Online Data Poisoning Attack
via Stochastic Optimal Control and Reinforcement Learning

Xuezhou Zhang and Jerry Zhu
University of Wisconsin-Madison

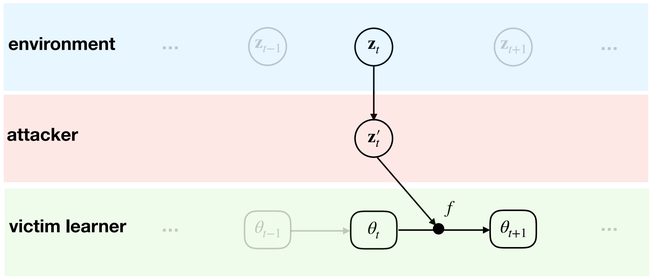
Introduction

We study the problem of data poisoning attack against **online learning systems**. **Data poisoning attack** studies methods to control a machine learning system by contaminating its training data. Different from prior work in the off-line setting, the main challenge in the online setting is that the future training data is **stochastic** and **unknown**, and the attacker has to perform attacks under uncertainty.

Problem Definition:

At time step t,

- **Environment**: Generates a data point $\boldsymbol{z}_t$ from the underlying data distribution P.
- **Online Learning**: Performs learning update: $\theta_{t+1} = f(\theta_t, \mathbf{z}_t)$.
- **Attacker**: Sits in between the environment and the online learning system, and can perturb $\boldsymbol{z}_t$ to $\boldsymbol{z}'_t$.



- **Attacking Objective:** (a) Manipulate the learned model, while (b) stay undetected. It can be captured by the weighted sum of an **attack loss** function and a **perturbation cost** function:

$$g_t(\theta_t, \mathbf{z}_t, \mathbf{z}'_t) = \lambda l_t(\theta_t) + c_t(\mathbf{z}_t, \mathbf{z}'_t).$$

Examples: ***targeted** attacks* $l_t(\theta_t) := \|\theta_t - \theta_{target}\|$
***aversion** attacks* $l_t(\theta_t) := -\|\theta_t - \theta_{clean}\|$
***backdoor** attacks* $l_t(\theta_t) := \ell(\theta_t, \mathbf{z}^*)$

Formulate As Stochastic Optimal Control:

The attacker's optimal attack problem is characterized as a stochastic optimal control problem, namely finding a **control policy** that minimizes the **expected discounted cumulative loss**:

$$\begin{aligned} \min_{\phi \in \Phi} \quad & \mathbb{E}_P \left[\sum_{t=0}^{\infty} \gamma^t g_t(\theta_t, \mathbf{z}_t, \phi(\theta_t, \mathbf{z}_t)) \right] \\ \text{s.t.} \quad & \theta_{t+1} = f(\theta_t, \phi(\theta_t, \mathbf{z}_t)), t \geq 0 \\ & \theta_0 \text{ given.} \end{aligned}$$

Challenge: The underlying distribution P is unknown. The only knowledge about P is the historical data points $D_t = \{\mathbf{z}_0, ..., \mathbf{z}_t\}$ generated from P.

Methods

Near-Optimal Attacks via Model Predictive Control (MPC):

At every time step t, the attacker solves for the **surrogate attack problem:**

$$\begin{aligned} \phi_t = \arg\min_{\phi \in \Phi} \quad & \mathbb{E}_{D_t} \left[\sum_{\tau=t}^{\infty} \gamma^{\tau-t} g_\tau(\theta_\tau, \mathbf{z}_\tau, \phi(\theta_\tau, \mathbf{z}_\tau)) \right] \\ \text{s.t.} \quad & \theta_{\tau+1} = f(\theta_\tau, \phi(\theta_\tau, \mathbf{z}_\tau)), \tau \geq t \\ & \theta_t \text{ given.} \end{aligned}$$

based on the current knowledge D_t, and then perform **one step attack:**

$$\mathbf{z}'_t = \phi_t(\theta_t, \mathbf{z}_t).$$

Such repeated procedure of (re)-planning but only executing the immediate action is characteristic of **Model Predictive Control** (MPC).

Solve as Nonlinear Programming (NLP):

Further approximate the objective by (a) introducing a **finite time horizon** h and (b) replacing the expectation with **random instantiation** $\mathbf{z}_{\tau:t+h-1} \sim D_t$.

$$\begin{aligned} \min_{\mathbf{z}'_{t:t+h-1}} \quad & \sum_{\tau=t}^{t+h-1} \gamma^{\tau-t} g_\tau(\theta_\tau, \mathbf{z}_\tau, \mathbf{z}'_\tau) \\ \text{s.t.} \quad & \theta_{\tau+1} = f(\theta_\tau, \mathbf{z}'_\tau), \forall \tau = t, ..., t+h-1 \\ & \theta_t \text{ given, } \mathbf{z}_{t:t+h-1} \sim D_t \text{ and fixed.} \end{aligned}$$

It can then be solved using modern NLP solver such as IPOPT[1].

Solve using Deep Deterministic Policy Gradient (DDPG):

One can directly solve for the optimal policy ϕ_t using policy learning. In our problem, the action space is **continuous**. Therefore, we use the DDPG[2] method which learns a deterministic policy over continuous action space. Roughly, it simultaneously learns an **actor network** $\mu(s)$ parametrized by θ^μ and a **critic network** $Q(s,a)$ parametrized by θ^Q. The actor network represents the currently learned policy while the critic network estimates the action-value function of the current policy, whose functional gradient guides the actor network to improve its policy. The policy gradient can be written as:

$$\nabla_{\theta^\mu} \mu = \mathbb{E}_\mu [\nabla_a Q(s, a|\theta^Q) \nabla_{\theta^\mu} \mu(s|\theta^\mu)]$$

Advantages of DDPG vs. NLP:

1. DDPG actually learns a policy ϕ_t that can **generalize** and be applied to more than one future steps of attack.
2. DDPG is a **model-free** method. It doesn't require the analytical form of the learner's update rule f. Therefore, it also applies to the **black-box** attack setting, where the exact learning rule is unknown to the attacker.

Experiments

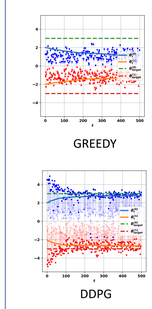
Baselines:

- **Null Attack**: baseline without attack, namely $\boldsymbol{z}'_t = \boldsymbol{z}_t$.
- **Greedy Attack**: $\mathbf{z}_t^{greedy} = \arg\min_{\mathbf{z}} g_t(\theta_t, \mathbf{z}_t, \mathbf{z})$.
- **Clairvoyant Attack**: An idealized attacker who knows the exact **past, present, and future** data stream, and solves for the deterministic optimal attack problem as a nonlinear program.

Victim Learners:

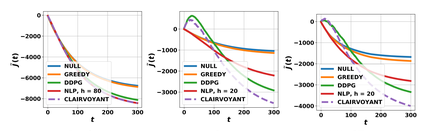
- **Online Logistic Regression:** The learner's update rule is one step of gradient descent on the log likelihood with step size η.
- **Online Soft k-means:** The learner updates all the centroids weighted by their distances to the current data point using the softmax function.

Selected Experiment Results:



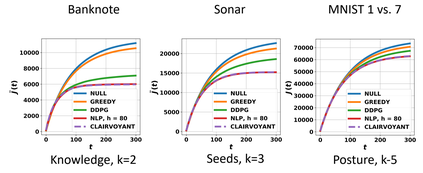
Synthetic Experiment

Online LR:



Banknote Sonar MNIST 1 vs. 7

Online KM:



Knowledge, k=2 Seeds, k=3 Posture, k-5

Real Data Experiment

Conclusions

Takeaway messages:

- **Optimal control-based methods** NLP and DDPG achieve **significantly better** performance than heuristic methods such as GREEDY, and in some cases they even achieve clairvoyant-level performance.
- In the case that the learner's dynamics f is **known** to the attacker and is **differentiable**, and that the induced nonlinear program can be solved efficiently, NLP is a strong attack method.
- DDPG, on the other hand, is able to learn a **reasonable attack policy** given enough data. The attack policy can be fixed and deployed, which is advantageous when the **data stream comes in quickly** and leaves no time to re-do planning in MPC/NLP.

Possible direction of future work:

- How to perform attacks if the data generating distribution P is not fixed?
- How to tackle high dimensional tasks? Action space dim = 784 for MNIST.
- How should the victim defend against such attacks?

Contact

Xuezhou Zhang
University of Wisconsin, Madison
zhangxz1123@cs.wisc.edu

References

1. 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.
2. 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.

© Poster Template by Genigraphics® 1.800.790.4001 www.genigraphics.com