Online Data Poisoning Attacks

Xuezhou Zhang ZHANGXZ1123@CS.WISC.EDU
Department of Computer Sciences, University of Wisconsin-Madison
Xiaojin Zhu JERRYZHU@CS.WISC.EDU
Department of Computer Sciences, University of Wisconsin-Madison
Laurent Lessard LAURENT.LESSARD@WISC.EDU
Department of Electrical and Computer Engineering, University of Wisconsin-Madison

Editors: A. Bayen, A. Jadbabaie, G. J. Pappas, P. Parrilo, B. Recht, C. Tomlin, M.Zeilinger

Abstract

We study data poisoning attacks in the online learning setting, where training data arrive sequentially, and the attacker is eavesdropping the data stream and has the ability to contaminate the *current data point* to affect the online learning process. We formulate the optimal online attack problem as a stochastic optimal control problem, and provide a systematic solution using tools from model predictive control and deep reinforcement learning. We further provide theoretical analysis on the regret suffered by the attacker for not knowing the true data sequence. Experiments validate our control approach in generating near-optimal attacks on both supervised and unsupervised learning tasks.

1. Introduction

Data poisoning attacks are a family of adversarial attack where an attacker contaminates the training data of a victim learner to control the learning process Xiao et al. (2015); Mei and Zhu (2015); Burkard and Lagesse (2017); Chen and Zhu (2019); Jun et al. (2018); Li et al. (2016). Prior work on data poisoning focused almost exclusively on the offline learning setting, where the attacker contaminates a set of pre-collected training points, and then the victim learner learns from the contaminated dataset. Biggio et al. (2012); Muñoz-González et al. (2017); Xiao et al. (2015); Mei and Zhu (2015); Sen et al. (2018); Chen et al. (2017). However, real-world machine learning systems often update in an online fashion. For example, in e-commerce applications, user-generated data arrives sequentially, and the learner usually updates daily to learn from the newly-acquired data.

Compared to the offline setting, the online attacker faces several unique challenges: (1) In the offline setting, it is often assumed that the attacker observes the whole dataset. However, in the online setting, the attacker can only observe the current data when making decision. (2) In the offline setting, the attacker only needs to make one decision, while in the online setting, the attacker is required to make a sequence of decisions to perform the attack through time. These unique challenges makes classic data poisoning attack framework unapplicable to the online regime.

This paper presents a principled study of online data poisoning attacks. Our key contributions inlcude: (1) we provide a mathematical framework for online poisoning attacks based on optimal control, (2) we design and implement two practical algorithms that achieve near-optimal attack performance in both synthetic and real-data experiments, and (3) We develop a theoretical analysis upper-bounding the regret suffered by the attacker for not knowing the real data sequence. Taken together, this paper builds a foundation for future studies of defense against online data poisoning.

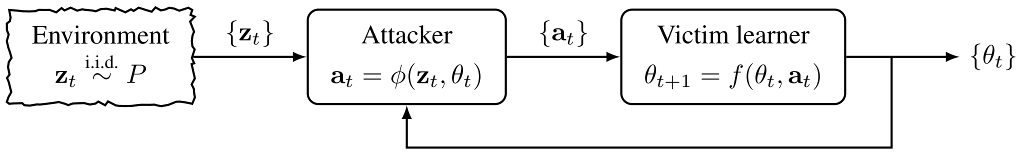

Environment
$\mathbf{z}_t \overset{\text{i.i.d.}}{\sim} P$
$\{\mathbf{z}_t\}$
Attacker
$\mathbf{a}_t = \phi(\mathbf{z}_t, \theta_t)$
$\{\mathbf{a}_t\}$
Victim learner
$\theta_{t+1} = f(\theta_t, \mathbf{a}_t)$
$\{\theta_t\}$

Figure 1: Online data poisoning attack diagram. At time t, the attacker observes the training sample $\mathbf{z}_t$ and the learner's model θ_t in an online fashion, and perturb $\mathbf{z}_t$ into $\mathbf{a}_t$.

2. Related Work

Data poisoning attacks have been studied against a wide range of learning systems, while focusing almsot exclusively on the offline settings. Examples of offline or batch poisoning attacks against SVM include Biggio et al. (2012); Burkard and Lagesse (2017); Xiao et al. (2015). Such attacks are generalized into a bilevel optimization framework against general offline learners with a convex objective function in Mei and Zhu (2015). A variety of attacks against other learners have been developed, including neural networks Koh and Liang (2017); Muñoz-González et al. (2017), autoregressive models Alfeld et al. (2016); Chen and Zhu (2019), linear and stochastic bandits Jun et al. (2018); Ma et al. (2018), collaborative filtering Li et al. (2016), and models for sentiment analysis Newell et al. (2014). In contrast, our paper focuses on the online setting, where the attacker has to act sequentially during the training process.

There is an intermediate attack setting between offline and online, which we call **clairvoyant online attacks**, where the attacker performs actions sequentially but **has full knowledge of all future input data** $\mathbf{z}_{t+1}, \mathbf{z}_{t+2}, \ldots$ Examples include heuristic attacks against SVM learning from data streams Burkard and Lagesse (2017) and binary classification with an online gradient descent learner Wang and Chaudhuri (2018). Our paper focuses instead on the more realistic setting where the attacker has no knowledge of the future data stream. More broadly, our paper advocates for a general optimal control viewpoint that is not restricted to specific learners such as SVM.

The parallel line of work studying **online teaching** also considers the sequential control problem of machine learners, where the goal is to choose a sequence of training examples that accelerates learning Liu et al. (2017); Lessard et al. (2018). However, Liu et al. (2017) solves the problem using a greedy heuristic that we show in Section 6 performs poorly compared to our optimal control approach. On the other hand, Lessard et al. (2018) finds optimal teaching sequences but is restricted to an ordinary linear regression learner.

3. Problem Definition

Figure 1 provides a overview of the online data poisoning problem. We consider three entities in this problem: a stochastic ***environment***, a sequential learning ***victim***, and the online ***attacker***. In the absence of attacks, at time t the environment draws a training data $\mathbf{z}_t \overset{\text{i.i.d.}}{\sim} P$. For example, $\mathbf{z}_t$ can be a feature-label pair $\mathbf{z}_t := (\mathbf{x}_t, y_t)$ in supervised learning or just the features $\mathbf{z}_t := \mathbf{x}_t$ in unsupervised learning. The victim maintains a model $\theta_t \in \Theta$. Upon receiving $\mathbf{z}_t$, the victim performs one step of the sequential update defined by a function f: $\theta_{t+1} = f(\theta_t, \mathbf{z}_t)$. For example, f can be the online gradient descent $f(\theta_t, \mathbf{z}_t) := \theta_t - \eta \nabla \ell(\theta_t, \mathbf{z}_t)$ under learner loss ℓ and step size η. We now introduce the attacker by defining its knowledge, allowed actions, and goals:

- **Knowledge:** We assume that at time t, the attacker has knowledge of the victim's update function f, the victim's current model θ_t, data $\mathbf{z}_{0:t}$ generated by the environment so far, and optionally n "pre-attack" data $\mathbf{z}_{-n:-1} \overset{\text{i.i.d.}}{\sim} P$. However, at time t the attacker does not have the clairvoyant knowledge of future data points $\mathbf{z}_{t+1}, \mathbf{z}_{t+2}, \ldots$, nor does it have the knowledge of the environment distribution P. The knowledge of f and θ_t are classic assumptions in the *white-box attack* setting Xiao et al. (2015); Mei and Zhu (2015); Burkard and Lagesse (2017); Chen and Zhu (2019); Jun et al. (2018); Li et al. (2016).

- **Action:** The attacker can perform only one type of action: At time step t, after the environment draws a data point $\mathbf{z}_t$, the attacker can perturb the data point into $\mathbf{a}_t \in \mathcal{Z}$. However, he is not able to go back to perturb any data points generated in previous iterations, which is distinct from the assumption in prior work Wang and Chaudhuri (2018). The perturbation incurs a perturbation cost $g_{\text{per}}(\mathbf{z}_t, \mathbf{a}_t)$, which reflects the price to attack. For example, $g_{\text{per}}(\mathbf{z}_t, \mathbf{a}_t) := \|\mathbf{a}_t - \mathbf{z}_t\|_p$ if $\mathcal{Z}$ is endowed with an appropriate p-norm.

- **Objective:** The attacker's goal, informally, is to force the victim's learned models θ_t to satisfy certain nefarious properties at each step while paying a small perturbation cost. These "nefarious properties" (rather the inability to achieve them) are captured by a nefarious cost $g_{\text{nef}}(\theta)$. For example, if the attacker wants to force the victim to learn/maintain a target model $\theta^\dagger$, we can define $g_{\text{nef}}(\theta) = \|\theta - \theta^\dagger\|^2$. To balance nefarious properties with perturbation cost, the attacker defines a *running cost* g at time t:

$$g(\theta_t, \mathbf{z}_t, \mathbf{a}_t) := g_{\text{per}}(\mathbf{z}_t, \mathbf{a}_t) + \lambda g_{\text{nef}}(\theta_t), \tag{1}$$

  where λ is a weight chosen by the attacker to balance the two. The attacker desires small cumulative running costs, which is the topic of Section 4.

4. An Optimal Control Formulation

We now precisely define the notion of **optimal** online data poisoning attacks. To do so, we cast the online data poisoning attack as a *Markov Decision Process (MDP)* $\mathcal{M} = (S, A, T, g, \gamma, \mathbf{s}_0)$.

- **State:** The state $\mathbf{s}_t$ at time t is the stacked vector $\mathbf{s}_t := [\theta_t, \mathbf{z}_t]^\top$ consisting of the victim's current model θ_t and the incoming environment data point $\mathbf{z}_t$. The state space is $S := \Theta \times \mathcal{Z}$.

- **Action:** The attacker's action is the perturbed training point $\mathbf{a}_t$. The action space is $A := \mathcal{Z}$.

- **Transition:** The state transition function $T : S \times A \to \Delta_S$ describes the conditional probability on the next state given current state and attack action, where Δ_S is the probability simplex over S. In our online attack problem, $T(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t) = T([\theta_{t+1}, \mathbf{z}_{t+1}]^\top \mid \mathbf{s}_t, \mathbf{a}_t) = Pr(f(\theta_t, \mathbf{a}_t) = \theta_{t+1}) \cdot P(\mathbf{z}_{t+1})$. For concreteness, in this paper, we focus on the case where f is deterministic, and thus the stochasticity is solely in the $\mathbf{z}_{t+1}$ component inside $\mathbf{s}_{t+1}$, which has a marginal distribution P. Therefore, $T\big([f(\theta_t, \mathbf{a}_t), \mathbf{z}_{t+1}]^\top \bigm| \mathbf{s}_t, \mathbf{a}_t\big) = P(\mathbf{z}_{t+1})$.

- **Cost:** The quality of control at time t is specified by the **running cost** $g(\theta_t, \mathbf{z}_t, \mathbf{a}_t)$ in (1), to be minimized. From now on, we overload the notation and write the running cost as $g(\mathbf{s}_t, \mathbf{a}_t)$.

- **Discounting factor:** We present online data poisoning attack with an infinite time horizon (the finite horizon case is similar but omitted due to space). We assume that the attacker cares about discounted cost with discounting factor γ.
- **Initial state distribution:** We assume that the initial model θ_0 is fixed and known to the attacker while the first data point $\mathbf{z}_0$ is sampled from P, i.e. the initial state distribution is defined as $\mu_0(\theta_0, \mathbf{z}_0) = P(\mathbf{z}_0)$.

A **policy** is a function $\phi : S \to A$ that the attacker uses to choose the attack action $\mathbf{a}_t := \phi(\mathbf{s}_t) = \phi([\theta_t, \mathbf{z}_t]^\top)$ based on the current victim model θ_t and the current environment input $\mathbf{z}_t$. The **value** $V^{\phi}_{\mathcal{M}}(\mathbf{s})$ of a state $\mathbf{s}$ is the expected discounted cumulative cost starting at $\mathbf{s}$ and following policy ϕ:

$$V^{\phi}_{\mathcal{M}}(\mathbf{s}) := \mathbb{E}_{\mathcal{M}} \sum_{t=0}^{\infty} \gamma^t g(\mathbf{s}_t, \phi(\mathbf{s}_t))\Big|_{\mathbf{s}_0=\mathbf{s}} \tag{2}$$

where the expectation is over the transition T. Overall, the attacker wants to find the optimal attack policy $\phi^\star_{\mathcal{M}}$ that minimizes the expected value at the initial state. Denote the attacker's objective as $J_{\mathcal{M}}(\phi) := \mathbb{E}_{\mathbf{s}\sim\mu_0} V^{\phi}_{\mathcal{M}}(\mathbf{s})$, and the attacker's optimal attack policy as $\phi^\star_{\mathcal{M}} = \arg\min_{\phi} J_{\mathcal{M}}(\phi)$.

It may seem fortunate for the victim that the attacker cannot directly solve this optimal attack problem because it does not know the environment data distribution P and thus cannot evaluate the expectation. However, as we show next, the attacker can use model predictive control to approximately and incrementally solve for the optimal attack policy as it gathers more information about P.

5. Practical Attack Algorithms via Model Predictive Control

The key obstacle that prevents the attacker from obtaining an optimal attack is the unknown data distribution P. However, the attacker can build an increasingly accurate **empirical distribution** $\hat{P}_t$ from the data seen so far. With $\hat{P}_t$ at time t, the attacker can construct a surrogate MDP $\hat{\mathcal{M}}_t = (S, A, \hat{T}_t, g, \gamma, \hat{\mu}_t)$, solve for the optimal policy $\phi^\star_{\hat{\mathcal{M}}_t} = \arg\min_\phi J_{\hat{\mathcal{M}}_t}(\phi)$ on $\hat{\mathcal{M}}_t$, and use $\phi^\star_{\hat{\mathcal{M}}_t}$ to perform the current attack action: $\mathbf{a}_t = \phi^\star_{\hat{\mathcal{M}}_t}(\mathbf{s}_t)$. This repeated procedure of (re)-planning ahead but only executing one action is called ***Model Predictive Control (MPC)*** Borrelli et al. (2017); Mayne et al. (2000). Next, we present two algorithms that practically solve the surrogate MDP, and discuss occasions in which we would prefer one over the other.

5.1. Algorithm NLP: Trajectory Planning with Model-based Nonlinear Programming

To perform exact planning over uncertain future trajectories, the attacker further approximates the surrogate objective as $J_{\hat{\mathcal{M}}_t}(\phi) \approx \mathbb{E}_{\hat{P}_t}\left[\sum_{\tau=t}^{t+h-1} \gamma^{\tau-t} g(\mathbf{s}_\tau, \phi(\mathbf{s}_\tau))\right] \approx \sum_{\tau=t}^{t+h-1} \gamma^{\tau-t} g(\mathbf{s}_\tau, \mathbf{a}_\tau)\Big|_{\mathbf{z}_{t:t+h-1}}$. The first approximation truncates at h steps after t, making it a finite-horizon control problem. The second approximation does two things: (i) It replaces the expectation by one sampled trajectory of the future input sequence, i.e. $\mathbf{z}_{t:t+h-1} \sim \hat{P}_t$. (ii) Instead of optimizing over a policy ϕ, it locally searches for the action sequence $\mathbf{a}_{t:t+h-1} \in \mathcal{Z}$. The attacker now solves the following nonlinear optimization problem at every time t: $\min_{\mathbf{a}_{t:t+h-1}} \sum_{\tau=t}^{t+h-1} \gamma^{\tau-t} g(\mathbf{s}_\tau, \mathbf{a}_\tau)$. Let $\mathbf{a}^\star_{t:t+h-1}$ be a solution. The NLP algorithm then performs $\mathbf{a}^\star_t$ and moves on to $t+1$. In practice, the attacker can solve modest-sized problems using modern nonlinear programming solvers such as IPOPT Wächter and Biegler (2006).

5.2. Algorithm DDPG: Policy Learning with Deep Reinforcement Learning

Instead of truncating and sampling to approximate the surrogate attack problem with a deterministic nonlinear program, one can directly solve for the optimal surrogate policy $\phi^\star_{\hat{\mathcal{M}}_t}$ using reinforcement learning. In this paper, we utilize **deep deterministic policy gradient (DDPG)** Lillicrap et al. (2015) to handle a continuous action space. DDPG learns a deterministic policy with an actor-critic framework. Roughly speaking, it simultaneously learns an **actor network** $\mu(s)$ parametrized by θ^μ and a **critic network** $Q(s, a)$ parametrized by θ^Q. The critic network represents the current estimate of the action-value function, while the actor network tries to approximate $\max_a Q(s, a)$. We refer the reader to the original paper Lillicrap et al. (2015) for a more detailed discussion of this algorithm.

One major advantage of policy learning over planning is that the policy $\phi^\star_{\hat{\mathcal{M}}_t}$ can be used for multiple steps with minimal computational overhead, whereas the planning approach requires solving a nonlinear program at every step. To demonstrate the generalizability of the learned policy, in our experiments described later, we only allow the DDPG to train once at the beginning of the attack on the surrogate MDP $\hat{\mathcal{M}}_0$ based on the pre-attack data $\mathbf{z}_{-n:-1}$. The learned policy $\phi_{\hat{\mathcal{M}}_0}$ is then applied to all later attack rounds without retraining.

6. Regret Analysis

The fundamental statistical limit of a realistic attacker is its lack of knowledge on the environment data distribution P. An idealized attacker with knowledge of P can find the optimal control policy $\phi^\star_{\mathcal{M}}$. In contrast, a realistic attacker will find an optimal policy with respect to its estimated MDP $\hat{\mathcal{M}}$: $\phi^\star_{\hat{\mathcal{M}}} = \arg\min_\phi J_{\hat{\mathcal{M}}}(\phi)$, but $\phi^\star_{\hat{\mathcal{M}}}$ is in general suboptimal with respect to the true MDP $\mathcal{M}$. We are interested in the **regret**, defined as the optimality gap between the attack cost achieved by the realistic attacker and the cost of the optimal attack strategy in hindsight, i.e. $V_{\mathcal{M}}^{\phi^\star_{\hat{\mathcal{M}}}}(s) - V_{\mathcal{M}}^{\phi^\star_{\mathcal{M}}}(s)$.

In this section, we present a theoretical analysis relating the optimality gap to the quality of estimated $\hat{P}$. Our analysis is a natural extension to the Simulation Lemma in tabular reinforcement learning Kearns and Singh (2002) and that of Azar et al. (2017). We assume that both $\mathcal{Z}$ and Θ are compact, and the running cost g is continuous and thus bounded on its compact domain. Without loss of generality, we assume $g \in [0, C_{\max}]$.

Theorem 6.1 *Consider two attack MDPs $\mathcal{M}, \hat{\mathcal{M}}$ that differ only in state transition, induced by P and $\hat{P}$, respectively. Assume that $\|\hat{P} - P\|_1 := \int_{\mathcal{Z}} \left|\hat{P}(\mathbf{z}) - P(\mathbf{z})\right| \mathrm{d}\mathbf{z} \le \varepsilon$. Let $\phi^\star_{\mathcal{M}}$ denote the optimal policy on $\mathcal{M}$ and $\phi^\star_{\hat{\mathcal{M}}}$ the optimal policy on $\hat{\mathcal{M}}$. Then, $\sup_{s\in S} V_{\mathcal{M}}^{\phi^\star_{\hat{\mathcal{M}}}}(s) - V_{\mathcal{M}}^{\phi^\star_{\mathcal{M}}}(s) \le \frac{\gamma C_{\max}\varepsilon}{(1-\gamma)^2}$.*

Theorem 6.1 implies that optimality gap is at most linear in $\varepsilon := \|\hat{P} - P\|_1$. To relate to sample complexity, classic Results on Kernel Density Estimation (KDE) suggest that the L_1 distance between P and the kernel density estimator $\hat{P}_n$ based on n samples converges to zero asymptotically in a rate of $O(n^{-s/d+2s})$ for some constant s (e.g. Theorem 9 in Holmström and Klemelä (1992)).

In the experiment section below, the environment data stream is generated from a uniform distribution on a finite data set, in which case P is a multinomial distribution. Under this special setting, we are able to provide a finite-sample bound of order $O(n^{-1/2})$ that matches with the best achievable asymptotic rate above, i.e. as $s \to \infty$.

Corollary 6.2 *Consider an MDP $\mathcal{M}$ induced by a multinomial distribution P with support cardinality N, and a surrogate MDP $\hat{\mathcal{M}}$ induced by the empirical distribution $\hat{P}$ on n i.i.d. samples, i.e.*

$\hat{P}(i) = \frac{1}{n}\sum_{j=1}^{n} I_{x_j=i}$. *Then, with probability at least* $1-\delta$*, we have* $\sup_{s\in S} V_{\mathcal{M}}^{\phi^\star_{\hat{\mathcal{M}}}}(s) - V_{\mathcal{M}}^{\phi^\star_{\mathcal{M}}}(s) \le \frac{2\gamma C_{\max}}{(1-\gamma)^2}\sqrt{\frac{1}{2n}\ln\frac{2^{N+1}}{\delta}} = O(n^{-1/2})$.

7. Experiments

In this section, we empirically evaluate our attack algorithms NLP and DDPG against several baselines on synthetic and real datasets. As an empirical measure of attack efficacy, we compare the attack methods by their **empirical discounted cumulative cost** $\tilde{J}(t) := \sum_{\tau=0}^{t}\gamma^\tau g(\theta_\tau, \mathbf{z}_\tau, \mathbf{a}_\tau)$, where $\tilde{J}(t)$ is computed on an instantiation of the environment data stream $\mathbf{z}_0, \ldots, \mathbf{z}_t$. Better attack methods have smaller $\tilde{J}(t)$. We compare our algorithms with the following baseline Attackers:

Null Attack: This is the baseline without attack, namely $\mathbf{a}_t^{\text{Null}} = \mathbf{z}_t$ for all t. We expect the null attack to form an upper bound on any attack method's empirical discounted cumulative cost $\tilde{J}(t)$.

Greedy Attack: The greedy strategy has been applied to solve the sequential teaching problem (Liu et al. (2017); Lessard et al. (2018)). At time step t the greedy attacker uses a time-invariant attack policy which minimizes the current step's running cost g, i.e. $\mathbf{a}_t^{\text{Greedy}} = \arg\min_{\mathbf{a}} g(\theta_t, \mathbf{z}_t, \mathbf{a})$.

Clairvoyant Attack: A clairvoyant attacker is an idealized attacker who knows the evaluation horizon T and the whole data sequence $\mathbf{z}_{0:T-1}$ upfront. The clairvoyant attacker solves a finite time-horizon optimal control problem, equivalent to the problem solved by NLP, but on the actual data sequence. The clairvoyant attacker has strictly more information, and we expect it to form a lower bound on realistic attack methods in terms of $\tilde{J}(t)$.

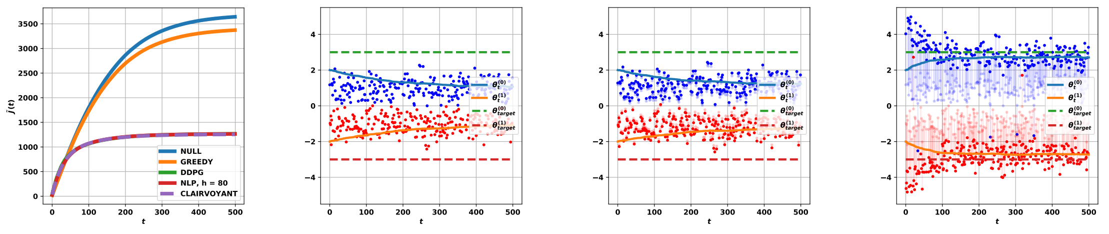
In our experiments, we evaluate all attacks on two victim learners: online logistic regression, a supervised learner, and online soft k-means clustering, an unsupervised learner.

Online logistic regression: Online logistic regression performs a binary classification task. The incoming data takes the form of $\mathbf{z}_t = (\mathbf{x}_t, y_t)$, where $\mathbf{x}_t \in \mathbb{R}^d$ is the feature vector and $y_t \in \{-1, 1\}$ is the binary label. In the experiments, we focus on attacking the feature part of the data, as is done in a number of prior works Mei and Zhu (2015); Koh and Liang (2017); Wang and Chaudhuri (2018). The learner's update rule is one step of gradient descent on the log likelihood with step size η: $f(\theta, (\mathbf{x}, y)) = \theta + \eta\frac{y\mathbf{x}}{1+\exp(y\theta^\top\mathbf{x})}$. The attacker wants to force the victim learner to stay close to a target parameter $\theta^\dagger$. The attacker's cost function is defined as $g(\theta_t, (\mathbf{x}_t, y), (\mathbf{x}'_t, y)) = \|\mathbf{x}'_t - \mathbf{x}_t\|^2 - \lambda\cos(\theta_t, \theta^\dagger)$.

Online soft k-means: Online soft k-means performs a k-means clustering task Bezdek et al. (1984). The incoming data contains only the feature vector, i.e. $\mathbf{z}_t = \mathbf{x}_t$. The learner's update rule is one step of soft k-means update with step size η on all centroids, i.e. $f(\theta^{(j)}, \mathbf{a}) = \theta^{(j)} + \eta r_j(\mathbf{a} - \theta^{(j)})$, $j = 1, \ldots, k$, where $\mathbf{r} = \text{softmax}\left(-\|\mathbf{a} - \theta^{(1)}\|^2, \ldots, -\|\mathbf{a} - \theta^{(k)}\|^2\right)$. Similar to online logistic regression, we consider a targeted attack objective with target centroid $\theta^{\dagger(j)}$. The attacker's cost function is defined as $g(\theta_t, \mathbf{z}_t, \mathbf{a}_t) = \|\mathbf{a}_t - \mathbf{z}_t\|^2 + \lambda\sum_{j=1}^{k}\|f(\theta, \mathbf{a})^{(j)} - \theta^{\dagger(j)}\|^2$.

7.1. Synthetic Data Experiments

We first show a synthetic data experiment where the attack policy can be visualized. The environment is a mixture of two 1D Gaussian: $P = \frac{1}{2}N(\theta^{(1)}, 1) + \frac{1}{2}N(\theta^{(2)}, 1)$ with $\theta^{(1)} = -1$ and $\theta^{(2)} = +1$. The victim learner is online soft k-means with $k = 2$ and initial parameter $\theta_0^{(1)} = -2, \theta_0^{(2)} = +2$. The attack target is $\theta^{\dagger(1)} = -3$ and $\theta^{\dagger(2)} = +3$, namely the opposite of how the victim's parameters should move. We set the learning rate $\eta = 0.01$, cost weight $\lambda = 10$, discounting factor $\gamma = 0.99$,



(a) Attack Costs (b) NULL Attack (c) GREEDY Attack (d) NLP Attack

Figure 2: Synthetic data experiments. In (b)-(d), transparent blue and red dots indicate clean positive and negative data point $\mathbf{z}_t$ at time step t, solid dots indicate attacker-perturbed data point $\mathbf{a}_t$, vertical lines in between indicate the amount of perturbation. DDPG overlaps with Clairvoyant in (a).

evaluation length $T = 500$ and look-ahead horizon for NLP $h = 100$. Following the above specification, we run each attack method on the same data stream and compare their behaviors.

Results: Figure 2a shows the empirical discounted cumulative cost $\tilde{J}(t)$ as the attacks go on. On this toy example, the null attack baseline achieves $\tilde{J}(T) = 3643$ at $T = 500$. The greedy attacker is only slight more effective at $\tilde{J}(T) = 3372$. NLP and DDPG achieve 1265 and 1267, respectively, almost matching Clairvoyant's 1256. As expected, the null and clairvoyant attacks form upper and lower bounds on $\tilde{J}(t)$. Figure 2b-d shows the victim's θ_t trajectory as attacks go on. Without attack (null), θ_t converges to the true parameter -1 and $+1$. The greedy attack only perturbs each data point slightly, failing to force θ_t toward attack targets. This failure is due to its greedy nature: the immediate cost g_t at each round is indeed minimized, but not enough to move the model parameters close to the target parameters. In contrast, NLP and DDPG (trajectory similar to NLP, not shown) exhibit a different strategy in the earlier rounds. They inject larger perturbations to the data points and sacrifice larger immediate costs in order to drive the victim's model parameters quickly towards the target. In later rounds they only need to stabilize the victim's parameters near the target with a smaller per-step cost.

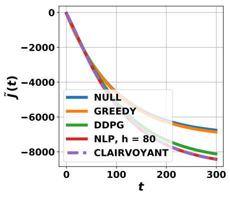
7.2. Real Data Experiments

In the real data experiments, we run each attack method on 10 data sets across two victim learners.

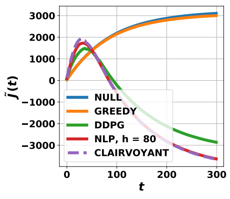
Datasets: We use 5 datasets for online logistic regression: Banknote Authentication (with feature dimension $d = 4$), Breast Cancer ($d = 9$), Cardiotocography ($d = 25$), Sonar ($d = 60$), and MNIST 1 vs. 7 ($d = 784$), and 5 datasets for online k-means clustering: User Knowledge ($d = 6, k = 2$), Breast Cancer ($d = 10, k = 2$), Seeds ($d = 8, k = 3$), posture ($d = 11, k = 5$), MNIST 1 vs. 7 ($d = 784, k = 2$). All datasets except for MNIST can be found in the UCI Repository Dua and Graff (2017). Note that Breast Cancer and MNIST are shared across both tasks.

Preprocessing: To reduce the running time, for datasets with dimensionality $d > 30$, we reduce the dimension to 30 via PCA projection. Then, all datasets are normalized so that each feature has mean 0 and variance 1. Each dataset is then turned into a data stream via uniform sampling with replacement.

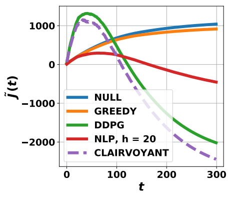
Experiment Setup: In order to demonstrate the robustness of our methods, we draw both the victim's initial model θ_0 and the attacker's target $\theta^\dagger$ randomly from a standard Gaussian distribution of the appropriate dimension. Across all datasets, we use the following parameters: $\eta = 0.01, \gamma = 0.99, T = 300$. For online logistic regression $\lambda = 100$ while for online k-means $\lambda = 10$.



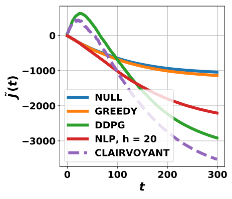
(a) Banknote



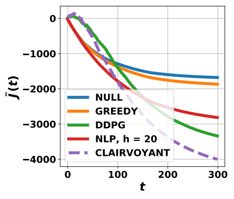
(b) Breast



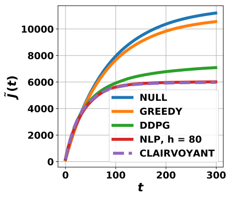
(c) CTG



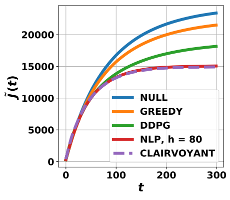
(d) Sonar



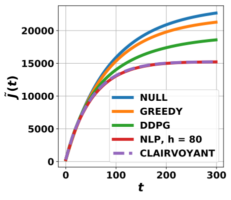
(e) MNIST 1 vs. 7



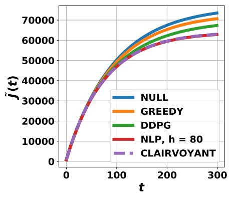
(f) Knowledge



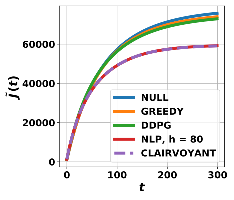
(g) Breast



(h) Seeds



(i) Posture



(j) MNIST 1 vs. 7

Figure 3: The empirical discounted cumulative reward $\tilde{J}(t)$ for the five attack methods across 10 real datasets. The first row is on online logistic regression and the second row is on online k-means.

For DDPG attacker we only perform policy learning at the beginning to obtain $\phi_{\hat{\mathcal{M}}_0}$, which is fixed and deployed. We give a pre-attack dataset $\mathbf{z}_{-n:-1}$ of size $n = 1000$ to DDPG and NLP. For NLP attack we set the look-ahead horizon h such that the total runtime to perform $T = 300$ attacks does not exceed the DDPG training time. This results in $h = 20$ for online logistic regression on CTG, Sonar and MNIST, and $h = 80$ in all other experiments.

Results: The experiment results are shown in figure 3. NLP achieves clairvoyant-matching performance on all 7 datasets in which it is given a large enough look-ahead horizon. DDPG follows closely next to NLP and Clairvoyant on most of the datasets, indicating that the pre-trained policy $\phi_{\hat{\mathcal{M}}_0}$ can achieve reasonable attack performance in most cases. On the 3 datasets where $h = 20$ for NLP, DDPG exceeds the short-sighted NLP, indicating that when the computational resource is limiting, DDPG has an advantage by avoiding the iterative retraining that NLP cannot bypass. GREEDY does not do well on any of the 10 datasets, achieving only a slightly lower cost than the NULL baseline. This matches our observations in the synthetic experiment.

DDPG vs. NLP: As we have seen in the experiments, the NLP method enjoys stability and high performance given enough computational resources, whereas the DDPG method has the advantage of being able to generalize and applied without retraining. Therefore, a realistic strategy is to use NLP when the problem is of mederate size, and the learner update is performed in a low frequency, e.g. once per day, and to use DDPG when the problem size is large and the learner update happens in a high frequency and does not permit retraining at every timestep.

8. Conclusion

In this paper, we formulated online poisoning attacks as an adaptive control problem. We proposed two attack algorithms, based on model-based planning and deep reinforcement learning, and showed that both are able to achieve near clairvoyant-level performance. We provided a regret analysis on the cost achieved by a realistic attacker and showed that despite the restricted knowledge, the optimality gap is upper-bounded by order of $O(n^{-1/2})$, given the resource of n data samples.

ACKNOWLEDGMENTS

This work is supported in part by NSF 1750162, 1837132, 1545481, 1704117, 1623605, 1561512, the MADLab AF Center of Excellence FA9550-18-1-0166, and the University of Wisconsin.

References

Scott Alfeld, Xiaojin Zhu, and Paul Barford. Data poisoning attacks against autoregressive models. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, pages 263–272, 2017.

James C Bezdek, Robert Ehrlich, and William Full. Fcm: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2-3):191–203, 1984.

Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012.

Francesco Borrelli, Alberto Bemporad, and Manfred Morari. *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

Cody Burkard and Brent Lagesse. Analysis of causative attacks against svms learning from data streams. In *Proceedings of the 3rd ACM on International Workshop on Security And Privacy Analytics*, pages 31–36. ACM, 2017.

Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

Yiding Chen and Xiaojin Zhu. Optimal adversarial attack on autoregressive models. *arXiv preprint arXiv:1902.00202*, 2019.

Dheeru Dua and Casey Graff. Uci machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Lasse Holmström and Jussi Klemelä. Asymptotic bounds for the expected l1 error of a multivariate kernel density estimator. *Journal of multivariate analysis*, 42(2):245–266, 1992.

Kwang-Sung Jun, Lihong Li, Yuzhe Ma, and Jerry Zhu. Adversarial attacks on stochastic bandits. In *Advances in Neural Information Processing Systems*, pages 3644–3653, 2018.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine learning*, 49(2-3):209–232, 2002.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1885–1894. JMLR. org, 2017.

Laurent Lessard, Xuezhou Zhang, and Xiaojin Zhu. An optimal control approach to sequential machine teaching. *arXiv preprint arXiv:1810.06175*, 2018.

Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. Data poisoning attacks on factorization-based collaborative filtering. In *Advances in neural information processing systems*, pages 1885–1893, 2016.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Weiyang Liu, Bo Dai, Ahmad Humayun, Charlene Tay, Chen Yu, Linda B Smith, James M Rehg, and Le Song. Iterative machine teaching. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2149–2158. JMLR. org, 2017.

Yuzhe Ma, Kwang-Sung Jun, Lihong Li, and Xiaojin Zhu. Data poisoning attacks in contextual bandits. In *International Conference on Decision and Game Theory for Security*, pages 186–204. Springer, 2018.

David Q Mayne, James B Rawlings, Christopher V Rao, and Pierre OM Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

Shike Mei and Xiaojin Zhu. Using machine teaching to identify optimal training-set attacks on machine learners. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

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*, pages 27–38. ACM, 2017.

Andrew Newell, Rahul Potharaju, Luojie Xiang, and Cristina Nita-Rotaru. On the practicality of integrity attacks on document-level sentiment analysis. In *Proceedings of the 2014 Workshop on Artificial Intelligent and Security Workshop*, pages 83–93. ACM, 2014.

Ayon Sen, Scott Alfeld, Xuezhou Zhang, Ara Vartanian, Yuzhe Ma, and Xiaojin Zhu. Training set camouflage. In *International Conference on Decision and Game Theory for Security*, pages 59–79. Springer, 2018.

Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.

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

Huang Xiao, Battista Biggio, Blaine Nelson, Han Xiao, Claudia Eckert, and Fabio Roli. Support vector machines under adversarial label contamination. *Neurocomputing*, 160:53–62, 2015.