Model-Targeted Poisoning Attacks with Provable Convergence

Fnu Suya [1] **Saeed Mahloujifar** [2] **Anshuman Suri** [1] **David Evans** [1] **Yuan Tian** [1]

Abstract

In a poisoning attack, an adversary who controls a small fraction of the training data attempts to select that data, so a model is induced that misbehaves in a particular way. We consider poisoning attacks against convex machine learning models and propose an efficient poisoning attack designed to induce a model specified by the adversary. Unlike previous model-targeted poisoning attacks, our attack comes with provable convergence to *any* attainable target model. We also provide a lower bound on the minimum number of poisoning points needed to achieve a given target model. Our method uses online convex optimization and finds poisoning points incrementally. This provides more flexibility than previous attacks which require an a priori assumption about the number of poisoning points. Our attack is the first model-targeted poisoning attack that provides provable convergence for convex models. In our experiments, it either exceeds or matches state-of-the-art attacks in terms of attack success rate and distance to the target model.

1. Introduction

Machine learning often requires extensive labeled training data, often collected from untrusted sources. A typical application is email spam filtering: a spam detector filters messages based on features (e.g., presence of certain words) and trains the model using emails labeled by users. In such a setting, spammers can generate spam messages that inject benign words likely to occur in legitimate emails. Models trained on these spam messages observe significant drops in filtering accuracy (Nelson et al., 2008; Huang et al., 2011). Such attacks are known as *poisoning attacks*, and a training process that uses labels or data from untrusted sources is potentially vulnerable to them.

[1]University of Virginia [2]Princeton University. Correspondence to: Fnu Suya <suya@virginia.edu>, Saeed Mahloujifar <sfar@princeton.edu>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

Most work on poisoning attacks has considered one of two extremal attacker objectives: *indiscriminate* attacks, where the adversary's goal is simply to decrease the overall accuracy of the model (Biggio et al., 2012; Xiao et al., 2012; Mei & Zhu, 2015b; Steinhardt et al., 2017; Koh et al., 2018); and *instance* attacks, where the goal is to induce a classifier that misclassifies a particular input (Shafahi et al., 2018; Zhu et al., 2019; Koh & Liang, 2017; Geiping et al., 2021; Huang et al., 2020). Recently, Jagielski et al. (2019) introduced a richer attacker objective known as a *subpopulation* attack, where the goal is to increase the error rate or obtain a particular output for a defined subset of the data distribution.

Depending on the general approach, poisoning attacks can be categorized as *objective-driven* or *model-targeted*. Objective-driven poisoning attacks have a specified attacker objective (such as reducing the overall accuracy of the victim model), and aim to induce a model that maximizes that objective. Model-targeted attacks have a specific target model in mind (that satisfies some attacker objective) and aim to induce a victim model as close as possible to that target model. Objective-driven attacks are most commonly studied in the existing literature, and indeed, it is natural to think about attacks in terms of an adversary's goals. We argue, though, that breaking poisoning attacks into the two steps of first finding a model to target and then selecting poisoning points to induce that model has significant advantages. This view leads to improvements in our understanding of poisoning attacks and simplifies the task of designing effective attacks for various objectives. Importantly, it can also lead to more effective poisoning attacks.

For objective-driven attacks, gradient-based local optimization is most commonly used to construct poisoning points for a particular attacker objective (Biggio et al., 2012; Xiao et al., 2012; Mei & Zhu, 2015b; Koh & Liang, 2017; Shafahi et al., 2018; Zhu et al., 2019). These attacks can also be modified to fit other attacker objectives. However, since they are based on local optimization techniques, they often get stuck into bad local optima and fail to find effective sets of poisoning points (Steinhardt et al., 2017; Koh et al., 2018). The min-max attack by Steinhardt et al. (2017) circumvents the issue of local optima by adopting online learning but only applies to the indiscriminate setting. In contrast, model-targeted attacks can incorporate any attacker objective into a target model. Thus, the same model-targeted attack methods can be directly applied to a range of indiscriminate and subpopulation attacks by finding a suitable target model. In addition, one can avoid the issue of local optima in poisoning by working with a target model (Koh et al., 2018).

Contributions. Our main contribution is a principled and general model-targeted poisoning method, along with proof that the model it induces converges to the target model. We study poisoning attacks on convex models because poisoning attacks in these simpler settings are still not well understood. In addition, many important industrial applications continue to rely on simple models due to their easiness in model debugging and low computational cost. For many applications, such simple convex models also have comparable or better performances than complex deep neural networks (Dacrema et al., 2019; Tramèr & Boneh, 2021).

Our attack method works in an online fashion and finds effective poisoning points incrementally. For settings where the loss function is convex and proper regularization is adopted in training, we prove that the model induced by training on the poisoned data set converges to the target model as the number of poisoning points increases (Theorem 1). Previous model-targeted attacks lack such convergence guarantees. In addition, our attack applies to incremental poisoning scenarios. It does not require a predetermined poisoning rate while previous model-targeted attacks assume a priori number of poisoning points, which is typically unavailable in practice. We then prove a lower bound on the minimum number of poisoning points needed to reach the target model (Theorem 2). Such a lower bound can be used to estimate the optimality of model-targeted poisoning attacks and also indicate the intrinsic hardness of attacking different targets.

We evaluate our attack and compare it to the state-of-the-art model-targeted attack (Koh et al., 2018). We evaluate the convergence of our attack to the target model and find that our attack can induce models closer to the target model for all target models we tried for the same number of poisoning points (Section 5). The success rate of our attack exceeds that of the state-of-the-art attack in subpopulation attack scenarios and is comparable for indiscriminate attacks. As a supplement, we also compare our attack to the state-of-the-art objective-driven attacks (Appendix D). With carefully selected target models, we show that our attack is also more effective in achieving the attacker objectives than existing objective-driven attacks.

In this work, we mostly focus on the effectiveness of our poisoning attack in inducing a given target model that satisfies the underlying objective of the adversary. These target models are selected using heuristic, objective-driven methods. When employing our attack, we aim to induce that target model with fewer poisoning points than were needed by the heuristic method. Note that the selection of this target model could be improved to improve the effectiveness of our attack. We defer to future work a full exploration of how to improve the selection of target models for particular attacker objectives (but include some preliminary results on this in Appendix C.3). Also, note that the goal of inducing a target model also aligns with the goal of previous model-targeted poisoning attacks (Koh et al., 2018; Mei & Zhu, 2015b).

2. Problem Setup

The poisoning attack proposed in this paper applies to multiclass prediction tasks or regression problems (by treating the response variable as an additional data feature), but for simplicity of presentation we consider a binary prediction task, $h : \mathcal{X} \rightarrow \mathcal{Y}$, where $X \subseteq \mathbb{R}^d$ and $\mathcal{Y} = \{+1, -1\}$. The prediction model h is characterized by parameters $\theta \in \Theta \subseteq \mathbb{R}^d$. We define the non-negative convex loss on an individual point, (x, y), as $l(\theta; x, y)$ (e.g., hinge loss for SVM model). We also define the empirical loss over a set of points A as $L(\theta; A) = \sum_{(x,y)\in A} l(\theta; x, y)$.

We adopt the game-theoretic formalization of the poisoning attack process from Steinhardt et al. (2017) to describe our model-targeted attack scenario:

1. N data points are drawn uniformly at random from the true data distribution over $\mathcal{X} \times \mathcal{Y}$ and form the clean training set, $\mathcal{D}_c$.
2. The adversary, with knowledge of $\mathcal{D}_c$, the model training process and the model space Θ, generates a target classifier $\theta_p \in \Theta$ that satisfies the attack goal.
3. The adversary produces a set of poisoning points, $\mathcal{D}_p$, with the knowledge of $\mathcal{D}_c$, model training process, Θ and θ_p.
4. Model builder trains the model on $\mathcal{D}_c \cup \mathcal{D}_p$ and produces a classifier, θ_{atk}.

The adversary's goal is that the induced classifier, θ_{atk}, is close to the desired target classifier, θ_p (Section 4.2 discusses how this distance is measured). Step 2 corresponds to the target classifier generation process. Our attack works for any target classifier, and in the paper we do not focus on the question of how to find the best target classifier to achieve a particular adversarial goal but simply adopt the heuristic target classifier generation process from Koh et al. (2018). Step 3 corresponds to our model-targeted poisoning attack and is also the main contribution of the paper.

We assume the model builder trains a model through empirical risk minimization (ERM) and the training process details are known to the attacker:

$$\theta_c = \underset{\theta\in\Theta}{\arg\min}\, \frac{1}{|\mathcal{D}_c|} L(\theta; \mathcal{D}_c) + C_R \cdot R(\theta) \tag{1}$$

where $R(\theta)$ is the nonnegative regularization function (e.g., $\frac{1}{2}\|\theta\|_2^2$ for SVM model).

Threat Model. We assume an adversary with full knowledge of training data, model space, and training process. Although this may be unrealistic for many scenarios, this setting allows us to focus on a particular aspect of poisoning attacks and is the setting used in many prior works (Biggio et al., 2011; Mei & Zhu, 2015b; Steinhardt et al., 2017; Koh et al., 2018; Shafahi et al., 2018). We assume an addition-only attack where the attacker only adds poisoning points into the clean training set. A stronger attacker may be able to modify or remove existing points, but this typically requires administrative access to the system, which is generally hard to obtain. The added points are unconstrained, other than being valid value elements of the input space. They can have arbitrary features and labels, which enables us to perform the worst-case analysis on the robustness of models against addition-only poisoning attacks. Although some previous works also allow arbitrary selection of the poisoning points (Biggio et al., 2011; Mei & Zhu, 2015b; Steinhardt et al., 2017; Koh et al., 2018), others put different restrictions on the poisoning points. A clean-label attack assumes adversaries can only perturb the features of the data, but the label is given by a labeling oracle (Koh & Liang, 2017; Shafahi et al., 2018; Zhu et al., 2019; Huang et al., 2020). In label-flipping attacks, adversaries are only allowed to change the labels (Biggio et al., 2011; Xiao et al., 2012; 2015; Jagielski et al., 2019). These restricted attacks are weaker than unrestricted poisoning attacks (Koh et al., 2018; Hong et al., 2020).

3. Related Work

Here, we summarize previous work on objective-driven or model-targeted poisoning attacks.

Objective-Driven Attacks. The most commonly used poisoning strategy for objective-driven attacks is a gradient-based attack. Gradient-based attacks on classification tasks iteratively modify a candidate poisoning point $(\hat{x}, \hat{y})$ in the set $\mathcal{D}_p$ based on the test loss (related to the attacker objective) defined on $\hat{x}$ while keeping $\hat{y}$ fixed. This kind of attack was first studied on SVM models by Biggio et al. (2012), and later extended to logistic regression (Demontis et al., 2019) and larger neural network models (Koh & Liang, 2017; Shafahi et al., 2018; Zhu et al., 2019; Huang et al., 2020). Jagielski et al. (2018) studied gradient attacks and principled defenses on linear regression tasks. In addition to classification and regression tasks, gradient-based poisoning attacks have also been applied to topic modeling (Mei & Zhu, 2015a), collaborative filtering (Li et al., 2016) and algorithmic fairness (Solans et al., 2020).

Researchers have also explored using generative adversarial networks to craft poisoning points efficiently for larger neural networks, but with limited effectiveness (Yang et al., 2017; Muñoz-González et al., 2019). The strongest attack so far is the min-max attack (Steinhardt et al., 2017), which only works for the indiscriminate attack setting but additionally provides a certificate on worst-case test loss for a fixed number of poisoning points. Our adoption of online convex optimization to instantiate our model-targeted attacks is inspired by Steinhardt et al. (2017), but applies it to a more general attack scenario. Our approach is also related to (Wang & Chaudhuri, 2018)'s poisoning attack against online learning, which considers a setting where training data arrives in a streaming manner. In contrast, we consider the offline setting with training data being fixed.

Another kind of poisoning attack objective is when an adversary guarantees an increase in the probability of an arbitrary "bad" property that otherwise would have some non-negligible chance of occurring naturally (Mahloujifar et al., 2019a; 2018; 2019b). These attacks cannot be applied in the model-targeted setting, though, since there is no known way to construct a target model with the desired property.

Model-Targeted Attacks. Mei & Zhu (2015b) first introduced a specific target model into a poisoning attack and then utilized the Karush-Kuhn-Tucker (KKT) conditions (Karush, 1939; Kuhn & Tucker, 1951) to transform the poisoning problem into a tractable form. However, their attack is still uses gradient-based local optimization techniques and suffers from bad local optima (Steinhardt et al., 2017; Koh et al., 2018). Koh et al. (2018) proposed the KKT attack with a target model, which converts the complicated bi-level optimization into a simple convex optimization problem utilizing the KKT conditions and the Carathéodory number of the set of scaled gradients, avoiding the local optima issues. However, their attack is limited to margin-based losses, cannot scale to multi-class classification, and does not provide a guarantee on the number of poisoning points required to converge to the target classifier. Additionally, these two attacks require knowing the number of poisoning points before running the attack, which is often impractical.

There are also model-targeted attacks proposed in different settings. Zhang et al. (2020) presented a poisoning attack against online learning which assumes the training data arrives sequentially. Ma et al. (2019) proposed poisoning attacks that work for objective-driven or model-targeted settings. Related to our Theorem 2, they also derived a lower bound on the number of poisoning points (needed to induce a target model). However, their attacks and lower bound only apply when differential privacy is deployed during the model training process and hence compromises the model utility.

4. Poisoning Attack with a Target Model

Our new poisoning attack determines a target model and selects poisoning points to achieve that target model. The target model generation is not our focus and we adopt the heuristic approach proposed by Koh et al. (2018). For the new poisoning attack, we first show how the algorithm generates the poisoning points (Section 4.1). Then, we prove that the generated poisoning points, once added to the clean data, can produce a classifier that asymptotically converges to the target classifier (Section 4.2).

4.1. Model-Targeted Poisoning with Online Learning

The main idea of our attack, shown in Algorithm 1, is to sequentially add the poisoning point that has the maximum loss difference between the intermediate model obtained so far (that is, the model induced by training on the clean data and generated poisoning points from previous iterations) and the target model. By repeating this process, we can gradually minimize the maximum loss difference between the induced intermediate classifier and the target classifier, eventually inducing a classifier that has a similar loss distribution as the target classifier. We show in Section 4.2 why similar loss distribution implies convergence.

Algorithm 1 ModelTargetedPoisoning

Input: $\mathcal{D}_c$, the loss functions (L and l), target model θ_p, regularization strength C_R

Output: $\mathcal{D}_p$

```
1: $\mathcal{D}_p = \emptyset$
2: while stopping criteria not met do
3:     $\theta_t = \arg\min \frac{1}{|\mathcal{D}_c \cup \mathcal{D}_p|} L(\theta; \mathcal{D}_c \cup \mathcal{D}_p) + C_R \cdot R(\theta)$
4:     $(x^*, y^*) = \arg\max_{\mathcal{X}\times\mathcal{Y}} l(\theta_t; x, y) - l(\theta_p; x, y)$
5:     $\mathcal{D}_p = \mathcal{D}_p \cup \{(x^*, y^*)\}$
6: end while
   return $\mathcal{D}_p$
```

Algorithm 1 requires the input of clean training set $\mathcal{D}_c$, the Loss function (L for a set of points and l for individual point), and the target model θ_p. The output from Algorithm 1 will be the set of poisoning points $\mathcal{D}_p$. The algorithm is simple: first, adversaries train the intermediate model θ_t on the mixture of clean and poisoning points $\mathcal{D}_c \cup \mathcal{D}_p$ with $\mathcal{D}_p$ an empty set in the first iteration (Line 3). Then, it searches for the point that maximizes the loss difference between θ_t and θ_p (Line 4). After the point of maximum loss difference is found, it is added to the poisoning set $\mathcal{D}_p$ (Line 5). The whole process repeats until the stopping condition is satisfied (Line 2). The stopping condition is flexible and it can take various forms: 1) adversary has a budget T on the number of poisoning points, and the algorithm halts when the algorithm runs for T iterations; 2) the intermediate classifier θ_t is closer to the target classifier (than a preset threshold ϵ) in terms of the maximum loss difference, and more details regarding this distance metric will be introduced in Section 4.2; 3) adversary has some requirement on the accuracy and the algorithm terminates when θ_t satisfies the accuracy requirement. Since we focus on producing a classifier close to the target model, we adopt the second stop criterion that measures the distance with respect to the maximum loss difference, and report results based on this criterion in Section 5. Algorithm 1 selects poisoning points in $\mathcal{D}_p$ sequentially in order. However, we assume the adversary has no control over the training order, so when the victim trains a model on $\mathcal{D}_c \cup \mathcal{D}_p$ the training set is shuffled randomly.

A nice property of Algorithm 1 is that the classifier θ_{atk} trained on $\mathcal{D}_c \cup \mathcal{D}_p$ is close to the target model θ_p and asymptotically converges to θ_p. Details of the convergence will be shown in the next section. The algorithm may appear to be slow, particularly for larger models due to the requirement of repeatedly training a model in line 3. However, this is not an issue. First, as will be shown in the next section, the algorithm is an online optimization process and line 3 corresponds to solving the online optimization problem exactly. However, people often use the very efficient online gradient descent method to approximately solve the problem and its asymptotic performance is the same (Shalev-Shwartz, 2012). Second, if we solve the optimization problem exactly, we can add multiple copies of (x^*, y^*) into $\mathcal{D}_p$ each time. This reduces the overall iteration number, and hence reduces the number of times retraining models. The proof of convergence will be similar. For simplicity in interpreting the results, we do not use this in our experiments and add only one copy of (x^*, y^*) each iteration. However, we also tested the performance by adding two copies of (x^*, y^*) and find that the attack results are nearly the same while the efficiency is improved significantly. For example, for experiments on MNIST 1–7 dataset, by adding 2 copies of points, with the same number of poisoning points, the attack success rate decreases at most by 0.7% while the execution time is reduced approximately by half.

4.2. Convergence of Our Poisoning Attack

Before proving the convergence of Algorithm 1, we first formally define the attainable models in Definition 1. Then, we define a general closeness measure based on their prediction performance to measure the distance of the model θ_{atk} trained on $\mathcal{D}_c \cup \mathcal{D}_p$ to the target model θ_p. Both definitions will be used to state the convergence theorem in Theorem 1.

Definition 1 (Attainable models)**.** We say θ is C_R-attainable with respect to loss function l and regularization function R if there exists a training set $\mathcal{D}$ such that

$$\theta = \arg\min_{\theta\in\Theta} \frac{1}{|D|} \cdot L(\theta; \mathcal{D}) + C_R \cdot R(\theta)$$

and θ is the unique minimizer for the above.

Definition 2 (Loss-based distance and ϵ-close)**.** For two models θ_1 and θ_2, a space $\mathcal{X} \times \mathcal{Y}$ and a loss $l(\theta; x, y)$, we define *loss-based distance* $D_{l,\mathcal{X},\mathcal{Y}} : \Theta \times \Theta \to R$ as

$$D_{l,\mathcal{X},\mathcal{Y}}(\theta_1, \theta_2) = \max_{(x,y)\in\mathcal{X}\times\mathcal{Y}} l(\theta_1; x, y) - l(\theta_2; x, y),$$

and we say model θ_1 is *ϵ-close* to model θ_2 when the loss-based distance from θ_1 to θ_2 is upper bounded by ϵ.

Measuring model distance We use loss-based distance to capture the "behavioral" distance between two models. Namely, if θ_1 is ϵ-close (as measured by loss-based distance) to θ_2 and vice versa, then θ_1 and θ_2 would have an almost equal loss on all the points, meaning that they have almost the same behavior across all the space. Note that our general definition of loss-based distance does not have the symmetry property of metrics and hence is not a metric. However, it has some other properties of metrics in the space of attainable models. For example, if some model θ is attainable using ERM, no model could have a negative distance to it. To further show the value of this distance notion, in Appendix B we demonstrate an $O(\epsilon)$ upper bound on the ℓ_1-norm of difference between two models that are ϵ-close with respect to loss-based distance for the special case of Hinge loss. For Hinge loss, it also satisfies the *bi-directional closeness*, that is if θ_1 is ϵ-close to θ_2, then θ_2 is $O(\epsilon)$-close to θ (details can be found in Corollary 3), and the proof details can be found in Appendix B. In the rest of the paper, we will use the terms ϵ-close or ϵ-closeness to denote that a model is ϵ away from another model based on the loss-based distance. Our convergence theorem uses the loss-based distance to establish that the attack of Algorithm 1 produces model that converges to the target classifier:

Theorem 1. For $\delta > 0$, if θ_p is a $C_R(1 + \delta)$-attainable model, after at most T steps, Algorithm 1 will produce the poisoning set $\mathcal{D}_p$ so that a classifier trained on $\mathcal{D}_c \cup \mathcal{D}_p$ using Eq. (1) is ϵ-close to θ_p, with respect to loss-based distance, $D_{l,\mathcal{X},\mathcal{Y}}$, for

$$\epsilon = \frac{\alpha(T) + L(\theta_p; D_c) - L(\theta_c; D_c)}{T \cdot {}^{\delta}\!/_{1+\delta}}$$

where $\alpha(T)$ is the regret of the follow-the-leader algorithm for a series of loss functions $\ell_i(\cdot) = \ell(\cdot, x_i, y_i) + C_R \cdot R(\cdot)$ and (x_i, y_i) is the ith poisoning point.

Remark 1. Sublinear regret bounds for follow-the-leader can be applied to show the convergence. Here, we adopt the regret analysis from McMahan (2017). Specifically, $\alpha(T)$ is in the order of $O(\log T)$) and we have $\epsilon \le O(\frac{\log T}{T})$ when the loss function is Lipschitz continuous and the regularizer $R(\theta)$ is strongly convex, and $\epsilon \to 0$ when $T \to +\infty$. $\alpha(T)$ is also in the order of $O(\log T)$ when the loss function used for training is strongly convex and the regularizer is convex. Strong convexity is critical for the convergence of our attack since the attack may not converge in the general setting of convex loss functions without a strongly convex regularizer. In the general case, the loss function can be "unstable" across iterations, and the learned model weights in neighboring iterations can change drastically.

Proof idea. The full proof of Theorem 1 is in Appendix A. Here, we only summarize the high-level proof idea. The key idea is to frame the poisoning problem as an online learning problem. In this formulation, each step of the online learning problem corresponds to the i^{th} poison point (x_i, y_i). In particular, the loss function at iteration i of the online learning problem is set to $l(\cdot; x_i, y_i)$. Then, we show that by defining the parameters of the online learning problem carefully, the output of the follow-the-leader (FTL) algorithm (Shalev-Shwartz, 2012) at iteration i is a model that is identical to training a model on a dataset consisting of the clean points and the first $i - 1$ poisoning points. On the other hand, the way the poisoning points are selected, we can show that at the i^{th} iteration the maximum loss difference between the target model and the best induced model so far would be smaller than the regret of the FTL algorithm divided by the number of poisoning points. The convergence bound of Theorem 1 boils down to regret analysis of the FTL algorithm based on the loss function. Since we are assuming the loss function is convex with a strongly convex regularizer (or a strongly convex loss function with a convex regularizer), we can show that the regret is bounded by $O(\log T)$ and hence the loss distance between the induced model and the target model converges to 0.

Implications of Theorem 1 The theorem says that the loss-based distance of the model trained on $\mathcal{D}_c \cup \mathcal{D}_p$ to the target model correlates to the loss difference between the target model and the clean model θ_c (trained on $\mathcal{D}_c$) on $\mathcal{D}_c$, and correlates inversely with the number of poisoning points. Therefore, it implies 1) if the target classifier θ_p has a lower loss on $\mathcal{D}_c$, then it is easier to achieve the target model, and 2) with more poisoning points, we get closer to the target classifier and our attack will be more effective. The theorem also justifies the motivation behind the heuristic method in Koh et al. (2018) to select a target classifier with a lower loss on clean data. For the indiscriminate attack scenario, we also improve the heuristic approach by adaptively updating the model and producing target classifiers with a much lower loss on the clean set. This helps to empirically validate our theorem. Details of the original and improved heuristic approach and relevant experiments are in Appendix C.3.

4.3. Lower Bound on the Number of Poisoning Points

We first provide the lower bound on the number of poisoning points required for producing the target classifier in the addition-only setting (Theorem 2) and then explain how the lower bound estimation can be incorporated into Algorithm 1. The intuition behind the theorem below is, when the number of poisoning points added to the clean training set is smaller than the lower bound, there always exists a classifier θ with lower loss compared to θ_p and hence the target classifier cannot be attained. The full proof of the theorem is in Appendix A.

Theorem 2 (Lower Bound)**.** Given a target classifier θ_p, to reproduce θ_p by adding the poisoning set $\mathcal{D}_p$ into $\mathcal{D}_c$, the number of poisoning points $|\mathcal{D}_p|$ cannot be lower than

$$\sup_\theta z(\theta) = \frac{L(\theta_p; \mathcal{D}_c) - L(\theta; \mathcal{D}_c) + NC_R(R(\theta_p) - R(\theta))}{\sup_{x,y}\big(l(\theta; x, y) - l(\theta_p; x, y)\big) + C_R(R(\theta) - R(\theta_p))}.$$

Corollary 1. If we further assume bi-directional closeness in the loss-based distance, we can also derive the lower bound on number of poisoning points needed to induce models that are ϵ-close to the target model. More precisely, if θ_1 being ϵ-close to θ_2 implies that θ_2 is also $k \cdot \epsilon$ close to θ_1, then we have,

$$\sup_\theta z'(\theta) = \frac{L(\theta_p; \mathcal{D}_c) - L(\theta; \mathcal{D}_c) - NC_R \cdot R^* - Nk\epsilon}{\sup_{x,y}\big(l(\theta; x, y) - l(\theta_p; x, y)\big) + C_R \cdot R^* + k\epsilon}.$$

where R^* is an upper bound on the regularizer $R(\theta)$.

The formula for the lower bound in Theorem 2 (and also the lower bound in Corollary 1) can be easily incorporated into Algorithm 1 to obtain a tighter theoretical lower bound. We simply need to check all of the intermediate classifiers θ_t produced during the attack process and replace θ with θ_t, and the lower bound can be computed for the pair of θ_t and θ_p. Algorithm 1 then additionally returns the lower bound, which is the highest lower bound computed from our poisoning procedure. When the loss function is Lipschitz continuous and the data and model space are closed convex sets (all common in practice), the loss function is bounded and the returned lower bound will often be nonzero. We empirically show this for linear SVM models in Table 1 and Table 2. We note that, for unbounded loss difference, the lower bounds of Theorem 2 and Corollary 1 will be 0. But, this doesn't mean that our results are vacuous—it means the attacker is very powerful and our attack will converge with only a few poisoning points (possibly even just one) and the lower bound of 0 is close to the number of poisoning points used by the adversary.

5. Experiments

We present the experimental results by showing the convergence of Algorithm 1, the comparison of attack success rates to state-of-the-art model-targeted poisoning attack, and the theoretical lower bound for inducing a given target classifier and its gap to the number of poisoning points used by our attack. All of our evaluation code is available at: https://github.com/suyeecav/model-targeted-poisoning.

Datasets and Subpopulations. We experiment on both the practical subpopulation and the conventional indiscriminate attack scenarios. We selected datasets and models for our experiments based on evaluations of previous poisoning attacks (Biggio et al., 2012; Mei & Zhu, 2015a; Koh et al., 2018; Steinhardt et al., 2017; Koh & Liang, 2017; Jagielski et al., 2019). For the subpopulation attack experiments, we use the Adult dataset (Dua & Graff, 2017), which was used for evaluation by (Jagielski et al., 2019). We downsampled the Adult dataset to make it class-balanced and ended up with 15,682 training and 7,692 test examples. Each example has the dimension of 57 after one-hot encoding the categorical attributes. For the indiscriminate setting, we use the Dogfish (Koh & Liang, 2017) and MNIST 1–7 datasets (LeCun, 1998)[1]. The Dogfish dataset contains 1,800 training and 600 test samples. We use the same Inception-v3 features (Szegedy et al., 2016) as in Koh & Liang (2017); Steinhardt et al. (2017); Koh et al. (2018) and each image is represented by a 2,048-dimensional vector. The MNIST 1–7 dataset contains 13,007 training and 2,163 test samples, and each image is flattened to a 784-dimensional vector.

We identify the subpopulations for the Adult dataset using k-means clustering techniques (ClusterMatch in Jagielski et al. (2019)) to obtain different clusters ($k = 20$). For each cluster, we select instances with the label "$\leq$ 50K" to form the subpopulation (indicating all instances in the subpopulation are in the low-income group). This way of defining subpopulation is rather arbitrary (in contrast to a more likely attack goal that would select subpopulations based on demographic characteristics), but enables us to simplify analyses. From the 20 subpopulations obtained, we select three subpopulations with the highest test accuracy on the clean model. They all have 100% test accuracy, indicating all instances in these subpopulations are correctly classified as low income. This enables us to use "attack success rate" and "accuracy" without any ambiguity on the subpopulation—for each of our subpopulations, all instances are originally classified as low income, and the simulated attacker's goal is to have them classified as high income.

Models and Attacks. We conduct experiments on linear SVM and logistic regression (LR) models. Although our theoretical results do not apply to non-convex models, for curiosity we also tested our attack on deep neural networks and report results in Appendix E.

[1]MNIST 1–7 dataset is a subset of the well-known MNIST dataset that only contains digit 1 and 7.

We use the heuristic approach from Koh et al. (2018) to generate target classifiers for both attack settings. In the subpopulation setting, for each subpopulation, we generate a target model that has 0% accuracy (100% attacker success) on the subpopulation, indicating that all subpopulation instances are now classified as high income. In the indiscriminate setting, for MNIST 1–7, we aim to generate three target classifiers with overall test errors of 5%, 10%, and 15%. For SVM, we obtained target models of test accuracies of 94.0%, 88.8%, and 83.3%, and for LR, the target models are of test accuracies of 94.7%, 89.0%, and 84.5%. For Dogfish, we aim to generate target models with overall test errors of 10%, 20%, and 30%. For SVM, we obtained target models of test accuracies of 89.3%, 78.3%, and 67.2% and for logistic regression, we obtained target models of test accuracies of 89.0% 79.5%, and 67.3%. The test accuracy of the clean SVM model is 78.5% on Adult, 98.9% on MNIST 1–7 and is 98.5% on Dogfish. The test accuracy of clean LR model is 79.9% on Adult, 99.1% on MNIST 1–7 and 98.5% on Dogfish.

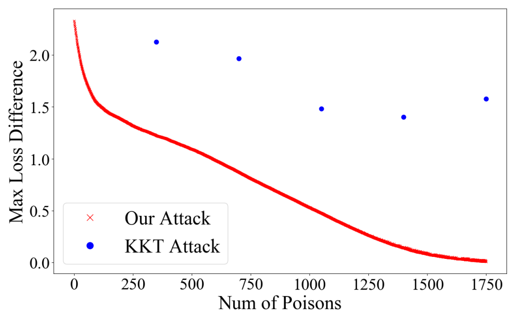
We compare our model-targeted poisoning attack in Algorithm 1 to the state-of-the-art KKT attack (Koh et al., 2018). We omit the model-targeted attack from Mei & Zhu (2015b) because there is no open source implementation and this attack is also reported to underperform the KKT attack (Koh et al., 2018). Our main focus here is to compare with other model-targeted attacks in terms of achieving the target models, but we also do include experiments (in Appendix D) comparing our attack to existing objective-driven attacks, where the target model for our attack is selected to achieve that objective. With a carefully selected target model, our attack can also outperform the state-of-the-art objective-driven attack.

Both our attack and the KKT attack take as input a target model and the original training data, and output a set of poisoning points intended to induce a model as close as possible to the target model when the poisoning points are added to the original training data. We compare the effectiveness of the attacks by testing them using the same target model and measuring convergence of their induced models to the target model.

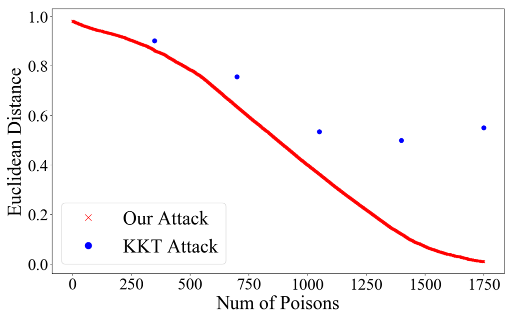
The KKT attack requires the number of poisoning points as an input, while our attack is more flexible and can produce poisoning points in priority order without a preset number. As a stopping condition for our experiments, we use either a target number of poisoning points or a threshold for ϵ-close distance to the target model. Since we do not know the number of poisoning points needed to reach some attacker goal in advance for the KKT attack, we first run our attack and produce a classifier that satisfies the selected ϵ-close distance threshold. The loss function is hinge loss for SVM and logistic loss for LR. For SVM model, we set ϵ as 0.01 on Adult, 0.1 on MNIST 1–7 and 2.0 on Dogfish dataset. For LR model, we set ϵ as 0.05 on Adult, 0.1 on MNIST 1–7 and 1.0 on Dogfish. Then, we use the size of the poisoning set returned from our attack (denoted by n_p) as the input to the KKT attack for the target number of poisons needed. We also compare the two attacks with varying numbers of poisoning points up to n_p. For the KKT attack, its entire optimization process must be rerun whenever the target number of poisoning points changes. Hence, it is infeasible to evaluate the KKT attack on many different poisoning set sizes. In our experiments, we run the KKT attack five poisoning set sizes: $0.2 \cdot n_p$, $0.4 \cdot n_p$, $0.6 \cdot n_p$, $0.8 \cdot n_p$, and n_p. For our attack, we simply run iterations up to the maximum number of poisoning points, collecting a data point for each iteration up to n_p. In Appendix C, we also plot the performance of our attack with respect to the number of poisoning points added across iterations. Generation of poisoning points can be done offline, so we do not focus on the computational cost of different poisoning attacks here. The two attacks we consider are both very efficient, and all of our experiments can be run on a typical laptop.

Convergence. Figure 1 shows the convergence of Algorithm 1 using both maximum loss difference and Euclidean distance to the target, and the result is reported on the first subpopulation (Cluster 0) of Adult and the model is SVM. The maximum number of poisoning points (n_p) for the experiments is obtained when the classifier from Algorithm 1 is 0.01-close to the target classifier. Our attack steadily reduces the maximum loss difference and Euclidean distance to the target model, in contrast to the KKT attack which does not seem to converge towards the target model reliably. Concretely, at the maximum number of poisons in Figure 1, both the maximum loss difference and Euclidean distance of our attack (to the target) are less than 2% of the corresponding distances of the KKT attack. Similar observations are also observed for the indiscriminate and other subpopulation attack settings, see Appendix C.

We believe our attack outperforms the KKT attack in convergence to the target model because it approaches the target classifier differently. The foundation of the KKT attack is that for binary classification, for any target classifier generated by training on a set $D_c \cup D_p$ with $|D_p| = n$, the (exact) same classifier can also be obtained by training on set $D_c \cup D'_p$ with $|D'_p| \leq n$. This poisoning set D'_p only contains two distinct points, one from each class. In practice, the KKT attack often aims to induce the exact same classifier with much fewer poisoning points, which may not be feasible and leads the KKT attack to fail. In contrast, our attack does not try to obtain the exact target model but just selects each poisoning point in turn as the one with the best expected impact. Hence, our attack gets close to the target model with fewer poisoning points than the number



(a) Max Loss Difference



(b) Euclidean Distance

Figure 1. Convergence to the target model. Results shown are for the first subpopulation (Cluster 0) and the model is SVM.

Model/ Dataset	Target Model	n_p	Lower Bound	$0.2n_p$ KKT	Ours	$0.4n_p$ KKT	Ours	$0.6n_p$ KKT	Ours	$0.8n_p$ KKT	Ours	n_p KKT	Ours
SVM/ Adult	Cluster 0	1,866	1,667±2.4	96.8	98.4	65.4	**51.6**	14.9	35.6	1.1	2.7	15.4	**0.5**
	Cluster 1	2,097	1,831.4±5.0	72.2	77.1	41.0	**23.6**	2.8	**0.7**	1.4	**0.7**	6.9	**0.0**
	Cluster 2	2,163.3±2.5	1,863.0±9.2	94.9 ± 0.7	**24.3± 0.3**	15.9	20.3	34.3 ± 0.2	**12.1**	21.6 ± 0.1	**0.3**	20.3 ± 0.7	**0.3**
LR/ Adult	Cluster 0	2,005	N/A	82.1 ± 1.0	**75.6**	71.7 ± 0.4	**42.2**	46.7	**15.9**	36.6 ± 2.1	**1.9**	24.3 ± 0.8	**0.4**
	Cluster 1	1,630±1.1	N/A	98.1	**94.9**	97.2	**79.0**	96.7	**34.1**	95.8	**6.1**	95.8	**0.5**
	Cluster 2	2,428	N/A	97.9	**94.5**	93.9	**45.8**	89.8 ± 0.7	**5.8**	79.2 ± 4.6	**0.6**	60.2 ± 1.7	**0.6**

Table 1. Subpopulation attack on Adult: comparison of test accuracies on subpopulations (%). Target models for Adult dataset consist of models with 0% accuracy on the selected subpopulations (Cluster 0 - Cluster 2). n_p denotes the maximum number of poisoning points used by our attack, and xn_p denotes comparing the two attacks at xn_p poisoning points. n_p is set by running our attack till the induced model becomes 0.01-close to the target model. All results are averaged over 4 runs and standard error is 0 (exact same number of misclassifications across the runs), except where reported. We do not show lower bound for LR because we can only compute an approximate maximum loss difference and the lower bound will no longer be valid.

Model/ Dataset	Target Model	n_p	Lower Bound	$0.2n_p$ KKT	Ours	$0.4n_p$ KKT	Ours	$0.6n_p$ KKT	Ours	$0.8n_p$ KKT	Ours	n_p KKT	Ours
SVM/ MNIST 1-7	5% Error	1,737	874	97.3	**97.1**	96.4	**96.1**	95.7	95.7	94.9	94.9	94.3	94.6
	10% Error	5,458	3,850.4±0.8	95.8	**95.5**	93.4	**92.1**	92.7	**90.9**	91.1	**90.7**	90.2	90.2
	15% Error	6,192	4,904	98.3	**97.8**	96.3	98.1	97.2	97.3	98.3	**92.7**	82.7	85.9
SVM/ Dogfish	10% Error	32	15	97.0	**95.8**	94.0	**93.3**	92.2	**91.2**	90.7	**90.2**	90.3	**89.8**
	20% Error	89	45	95.5	95.7	92.5	**92.2**	90.3	**88.7**	84.7	84.7	82.3	**82.0**
	30% Error	169	83	95.5	95.7	93.5	**90.7**	88.3	**82.5**	78.3	**75.2**	71.8	**71.7**
LR/ MNIST 1-7	5% Error	756	N/A	97.5	**96.9**	97.4	**96.5**	97.2	**96.0**	96.9	**95.7**	96.9	**95.2**
	10% Error	2,113	N/A	97.0	**95.7**	96.9	**93.8**	96.8	**92.3**	96.2	**91.4**	96.4	**90.4**
	15% Error	3,907	N/A	96.9	**95.4**	97.0	**93.3**	97.1	**90.7**	97.1	**88.3**	97.1	**87.1**
LR/ Dogfish	10% Error	62	N/A	98.8	**93.0**	98.5 ± 0.1	**89.7 ± 0.3**	98.8	**89.2**	98.8	**89.2**	98.8	**89.0**
	20% Error	120	N/A	98.5	**93.2**	99.2	**88.2**	99.3	**85.3**	99.5	**83.0**	99.5	**80.7**
	30% Error	181	N/A	97.8	**92.3**	98.8	**85.7 ± 0.2**	99.2	**81.3 ± 0.3**	99.5	**75.7**	99.5	**72.5 ± 0.2**

Table 2. Indiscriminate attack on MNIST 1–7 and Dogfish: comparison of overall test accuracies (%). The target models are of certain overall test errors. n_p is set by running our attack till the induced model becomes ϵ-close to the target model and we set ϵ as 0.1 for MNIST 1–7 and 2.0 for Dogfish dataset. All results are averaged over 4 runs and standard error is 0 (exact same number of misclassifications across the runs), except where reported.

of points used to exactly produce the target model.

Attack Success. Next, we compare the classifiers induced by the two attacks in terms of the attacker's goal. Table 1 summarize the results of the subpopulation attacks, where attack success is measured on the targeted cluster. At the maximum number of poisons, our attack is much more successful than the KKT attack, for both the SVM and LR models. For example, on Cluster 1 with LR, the induced classifier from our attack has 0.5% accuracy compared to the 95.8% accuracy of KKT. Table 2 shows the results of indiscriminate attacks on MNIST 1–7 and Dogfish, and the attack success is the overall test error. For the indiscriminate attack on SVM, both on MNIST 1–7 and Dogfish, the two attacks have similar performance while for LR, our attack is much better than the KKT attack. KKT's failure on LR is that its objective function becomes highly non-convex and is very hard to optimize. More details about the formulation can be found in Koh et al. (2018). For logistic loss, our attack also needs to maximize a non-concave maximum loss difference[2] (Step 4 in Algorithm 1). However, this objective is much easier to optimize than that of the KKT attack and our attack is still very effective on LR models.

Optimality of Our Attack. To check the optimality of our attack, we calculate a lower bound on the number of poisoning points needed to induce the model induced by the poisoning points found by our attack. We calculate this lower bound on the number of poisons using Theorem 2 (details in Section 4.3). Note that Theorem 2 provides a valid lower bound based on any intermediate model. To get a lower bound on the number of poisoning points, we only need to use Theorem 2 on the encountered intermediate models and report the best one. We do this by running Algorithm 1 using the induced model (and not the previous target model) as the target model, terminating when the induced classifier is ϵ-close to the given target model. Note that for LR, maximizing the loss difference is not concave and therefore, we cannot obtain the actual maximum loss difference, which is required in the denominator in Theorem 2. Therefore, we only report results on SVM. For the subpopulation attack on Adult, we set $\epsilon = 0.01$ and for the indiscriminate attack on MNIST 1–7 and Dogfish, we set ϵ to 0.1 and 2.0, respectively. We then consider all the intermediate classifiers that the algorithm induced across the iterations. Our calculated lower bound in Table 1 (Column 3-4) shows that for the Adult dataset, the gap between the lower bound and the number of used poisoning points is relatively small. This means our attack is nearly optimal in terms of minimizing the number of poisoning points needed to induce the target classifier. However, for the MNIST 1–7 and Dogfish datasets in Table 2, there still exists some gap between the lower bound and the number of poisoning points used by our attack, indicating there might exist more efficient model-targeted poisoning attacks.

6. Limitations

Our theoretical analysis only applies to convex loss functions and it is an interesting future direction to extend our theoretical analysis into non-convex loss functions (e.g., DNNs). The efficient search of the point with maximum loss difference requires the difference of the loss function to be concave or in some easily optimizable forms where the global optima can be easily found. This excludes some classes of common loss functions such as the logistic loss, but empirically, we find that gradient based optimization techniques can still solve the optimization problem well and our attack is very effective in these cases (see the results on LR models). Our work mainly focuses on inducing given target models and does not provide a principled way to generate target classifiers such that the attacker objectives can be achieved efficiently using our attack. In Appendix D, we show some heuristics in selecting better target models, but a systematic investigation on the generation of better target classifiers is the important next step.

We have not considered defenses in this work, and it is an important and interesting direction to study the effectiveness of our attack against data poisoning defenses. Defenses may be designed to limit the search space of the points with maximum loss difference and hence increase the number of poisoning points needed. We also leave the investigation of the application of our model-targeted attacks in other attacker objectives, e.g., backdoor attacks and privacy attacks, for future work.

[2]We use Adam optimizer (Kingma & Ba, 2014) with random restarts to solve this maximization problem approximately.

7. Conclusion

We propose a general poisoning framework with provable guarantees to approach any attainable target classifier, along with a lower (Theorem 2) and upper bound (the proposed poisoning attack) on the number of poisoning points needed. Our attack is a generic tool that first captures the adversary's goal as a target model and then focuses on the power of attacks to induce that model. This separation enables future work to explore the effectiveness of poisoning attacks corresponding to different adversarial goals.

Acknowledgements

This work was partially funded by awards from the National Science Foundation (NSF) SaTC program (Center for Trustworthy Machine Learning, #1804603) and NSF grants #1943100, #2002985 and #1823325.

References

Biggio, B., Nelson, B., and Laskov, P. Support Vector Machines Under Adversarial Label Noise. In *Asian Conference on Machine Learning*, 2011.

Biggio, B., Nelson, B., and Laskov, P. Poisoning Attacks against Support Vector Machines. In *International Conference on Machine Learning*, 2012.

Dacrema, M. F., Cremonesi, P., and Jannach, D. Are We Really Making Much Progress? A Worrying Analysis of

Recent Neural Recommendation Approaches. In *ACM Conference on Recommender Systems*, 2019.

Demontis, A., Melis, M., Pintor, M., Jagielski, M., Biggio, B., Oprea, A., Nita-Rotaru, C., and Roli, F. Why Do Adversarial Attacks Transfer? Explaining Transferability of Evasion and Poisoning Attacks. In *USENIX Security Symposium*, 2019.

Diamond, S. and Boyd, S. CVXPY: A Python-Embedded Modeling Language for Convex Optimization. *The Journal of Machine Learning Research*, 17(1):2909–2913, 2016.

Dua, D. and Graff, C. UCI Machine Learning Repository, 2017. URL https://archive.ics.uci.edu/ml.

Geiping, J., Fowl, L., Huang, W. R., Czaja, W., Taylor, G., Moeller, M., and Goldstein, T. Witches' Brew: Industrial Scale Data Poisoning via Gradient Matching. In *International Conference on Learning Representations*, 2021.

Gurobi Optimization, Inc. Gurobi optimizer reference manual, 2020. URL https://www.gurobi.com/documentation/9.1/refman/index.html.

Hong, S., Chandrasekaran, V., Kaya, Y., Dumitraş, T., and Papernot, N. On the Effectiveness of Mitigating Data Poisoning Attacks with Gradient Shaping. *arXiv preprint arXiv:2002.11497*, 2020.

Huang, L., Joseph, A. D., Nelson, B., Rubinstein, B. I., and Tygar, J. D. Adversarial Machine Learning. In *4th ACM Workshop on Security and Artificial Intelligence*, 2011.

Huang, W. R., Geiping, J., Fowl, L., Taylor, G., and Goldstein, T. MetaPoison: Practical General-purpose Clean-label Data Poisoning. In *Advances in Neural Information Processing Systems*, 2020.

Jagielski, M., Oprea, A., Biggio, B., Liu, C., Nita-Rotaru, C., and Li, B. Manipulating Machine Learning: Poisoning Attacks and Countermeasures for Regression Learning. In *IEEE Symposium on Security and Privacy*, 2018.

Jagielski, M., Hand, P., and Oprea, A. Subpopulation Data Poisoning Attacks. In *NeurIPS 2019 Workshop on Robust AI in Financial Services*, 2019.

Karush, W. Minima of functions of several variables with inequalities as side conditions. Master's thesis, University of Chicago, 1939.

Kingma, D. P. and Ba, J. ADAM: A Method for Stochastic Optimization. In *International Conference on Learning Representations*, 2014.

Koh, P. W. and Liang, P. Understanding Black-box Predictions via Influence Functions. In *International Conference on Machine Learning*, 2017.

Koh, P. W., Steinhardt, J., and Liang, P. Stronger Data Poisoning Attacks Break Data Sanitization Defenses. *arXiv preprint arXiv:1811.00741*, 2018.

Kuhn, H. W. and Tucker, A. W. Nonlinear programming. In *Second Berkeley Symposium on Mathematical Statistics and Probability*, 1951.

LeCun, Y. The MNIST database of handwritten digits, 1998.

Li, B., Wang, Y., Singh, A., and Vorobeychik, Y. Data Poisoning Attacks on Factorization-Based Collaborative Filtering. In *Advances in Neural Information Processing Systems*, 2016.

Ma, Y., Zhu, X., and Hsu, J. Data Poisoning against Differentially-Private Learners: Attacks and Defenses. *International Joint Conference on Artificial Intelligence*, 2019.

Mahloujifar, S., Diochnos, D. I., and Mahmoody, M. Learning under p-Tampering Attacks. In *Algorithmic Learning Theory*, 2018.

Mahloujifar, S., Diochnos, D. I., and Mahmoody, M. The Curse of Concentration in Robust Learning: Evasion and Poisoning Attacks from Concentration of Measure. In *AAAI Conference on Artificial Intelligence*, 2019a.

Mahloujifar, S., Mahmoody, M., and Mohammed, A. Universal Multi-Party Poisoning Attacks. In *International Conference on Machine Learning*, 2019b.

McMahan, H. B. A Survey of Algorithms and Analysis for Adaptive Online Learning. *The Journal of Machine Learning Research*, 18(1):3117–3166, 2017.

Mei, S. and Zhu, X. The Security of Latent Dirichlet Allocation. In *Artificial Intelligence and Statistics*, 2015a.

Mei, S. and Zhu, X. Using Machine Teaching to Identify Optimal Training-Set Attacks on Machine Learners. In *AAAI Conference on Artificial Intelligence*, 2015b.

Muñoz-González, L., Pfitzner, B., Russo, M., Carnerero-Cano, J., and Lupu, E. C. Poisoning Attacks with Generative Adversarial Nets. *arXiv preprint arXiv:1906.07773*, 2019.

Nelson, B., Barreno, M., Chi, F. J., Joseph, A. D., Rubinstein, B. I., Saini, U., Sutton, C. A., Tygar, J. D., and Xia, K. Exploiting Machine Learning to Subvert Your Spam Filter. In *USENIX Workshop on Large Scale Exploits and Emergent Threats*, 2008.

Shafahi, A., Huang, W. R., Najibi, M., Suciu, O., Studer, C., Dumitras, T., and Goldstein, T. Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks. In *Advances in Neural Information Processing Systems*, 2018.

Shalev-Shwartz, S. Online Learning and Online Convex Optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2012.

Solans, D., Biggio, B., and Castillo, C. Poisoning Attacks on Algorithmic Fairness. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2020.

Steinhardt, J., Koh, P. W. W., and Liang, P. S. Certified Defenses for Data Poisoning Attacks. In *Advances in Neural Information Processing Systems*, 2017.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Tramèr, F. and Boneh, D. Differentially Private Learning Needs Better Features (or Much More Data). *International Conference on Learning Representations*, 2021.

Wang, Y. and Chaudhuri, K. Data Poisoning Attacks against Online Learning. *arXiv preprint arXiv:1808.08994*, 2018.

Xiao, H., Xiao, H., and Eckert, C. Adversarial Label Flips Attack on Support Vector Machines. In *European Conference on Artificial Intelligence*, 2012.

Xiao, H., Biggio, B., Nelson, B., Xiao, H., Eckert, C., and Roli, F. Support Vector Machines under Adversarial Label Contamination. *Neurocomputing*, 160:53–62, 2015.

Yang, C., Wu, Q., Li, H., and Chen, Y. Generative Poisoning Attack Method Against Neural Networks. *arXiv preprint arXiv:1703.01340*, 2017.

Zhang, X., Zhu, X., and Lessard, L. Online data poisoning attacks. In *Conference on Learning for Dynamics and Control*, 2020.

Zhu, C., Huang, W. R., Shafahi, A., Li, H., Taylor, G., Studer, C., and Goldstein, T. Transferable Clean-Label Poisoning Attacks on Deep Neural Nets. In *International Conference on Machine Learning*, 2019.