Bandit Overfitting in Offline Policy Learning

David Brandfonbrener **William F. Whitney** **Rajesh Ranganath** **Joan Bruna**
New York University

Abstract

We study the offline policy learning problem in a contextual bandit framework. Specifically, we focus on the issue of overfitting which is especially important in a modern context where we often use overparameterized models that can interpolate the data. Our first contribution is to introduce a regret decomposition into approximation, estimation, and bandit errors that emphasizes the distinction between the policy learning and supervised learning problems. The bandit error measures the error from overfitting to the single action observed at each context, which we call "bandit overfitting". Our second contribution is to show both in theory and experiments how bandit overfitting is different for policy-based versus value-based algorithms when we use overparameterized models. We find that bandit overfitting can become a severe problem for policy-based algorithms, but value-based algorithms effectively reduce the policy learning problem to regression and thus avoid the worst problems of bandit overfitting.

1 Introduction

In the offline policy learning problem, we are given a dataset of context, action, reward tuples collected by some behavior policy, and the goal is to learn a new policy which maximizes the expected reward. The problem can represent many decision making applications. For example, problems in recommender systems (Li et al., 2010; Bottou et al., 2013), robotics (Pinto and Gupta, 2016), and healthcare decision making (Prasad et al., 2017; Raghu et al., 2017) can all be cast as offline policy learning problems. In these domains it is often critical to be able to learn the policy offline without deploying exploratory policies in the real world.

We focus on understanding the offline policy learning problem better by focusing on the issue of overfitting. Understanding overfitting is especially important in a modern context where we often use overparameterized models that are capable of interpolating the training data. We show that there are important distinctions in (1) how overfitting impacts policy learning versus supervised learning and (2) how overfitting impacts policy-based versus value-based algorithms.

To frame our discussion, we introduce a novel regret decomposition into approximation, estimation, and "bandit" errors. The bandit error captures the error that is due to only observing the action selected by the behavior policy at each point in the dataset. This decomposition is conceptually useful because it allows us to disentangle the the estimation error which captures overfitting due to noise in the rewards from the bandit error due to the actions. This gives us a notion of bandit overfitting which extends the idea of "propensity overfitting" from Swaminathan and Joachims (2015b); Joachims et al. (2018).

Armed with this framework, we compare policy-based and value-based algorithms in the overparameterized setting. We show that policy optimization can suffer from bandit overfitting where bandit error dominates the regret, and can fail to recover the optimal policy in the limit of infinite data with nonparametric models, even with the addition of constant baselines. In contrast, value-based learning effectively reduces the problem to regression where recent work demonstrates that overparameterized regression generalizes well under mild assumptions on the target function.

Finally, we experimentally confirm that the intuitions from the theory hold beyond the strict settings of the theory itself. Specifically, we look at neural network models on toy problems, problems with real images as contexts, and a simulated economic problem. We find that indeed policy-based algorithms suffer from bandit overfitting more than their value-based counterparts in these overparameterized settings.

Preprint.

2 Setup

2.1 Offline contextual bandit problem

First we will define the online contextual bandit problem (Langford and Zhang, 2008). Let the context space $\mathcal{X}$ be infinite and the action space $\mathcal{A}$ be finite with $|\mathcal{A}| = K < \infty$. At each round, a context $x \in \mathcal{X}$ and a full feedback reward vector $r \in [r_{\min}, r_{\max}]^K$ are drawn from a joint distribution $\mathcal{D}$. Note that r can be dependent on x since they are jointly distributed. A policy π maps contexts to distributions over actions, $\pi : \mathcal{X} \to \mathcal{P}(\mathcal{A})$. An action a is sampled according to $\pi(a|x)$ and the reward on the round is $r(a)$, the component of the vector r corresponding to a. We use "bandit feedback" to refer to only observing $r(a)$. This contrasts with the "full feedback" problem where at each round the full vector of rewards r is revealed, independent of the action.

In the offline setting we get finite dataset of N rounds with a fixed behavior policy β. Then we denote the dataset as $S = \{x_i, r_i, a_i, p_i\}_{i=1}^N$ where p_i is the observed propensity $p_i = \beta(a_i|x_i)$. The tuples in the datasets lie in $\mathcal{X} \times [r_{min}, r_{max}]^K \times \mathcal{A} \times [0, 1]$ and are drawn i.i.d from the joint distribution induced by $\mathcal{D}$ and β. From S we define the datasets S_B for bandit feedback and S_F for full feedback:

$$S_B = \{(x_i, r_i(a_i), a_i, p_i)\}_{i=1}^N, \quad S_F = \{(x_i, r_i)\}_{i=1}^N.$$

The goal is to take in a dataset and produce a policy π so as to maximize the value $V(\pi)$ defined as

$$V(\pi) := \mathbb{E}_{x,r\sim\mathcal{D}}\mathbb{E}_{a\sim\pi(\cdot|x)}[r(a)].$$

We will use π^* to denote the policy that maximizes V. Finally, define the Q function at a particular context, action pair as

$$Q(x, a) := \mathbb{E}_{r|x}[r(a)].$$

2.2 Algorithms

Now that we have defined the problem, we define the main families of algorithms that we will analyze. This is not meant to be a comprehensive account of all algorithms, but a broad strokes picture of the vanilla versions of the most popular algorithms.

Supervised learning with full feedback. In a full feedback problem, empirical value maximization (the analog to standard empirical risk minimization) is defined by maximizing the empirical value $\hat{V}_F$:

$$\hat{V}_F(\pi; S_F) := \frac{1}{N}\sum_{i=1}^N \langle r_i, \pi(\cdot|x_i)\rangle \tag{1}$$

$$\pi_F := \arg\max_{\pi\in\Pi} \hat{V}_F(\pi; S_F). \tag{2}$$

Importance weighted policy optimization. In the bandit problem, importance weighted policy optimization directly optimizes the policy to maximize an estimate of the value. Since we only observe the rewards of the behavior policy, we use importance weighting to get an unbiased value estimate to maximize. Explicitly:

$$\hat{V}_B(\pi; S_B) := \frac{1}{N}\sum_{i=1}^N r_i(a_i)\frac{\pi(a_i|x_i)}{p_i} \tag{3}$$

$$\pi_B := \arg\max_{\pi\in\Pi} \hat{V}_B(\pi; S_B). \tag{4}$$

Note that this is the "vanilla" version of the algorithm and modifications like regularizers, baselines, and clipped importance weights have been proposed (Bottou et al., 2013; Dudík et al., 2011; Joachims et al., 2018; Strehl et al., 2010; Swaminathan and Joachims, 2015a,b). The most relevant modification for our analysis is the introduction of constant baselines (Williams, 1992; Joachims et al., 2018). Adding a baseline b modifies the algorithm as follows:

$$\hat{V}_{B,b}(\pi; S_B) := \frac{1}{N}\sum_{i=1}^N (r_i(a_i) - b)\frac{\pi(a_i|x_i)}{p_i} \tag{5}$$

$$\pi_{B,b} := \arg\max_{\pi\in\Pi} \hat{V}_{B,b}(\pi; S_B). \tag{6}$$

This modification is discussed in Sections 5.2 and 6.2.

Value-based learning. Another simple algorithm is to first learn the Q function and then use a greedy policy with respect to this estimated Q function. Explicitly:

$$\hat{Q}_{S_B} := \arg\min_{f\in\mathcal{Q}} \sum_{i=1}^N (f(x_i, a_i) - r_i(a_i))^2 \tag{7}$$

$$\pi_{\hat{Q}_{S_B}}(a|x) := \mathbb{1}\left[a = \arg\max_{a'} \hat{Q}_{S_B}(x, a')\right]. \tag{8}$$

The RL literature also often defines a class of model-based algorithms, but in the contextual bandit problem there are no state transitions so model-based algorithms are equivalent to value-based algorithms.

3 Regret decomposition

In supervised learning, the standard decomposition of the excess risk separates the approximation and estimation error (Bottou and Bousquet, 2008). The approximation error is due to our limited function class and the estimation error is due to minimizing the empirical risk rather than the true risk. Since the full feedback policy learning problem is equivalent to supervised learning, the same decomposition applies. Formally, consider a full feedback algorithm $\mathcal{A}_F$ which

takes the dataset S_F and produces a policy π_F. Then

$$\underbrace{\mathbb{E}_S[V(\pi^*) - V(\pi_F)]}_{\text{regret}} = \underbrace{V(\pi^*) - \sup_{\pi\in\Pi} V(\pi)}_{\text{approximation error}} + \underbrace{\mathbb{E}_S[\sup_{\pi\in\Pi} V(\pi) - V(\pi_F)]}_{\text{estimation error}}.$$

We can instead consider a bandit feedback algorithm $\mathcal{A}_B$ which takes the dataset S_B and produces a policy π_B. To extend the above decomposition to the bandit problem we add a new term, the bandit error, that results from having access to S_B rather than S_F. Now we can decompose the regret as:

$$\underbrace{\mathbb{E}_S[V(\pi^*) - V(\pi_B)]}_{\text{regret}} = \underbrace{V(\pi^*) - \sup_{\pi\in\Pi} V(\pi)}_{\text{approximation error}} + \underbrace{\mathbb{E}_S[\sup_{\pi\in\Pi} V(\pi) - V(\pi_F)]}_{\text{estimation error}} + \underbrace{\mathbb{E}_S[V(\pi_F) - V(\pi_B)]}_{\text{bandit error}}.$$

Disentangling sources of error. The approximation error is the same quantity that we encounter in the supervised learning problem, measuring how well our function class can do. The estimation error measures the error due to overfitting on finite contexts and noisy rewards. The bandit error accounts for the error due to only observing the actions chosen by the behavior policy. This is not quite analogous to overfitting to noise in the rewards since stochasticity in the actions is actually required to have the coverage of context-action pairs needed to learn a policy. While the standard approximation-estimation decomposition could be directly extended to the bandit problem, our approximation-estimation-bandit decomposition is more conceptually useful since it disentangles these two types of error that will affect an algorithm.

When is bandit error positive? Usually, we think about an error decomposition as having each term be positive. This is not necessarily the case with our decomposition, but we view this as a feature rather than a bug. Intuitively, the bandit error term captures the contribution of the actions selected by the behavior policy. If the behavior policy is nearly optimal and the rewards are highly stochastic, there may be more signal in the actions selected by the behavior policy than the observed rewards. Thus overfitting the actions chosen by behavior policy can sometimes be beneficial, causing the bandit error to be negative. The two terms disentangle the approximation error (due to reward noise) from bandit error (due to behavior actions).

4 Motivating example

Using the definitions and regret decomposition, in this section we illustrate the problem that we call *bandit overfitting*. Essentially, bandit overfitting happens when a policy is able to maximize its objective by fitting the actions taken by the behavior policy rather than maximizing the rewards. This causes the bandit error term to be large. We will show that bandit overfitting is most pronounced in policy optimization since the objective encourages the learned policy to imitate the behavior policy whenever the rewards are positive.

The core issue can be seen in the following toy example, illustrated in Figure 1. Take a bandit problem with two actions (called 1 and 2) with constant rewards of 1 and 2 respectively. Let the context distribution be uniform over the interval [-1,1], and the behavior policy be uniform across the two actions at every context:

$$x \sim U([-1,1]), \quad r|x = (1,2), \quad \beta(a|x) = 0.5 \;\forall\; x, a.$$

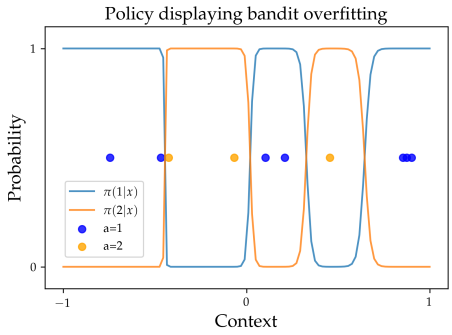

Figure 1: Dots represent datapoints, colored by which action was observed at that context. Blue dots have reward of 1 and orange dots have reward 2. The policy π is a neural net trained to perfectly maximize $\hat{V}_B$.

The true optimal policy π^* chooses action 2 for all x and $V(\pi^*) = 2$. But, a policy π_B which copies the observed actions by setting $\pi_B(a_i|x_i) = 1$ for all i will maximize $\hat{V}_B$ despite having a lower true value. In contrast, with full feedback the algorithm sees both actions at every context, so $\hat{V}_F$ is maximized by always correctly choosing action 2.

To demonstrate this intuition precisely, assume that exactly half of the observed points have action 1 and half have action 2. Then

$$\hat{V}_B(\pi_B; S_B) = \frac{1}{N}\sum_{i=1}^{N} r_i(a_i)\frac{\pi_B(a_i|x_i)}{1/2} = \frac{2}{N}\sum_{i=1}^{N} r_i(a_i)$$
$$= \frac{2}{N}\left(\frac{N}{2}(1) + \frac{N}{2}(2)\right) = 3.$$

On the other hand $\hat{V}_B(\pi^*; S_B) = 2$.

In terms of true expected value, the policies that always choose action 2 will have value of 2: $V(\pi^*) = 2$ and $V(\pi_F) \approx 2$. But choosing action 1 half the time with π_B means that we expect $V(\pi_B) \approx 1.5$. Thus, in this problem, we expect the bandit error to be approximately 0.5. This example shows how even without noise in the rewards, policy optimization can suffer from bandit overfitting due to the observed actions.

5 Theoretical results

In this section we present our main theoretical results in the overparameterized setting which (1) illustrate the differences between the supervised learning problem and the policy learning problem, and (2) show that policy-based algorithms can have a serious problem with bandit overfitting, and (3) show that bandit overfitting is less severe in value-based algorithms. Throughout this section we will make the following assumption of strict positivity.

Assumption 1 (Strict positivity)**.** *We have strict positivity if $\beta(a|x) \geq \tau > 0$ for all a, x. Thus, in any dataset we will have $p_i = \beta(a_i|x_i) \geq \tau > 0$.*

There is important work that focuses on making successful algorithms when τ is very small or even 0 by making algorithmic modifications like clipping (Bottou et al., 2013; Strehl et al., 2010; Swaminathan and Joachims, 2015a) and behavior constraints (Fujimoto et al., 2018). However, these issues are orthogonal to the main contribution of our paper, so we focus on the setting with strict positivity.

Before exploring what happens with overparameterized models, it is important to recall the results for small model classes in this setting (where small means either finite or small VC dimension). With small model classes, Strehl et al. (2010) and Chen and Jiang (2019) show regret bounds for policy optimization and value-based learning respectively. These bounds scale with $\frac{1}{\sqrt{N}}$ and the logarithm of the size of the model class as in supervised learning. They also depend on $\frac{1}{\tau}$, since with bandit feedback, the effective sample size is smaller by a factor of τ. Value-based learning has worse dependence on model misspecification since the algorithm does not directly optimize an estimate of the target objective. For completeness, we provide the full formal results in Appendix F and explicitly bound each term of the regret decomposition for each algorithm.

While these results are very clean, they are not useful in the modern setting where overparameterized neural networks give superior empirical performance. As illustrated empirically by Zhang et al. (2016) and theoretically by Nagarajan and Kolter (2019), modern neural networks are sufficiently overparameterized to interpolate even noisy labels and such uniform convergence bounds are rendered vacuous.

In this paper, we focus on the large model classes that arise in modern machine learning which can interpolate the dataset. Formally, an interpolator is a function approximator that will exactly optimize the objective function at every datapoint in the training set. When the objective $L(\theta)$ can be decomposed into a sum over datapoints z_i as $\frac{1}{N}\sum_{i=1}^{N} \ell(\theta, z_i)$, an interpolator can find θ to exactly minimize every $\ell(\theta, z_i)$.

5.1 Vanilla policy optimization

In this subsection we will show that bandit error can become a severe problem under interpolation for policy optimization without a proper baseline.

To understand how interpolators behave in this problem, we first need to understand what happens at the observed contexts in the dataset. The following lemma shows how to choose an optimal action at each observed context. The proof is in Appendix A.

Lemma 1 (Interpolating action)**.** *Define*

$$a_B(i) = \begin{cases} a_i & r_i(a_i) > 0 \\ \text{any } a \neq a_i & \text{otherwise.} \end{cases}$$

Let $\pi_B(a|x_i) = \mathbb{1}[a = a_B(i)]$, then

$$\sup_{\pi} \hat{V}_B(\pi) = \hat{V}_B(\pi_B).$$

This tells us that the signal being provided by the objective is not helping to recover the optimal policy. Rather, policy optimization is only copying the observed action up to the sign of the observed reward.

For the remainder of this subsection, we focus on using the Lemma to prove lower bounds on the regret of interpolating policy optimization when we generalize away from the data. To simplify the results, we consider the case with two actions, $K = 2$. Note that since we are proving lower bounds on performance, taking $K = 2$ is the most difficult case. Adding more actions will make problem instances harder which would make proving lower bound easier.

Our first regret bound considers perhaps the simplest interpolator, a nearest neighbor policy. We use this to illustrate that even with noiseless rewards and infinite data, bandit overfitting can prevent an interpolating policy class from finding the optimal policy. Moreover, we demonstrate the dependence on the behavior policy by incorporating the probability that the behavior policy chooses an optimal action. As hinted at in the example above, when the rewards are positive the policy is encouraged to clone the behavior and when the

rewards are negative the policy will anti-clone the behavior. This tendency allows us to construct problems much like the motivating example where nearest neighbor policies will fail to recover the optimal policy since they fit the behavior actions rather than the rewards.

Theorem 1 (One nearest neighbor)**.** *Assume $K = 2$, noiseless rewards, and that π^* is a piecewise continuous function of x. Let $\Delta_r = r_{\max} - r_{\min}$. Let π_B, π_F be defined by one nearest neighbor rules that interpolate their respective objectives. Let $p^*_\beta = P_{x,a_\beta \sim \beta|x, a^* \sim \pi^*|x}(a_\beta = a*)$ be the probability that the behavior policy chooses the optimal action. Then there exist problem instances where*

$$\limsup_{N\to\infty} \mathbb{E}_S[V(\pi_F) - V(\pi_B)] = \Delta_r \max\{p^*_\beta, 1 - p^*_\beta\}.$$

But, for all problem instances

$$\limsup_{N\to\infty} \mathbb{E}_S[V(\pi^*) - V(\pi_F)] = 0.$$

The proof is in Appendix A. This result shows that using a nearest neighbor scheme to generalize based on the signal provided by the bandit policy optimization objective is not sufficient to learn an optimal policy. It also shows that problems can penalize either good behavior policies or bad behavior policies. We verify this intuition empirically with neural network approximators in Section 6.

To move to general function classes, we will now present a result that reduces lower bounding the bandit error of policy optimization to lower bounding the gap between noisy and noiseless classification. Since the result is a statement about the objectives themselves, it applies to all function classes including overparameterized ones. In the overparameterized case, we expect noisy classification to be especially hard since the model will be able to fit the noise in the labels. Any hope of generalization will fall back on the inductive biases since we do not have the constraints of a small model class to help ignore the noise. But empirically, this does not happen as seen in Zhang et al. (2016) where neural nets trained on noisy labels are not able to generalize.

Our theorem relies on taking any classification problem and constructing a bandit problem with deterministic rewards where the optimal policy is equivalent to the optimal classifier. By choosing the behavior policy to be a noisy version of the optimal policy, the bandit policy optimization objective becomes an instance of classification with noisy training labels. The proof can be found in Appendix A.

Theorem 2 (Noisy classification reduction)**.** *Take any noise level $\eta < 1/2$ and any binary classification problem $\mathcal{C}$ consisting of a distribution $\mathcal{D}_\mathcal{C}$ over $\mathcal{X}$ and a labeling function $y_\mathcal{C} : \mathcal{X} \to \{-1, 1\}$. There exists an offline contextual bandit problem $\mathcal{B}$ with noiseless rewards such that*

1. *Maximizing $\hat{V}_B$ in $\mathcal{B}$ is equivalent to minimizing the 0/1 loss on a training set drawn from $\mathcal{C}$ where labels are flipped with probability η.*
2. *Maximizing $\hat{V}_F$ in $\mathcal{B}$ is equivalent to minimizing the 0/1 loss on a training set drawn from $\mathcal{C}$ with noiseless training labels.*

This theorem tells us that even in noiseless reward problems, finding an optimal policy by optimizing $\hat{V}_B$ is at least as hard as solving a classification problem when only given access to noisy labels. However, if we were given full feedback in that bandit problem, finding the optimal policy by optimizing $\hat{V}_F$ is as easy as solving a classification problem with clean labels.

The important consequence of the theorem is that the bandit error is lower bounded by the gap between the risk of classifiers trained on noisy versus clean labels. This applies to any function class, including those that are overparameterized. In the overparameterized case, any optimizer of the objective will exactly fit the noise so any hope for generalization must come from inductive bias of the learning algorithm. While we are not familiar with any theoretical lower bounds on this problem, empirically models trained to interpolate noisy labels do not perform well (Zhang et al., 2016).

In this subsection we have shown that the policy optimization objective provides a poor signal at each datapoint independently. As a consequence, a nearest neighbor policy cannot recover the optimal policy even with noiseless rewards and infinite data. More generally, for any overparameterized policy class to perform well with policy optimization it must at least have a strong enough inductive bias to be able to solve classification problems while interpolating noisy labels.

5.2 Policy optimization with baselines

Now we show that the introduction of a constant baseline as in Equation (5) can help, but not solve the issues of bandit overfitting in interpolating models.

By incorporating such a baseline, the algorithm shifts the rewards. Looking back at Lemma 1, shifting the rewards means that an interpolating policy for $\hat{V}_{B,b}$ will choose a_i whenever $r_i(a_i) - b > 0$. If the baseline exactly ensures that the optimal action has positive reward while non-optimal actions have negative reward, then interpolating $\hat{V}_{B,b}$ can indeed provide a signal to match the optimal policy. However, as the following theorem shows, introducing a constant baseline is in

general not sufficient to prevent bandit overfitting. The proof is similar to the proof without a baseline but requires a slightly more complicated reward function so that no baseline can perfectly identify the optimal action.

Theorem 3 (Baselines). *Assume $K = 2$, noiseless rewards, and $\Delta_r = r_{\max} - r_{\min}$. Let $\pi_{B,b}$ be defined by a one nearest neighbor rule to interpolate $\hat{V}_{B,b}$. Let $p^*_\beta = P_{x,a_\beta\sim\beta|x,a^*\sim\pi^*|x}(a_\beta = a*)$ be the probability that the behavior policy chooses the optimal action. Then there exist problem instances such that for any choice of baseline b and any $\epsilon > 0$,*

$$\limsup_{N\to\infty} \mathbb{E}_S[V(\pi_F) - V(\pi_{B,b})] \geq \frac{1}{4}(\Delta_r - \epsilon)\min\{p^*_\beta, 1 - p^*_\beta\}.$$

This theorem can be compared directly with Theorem 1. Doing this we notice that the introduction of the baseline reduced the worst case asymptotic regret by a factor of 4 and improves the dependence on the behavior policy. The dependence on the behavior policy is improved since with an optimal baseline the effective rewards are never strictly positive or strictly negative, so the interpolating policy is not forced to clone or anti-clone the behavior policy. While the baseline clearly helps, it does not solve the problem.

While traditionally the use of baselines is motivated as a method of variance reduction (Greensmith et al., 2004), our work instead suggests that baselines can be necessary for consistency, i.e. recovering the optimal policy in the limit of infinite data.

An extension of baselines that is common in reinforcement learning is to learn a context-dependent baseline function, typically the expected value in a given context (Sutton and Barto, 2018). While a full examination of learned baselines is beyond the scope of this paper, we discuss them at more length in Appendix B.

5.3 Value-based learning

While policy optimization can be sensitive to bandit overfitting, we will show that value-based learning is not when we make some structural assumptions on the true Q function. The following theorem reduces bounding the regret of value-based learning to bounding the generalization of the learned Q function. This makes work about generalization of interpolating regression directly applicable.

Theorem 4 (Reduction to regression). *Assuming strict positivity, then with $\hat{Q}_{S_B}$ as defined in (7) then*

$$V(\pi^*) - V(\pi_{\hat{Q}}) \leq \frac{2}{\sqrt{\tau}}\sqrt{\mathbb{E}_{x,a\sim\beta}[(Q(x,a) - \hat{Q}_{S_B}(x,a))^2]}.$$

A proof can be found in Appendix C. Similar results are presented as intermediate results in Chen and Jiang (2019); Munos and Szepesvári (2008). The implication of this result that we want to emphasize is that any generalization guarantees for overparameterized regression immediately become guarantees for value-based learning in offline contextual bandits. The following results demonstrate a few of these guarantees, which all require some sort of regularity assumption on the true Q function to bound the regression error.

Interpolating regression guarantees. The results of Cover (1968) imply the consistency of a one nearest neighbor regressor when the rewards are noiseless and Q is piecewise continuous. This contrasts nicely with Theorem 1. The results of Bartlett et al. (2020) give finite sample rates for overparameterized linear regression by the minimum norm interpolator depending on the covariance matrix of the data and assuming that the true function is realizable. The results of Belkin et al. (2019) imply that under smoothness assumptions on Q, a particular singular kernel will interpolate the data and have optimal non-parametric rates. After applying our reduction, the rates are no longer optimal for the policy learning problem. The results of Bach (2017) show how choosing the minimum norm infinite width neural network in a particular function space can yield adaptive finite sample guarantees for many types of underlying structure in the Q function.

What makes value learning different? The above formal results show a gap between policy-based and value-based learning. Now we attempt to provide an intuitive explanation for this gap. The key difference between parameterizing a policy and a Q function is that the policy must produce a normalized distribution over actions. When combined with interpolation and bandit feedback, this normalization becomes a serious liability. Formally, consider a single datapoint $x, a, r(a)$ in the case of positive rewards and interpolating policy classes. Then, letting e_a be the standard basis vector:

$$\pi_B(\cdot|x) = \arg\max_{p\in\Delta^K} \frac{r(a)}{\beta(a|x)}p(a) = e_a$$
$$\hat{Q}_{S_B}(a|x) = \arg\min_{q\in\mathbb{R}}(r(a) - q)^2 = r(a).$$

The behavior of $\hat{Q}_{S_B}$ at actions other than a is not controlled by the behavior at a, while the behavior of π_B is highly dependent across actions. Fundamentally, value-based learning attempts to model the outcomes of counterfactual actions by generalizing from nearby contexts for each action. On the other hand, policy optimization generalizes both across actions via normalization and across contexts via conditioning on x.

6 Experiments

The above theory primarily considers interpolating classes where the size of the class increases with the data. However, the practical regime that we care about is not exactly covered by the theory: a fixed finite dataset and an overparameterized neural network approximator. In this section we see that the theoretical results are borne out by neural nets.

6.1 Bandit overfitting and the behavior

First we consider the same toy problem with two actions from Section 4, where the actions have constant rewards of 1 and 2 respectively. We use this to empirically confirm the results of Theorem 1 with a neural network policy. We construct datasets by defining a class of behavior policies by splitting the domain into three sections and setting $\beta(1|x) = p_i$ when x is in the ith section. We choose $p_i \in \{0.05, 0.35, 0.65, 0.95\}$. This gives us 64 policies of varying quality (values between 1.05 and 1.95) with strict positivity ($\tau \geq 0.05$). For each behavior policy we sample a training set of 20 training points and report the value of the learned policy on a held out test set of 1000 points. To parameterize the policies we use a one layer neural network with width 512 as our function approximator and use a 5 dimensional encoding of the features in the frequency domain. Full details are in Appendix D.

We consider two settings, one where the rewards are all positive and one where the rewards are all negative. Results are shown in Figure 2. We find that the value of the policy learned by policy optimization clones the behavior policy with positive rewards and anti-clones the behavior with negative rewards, as the theory would predict. These results confirm that the insights from Theorem 1 transfer to neural network policies.

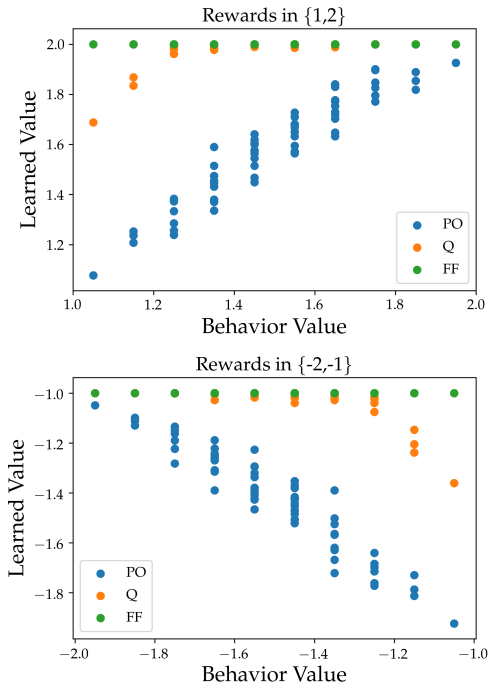

Figure 2: Value evaluated on the test set for policies trained by policy optimization (PO), value-based learning (Q), and learning with full feedback (FF). Higher is better.

6.2 Bandit overfitting with baselines

Classification: CIFAR-10. In some problems, coming up with a baseline that reduces bandit overfitting is simple. The most obvious of these is a classification problem where we know that rewards only take two values and for every context only one action gets higher reward. To compare to prior work, we will consider a bandit version of CIFAR-10 (Krizhevsky, 2009) as in Joachims et al. (2018).

To turn CIFAR into an offline bandit problem we view each possible label as an action and assign reward of 2 for a correct label/action and 1 for an incorrect label/action. We use 2 and 1 rather than 1 and 0 to better illustrate the effect of baselines. We use two different behavior policies to generate training data: (1) the hand-crafted policy used in (Joachims et al., 2018) and (2) a uniformly random behavior policy. We train Resnet-18 (He et al., 2016) models using Pytorch (Paszke et al., 2019). As in Joachims et al. (2018) we consider a hyperparameter search over constant baselines. Full details about the training procedure are in Appendix D and results are illustrated in Table 1.

	HC	**Uniform**	**FF**
vanilla PO	0.846	0.792	-
PO w/baseline	0.095	0.271	-
Value-based	**0.080**	**0.143**	-
Supervised	-	-	**0.058**

Table 1: Regret of the policies learned by each different algorithm on CIFAR-10 (lower is better, zero is optimal). The columns show different datasets: hand crafted behavior policy (HC), uniform behavior policy, and full feedback (FF). The rows show different algorithms: policy optimization with no baseline (vanilla PO), policy optimization with a tuned baseline, value-based learning, and supervised learning.

These results show that baselines can dramatically help in classification problems, but that value-based algorithms outperform policy optimization, especially when the behavior policy is more stochastic. Note that our results for policy optimization with a tuned

baseline on the hand-crafted policy outperform those reported by Joachims et al. (2018), but are still worse than value-based learning.

Continuous reward: World3. To get a bandit problem with richer structure than a classification problem, but where we still have access to counterfactual outcomes, we consider a simulated experiment from the `whynot` package (Miller et al., 2020). Specifically, we consstruct a contextual bandit problem using the World3 simulator which is a differential equation based model with 12 state variables and 11 simulation parameters. While the exact details of the simulator are unimportant, the reward is a continuous variable with much richer structure than a classification problem. Full details about the simulator and our experiment can be found in Appendix D.

Briefly, we sample a training set of 1000 datapoints using a uniformly random behavior policy and plot the regret on a test set of 5000 points of policy optimization across baselines that cover the range of the rewards. We include flat lines value-based and full feedback (which do not depend on this hyperparameter), as well as a uniformly random policy.

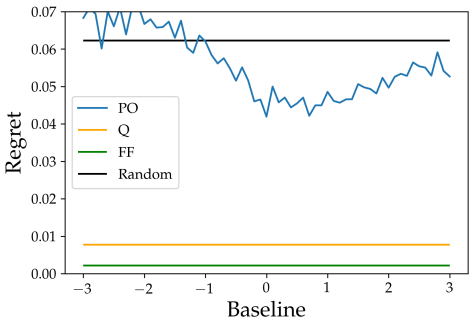

Figure 3: Regret of policy optimization across different choices of baseline (lower is better, zero is optimal).

This shows how baselines can provide some help against bandit overfitting, but policy optimization is still several times worse than value-based learning in this problem even with this hyperparameter tuning. When the rewards vary more widely in scale and correlate with the contexts, simple baselines are insufficient to combat bandit overfitting.

7 Related work

Our concept of bandit error and the resulting phenomena of "bandit overfitting" builds on the idea of "propensity overfitting" raised by Swaminathan and Joachims (2015b); Joachims et al. (2018). Specifically, we provide a more formal definition of the problem via our decomposition and consider the overparameterized model setting. We also show how their proposed solution of constant baselines can be deficient and consider value-based learning algorithms as an alternative (which they do not do in that work). See Appendix E for a longer discussion of the connection to propensity overfitting and an example that shows how the solution of self-normalized estimates proposed by Swaminathan and Joachims (2015b); Joachims et al. (2018) does not solve the bandit overfitting problem.

The offline policy learning problem has been well studied under finite and small VC dimension model classes in the bandit community (Strehl et al., 2010; Swaminathan and Joachims, 2015a,b; Joachims et al., 2018). Similar work has also come out of the causal inference community (Bottou et al., 2013; Athey and Wager, 2017; Kallus, 2018; Zhou et al., 2018). Related work has also come out of the RL theory community for the more general full RL problem (Munos and Szepesvári, 2008; Chen and Jiang, 2019). All these results rely on having a small policy class and then applying standard ideas of uniform convergence. In this work, we instead consider a modern setting where our very large model classes render such bounds vacuous and consider the type of overfitting that emerges in this setting.

Our regret decomposition extends prior work from supervised learning that decomposes excess risk into estimation and approximation error (Vapnik, 1982; Bottou and Bousquet, 2008) by adding the bandit error.

8 Discussion

We have examined overfitting in the offline contextual bandit problem. We introduced a new regret decomposition to separate the effects of estimation error and bandit error and showed that policy-based algorithms can be severely harmed by bandit error when using interpolating models while value-based algorithms are more robust in our setting.

It is important to emphasize that our results may not apply beyond the setting we consider in this paper. Explicitly, when there is no strict positivity, there is unobserved confounding, there are very many or continuous actions, or the model classes are small and misspecified then policy optimization may have lower regret and lower bandit error than value-based learning.

In future work we hope to extend the ideas from the bandit setting to the full RL problem with longer horizon that requires temporal credit assignment. We predict that bandit overfitting remains a significant issue there. We also hope to leverage some of the theoretical understanding from this paper into algorithmic improvements to combat bandit overfitting.

Acknowledgements

We would like to thank Aahlad Puli for thoughtful conversations and Aaron Zweig, Min Jae Song, and Evgenii Nikishin for comments on earlier drafts.

This work is partially supported by the Alfred P. Sloan Foundation, NSF RI-1816753, NSF CAREER CIF 1845360, NSF CHS-1901091, Samsung Electronics, and the Institute for Advanced Study. DB is supported by the Department of Defense (DoD) through the National Defense Science & Engineering Graduate Fellowship (NDSEG) Program.

References

Susan Athey and Stefan Wager. Efficient policy learning. *arXiv preprint arXiv:1702.02896*, 2017.

Francis Bach. Breaking the curse of dimensionality with convex neural networks. *The Journal of Machine Learning Research*, 18(1):629–681, 2017.

Peter L Bartlett, Philip M Long, Gábor Lugosi, and Alexander Tsigler. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 2020.

Mikhail Belkin, Alexander Rakhlin, and Alexandre B Tsybakov. Does data interpolation contradict statistical optimality? In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1611–1619. PMLR, 2019.

Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *Advances in neural information processing systems*, pages 161–168, 2008.

Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, Denis X Charles, D Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. Counterfactual reasoning and learning systems: The example of computational advertising. *The Journal of Machine Learning Research*, 14(1): 3207–3260, 2013.

Jinglin Chen and Nan Jiang. Information-theoretic considerations in batch reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning*. PMLR, 2019.

Minmin Chen, Ramki Gummadi, Chris Harris, and Dale Schuurmans. Surrogate objectives for batch policy optimization in one-step decision making. In *Advances in Neural Information Processing Systems*, pages 8825–8835, 2019.

T Cover. Estimation by the nearest neighbor rule. *IEEE Transactions on Information Theory*, 14(1): 50–55, 1968.

T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

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

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. *arXiv preprint arXiv:1812.02900*, 2018.

Evan Greensmith, Peter L Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5(Nov):1471–1530, 2004.

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

Thorsten Joachims, Adith Swaminathan, and Maarten de Rijke. Deep learning with logged bandit feedback. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=SJaP_-xAb`.

Nathan Kallus. Balanced policy evaluation and learning. In *Advances in Neural Information Processing Systems*, pages 8895–8906, 2018.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

John Langford and Tong Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in neural information processing systems*, pages 817–824, 2008.

Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, pages 661–670, 2010.

John Miller, Chloe Hsu, Jordan Troutman, Juan Perdomo, Tijana Zrnic, Lydia Liu, Yu Sun, Ludwig Schmidt, and Moritz Hardt. Whynot, 2020. URL `https://doi.org/10.5281/zenodo.3875775`.

Rémi Munos and Csaba Szepesvári. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9(May):815–857, 2008.

Vaishnavh Nagarajan and J Zico Kolter. Uniform convergence may be unable to explain generalization in deep learning. In *Advances in Neural Information Processing Systems*, pages 11615–11626, 2019.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in*

neural information processing systems, pages 8026–8037, 2019.

Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 3406–3413. IEEE, 2016.

N. Prasad, Li-Fang Cheng, C. Chivers, Michael Draugelis, and B. Engelhardt. A reinforcement learning approach to weaning of mechanical ventilation in intensive care units. *ArXiv*, abs/1704.06300, 2017.

Aniruddh Raghu, M. Komorowski, I. Ahmed, L. A. Celi, Peter Szolovits, and M. Ghassemi. Deep reinforcement learning for sepsis treatment. *ArXiv*, abs/1711.09602, 2017.

Alex Strehl, John Langford, Lihong Li, and Sham M Kakade. Learning from logged implicit exploration data. In *Advances in Neural Information Processing Systems*, pages 2217–2225, 2010.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Adith Swaminathan and Thorsten Joachims. Counterfactual risk minimization: Learning from logged bandit feedback. In *International Conference on Machine Learning*, pages 814–823, 2015a.

Adith Swaminathan and Thorsten Joachims. The self-normalized estimator for counterfactual learning. In *advances in neural information processing systems*, pages 3231–3239, 2015b.

Vladimir Vapnik. *Estimation of Dependences Based on Empirical Data: Springer Series in Statistics (Springer Series in Statistics)*. Springer-Verlag, Berlin, Heidelberg, 1982. ISBN 0387907335.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

Zhengyuan Zhou, Susan Athey, and Stefan Wager. Offline multi-action policy learning: Generalization and optimization. *arXiv preprint arXiv:1810.04778*, 2018.

Appendix

A Policy Optimization Proofs

A.1 Interpolating action

Lemma 1 (Interpolating action)**.** *Define*

$$a_B(i) = \begin{cases} a_i & r_i(a_i) > 0 \\ \text{any } a \neq a_i & \text{otherwise.} \end{cases}$$

Let $\pi_B(a|x_i) = \mathbb{1}[a = a_B(i)]$*, then*

$$\sup_\pi \hat{V}_B(\pi) = \hat{V}_B(\pi_B).$$

Proof. Expanding the definition of $\hat{V}_B$ and using the definitions of π_B and $a_B(i)$ from the theorem statement we have

$$\hat{V}_B(\pi_B) = \frac{1}{N}\sum_{i=1}^N r_i(a_i)\frac{\pi_B(a_i|x_i)}{p_i} = \frac{1}{N}\sum_{i=1}^N r_i(a_i)\frac{\mathbb{1}[a_i = a_B(i)]}{p_i} \tag{9}$$

$$= \frac{1}{N}\sum_{i=1}^N r_i(a_i)\frac{\mathbb{1}[r_i(a_i) > 0]}{p_i} = \frac{1}{N}\sum_{i=1}^N \sup_{p\in[0,1]} r_i(a_i)\frac{p}{p_i} \tag{10}$$

$$= \sup_\pi \frac{1}{N}\sum_{i=1}^N r_i(a_i)\frac{\pi(a_i|x_i)}{p_i} = \sup_\pi \hat{V}_B(\pi). \tag{11}$$

□

A.2 Nearest Neighbor

Theorem 1 (One nearest neighbor)**.** *Assume* $K = 2$*, noiseless rewards, and that* π^* *is a piecewise continuous function of* x*. Let* $\Delta_r = r_{\max} - r_{\min}$*. Let* π_B, π_F *be defined by one nearest neighbor rules that interpolate their respective objectives. Let* $p^*_\beta = P_{x,a_\beta\sim\beta|x,a^*\sim\pi^*|x}(a_\beta = a*)$ *be the probability that the behavior policy chooses the optimal action. Then there exist problem instances where*

$$\limsup_{N\to\infty} \mathbb{E}_S[V(\pi_F) - V(\pi_B)] = \Delta_r \max\{p^*_\beta, 1 - p^*_\beta\}.$$

But, for all problem instances

$$\limsup_{N\to\infty} \mathbb{E}_S[V(\pi^*) - V(\pi_F)] = 0.$$

Proof. First we need to formally define the nearest neighbor rules that interpolate the objectives $\hat{V}_B$ and $\hat{V}_F$. These are simple in the case of two actions. Let $i(x)$ be the index of the nearest neighbor to x in the dataset. Then

$$\pi_B(a|x) = \begin{cases} 1 & \big(a = a_{i(x)} \text{ AND } r_{i(x)}(a_{i(x)}) > 0\big) \text{OR} \big(a \neq a_{i(x)} \text{ AND } r_{i(x)}(a_{i(x)}) \leq 0\big) \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

This is saying that π_B chooses the same action as the observed nearest neighbor if that reward was positive, and the opposite action if that was negative. And for the full feedback we just choose the best action from the nearest datapoint.

$$\pi_F(a|x) = \begin{cases} 1 & a = \arg\max_{a'} r_{i(x)}(a') \\ 0 & otherwise. \end{cases} \tag{13}$$

Now, we will show that in the limit of infinite data, π_F has no regret. Since the rewards are noiseless, the maximum observed reward at a context is exactly the optimal action at that context. Thus, we precisely have a classification problem with noiseless labels so that the Bayes risk is 0. Since we assumed that π^* is piecewise continuous, the class conditional densities (determined by the indicator of the argmax of Q) are piecewise continuous. This allows us to apply the classic result of Cover and Hart (1967) that a nearest neighbor rule has asymptotic risk less twice the Bayes risk, which in this case is zero. This means that asymptotically $P(\pi_F(a|x) \neq \pi^*(a|x)) = 0$ which immediately gives the second desired result of zero regret in the limit of infinite data under full feedback.

The proof of the first result of non-vanishing bandit error consists of constructing problem instances that achieves this bandit error. We require two different constructions, one for $p^*_\beta < 1/2$ and one for $p^*_\beta \geq 1/2$. We will present the construction for $p^*_\beta < 1/2$ (i.e. a bad behavior policy) which uses positive rewards. The construction for $p^*_\beta \geq 1/2$ is analogous but with negative rewards.

To construct the example, take a bandit problem with two actions (called 1 and 2):

$$x \sim U([-1, 1]), \quad r|x = (1, 1 + \Delta_r), \quad \beta(2|x) = p^*_\beta \ \forall \ x, a$$

The true optimal policy has $\pi^*(2|x) = 1$ for all x and $V(\pi^*) = 1 + \Delta_r$. The policy with full feedback π_F is to always choose action 2, since every observation will show that action 2 is better.

Now we note that since rewards are always positive, we can simplify the definition of π_B as

$$\pi_B(a|x) = \mathbb{1}[a = a_{i(x)}]. \tag{14}$$

Then we have that

$$\begin{aligned} V(\pi_F) - V(\pi_B) &= \mathbb{E}_x[\mathbb{E}_{a\sim\pi_F|x}[Q(x,a)] - \mathbb{E}_{a\sim\pi_B|x}[Q(x,a)]] && (15)\\ &= \mathbb{E}_x[\Delta_r + 1 - (\pi_B(1|x) + \pi_B(2|x)(\Delta_r + 1))]] && (16)\\ &= \Delta_r + 1 - \mathbb{E}_x[\mathbb{1}[a_{i(x)} = 1] + (\Delta_r + 1)\mathbb{1}[a_{i(x)} = 2]] && (17) \end{aligned}$$

Taking expectation over S we get

$$\begin{aligned} \mathbb{E}_S[V(\pi_F) - V(\pi_B)] &= \mathbb{E}_S[\Delta_r + 1 - \mathbb{E}_x[\mathbb{1}[a_{i(x)} = 1] + (\Delta_r + 1)\mathbb{1}[a_{i(x)} = 2]]] && (18)\\ &= \Delta_r + 1 - \mathbb{E}_x[P_S(a_{i(x)} = 1) + (\Delta_r + 1)P_S(a_{i(x)} = 2)]] && (19)\\ &= \Delta_r + 1 - \mathbb{E}_x[(1 - p^*_\beta) + (\Delta_r + 1)p^*_\beta]] && (20)\\ &= (1 - p^*_\beta)\Delta_r && (21) \end{aligned}$$

This construction did not depend on the size of the dataset, so it is even true as the number of datapoints tends to infinity. The analogous construction for $p^*_\beta \geq 1/2$ gives bandit error of $p^*_\beta\Delta_r$ and we get the result by just choosing the worse larger bandit error depending on p^*_β. □

A.3 Noisy classification

Theorem 2 (Noisy classification reduction)**.** *Take any noise level $\eta < 1/2$ and any binary classification problem $\mathcal{C}$ consisting of a distribution $\mathcal{D}_\mathcal{C}$ over $\mathcal{X}$ and a labeling function $y_\mathcal{C} : \mathcal{X} \to \{-1, 1\}$. There exists an offline contextual bandit problem $\mathcal{B}$ with noiseless rewards such that*

1. *Maximizing $\hat{V}_B$ in $\mathcal{B}$ is equivalent to minimizing the 0/1 loss on a training set drawn from $\mathcal{C}$ where labels are flipped with probability η.*

2. *Maximizing $\hat{V}_F$ in $\mathcal{B}$ is equivalent to minimizing the 0/1 loss on a training set drawn from $\mathcal{C}$ with noiseless training labels.*

Proof. First we will construct the bandit problem $\mathcal{B}$ with two actions corresponding to the classification problem $\mathcal{C}$. For any constant $c_r > 0$ we define $\mathcal{B}$ by

$$x \sim \mathcal{D}_{\mathcal{C}}, \qquad r|x = \begin{cases} c_r(1-\eta, \eta) & y_{\mathcal{C}}(x) = 1 \\ c_r(\eta, 1-\eta) & y_{\mathcal{C}}(x) = -1 \end{cases}, \qquad \beta(1|x) = \begin{cases} 1-\eta & y_{\mathcal{C}}(x) = 1 \\ \eta & y_{\mathcal{C}}(x) = -1 \end{cases} \tag{22}$$

Now we will show that in this problem, $\hat{V}_B$ is equivalent to the $0/1$ loss for $\mathcal{C}$ with noisy labels. To do this first note that by construction, for x with $y_{\mathcal{C}}(x) = 1$ we have $\frac{r(1)|x}{\beta(1|x)} = \frac{c_r(1-\eta)}{1-\eta} = c_r$ and $\frac{r(2)|x}{\beta(2|x)} = \frac{c_r\eta}{\eta} = c_r$, and similarly for x with $y_{\mathcal{C}}(x) = -1$ we have $\frac{r(1)|x}{\beta(1|x)} = \frac{c_r\eta}{\eta} = c_r$ and $\frac{r(2)|x}{\beta(2|x)} = \frac{c_r(1-\eta)}{1-\eta} = c_r$.

$$\hat{V}_B(\pi) = \frac{1}{N}\sum_{i=1}^{N} r_i(a_i)\frac{\pi(a_i|x_i)}{\beta(a_i|x_i)} = \frac{1}{N}\sum_{i=1}^{N} \frac{r_i(a_i)}{\beta(a_i|x_i)}\pi(a_i|x_i) \tag{23}$$

$$= \frac{c_r}{N}\sum_{i=1}^{N} \pi(a_i|x_i) \tag{24}$$

This is equivalent to $0/1$ loss with noisy labels since β generates a_i according to $y_{\mathcal{C}}$ where the label is flipped with probability η.

Now we will show that $\hat{V}_F$ is equivalent to the $0/1$ loss for $\mathcal{C}$ with clean labels. Note that by construction $r(a)|x = c_r\eta + \pi^*(a|x)c_r(1-2\eta)$. So,

$$\hat{V}_F(\pi) = \frac{1}{N}\sum_{i=1}^{N}\langle r_i, \pi(\cdot|x_i)\rangle = \frac{c_r}{N}\sum_{i=1}^{N}\langle \eta\mathbf{1} + (1-2\eta)\pi^*(\cdot|x_i), \pi(\cdot|x_i)\rangle \tag{25}$$

$$= \frac{c_r\eta}{N} + \frac{c_r(1-2\eta)}{N}\sum_{i=1}^{N}\langle \pi^*(\cdot|x_i), \pi(\cdot|x_i)\rangle \tag{26}$$

This is equivalent to $0/1$ loss with noisy labels since π^* exactly corresponds to $y_{\mathcal{C}}$. □

A.4 Baselines

Theorem 3 (Baselines)**.** *Assume $K = 2$, noiseless rewards, and $\Delta_r = r_{\max} - r_{\min}$. Let $\pi_{B,b}$ be defined by a one nearest neighbor rule to interpolate $\hat{V}_{B,b}$. Let $p^*_\beta = P_{x,a_\beta\sim\beta|x,a^*\sim\pi^*|x}(a_\beta = a*)$ be the probability that the behavior policy chooses the optimal action. Then there exist problem instances such that for any choice of baseline b and any $\epsilon > 0$,*

$$\limsup_{N\to\infty} \mathbb{E}_S[V(\pi_F) - V(\pi_{B,b})] \geq \frac{1}{4}(\Delta_r - \epsilon)\min\{p^*_\beta, 1-p^*_\beta\}.$$

Proof. The proof consists of a construction of the hard problem instances. The hard problems look much like the hard problem above, except we modify the reward function so that no baseline can completely separate the two actions.

To construct the example, take any $\epsilon > 0$ and define a bandit problem with two actions (called 1 and 2):

$$x \sim U([-1,1]), \quad r|x = \begin{cases} (1, 1+\frac{\Delta_r-\epsilon}{2}) & x < 0 \\ (1+\frac{\Delta_r+\epsilon}{2}, 1+\Delta_r) & x \geq 0 \end{cases}, \quad \beta(2|x) = p^*_\beta \ \forall\ x, a$$

The true optimal policy has $\pi^*(2|x) = 1$ for all x and $V(\pi^*) = 1 + \Delta_r$. The policy with full feedback π_F is to always choose action 2, since every observation will show that action 2 is better.

We will refer to $r_i(a_i) - b$ as the "effective reward" at datapoint i. As a direct consequence of Theorem 1 we get that conditioned on x falling within some interval I in $[-1, 1]$ where the effective rewards are constant and strictly greater than zero, we will have bandit error of $\frac{|I|}{2}(r(2) - r(1))p^*_\beta$ while if the effective rewards are strictly negative we will have bandit error of $\frac{|I|}{2}(r(2) - r(1))(1 - p^*_\beta)$.

Since the nearest neighbor classifier in a two action problem will strictly depend of the sign of the effective reward, there are 5 cases to consider for the baseline: (1) $b < 1$, (2) $1 \leq b < 1 + \frac{\Delta_r - \epsilon}{2}$, (3) $1 + \frac{\Delta_r - \epsilon}{2} \leq b < 1 + \frac{\Delta_r + \epsilon}{2}$, (4) $1 + \frac{\Delta_r + \epsilon}{2} \leq b < 1 + \Delta_r$, and (5) $1 + \Delta_r \leq b$. Note also that the reward function forces us to consider two intervals $I_1 = [-1, 0)$ and $I_2 = [0, 1]$, each of measure 1 under the distribution over x. We will show that in the best case we can achieve the bound in the theorem statement.

Case 1 ($b < 1$): The effective rewards are strictly positive always. Thus the bandit error will be the sum of the error on the intervals I_1 and I_2. This is $\frac{1}{2}(\frac{\Delta_r - \epsilon}{2})p^*_\beta + \frac{1}{2}(\frac{\Delta_r - \epsilon}{2})p^*_\beta = \frac{1}{2}(\Delta_r - \epsilon)p^*_\beta$.

Case 2 ($1 \leq b < 1 + \frac{\Delta_r - \epsilon}{2}$): The effective rewards are properly split on I_1 and strictly positive on I_2. Thus the bandit error will be $0 + \frac{1}{2}(\frac{\Delta_r - \epsilon}{2})p^*_\beta = \frac{1}{4}(\Delta_r - \epsilon)p^*_\beta$.

Case 3 ($1 + \frac{\Delta_r - \epsilon}{2} \leq b < 1 + \frac{\Delta_r + \epsilon}{2}$): The effective rewards are negative on I_1 and positive on I_2. Thus the bandit error will be $\frac{1}{2}(\frac{\Delta_r - \epsilon}{2})(1 - p^*_\beta) + \frac{1}{2}(\frac{\Delta_r - \epsilon}{2})p^*_\beta = \frac{1}{4}(\Delta_r - \epsilon)$.

Case 4 ($1 + \frac{\Delta_r + \epsilon}{2} \leq b < 1 + \Delta_r$): The effective rewards are negative on I_1 and properly split on I_2. Thus the bandit error will be $\frac{1}{2}(\frac{\Delta_r - \epsilon}{2})(1 - p^*_\beta) + 0 = \frac{1}{4}(\Delta_r - \epsilon)(1 - p^*_\beta)$.

Case 5 ($1 + \Delta_r \leq b$): The effective rewards are always negative. Thus the bandit error will be $\frac{1}{2}(\frac{\Delta_r - \epsilon}{2})(1 - p^*_\beta) + \frac{1}{2}(\frac{\Delta_r - \epsilon}{2})(1 - p^*_\beta) = \frac{1}{2}(\Delta_r - \epsilon)(1 - p^*_\beta)$.

Looking back, the best baseline is either case 2 or case 4, so under any baseline the bandit error is at least $\frac{1}{4}(\Delta_r - \epsilon)\min\{p^*_\beta, 1 - p^*_\beta\}$. This construction did not depend on the size of the dataset, so it is even true as the number of datapoints tends to infinity. □

B Discussion of context-dependent baselines

An extension of the idea of baselines that is common is reinforcement learning is to learn a context-dependent baseline function $b : \mathcal{X} \to \mathbb{R}$ which is usually an estimate of the value function (Sutton and Barto, 2018). Then, looking back at Lemma 1, shifting the reward will mean that an interpolating policy for $\hat{V}_{B,b}$ will instead choose a_i whenever $r_i(a_i) - b(x_i) > 0$.

This analysis suggests that a context-dependent baseline prevents bandit overfitting whenever the following property holds at a context x: $r(a) > b(x)$ for the optimal a and $r(a) \leq b(x)$ for all other a (where r is conditioned on x). We will say that such a baseline *picks out the optimal action* at that context. If a baseline picks out the optimal action at every context, then there would be no issue with irreducible bandit overfitting, but learning such a baseline essentially requires solving the problem by learning a Q function. This makes it unclear whether such a strategy is fundamentally solving the problem or just dressing value-base learning up as policy optimization.

There are special cases (like classification problems with bandit feedback) where the reward structure is known in advance and makes finding a baseline easy. In these special cases learning a baseline that picks out the optimal action everywhere may be much easier than learning the Q function.

Finally, while it is beyond the scope of this paper, we hope to expand upon this insight that a baseline ought to pick out the optimal action in future work.

C Value-based learning

Theorem 4 (Reduction to regression)**.** *Assuming strict positivity, then with $\hat{Q}_{S_B}$ as defined in (7) then*

$$V(\pi^*) - V(\pi_{\hat{Q}}) \leq \frac{2}{\sqrt{\tau}}\sqrt{\mathbb{E}_{x,a\sim\beta}[(Q(x,a) - \hat{Q}_{S_B}(x,a))^2]}.$$

Proof. The proof follows directly from linking the subsequent lemmas with $\hat{\pi} = \pi_{\hat{Q}_{S_B}}$ and Π be the set of all

policies in Lemma 2. □

Lemma 2 (Mismatch: from MSE to Regret)**.** *Assume strict positivity. Let $\hat{\pi}$ be the greedy policy with respect to some $\hat{Q}$ and let Π be any class of policies to compete against, which contains $\hat{\pi}$. Then*

$$\sup_{\pi\in\Pi} V(\pi) - V(\hat{\pi}) \leq 2\sqrt{\sup_{\pi\in\Pi} \mathbb{E}_{x,a\sim\mathcal{D},\pi}[(Q(x,a)-\hat{Q}(x,a))^2]} \tag{27}$$

Proof. We can expand the definition of regret and then add and subtract and apply a few inequalities. Let $\bar{\pi}$ be the policy in Π which maximizes V. Then

$$\begin{aligned}
\sup_{\pi\in\Pi} V(\pi) - V(\hat{\pi}) &= \mathbb{E}_x\Big[\mathbb{E}_{a\sim\bar{\pi}|x}[Q(x,a)] - \mathbb{E}_{a\sim\hat{\pi}|x}[Q(x,a)]\Big] && (28)\\
&= \mathbb{E}_x\Big[\mathbb{E}_{a\sim\bar{\pi}|x}[Q(x,a)] - \mathbb{E}_{a\sim\hat{\pi}|x}[\hat{Q}(x,a)] + \mathbb{E}_{a\sim\hat{\pi}|x}[\hat{Q}(x,a)] - \mathbb{E}_{a\sim\hat{\pi}|x}[Q(x,a)]\Big] && (29)\\
&\leq \mathbb{E}_x\Big[\mathbb{E}_{a\sim\bar{\pi}|x}[|Q(x,a)-\hat{Q}(x,a)|] + \mathbb{E}_{a\sim\hat{\pi}|x}[|Q(x,a)-\hat{Q}(x,a)|]\Big] && (30)\\
&\leq \sqrt{\mathbb{E}_x\mathbb{E}_{a\sim\bar{\pi}|x}[(Q(x,a)-\hat{Q}(x,a))^2]} + \sqrt{\mathbb{E}_x\mathbb{E}_{a\sim\hat{\pi}|x}[(Q(x,a)-\hat{Q}(x,a))^2]} && (31)\\
&\leq 2\sqrt{\sup_{\pi\in\Pi}\mathbb{E}_x\big[\mathbb{E}_{a\sim\pi|x}[(Q(x,a)-\hat{Q}(x,a))^2]\big]} && (32)
\end{aligned}$$

The first inequality holds since $\hat{\pi}$ maximizes $\hat{Q}$ and by using the definition of absolute value, the second by Jensen, and the third by introducing the supremum. □

Lemma 3 (Transfer: from β to π)**.** *Assume strict positivity and take any Q-function $\hat{Q}$ and any policy π, then*

$$\mathbb{E}_{x,a\sim\mathcal{D},\pi}[Q(x,a)-\hat{Q}(x,a))^2] < \frac{1}{\tau}\bigg(\mathbb{E}_{x,a\sim\mathcal{D},\beta}[(Q(x,a)-\hat{Q}(x,a))^2]\bigg). \tag{33}$$

Proof. Let π be any policy. Then

$$\begin{aligned}
\mathbb{E}_x\mathbb{E}_{a\sim\pi|x}[(Q(x,a)-\hat{Q}(x,a))^2] &= \int_x p(x)\sum_a \pi(a|x)(Q(x,a)-\hat{Q}(x,a))^2dx && (34)\\
&= \int_x \sum_a \pi(a|x)\frac{\beta(a|x)}{\beta(a|x)}p(x)(Q(x,a)-\hat{Q}(x,a))^2dx && (35)\\
&< \frac{1}{\tau}\int_x\sum_a \beta(a|x)p(x)(Q(x,a)-\hat{Q}(x,a))^2dx && (36)\\
&= \frac{1}{\tau}\mathbb{E}_{x,a\sim\mathcal{D},\beta}[(Q(x,a)-\hat{Q}(x,a))^2] && (37)
\end{aligned}$$

where we use a multiply and divide trick and apply the definition of strict positivity to ensure that $\frac{\pi(a|x)}{\beta(a|x)} < \frac{1}{\tau}$. □

D Experiments

D.1 Bandit overfitting and the behavior policy in a toy problem

Data. Data is generated according to:

$$x \sim U([-1,1]), \quad r|x=(1,2), \quad a\sim\beta(\cdot|x)$$

Where we consider 64 behavior policies for β as defined in the main text. Formally, let p_1,p_2,p_3 be probabilities in $[0.05, 0.35, 0.65, 0.95]$, then define

$$\beta_{p_1,p_2,p_3}(1|x) := \begin{cases} p_1 & x\in[-1,-1/3] \\ p_2 & x\in(-1/3,1/3) \\ p_3 & x\in[1/3,1]. \end{cases} \tag{38}$$

We consider all possible such behavior policies. For each policy we draw a training set of 20 points and a test set of 1000 points. To facilitate learning we use a 5 dimension feature representation of x as $\sin(2^i x)$ for $i \in \{0, \cdots, 4\}$.

Model. We use an MLP with one hidden layer of width 512 and a 2 dimensional output for both policies and Q functions. Policies are defined by taking the softmax of the output of the MLP.

Learning. We train the models using Adam with a learning rate of 0.001 and batch size of 5 for 1000 epochs.

D.2 Bandit CIFAR-10

D.2.1 Detailed experimental setup

Data. We use a bandit version of the CIFAR-10 dataset (Krizhevsky, 2009). The conversion from classification to bandit is made just as for MNIST.

We use two different behavior policies. One is a uniform behavior that selects each action with probability 0.1 and the other is the hand-crafted behavior policy from Joachims et al. (2018), which we will refer to as HC.

Model. We use a ResNet-18 (He et al., 2016) from PyTorch (Paszke et al., 2019) for both the policy and the Q function. The only modification we make to accommodate for the smaller images in CIFAR is to remove the first max-pooling layer.

Learning. We train using SGD with momentum 0.9 and a batch size 128 for 1000 epochs. We use a learning rate of 0.1 for the first 200 epochs, 0.01 for the next 200, and 0.001 for the last 600. To improve stability we use gradient clipping and reduce the learning rate in the very first epoch to 0.01.

D.2.2 Extended Results

Here we present the learning curves for the various models we trained.

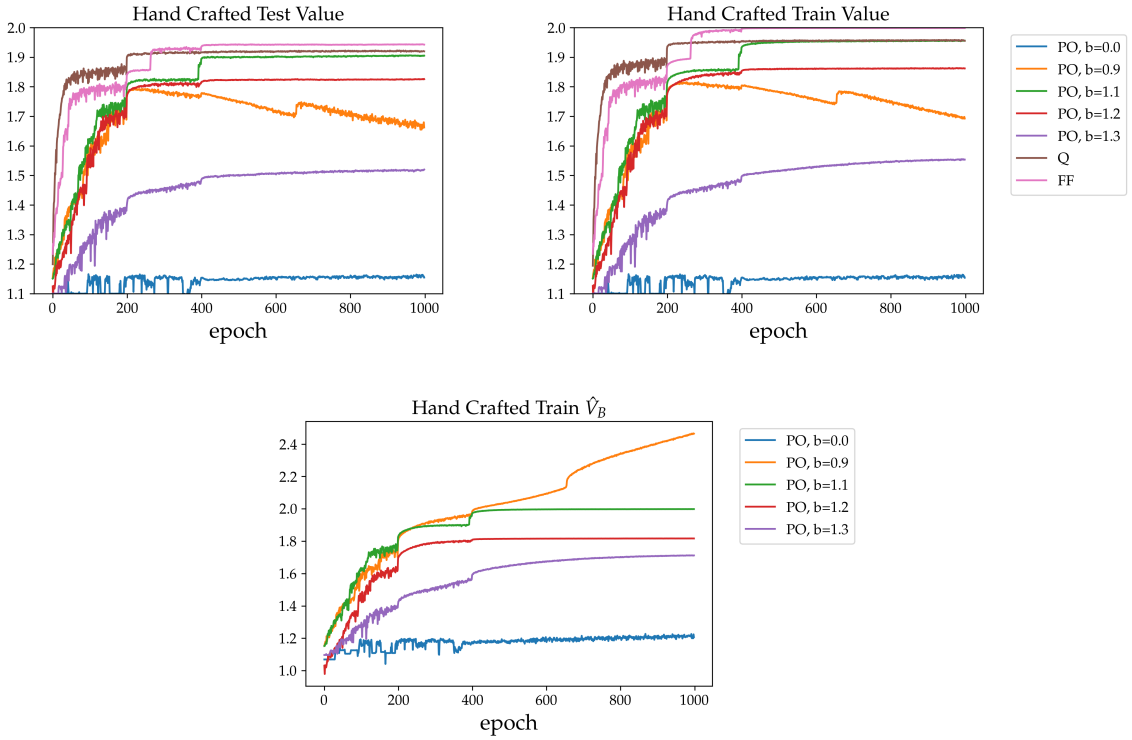

Figure 4: Learning curves for all algorithms trained on the hand-crafted actions for CIFAR-10

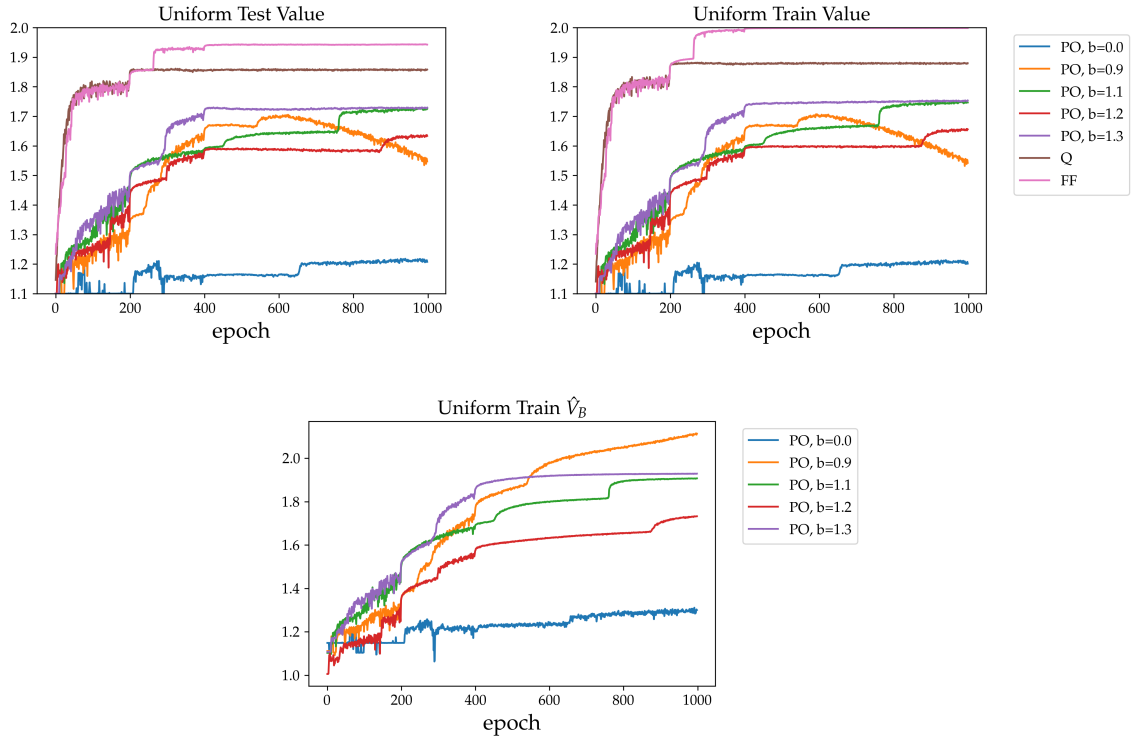

Figure 5: Learning curves for all algorithms trained on the uniform actions for CIFAR-10

Analysis. These learning curves show how training policies directly by stochastic gradient ascent on $\hat{V}_B$ can be difficult (as observed in Chen et al. (2019)). The learning curves will often have long periods of no improvement signifying a difficult optimization landscape. This is especially true for the case where no baseline is present and optimization can barely proceed at all.

Another important observation is that for a baseline of 0.9 we have effective rewards of 0.1 and 1.1. In this case the policy begins to learn as it would if the rewards for misclassified examples were negative and then the bandit overfitting kicks in as the policy learned to imitate the misclassified examples to increase $\hat{V}_B$.

Finally, as in Joachims et al. (2018) the performance is highly dependent on the baseline even among those that ensure that misclassified examples have negative reward while correctly classified examples have positive reward. We speculate that this is due to optimization issues rather than statistical issues. Baselines do after all also serve to reduce variance in the gradients.

D.3 World3 experiment in `whynot`

D.3.1 Detailed experimental setup

Data. Data is generated according to the World3 environment from `whynot` (Miller et al., 2020). Specifically, we run the simulator from 1975 to 2050. We randomize the initial state of the simulator and that random initial state is the context in the bandit problem. The two actions are `treat` or `no treat`, where the treatment in this case is reducing the persistent pollution generation factor from 1.0 to 0.85. The reward corresponds to the total population in 2050. So, the problem is essentially testing the effect of reducing pollution on population in the simulator.

We normalize the data as follows to create a meaningful bandit problem. Contexts: we calculate the mean and standard deviation on the training set and use these to whiten the data. Rewards: to get the rewards in a reasonable scale we subtract 1.5×10^8 and divide by 1×10^9. To ensure that `treat` does not always dominate `no treat` we subtract 8 from the rewards of `treat`. This can be seen as modeling a cost incurred by choosing the

`treat` action.

We sample a train set of 1000 points and a test set of 5000 points from this process using a uniform behavior policy.

Model. We use an MLP with one hidden layer of width 512 and a 2 dimensional output for both policies and Q functions. Policies are defined by taking the softmax of the output of the MLP.

Learning. We train using SGD with learning rate 0.1, momentum 0.9, and batch size 128 for 10000 epochs for all models.

D.3.2 Extended results

Here we present a few learning curves for the models trained in this problem.

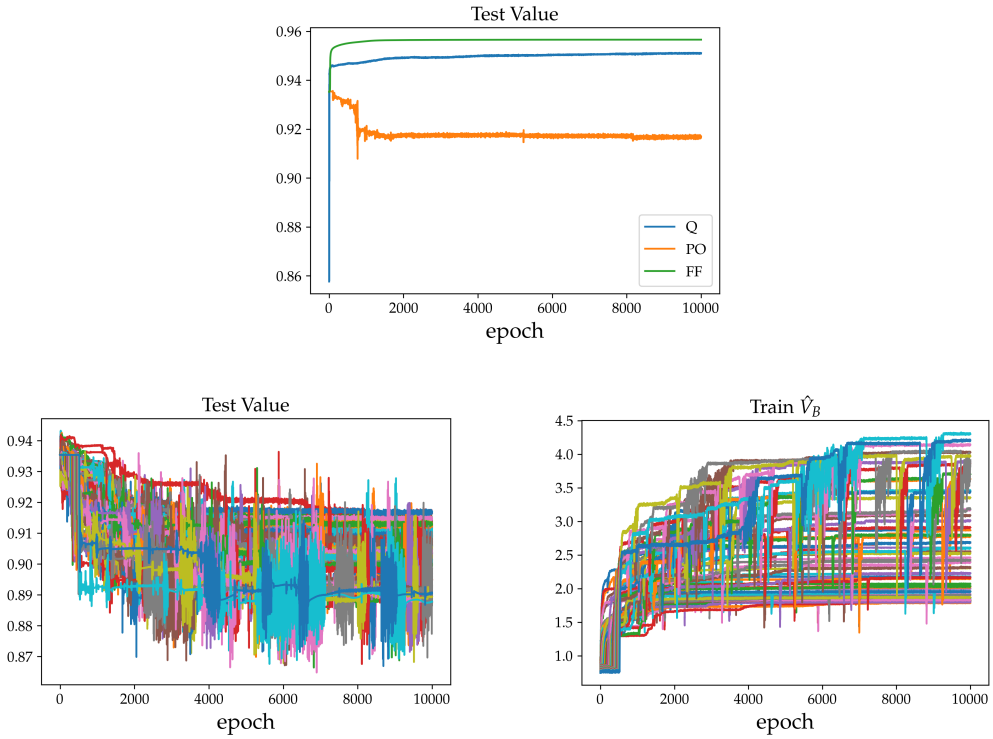

Figure 6: Top: Learning curves for all algorithms trained on the World3 environment with no baseline. Bottom: Learning curves for policy optimization across all baselines.

Analysis. Here we show that policy optimization has qualitiatively the same behavior in the World3 problem across all baselines. As the estimated value increases, the test value decreases. This is indicative of bandit overfitting. Tuning the baseline is not sufficient to solve this problem since the reward function is much richer and more context dependent than in a classification problem.

E Discussion of propensity overfitting

Swaminathan and Joachims (2015b) raise an issue that is similar to bandit overfitting that they call "propensity overfitting" and propose a self-normalized estimator in an attempt to alleviate the problem. They explain the issue as overfitting towards the sum of propensities $\hat{P}_N$ defined below and they propose to maximize the self-normalized

$\hat{V}_{\text{SNIW}}$:

$$\hat{P}_N(\pi; S_B) = \frac{1}{N}\sum_{i=1}^{N} \frac{\pi(a_i|x_i)}{\beta(a_i|x_i)}, \qquad \hat{V}_{\text{SNIW}}(\pi; S_B) = \frac{\hat{V}_{\text{IW}}(\pi; S)}{\hat{P}_N(\pi; S_B)} = \frac{\sum_{i=1}^{N} r_i(a_i)\frac{\pi(a_i|x_i)}{\beta(a_i|x_i)}}{\sum_{i=1}^{N}\frac{\pi(a_i|x_i)}{\beta(a_i|x_i)}} \tag{39}$$

The first thing to note is that the notion of overfitting towards $\hat{P}_N$ is not as precise as our definition of bandit error as a way to measure bandit overfitting. Second, the proposed self-normalized estimator is still vulnerable to bandit overfitting. Consider the following modified version of our two action example:

$$x \sim U([-1, 1]), \qquad r|x = (1 + x, 2 + x), \qquad \beta(a|x) = 1/2 \ \forall x, a \tag{40}$$

Clearly the optimal policy π^* always chooses action 2. With access to the full reward, the learned policy would see that at every datapoint, the objective $\hat{V}_F$ is maximized by choosing action 2 as desired. However, assuming without loss of generality by reordering indices that $x_1 \approx 1$ and $a_1 = 2$. Then a policy $\hat{\pi}$ that sets $\hat{\pi}(a_1|x_1) = 1$ and $\hat{\pi}(a_i|x_i) = 0$ for all $i > 1$ has value estimates:

$$\hat{V}_{SNIW}(\hat{\pi}) = \frac{(2 + x_1)\frac{1}{1/2}}{\frac{1}{1/2}} = 2 + x_1 \approx 3 \tag{41}$$

$$\mathbb{E}[\hat{V}_{SNIW}(\pi^*)] = \frac{\frac{N}{2}\mathbb{E}[2 + x]\frac{1}{1/2}}{\frac{N}{2}\frac{1}{1/2}} = \mathbb{E}[2 + x] = 2 \tag{42}$$

Thus, $\hat{V}_{SNIW}$ can be optimized by suboptimal policies like $\hat{\pi}$. This shows how the self-normalized estimator is insufficient to solve the problem of bandit overfitting.

Joachims et al. (2018) show that a constant baseline approximates a self-normalized estimate. As shown in the main paper, constant baselines are insufficient to prevent bandit overfitting. This is because when the reward has rich structure that depends on the contexts, it is not possible to find a baseline that picks out the optimal action.

F Small model classes

In this section we state and prove theorems that bound each term of the full regret decomposition for each algorithm we consider when we use finite model classes. Similar results can be shown for other classical notions of model class complexity.

Theorem 5 (Policy optimization with a small model class)**.** *Assume strict positivity and a finite policy class* Π*. Let* $\varepsilon_\Pi = V(\pi^*) - \sup_{\pi\in\Pi} V(\pi)$*. Denote* $\Delta_r = r_{max} - r_{min}$*. Then we have that for any* $\delta > 0$ *with probability* $1 - \delta$ *each of the following holds:*

$$\textit{Approximation Error} = V(\pi^*) - \sup_{\pi\in\Pi} V(\pi) \le \varepsilon_\Pi$$

$$\textit{Estimation Error} = \sup_{\pi\in\Pi} V(\pi) - V(\pi_F) \le 2\Delta_r\sqrt{\frac{\log(2|\Pi|/\delta)}{2N}}$$

$$\textit{Bandit Error} = V(\pi_F) - V(\pi_B) \le \frac{2\Delta_r}{\tau}\sqrt{\frac{\log(2|\Pi|/\delta)}{2N}}$$

Proof. The bound on approximation error follows directly from the definition of ε_Π. The bound on the estimation error follows from a standard application of a Hoeffding bound on the random variables $X_i = \langle r_i, \pi(\cdot|x_i)\rangle$ which are bounded by Δ_r and a union bound over the policy class.

The bound on bandit error essentially follows Theorem 3.2 of Strehl et al. (2010), we include a proof for completeness:

$$\begin{aligned} V(\pi_F) - V(\pi_B) &= V(\pi_F) - \hat{V}_B(\pi_B) + \hat{V}_B(\pi_B) - V(\pi_B) \\ &\le V(\pi_F) - \hat{V}_B(\pi_F) + \hat{V}_B(\pi_B) - V(\pi_B) \\ &\le 2\sup_{\pi\in\Pi} |V(\pi) - \hat{V}_B(\pi)| \\ &\le \frac{2\Delta_r}{\tau}\sqrt{\frac{\log(2|\Pi|/\delta)}{2N}} \end{aligned}$$

The first inequality comes from the definition of π_B. The second comes since both $\pi_F, \pi_B \in \Pi$. And the last inequality follows from an application of a Hoeffding bound on the random variables $X_i = r_i(a_i)\frac{\pi(a_i|x_i)}{p_i}$ which are bounded by $\frac{\Delta_r}{\tau}$ and a union bound over the policy class. □

Theorem 6 (Value-based learning with a small model class)**.** *Assume strict positivity and a finite function class $\mathcal{Q}$ which induces a finite class of greedy policies $\Pi_{\mathcal{Q}}$. Let $\varepsilon_{\mathcal{Q}} = \inf_{\widehat{Q}\in\mathcal{Q}} \mathbb{E}_{x,a\sim\mathcal{D},\beta}[(Q(x,a) - \widehat{Q}(x,a))^2]$. Denote $\Delta_r = r_{max} - r_{min}$. Then we have that for any $\delta > 0$ with probability $1-\delta$ each of the following holds:*

$$\textit{Approximation Error} = V(\pi^*) - \sup_{\pi\in\Pi_{\mathcal{Q}}} V(\pi) \leq 2\sqrt{\varepsilon_{\mathcal{Q}}/\tau} \tag{43}$$

$$\textit{Estimation Error} = \sup_{\pi\in\Pi_{\mathcal{Q}}} V(\pi) - V(\pi_F) \leq 2\Delta_r\sqrt{\frac{\log(|\mathcal{Q}|/\delta)}{2N}} \tag{44}$$

$$\textit{Bandit Error} = V(\pi_F) - V(\pi_{\widehat{Q}}) \leq \frac{10\Delta_r}{\sqrt{\tau}}\sqrt{\frac{\log(|\mathcal{Q}|/\delta)}{N}} + 6\sqrt{\Delta_r}\left(\frac{\log(|\mathcal{Q}|/\delta)}{\tau N}\varepsilon_{\mathcal{Q}}\right)^{1/4} + 2\sqrt{\varepsilon_{\mathcal{Q}}/\tau} \tag{45}$$

Proof. To bound the approximation error, we can let $\hat{\pi}$ be the greedy policy associated with a Q-function $\widehat{Q}$ and apply Lemmas 2 and 3. This gives us

$$V(\pi^*) - \sup_{\hat{\pi}\in\Pi_{\mathcal{Q}}} V(\hat{\pi}) = \inf_{\widehat{Q}\in\mathcal{Q}} [V(\pi^*) - V(\hat{\pi})] \leq \inf_{\widehat{Q}\in\mathcal{Q}} \frac{2}{\sqrt{\tau}}\sqrt{\mathbb{E}_{x,a\sim\mathcal{D},\beta}[(Q(x,a) - \widehat{Q}(x,a))^2]} = 2\sqrt{\varepsilon_{\mathcal{Q}}/\tau}. \tag{46}$$

The bound on the estimation error follows the same as before from standard uniform convergence arguments.

The bound on the bandit error follows by again applying Lemmas 2 and 3 and then making the concentration argument from Lemma 16 of Chen and Jiang (2019). Explicitly, our Lemmas give us

$$V(\pi_F) - V(\pi_{\widehat{Q}}) \leq V(\pi^*) - V(\pi_{\widehat{Q}}) \leq \frac{2}{\sqrt{\tau}}\sqrt{\mathbb{E}_{x,a\sim\mathcal{D},\beta}[(Q(x,a) - \widehat{Q}(x,a))^2]}. \tag{47}$$

Then, to bound the squared error term, we can add and subtract:

$$\mathbb{E}_{x,a\sim\mathcal{D},\beta}[(Q(x,a) - \widehat{Q}(x,a))^2] = \mathbb{E}_{x,a\sim\mathcal{D},\beta}[(Q(x,a) - \widehat{Q}(x,a))^2] - \inf_{\bar{Q}\in\mathcal{Q}} \mathbb{E}_{x,a\sim\mathcal{D},\beta}[(Q(x,a) - \bar{Q}(x,a))^2] \tag{48}$$

$$+ \inf_{\bar{Q}\in\mathcal{Q}} \mathbb{E}_{x,a\sim\mathcal{D},\beta}[(Q(x,a) - \bar{Q}(x,a))^2] \tag{49}$$

$$\leq \mathbb{E}_{x,a\sim\mathcal{D},\beta}[(Q(x,a) - \widehat{Q}(x,a))^2] - \inf_{\bar{Q}\in\mathcal{Q}} \mathbb{E}_{x,a\sim\mathcal{D},\beta}[(Q(x,a) - \bar{Q}(x,a))^2] \tag{50}$$

$$+ \varepsilon_{\mathcal{Q}}. \tag{51}$$

Now we want to show that the difference in squared error terms concentrates for large N. This is precisely what Lemma 16 from Chen and Jiang (2019) does using a one-sided Bernstein inequality. This gives us for any $\delta > 0$ an upper bound with probability $1-\delta$ of

$$\frac{56\Delta_r^2\log(|\mathcal{Q}|/\delta)}{3N} + \sqrt{\varepsilon_{\mathcal{Q}}\frac{32\Delta_r^2\log(|\mathcal{Q}|/\delta)}{N}} \tag{52}$$

Plugging this in and simplifying the constants gives the result. □

Analysis. Comparing the two bounds we see that value-based learning has worse dependence on model misspecification because it is not directly trying to optimize the value of the learned policy. However, value-based methods do have better dependence on τ. So in the case with no misspecification, the bounds for value-based learning are slightly better.