DATA POISONING ATTACK ON ONLINE LEARNERS VIA DUPLICATION AND DELETION

Anonymous authors
Paper under double-blind review

ABSTRACT

The recent trend in modern predictive models, such as LLMs, has prompted a change in the data source from curated datasets to a stream of real-time data. This change introduces a new vulnerability that an adversary could exploit to cause substantial damage while evading detection. However, the adversary is constrained in designing attacks based solely on observations available at decision time. Understanding the optimal attack strategy provides critical insights for designing defense mechanisms that secure predictive models and improve their resilience. In this paper, we study a white-box data poisoning attack against an online learner, where the adversary seeks to steer the learner's parameters toward a specific target. The adversary is restricted to deleting and duplicating existing observations within each incoming batch, without modifying their features and labels. We frame the attacker-defender interaction as a constrained Markov Decision Process (MDP) and propose a reinforcement learning (RL) agent that approximately derives the near-optimal attack. We provide theoretical analysis and numerical experiments to show that our algorithm achieves lower cumulative attack cost than a generic model-free RL baseline.

1 INTRODUCTION

Recently, data-driven predictive models, mainly machine learning (ML) models, have been integrated into many applications, yet the data-driven nature makes them susceptible to deliberate manipulation and tampering of the data and learning algorithm they rely on. Adversarial machine learning (AML) studies these vulnerabilities by specifying an attacker's objectives, knowledge, and capabilities, and examining the resulting consequences for learning and prediction (Biggio et al., 2014). Parallel to the study of adversarial attacks is the development of defense mechanisms to counteract a specific attack (Goldblum et al., 2023). Previous literature (Biggio & Roli, 2018) distinguishes attacks by their access to the system and the stage of the learning pipeline at which they intervene.

One such attack is *data poisoning*, in which an adversary manipulates a small fraction of the training data, for example, by adding or modifying samples, to degrade the model's performance at test time (Biggio & Roli, 2018). An adversary may obtain control over the training pipeline legitimately, by invoking the right to erasure (Marchant et al., 2022), or illegitimately, by compromising the training pipeline (Shumailov et al., 2021).

While the study of data poisoning has been developed largely in the offline setting (Biggio et al., 2012), a growing body of work studies sequential victims (Wang & Chaudhuri, 2018). A recurring finding is that attacks which ignore the effect of the current manipulation on future updates are weaker than those that plan over the horizon (Zhang et al., 2020; Pang et al., 2021). In practice, many ML systems in deployment update their internal learning models incrementally as new observations arrive. For example, LinkedIn uses incremental and online training to provide users' feeds, advertisements, and job recommendations (Zhu, 2025).

Building on prior studies of online poisoning (Margiotta et al., 2023; Zhang et al., 2020) that derive the optimal attack based on the previously observed data, we formulate the poisoning attack as a constrained finite-horizon Markov decision process. The attacker intercepts a batch of data intended to be sent to the victim and select to duplicate or delete each observation. The main challenge is the joint decision of sample manipulations.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

That is, deletion or duplication of any observation changes the feasible actions for the remaining observations through the shared budget and batch-size constraint. Each feasible allocation must be evaluated according to its immediate and subsequent effects on the learning model. To address this combinatorial planning problem with partially unknown transition dynamics, we exploit the known learner update through *Fitted Expected-Value Iteration (FEVI)*, a modification of the *Fitted Value Iteration* (FVI) (Munos & Szepesvári, 2008), and construct a separable surrogate objective function that enables efficient allocation of deletions and duplications.

Main Contributions

- We formulate sequential data poisoning by an adversary that only deletes and duplicates samples from i.i.d. batches subject to a per-sample replication budget, per-batch edit budget, and batch-size preservation. This formulation captures an adversary with filtering and replay access, which is a weaker yet more realistic capability than arbitrary perturbations.
- We propose an FEVI attacker that exploits the victim's known one-step update and linearized surrogate minimization to avoid learning the state's values over the whole state space. Our approximation achieves lower cumulative cost compared to model-free RL solutions in the experiments for both simulated and real datasets.
- We provide a finite-sample analysis of FEVI, showing that the linearized surrogate incurs only a small, controlled error at each stage, and that the optimality gap of the resulting policy decays at rate $O(n^{-1/2})$ up to function-approximation and surrogate errors.

2 Related Works

There has been a substantial line of work in data poisoning. Prior work has investigated poisoning attacks against support vector machines (Biggio et al., 2012), linear regression (Li et al., 2021; Wen et al., 2021; Müller et al., 2020), logistic regression (Mei & Zhu, 2015; Wen et al., 2020), and multilayer perceptrons (Muñoz-González et al., 2017; Lu et al., 2022). Complementary theoretical studies have examined which target models an attacker can induce, how poisoning budgets constrain achievable damage, and how attacks affect learning dynamics (Suya et al., 2021; Margiotta et al., 2023).

Our study focuses on *targeted attacks*, where the attacker steers the victim's model toward the target. While there are many types of targeted attacks, such as *instance-targeted attacks* (Shafahi et al., 2018; Zhu et al., 2019) and *backdoor attacks* (Gu et al., 2019; Turner et al., 2019), we investigate an attacker steering the learned parameters toward target parameters $\theta^\dagger$ that they want in order to achieve their goal. This is similar to how the adversary in Carreau et al. (2025); Zhang et al. (2020); Margiotta et al. (2023) operates.

To the best of our knowledge, the first formulation of online data poisoning as a stochastic optimal control problem, equivalently a Markov decision process, was done by Zhang et al. (2020), in which the adversary perturbs each arriving data point without constraint. Such attacks, however, are prone to detection, with many defenses developed to filter and remove anomalous samples (Steinhardt et al., 2017).

This motivates attacks that act only on the real data. Shumailov et al. (2021) limits the attack to reordering of the data points in and between mini-batches during the ML training with Stochastic Gradient Descent (SGD). Broderick et al. (2021) deletes a small fraction of observations and shows that it can overturn the statistical conclusion in classical statistical models.Carreau et al. (2025) proposes an attack on Bayesian inference that is based only on deleting and replicating true observations, steering a Bayesian posterior toward a target. These attacks, however, are all one-shot.

Online poisoning treats attacks as decisions whose effects accumulate through learning. Prior work studies both attacks with clairvoyant knowledge of the entire data stream (Wang & Chaudhuri, 2018) and policies based on a current observation (Zhang et al., 2020). The accumulate attacks exploit how earlier interventions influence subsequent updates (Pang et al., 2021). A parallel study in this field is conducted as sequential machine teaching that similarly selects samples to guide a learner toward a target (Liu et al., 2017; Margiotta et al., 2023). We consider an attacker who observes the current learner and batch without access to future observations.

3 METHODOLOGY

Our framework considers an online data poisoning attack, framed as a sequential interaction between the attacker and the defender, who receives a batch of data at every discrete time step from an environment. To better understand how an adversary would design an attack in our scenario, we aim to rigorously formulate the data poisoning paradigm as an MDP and derive the optimal attack under the constraints that batch size must remain unchanged, and each data point cannot be duplicated more than a certain number of times.

Section 3.1 formally casts this interaction as a finite-horizon constrained MDP and states the Bellman optimality equation. Section 3.2 specifies the victim's learning model. Section 3.3 addresses the continuous state space via fitted expected-value iteration, and Section 3.4 reduces the combinatorial action space to an integer resource allocation problem solved exactly by greedy allocation.

3.1 MDP FORMULATION

Formally, we consider the online data poisoning attack as a sequential interaction between the online learning victim and the data poisoning attacker with a stochastic environment that constantly generates the output from the time-invariant law $\mathcal{P}_{\mathcal{Z}}$ induced by the sample space $\mathcal{Z}$. At time t, the environment draws an i.i.d. batch of m samples: $z_{it} \sim \mathcal{P}_{\mathcal{Z}}$ and forms a batch $\mathbf{z}_t = \{z_{it}\}_{i=1}^m$, where $z_{it} \in \mathcal{Z}$. The goal of the victim is to estimate the innate stochastic mechanism of $\mathcal{P}_{\mathcal{Z}}$ via a function $f \in \mathcal{F}$. In this paper, we limit the function class $\mathcal{F}$ to be a parametric model with parameter $\theta \in \mathbb{R}^D$ for some $D \in \mathbb{N}$ and approximate θ^*.

Without an attack, the victim performs a traditional model improvement via, for example, gradient descent. The attacker attempts to manipulate and disrupt the victim's parameter estimation pipeline via deletion and duplication of each sample in $\mathbf{z}_t$. To formulate this, the attack at time t is represented by a vector $\boldsymbol{a}_t \in \{0, 1, \ldots, L\}^m$, where L is the maximum repetitions per sample and, thus,$a_{t,i} = 0$ deletes the sample, $a_{t,i} = 1$ retains it, and $a_{t,i} = l \in \{2, \ldots, L\}$ duplicates it $l - 1$ times.

This action formulation results in a combinatorial action space, which can make exhaustive search for the optimal attack computationally prohibitive as the batch size grows. The attacker performs targeted data poisoning, moving the estimate θ_t toward the target $\theta^\dagger$, by minimizing the cost function g that is a sum of two distinct terms.

The first term corresponds to the nefarious cost, g_{nef}, quantifying how close the attacker's goal is achieved, and the second term corresponds to the perturbation cost g_{per}, reflecting the cost of duplicating and deleting samples. While there are many choices of g_{nef}, our target attack specifies $g_{\text{nef}}(\theta_t; \theta^\dagger) = \frac{1}{2}||\theta_{t+1} - \theta^\dagger||_2^2$. Each perturbation is assigned a fixed $\lambda \in \mathbb{R}^+$ cost, resulting in $g_{\text{per}}(\boldsymbol{a}_t; \lambda) = \frac{\lambda}{2m}||\boldsymbol{a}_t - \mathbf{1}||_1$. Combining these yields

$$g(\mathbf{z}_t, \theta_t, \boldsymbol{a}_t) = g_{\text{per}}(\boldsymbol{a}_t; \lambda) + g_{\text{nef}}(\theta_t; \theta^\dagger) = \frac{1}{2}||\theta_{t+1} - \theta^\dagger||_2^2 + \frac{\lambda}{2m}||\boldsymbol{a}_t - \mathbf{1}||_1. \tag{1}$$

Moreover, to reflect the limited capacity of the attacker, we impose the budget constraints $||\boldsymbol{a}_t - \mathbf{1}||_1 \leq B$ and $||\boldsymbol{a}_t||_\infty \leq L$ for any $t = 1, \ldots, T$, motivated by Carreau et al. (2025). To prevent unrestricted duplication, we require the attacked batch to retain its original size, m, imposing the constraint $\sum_{i=1}^m a_i = m$. These constraints reflect the limit to which the attacker can manipulate the dataset without being detected.

The goal of a rational attacker is to minimize the cost function g while accounting for the future cost subject to the budget and batch size constraints. However, the attacker lacks information about the underlying data-generating law $\mathcal{P}$ from which the future batches $(\mathbf{z}_{t+1}, \mathbf{z}_{t+2}, \ldots, \mathbf{z}_T)$ are drawn. Thus, we formulate this problem as a stochastic control problem with unknown transition probabilities. We establish the Markov property of $\mathbf{s}_t = (\theta_t, \mathbf{z}_t)$ in Appendix A.1, which allows us to formulate this process as a finite-horizon Markov Decision Process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, T, g, \mu)$.

The **state** $\mathbf{s}_h \in \mathcal{S}$ is the function of the current learner's parameter θ_h and clean data batch $\mathbf{z}_h$ such that $s_h = \psi(\mathbf{z}_h, \theta_h)$. In this work, the function $\psi(\cdot)$ is simply taken to be the concatenation between $\mathbf{z}_h$ and θ_h, i.e., $\mathbf{s}_h = [\mathbf{z}_h, \theta_h]^\top$. The **action** $\boldsymbol{a}_h \in \mathcal{A}$ is an integer-valued vector, permitting deletion, retention, and duplication of each sample. The action space is given by $\mathcal{A} = [L]^N \subset \mathbb{N}^N \cup \{0\}^N$, which is combinatorial.

The **transition probability kernel** $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, where $\Delta(\cdot)$ is a conditional probability simplex over the sample space $\mathcal{S}$ given the current state and action. Recall that $\mathcal{S}$ is a product of Θ and $\mathcal{Z}^{\otimes m}$. Throughout this work, the update function f is deterministic; thus, the randomness in the transition kernel is induced only by the product space $\mathcal{Z}^{\otimes m}$:

$$\mathcal{P}(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \boldsymbol{a}_t) = \delta_{f(\mathbf{s}_t, \boldsymbol{a}_t)}(\theta_{t+1}) \cdot \mathcal{P}(\mathbf{z}_{t+1}). \tag{2}$$

The **intermediate cost** $g(\theta_t, \mathbf{z}_t, \boldsymbol{a}_t)$ (equation 1) quantifies the myopic goal of the data poisoning attacker that aims to minimize and cause the most damage to the victim at only the current time t. The **initial state distribution** $\mu \in \Delta(\mathcal{S})$ specifies where the first state is drawn from, i.e., $\mathbf{s}_0 \sim \mu$. The initial victim's parameters θ_0 depend on their learning algorithm. The **horizon** $T \in \mathbb{N}$ specifies the number of steps over which the objective function is minimized.

Since our MDP is finite-horizon, the goal of the attacker is to learn a time-dependent optimal policy $\pi = (\pi_1, \ldots, \pi_T)$ for a finite-horizon MDP $\mathcal{M}$. A policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ maps a state space to a simplex over the action space. This definition includes the deterministic policy by specifying the probability simplex to be a Dirac delta distribution. To solve the above problem, we will frame this problem in a Bellman Optimality equation; we first define the value function of policy π, V_t^π, and denote the expectation of the value function of policy π given the current state $\mathbf{s}$ and action $\boldsymbol{a}$, $\bar{V}_t^\pi$:

$$V_t^\pi(\theta, \mathbf{z}) = \mathbb{E}_\pi \left[\sum_{\tau=t}^{T-1} g(\theta_\tau, \mathbf{z}_\tau, \boldsymbol{a}_\tau) \,\middle|\, \theta_t = \theta, \mathbf{z}_t = \mathbf{z} \right] \quad \forall t = 0, \ldots, T-1 \quad ; \quad V_T^\pi \equiv 0.$$

$$\bar{V}_t^\pi\big(f(\theta, \mathbf{z}, \boldsymbol{a})\big) \coloneqq \mathbb{E}_{\mathbf{s}' \sim \mathcal{P}(\cdot \mid \mathbf{s}, \boldsymbol{a})} \left[V_t^\pi(\mathbf{s}') \right] = \int V_t^\pi\big(f(\theta, \mathbf{z}, \boldsymbol{a}),\, \mathbf{z}'\big)\, d\mathcal{P}_{\mathcal{Z}}^{\otimes m}(\mathbf{z}').$$

Thus, the Bellman Optimality equation of state $\mathbf{s} = (\theta, \mathbf{z})$ is given by

$$V_t^*(\theta, \mathbf{z}) = \min_{\boldsymbol{a} \in \mathcal{A}} \left\{ g(\theta, \mathbf{z}, \boldsymbol{a}) + \bar{V}_{t+1}^*(f(\theta, \mathbf{z}, \boldsymbol{a})) \right\}. \tag{3}$$

where $\mathcal{A} = \{\boldsymbol{a} \in \{0, 1, \ldots, L\}^m : \sum_{i=1}^m a_i = m, ||\boldsymbol{a} - \mathbf{1}||_1 \leq B\}$. The optimal control policy π^* is derived by taking a minimum over the attacker's cumulative cost from following policy π, $J(\pi) \coloneqq \mathbb{E}_{\mathbf{s} \sim \mu} \left[V_0^\pi(\mathbf{s}) \right]$, which is equivalent to $\pi^* \in \arg\min_{\pi \in \Pi} J(\pi)$, where Π represents a policy space.

3.2 Victim's learning model

The *victim* or *defender* maintains parameters θ_t of a predictive model and updates them after receiving each batch $\mathbf{z}_t = (z_{t,1}, \ldots, z_{t,m})$. Before the update, the attacker applies $a_{t,i}$ to each observation. This changes how often an observation contributes to the update without changing its features or label. We assume that the batch's effect decomposes into per-sample contributions. The following assumption specifies how the integer attack affects the update.

Assumption 1. *(Linear separability over samples) The one-step update function of the victim f at time t based on the batch $\mathbf{z}_t$ is deterministic can be written as:*

$$\theta_{t+1} = f(\theta_t, \mathbf{z}_t, \mathbf{a}_t) = \theta_t + \frac{\eta_t}{m} \sum_{i=1}^m \mathbf{a}_{t,i} \cdot \phi(\theta_{t,i}, z_{t,i}) = \theta_t + \frac{\eta_t}{m} \langle \phi(\theta_t, \mathbf{z}_t), \mathbf{a}_t \rangle.$$

where η_t is a learning rate and $\phi(\cdot, \cdot)$ is a per-sample contribution to the victim's parameters.

Many standard ML parameter update algorithms satisfy assumption 1, for example, Stochastic Gradient Descent (SGD). In this work, we will study the setting where the victim follows the batch gradient descent update such that, given a victim's differentiable loss function $l : \Theta \times \mathcal{Z}^{\otimes m} \to \mathbb{R}$, the one-step update function is the negative gradient direction, $\phi(\theta, z_i) = -\nabla l(\theta, z_i)$.

3.3 Fitted Expected-Value Iteration

We build on fitted value iteration (Ernst et al., 2005), which approximates dynamic programming by a sequence of regression problems, fitting a function approximator to Bellman targets computed from the current estimate. Whereas Fitted Q-Iteration (FQI) approximates the action-value function,

Algorithm 1 Fitted Expected-Value Iteration (FEVI)

Require: Replay buffers $\{\mathcal{R}_t\}_{t=0}^{T-1}$, approximator $h(\cdot;\omega)$ with $\omega \in \Omega$, sample sizes N, M

$h_T \equiv 0$
for $t = T-1, \dots, 0$ **do**
  Sample $\theta^{(1)}, \dots, \theta^{(N)}$ from $\mathcal{R}_t$
  for $n = 1, \dots, N$ **do**
    Draw $\mathbf{Z}^{(n,1)}, \dots, \mathbf{Z}^{(n,M)}$ from $\mathcal{R}_t$ independent of $\theta^{(n)}$
    $Y_t^{(n)} \leftarrow \frac{1}{M} \sum_{j=1}^{M} \min_{\mathbf{a}\in\mathcal{A}} \Big\{ g_t(\theta^{(n)}, \mathbf{Z}^{(n,j)}, \mathbf{a}) + h_{t+1}\big(f(\theta^{(n)}, \mathbf{Z}^{(n,j)}, \mathbf{a})\big) \Big\}$
  end for
  $\omega_t \in \arg\min_{\omega\in\Omega} \frac{1}{N} \sum_{n=1}^{N} \big[h(\theta^{(n)};\omega) - Y_t^{(n)}\big]^2$
  $h_t(\cdot) \leftarrow h(\cdot;\omega_t)$
end for
return $\{h_t\}_{t=0}^{T}$

FEVI approximates the integrated value $\bar{V}_t$, the value of the post-decision state (Powell, 2011). This allows the policy to be derived by solving the constrained action-selection problem directly, without parameterizing an actor, and lets our analysis build on existing finite-sample bounds for fitted value iteration (Munos & Szepesvári, 2008; Agarwal et al., 2026).

Our formulation has a structure that allows a simplification of the value function approximated by standard FVI. From equation 2, it is established that the transition probability kernel only depends on the distribution of the batches, $\mathcal{P}_{\mathcal{Z}}^{\otimes m}$. Taking the expected value of both sides of equation 3,

$$\bar{V}_t(\theta) \;=\; \mathbb{E}_{\mathbf{z}\sim\mathcal{P}_{\mathcal{Z}}^{\otimes m}}\Big[\min_{\boldsymbol{a}\in\mathcal{A}} \big\{ g(\theta, \mathbf{z}, \boldsymbol{a}) + \bar{V}_{t+1}\big(f(\theta, \mathbf{z}, \boldsymbol{a})\big) \big\} \Big], \qquad \bar{V}_T \equiv 0. \tag{4}$$

This recursion is closed over $\bar{V}_k$ for $k = 0, \dots, T-1$, implying that an estimation of any $\bar{V}_t$ can be done via *backward induction*. In particular, conditional on a fixed $\bar{V}_{t+1}$, an expected value of the Bellman operator (right-hand side of 4) defines a regression target for $\bar{V}_t$. We call this augmented FVI, *Fitted Expected-Value Iteration (FEVI)*. This approach reduces the space where $\bar{V}_t$ needs to be estimated from $\Theta \times \mathcal{Z}^{\otimes m}$ to only the parameter space Θ.

For each time t, let $h_t := h(\cdot\,;\omega_t)$ be a function class parameterized by ω_t. Our goal is to find the best approximator h_t^* that fits to the Bellman target Y_t:

$$Y_t^{(n,j)} = \min_{\boldsymbol{a}\in\mathcal{A}} \Big\{ g_t(\theta^{(n)}, Z^{(n,j)}, \mathbf{a}) + h_{t+1}\big(f(\theta^{(n)}, Z^{(n,j)}, \boldsymbol{a})\big) \Big\} \quad ; \quad Y_t^{(n)} = \frac{1}{M} \sum_{j=1}^{M} Y_t^{(n,j)}. \tag{5}$$

We summarize the procedure to fit such h_t^* in Algorithm 1. The obvious issue with this method is that every Bellman target requires a minimization over $\mathcal{A}$, a set of cardinality $O\big((L+1)^m\big)$.

3.4 Decomposition of the Combinatorial Action Space

Optimizing over the feasible action space is a main computational challenge in FEVI. The constraints couple deletion and duplication decisions across samples, and exhaustive enumeration of feasible actions can become prohibitively expensive as the batch size grows. To address this challenge, we construct a separable surrogate of the Bellman minimization that can be solved efficiently by greedy allocation. Specifically, the minimization in the Bellman target equation 5 is

$$\begin{aligned} \min_{\boldsymbol{a}} \quad & g_t\big(\theta^{(n)}, \mathbf{Z}^{(n,j)}, \boldsymbol{a}\big) + h_{t+1}\big(f(\theta^{(n)}, \mathbf{Z}^{(n,j)}, \boldsymbol{a})\big) \\ \text{s.t.} \quad & \sum_{i=1}^{m} a_i = m, \qquad \sum_{i=1}^{m} |a_i - 1| \le B, \\ & a_i \in \{0, 1, \dots, L\}, \qquad i = 1, \dots, m. \end{aligned} \tag{6}$$

For the following derivation, we fix t, abbreviate $\theta^{(n)}$ and $\mathbf{Z}^{(n,j)}$ by θ and $\mathbf{Z}$. A separable convex objective over this action space gives an integer resource allocation problem with an ℓ_1-distance constraint and coordinate upper bounds. This structure connects our surrogate minimization to the resource allocation framework of Minamikawa & Shioura (2019).

To obtain the surrogate objective, we retain the first-order dependence on the victim's parameter update and the action penalty. The squared-distance component of g_t can be expanded algebraically, while the fitted function h_{t+1} is approximated using the following smoothness assumption.

Assumption 2. *(Smoothness of fitted function) Let $\mathcal{D} \subseteq \mathbb{R}^d$ be a convex domain of the learner's parameter vector. For any $t = 0, \ldots, T-1$, the fitted value function $h_{t+1} : \mathcal{D} \to \mathbb{R}$ is continuously differentiable and Λ_{t+1}-smooth such that, for $\theta, \theta' \in \mathcal{D}$,*

$$||\nabla h_{t+1}(\theta) - \nabla h_{t+1}(\theta')||_2 \leq \Lambda_{t+1}||\theta - \theta'||_2.$$

Under Assumption 1, define $\phi_i := \phi(\theta, z_i)$ and $u_t(\boldsymbol{a}) := \frac{\eta_t}{m}\sum_{i=1}^m a_i\phi_i$, so that $f(\theta, Z, \boldsymbol{a}) = \theta + u_t(\boldsymbol{a})$. Applying Taylor's expansion under Assumption 2 yields

$$h_{t+1}(\theta + u) = h_{t+1}(\theta) + \langle \nabla h_{t+1}(\theta), u\rangle + R_{t+1}(\theta, u),$$

where $|R_{t+1}(\theta, u)| \leq \frac{\Lambda_{t+1}}{2}\|u\|_2^2$ whenever the segment from θ to $\theta + u$ lies in $\mathcal{D}$. Denoting the objective in equation 6 by F_t, we obtain

$$\begin{aligned} F_t(\boldsymbol{a}) &:= g_t(\theta, Z, \boldsymbol{a}) + h_{t+1}\big(f(\theta, Z, \boldsymbol{a})\big) \\ &= C_t(\theta) + \frac{\eta_t}{m}\sum_{i=1}^m \alpha_{t,i}a_i + \frac{\lambda}{2m}\|\boldsymbol{a} - \mathbf{1}\|_1 + \frac{\eta_t^2}{2m^2}\sum_{i,j=1}^m a_i a_j\, \phi_i^\top \phi_j + R_{t+1}\big(\theta, u_t(\boldsymbol{a})\big). \end{aligned} \tag{7}$$

where $C_t(\theta) := \frac{1}{2}\|\theta - \theta^\dagger\|_2^2 + h_{t+1}(\theta)$ and $\alpha_{t,i} := \left\langle \theta - \theta^\dagger + \nabla h_{t+1}(\theta), \phi_i \right\rangle$.

The scores $\alpha_{t,i}$ may take either sign. Next, dropping the quadratic update term and the remainder gives the surrogate objective:

$$\widetilde{F}_t(\boldsymbol{a}) := C_t(\theta) + \frac{\eta_t}{m}\sum_{i=1}^m \alpha_{t,i}a_i + \frac{\lambda}{2m}\|\boldsymbol{a} - \mathbf{1}\|_1. \tag{8}$$

The discarded terms satisfy $|F_t(\boldsymbol{a}) - \widetilde{F}_t(\boldsymbol{a})| \leq \frac{1+\Lambda_{t+1}}{2}\|u_t(\boldsymbol{a})\|_2^2$. Thus, the approximation error is controlled by the magnitude of the victim's update and the smoothness of the fitted function. Theorem 1 in the next section further bounds the increase in F_t incurred by selecting a minimizer of $\widetilde{F}_t$. To minimize the surrogate, we denote $c_{t,i} := \frac{\eta_t}{m}\alpha_{t,i}, \rho := \frac{\lambda}{2m}$, and $q_{t,i}(s) := c_{t,i}s + \rho|s - 1|$.

Apart from the action-independent constant $C_t(\theta)$, the surrogate is $\sum_{i=1}^m q_{t,i}(a_i)$, a sum of convex coordinate costs. Their marginal costs are

$$d_{t,i}(s) := q_{t,i}(s+1) - q_{t,i}(s) = \begin{cases} c_{t,i} - \rho, & s = 0, \\ c_{t,i} + \rho, & s \geq 1. \end{cases}$$

We adopt the greedy allocation algorithm of Minamikawa & Shioura (2019) and incorporate the maximum repetition bounds $a_i \leq L$. Starting from $\boldsymbol{a} = \mathbf{0}$, the procedure allocates m units sequentially, choosing a feasible step with the smallest marginal cost.An increment to the sample i is allowed when $a_i < L$ and either $a_i = 0$ or $\sum_{i=1}^m (a_i - 1)_+ < \lfloor B/2 \rfloor$.

These conditions enforce the feasibility structure that guarantees the optimality of the greedy allocation when applied to a separable convex objective. Algorithm 2 requires at most $O(m^2)$ to solve the minimization problem over the action space compared to $O((L+1)^m)$ of the original minimization problem. Algorithm 2 in Appendix A.4 summarizes this procedure.

4 Theoretical Analysis

The main challenges for an attacker in our setting that prevent them from deriving the optimal policy, π^*, are the lack of knowledge about the data-generating distribution $\mathcal{P}_{\mathcal{Z}}^{\otimes m}$ and the difficulty of solving the Bellman minimization in equation 4. The latter is particularly aggravating because

the minimization is required to both construct regression targets in Algorithm 1 and to select actions from the fitted sequence $\{h_t\}_{t=0}^{T-1}$. Our analysis examines how these approximations affect the attacker's expected cumulative cost, measured by the *optimality gap*, $J(\widehat{\pi}) - J(\pi^*)$, with both policies evaluated under the true data-generating distribution $\mathcal{P}_{\mathcal{Z}}^{\otimes m}$.

We first isolate the error introduced by surrogate optimization by fixing the fitted function h_{t+1}. Theorem 1 bounds the increase in F_t when the surrogate minimizer $\widehat{\boldsymbol{a}}$ is evaluated instead of the true minimizer $\boldsymbol{a}$. This theorem establishes the local approximation error that will be used in the analysis of the optimality gap (Theorem 2).

Theorem 1. *Under Assumption 1, 2, $||\phi_i||_2 \leq G$ for some constant $G \geq 0$ and for every $i \in [m]$, and h_{t+1} is continuously differentiable on $\mathcal{D}_t = \{x : ||x - \theta||_2 \leq \eta_t G\}$, denote the surrogate optimal action $\widehat{\boldsymbol{a}} \in \arg\min_{\boldsymbol{a}\in\mathcal{A}} \tilde{F}_t(\boldsymbol{a})$ and true optimal action $\boldsymbol{a}^* \in \arg\min_{\boldsymbol{a}\in\mathcal{A}} F_t(\boldsymbol{a})$. Then, for each $t = 0, \ldots, T-1$, the approximation error is bounded such that*

$$0 \leq F_t(\widehat{\boldsymbol{a}}) - F_t(\boldsymbol{a}^*) \leq (1 + \Lambda_{t+1})\eta_t^2 G^2 \min\left\{1, \frac{2B}{m}\right\} =: \kappa_t.$$

Proof. Appendix A.2 □

Our analysis extends the finite-sample analysis of the classical Fitted-Q Iteration (Agarwal et al., 2026) to incorporate the approximation error from Theorem 1 and the unique transition probability structure. The full list of assumptions and notation is shown in Appendix A.3.

Theorem 2. *Let π^* be the optimal policy on $\mathcal{M}$ and $\widehat{\pi}$ be the optimal policy derived from solving the surrogate problem. Then, with probability at least $1 - \delta$,*

$$0 \leq J(\widehat{\pi}) - J(\pi^*) \leq O\left(C_{\max}\sqrt{C_{cov}}T^2\sqrt{\frac{\log(2F_{max}T/\delta)}{n}} + T\sqrt{C_{cov}\mathcal{E}_{app}} + \sqrt{C_{cov}}\sum_{t=0}^{T-1}\kappa_t\right).$$

where C_{cov}, $\mathcal{E}_{app}$, and F_{max} are constants defined in Appendix A.3

Proof. Appendix A.3 □

Theorem 2 tells us that the FEVI algorithm achieves a finite-sample bound of order $O(n^{-1/2})$, up to logarithmic factors. The constant C_{cov} captures how well the training distribution covers the parameter values encountered during an attack. When function approximation $\mathcal{E}_{\text{app}}$ and surrogate optimization κ_t errors are present, they contribute additional terms that need not vanish as $n \to \infty$.

5 EXPERIMENTS

In this section, we empirically compare the proposed algorithm *FEVI* (Algorithm 1 with greedy allocation 2) against the model-free RL algorithms that can handle the combinatorial space. To measure their performance, we report multiple metrics, including **empirical cumulative cost** $\widetilde{J}(t) := \sum_{\tau=0}^{t} g_\tau(\theta_\tau, \mathbf{z}_\tau, \boldsymbol{a}_\tau)$, **validation set cost**, and **time spent in training**. These metrics measure the strength of the attacks: the stronger and more efficient the attack, the lower the cost and training time. These algorithms are compared against the baseline attackers.

Null attack represents the upper bound of the empirical cumulative cost $\widetilde{J}(t)$, which is equivalent to setting $\boldsymbol{a}_t^{\text{null}} = 0$ for all $t = 0, \ldots, T-1$. **Greedy attack** concerns only the immediate cost g_t while ignoring the future cost approximated by $\bar{V}_t$. At each stage $t = 0, \ldots, T-1$, the greedy attacker either solves $\boldsymbol{a}_t^{\text{greedy}} = \arg\min_{\boldsymbol{a}\in\mathcal{A}} g_t(\theta_t, \mathbf{z}_t, \boldsymbol{a})$ by optimization software like Gurobi or approximately solves it by greedy allocation (Algorithm 2). In this work, we use a Gurobi academic license to solve the immediate cost minimization problem directly.

For the model-free RL baselines, we employ **Trust-Region Policy Optimization** (TRPO) (Schulman et al., 2015) and **Branching Dueling Q-Network** (BDQ) (Tavakoli et al., 2018). TRPO is an on-policy policy-gradient method that produces actions from a parameterized policy rather than by

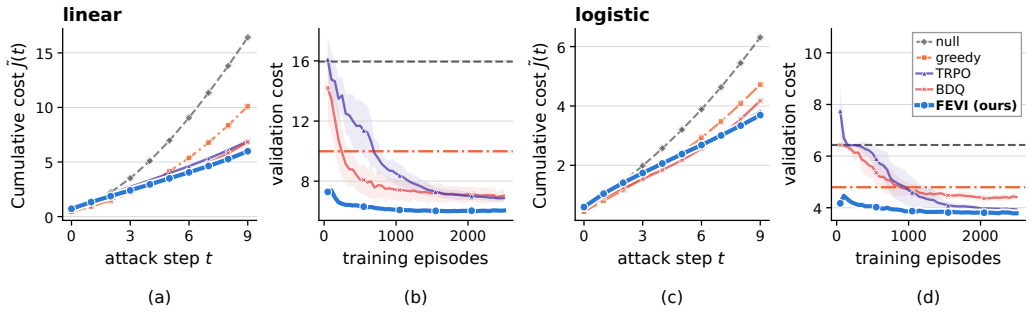

Figure 1: **Attack performance on simulated data**. Empirical cumulative cost $\widetilde{J}(t)$ after 2,500 training episodes (a, c) and validation cost during training (b, d), against the online linear (a, b) and online logistic (c, d) regression victims. Lower cost indicates a stronger attack.

optimizing over the action space. Our constraints, however, couple the decisions to delete or duplicate each observation, so the policy cannot simply choose an edit for each point independently. We are required to design the actor network architecture to enforce the constraint.

In contrast, BDQ (Tavakoli et al., 2018) is a value-based method that uses a shared network to process the state, including the entire batch, and a single advantage head shared across samples to estimate the value of each sample's possible deletion/duplication. The constrained action selection in BDQ reduces to our greedy allocation (Algorithm 2) with scores derived from BDQ's per-sample values in place of α_i. However, each BDQ update performs many gradient steps that evaluate every sample of every transition in the minibatch.

5.1 Experiments: simulated data

We study two victim models: **online linear regression** and **online logistic regression**. For each task we draw $\tilde{\theta} \sim \mathcal{N}(0, \boldsymbol{I}_d)$ and set $\theta^\star = \gamma\,\tilde{\theta}/\|\tilde{\theta}\|$, so that $\theta^\star$ is uniform on the sphere of radius γ, and draw features $\mathbf{x}_{t,i} \sim \mathcal{N}(0, \boldsymbol{I}_d)$. Labels are generated from the respective model. For the linear victim, $\mathbf{y}_t = \boldsymbol{X}_t\theta^\star + \epsilon_t$ with $\epsilon_t \sim \mathcal{N}(0, \sigma^2\boldsymbol{I}_m)$ and design matrix $\boldsymbol{X}_t \in \mathbb{R}^{m\times d}$. For the logistic victim, we draw $\mathbf{y}_{t,i} \sim \mathrm{Bernoulli}\big(\mathrm{logit}^{-1}(\langle \mathbf{x}_{t,i}, \theta^\star\rangle)\big)$. The radius γ controls the signal strength, as studied by Sur & Candès (2019). In our case, we fix $\gamma = \sqrt{5}$ for the linear victim and $\gamma = 2$ for the logistic victim. We also set $\sigma^2 = 0.01$. For both tasks, we randomly draw the target $\theta^\dagger \sim \mathcal{N}(0, \boldsymbol{I}_d)$.

For both tasks, we set the maximum repetition $L = 2$, maximum per-batch delete/duplicate $B = 10$, feature dimension $d = 5$, batch size $m = 10$, horizon $T = 10$, and penalty $\lambda = 1$. The victim's learning rate is set to $\eta = 0.2$ for the linear victim and $\eta = 1$ for the logistic victim. All neural networks are fully connected and have two hidden layers, each with 128 nodes. The weight decay is set at 10^{-3} to improve fitting. In this work, we restrict the function class of h to fully-connected neural networks. More details about the experimental setting can be found in Appendix A.5.

We train every RL method (FEVI, TRPO, and BDQ) for 2,500 episodes, updating each model every 50 episodes and evaluating on a fixed set of 200 validation episodes. FEVI and BDQ are off-policy methods, so we run an ϵ-greedy policy with decaying ϵ. After training, we test each model on 200 new episodes and report its average episodic cost. Each experiment is repeated for 10 seeds. Figure 1 shows the average validation episodic cost and the empirical cumulative cost for both tasks.

On the linear regression victim, every attacker improves on the null attack, which incurs a cumulative cost of 16.42 over the horizon. The greedy attacker reduces this to 10.12, and the RL methods reduce it further. TRPO and BDQ reach 6.89 and 6.84, while FEVI attains the lowest cost, 5.99. On the logistic regression victim, the null attack incurs 6.31, and the greedy attacker, BDQ, and TRPO reduce it to 4.72, 4.17, and 3.82, respectively. FEVI again attains the lowest cost, 3.69, although its margin over TRPO narrows from about 13% on the linear victim to about 3%. FEVI's advantage is most pronounced in how quickly it learns. As panels (b) and (d) show, FEVI outperforms the

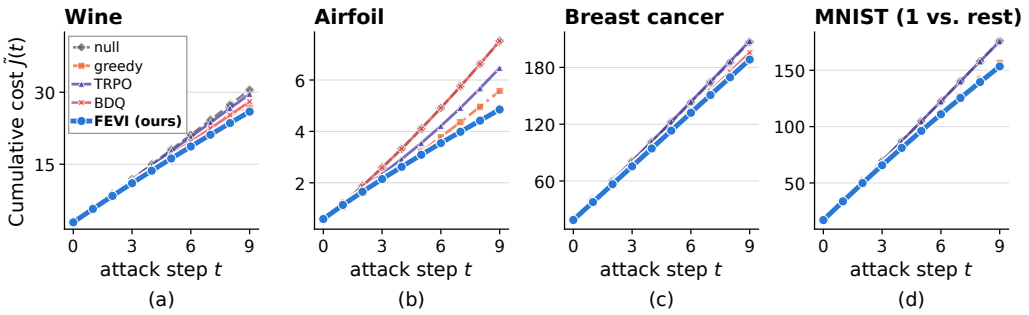

Figure 2: **Attack performance on real data**. Empirical cumulative cost $\widetilde{J}(t)$ after 1,000 training episodes against the online linear regression victim on Wine (a) and Airfoil (b), and the online logistic regression victim on Breast Cancer (c) and MNIST, digit 1 versus the rest (d). Lower cost indicates a stronger attack.

greedy attacker from its first evaluation and converges within roughly 500 episodes on both victims, whereas TRPO and BDQ need well over 1,000 episodes to plateau.

FEVI is also cheaper per episode: BDQ takes on average 1097.6 seconds to train for 2,500 episodes, nearly ten times longer than FEVI (114.6 seconds) or TRPO (123.8 seconds) on the linear regression task. For the logistic regression task, FEVI trains for 2500 episodes in 127.4 seconds on average, while TRPO uses 138.9 seconds and BDQ uses almost three times longer than FEVI (376.3 seconds).

5.2 Experiments: Real Data

To evaluate our method beyond synthetic data, we use four real datasets: Wine $(d = 11)$, Airfoil $(d = 5)$, Breast Cancer $(d = 30)$, and MNIST $(d = 784)$. We obtain the first three datasets from the UCI Machine Learning Repository (Kelly et al.) and MNIST through the PyTorch library. We standardize every feature to zero mean and unit variance. For MNIST, we further reduce the dimension from 784 to 30 by PCA, projecting onto the 30 leading principal components. At each stage $t = 1, \ldots, T-1$, m samples are independently drawn with replacement.

Similar to the synthetic data experiment, we first train FEVI, TRPO, and BDQ with 1000 episodes and update the models every 50 episodes. To ensure that the victim will be able to fit its learning model to the dataset well, we conducted a grid search and set the victim's learning rate at $\eta = 0.1, 0.2, 1, 0.3$ for Wine, Airfoil, Breast Cancer, and MNIST datasets, respectively. Figure 2 shows the empirical cumulative cost across attack steps of all datasets.

FEVI attains the lowest cumulative cost on all four real datasets, reducing the cost of the null attack by 14.9% on Wine, 35.5% on Airfoil, 9.0% on Breast Cancer, and 12.8% on MNIST. The model-free baselines are less reliable than on synthetic data. On Airfoil, BDQ's cost coincides with that of the null attack at every step, indicating that it learned not to edit the batch at all, while TRPO barely improves on the null attack on MNIST and slightly exceeds it on Breast Cancer. As a result, the greedy attacker outperforms both RL baselines on three of the four datasets.

6 Conclusion

In this paper, we frame online data poisoning as a sequential decision problem in which an attacker deletes and duplicates observations in each batch under balance and budget constraints. Formulating the attack as a finite-horizon MDP with a constrained combinatorial action space, we propose FEVI, a fitted value iteration method whose value depends only on the model parameters and whose actions solve the constrained selection problem directly. On synthetic and real data, FEVI attains lower cumulative cost than all baselines while requiring only a fraction of BDQ and TRPO training episodes. Lastly, we provided an analysis of the optimality gap and the approximation error of the linearized surrogate objective function. The future direction includes a low-rank approximation of the pairwise interactions in the original objective function.

AI USE STATEMENT

In this work, we used generative AI tools in the form of Claude's Opus 5, Opus 5.5, and ChatGPT's Sol 5.6 for literature search, proofreading, polishing the writing, and discussing proof ideas. We also used Claude Code to help implement the experiment. The code is a collaborative effort of the authors and the generative AI tools. However, generative AI tools played no role in formulating the research question or deriving the proposed solution. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The code that generates Figures 1 and 2 can be found in GitHub.

REFERENCES

Alekh Agarwal, Kianté Brantley, Nan Jiang, Sham M. Kakade, and Wen Sun. Reinforcement learning: Theory and algorithms, 2026. URL https://rltheorybook.github.io/rltheorybook_ABJKS.pdf. Working draft, June 27, 2026.

Battista Biggio and Fabio Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331, 2018.

Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Conference on Machine Learning*, ICML'12, pp. 1467–1474, Edinburgh, Scotland, 2012. Omnipress.

Battista Biggio, Giorgio Fumera, and Fabio Roli. Security evaluation of pattern classifiers under attack. *IEEE Transactions on Knowledge and Data Engineering*, 26(4):984–996, 2014. doi: 10.1109/TKDE.2013.57.

Tamara Broderick, Ryan Giordano, and Rachael Meager. An automatic finite-sample robustness metric: When can dropping a little data make a big difference? *arXiv preprint arXiv:2011.14999*, 2021.

Matthieu Carreau, Roi Naveiro, and William N. Caballero. Poisoning Bayesian inference via data deletion and replication. In *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, volume 258 of *Proceedings of Machine Learning Research*, pp. 883–891. PMLR, 2025. URL https://proceedings.mlr.press/v258/carreau25a.html.

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6(17):503–556, 2005.

Micah Goldblum, Dimitris Tsipras, Chulin Xie, Xinyun Chen, Avi Schwarzschild, Dawn Song, Aleksander Madry, Bo Li, and Tom Goldstein. Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):1563–1580, 2023. doi: 10.1109/TPAMI.2022.3162397.

Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. BadNets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7:47230–47244, 2019. doi: 10.1109/ACCESS.2019.2909068.

Kevin Jamieson. CSE 542: Statistical reinforcement learning, lecture 14: Fitted Q-iteration. University of Washington lecture slides, 2026. URL https://courses.cs.washington.edu/courses/cse542/26sp/resources/lecture_7.pdf.

Markelle Kelly, Rachel Longjohn, and Kolby Nottingham. The UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences. URL https://archive.ics.uci.edu.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Fuwei Li, Lifeng Lai, and Shuguang Cui. Optimal feature manipulation attacks against linear regression. *IEEE Transactions on Signal Processing*, 69:5580–5594, 2021.

Weiyang Liu, Bo Dai, Ahmad Humayun, Charlene Tay, Chen Yu, Linda B. Smith, James M. Rehg, and Le Song. Iterative machine teaching. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2149–2158. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/liu17b.html.

Yiwei Lu, Gautam Kamath, and Yaoliang Yu. Indiscriminate data poisoning attacks on neural networks. *Transactions on Machine Learning Research*, 2022. URL https://openreview.net/forum?id=x4hmIsWu7e.

Neil G. Marchant, Benjamin I. P. Rubinstein, and Scott Alfeld. Hard to forget: Poisoning attacks on certified machine unlearning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 7691–7700, 2022. doi: 10.1609/aaai.v36i7.20736.

Riccardo Giuseppe Margiotta, Sebastian Goldt, and Guido Sanguinetti. Attacks on online learners: A teacher-student analysis. In *Advances in Neural Information Processing Systems*, volume 36, pp. 245–263, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/46e37aeccafc3b4b697b17b8a36f3b30-Abstract-Conference.html.

Shike Mei and Xiaojin Zhu. Using machine teaching to identify optimal training-set attacks on machine learners. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI-15)*, pp. 2871–2877, 2015. URL https://ojs.aaai.org/index.php/AAAI/article/view/9569.

Norito Minamikawa and Akiyoshi Shioura. Separable convex resource allocation problem with L_1-distance constraint. *Journal of the Operations Research Society of Japan*, 62(3):109–120, 2019. doi: 10.15807/jorsj.62.109.

Nicolas Michael Müller, Daniel Kowatsch, and Konstantin Böttinger. Data poisoning attacks on regression learning and corresponding defenses. In *2020 IEEE 25th Pacific Rim International Symposium on Dependable Computing (PRDC)*, pp. 80–89. IEEE, 2020. doi: 10.1109/PRDC50213.2020.00017.

Rémi Munos and Csaba Szepesvári. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9(27):815–857, 2008. URL http://jmlr.org/papers/v9/munos08a.html.

Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C. Lupu, and Fabio Roli. Towards poisoning of deep learning algorithms with back-gradient optimization. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 27–38, 2017.

Tianyu Pang, Xiao Yang, Yinpeng Dong, Hang Su, and Jun Zhu. Accumulative poisoning attacks on real-time data. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 13410–13423, 2021.

Warren B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. John Wiley & Sons, Hoboken, NJ, 2 edition, 2011.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1889–1897, Lille, France, 07–09 Jul 2015. PMLR.

Ali Shafahi, W. Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *Advances in Neural Information Processing Systems*, volume 31, pp. 6106–6116. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/hash/22722a343513ed45f14905eb07621686-Abstract.html.

Ilia Shumailov, Zakhar Shumaylov, Dmitry Kazhdan, Yiren Zhao, Nicolas Papernot, Murat A. Erdogdu, and Ross J. Anderson. Manipulating SGD with data ordering attacks. In *Advances in Neural Information Processing Systems*, volume 34, pp. 18021–18032, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/959ab9a0695c467e7caf75431a872e5c-Abstract.html.

Jacob Steinhardt, Pang Wei W. Koh, and Percy S. Liang. Certified defenses for data poisoning attacks. In *Advances in Neural Information Processing Systems*, volume 30, pp. 3517–3529, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/9d7311ba459f9e45ed746755a32dcd11-Abstract.html.

Pragya Sur and Emmanuel J. Candès. A modern maximum-likelihood theory for high-dimensional logistic regression. *Proceedings of the National Academy of Sciences*, 116(29):14516–14525, 2019. doi: 10.1073/pnas.1810420116.

Fnu Suya, Saeed Mahloujifar, Anshuman Suri, David Evans, and Yuan Tian. Model-targeted poisoning attacks with provable convergence. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9995–10006. PMLR, 18–24 Jul 2021.

Arash Tavakoli, Fabio Pardo, and Petar Kormushev. Action branching architectures for deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, pp. 4131–4138, 2018. doi: 10.1609/aaai.v32i1.11798. URL https://ojs.aaai.org/index.php/AAAI/article/view/11798.

Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Label-consistent backdoor attacks, 2019.

Yizhen Wang and Kamalika Chaudhuri. Data poisoning attacks against online learning. *arXiv preprint arXiv:1808.08994*, 2018.

Jialin Wen, Benjamin Zi Hao Zhao, Minhui Xue, and Haifeng Qian. PALOR: Poisoning attacks against logistic regression. In *Information Security and Privacy (ACISP 2020)*, volume 12248 of *Lecture Notes in Computer Science*, pp. 447–460. Springer, Cham, 2020. doi: 10.1007/978-3-030-55304-3_23.

Jialin Wen, Benjamin Zi Hao Zhao, Minhui Xue, Alina Oprea, and Haifeng Qian. With great dispersion comes greater resilience: Efficient poisoning attacks and defenses for linear regression models. *IEEE Transactions on Information Forensics and Security*, 16:3709–3723, 2021. doi: 10.1109/TIFS.2021.3087332.

Xuezhou Zhang, Xiaojin Zhu, and Laurent Lessard. Online data poisoning attacks. In *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, volume 120 of *Proceedings of Machine Learning Research*, pp. 201–210. PMLR, 2020. URL https://proceedings.mlr.press/v120/zhang20b.html.

Chen Zhu. Building the incremental and online training platform at LinkedIn. *LinkedIn Engineering Blog*, October 2025. URL https://www.linkedin.com/blog/engineering/infrastructure/incremental-and-online-training-platform-at-linkedin.

Chen Zhu, W. Ronny Huang, Hengduo Li, Gavin Taylor, Christoph Studer, and Tom Goldstein. Transferable clean-label poisoning attacks on deep neural nets. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7614–7623. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/zhu19a.html.

A Appendix

A.1 Proof of Markov Property

Proposition 1. *Let $H_t \coloneqq \sigma(\mathbf{a}_0, \mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_t)$ be the σ-algebra of the history up until time t. The stochastic process $\{\mathbf{s}_t\}_{t=0}^{T}$ has a Markov property such that*

$$\mathbb{P}(\mathbf{s}_{t+1} \in \cdot \mid H_t, \mathbf{a}_t) = \mathbb{P}(\mathbf{s}_{t+1} \in \cdot \mid \mathbf{s}_t, \mathbf{a}_t).$$

Proof. Let $\mathcal{G}_t := H_t \vee \sigma(\boldsymbol{a}_t)$ and fix a measurable set $E \subseteq \mathcal{S}$. Given $\mathcal{G}_t$, the next parameter $f_t(\theta_t, \mathbf{z}_t, \boldsymbol{a}_t)$ is determined, while the next batch is drawn independently from $\mathbb{P}_{\mathcal{Z}}^{\otimes m}$. Therefore,

$$\mathbb{P}(\mathbf{s}_{t+1} \in E \mid \mathcal{G}_t) = \int_{\mathcal{Z}^m} \mathbf{1}_E\big(f_t(\theta_t, \mathbf{z}_t, \boldsymbol{a}_t), \mathbf{z}'\big)\, \mathbb{P}_{\mathcal{Z}}^{\otimes m}(d\mathbf{z}').$$

The right-hand side is measurable with respect to $\sigma(\mathbf{s}_t, \boldsymbol{a}_t)$ and depends on no earlier history. Applying the tower property therefore yields

$$\mathbb{P}(\mathbf{s}_{t+1} \in E \mid H_t, \boldsymbol{a}_t) = \mathbb{P}(\mathbf{s}_{t+1} \in E \mid \mathbf{s}_t, \boldsymbol{a}_t) \quad \text{almost surely.}$$

□

A.2 Proof of Theorem 1

Proof. To be concise, let $u_t(a) \coloneqq \frac{\eta_t}{m}\sum_{i=1}^{m} a_i\phi_i$. Since $a_i \geq 0$ and $\sum_i a_i = m$, the triangle inequality implies:

$$||u(a)||_2 = \Big|\Big|\frac{\eta_t}{m}\sum_{i=1}^{m} a_i\phi_i\Big|\Big| \leq \frac{\eta_t}{m}\sum_{i=1}^{m} a_i||\phi_i||_2 \leq \frac{\eta_t G}{m}\sum_{i=1}^{m} a_i = \eta_t G. \tag{9}$$

Thus, $f(\theta, Z, \boldsymbol{a}) \in \mathcal{D}$, implying that all feasible update remains in the smooth region. Next, define $\psi_t(x) \coloneqq \frac{1}{2}||x - \theta^\dagger||_2^2 + h_{t+1}(x)$. Its gradient is $\nabla\psi_t(x) = x - \theta^\dagger + \nabla h_{t+1}(x)$. Consequently, for $x, y \in \mathcal{D}$,

$$||\nabla\psi_t(x) - \nabla\psi_t(y)||_2 \leq ||x - y||_2 + ||\nabla h_{t+1}(x) - \nabla h_{t+1}(y)||_2 \leq (1 + \Lambda_{t+1})||x - y||_2. \tag{10}$$

Define the approximation error: $r_t(x) \coloneqq \psi_t(x) - \psi_t(\theta) - \langle\nabla\psi_t(x), x - \theta\rangle$ By construction, we have

$$F_t(a) = \tilde{F}_t(a) + r_t\big(\theta + u(a)\big). \tag{11}$$

Then, the fundamental theorem of Calculus tells us that

$$\begin{aligned} r_t(\theta + u) &= \psi_t(\theta + u) - \psi_t(\theta) - \langle\nabla\psi_t(\theta), u\rangle \\ &= \int_0^1 \langle\nabla\psi_t(\theta + su), u\rangle\, ds - \langle\nabla\psi_t(\theta), u\rangle \\ &= \int_0^1 \langle\nabla\psi_t(\theta + su) - \nabla\psi_t(\theta), u\rangle\, ds. \end{aligned}$$

Applying Cauchy-Schwarz inequality and equation 10,

$$\begin{aligned} |r_t(\theta + u)| &\leq \int_0^1 ||\nabla\psi_t(\theta + su) - \nabla\psi_t(\theta)||_2||u||_2\, ds \\ &\leq (1 + \Lambda_{t+1})\int_0^1 s||u||_2^2\, ds \\ &= \frac{1 + \Lambda_{t+1}}{2}||u||_2^2 \\ &\leq \frac{1 + \Lambda_{t+1}}{2}\eta_t^2 G^2. \qquad \text{(by equation 9)} \end{aligned}$$

Therefore, for any feasible actions $\boldsymbol{a}, \boldsymbol{b} \in \mathcal{A}$, we have

$$|r_t(\theta + u(a)) - r_t(\theta + u(b))| \leq |r_t(\theta + u(a))| + |r_t(\theta + u(b))| \leq (1 + \Lambda_{t+1})\eta_t^2 G^2. \tag{12}$$

On the other hand, we may approach this bound from the editing budget constraint. First, notice that for any feasible actions $\boldsymbol{a}, \boldsymbol{b} \in \mathcal{A}$,

$$||\boldsymbol{a} - \boldsymbol{b}||_1 \leq ||\boldsymbol{a} - \mathbf{1}||_1 + ||\boldsymbol{b} - \mathbf{1}||_1 \leq 2B.$$

Thus,

$$||u(a) - u(b)||_2 = \left|\left|\frac{\eta_t}{m}\sum_{i=1}^{m}(a_i - b_i)\phi_i\right|\right| \leq \frac{\eta_t G}{m}||a - b||_1 \leq \frac{2\eta_t GB}{m}.$$

Moreover,

$$\begin{aligned} ||\nabla r_t(x)||_2 &= ||\nabla\psi_t(x) - \nabla\psi_t(\theta)||_2 \\ &\leq (1 + \Lambda_{t+1})||x - \theta||_2 \\ &\leq (1 + \Lambda_{t+1})\eta_t G. \end{aligned} \tag{13}$$

Integrating this gradient along the segment between $x_a = \theta + u(a)$ and $x_b = \theta + u(b)$, we obtain:

$$\begin{aligned} |r_t(x_a) - r_t(x_b)| &= \left|\int_0^1 \langle\nabla r_t(x_b + s(x_a - x_b)), x_a - x_b\rangle\, ds\right| \\ &\leq (1 + \Lambda_{t+1})\eta_t G||x_a - x_b||_2 \\ &\leq \frac{2(1 + \Lambda_{t+1})\eta^2 G^2 B}{m}. \end{aligned}$$

Combining equation 12 and equation 13 yields

$$|r_t(\theta + u(a)) - r_t(\theta + u(b))| \leq (1 + \Lambda_{t+1})\eta_t^2 G^2 \min\left\{1, \frac{2B}{m}\right\}. \tag{14}$$

Lastly, since $\widehat{\boldsymbol{a}}$ minimizes the surrogate objective, we have that $\tilde{F}_t(\widehat{\boldsymbol{a}}) \leq \tilde{F}_t(\boldsymbol{a}^*)$. Using the decomposition in equation 11,

$$\begin{aligned} F_t(\widehat{\boldsymbol{a}}) - F_t(\boldsymbol{a}^*) &= \tilde{F}_t(\widehat{\boldsymbol{a}}) - \tilde{F}_t(\boldsymbol{a}^*) + r_t(\theta + u(\hat{\boldsymbol{a}})) - r_t(\theta + u(\boldsymbol{a}^*)) \\ &\leq r_t(\theta + u(\hat{\boldsymbol{a}})) - r_t(\theta + u(\boldsymbol{a}^*)) \\ &\leq (1 + \Lambda_{t+1})\eta_t^2 G^2 \min\left\{1, \frac{2B}{m}\right\}. \end{aligned}$$

The non-negative lower bound follows from the optimality of $\boldsymbol{a}^*$ of F_t. □

A.3 Proof of Theorem 2

We assume without loss of generality that

1. $0 \leq g_t \leq C_{\max}$ for all $t = 0, \ldots, T-1$.
2. For each $t = 0, \ldots, T-1$, let $\mathcal{F}_t \subseteq \{h : \Theta \to [0, (T-t)C_{\max}]\}$ and $F_T \equiv \{0\}$, we define $F_{\max} \coloneqq \max_{0\leq t\leq T} |\mathcal{F}_t|$.
3. (Concentrability) Suppose that sample j is independently sampled such that $\theta_t^{(j)} \sim \nu_t$ and $\mathbf{Z}_t^{(j)} \sim \mathcal{P}_{\mathcal{Z}}^{\otimes m}$, then let ρ_t^π be the distribution of θ_t under policy π starting from μ, we have

$$\rho_t^\pi \ll \nu_t, \qquad \left\|\frac{d\rho_t^\pi}{d\nu_t}\right\|_\infty \leq C_{\text{cov}}.$$

   imposing over all admissible policies π.
4. (Inherent Bellman error) For the Bellman operator,

$$(\mathcal{T}_t v)(\theta) \coloneqq \mathbb{E}_{\mathbf{Z}\sim\mathcal{P}_{\mathcal{Z}}^{\otimes m}}\left[\min_{a\in\mathcal{A}}\{g_t(\theta, \mathbf{Z}, \boldsymbol{a}) + v(f_t(\theta, \mathbf{Z}, \boldsymbol{a}))\}\right].$$

   Then, the inherent Bellman error is given by

$$\mathcal{E}_{\text{app}} = \max_{0\leq t<T}\sup_{v\in\mathcal{F}_{t+1}}\inf_{h\in\mathcal{F}}||h - \mathcal{T}_t v||^2_{L^2(\nu_t)}.$$

Theorem 2 (Full Version). *For any $\delta \in (0,1)$, with probability at least $1-\delta$, we have*

$$J(\hat{\pi}) - J(\pi^*) \leq C_{max} T(T+1)\sqrt{20 C_{cov}}\sqrt{\frac{\log(2TF_{max}/\delta)}{n}} + 2T\sqrt{3C_{cov}\mathcal{E}_{app}} + \left(1 + 2\sqrt{3C_{cov}}\right)\sum_{t=0}^{T-1}\kappa_t.$$

Proof. Let $U_t = (T-t)C_{\max}$ and let $\mathcal{F}_t \subseteq \{v : \Theta \to [0, U_t]\}$ be the finite function class at stage t, with $\mathcal{F}_T = \{0\}$ and $h_T \equiv 0$.

For a fitted function $v \in \mathcal{F}_{t+1}$, define

$$Q_t^v(\theta, \mathbf{z}, \mathbf{a}) = g_t(\theta, \mathbf{z}, \mathbf{a}) + v(f_t(\theta, \mathbf{z}, \mathbf{a}))\,,$$

and let $\mathbf{a}_t^v(\theta, \mathbf{z})$ be a minimizer of its linearized surrogate 8. Introduce the operator

$$(\widetilde{\mathcal{T}}_t v)(\theta) = \mathbb{E}_{\mathbf{Z}}[Q_t^v(\theta, \mathbf{Z}, \mathbf{a}_t^v(\theta, \mathbf{Z}))]\,.$$

Thus, $\widetilde{\mathcal{T}}_t v$ evaluates the original Bellman objective at the action selected by the surrogate. The fitted policy is $\hat{\pi}_t(\theta, \mathbf{z}) = \mathbf{a}_t^{h_{t+1}}(\theta, \mathbf{z})$. By Theorem 1 and the smoothness assumptions 2,

$$0 \leq Q_t^v(\theta, \mathbf{z}, \mathbf{a}_t^v(\theta, \mathbf{z})) - \min_{\mathbf{a}\in\mathcal{A}} Q_t^v(\theta, \mathbf{z}, \mathbf{a}) \leq \kappa_t,$$

where

$$\kappa_t = (1 + \Lambda_{t+1})\eta_t^2 G^2 \min\left\{1, \frac{2B}{m}\right\}.$$

Taking expectations over an independent batch $\mathbf{Z}$ gives the pointwise operator bound:

$$0 \leq \widetilde{\mathcal{T}}_t v - \mathcal{T}_t v \leq \kappa_t. \tag{15}$$

Fix a stage t and condition on the data used at stages $t+1, \ldots, T-1$. Then, h_{t+1} is fixed and independent of the stage-t samples. For $X_{t,i} \sim \nu_t$ and an independent fresh batch $\mathbf{Z}_{t,i} \sim P_Z^{\otimes m}$, the regression target is

$$Y_{t,i} = Q_t^{h_{t+1}}(X_{t,i}, \mathbf{Z}_{t,i}, \hat{\pi}_t(X_{t,i}, \mathbf{Z}_{t,i}))\,.$$

Its conditional mean is

$$q_t(\theta) := \mathbb{E}[Y_{t,i} \mid X_{t,i} = \theta, h_{t+1}] = (\widetilde{\mathcal{T}}_t h_{t+1})(\theta).$$

Since $0 \leq g_t \leq C_{\max}$ and $h_{t+1} \in [0, U_{t+1}]$, both $Y_{t,i}$ and q_t take values in $[0, U_t]$.

We may therefore apply the finite-class least-squares generalization bound of (Agarwal et al., 2026, Lemma A.11) to the empirical squared-loss minimizer h_t. With conditional probability at least $1 - \delta/T$,

$$\|h_t - q_t\|_{L^2(\nu_t)}^2 \leq \frac{20U_t^2\ell_t}{n} + 3\inf_{u\in\mathcal{F}_t}\|u - q_t\|_{L^2(\nu_t)}^2, \qquad \ell_t = \log\frac{2T|\mathcal{F}_t|}{\delta}.$$

Equation 15 and the triangle inequality for ℓ_2-normed space imply

$$\begin{aligned}\inf_{u\in\mathcal{F}_t}\|u - q_t\|_{L^2(\nu_t)} &\leq \inf_{u\in\mathcal{F}_t}\|u - \mathcal{T}_t h_{t+1}\|_{L^2(\nu_t)} + \|\mathcal{T}_t h_{t+1} - q_t\|_{L^2(\nu_t)} \\ &\leq \sqrt{\epsilon_{\text{app}}} + \kappa_t.\end{aligned}$$

Removing the conditioning and taking a union bound over the T stages shows that, with probability at least $1-\delta$, simultaneously for every t,

$$\|\Delta_t\|_{L^2(\nu_t)}^2 \leq \frac{20U_t^2\ell_t}{n} + 3\left(\sqrt{\epsilon_{\text{app}}} + \kappa_t\right)^2, \qquad \Delta_t := h_t - \widetilde{\mathcal{T}}_t h_{t+1}. \tag{16}$$

We now adapt the finite-horizon error-propagation argument of (Jamieson, 2026, Lemma A). Let $\bar{V}_t^\pi(\theta)$ denote the expected remaining cost before the stage-t batch is observed, and let ρ_t^π be the corresponding distribution of θ_t. For a fixed policy π, define

$$(\mathcal{K}_t^\pi u)(\theta) = \mathbb{E}_{\mathbf{Z}}[u(f_t(\theta, \mathbf{Z}, \pi_t(\theta, \mathbf{Z})))]\,.$$

Also set $d_t = \widetilde{\mathcal{T}}_t h_{t+1} - \mathcal{T}_t h_{t+1}$. By equation 15, $0 \leq d_t \leq \kappa_t$.

The correction needed for the surrogate follows from the two Bellman relations, both of which are recursions:

$$h_t - \bar{V}_t^* \le \Delta_t + d_t + \mathcal{K}_t^{\pi^*}(h_{t+1} - \bar{V}_{t+1}^*),$$
$$\bar{V}_t^{\hat{\pi}} - h_t = -\Delta_t + \mathcal{K}_t^{\hat{\pi}}(\bar{V}_{t+1}^{\hat{\pi}} h_{t+1}).$$

To derive these relations, notice that $\bar{V}_T^{\pi} = \bar{g}_t^{\pi} + \mathcal{K}_t^{\pi}\bar{V}_{t+1}^{\pi}$, where $\bar{g}_t^{\pi} = \mathbb{E}_{\mathbf{Z}}\left[g_t(\theta, \mathbf{Z}, \pi_t(\theta, \mathbf{Z}))\right]$. The exact Bellman operator satisfies $\mathcal{T}_t v \le \bar{g}_t^{\pi} + \mathcal{K}_t^{\hat{\pi}} v$ for every policy π because the policy minimizing over actions cannot cost more than the action prescribed by any policy π. Next, because $\hat{\pi}_t$ is the policy derived from the surrogate, we can write

$$\widetilde{\mathcal{T}}_t h_{t+1} = \bar{g}_{t+1}^{\hat{\pi}} + \mathcal{K}_t^{\hat{\pi}} h_{t+1}.$$

From the above definitions, we get the relation:

$$h_t = \Delta_t + d_t + \mathcal{T}_t h_{t+1} = \Delta_t + \widetilde{T}_t h_{t+1}. \tag{17}$$

For the first relation, we subtract $\bar{V}_t^*$ from both sides:

$$\begin{aligned} h_t - \bar{V}_t^* &= \Delta_t + d_t + \mathcal{T}_t h_{t+1} - \bar{V}_t^* \\ &= \Delta_t + d_t + \bar{g}_t^{\pi^*} + \mathcal{K}_t^{\pi^*} h_{t+1} - \bar{V}_t^* \\ &\le \Delta_t + d_t + \mathcal{K}_t^{\pi^*}\left(h_{t+1} - \bar{V}_{t+1}^*\right) \end{aligned}$$

The equality in the second line follows from applying the above equality with the optimal policy π^*. The last inequality holds because the optimal policy satisfies $\bar{V}_t^* = \bar{g}_t^{\pi^*} + \mathcal{K}_t^{\pi^*}\bar{V}_{t+1}^*$. Next, to get the second relation, we start from modifying equation 17:

$$h_t = \Delta_t + \bar{g}_t^{\hat{\pi}} + \mathcal{K}_t^{\hat{\pi}} h_{t+1}$$

and get:

$$\begin{aligned} \bar{V}_t^{\hat{\pi}} - h_t &= \bar{g}_t^{\hat{\pi}} + \mathcal{K}_t^{\hat{\pi}}\bar{V}_{t+1}^{\hat{\pi}} - \Delta_t - \bar{g}_t^{\hat{\pi}} - \mathcal{K}_t^{\hat{\pi}} h_{t+1} \\ &= -\Delta_t + \mathcal{K}_t^{\hat{\pi}}\left(\bar{V}_{t+1}^{\hat{\pi}} - h_{t+1}\right). \end{aligned}$$

From the standard Bellman error decompositions, we apply the telescoping argument to these relations, with $h_T = \bar{V}_T^{\pi} = 0$, which yields

$$J(\hat{\pi}) - J(\pi^*) \le \sum_{t=0}^{T-1}\left[\mathbb{E}_{\rho_t^{\pi^*}}[\Delta_t] - \mathbb{E}_{\rho_t^{\hat{\pi}}}[\Delta_t] + \mathbb{E}_{\rho_t^{\pi^*}}[d_t]\right].$$

Under concentrability, $\left|\mathbb{E}_{\rho_t^{\pi}}\Delta_t\right| \le \sqrt{C_{\mathrm{cov}}}\,\|\Delta_t\|_{L^2(\nu_t)}$for $\pi = \pi^*$ and $\pi = \hat{\pi}$. Together with $0 \le d_t \le \kappa_t$, this gives

$$J(\hat{\pi}) - J(\pi^*) \le 2\sqrt{C_{\mathrm{cov}}}\sum_{t=0}^{T-1}\|\Delta_t\|_{L^2(\nu_t)} + \sum_{t=0}^{T-1}\kappa_t. \tag{18}$$

Assuming that the event in equation 16 holds, substitution into equation 18 gives

$$J(\hat{\pi}) - J(\pi^*) \le 2\sqrt{C_{\mathrm{cov}}}\sum_{t=0}^{T-1}\sqrt{\frac{20U_t^2\ell_t}{n} + 3\left(\sqrt{\epsilon_{\mathrm{app}}} + \kappa_t\right)^2} + \sum_{t=0}^{T-1}\kappa_t.$$

Let$F_{\max} = \max_{0\le t<T}|\mathcal{F}_t|$ and $\ell = \log(2TF_{\max}/\delta)$. By the fact that $\ell_t \le \ell$, $\sqrt{x+y} \le \sqrt{x} + \sqrt{y}$, and

$$\sum_{t=0}^{T-1} U_t = \frac{C_{\max}T(T+1)}{2},$$

we finally obtain

$$\begin{aligned} J(\hat{\pi}) - J(\pi^*) \le{}& C_{\max}\sqrt{20C_{\mathrm{cov}}}\,T(T+1)\sqrt{\frac{\ell}{n}} + 2T\sqrt{3C_{\mathrm{cov}}\epsilon_{\mathrm{app}}} \\ &+ \left(1 + 2\sqrt{3C_{\mathrm{cov}}}\right)G^2\min\left\{1, \frac{2B}{m}\right\}\sum_{t=0}^{T-1}(1+\Lambda_{t+1})\eta_t^2. \end{aligned}$$

This establishes the claimed bound with probability at least $1-\delta$. □

A.4 Modified Greedy Allocation

Algorithm 2 Surrogate Minimization via Greedy Allocation (Minamikawa & Shioura, 2019)

Require: Parameters θ, batch $Z = (Z_i)_{i=1}^m$, fitted value function h_{t+1}, target $\theta^\dagger$, learning rate η_t, budget B, repetition limit L, penalty λ

Ensure: Feasible action a and Bellman-target contribution y

$\rho \leftarrow \lambda/2m$, $k \leftarrow \lfloor B/2 \rfloor$, $a \leftarrow \mathbf{0}$, $D \leftarrow 0$
$v \leftarrow \theta - \theta^\dagger + \nabla h_{t+1}(\theta)$
for $i = 1, \ldots, m$ **do**
    $\phi_i \leftarrow \phi(\theta, Z_i)$
    $\alpha_i \leftarrow \frac{\eta_t}{m} \langle v, \phi_i \rangle$
end for
Define marginal costs $d_i(s) = \begin{cases} \alpha_i - \rho, & s = 0, \\ \alpha_i + \rho, & s \geq 1. \end{cases}$
for $r = 1, \ldots, m$ **do**
    $\mathcal{I} \leftarrow \{i \in \{1, \ldots, m\} : a_i < L,\ (a_i = 0 \text{ or } D < k)\}$
    Choose $i^\star \in \arg\min_{i \in \mathcal{I}} d_i(a_i)$
    if $a_{i^\star} \geq 1$ **then**
        $D \leftarrow D + 1$
    end if
    $a_{i^\star} \leftarrow a_{i^\star} + 1$
end for
$\theta^+ \leftarrow \theta + \frac{\eta_t}{m} \sum_{i=1}^m a_i \phi_i$
$y \leftarrow \frac{1}{2} \|\theta^+ - \theta^\dagger\|_2^2 + \rho \|a - \mathbf{1}\|_1 + h_{t+1}(\theta^+)$
return (a, y)

A.5 Experiment Specification

We specify the information about TRPO and BDQ, as well as the values of hyperparameters used in both the synthetic and real data experiments as follows:

1. **Feature engineering for TRPO and BDQ**: Since FEVI exploits the victim's per-sample updates $\phi_i = -\nabla l(\theta_t, z_{t,i})$ when computing its scores $\alpha_{t,i}$, for a fair comparison, we provide the same information to TRPO and BDQ. All networks (BDQ's Q-network and TRPO's actor and critic) receive the following inputs:
    (a) the victim's current parameters $\theta_t \in \mathbb{R}^d$;
    (b) their offset from the target, $\theta_t - \theta^\dagger \in \mathbb{R}^d$;
    (c) a one-hot encoding of the attack step t, in $\{0, 1\}^T$;
    (d) for each sample i, its scaled update $\frac{\eta_t}{m} \phi_i \in \mathbb{R}^d$;
    (e) for each sample i, its residual $e_i \in \mathbb{R}$, the victim's current prediction minus the label: $e_i = \langle \theta_t, x_{t,i} \rangle - y_{t,i}$ for the linear victim and $e_i = \sigma(\langle \theta_t, x_{t,i} \rangle) - y_{t,i}$ for the logistic victim.

    Input variables in groups $(a), (b), (c)$ are concatenated and fed directly to the main network. On the other hand, variables in group (d) and (e) are embedded first
2. **TRPO**: We follow the model and training procedure of Schulman et al. (2015). The design of this actor network architecture is specific to $L = 2$ since it reduces the decision from deletion/duplication to swapping one sample out and another sample in. The actor outputs three quantities: logits over the number of swapping pairs $r \in \{0, \ldots, r_{\max}\}$, where $r_{\max} = \min\{\lfloor B/2 \rfloor, \lfloor m/2 \rfloor\}$ ($r_{\max} = 5$ in our experiments corresponding to $m = 10$ and $B = 10$), and a delete score and a duplicate score for each sample. During training, an action is sampled in three steps:
    (a) draw r from the softmax over the pair logits;
    (b) draw r samples to delete, one at a time and without replacement, from a softmax over the delete scores;

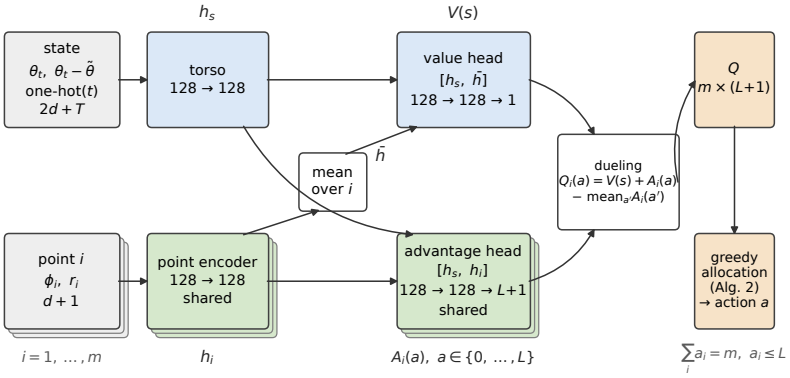

Figure 3: BDQ's neural network architecture

(c) draw r samples to duplicate in the same way from the duplicate scores, among the samples not deleted.

Deleted samples receive $a_i = 0$, duplicated samples $a_i = 2$, and all others $a_i = 1$. The log-probability of the action, required by the policy-gradient update, is the sum of the log-probabilities of the individual draws. At evaluation, the actor acts deterministically: it selects the most likely r, deletes the r samples with the highest delete scores, and duplicates the r remaining samples with the highest duplicate scores. Every action is feasible by construction since $\sum_i a_i = m$ and $\|\boldsymbol{a} - \mathbf{1}\|_1 = 2r \le B$.

For the linear regression victim, the critic is trained with learning rate 10^{-3} and takes 40 gradient steps per actor update. For the logistic regression victim, the learning rate is 0.1 with 100 critic steps per actor update. The KL trust-region radius is 0.05 for the linear victim and 0.1 for the logistic victim. All hyperparameters were tuned with penalty $\lambda = 1$. The critic is optimized with Adam (Kingma & Ba, 2015), and all networks use $\tanh$ activations. The networks are updated every 50 episodes. Since TRPO is on-policy, each update uses only the 50 most recent episodes, which are then discarded. For additional exploration, actions are sampled from the actor with probability $1 - \epsilon$ and drawn uniformly from the feasible set with probability ϵ, where $\epsilon = 0.5$ and gradually decays until $\epsilon = 0.05$ on the last update.

3. **BDQ**: We adapt the Branching Dueling Q-Network of Tavakoli et al. (2018) to batches of exchangeable samples. Given a state $s_t = (\theta_t, \mathbf{z}_t)$, a torso network maps the state vector $[\theta_t,\ \theta_t - \theta^\dagger,\ \text{onehot}(t)] \in \mathbb{R}^{2d+T}$ to an embedding h_s, and a point encoder, shared across samples, maps each per-sample feature $[\frac{\eta_t}{m}\phi_i,\ e_i] \in \mathbb{R}^{d+1}$ to an embedding h_i. An advantage head maps $[h_s, h_i]$ to advantages $A_i(s, \cdot) \in \mathbb{R}^{L+1}$ over $L + 1$ possible actions (delete, keep, or duplicate for at most L times), and a value head maps $[h_s, \frac{1}{m}\sum_i h_i]$ to a scalar $V(s)$. Following the dueling aggregation, the per-sample action values are $Q_i(s, a_i) = V(s) + A_i(s, a_i) - \frac{1}{L+1}\sum_a A_i(s, a)$, yielding an $m \times (L + 1)$ output. All networks use hidden width 128 and $\tanh$ activations. See Figure 3 for the diagram of this network architecture.

Our design is different from standard BDQ in three ways. Standard BDQ assigns each action dimension a branch with its own weights, which assumes that each dimension has a fixed meaning. Here, the i-th sample is an arbitrary draw from the batch, so they are exchangeable. We instead share a single advantage head across samples and condition it on each sample's features. The value head likewise pools over sample embeddings since the batch is part of the state. Finally, because our MDP has a finite horizon, the attack step enters the state as a one-hot encoding, and the values are costs to be minimized.

To select an action, the attacker applies greedy allocation (Algorithm 2) to the marginal costs $Q_i(s, a + 1) - Q_i(s, a)$, which yields a feasible action. The network is trained by double Q-learning with the mean-over-branches target of Tavakoli et al. (2018), $y = g_t + \frac{1}{m}\sum_{i=1}^{m} Q_i^-\big(s', \arg\min_a Q_i(s', a)\big)$, where Q^- is a target network refreshed by a hard copy at each update. The learning rate is set at $\eta = 3 \times 10^{-3}$ for the linear victim and

$\eta = 10^{-3}$ for the logistic victim. Since the BDQ is an off-policy method, it utilizes an ϵ-greedy policy to explore the environment and store triplets of state, action, and reward. Every 50 episodes, the model performs 1000 updates in the linear regression task and 250 updates in the logistic regression task via Adam.