A Flexible Poisoning Attack Against Machine Learning

Wenbo Jiang[†] [S], Hongwei Li[†] [§](corresponding author), Sen Liu[†], Yanzhi Ren[†], Miao He[*]
[†]School of Computer Science and Engineering, University of Electronic Science and Technology of China, China
[S]CETC Big Data Research Institute Co., Ltd., Guiyang 550022, China
[§]Science and Technology on Communication Security Laboratory, Chengdu 610041, China
[*]Fortinet Technologies (Canada) ULC, Ottawa, ON, Canada

***Abstract*—Recent years have witnessed tremendous academic efforts and industry growth in machine learning. The security of machine learning has become increasingly prominent. Poisoning attack is one of the most relevant security threats to machine learning which focuses on polluting the training data that machine learning needs during the training process. Specifically, the attacker blends crafted poisoning samples into training data in order to make the learned model beneficial to him. To the best of our knowledge, existing researches about poisoning attack focused on either integrity attack or availability attack, which did not unify these two attacks together. Aside from that, from the attacker's perspective, attacker's strategy is not flexible enough. Finally, existing proposals only concentrated on increasing the test error of the learned model but ignored the importance of the concealment of attack. To overcome these issues, we firstly present a thorough adversarial model for poisoning attack in which attacker's strategy is defined from two aspects, i.e., the effect of attack and the concealment of attack. Then we unify integrity attack and availability attack together in similar formulations. Furthermore, in order to enhance flexibility, a tradeoff parameter is inserted into attacker's objective function which means the attacker can balance the attraction of effect against the requirement of concealment. Finally, as examples, extensive experiments are conducted on linear regression and logistic regression to demonstrate the effectiveness of attack.**

***Index Terms*—Poisoning attack, Machine learning.**

I. Introduction

In recent years, machine learning has received widely attention and become a vital tool to address the so-called big-data problems. In reality, it has shown promising application prospects in many fields, such as spam detection [1] and image recognition [2]. However, while machine learning brings great convenience to people, it faces numerous security threats as well.

According to [3], attacks on machine learning can be classified from three different perspectives. As we can see from Fig. 1, in terms of the attacks' impact on the classifier, they can be categorized into exploratory attack (exploitation of the classifier) and causative attack (affecting the training dataset). Based on the security violation caused by attacks, they fall into three categories: integrity attack, availability attack and privacy attack. According to the purpose of attacks, they can be divided into targeted attack and indiscriminate attack.

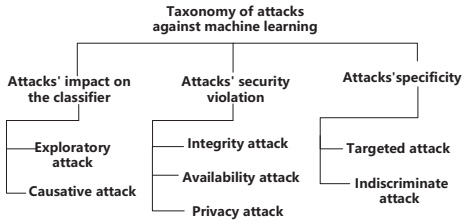


Fig. 1. Taxonomy of attacks against machine learning

Among the different attacks against machine learning, poisoning attack (or causative attack) is considered one of the most relevant security threats and has been intensively investigated [4]–[17]. It is commonly known that collecting data phase is an essential process of machine learning because the collected training data can directly influence the classification and prediction models. Yet the security of this process is frequently overlooked, which gives attackers a chance to pollute the training data to mislead the model. Poisoning attack is such an attack that the attacker manipulates a fraction of the training data. The architecture of poisoning attack is illustrated in the following Fig. 2.

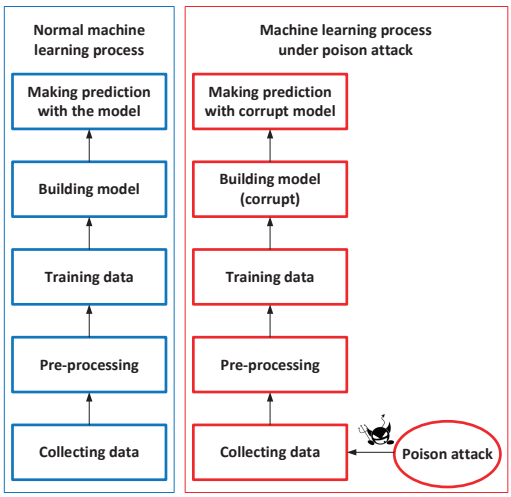


Fig. 2. Architecture of poisoning attack

In practice, the original training data of the machine learn-

978-1-5386-8088-9/19/$31.00 ©2019 IEEE

ing is mostly confidential, it is generally impossible for an attacker to manipulate it. However, many machine learning systems need additional training data (maybe stored in the cloud [18] [19] [20]) to update the model in order to enhance the adaptability, thus giving the attacker a handel for exploiting. Sophisticated attackers may inject some carefully forged malicious data samples (with wrong labels) and destroy the probability distribution of the original training data, so as to reduce the classification or clustering precision of the learned model. Such attack has been practically demonstrated in a number of applications such as handwritten digit recognition [5] and PDF malware detection [6].

Although considerable research efforts have been devoted to poisoning attack [4]–[17] and some encouraging progress has been made, it is less successful and still requires further research. Firstly, existing researches about poisoning attack did not unify integrity attack and availability attack together. Besides, attacker's strategy lacked flexibility because of a fixed objective function. Finally, existing proposals only concentrated on increasing the test error of the learned model, it may be more reasonable if they had considered the concealment of attack.

In this paper, we propose a detailed adversarial model for poisoning attacks and define the attacker's strategy from two aspects, i.e., the effect of attack and the concealment of attack. Our contributions can be summarized in four aspects as follows:

- We propose a thorough adversarial model for poisoning attacks and unify integrity attack and availability attack together by employing similar formulations.
- As for attacker's strategy, in order to consider the problem all sidedly, both the effect of attack and the concealment of attack are taken into consideration. Specifically, we use the offset of the learned model to characterize the effect of attack and use the distance between the poisoning point and the center point of training data to characterize the concealment of attack. In addition, we utilize the 3σ-Principle to restrict the variation of the poisoning point to evade outlier detection.
- Furthermore, we define attacker's objective function with a tradeoff parameter which can help the attacker balance the attraction of effect against the requirement of concealment.
- We choose linear regression and logistic regression as exemplary machine learning models to perform our attack. Experiments are conducted to deliver that our attack is very effective for these two machine learning models.

The remainder of this paper is organized as follows. We present the related work about poison attack in section II. Then we carry out the details of adversarial model and attack methodology in section III and IV, respectively. Experiments will be given in section V. Finally, section VI concludes the paper.

II. Related Work

The earliest study on poisoning attacks was developed by Rubinstein et al. [4], who only investigated simple anomaly detection method. Biggio et al. [5] drew attention to poisoning attacks against Support Vector Machines (SVMs) and explored a family of poisoning attacks, where the attacker is allowed to progressively inject malicious points to the training data in order to decrease the SVM's classification accuracy. The attacker uses a gradient ascent strategy to generate malicious points in which the gradient is calculated based on properties of the SVM's optimal solution. Xiao et al. [6] paid attention to feature selection algorithms. They proposed a framework to categorize and provided a better understanding of the different attacks performed on feature selection algorithms. In addition, currently widely used algorithms Latent Dirichlet Allocation (LDA) [7], autoregressive models [8] and Principal Component Analysis (PCA) [9] are also facing the threat of poisoning attack. Mei et al. [10] focused on realizing an algorithmic framework for identifying the optimal training set attack leveraging on the idea of *machine teaching*. Specifically, machine teaching [11] is the inverse problem to machine learning, in which the learning algorithm L, target model θ^* are known, one needs to find the smallest training set D such that $L(D) = \theta^*$. In their attack, the teacher plays the role of the attacker to maximally mislead a learner by carefully generating the poisoning samples. [12] explored the poisoning attack on multiclass problems which employed the idea of back-gradient optimization to optimize the poisoning samples. Zhao et al. [13] presented an attack against a substitute model and transferred it to a black-box victim model. Then, it utilized a Projected Gradient Ascent (PGA) algorithm to solve the optimization program. Jagielski et al. [14] focused on poisoning attack against regression learning which includes Least Absolute Shrinkage and Selection Operator (LASSO), Ordinary Least Squares (OLS), ridge regression and elastic-net regression.

As for clustering algorithms, although the label cannot be modified, the literature [15]–[17] introduced poisoning attack for single-linkage hierarchical clustering and complete-linkage hierarchical clustering. A heuristic strategy is adopted to measure poisoning sample's impact on the accuracy of classification, so as to generate better poisoning samples to obtain lower clustering accuracy.

III. Adversarial Model

In order to perform an optimal poisoning attack, it is necessary to disclose one's assumptions of the adversarial model. In this section, we systematically summarize the work previously proposed in [3], [6], [14], [21]–[24] and present our adversarial model for poisoning attack.

A. Attacker's Goal

In poisoning attack, attacker's goal is to affect the learning model during the training process. According to the expected degree of security damage caused by the attacker, the attacker's goals can be roughly classified into three categories.

- Availability attack. The attacker's goal is to affect the learning model indiscriminately which makes the functionality of the system unavailable. For example, an attack is called availability attack if the attacker generates a non-targeted attack to maximize the classifier's error rate or causes a denial of service.
- Integrity attack. The attack is considered an integrity attack, if the attacker has an arbitrary objective, whose goal is to cause specific error on the learning model. For instance, the attacker makes the learning model's parameter values close to the value expected by him or makes the model output mis-predictions for certain test samples.
- Privacy attack. The training process of machine learning now requires a lot of data. A lot of data is collected through crowdsourcing technology, which contains a lot of user privacy information (such as photos and voice information), and even sensitive information (such as medical data). The attacker's goal is to impinge upon user's privacy during these processes, such attacks fall within the realm of privacy attack.

B. Attacker's Knowledge

The attacker can have different levels of knowledge of the targeted learning model. Generally, the adversary can have information about the targeted learning model in four components:

1) The training data D_{tr}: The attacker knows the original training data before performing attack.
2) The feature values: The attacker knows how features are computed for each sample before feature selection.
3) The learning algorithm: The attacker knows the learning algorithm along with the objective function L minimized during training.
4) The trained parameters $\boldsymbol{\theta}$ (the math symbol in the bold letter means it is a vector): The attacker knows the parameters of the learned model $\boldsymbol{\theta}$.

Typically, there are two main assumptions about knowledge of adversary, referred to as perfect and limited knowledge. Moreover, depending on the different levels of knowledge of adversary, one can envisage different attack scenarios.

According to *Kerckhoffs' Principle* [25], a system should be secure even if everything about the system is public knowledge. Ideally, a secure system should make minimal assumptions about what can be kept secret from an adversary. Thus, in this paper, we assume that the attacker has full knowledge of the attacked system (usually referred to as perfect knowledge case).

C. Attacker's Capability

Capability of the attacker mainly refers to the attacker's ability to control the training data. It can be defined from the following aspects:

- In what ways and to what extent does the attacker controls the training data? This is the first aspect of attacker's capability. For example, modifying the original training data or injecting some carefully generated malicious data samples. In most cases, since the original training data of the machine learning algorithm system is mostly confidential, it is generally not easy for an attacker to modify the original training data (unless the entire system is compromised). However, in order to enhance the adaptability, many machine learning systems need new training data to update models, which give attacker a chance to inject the poisoning data the into the training data.
- What is the largest proportion of poisoning points to the overall training data? This is another aspect of attacker's capability. Actually, attacker is typically assumed to be able to inject a small fraction of poisoning points into the training data.

D. Attacker's Strategy

In this paper we assume the goal of the attacker is to destroy the integrity and availability of the system, the attacker has full knowledge of the learning algorithm.

In terms of integrity attack, the attacker may have a specific target model $\boldsymbol{\theta_t}$ in mind. His goal is to make the learned model $\boldsymbol{\theta_p}$ (contains poisoning points) to be close to $\boldsymbol{\theta_t}$. We define a function E_I=$||\boldsymbol{\theta_p} - \boldsymbol{\theta_t}||_2^2$ to evaluate the effect of attack. The smaller the value of E_I, the better the effect of attack.

Similarly, we define E_A to evaluate the effect of availability attack in which the attacker don't have a target model in mind. It can be denoted as E_A=$||\boldsymbol{\theta_p} - \boldsymbol{\theta_{D_{tr}}}||_2^2$ ($\boldsymbol{\theta_{D_{tr}}}$ means the parameter learned by original training data D_{tr}) and the greater the value of E_A, the better the effect.

Meanwhile, the attack sample $\boldsymbol{x_p}$ may subject to some feasibility constraints, for instance, some of its features have a specific reasonable range, it is obviously unreasonable that the age of a customer is -20 or 200 years old. We assume that the training data follows a Gaussian distribution. According to 3σ-Principle, the probability that a value is 3σ greater than the average value μ or 3σ less than μ is $P(|x-\mu| > 3\sigma) <= 0.003$, which is a very small probability event. So we confine the value of $\boldsymbol{x_p}$ to a bound (3σ) on its distance from center point of training data $\boldsymbol{x_c}$. In fact, if the poison sample is too far away from other samples, it may be cleaned up in the process of outlier detection.

In addition, the attacker may prefer a concealed attack in order to evade detection. So we suggest a function C to indicate the concealment of the poisoning point which can be defined as C=$||\boldsymbol{x_p} - \boldsymbol{x_c}||_2^2$ and the smaller the value of C, the better the concealment.

Altogether, we use W to represent the attacker's overall objective function and his strategy can be formulated as:

- Integrity attack:

$$arg\,min_{\boldsymbol{x_p}} \qquad W_I = \alpha C(\boldsymbol{x_p}) + (1-\alpha) E_I(\boldsymbol{\theta_p}) \quad (1)$$

$$s.t. \qquad \boldsymbol{\theta_p} \in arg\,min_{\boldsymbol{\theta}}\, L(D_{tr} \cup \boldsymbol{x_p}, \boldsymbol{\theta}) \quad (2)$$

- Availability attack:

$$arg\,min_{\boldsymbol{x_p}} \quad W_A = \alpha C(\boldsymbol{x_p}) - (1-\alpha)E_A(\boldsymbol{\theta_p}) \quad (3)$$

$$s.t. \quad \boldsymbol{\theta_p} \in arg\,min_{\boldsymbol{\theta}}\, L(D_{tr} \cup \boldsymbol{x_p}, \boldsymbol{\theta}) \quad (4)$$

It should be pointed out that the problems above are both bilevel optimization problems [26]. The optimization over $\boldsymbol{x_p}$ in equation (1) and (3) are called the upper-level problems while the optimization over $\boldsymbol{\theta_p}$ in equation (2) and (4) are called the lower-level problems. L in equation (2) and (4) is the learning algorithm's objective function minimized during training.

It is important to highlight that we insert a tradeoff parameter α into attacker's strategy which gives the attacker much more flexibility. $\alpha = 0.5$ means the attacker considers both the concealment and effect of attack and $\alpha = 0$ indicates the attacker only wants to perform an effective attack, regardless of the concealment.

IV. Attack Methodology

In this section, we will first present an exemplary poisoning attack algorithm employing gradient-descent algorithm and then we will discuss how to compute the required gradient.

A. Poisoning Attack Algorithm

As for integrity attack (gradient in availability attack can be performed in the same way), an exemplary poisoning attack algorithm leveraging on gradient descent algorithm will be given in algorithm 1. It is important to highlight that in order to optimize the attack with multiple poisoning points (precisely, m poisoning points), our strategy is to iteratively optimize one poisoning sample at a time.

Algorithm 1 Poisoning Attack Algorithm

Input: original training data $D = D_{tr}$; m initial poisoning points $D_m^{(0)} = \{\boldsymbol{x_p}^{(0)}, y_p\}_{c=1}^{m}$; step size s; learning algorithm's objective function L; tradeoff parameter α; a small positive constant ε.

Output: optimal m poisoning points $D_m = \{\boldsymbol{x_p}, y_p\}_{c=1}^{m}$

```
for each q ∈ [1, m] do
    i ← 0 (iteration counter)
    repeat
        θ^(i) ← arg min_θ L(D ∪ {x_p^(i)}_q, θ)
        W_I^(i) ← W_I({x_p^(i)}_q, θ^(i))
        {x_p^(i+1)}_q ← {x_p^(i)}_q − s∇_{x_p} W_I^(i)
        i ← i + 1
    until |W_I^(i+1) − W_I^(i)| < ε
    {x_p}_q ← {x_p^(i)}_q
    D ← D ∪ {x_p}_q
end for
return D_m = {x_p, y_p}_{c=1}^m
```

$\boldsymbol{x_p}$ refers to the feature vector of the poisoning point and y_p is its response variable. In addition, the domain of $\boldsymbol{x_p}$ is limited with a bound (3σ) on its distance from center point of training data $\boldsymbol{x_c}$ (as we discussed in section II).

As for initialization strategy of poisoning point, according to previous work on poisoning attack [5], [13], [6], a simple initialization strategy for poisoning attack on classification problems is to randomly clone a subset of the original training data and flip their labels. Similarly, a simple initialization approach for poisoning attack on regression problems is to randomly clone a subset of the training data and keep the response variable unchanged.

B. Gradient Computation

The algorithm mentioned above is a standard gradient-descent algorithm, the most troublesome problem is how to calculate the required gradient $\nabla_{\boldsymbol{x_p}} W_I(\boldsymbol{x_p}, \boldsymbol{\theta})$.

It can be simplified as:

$$\begin{aligned}\nabla_{\boldsymbol{x_p}} W_I(\boldsymbol{x_p}, \boldsymbol{\theta}) &= \alpha \nabla_{\boldsymbol{x_p}} C(\boldsymbol{x_p}) + (1-\alpha)\nabla_{\boldsymbol{x_p}} E_I(\boldsymbol{\theta}) \\ &= \alpha(\boldsymbol{x_p} - \boldsymbol{x_c}) + (1-\alpha)\nabla_{\boldsymbol{x_p}} E_I(\boldsymbol{\theta})\end{aligned}$$

after that, we assume that E does not depend directly on $\boldsymbol{x_p}$, but only through $\boldsymbol{\theta}$ and then calculate $\nabla_{\boldsymbol{x_p}} E(\boldsymbol{\theta})$ employing the chain rule as:

$$\nabla_{\boldsymbol{x_p}} E_I(\boldsymbol{\theta}) = \nabla_{\boldsymbol{x_p}} \boldsymbol{\theta}(\boldsymbol{x_p})^\top \cdot \nabla_{\boldsymbol{\theta}} E_I \quad (5)$$

$\nabla_{\boldsymbol{\theta}} E_I$ is often easy to compute, so we concentrate on the computation of $\nabla_{\boldsymbol{x_p}} \boldsymbol{\theta}(\boldsymbol{x_p})^\top$. To overcome this problem, as in [14], [24] and [10], we leverage on Karush-Kuhn-Tucker (KKT) conditions to replace the inner learning problem with its equilibrium conditions, i.e., $\nabla_{\boldsymbol{\theta}} L(D_{tr} \cup \boldsymbol{x_p}, \boldsymbol{\theta}) = 0$ and the original bilevel problem can be reduced to a single-level constrained optimization problem. Then, by assuming that aforementioned equation is differentiable w.r.t. $\boldsymbol{x_p}$, we obtain:

$$\nabla_{\boldsymbol{x_p}} (\nabla_{\boldsymbol{\theta}} L(D_{tr} \cup \boldsymbol{x_p}, \boldsymbol{\theta})) = 0 \quad (6)$$

Apparently, the function L depends explicitly on $\boldsymbol{x_p}$ in its first argument, and implicitly through the parameters $\boldsymbol{\theta}$. Thus, equation (6) can be computed as:

$$\nabla_{\boldsymbol{x_p}} \nabla_{\boldsymbol{\theta}} L + \nabla_{\boldsymbol{x_p}} \boldsymbol{\theta}^\top \cdot \nabla^2_{\boldsymbol{\theta}} L = 0 \quad (7)$$

After that, solving for $\nabla_{\boldsymbol{x_p}} \boldsymbol{\theta}^\top$, one yields:

$$\nabla_{\boldsymbol{x_p}} \boldsymbol{\theta}^\top = \begin{pmatrix} \frac{\partial \boldsymbol{\omega}}{\partial \boldsymbol{x_p}}^\top & \frac{\partial b}{\partial \boldsymbol{x_p}}^\top \end{pmatrix} = -\nabla_{\boldsymbol{x_p}} \nabla_{\boldsymbol{\theta}} L (\nabla^2_{\boldsymbol{\theta}} L)^{-1} \quad (8)$$

and according to previous work in [6], [14]:

$$\nabla_{\boldsymbol{x_p}} \boldsymbol{\theta}^\top = -\frac{1}{n} \begin{pmatrix} \boldsymbol{M} & \boldsymbol{\omega} \end{pmatrix} \begin{pmatrix} \boldsymbol{\Sigma} + \lambda \boldsymbol{g} & \boldsymbol{\mu} \\ \boldsymbol{\mu}^\top & 1 \end{pmatrix}^{-1} \quad (9)$$

where $\Sigma = \frac{1}{n}\sum_i \boldsymbol{x_i}\boldsymbol{x_i}^\top$, λ is the regularization parameter, $\mu = \frac{1}{n}\sum_i \boldsymbol{x_i}$ and $\boldsymbol{M} = \boldsymbol{x_p}\boldsymbol{\omega}^\top + (f(\boldsymbol{x_p}) - y_p)\mathrm{II}_d$. g equals zero for OLS and LASSO, the identity matrix II_d for ridge regression, and $(1-\rho)\mathrm{II}_d$ for the elastic net.

So the derivatives $\nabla_{\boldsymbol{x_p}} \boldsymbol{\theta}^\top$ can be finally computed by equation (9) and then substituted into equation (5) to obtain the final gradient.

V. EXPERIMENTS

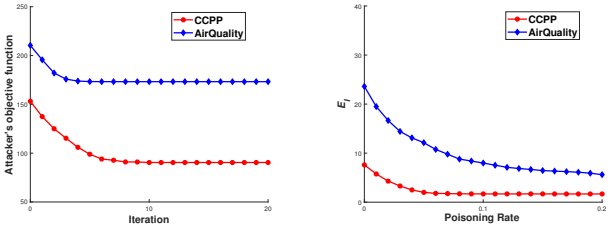
In this section, we implement our attack in Python and run our experiments on 16 core Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz 16G machine. As for regression problem, we choose linear regression as an exemplary learning model and perform integrity attack on it. In terms of classification problem, we choose logistic regression as an example. The target models $\boldsymbol{\theta_t}$ in these integrity attacks are set appropriately and the step size s of the gradient in algorithm 1 is set to $s = 1/i$.

It is worthwhile mentioning that availability attack can be conducted in the same way and our attack can be implemented to other machine learning models using the same methodology presented in this work.

A. Attack linear regression

We choose two different data sets to serve as the original training data D_{tr}, which come from machine learning data published by University of California Irvine. One data set is collected from a Combined Cycle Power Plant (CCPP)[1] over 6 years which contains 9568 points. Another data set[2] is the hourly averaged responses information about Air Quality collected from an array of 5 metal oxide chemical sensors embedded in an Air Quality Chemical Multisensor Device which contains 9358 instances. Then we use the hold-out method to split the original data into 70% for training, 30% for testing. From the perspective of the attacker, we present two strategies by setting α to different values.

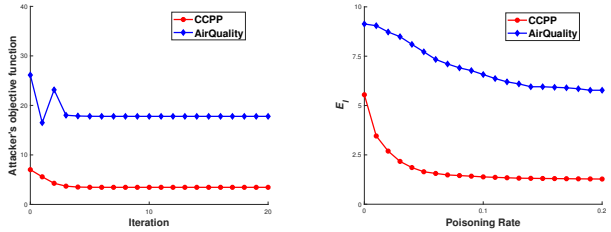
- An attacker who balances the concealment and the effect of attack ($\alpha = 0.5$): in such a scenario, the attacker wants to make the learned model beneficial to him as much as possible with as much concealment as possible. So we set $\alpha = 0.5$ to balance the attraction of effect against the requirement of concealment. According to the attack methodology described in section IV, we firstly generate one poisoning point and iteratively optimize it using gradient-descent algorithm. Fig 3(a) describes the value of attacker's objective function W with the increase of iteration and we can obviously observe a decrease in W with the increase of iteration. After that, we inject the poisoning data sample to the original data and carry out the above processes repeatedly. Fig 3(b) shows the value of E_I=$||\boldsymbol{\theta_p} - \boldsymbol{\theta_t}||_2^2$ with the increase of poisoning rate (the proportion of poisoning points). The smaller the value of E_I, the closer the learned model $\boldsymbol{\theta_p}$ is to the targeted model $\boldsymbol{\theta_t}$, the better the effect of attack. From the results we have obtained, we can reach a conclusion that our poisoning attack is effective because there is a distinct decline in E_I with the increase of poisoning rate.



(a) Single poisoning point (b) Multiple poisoning points

Fig. 3. Poisoning attack on linear regression when $\alpha = 0.5$

- An attacker who only pursuits the effect of attack ($\alpha = 0$): in this case, the attacker concentrates on the effect of attack, regardless of its concealment. So we set $\alpha = 0$ and attacker's objective function reduces to the effect of attack E_I. Even though, the value of $\boldsymbol{x_p}$ is confined to a bound (3σ) on its distance from center point of training data $\boldsymbol{x_c}$ in order to evade outlier detection. Similarly, we iteratively optimize one poisoning point at a time and produce poisoning point one by one. Fig 4(a) shows the value of attacker's objective function W with the increase of iteration and Fig 4(b) describes the value of E_I with the increase of poisoning rate.



(a) Single poisoning point (b) Multiple poisoning points

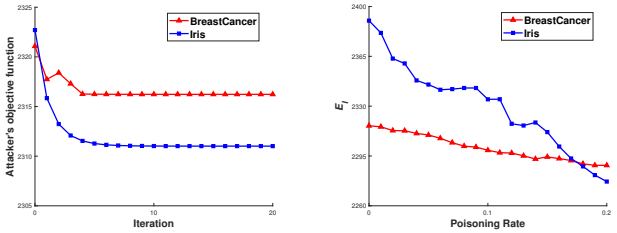
Fig. 4. Poisoning attack on linear regression when $\alpha = 0$

B. Attack logistic regression

The two original training data D_{tr} of logistic regression also come from machine learning data published by University of California Irvine. One is the Iris data set[3] and another is the breast cancer data set[4]. Similarly, we use hold-out method to split the original data into 70% for training, 30% for testing and two strategies are presented:

- An attacker who balances the concealment and the effect of attack ($\alpha = 0.5$): Fig 5(a) describes the value of attacker's objective function W with the increase of iteration and Fig 5(b) shows the value of E_I with the increase of poisoning rate.

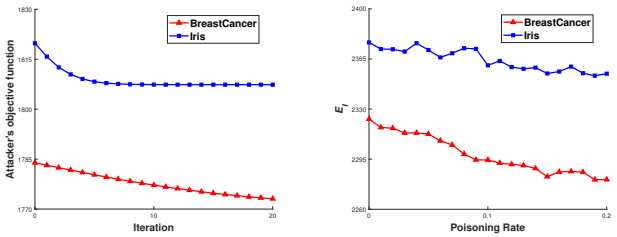
[1]http://archive.ics.uci.edu/ml/datasets/Combined+Cycle+Power+Plant
[2]http://archive.ics.uci.edu/ml/datasets/Air+Quality
[3]http://archive.ics.uci.edu/ml/datasets/Iris
[4]http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/



(a) Single poisoning point (b) Multiple poisoning points

Fig. 5. Poisoning attack on logistic regression when $\alpha = 0.5$

- An attacker who only pursuits the effect of attack ($\alpha = 0$): Fig 6(a) shows the value of attacker's objective function W with the increase of iteration and Fig 6(b) describes the value of E_I with the increase of poisoning rate.



(a) Single poisoning point (b) Multiple poisoning points

Fig. 6. Poisoning attack on logistic regression when $\alpha = 0$

VI. Conclusions

In this paper, we present a thorough adversarial model for poisoning attack, in which we unify integrity attack and availability attack together in similar formulations. After that, we take the concealment of attack into consideration and define attacker's strategy from two perspectives, i.e., the effect of attack and the concealment of attack. Furthermore, we add a tradeoff parameter to attacker's objective function which can help the attacker balance the attraction of effect against the requirement of concealment. Finally, linear regression and logistic regression are chosen as examples and attacks are performed on them. The effectiveness of the attack is demonstrated by experimental results.

Acknowledgment

This work is supported by the National Key R&D Program of China under Grants 2017YFB0802300 and 2017YFB0802000.

References

[1] A. H. Wang, "Don't follow me: Spam detection in twitter," in *International Conference on Security and Cryptography*, 2011, pp. 142–151.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[3] M. Barreno, B. Nelson, R. Sears, A. D. Joseph, and J. D. Tygar, "Can machine learning be secure?" in *ACM Symposium on Information, Computer and Communications Security*, 2006, pp. 16–25.

[4] B. I. P. Rubinstein, B. Nelson, L. Huang, A. D. Joseph, S. H. Lau, S. Rao, N. Taft, and J. D. Tygar, "Antidote:understanding and defending against poisoning of anomaly detectors," in *ACM SIGCOMM Conference on Internet Measurement 2009, Chicago, Illinois, Usa, November*, 2009, pp. 1–14.

[5] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," *international conference on machine learning*, pp. 1807–1814, 2012.

[6] H. Xiao, B. Biggio, G. Brown, G. Fumera, C. Eckert, and F. Roli, "Is feature selection secure against training data poisoning," *international conference on machine learning*, vol. 2, pp. 1689–1698, 2015.

[7] S. Mei and X. Zhu, "The security of latent dirichlet allocation," *Artificial Intelligence and Statistics*, pp. 681–689, 2015.

[8] S. Alfeld, X. Zhu, and P. Barford, "Data poisoning attacks against autoregressive models," in *national conference on artificial intelligence*, 2016, pp. 1452–1458.

[9] B. Biggio, G. Fumera, F. Roli, and L. Didaci, "Poisoning adaptive biometric systems," in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*. Springer, 2012, pp. 417–425.

[10] S. Mei and X. Zhu, "Using machine teaching to identify optimal training-set attacks on machine learners," in *national conference on artificial intelligence*, 2015, pp. 2871–2877.

[11] X. Zhu, "Machine teaching: An inverse problem to machine learning and an approach toward optimal education," in *national conference on artificial intelligence (AAAI)*, 2015, pp. 4083–4087.

[12] L. Muñoz-González, B. Biggio, A. Demontis, A. Paudice, V. Wongrassamee, E. C. Lupu, and F. Roli, "Towards poisoning of deep learning algorithms with back-gradient optimization," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. ACM, 2017, pp. 27–38.

[13] M. Zhao, B. An, W. Gao, and T. Zhang, "Efficient label contamination attacks against black-box learning models," in *Proceedings of the IJCAI*, 2017, pp. 3945–3951.

[14] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nitarotaru, and B. Li, "Manipulating machine learning: Poisoning attacks and countermeasures for regression learning," *ieee symposium on security and privacy*, pp. 931–947, 2018.

[15] B. Biggio, I. Pillai, D. Ariu, M. Pelillo, and F. Roli, "Is data clustering in adversarial settings secure?" in *Proceedings of the 2013 ACM workshop on Artificial intelligence and security*, 2013, pp. 87–98.

[16] B. Biggio, K. Rieck, D. Ariu, C. Wressnegger, I. Corona, G. Giacinto, and F. Roli, "Poisoning behavioral malware clustering," in *The Workshop on Artificial Intelligent & Security Workshop*, 2014, pp. 27–36.

[17] B. Biggio, S. R. Bulo, I. Pillai, M. Mura, E. Z. Mequanint, M. Pelillo, and F. Roli, "Poisoning complete-linkage hierarchical clustering," *Lecture Notes in Computer Science*, vol. 8621, pp. 42–52, 2014.

[18] H. Li, Y. Yang, T. H. Luan, X. Liang, L. Zhou, and X. S. Shen, "Enabling fine-grained multi-keyword search supporting classified subdictionaries over encrypted cloud data," *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 3, pp. 312–325, 2016.

[19] H. Li, D. Liu, Y. Dai, and T. H. Luan, "Engineering searchable encryption of mobile cloud networks: When qoe meets qop," *IEEE Wireless Communications*, vol. 22, no. 4, pp. 74–80, 2015.

[20] H. Li, Y. Dai, L. Tian, and H. Yang, "Identity-based authentication for cloud computing," in *IEEE International Conference on Cloud Computing*. Springer, 2009, pp. 157–166.

[21] D. Lowd and C. Meek, "Adversarial learning," in *Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, 2005, pp. 641–647.

[22] B. Biggio, G. Fumera, and F. Roli, "Security evaluation of pattern classifiers under attack," *IEEE Transactions on Knowledge & Data Engineering*, vol. 26, no. 4, pp. 984–996, 2014.

[23] J. D. Tygar, "Adversarial machine learning," in *ACM Workshop on Security and Artificial Intelligence*, 2011, pp. 43–58.

[24] L. Munozgonzalez, B. Biggio, A. Demontis, A. Paudice, V. Wongrassamee, E. Lupu, and F. Roli, "Towards poisoning of deep learning algorithms with back-gradient optimization," *arXiv: Learning*, pp. 27–38, 2017.

[25] A. Kerckhoffs, "La cryptographie militaire," *Journal Des Sciences Militaires*, vol. ix, pp. 5–83.

[26] B. Colson, P. Marcotte, and G. Savard, "An overview of bilevel optimization," *Annals of Operations Research*, vol. 153, no. 1, pp. 235–256, 2007.