dataset. The best policy can then be chosen with off-policy policy evaluation (OPE) approaches [51, 38]. These techniques attempt to estimate the expected agent performance by only looking at the pre-recorded dataset [26, 39, 34, 68].

Unfortunately, OPE estimates are not yet precise enough to be relied upon for choosing which agent to deploy. In particular, OPE is intrinsically limited by the pre-recorded dataset. When there is a distribution shift between a trained policy and a behavioural policy, *i.e.*, when the trained policy behaves very differently from the data collection policy, OPE is imprecise [36]. In many applications, however, a subset of policies can be evaluated *online* prior to final deployment [21, 17]. For example, in robotics a small subset of policies, which were trained offline, can be evaluated on the real hardware, even though this process is laborious. Another example is recommender systems, which allow for restricted policy evaluation via A/B tests on a small fraction of user traffic. Both examples could benefit from sequential decision tools to select which policies to test in practice.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

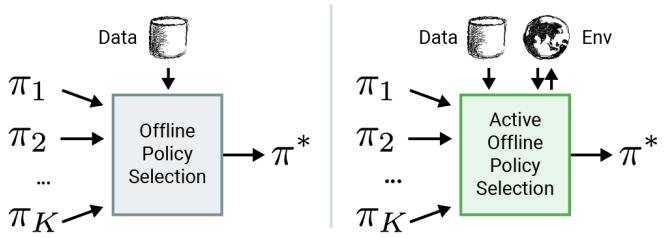

Figure 1: Offline policy selection attempts to choose the best policy from a set of policies, given only a dataset of logged experiences. Active offline policy selection additionally assumes a restricted environment interaction budget which is assigned intelligently to evaluate the most promising policies.

In this paper, we introduce the active offline policy selection problem. The goal is to identify the best policy for deployment by relying both on offline evaluation with logged data and online evaluation with limited interaction (Fig. 1). To address this problem we advance a solution based on Bayesian optimization (BO). It entails learning a Gaussian process (GP) surrogate function that maps policies to expected returns. Then, we build on the GP statistics to construct an acquisition function to decide which policy to try next. Applying BO in this setting is non-trivial for two reasons. First, at the start of online evaluation when the number of observations is tiny, BO cannot identify informative policies due to the difficulties of hyper-parameter optimisation. We overcome this obstacle by a warm start of BO with existing OPE estimates which are incorporated as additional noisy observations (subsection 2.2). Second, when the number of candidate policies is large (or even larger than the interaction budget), it is impossible to execute all policies in the environment enough to evaluate each expected value precisely. We make our method data efficient by considering correlation between policies through a kernel function [24] (Fig. 3). Then, information about the performance of one policy informs us about the performance of similar behaving policies without costly execution in the environment.

The contributions of this paper are as follows.

1. We introduce the active offline policy selection problem. This problem is important in practice, but so far it has not been studied in the literature to the best of our knowledge.
2. We build a BO solution with an extended observation model to integrate both OPE estimates and interaction data. The use of OPE estimates enables us to warm-start learning a GP.
3. We propose a novel GP kernel to capture the dependency between the policies through the actions that they take. As a result, our method infers the value of one policy from those of similar policies.

The rest of this paper is organized as follows. Section 1.1 gives background on offline policy evaluation and selection. In section 2, we propose a Gaussian process model for policy value estimation, explain how we approach the active policy selection problem with Bayesian optimisation and introduce a novel kernel based on the similarity of policy outputs. Section 3 discusses the related work. Section 4 shows that active policy evaluation can improve upon the offline evaluation after merely a few interactions, and our kernel ensures that our selection is data-efficient. Additionally, our method works reliably with OPEs of varying amount and quality. The paper is concluded with a discussion in section 5.

1.1 Off-policy policy evaluation and selection

We consider a Markov decision process (MDP) defined by a tuple $(S, A, T, R, d_0, \gamma)$, with state space S, action space A, transition distribution $T(s'|s,a)$, reward function $R(s,a)$, initial state distribution $d_0(s)$, and discount factor $\gamma \in [0,1]$. A policy $\pi(a|s)$ maps from S to A. The value of a policy π is measured by the expected sum of discounted rewards:

$$\mu_\pi = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right], \text{ with } s_0 \sim d_0(\cdot), a_t \sim \pi(\cdot|s_t), s_{t+1} \sim T(\cdot|s_t, a_t)\,. \tag{1}$$

Typically, estimating the value of a given policy requires executing the policy in the environment many times. However, practical constraints may make it difficult to run policies of unknown quality in

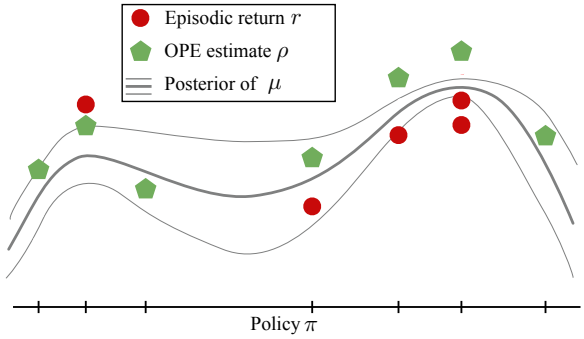

Figure 2: Gaussian process of the policy value function. We observe OPE estimates for all policies and noisy episodic returns for some of them. Neighboring policies (measured by kernel $\mathcal{K}$) have similar values, and the posterior variance is lower where there are more observations. Policies are aligned in 1D for visualization purposes, but they actually reside in a high dimensional space.

the environment frequently. To address this, off-policy policy evaluation (OPE) estimates a value $\hat{\mu}_\pi$ using a dataset of trajectories $\mathcal{D}$ collected by a separate behavior policy π_β. In offline policy selection (OPS) the task is to select the policy with the highest value from a set of policies $\pi_1, \pi_2, \ldots, \pi_K$ given access only to a dataset $\mathcal{D}$ (Fig. 1, left). A straightforward approach is to select the policy with the highest OPE estimate $\hat{\mu}_{\pi_1}, \ldots, \hat{\mu}_{\pi_K}$ as in [18], although depending on the selection metric other approaches might be preferable [67].

2 Active offline policy selection

2.1 Problem definition

We now formally define the active offline policy selection problem. Suppose we are given a set of K candidate policies $\pi_k, 1 \leq k \leq K$. Denote by μ_k the unknown policy value and by ρ_k an OPE estimate computed from an offline dataset. At every step, i, we can choose a policy k_i to execute in the environment once and observe the (discounted) episodic return r_i, which is a noisy sample of the true policy value: $\mathbb{E}[r_i] = \mu_{k_i}$. We would like to find the policy with the highest value, $k^* = \arg\max_k \mu_k$, with as few trajectories as possible. Simple regret of the recommended policy measures of how close we are to this target:

$$\text{regret} = \mu_{k^*} - \mu_{\hat{k}}\,. \tag{2}$$

To solve this sequential decision problem, we introduce GP over the policy returns and OPE estimates (Sec. 2.2), and use these as part of a BO strategy (Sec. 2.3). We design the GP kernel over policies to model their correlated returns (Sec. 2.3).

2.2 Gaussian process over policy values

Given a limited interaction budget and a large number of policies, it is impossible to obtain accurate estimates of their values. Fortunately, similar policies tend to have similar values. Thus, propagating the information between them helps to improve value prediction. Formally, we can use GP to model this correlation so as to predict a distribution over the policy values.

A GP is a stochastic process, that is an indexed collection of random variables, such that every finite collection of these random variables has a multivariate normal distribution [52]. In our case, we assume that the policy value μ as a function of policy π follows a GP, with a kernel function $\mathcal{K}(\pi_i, \pi_j)$ over policies and mean m. Specifically, we consider the following generative process for the latent policy value $\mu(\pi)$, OPE estimate ρ, and episodic return r:

$$\mu(\pi) \sim \text{GP}(m(\cdot), \mathcal{K}(\cdot,\cdot)), \quad \rho \sim \mathcal{N}(\mu(\pi), \sigma_\rho^2), \quad r \sim \mathcal{N}(\mu(\pi), \sigma_r^2). \tag{3}$$

We use a constant mean m without loss of generality. We assume a flat prior for the hyper-parameter m, and weakly informative inverse Gamma prior for the variance σ_ρ^2 and σ_r^2. Our probabilistic model is illustrated in Fig. 2.

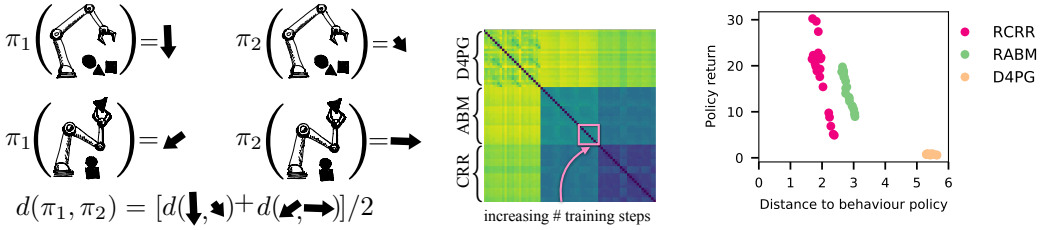

Figure 3: Left. The distance between two policies π_1 and π_2 is computed as an average distance across pairs of action vectors that are suggested by each policy on a set of states. Middle. Pairwise distance matrix between policies on `humanoid-run` task. High distance is shown in yellow and low distance is in dark blue. The 3×3 block structure of the matrix reflects the 3 types of training algorithm. Smaller blocks correspond to the increasing number of training steps with the same hyperparameters. Right. Policy value (μ_π) versus action distance to the behavior policy $\bar{d}(\pi, \pi_\beta)$. Policies with the same training algorithm are in the same color.

As we have a finite number of policies, the input space is a finite set $\mathcal{X} = \{\pi_1, \ldots, \pi_K\}$, where each index corresponds to a candidate policy. The joint distribution of the K policy values reduces to a multivariate normal distribution,

$$\mu_1, \ldots, \mu_K \sim \mathcal{N}(m\mathbf{1}_K, \mathbf{K}), \text{ with } \mathbf{K}_{k,k'} = \mathcal{K}(\pi_k, \pi_{k'}), \tag{4}$$

where $\mathbf{1}$ is a K-length vector of ones. For each policy π_k, we have one noisy observation from an OPE method ρ_k, and zero or multiple noisy episodic return observations r_k^i. As the policies are related through the covariance matrix $\mathbf{K}$, observing a return sample from one policy updates the posterior distribution of all of them. Intuitively, the number of policies that we need to execute to find the best policy depends on their diversity instead of the quantity.

Given the OPE estimates $\boldsymbol{\rho} = [\rho_1, \ldots, \rho_K]$ and return observations $\mathbf{r}_k = [r_k^1, \ldots, r_k^{N_k}]$ for all policies k, the posterior distribution of the mean return is also a Gaussian,

$$\begin{aligned}
&\mu_1, \ldots, \mu_K | \boldsymbol{\rho}, \{\mathbf{r}_k\}_k \sim \mathcal{N}(\mathbf{m}, \boldsymbol{\Sigma}), \text{ with} \\
&\mathbf{m} = \mathbf{K}(\mathbf{K} + \boldsymbol{\Lambda})^{-1}\mathbf{y}, \qquad \boldsymbol{\Sigma} = \mathbf{K} - \mathbf{K}(\mathbf{K} + \boldsymbol{\Lambda})^{-1}\mathbf{K} \\
&\mathbf{y}_k = \left(\frac{1}{\sigma_\rho^2} + \frac{N_k}{\sigma_r^2}\right)^{-1} \left(\frac{\rho_k}{\sigma_\rho^2} + \frac{1}{\sigma_r^2}\sum_{i=1}^{N_k} r_k^i\right), \qquad \boldsymbol{\Lambda}_{kk} = \left(\frac{1}{\sigma_\rho^2} + \frac{N_k}{\sigma_r^2}\right)^{-1}
\end{aligned} \tag{5}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix with diagonal entries defined above. For deployment, we choose the policy with the highest posterior mean: $\hat{k} = \arg\max_k \mathbf{m}_k$.

2.3 Kernel

A key component of the GP model is the kernel that measures our belief about the policy correlation. To obtain a kernel we make a key assumption: *Policies that are similar in the actions that they take, yield similar returns*. Then, our insight is to measure the distance between the policies through the actions that each of them takes on a fixed set of states from the offline dataset (Fig. 3, left).

Formally, to measure the similarity between π_1 and π_2, we obtain the policy actions at a given state s and measure the distance between these two action vectors $d(\pi_1(s), \pi_2(s))$ [1]. We then define the distance between the two policies as the average distance over the offline dataset $\mathcal{D}$,

$$\bar{d}(\pi_1, \pi_2) = \mathbb{E}_{s \sim \mathcal{D}}\, d(\pi_1(s), \pi_2(s)). \tag{6}$$

In practice we approximate $\bar{d}$ using a randomly sampled subset of states $\mathcal{D}' \subset \mathcal{D}$.

An example of a pairwise distance matrix for a set of policies is shown in Fig. 3 (middle). Rows and columns are the policies which are arranged first by the type of the training algorithm, then by hyperparameters and finally by the training steps (none of which is observed by our method). Notice

[1] We use Euclidean distance for deterministic policies with continuous actions, but an extension to stochastic and discrete actions is straightforward.

the block structure of this figure that reflects the different nature of training algorithms. Smaller blocks with increasing distances correspond to policies with more training steps and same hyperparameters.

Additionally, Fig. 3 (right) shows an example environment where the value of a policy produced by the same algorithm changes gradually as a function of the distance to a behaviour policy. It provides convincing evidence that our kernel construction is informative for predicting the value of a policy.

Finally, we compute Matérn $1/2$ kernel as: $\mathcal{K}(\pi_1, \pi_2) = \sigma_k^2 \exp(-\bar{d}(\pi_1, \pi_2)/l)$, where σ and l are the trainable variance and length-scale hyperparameters. This is a popular kernel choice [54]. As the distance metric is computed as the average distance over states, using it in this kernel is equivalent to a multiplication of kernels on these states and it is a valid positive semi-definite matrix.

2.4 Active offline policy selection with Bayesian optimization

Under a GP formulation, we can now employ Bayesian optimization (BO) [54] to search for the best policy efficiently. BO optimizes an unknown function $\mu(\pi)$ with $\pi \in \mathcal{X}$ using a limited number of function queries that render noisy observations. BO has been used successfully in many applications including experimental design [5], preference learning and interactive machine learning [8, 7], robotics [40, 41] and hyper-parameter optimization [57, 25, 11]. The key component of a BO algorithm is the acquisition function $u_i(\pi)$ that balances exploration and exploitation when selecting a query. Widely used acquisition functions include upper confidence bounds (UCB), expected improvement (EI) and Thompson Sampling. Here, we simply choose the UCB acquisition function [58, 28] [2]. Specifically, at every step i, we compute the score for a policy π_k using

$$u_i(k) = \mathbf{m}_k + \sqrt{\beta_i \mathbf{\Sigma}_{kk}}, \tag{7}$$

where β_i is a constant depending on i. We then choose the policy with the highest score, $k_i = \arg\max_k u_i(k)$, to execute next. After observing a new return r_{k_i}, we update $\mathbf{m}$ and $\mathbf{\Sigma}$ in (5). We also update the hyper-parameters (m, σ_ρ^2, σ_r^2, σ_k^2 and l) with the maximum a posteriori estimate (see detailed derivation in the appendix).

Estimating the hyper-parameters with limited data is challenging in practice. Fortunately, in the active offline policy selection formulation, we can take advantage of the OPE estimates as a priori observations to fit the hyper-parameters before launching the interactive optimization process. To ablate the value of OPE, in some of our experiments we will assume no OPE estimates and instead execute 5 randomly sampled policies before fitting the hyperparameters.

3 Related work

Offline RL is becoming popular because it can leverage offline data and due to the availability of the benchmark datasets [21, 17]. We categorize offline RL methods into the following groups: 1) policy-constraint approaches that regularize the learned policy to stay close to the behavior policy like CRR [66] and BCQ [19], 2) value-based approaches to have more conservative estimates, either through regularization or uncertainty [33, 20], 3) model-based approaches [69, 29], and 4) typical off-policy RL methods such as D4PG [6].

Off-policy evaluation has been studied extensively across many domains [37, 64, 27, 47]. Generally OPE methods can be grouped into methods that use importance sampling for the policy [50] or stationary state distribution [39, 46, 65], value function methods [61, 42, 62], and learned transition models [70], as well as methods that combine two or more approaches [45, 13, 26, 16].

Offline policy selection (OPS) is a related practical problem that receives increasing interest [48, 67, 18]. Our work has two notable distinctions in contrast to these prior works: 1) we consider similarities between policies when making value predictions, and 2) we focus on active offline policy selection, where we allow a small number of evaluations in the environment to quickly improve our estimate of which policy is best.

Policy Evaluation Networks [22] work considers a *network fingerprinting* that learns a policy embedding through the output of the policy on *probing states*. The principal difference to our approach is that we only consider a very limited online policy evaluation budget (in the order of 100) while

[2] Experiments with other acquisition functions appear in the appendix.

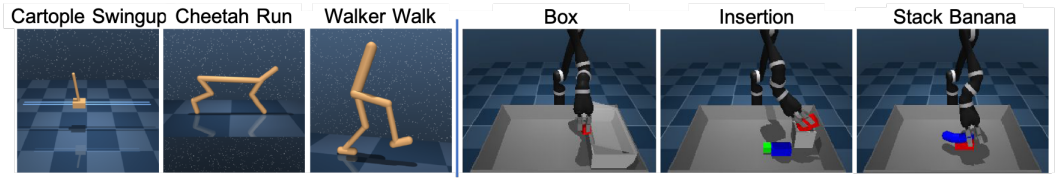

Figure 4: Environment used in the experiments. 9 environments from the dm-control domain (3 examples on the left) and 4 from the MPG domain (3 examples on the right).

network fingerprinting needs many policy evaluations to train policy embeddings (in the order of $100\,000$). Related to the policy representation through actions, classifiers in active learning were also characterised by their predicted class probabilities on a subset of datapoints [32].

Applying BO in a finite set of indices reduces to a Bayesian multiarmed bandit problem. Exploring the shared structure among arms using Bayesian models has been studied in the literature [58, 30, 24]. Our work considers the Bayesian bandit model in the particular setting of policy selection and build our model by exploiting the offline dataset.

Active learning has been considered in RL [15, 3, 12, 31] for better exploration or learning reward models. It has not been studied for the OPE or OPS problems to the best of our knowledge.

4 Experiments

4.1 Datasets and environments

In our experiments we consider two domains (Fig. 4) with different properties. One domain is a standard benchmark in RL and another one is motivated by the challenges of evaluating control policies for robots. For each domain we describe the environments, datasets for offline RL and OPE, and sets of policies below.

DeepMind Control Suite (dm-control) This is a standard set of continuous control environments [63]. It contains 9 tasks of locomotion and manipulation and a selected subset of tasks is shown in Fig. 4, left. The observations are the state of the MPD, including joint angles and velocities. The dimensionality and semantics of the continuous actions vary across tasks. The dataset from RL Unplugged benchmark [21] is used for policies and kernel. The offline RL policies are obtained from DOPE [18], a benchmark for offline policy evaluation and selection. For each task, there are up to 96 policies trained with various algorithms (including behavior cloning(BC) [49], D4PG [6], ABM [55], and CRR [66]), and hyperparameters.

Manipulation Playground (MPG) This is a simulated robotics environment. The task is to control a simulated Kinova Jaco arm with 9 degrees of freedom. Joint velocity control of 6 arm and 3 hand joints is used. The policies are learnt from image input from frontal and in-hand cameras of 64×64 dimensions and proprioception. We perform 4 several manipulation tasks in 20×20 cm basket (Fig. 4, right). The dataset is generated by running an online RL algorithm following the protocol in [66]. The policies are generated following the protocol in [48]. For each task, there are up to 256 policies trained with various algorithms (including BC, D4PG, and CRR), and hyperparameters.

We can obtain OPE estimates for each policy by running Fitted Q-Evaluation (FQE) [42, 34] on the logged dataset. FQE is the default OPE method used unless stated otherwise.

4.2 Experimental setting and implementation

To evaluate the policy selection procedure in each experiment we randomly select a subset of K policies out of all trained policies. We set $K = 50$ for dm-control and $K = 200$ for MPG. Then, we repeat each experiment 100 times, and report the average results and the standard deviation of the mean estimate. As a performance metric we look at the simple regret (Eq. 2) as a function of the number of episodes. We normalise the scores by dividing them by the gap between the best and the worst performing policy to make the result comparable across domains and environments.

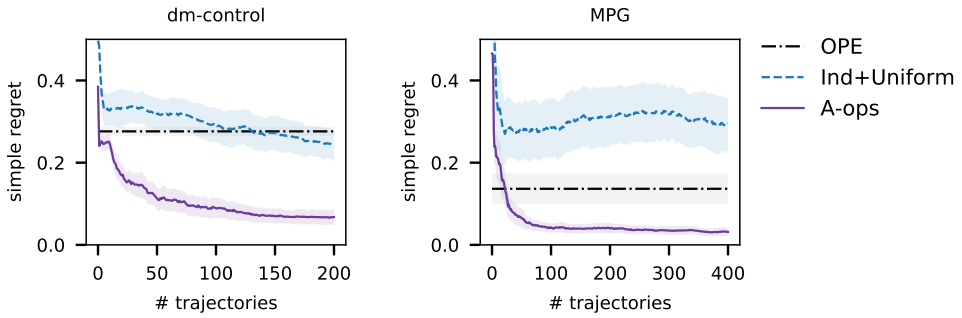

Figure 5: Comparison of A-OPS approach, offline policy selection with OPE, and online policy selection with IND+UNIFORM on dm-control and MPG environments. The results are averaged across 9 and 4 tasks and 100 experiments in each of them. Shaded areas show standard deviation of the mean. Our method A-OPS quickly surpasses the baselines and achieves the lowest regrets.

To study the contribution of modeling the policy correlation with GP, we consider a baseline model with independently distributed value μ_k across policies. The data generating process for the k-th policy follows a hierarchical Gaussian distribution,

$$r_k^i \overset{iid}{\sim} \mathcal{N}(\mu_k, \sigma_{k,r}^2), \forall i, \text{ with } \mu_k \sim \mathcal{N}(\rho_k, \sigma_{k,\rho}^2)\,, \tag{8}$$

where we assume a weakly informative inverse Gamma prior for variance $\sigma_{k,r}^2$ and $\sigma_{k,\rho}^2$ as in GP. We refer to this model as IND.

All online and offline RL algorithms, as well as offline policy evaluation algorithms in this work are implemented with Acme [23] and Reverb [10], and run on GPUs on an internal cluster. We implement GP and IND together with all BO algorithms using the same TensorFlow [1] codebase, and run all the policy selection experiments using CPUs.

We find using a constant value $\sqrt{\beta_i} = 5$ in GP-UCB works well across different budgets, Bayesian models, and environments, although a more sophisticated scheduling [59, 4, 24] could further improve the results. Details about the tasks and policies, a breakout of the results by tasks, and additional baselines and ablations are presented in the supplementary materials.

4.3 Comparison to the existing methods

In our main experiments we would like to verify that the proposed method performs better than two strategies commonly employed in practice (see section 1). To this end, we compare:

1. **A-OPS**: our proposed method that uses OPE estimates to bootstrap GP learning and performs UCB policy sampling as discussed in section 2.
2. **OPE**: standard offline policy selection that selects the policy with best OPE estimate [48].
3. **IND+UNIFORM**: standard online evaluation where we sample policies *uniformly* and select the one with the highest empirical mean return (estimated independently for each policy).

Fig. 5 presents the simple regret as a function of the number of executed trajectories. The results are averaged across 9 tasks from dm-control and 4 tasks from MPG. Notably, for the range of considered budgets, online policy selection IND+UNIFORM often performs worse than offline policy selection with OPE. It suggests single trajectory returns are noisier than OPE estimates in these environments. Naturally, we expect that given a larger budget the online baseline will eventually surpass its offline counterpart in all cases.

Our proposed method A-OPS combines OPE estimates with online policy evaluations. It quickly outperforms OPE and improves with more trajectories. A-OPS estimate of the best policy may be erroneous initially due to noisy reward samples (*e.g.*, in MPG), but the GP quickly updates thus allowing for significant improvements over OPE only after a few actively selected trajectories.

Table 1: Design of the ablation experiments across three axes: the use of OPE, GP or independent policy model, active policy selection or uniform sampling. We refer to our method that combines all the components as A-OPS.

	No OPE		**With OPE**	
	Independent	**GP**	**Independent**	**GP**
Uniform	IND+UNIFORM	GP+UNIFORM	IND+UNIFORM+OPE	GP+UNIFORM+OPE
UCB	IND+UCB	GP+UCB	IND+UCB+OPE	A-OPS

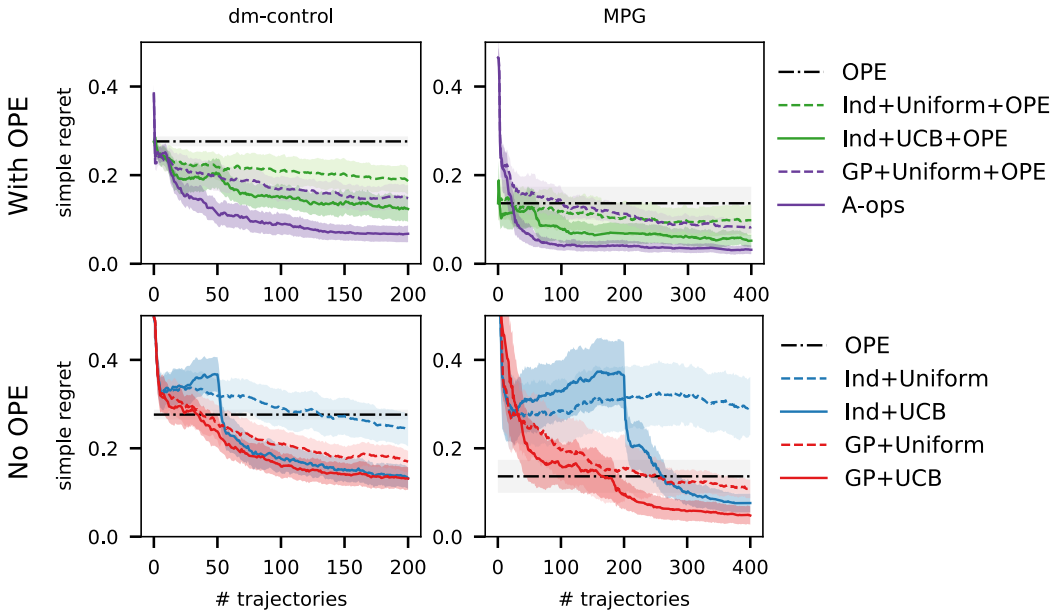

Figure 6: Ablations. In two domains, we ablate each of three components of our approach. 1) The use of OPE (with OPE in the first row and without in the second). 2) The choice of policy model: GP (purple and red) and independent (green and blue). 3) The policy selection strategy: active (dark, solid line) and uniform sampling (bright, dashed line). In general, active is better than uniform, methods that exploit policy correlation are better than methods that do not, and methods that use an OPE prior are better than those that do not.

4.4 Ablation studies

We are not aware of other works on active offline policy selection. Thus, in this section we investigate alternative solutions that help us to understand the importance of the proposed components of A-OPS.

We design our ablation studies to test the contribution of the three components of A-OPS:

1) *OPE evaluation*: Compare selection algorithms with or without OPE evaluations.
2) *active learning*: Compare active policy selection (UCB) with uniform sampling (UNIFORM).
3) *policy kernel*: Compare the GP model with the independent model baseline (IND).

Tab. 1 summarises the 8 possible combinations of the components. We refer to our full method (GP+UCB+OPE) simply as A-OPS. We also include the offline OPE estimates as before.

The results with all the baselines on two domains are presented in Fig. 6. First of all we observe that all the components are crucial for the performance of our method and excluding any of them leads to significantly worse results. Next, we answer three questions about each component individually.

1. How important is it to incorporate OPE estimates? Across all the experiments, incorporating OPE estimates (top row of Fig. 6) always considerably improves the results compared to starting policy evaluation from scratch (bottom row). For example, in MPG when A-OPS starts with OPE, it is able to cut the regret by more than twice before executing 100 trajectories. For this budget, GP+UCB, which starts from scratch, does not reach the same regret as OPE.

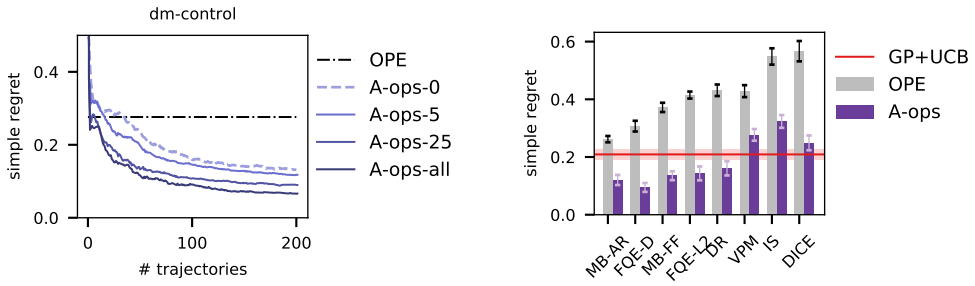

Figure 7: The performance of A-OPS on dm-control with varying number of FQE (left) and type of OPE estimate with a budget of 50 trajectories (right). A-OPS can improve upon the initial OPE estimates in all cases.

2. How informative is our kernel? To understand the value of modelling the correlation between policies through the kernel, we compare the GP method with the independent policy model. In Fig. 6 purple and red lines use GP as the policy model and green and blue use the Ind model. In the vast majority of settings (7 out of 8) the use of a kernel significantly improves the results and in one setting (IND+UNIFORM+OPE vs GP+UNIFORM+OPE) they perform on par. We conclude that the kernel is a key ingredient for improving data efficiency.

3. How important is the selection strategy? The active selection of policies is generally beneficial for identifying a good policy. For this we refer to the results of dashed lines that correspond to UNIFORM policy selection and solid lines that correspond to the use of UCB. Using IND+UCB yields a high regret in the initial exploration stage, but it improves substantially over IND+UNIFORM later. Moreover, incorporating OPE estimates and a kernel, resulting in A-OPS method, significantly shortens the exploration stage.

Incorporating the OPE estimates is clearly beneficial for policy selection. Here, we further study **How does the performance depend on the amount and quality of OPE estimates?**

Fig. 7 (left) depicts the performance of A-OPS by providing OPE estimates for a random subset of $k \in [0, 50]$ policies. Increasing the number of OPE observations consistently improves the results. This result demonstrates that our method A-OPS can seamlessly incorporate all the available data and benefit from the increasing number of initial observations.

Fig. 7 (right) compares A-OPS using estimates from different OPE algorithms in DOPE [18] with a small budget of 50 trajectories. A-OPS always improves upon the OPE estimate it starts with, and the performance clearly depends on the quality of the associated OPE estimate. Some OPE estimates (e.g., IS, VPM and DICE) have large estimation errors and even a negative correlation with the ranks of policies [18]. As expected, A-OPS is negatively impacted by poor OPE estimates. It should be possible to correct for this to a greater extent with more online observations. For now we follow the recommendation to use the FQE estimates.

5 Conclusions

In this paper, we introduced the active offline policy selection problem. We proposed a Bayesian optimization solution that integrates OPE estimates with evaluations obtained by interacting. We also proposed a novel kernel over policies for this domain.

Experiments on two simulated control domains showed that our proposed solution is very effective. The experiments also ablated the different components of the solution, highlighting the value of model correlation via the kernel over policies, active selection, and incorporation of OPE estimates in the observation model.

Having tested the method and demonstrated its efficacy, the obvious future plan is to conduct experiments in real domains. We have focused on reducing data complexity with Bayesian optimization, but have not considered the potential risks of evaluating offline policies directly in real environments. It is possible that we could alleviate this limitation in a similar fashion by applying constrained Bayesian optimization techniques [60] to search for the best policy without violating safety constraints.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. 2019.

[3] Takayuki Akiyama, Hirotaka Hachiya, and Masashi Sugiyama. Efficient exploration through active learning for value function approximation in reinforcement learning. *Neural Networks*, 23(5):639–648, 2010.

[4] Jean-Yves Audibert, Sébastien Bubeck, and Rémi Munos. Best arm identification in multi-armed bandits. In *Conference on Computational Learning Theory*, pages 41–53, 2010.

[5] Javad Azimi, Ali Jalali, and Xiaoli Zhang Fern. Hybrid batch bayesian optimization. In *ICML*, 2012.

[6] Gabriel Barth-Maron, Matthew W Hoffman, David Budden, Will Dabney, Dan Horgan, TB Dhruva, Alistair Muldal, Nicolas Heess, and Timothy Lillicrap. Distributed distributional deterministic policy gradients. In *International Conference for Learning Representations*, 2018.

[7] E. Brochu, T. Brochu, and N. de Freitas. A Bayesian interactive optimization approach to procedural animation design. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 103–112, 2010.

[8] E. Brochu, N. de Freitas, and A. Ghosh. Active preference learning with discrete choice data. In *Advances in Neural Information Processing Systems*, pages 409–416, 2007.

[9] Serkan Cabi, Sergio Gómez Colmenarejo, Alexander Novikov, Ksenia Konyushkova, Scott Reed, Rae Jeong, Konrad Zolna, Yusuf Aytar, David Budden, Mel Vecerik, Oleg Sushkov, David Barker, Jonathan Scholz, Misha Denil, Nando de Freitas, and Ziyu Wang. Scaling data-driven robotics with reward sketching and batch reinforcement learning. In *Robotics: Science and Systems Conference*, 2020.

[10] Albin Cassirer, Gabriel Barth-Maron, Eugene Brevdo, Sabela Ramos, Toby Boyd, Thibault Sottiaux, and Manuel Kroiss. Reverb: A framework for experience replay, 2021.

[11] Yutian Chen, Aja Huang, Ziyu Wang, Ioannis Antonoglou, Julian Schrittwieser, David Silver, and Nando de Freitas. Bayesian optimization in AlphaGo. *arXiv preprint arXiv:1812.06855*, 2018.

[12] Christian Daniel, Malte Viering, Jan Metz, Oliver Kroemer, and Jan Peters. Active reward learning. In *Robotics: Science and systems*, volume 98, 2014.

[13] Miroslav Dudík, John Langford, and Lihong Li. Doubly robust policy evaluation and learning. *arXiv preprint arXiv:1103.4601*, 2011.

[14] Gabriel Dulac-Arnold, Daniel J. Mankowitz, and Todd Hester. Challenges of real-world reinforcement learning. 2019.

[15] Arkady Epshteyn, Adam Vogel, and Gerald DeJong. Active reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pages 296–303, 2008.

[16] Mehrdad Farajtabar, Yinlam Chow, and Mohammad Ghavamzadeh. More robust doubly robust off-policy evaluation. In *International Conference on Machine Learning*, pages 1447–1456, 2018.

[17] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[18] Justin Fu, Mohammad Norouzi, Ofir Nachum, George Tucker, ziyu wang, Alexander Novikov, Mengjiao Yang, Michael R Zhang, Yutian Chen, Aviral Kumar, Cosmin Paduraru, Sergey Levine, and Thomas Paine. Benchmarks for deep off-policy evaluation. In *International Conference on Learning Representations*, 2021.

[19] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, 2019.

[20] Caglar Gulcehre, Sergio Gómez Colmenarejo, Ziyu Wang, Jakub Sygnowski, Thomas Paine, Konrad Zolna, Yutian Chen, Matthew Hoffman, Razvan Pascanu, and Nando de Freitas. Regularized behavior value estimation. *arXiv preprint arXiv:2103.09575*, 2021.

[21] Caglar Gulcehre, Ziyu Wang, Alexander Novikov, Tom Le Paine, Sergio Gomez Colmenarejo, Konrad Zolna, Rishabh Agarwal, Josh Merel, Daniel Mankowitz, Cosmin Paduraru, et al. Rl unplugged: A suite of benchmarks for offline reinforcement learning. *arXiv preprint arXiv:2006.13888*, 2020.

[22] Jean Harb, Tom Schaul, Doina Precup, and Pierre-Luc Bacon. Policy evaluation networks. *arXiv preprint arXiv:2002.11833*, 2020.

[23] Matt Hoffman, Bobak Shahriari, John Aslanides, Gabriel Barth-Maron, Feryal Behbahani, Tamara Norman, Abbas Abdolmaleki, Albin Cassirer, Fan Yang, Kate Baumli, Sarah Henderson, Alex Novikov, Sergio Gómez Colmenarejo, Serkan Cabi, Caglar Gulcehre, Tom Le Paine, Andrew Cowie, Ziyu Wang, Bilal Piot, and Nando de Freitas. Acme: A research framework for distributed reinforcement learning. *arXiv preprint arXiv:2006.00979"*, 2020.

[24] Matthew Hoffman, Bobak Shahriari, and Nando Freitas. On correlation and budget constraints in model-based bandit optimization with application to automatic machine learning. In *Artificial Intelligence and Statistics*, pages 365–374, 2014.

[25] F. Hutter, H. H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *LION*, pages 507–523, 2011.

[26] Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning*, volume 48, pages 652–661, 2016.

[27] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

[28] Emilie Kaufmann, Olivier Cappé, and Aurélien Garivier. On bayesian upper confidence bounds for bandit problems. In *Artificial intelligence and statistics*, pages 592–600, 2012.

[29] Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *arXiv preprint arXiv:2005.05951*, 2020.

[30] Andreas Krause and Cheng S Ong. Contextual gaussian process bandit optimization. *Advances in Neural Information Processing Systems 24*, pages 2447–2455, 2012.

[31] David Krueger, Jan Leike, Owain Evans, and John Salvatier. Active reinforcement learning: Observing rewards at a cost. *arXiv preprint arXiv:2011.06709*, 2020.

[32] Pascal Fua Ksenia Konyushkova, Raphael Sznitman. Discovering general-purpose active learning strategies. *arXiv preprint arXiv:1810.04114*, 2019.

[33] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779*, 2020.

[34] Hoang Le, Cameron Voloshin, and Yisong Yue. Batch policy learning under constraints. In *International Conference on Machine Learning*, pages 3703–3712. PMLR, 2019.

[35] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[36] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[37] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *World Wide Web*, pages 661–670, 2010.

[38] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 297–306, 2011.

[39] Qiang Liu, Lihong Li, Ziyang Tang, and Dengyong Zhou. Breaking the curse of horizon: Infinite-horizon off-policy estimation. In *Advances in Neural Information Processing Systems*, pages 5356–5366, 2018.

[40] Daniel J Lizotte, Tao Wang, Michael H Bowling, and Dale Schuurmans. Automatic gait optimization with gaussian process regression. In *IJCAI*, volume 7, pages 944–949, 2007.

[41] R. Martinez–Cantin, N. de Freitas, E. Brochu, J. Castellanos, and A. Doucet. A Bayesian exploration-exploitation approach for optimal online sensing and planning with a visually guided mobile robot. *Autonomous Robots*, 27(2):93–103, 2009.

[42] Martino Migliavacca, Alessio Pecorino, Matteo Pirotta, Marcello Restelli, and Andrea Bonarini. Fitted policy search: Direct policy search using a batch reinforcement learning approach. In *3rd International Workshop on Evolutionary and Reinforcement Learning for Autonomous Robot Systems*, page 35, 2010.

[43] Azalia Mirhoseini, Anna Goldie, Mustafa Yazgan, Joe Jiang, Ebrahim Songhori, Shen Wang, Young-Joon Lee, Eric Johnson, Omkar Pathak, Sungmin Bae, et al. Chip placement with deep reinforcement learning. *arXiv preprint arXiv:2004.10746*, 2020.

[44] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[45] Remi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 1054–1062. 2016.

[46] Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. Dualdice: Behavior-agnostic estimation of discounted stationary distribution corrections. In *Advances in Neural Information Processing Systems*, pages 2318–2328, 2019.

[47] Xinkun Nie, Emma Brunskill, and Stefan Wager. Learning when-to-treat policies. *arXiv preprint arXiv:1905.09751*, 2019.

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

[49] Dean A Pomerleau. ALVINN: an autonomous land vehicle in a neural network. In *Advances in Neural Information Processing Systems*, pages 305–313, 1988.

[50] Doina Precup. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, page 80, 2000.

[51] Doina Precup, Richard S. Sutton, and Satinder P. Singh. Eligibility traces for off-policy policy evaluation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, page 759–766, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[52] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[53] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy P. Lillicrap, and David Silver. Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model. *Nature*, 588(7839):604–609, 2020.

[54] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2015.

[55] Noah Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, Thomas Lampe, Roland Hafner, Nicolas Heess, and Martin Riedmiller. Keep doing what worked: Behavior modelling priors for offline reinforcement learning. In *International Conference for Learning Representations*, 2020.

[56] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484, 2016.

[57] J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, pages 2951–2959, 2012.

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

[59] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

[60] Yanan Sui, Alkis Gotovos, Joel Burdick, and Andreas Krause. Safe exploration for optimization with gaussian processes. In *International Conference on Machine Learning*, pages 997–1005. PMLR, 2015.

[61] Richard S Sutton, Hamid Reza Maei, Doina Precup, Shalabh Bhatnagar, David Silver, Csaba Szepesvári, and Eric Wiewiora. Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *International Conference on Machine Learning*, pages 993–1000, 2009.

[62] Richard S Sutton, A Rupam Mahmood, and Martha White. An emphatic approach to the problem of off-policy temporal-difference learning. *The Journal of Machine Learning Research*, 17(1):2603–2631, 2016.

[63] Yuval Tassa, Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, and Nicolas Heess. dm_control: Software and tasks for continuous control, 2020.

[64] Georgios Theocharous, Philip S Thomas, and Mohammad Ghavamzadeh. Personalized ad recommendation systems for life-time value optimization with guarantees. In *International Joint Conference on Artificial Intelligence*, 2015.

[65] Masatoshi Uehara, Jiawei Huang, and Nan Jiang. Minimax weight and Q-function learning for off-policy evaluation. In *Proceedings of Machine Learning and Systems*, pages 1023–1032. 2020.

[66] Ziyu Wang, Alexander Novikov, Konrad Żołna, Jost Tobias Springenberg, Scott Reed, Bobak Shahriari, Noah Siegel, Josh Merel, Caglar Gulcehre, Nicolas Heess, and Nando de Freitas. Critic regularized regression. In *Advances in Neural Information Processing Systems*, 2020.

[67] Mengjiao Yang, Bo Dai, Ofir Nachum, George Tucker, and Dale Schuurmans. Offline policy selection under uncertainty. *arXiv preprint arXiv:2012.06919*, 2020.

[68] Mengjiao Yang, Ofir Nachum, Bo Dai, Lihong Li, and Dale Schuurmans. Off-policy evaluation via the regularized lagrangian. *arXiv preprint arXiv:2007.03438*, 2020.

[69] Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: Model-based offline policy optimization. *arXiv preprint arXiv:2005.13239*, 2020.

[70] Michael R Zhang, Tom Le Paine, Ofir Nachum, Cosmin Paduraru, George Tucker, Ziyu Wang, and Mohammad Norouzi. Autoregressive dynamics models for offline policy evaluation and optimization. *arXiv preprint arXiv:2104.13877*, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See last paragraph of section 4 and section 5
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See section 5
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A] We do not include theoretical results.
   (b) Did you include complete proofs of all theoretical results? [N/A] We do not include theoretical results.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] The RL Unplugged dataset and the polices inlucded in DOPE benchmark are open sourced but the MPG environment and the policy selection code are proprietary. We have included detailed description of the experiment setting and our algorithm for reproducing the experiments in the DM Control Suite environments.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See subsection 4.2 and the appendix.

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See subsection 4.3.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See the policy selection settings in section 4.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

   (a) If your work uses existing assets, did you cite the creators? [Yes] We cite the code: Tensorflow, Acme, Reverb, environments: Mujoco, DM Control Suite, datasets: RL Unplugged, and policies: DOPE.

   (b) Did you mention the license of the assets? [Yes] Since we depend on many open source assets, we do not list the licenses in the main text, but we list them here. Tensorflow - Apache License 2.0, Acme - Apache License 2.0, Reverb - Apache License 2.0, DM Control Suite - Apache License 2.0, RL Unplugged - Apache License 2.0, DOPE - Apache License 2.0, Mujoco - Commercial, see their website for more details.

   (c) Did you include any new assets either in the supplemental material or as a URL? [No]

   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] We are using open sourced, commercial, and our own assets.

   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] We do not use data involving people.

5. If you used crowdsourcing or conducted research with human subjects...

   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]