Efficient Label Contamination Attacks Against Black-Box Learning Models

Mengchen Zhao[1], Bo An[1], Wei Gao[2], Teng Zhang[2]
[1]School of Computer Science and Engineering, Nanyang Technological University, Singapore
[2]National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
[1]zhao0204@e.ntu.edu.sg, boan@ntu.edu.sg
[2]{gaow, zhangt}@lamda.nju.edu.cn

Abstract

Label contamination attack (LCA) is an important type of data poisoning attack where an attacker manipulates the labels of training data to make the learned model beneficial to him. Existing work on LCA assumes that the attacker has full knowledge of the victim learning model, whereas the victim model is usually a black-box to the attacker. In this paper, we develop a Projected Gradient Ascent (PGA) algorithm to compute LCAs on a family of empirical risk minimizations and show that an attack on one victim model can also be effective on other victim models. This makes it possible that the attacker designs an attack against a substitute model and transfers it to a black-box victim model. Based on the observation of the transferability, we develop a defense algorithm to identify the data points that are most likely to be attacked. Empirical studies show that PGA significantly outperforms existing baselines and linear learning models are better substitute models than nonlinear ones.

1 Introduction

The security of machine learning has been a critical concern for adversarial applications such as spam filtering, intrusion detection and malware detection [Li and Vorobeychik, 2014; Zhao *et al.*, 2016]. Attack techniques on machine learning can be classified as two categories: exploratory attacks and causative attacks [Barreno *et al.*, 2010]. The exploratory attack exploits the vulnerabilities of a classifier but does not affect training. For example, hackers can obfuscate malware code in order to bypass detection. The causative attack (also known as poisoning attack) influences the training process by manipulating the training data. This paper focuses on the label contamination attack, a type of causative attack that usually happens when labels of training data are collected from external sources. For example, one can use crowdsourcing platforms (e.g., Amazon Mechanical Turk) to collect labels from human workers; Netflix relies on users' ratings to improve their recommendation systems; collaborative spam filtering updates the email classifier periodically based on end-users' feedback, where malicious users can mislabel emails in their inboxes to feed false data to the updating process.

This paper studies the label contamination attack against a broad family of binary classification models. We focus on answering three questions that have not been addressed by existing work. First, consider a highly motivated attacker with full knowledge of the victim learning model, how to compute the label contamination attack against the victim model? Second, if the victim learning model is a black-box, how does the attacker design effective attacks against it? Third, how to defend against the label contamination attacks?

Previous work on label contamination attacks has three limitations [Biggio *et al.*, 2011; Xiao *et al.*, 2012]. First, they restrict the attacker's goal to decrease the accuracy of a victim learning model, whereas in reality the attackers may have arbitrary objectives. Second, they focus on computing attacks against Support Vector Machines (SVMs) and their algorithms cannot generalize to other victim learning models. Third, they assume that the attacker has full knowledge of the victim learning model, which might be unrealistic in reality. Regarding the defense against poisoning attacks, there are generally two lines of research: robust learning focuses on improving the robustness of learning algorithms under contaminated data [Biggio *et al.*, 2011], and data sanitization focuses on removing suspicious data from training set [Chan *et al.*, 2017; Fefilatyev *et al.*, 2012]. Most robust learning and data sanitization techniques require a set of clean data, which is used to develop metrics for identifying future contaminated data. However, such techniques become useless when the set of clean data is hard to obtain, or is contaminated by the attacker.

In this paper, we make five key contributions. First, we extend the existing work on label contamination attack to allow a broad family of victim models and arbitrary attacker objectives. We formulate the optimal attack problem as a mixed-integer bilevel program. Second, we exploit the Representer Theorem [Kimeldorf and Wahba, 1970] and propose a Projected Gradient Ascent (PGA) algorithm to approximately solve the bilevel program. Third, we propose a substitute-based attack method for attacking black-box learning models, which leverages the *transferability* of label contamination attacks. To our knowledge, we are the first to study the transferability of poisoning attacks. Finally, we empirical analyze the transferabilities with respect to five representative substitute models and show that linear models are significantly better substitutes than nonlinear ones.

2 Related Work

Poisoning attack against machine learning algorithms has become an emerging research in the field of adversarial machine learning [Barreno *et al.*, 2006; Huang *et al.*, 2011; Kloft and Laskov, 2010; Lowd and Meek, 2005]. A pioneer work studies the poisoning attacks on Support Vector Machines (SVMs), where the attacker is allowed to progressively inject malicious points to the training data in order to maximize the classification error [Biggio *et al.*, 2012]. In recent years, poisoning attack is generalized to many popular machine learning techniques, including feature selection algorithms [Xiao *et al.*, 2015], autoregressive models [Alfeld *et al.*, 2016], latent Dirichlet allocation [Mei and Zhu, 2015a] and factorization-based collaborative filtering [Li *et al.*, 2016]. An algorithmic framework for identifying the optimal training set attacks is provided in [Mei and Zhu, 2015b]. Note that all of the aforementioned works assume that the attacker has full knowledge of the learning model.

There has been existing work that studies the transferability of evasion attack, a type of exploratory attack where the attacker perturbs the legitimate inputs to induce the trained classifier to misclassify them. The transferability of such an attack means that the inputs perturbed to induce one classifier can also induce other classifiers to produce misclassifications. The transferability of evasion attack among deep neural networks (DNNs) is demonstrated by [Papernot *et al.*, 2016b]. Then, an extensive study explored the transferability of evasion attacks among five classifiers, including SVM, logistic regression, decision tree, k-nearest neighbors and DNNs [Papernot *et al.*, 2016a]. However, none of the existing work studies the transferability of poisoning attacks.

3 Label Contamination Attacks

In this section, we first introduce the label contamination attack against linear classifiers and formulate the optimal attack problem as a bilevel optimization problem. Then, we generalize our framework to solve the optimal attack against nonlinear kernel machines. We begin by introducing the linear binary classification problem. Given a set of training data $D = \{(\mathbf{x}_i, y_i)|\mathbf{x}_i \in \mathbb{R}^k, y_i \in \{-1, +1\}\}_{i=1}^n$, a linear classifier can be solved from the following optimization problem.

$$\min_{f\in\mathcal{H}} \quad C\sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i)) + \frac{1}{2}||f||^2 \tag{1}$$

where $f(\mathbf{x}_i) = \mathbf{w}^\intercal\mathbf{x}_i + b$ is the decision function, $||f||^2 = ||\mathbf{w}||^2$ is square of the ℓ_2 norm of $\mathbf{w}$, $\mathcal{H}$ is the hypothesis space, L is the loss function and C is the regularization parameter. For a testing instance $\mathbf{x}_i$, its predicted label is $sgn(f(\mathbf{x}_i))$. Without loss of generality, we denote $\mathbf{x}_i$ as $(1, \mathbf{x}_i)$ and denote $\mathbf{w}$ as $(b, \mathbf{w})$ so that $f(\mathbf{x}_i)$ can be equivalently represented by $\mathbf{w}^\intercal\mathbf{x}_i$, where $\mathbf{w} \in \mathbb{R}^{k+1}$.

Attacker's goal: Most existing work on poisoning attacks assumes that the attacker's goal is to decrease the classifier's accuracy [Biggio *et al.*, 2012; Xiao *et al.*, 2012]. A recent work allows the attacker to have an arbitrary objective model (a classifier) and the attacker's goal is to make the learner's learned model close to the objective model [Mei and Zhu, 2015b]. However, they restrict the attacker's objective model to be a linear classifier. We extend their setting to allow the attacker to have an arbitrary objective model, which is represented by a function $f^* : \mathbf{x} \to \{-1, +1\}$. We define two kinds of attacks regarding to the attacker's incentives in real world.

- Integrity attack. The attacker has some test instances $\{\mathbf{x}_i\}_{i=n+1}^m$ and wants the labels predicted by the victim model similar to that predicted by f^*. For example, a spammer may only want certain spam to be classified as regular ones. Note that $\{\mathbf{x}_i\}_{i=n+1}^m$ can be a mixture of instances that the attacker has preference on and those he is neural about.
- Availability attack. The attacker wants to decrease the accuracy of the victim model. For example, an attacker may want to disturb a recommender system by deceasing the accuracy of its built-in classification models.

Attacker's capability: In data poisoning attacks, an attacker who takes full control of training data can create an arbitrary victim model. However, in reality, the attacker usually faces some constraints. In this work, we assume that the attacker can flip at most B labels of the training set D. We denote by $D' = \{(\mathbf{x}_i, y_i')\}_{i=1}^n$ the contaminated training set. We introduce a binary vector $\mathbf{z}$ and denote $y_i' = y_i(1 - 2z_i)$ so that $z_i = 1$ means that the label of sample i is flipped and $z_i = 0$ otherwise.

3.1 Attacking Linear Classifiers

In this paper, we consider three linear classifiers: SVM, Logistic Regression (LR) and Least-squares SVM (LS-SVM), but note that our methods allow general loss functions as long as they are differentiable. The three classifiers can be obtained by replacing the loss function L in Eq.(1) with the following loss functions.

- Hinge loss (SVM): $L^1(y_i, f(\mathbf{x}_i)) = \max\{0, 1-y_i f(\mathbf{x}_i)\}$
- Logistic loss (LR): $L^2(y_i, f(\mathbf{x}_i)) = \log(1 + \exp(-y_i f(\mathbf{x}_i)))$
- Squared hinge loss (LS-SVM): $L^3(y_i, f(\mathbf{x}_i))=(1-y_i f(\mathbf{x}_i))^2$

For attacking linear classifiers, the attacker first reduces his objective model f^* to a weight vector $\mathbf{w}^* \in \mathbb{R}^{k+1}$. In other words, $\mathbf{w}^*$ can be viewed as a linear classifier that is the closest to f^*. Specifically, in the integrity attack, $\mathbf{w}^*$ can be learned from $D_{in}^a = \{(\mathbf{x}_i, y_i)|y_i = f^*(\mathbf{x}_i)\}_{i=n+1}^m$. In the availability attack, $\mathbf{w}^*$ can be learned from $D_{av}^a = \{(\mathbf{x}_i, -y_i)|(\mathbf{x}_i, y_i) \in D\}_{i=1}^n$. In both integrity attack and availability attack, the attacker wants the learner's learned weight vector $\mathbf{w}$ as close to $\mathbf{w}^*$ as possible. Since $\mathbf{w}$ and $\mathbf{w}^*$ can be viewed as two hyperlines in a $k+1$-dimensional space, intuitively, the attacker's goal can be viewed as rotating $\mathbf{w}$ to $\mathbf{w}^*$. We assume that the attacker's goal is maximizing the cosine of the angle between $\mathbf{w}$ and $\mathbf{w}^*$ and define the attacker's utility function as:

$$U(\mathbf{w}, \mathbf{w}^*) = \frac{\mathbf{w}^\intercal\mathbf{w}^*}{||\mathbf{w}||||\mathbf{w}^*||}.$$

We formulate the optimal attack problem as the following

bilevel program.

$$\max_{\mathbf{z}} \quad \frac{\mathbf{w}^\intercal \mathbf{w}^*}{||\mathbf{w}||||\mathbf{w}^*||} \tag{2}$$

$$\text{s.t.} \quad f \in \arg\min_{g\in\mathcal{H}} \quad C\sum_{i=1}^{n} L(y_i', g(\mathbf{x}_i)) + \frac{1}{2}||g||^2 \tag{3}$$

$$\textstyle\sum_{i=1}^{n} z_i \leq B \tag{4}$$

$$y_i' = y_i(1-2z_i), \forall i \in [n] \tag{5}$$

$$z_i \in \{0,1\}, \forall i \in [n] \tag{6}$$

One can obtain the optimal attack problem on specific linear classifiers by replacing the loss function L in Eq.(3) with the associated loss functions. Eqs.(2-6) is a mixed-integer bilevel program, which is generally hard to solve. We will introduce the PGA algorithm to approximately solve Eqs.(2-6) in Section 4.

3.2 Attacking Kernel SVMs

A kernel machine applies a feature mapping $\phi : \mathbb{R}^k \to \mathbb{R}^r$ on training data so that the data could be more separable in higher dimensional space (usually $r > k$). A kernel SVM can be viewed as a linear SVM in the transformed feature space. Since r can be arbitrarily large, instead of solving the primal problem Eq.(1), one usually solves its dual problem:

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\boldsymbol{\alpha}^\intercal Q\boldsymbol{\alpha} - \textstyle\sum_{i=1}^{n}\alpha_i \tag{7}$$

$$0 \leq \alpha_i \leq C, \forall i \in [n] \tag{8}$$

where $Q_{ij} = y_i y_j \phi(\mathbf{x}_i)^\intercal \phi(\mathbf{x}_j)$. In practice, $\phi(\mathbf{x}_i)^\intercal \phi(\mathbf{x}_j)$ in Eq.(7) is usually replaced by a kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$ to facilitate computation. We classify the kernel functions into two classes: one with finite feature mapping (e.g., polynomial kernels) and the other with infinite feature mapping (e.g., radial basis function kernels). We will introduce how to attack these kernel SVMs separately.

For kernel SVMs with finite feature mapping, the attacker first reduces his objective model f^* to a weight vector $\mathbf{w}^* \in \mathbb{R}^{r+1}$. Similar to the linear classification case, in the integrity attack, $\mathbf{w}^*$ can be learned from $D_{in}^a = \{(\phi(\mathbf{x}_i), y_i) | y_i = f^*(\mathbf{x}_i)\}_{i=n+1}^m$. In the availability attack, $\mathbf{w}^*$ can be learned from $D_{av}^a = \{(\phi(\mathbf{x}_i), -y_i) | (\mathbf{x}_i, y_i) \in D\}_{i=1}^n$. For kernel SVMs with infinite feature mapping, we use the technique of random Fourier features [Rahimi and Recht, 2009; Rahimi *et al.*, 2007] to construct an approximate finite feature mapping. The random Fourier features are constructed by first sampling random vectors $\omega_1, ..., \omega_q$ from $p(\omega)$, where $p(\omega)$ is the Fourier transform of kernel function K. Then, $\mathbf{x}_i$ is transformed to $\phi(\mathbf{x}_i)$ with new features

$$\phi(\mathbf{x}_i) = (sin(\omega_1^\intercal \mathbf{x}_i), cos(\omega_1^\intercal \mathbf{x}_i), ..., sin(\omega_q^\intercal \mathbf{x}_i), cos(\omega_q^\intercal \mathbf{x}_i)).$$

One can refer to [Rahimi *et al.*, 2007] for detailed procedures. The random Fourier features ensures $\phi(\mathbf{x}_i)^\intercal \phi(\mathbf{x}_j) \approx K(\mathbf{x}_i, \mathbf{x}_j)$. Note that the dimension of ϕ is $2q$, where q is the number of random vectors drawn from $p(\omega)$. The attacker can construct his objective model $\mathbf{w}^* \in \mathbb{R}^{2q+1}$ similarly to the finite feature mapping case using the feature mapping ϕ.

Algorithm 1: Projected Gradient Ascent (PGA)

1 Input: Original training data $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, attacker's objective model $\mathbf{w}^*$, budget B, step size η, iteration limit t^{max};
2 Choose a random $\mathbf{z}^0 \in [0,1]^n$;
3 $\mathbf{y}'^0 \leftarrow$ Flip($\mathbf{z}^0$);
4 Train a classifier using training data $\{(\mathbf{x}_i, \mathbf{y}'^0)\}$;
5 Initialize dual variables $\boldsymbol{\alpha}^0$ and primal variables $\mathbf{w}^0$;
6 $t \leftarrow 1$;
7 **while** *Not converge and* $t < t^{max}$ **do**
8 $\quad \mathbf{z}^t \leftarrow$ Proj($\mathbf{z}^{t-1} + \eta \nabla_{\mathbf{z}^{t-1}} U$);
9 $\quad \mathbf{y}'^t \leftarrow$ Flip($\mathbf{z}^t$);
10 $\quad$ Retrain the classifier using training data $\{(\mathbf{x}_i, \mathbf{y}'^t)\}$;
11 $\quad$ Update $\boldsymbol{\alpha}^t, \mathbf{w}^t$;
12 $\quad t \leftarrow t+1$;
13 **end**
14 Output: Contaminated labels $\mathbf{y}'^t$.

Algorithm 2: Flip strategy

1 Input: $\mathbf{z}$, original labels $\mathbf{y}$, budget B;
2 $\Gamma \leftarrow$ Indices of $Sort([z_1, z_2, ..., z_n], \text{`}descent\text{'})$;
3 $j \leftarrow 1, y_i' \leftarrow y_i, \forall i \in [n]$;
4 **while** $\sum_{i=1}^n z_{\Gamma(j)} \leq B$ **do**
5 $\quad y'_{\Gamma(j)} \leftarrow -y_{\Gamma(j)}$;
6 $\quad j \leftarrow j+1$;
7 **end**
8 Output: Flipped labels $\mathbf{y}'$.

In order to obtain the optimal attack problem on kernel SVMs, we need to replace the lower level problem Eq.(3) with Eqs.(7-8) and add constraint Eq.(9) to the upper level problem, which is derived from the Representer Theorem [Kimeldorf and Wahba, 1970].

$$\mathbf{w} = \textstyle\sum_{i=1}^{n} \alpha_i y_i' \phi(\mathbf{x}_i) \tag{9}$$

4 Computing Attacking Strategies

Inspired by [Li *et al.*, 2016; Mei and Zhu, 2015b; Xiao *et al.*, 2015], we develop the PGA algorithm for computing approximate solutions of Eqs.(2-6) and show that PGA can also compute attack strategies on kernel SVMs. We first relax binary variables z_i to interval $[0,1]$ and solve the relaxed problem. PGA works by gradually updating $\mathbf{z}^t$ along its approximate gradients until converge or the iteration limit is reached. Since $\mathbf{z}^t$ is a real number vector and retraining the classifier requires $\mathbf{y}'^t$ to be a binary vector, we construct a *flip strategy* to project $\mathbf{z}^t$ to $\mathbf{y}'^t$. The flip strategy is shown in Algorithm 2. At each iteration, the projector Proj($\mathbf{z}$) first projects $\mathbf{z}$ to an ℓ_∞ norm ball by truncating each z_i into range $[0,1]$. Then the projected point is further projected to an ℓ_1 norm ball with diameter B, which ensures that $\sum_{i=1}^n z_i \leq B$.

Steps 4 and 10 in PGA involve training process of the victim model. If the victim model is SVM, in step 4 and 10 we solve the dual SVM problem Eqs.(7-8). If the victim model

is logistic regression, we solve the following dual logistic regression problem:

$$\max_{\boldsymbol{\alpha}} \quad \frac{1}{2}\boldsymbol{\alpha}^\intercal Q\boldsymbol{\alpha} + \sum_{i:\alpha_i>0} \alpha_i \log \alpha_i + \sum_{i:\alpha_i<C} (C-\alpha_i)\log(C-\alpha_i) \tag{10}$$

$$0 \leq \alpha_i \leq C, \forall i \in [n] \tag{11}$$

where $Q_{ij} = y'_i y'_j \mathbf{x}_i{}^\intercal \mathbf{x}_j$. If the victim model is least-squares SVM, we solve the dual least-squares SVM problem:

$$(Q + C^{-1} I_n)\boldsymbol{\alpha} = \mathbf{1}_n \tag{12}$$

where I_n is the $n \times n$ identical matrix and $\mathbf{1}_n$ is the n-dimensional vector of element 1. If the victim learning model is kernel SVM, we solve Eqs.(7-8) with $Q_{ij} = y'_i y'_j \phi(\mathbf{x}_i)^\intercal \phi(\mathbf{x}_j)$. In step 5 (step 11) of PGA, $\boldsymbol{\alpha}^0$ ($\boldsymbol{\alpha}^t$) is the solution of the problem solved in step 4 (step 10) and $\mathbf{w}^0$ ($\mathbf{w}^t$) is computed using Eq.(9).

In order to compute the gradient $\nabla_{\mathbf{z}} U$ in step 8, we first apply chain rule to arrive at:

$$\nabla_{\mathbf{z}} U = \nabla_{\mathbf{w}} U \cdot \nabla_{\mathbf{y}'} \mathbf{w} \cdot \nabla_{\mathbf{z}} \mathbf{y}' \tag{13}$$

The first and the third gradient can be easily computed as:

$$\frac{\partial U}{\partial w_j} = \frac{||\mathbf{w}||^2 w_j^* - \mathbf{w}^\intercal \mathbf{w}^* w_j}{||\mathbf{w}||^3 ||\mathbf{w}^*||} \tag{14}$$

$$\frac{\partial y'_i}{\partial z_j} = -\mathbb{1}(i=j) 2 y_i \tag{15}$$

where $\mathbb{1}(\cdot)$ is the indicator function. The second gradient $\nabla_{\mathbf{y}'} \mathbf{w}$ is hard to compute since it involves an optimization procedure. We leverage Eq.(9) to approximately compute the second gradient. If the victim learning model is linear, Eq.(9) is modified as $\mathbf{w} = \sum_{i=1}^n \alpha_i y'_i \mathbf{x}_i$. Taking the derivatives of both sides we have:

$$\frac{\partial w_j}{\partial y'_i} = \alpha_i x_{ij} \tag{16}$$

If the victim learning model is a kernel SVM, we can take derivatives of both sides of Eq.(9) and obtain:

$$\frac{\partial w_j}{\partial y'_i} = \alpha_i \phi(x_{ij}) \tag{17}$$

5 Attacking Black-Box Victim Models Using Substitutes

In previous sections we introduced how to compute attacks against a broad family of learning models. However, the attacker may not have full knowledge of the victim learning model in many real-world scenarios. Observing that an attack targets on one learning model can also be effective on another learning model even if the two models have different architectures, the attacker can design attack against a substitute model and then perform this attack on the victim learning model. A good substitute model is the one that the attack against it is also effective on a general family of learning models.

The effectiveness of substitute-based attacks on a victim model can be evaluated by the victim model's accuracy on a test set D_{test}. In the integrity attack, D_{test} can be $D_{in}^a = \{(\mathbf{x}_i, y_i) | y_i = f^*(\mathbf{x}_i)\}_{i=n+1}^m$ and in the availability attack D_{test} can be $D_{av}^a = \{(\mathbf{x}_i, -y_i) | (\mathbf{x}_i, y_i) \in D\}_{i=1}^n$. As discussed in Section 3, the attacker aims to increase the classification accuracy of the victim model on D_{test} because the attacker's objective model $\mathbf{w}^*$ is learned from D_{in}^a and D_{av}^a, with respect to integrity attack and availability attack. We denote by $M = \{M_1, M_2, ..., M_{|M|}\}$ the set of learning models and by τ_i the attack against model M_i. We denote by $M_j^{\tau_i}$ the victim model M_j learned under attack τ_i. Then the effectiveness of the attack against substitute M_i on victim model M_j can be evaluated by $accuracy(M_j^{\tau_i}, D_{test})$. We will evaluate the effectiveness of five substitute models on eight victim models in Section 7.3.

6 A Discussion on Possible Defenses

The ultimate goal of adversarial machine learning is to develop defense strategies based on the analysis of the attacker's strategic behavior. Regarding the label contamination attacks, there are two main difficulties in developing defense strategies. First, the attacker's goal is hard to estimate. For example, the attacker can perform integrity attack, availability attack, or even a hybrid attack. Since the attacker's strategy is optimized with respect to his goal, it is difficult for the learner to accurately estimate the attacker strategy without knowing his goal. Second, most existing defense methods (e.g., robust learning) require a set of clean data (true labels), and future data will be judged based on the metrics developed using the clean data. However, in practice, it is often expensive to obtain enough true labels, especially when domain experts are employed.

The analysis of data poisoning attacks provides opportunities to develop alternative defense strategies in the future. Here we discuss two possible future directions to develop defense strategies. 1) Discovering the characteristics of poisoned data and identify the data that are most likely to be attacked. For example, from Figure 1 we can see that the attacked data basically form two clusters (one cluster with big blue points and the other cluster with big red points). Therefore, if we have successfully identified an attacked point, we can look into its adjacent points. In addition, most attacked points are extreme points, which indicates that the extreme points are more likely to be attacked than those near centroid. 2) Game-theoretic modeling. Adversarial machine learning can be viewed as a game between the learner and the attacker. Previous work has applied game-theoretic analysis on test-set attacks. For instance, Alfeld *et al.* [2017] model the test-set attack problem as a Stackelberg game [An *et al.*, 2015]. They assume that the learner has a set of *explicit defense actions*, such as verifying data with third parties, and try to compute the optimal defense action. However, few work applies game-theoretic analysis on poisoning attacks. Although the study of poisoning attacks provide a framework to model the attacker behavior, but the learner's defense actions have not been considered in traditional poisoning attack setting. If we consider defense actions, such as giving penalty on detected attacker behavior, we might be able to develop realistic game models and more secure learning algorithms.

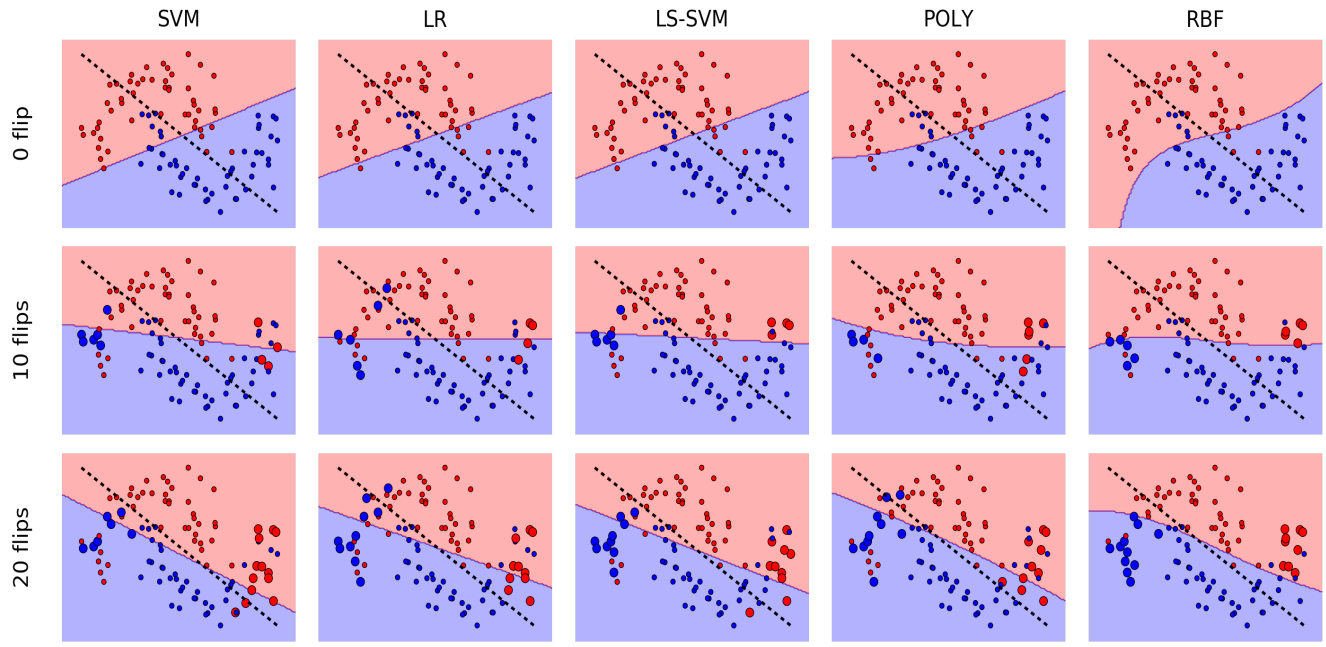

Figure 1: Decision boundaries (solid lines) of learned models under attacks with different attacker budgets. The dashed black lines represent the attacker's objective model. The attacker wants the points on the left side of this line to be classified as "red" and the points on the right side to be classified as "blue". The bigger red (blue) points are originally blue (red) and are flipped by the attacker.

7 Experimental Evaluation

In this section, we evaluate the proposed attack and defense algorithms and analyze the transferability of the attacks. We compute attacks against three linear learning models: SVM, logistic regression (LR), least-squares SVM (LS-SVM) and two nonlinear models: SVM with polynomial kernel (POLY) and radial basis function kernel (RBF). We will use five public data sets: Australian (690 points, 14 features), W8a (10000 points, 300 features), Spambase (4601 points, 57 features) [Lichman, 2013], Wine (130 points, 14 features) and Skin (5000 points, 3 features) [1]. All training processes are implemented with LIBSVM [Chang and Lin, 2011] and LIBLINEAR [Fan *et al.*, 2008]. All attacks computed by PGA are the best among 50 runs.

7.1 Integrity Attacks Visualization

We visualize the integrity attacks against SVM, LR, LS-SVM, POLY, RBF computed by PGA. We set the regularization parameter C=1 for all five models. We set the parameters d=2 for polynomial kernel and γ=0.1 for RBF kernel. The training set is a 2-D artificial data set containing 100 points. We ignore the process of generating the attacker's objective model and set it as an arbitrary one. Figure 1 shows how the attacks under different attacker budgets can affect the decision boundaries of victim models. We can see that the victim learning models can be converted to models that are very close to the attacker's objective model under only 20 flips. In addition, the attacked points with respect to different victim models are highly similar, which indicates that the attacks have transferability.

7.2 Solution Quality Comparison

We compute availability attacks against SVM using PGA and compare our solution with two baselines. The first baseline is a random flip strategy, where the attacker randomly flips the labels of training data under his budget. For each data set and budget, we compute the random attack for 50 times and report the best out of them. The second baseline, Adversarial Label Flip Attack on SVMs (ALFA) [Xiao *et al.*, 2012], is an existing algorithm that can compute attacks that decrease the accuracy of SVMs. ALFA works by iteratively solving a quadratic and a linear problem until convergence. Figure 2 shows that the attacks computed by PGA significantly outperform both baselines. On the W8a data set, the attacks computed by PGA decrease the victim model's accuracy from 90% to 30% with only 30% flips. We also find that PGA scales significantly better than ALFA. Because in each iteration of ALFA it solves two optimization problems and their sizes grow with the number of data points, while in each iteration of PGA it trains a linear classifier, which can be efficiently implemented with LIBLINEAR.

7.3 Transferability Analysis

We compute availability attacks against the aforementioned five substitute models using PGA and test the accuracy of eight victim models under the attacks. The victim models

[1] Except Spambase, all data sets can be downloaded from `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`.

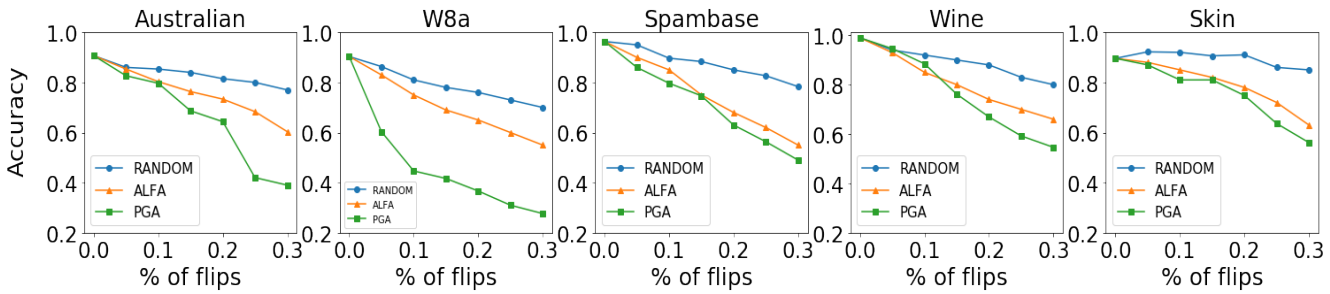

Figure 2: Accuracy of victim model under different attacker budgets. x-axis is the percentage of flipped points and y-axis is the accuracy of victim model on training set.

include the five substitute models and decision tree (DT), k-nearest neighbors (KNN) and Naive Bayes (NB). The DT, KNN and NB models are trained using MATLAB R2016b Statics and Machine Learning Toolbox and all parameters are set by default. We set the attacker's budget as 30% of the training points.

Table 1 shows the influence of the five attacks on the eight victim models. First, we can see from the diagonal values that if the substitute model and the victim model are of the same type, the attack can significantly degrade the accuracy of the victim model. Second, the performance of an attack designed for a linear model on another linear victim model are comparable with the attack designed for the victim model, which means that the attack designed for a linear model has a good transferability when victim models are also linear. Third, the attack designed for a linear model has a good transferability when the victim model is nonlinear. However, the attack designed for nonlinear models has a bad transferability when the victim models are linear. For example, on the Skin dataset, the attack designed for RBF can degrade the accuracy of an RBF model to 0.38. However, an SVM victim model under this attack can still achieve 0.94 accuracy, which means that the attack barely has influence on the SVM model. Fourth, on the Australian and the Skin dataset, the attacks designed for the five substitute models have similar transferability when the victim models are DT, KNN and NB. However on the Spambase dataset, the attacks designed for linear models have significantly better transferability than those designed for nonlinear models. In conclusion, attacks against linear models generally have a good transferability than that against nonlinear models.

8 Conclusion

This paper studies label contamination attacks against classification models. We first focused on the problem of optimal label contamination attack against a family of empirical risk minimization models. We formulated each optimal attack problem as a mixed integer bilevel program and developed the PGA algorithm to compute the near-optimal attacks. Then, we considered a more realistic scenario where the victim model are a black-box to the attacker. In such a scenario, we proposed a substitute-based attacking strategy for the attacker. In the experimental part, we studied the transferability of the label contamination attacks and demonstrated that the substitute-based attacks can be very effective against black-box learning models when appropriate substitute model is chosen. We also discussed about possible defenses to mitigate data poisoning attacks.

	SVM	LR	LS-SVM	POLY	RBF	DT	KNN	NB
SVM	0.40	0.55	0.42	0.63	0.49	0.68	0.70	0.39
LR	0.55	0.53	0.48	0.59	0.49	0.69	0.70	0.33
LS-SVM	0.53	0.54	0.25	0.63	0.64	0.66	0.70	0.33
POLY	0.67	0.68	0.55	0.53	0.52	0.62	0.70	0.43
RBF	0.82	0.78	0.69	0.67	0.55	0.64	0.70	0.48

(a)Australian dataset.

	SVM	LR	LS-SVM	POLY	RBF	DT	KNN	NB
SVM	0.45	0.47	0.48	0.62	0.55	0.68	0.70	0.35
LR	0.54	0.48	0.48	0.63	0.67	0.69	0.70	0.33
LS-SVM	0.53	0.50	0.50	0.63	0.66	0.68	0.69	0.36
POLY	0.74	0.73	0.74	0.76	0.74	0.70	0.70	0.61
RBF	0.83	0.81	0.82	0.84	0.54	0.71	0.71	0.78

(b)Spambase dataset.

	SVM	LR	LS-SVM	POLY	RBF	DT	KNN	NB
SVM	0.56	0.59	0.58	0.59	0.58	0.71	0.70	0.56
LR	0.57	0.59	0.57	0.61	0.78	0.70	0.70	0.56
LS-SVM	0.46	0.52	0.51	0.50	0.46	0.67	0.69	0.46
POLY	0.90	0.60	0.69	0.52	0.77	0.69	0.75	0.45
RBF	0.94	0.90	0.88	0.91	0.38	0.76	0.69	0.58

(c)Skin dataset.

Table 1: Accuracy of victim models under substitute-based attacks.

Acknowledgements

This research is supported by NRF2015 NCR-NCR003-004. Wei Gao is supported by the NSFC (61503179) and Jiangsu SF(BK20150586).

References

[Alfeld *et al.*, 2016] Scott Alfeld, Xiaojin Zhu, and Paul Barford. Data poisoning attacks against autoregressive models. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 1452–1458, 2016.

[Alfeld *et al.*, 2017] Scott Alfeld, Xiaojin Zhu, and Paul Barford. Explicit defense actions against test-set attacks. In *Proceedings of the 31th AAAI Conference on Artificial Intelligence*, pages 1274–1280, 2017.

[An *et al.*, 2015] Bo An, Milind Tambe, and Arunesh Sinha. Stackelberg security games (ssg): Basics and application

overview. *Improving Homeland Security Decisions. Cambridge University Press, forthcoming*, 2015.

[Barreno *et al.*, 2006] Marco Barreno, Blaine Nelson, Russell Sears, Anthony D Joseph, and J Doug Tygar. Can machine learning be secure? In *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security*, pages 16–25, 2006.

[Barreno *et al.*, 2010] Marco Barreno, Blaine Nelson, Anthony D Joseph, and JD Tygar. The security of machine learning. *Machine Learning*, 81(2):121–148, 2010.

[Biggio *et al.*, 2011] Battista Biggio, Blaine Nelson, and Pavel Laskov. Support vector machines under adversarial label noise. *The 3rd Asian Conference on Machine Learning*, 20:97–112, 2011.

[Biggio *et al.*, 2012] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Conference on Machine Learning*, pages 1807–1814, 2012.

[Chan *et al.*, 2017] Patrick PK Chan, Zhi-Min He, Hongjiang Li, and Chien-Chang Hsu. Data sanitization against adversarial label contamination based on data complexity. *International Journal of Machine Learning and Cybernetics*, pages 1–14, 2017.

[Chang and Lin, 2011] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[Fan *et al.*, 2008] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9(Aug):1871–1874, 2008.

[Fefilatyev *et al.*, 2012] Sergiy Fefilatyev, Matthew Shreve, Kurt Kramer, Lawrence Hall, Dmitry Goldgof, Rangachar Kasturi, Kendra Daly, Andrew Remsen, and Horst Bunke. Label-noise reduction with support vector machines. In *Proceedings of the 21st International Conference on Pattern Recognition*, pages 3504–3508, 2012.

[Huang *et al.*, 2011] Ling Huang, Anthony D Joseph, Blaine Nelson, Benjamin IP Rubinstein, and JD Tygar. Adversarial machine learning. In *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence*, pages 43–58, 2011.

[Kimeldorf and Wahba, 1970] George S Kimeldorf and Grace Wahba. A correspondence between bayesian estimation on stochastic processes and smoothing by splines. *The Annals of Mathematical Statistics*, 41(2):495–502, 1970.

[Kloft and Laskov, 2010] Marius Kloft and Pavel Laskov. Online anomaly detection under adversarial impact. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, pages 405–412, 2010.

[Li and Vorobeychik, 2014] Bo Li and Yevgeniy Vorobeychik. Feature cross-substitution in adversarial classification. In *Prodeedings of the 28th Advances in Neural Information Processing Systems*, pages 2087–2095, 2014.

[Li *et al.*, 2016] Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. Data poisoning attacks on factorization-based collaborative filtering. In *Prodeedings of the 30th Annual Conference on Neural Information Processing Systems*, pages 1885–1893, 2016.

[Lichman, 2013] M. Lichman. UCI machine learning repository, 2013.

[Lowd and Meek, 2005] Daniel Lowd and Christopher Meek. Adversarial learning. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 641–647, 2005.

[Mei and Zhu, 2015a] Shike Mei and Xiaojin Zhu. The security of latent dirichlet allocation. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, pages 681–689, 2015.

[Mei and Zhu, 2015b] Shike Mei and Xiaojin Zhu. Using machine teaching to identify optimal training-set attacks on machine learners. In *Proceedings of the 29st AAAI conference on Artificial intelligence*, pages 2871–2877, 2015.

[Papernot *et al.*, 2016a] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: From phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

[Papernot *et al.*, 2016b] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *Proceedings of the 1st IEEE European Symposium on Security and Privacy*, pages 372–387, 2016.

[Rahimi and Recht, 2009] Ali Rahimi and Benjamin Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*, pages 1313–1320, 2009.

[Rahimi *et al.*, 2007] Ali Rahimi, Benjamin Recht, et al. Random features for large-scale kernel machines. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems*, number 4, page 5, 2007.

[Xiao *et al.*, 2012] Han Xiao, Huang Xiao, and Claudia Eckert. Adversarial label flips attack on support vector machines. In *Proceedings of the 20th European Conference on Artificial Intelligence*, pages 870–875, 2012.

[Xiao *et al.*, 2015] Huang Xiao, Battista Biggio, Gavin Brown, Giorgio Fumera, Claudia Eckert, and Fabio Roli. Is feature selection secure against training data poisoning? In *Proceedings of the 32th International Conference on Machine Learning*, pages 1689–1698, 2015.

[Zhao *et al.*, 2016] Mengchen Zhao, Bo An, and Christopher Kiekintveld. Optimizing personalized email filtering thresholds to mitigate sequential spear phishing attacks. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 658–665, 2016.