On the Limits of Data Poisoning Attacks

A Dissertation

Presented to

the Faculty of the School of Engineering and Applied Science

University of Virginia

In Partial Fulfillment

of the requirements for the degree

Doctor of Philosophy (Computer Science)

by

Fnu Suya

Augest 2023

Approval Sheet

This dissertation is submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Computer Science)

Fnu Suya

This dissertation has been read and approved by the Examining Committee:

Mohammad Mahmoody, Committee Chair

David Evans, Advisor

Yuan Tian, Advisor

Farzad Farnoud, Committee Member

Cong Shen, Committee Member

Accepted for the School of Engineering and Applied Science:

Jennifer L. West, Dean, School of Engineering and Applied Science

Augest 2023

Abstract

Current machine learning models require large amounts of labeled training data, which are often collected from untrusted sources. Models trained on these potentially manipulated data points are prone to data poisoning attacks. My research aims to gain a deeper understanding of the limits of two types of data poisoning attacks: indiscriminate poisoning attacks, where the attacker aims to increase the test error on the entire dataset; and subpopulation poisoning attacks, where the attacker aims to increase the test error on a defined subset of the distribution. We first present an empirical poisoning attack that encodes the attack objectives into target models and then generates poisoning points that induce the target models (and hence the encoded objectives) with provable convergence. This attack achieves state-of-the-art performance for a diverse set of attack objectives and quantifies a lower bound on the performance of the best possible poisoning attacks. In the broader sense, because the attack guarantees convergence to the target model which encodes the desired attack objective, our attack can also be applied to objectives related to other trustworthy aspects (e.g., privacy, fairness) of machine learning.

Through experiments for the two types of poisoning attacks we consider, we find that some datasets in the indiscriminate setting and subpopulations in the subpopulation setting are highly vulnerable to poisoning attacks even when the poisoning ratio is low. But other datasets and subpopulations resist the best-performing known attacks even without any defensive protections. Motivated by the drastic differences in the attack effectiveness across datasets or subpopulations, we further investigate the possible factors related to the data distribution and learning algorithm that contribute to the disparate effectiveness of poisoning attacks. In the subpopulation setting, for the given learner, we identify the separability of the class-wise distributions and the loss difference between models that misclassify the subpopulations and the clean models are highly correlated to the empirical performance of state-of-the-art poisoning attacks and demonstrate them through visualizations. In the indiscriminate setting, we conduct a more thorough investigation by first showing under theoretical distributions that there are datasets that inherently resist the best possible poisoning attacks

when the class-wise data distributions are well-separated with low variance and the size of the constraint set containing all permissible poisoning points is also small. We then demonstrate that these identified factors are highly correlated to both the different empirical performances of the state-of-the-art attacks (as lower bounds on the limits of poisoning attacks) and the upper bounds on the limits across benchmark datasets. Finally, we discuss how understanding the limits of poisoning attacks might help in achieving stronger defenses against poisoning attacks in the future.

To my family.

Acknowledgements

Throughout my Ph.D. journey, I am deeply grateful for the invaluable help and support I have received from numerous individuals. I would like to express my sincere appreciation to all of them, as without their contributions, I would not have reached this milestone today.

First and foremost, I would like to extend my deepest gratitude and appreciation to my exceptional research advisors, Dr. David Evans and Dr. Yuan Tian. They have not only been inspiring researchers but also incredible mentors who have supported me both academically and personally. Their guidance, encouragement, and unwavering belief in me during challenging times have been instrumental in my accomplishments. Their dedication to rigorous research has had a profound impact on me, motivating me to continue in academia and aspire to be a mentor who inspires others. I hope my subsequent career will make you proud.

I am also deeply grateful to my Ph.D. committee members, Dr. Mohammad Mahmoody, Dr. Farzad Farnoud, and Dr. Cong Shen, for generously devoting their time and providing valuable insights that have shaped my dissertation work. I extend my special thanks to Dr. Mohammad Mahmoody for his support during my job search process. Additionally, I express my gratitude to Dr. Yangfeng Ji, Dr. Yanjun Qi, Dr. Tianhao Wang, Dr. Jundong Li and Dr. Haifeng Xu for their assistance and advice at various stages of my Ph.D. studies.

Furthermore, I would like to express my appreciation to my exceptional collaborators, who have not only been valuable colleagues but also wonderful friends. Specifically, I would like to thank Dr. Xiao Zhang, Dr. Saeed Mahloujifar, Anshuman Suri, Evan Rose, Dr. Yulong Tian, Scott Hong, Tingwei Zhang, Dr. Jianfeng Chi, Dr. Yi Chen, and Mingming Zha. Engaging in research discussions with each of you has been profoundly inspiring, and I have learned a great deal from our interactions. I would also like to acknowledge and thank my internship mentors, Dr. Ali Torkamani at Amazon, Dr. Aleksei Triastcyn and Dr. Hossein Hosseini at

Qualcomm, and Dr. Anit Kumar Sahu at Bosch, for providing me with enriching experiences in the field of machine learning security.

Moreover, I want to convey my deep appreciation to my girlfriend, Ningjin Wu, for her continuous support and encouragement during my Ph.D. journey. Finally, I would like to recognize my parents and grandmother for their unwavering love, consistent support, and belief in me. Their presence and encouragement, particularly during the most challenging moments of my Ph.D. life, have been pivotal in pursuing my Ph.D. dream.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Machine learning models have achieved remarkable performance in various domains such as computer vision (He et al., 2016; Radford et al., 2021) and natural language processing (OpenAI, 2023) as well as security-critical domains such as the detection of spam email (Dada et al., 2019), network intrusions (Zhang et al., 2022) and malware (Chang and Im, 2020). However, the success of these models often relies on collecting massive amounts of data from unverified sources where there is a risk that some malicious adversaries can inject a small fraction of carefully crafted poisoning samples to manipulate the behavior of the resultant model (known as *poisoned model*) to achieve certain objectives. A typical application is in spam filtering, where the spam detector is trained using data (i.e., emails) that are generated by users with labels often provided implicitly by user actions. In this setting, spammers can generate spam messages that inject benign words likely to occur in spam emails such that models trained on these spam messages will incur significant drops in filtering accuracy as benign and malicious messages become indistinguishable (Nelson et al., 2008; Huang et al., 2011). These kinds of attacks are known as *poisoning attacks*.

Depending on the attack goals, poisoning attacks can be categorized as: *targeted* poisoning attacks (Shafahi et al., 2018; Koh and Liang, 2017), where the adversary's goal is to induce a model that misclassifies a particular known instance; *indiscriminate* poisoning attacks let the induced model incur larger test errors compared to the model trained on the clean dataset (Biggio et al., 2011; Steinhardt et al., 2017); or *subpopulation* poisoning attacks (Jagielski et al., 2021), where the adversary's goal is to produce misclassifications on a defined subset of the distribution. Historically, indiscriminate attacks received more attention from the community and many successful attacks were demonstrated for the linear models (Biggio et al., 2011; 2012;

Xiao et al., 2012; Mei and Zhu, 2015b;a; Steinhardt et al., 2017; Koh et al., 2022) with some recent progress on achieving good performance against deep neural networks (Lu et al., 2022; 2023). Indiscriminate attacks may not represent the stealthiest attack goals as significant drops in overall test accuracy would be detected by the model trainer, who can reject models with abnormally low accuracies.

Compared to indiscriminate attacks, targeted and subpopulation poisoning attacks reflect stealthier attack objectives, and many successful targeted attacks are demonstrated for the deep neural networks (Koh and Liang, 2017; Shafahi et al., 2018; Zhu et al., 2019; Huang et al., 2020; Geiping et al., 2021; Guo and Liu, 2020; Aghakhani et al., 2021) in recent years. Exploration of subpopulation poisoning is rather limited, despite being seemingly more relevant to security threats in practical applications (Jagielski et al., 2021) among the three. In fact, the definition of "subpopulation" is general and can even cover the two extreme goals of targeted (i.e., a subpopulation that only contains a single test instance) and indiscriminate (i.e., a subpopulation that contains the entire dataset) attacks. Existing works on the three attack goals focus on proposing algorithmic poisoning strategies that work well in some settings but do not provide explanations on why the proposed attacks fail in other settings.

In this dissertation, we study subpopulation and indiscriminate poisoning attacks. We investigate subpopulation attacks because they can capture stealthy attack objectives that might be more motivated and also more general than the targeted attacks, as targeted attacks can be viewed as a special form of subpopulation attacks that only contain one point in the subpopulation. We also study the less stealthy indiscriminate attacks because they interfere with the underlying learning algorithms in the broader sense (Steinhardt et al., 2017). Understanding the vulnerabilities of these algorithms in an adversarial environment can provide deeper insights into their fundamental properties.

The goal of this dissertation, through a mix of theoretical and empirical work, is to gain a deeper understanding of the limits of poisoning attacks that inject poisoning points into the clean training set in the considered two types of poisoning attacks. Towards this goal, we first establish a lower bound on the limits of poisoning attacks, especially for the underexplored subpopulation settings, by proposing empirical poisoning strategies that achieve state-of-the-art performance (Chapter 3). Then, driven by the observation that the performance of the best-known attacks varies drastically for different learning tasks in both the subpopulation and indiscriminate settings, we identify properties of the subpopulation (Chapter 4) and the whole data distribution (Chapter 5) under the given learner that contribute to the drastic variance on empirical attack effectiveness across subpopulations in the subpopulation setting and across benchmark datasets in the indiscriminate setting respectively. In the indiscriminate settings in Chapter 5, we also show the relation of the identified factors

to the upper bound on the performance of best possible attacks and compute a non-trivial value for the upper bound. These computed upper bounds, related to the identified distributional properties under the given learner, still vary drastically across different datasets. Finally, we show how understanding the limits of poisoning attacks can help in proposing stronger defenses against data poisoning attacks (Section 5.6).

1.1 Contributions

This dissertation makes the following main contributions:

Model-targeted poisoning attacks. To understand the limits of poisoning attacks, we first provide a (tighter) lower bound on attack effectiveness of the best possible attacks by proposing an empirical poisoning strategy (Chapter 3). In the literature, one way to generate effective poisoning points is to first generate a target model using simple methods such as flipping the labels of existing training points and adding back to the original training data (Koh et al., 2022) and then leveraging optimization strategies to generate poisoning points that aim to induce the generated target model. We follow a similar procedure and propose a model-targeted poisoning (MTP) attack in Section 3.2 that generates the poisoning points in an online manner by leveraging information from the target model. This generation process can be modeled as an online convex optimization and has guaranteed asymptotic convergence to the target model, while the convergence guarantee is missing in prior model-targeted attacks (Koh et al., 2022; Mei and Zhu, 2015a). With a guaranteed convergence to the target model, our attack can be applied for diverse objectives including the considered indiscriminate and subpopulation settings (and even beyond security consequences), by simply generating the corresponding target models. Empirically, our attack achieves state-of-the-art performance in both the subpopulation and indiscriminate settings. Lastly, we also provide an improved target model generation process over the method in Koh et al. (2022) that enables even stronger MTP attacks that achieve similar attack objectives with fewer poisoning points (Section 3.4).

Understanding how subpopulation properties impact susceptibility. Next, in the more realistic subpopulation poisoning settings, we find that given a fixed learner and a dataset, some subpopulations of the dataset are significantly harder to poison than others under our state-of-the-art MTP attack. We then identify distributional and subpopulation properties under the given learner that are related to the different subpopulation susceptibilities by summarizing the observed patterns through extensive experiments on the synthetic and benchmark datasets (Chapter 4). In particular, we find that the overall distributional properties dominate the subpopulation susceptibility when the (sampled) datasets are less separable (i.e., all

subpopulations are vulnerable to poisoning) while for well-separated datasets, the individual subpopulation properties dominantly impact the susceptibility of subpopulations to poisoning attacks and more vulnerable subpopulations tend to have smaller loss difference between clean model and the target model generated from the simple label-flipping method that misclassifies the subpopulation, for both the synthetic (Section 4.3) and benchmark Adult (Section 4.4) datasets.

Understanding how distributional properties impact susceptibility. To understand how the learning algorithms will be impacted more broadly by data poisoning, we focus on investigating indiscriminate poisoning attacks (Chapter 5). The observation of disparate susceptibility across subpopulations motivates us to explore whether a similar variation of susceptibility also exists for different benchmark datasets against the best indiscriminate attacks, as indiscriminate attacks can be treated as special forms of subpopulation attacks that take the entire datasets as the subpopulations. By experimenting with existing state-of-the-art indiscriminate poisoning attacks (including the proposed MTP in Chapter 3) on linear models for various benchmark datasets, we observe that different datasets indeed have drastically different vulnerabilities to the best-performing poisoning attacks (Section 5.2). We then identify the distributional properties under the given learner that impact the performance of best possible optimal poisoning attacks on theoretical distributions and discover that a larger projected constraint size (Definition 5.4.8) is associated with a higher inherent vulnerability due to increased impact from the poisoning points, whereas projected data distributions with a larger separability and smaller standard deviation (Definition 5.4.9) are fundamentally less vulnerable to poisoning attacks due to the reduced sensitivity to misclassifications resulted from slight changes in the decision boundary (Section 5.4). Further, we discover that the factors identified on the theoretical distributions largely explain the drastic variation of best-performing empirical attacks across benchmark datasets, and these factors are also highly correlated to the upper bound on the performance of optimal data poisoning attacks for general distributions. We also show that minimizing the upper bound can provide a non-trivial quantity that upper bounds the performance of the optimal poisoning attacks and these upper bounds still vary significantly across different benchmark datasets (Section 5.5).

Implications for Improving Defenses. Understanding the limits of poisoning attacks in different poisoning settings might eventually give us more insights when designing effective defenses against data poisoning attacks. We first explain why certain data sanitization defenses work and then demonstrate how one might leverage well-trained feature extractors to obtain better feature representations for the candidate datasets and improve their resistance to indiscriminate poisoning attacks that inject additional poisoning points. We finally discuss how such an improved feature representation might complement the existing data sanitization

defenses proposed for the indiscriminate poisoning attacks, and also help develop better defenses for the subpopulation settings (Section 5.6).

1.2 Dissertation Structure

Chapter 2 introduces the preliminaries and the threat model in this dissertation, and then reviews the most relevant data poisoning attacks and data sanitization defenses. In Chapter 3, we provide details on the proposed model-targeted poisoning (MTP) attack and show its competitive performance in diverse attack objectives. In Chapter 4, we investigate subpopulation attacks and identify factors that are related to the drastic differences in the attack effectiveness of the MTP attack on different subpopulations. In Chapter 5, we study indiscriminate attacks and present factors that impact the performance of the optimal data poisoning attacks in theoretical settings and also the correlation to attack effectiveness of existing empirical attacks and the upper bound on the performance of optimal attacks for benchmark datasets. We also discuss the implication of the identified factors in designing better defenses against poisoning attacks. Finally, Chapter 6 concludes with a discussion of open questions and directions for future work.

Chapter 2

Background and Related Work

In this chapter, we first provide the preliminaries of the dissertation that contains the notations and also the problem setup (Section 2.1.1). Then we introduce the threat model considered in this dissertation (Section 2.1.2). In Section 2.2, we introduce the details on the state-of-the-art indiscriminate poisoning attacks, which also inspire the attack design in other attack goals (e.g., subpopulation and targeted) in the literature. At the end of this section, we also introduce some representative data sanitization defenses against poisoning (Section 2.2.3), which are used when we introduce how understanding the limits of poisoning attacks in the no-defense setting might enhance these defenses in Section 5.6.

2.1 Preliminaries and Threat Model

In this section, we first provide preliminaries on the considered convex models in this dissertation and also argue about the importance of studying them (Section 2.1.1). Then we discuss details on the considered threat model in this dissertation (Section 2.1.2).

2.1.1 Preliminaries on Convex Models

In this dissertation, we focus on convex models (mainly linear models), and below, we first argue about studying convex models and then provide related preliminaries on the notations and the problem setup for poisoning attacks on convex models.

Importance of studying convex models. We argue that studying convex models (or more precisely, linear models) are still relevant today, because attacks on them are still not well understood, despite extensive prior empirical work in this setting. Furthermore, insights obtained on the simpler convex models (Koh et al., 2022) can inspire the design of effective poisoning attacks on the challenging non-convex models such as deep neural networks (Lu et al., 2023). Besides the value in inspiring future research on complex models, linear models continue to garner significant interest due to their simplicity and high interpretability in explaining predictions (Liu et al., 2022; Ribeiro et al., 2016). In addition, these simple models also achieve competitive performance in many security-critical applications for which poisoning is relevant, including training with differential privacy (Tramèr and Boneh, 2021), recommendation systems (Ferrari Dacrema et al., 2019) and malware detection (Chen et al., 2023; 2021; Salah et al., 2020; Demontis et al., 2016; Šrndic and Laskov, 2013; Arp et al., 2014). From a practical perspective, linear models continue to be relevant—for example, Amazon SageMaker (Amazon, Inc., 2023), a scalable framework to train ML models intended for developers and business analysts, provides linear models for tabular data, and trains linear models (on top of pre-trained feature extractors) for images.

Notation. We use boldfaced lower letters such as $\boldsymbol{x}$ to denote vectors. For any set $\mathcal{A}$, $|\mathcal{A}|$ and $\mathbb{1}_{\mathcal{A}}(\cdot)$ denote the cardinality of $\mathcal{A}$ and the indicator function of $\mathcal{A}$, respectively. For any distribution μ, let $\mathrm{supp}(\mu)$ be the support of μ and use $\mathcal{S}$ to denotes the dataset sampled from μ. Given a dataset $\mathcal{S}$ sampled from μ, denote by $\hat{\mu}_{\mathcal{S}}$ the empirical measure with respect to $\mathcal{S}$. We use $\mathcal{N}(\gamma, \sigma^2)$ to denote the one-dimensional Gaussian distribution with mean γ and standard deviation σ. For any vector $\boldsymbol{x} \in \mathbb{R}^d$, denote by $\|\boldsymbol{x}\|_2$ the ℓ_2-norm of $\boldsymbol{x}$.

Convex surrogate loss. Throughout the dissertation, we mainly consider binary classification tasks using convex machine learning models and leave the systematic exploration of non-convex models such as neural networks for future work. The extension of our work to multi-class convex models is straightforward. Let $\mathcal{X} \subseteq \mathbb{R}^d$ be the input space and $\mathcal{Y} = \{-1, +1\}$ be the label space. Let μ_c be the joint distribution of clean inputs and labels. For standard classification tasks, the goal is to learn a hypothesis $h : \mathcal{X} \to \mathcal{Y}$ that minimizes $\mathrm{Risk}(h; \mu_c) = \mathbb{P}_{(\boldsymbol{x},y)\sim\mu_c}\big[h(\boldsymbol{x}) \neq y\big]$. However, directly minimizing the risk (or 0-1 loss) is computationally hard as it is neither convex nor differentiable. Therefore, in practice, typical machine learning methods find an approximately good hypothesis h by restricting the search space to a specific hypothesis class $\mathcal{H}$, then optimizing h by minimizing some convex surrogate loss that is efficient to optimize and the minimization of surrogate loss implies the minimization of $\mathrm{Risk}(h; \mu_c)$ (Bartlett et al., 2006). In particular, the optimization

problem can be formulated in the distributional setting as:

$$h_c = \operatorname*{argmin}_{h \in \mathcal{H}} \left[L(h; \mu_c) + C_R \cdot R(h) \right] \tag{2.1}$$

where the surrogate loss for h on a distribution μ is defined as $L(h; \mu) = \mathbb{E}_{(\boldsymbol{x},y)\sim\mu}\big[\ell(h; \boldsymbol{x}, y)\big]$ and $\ell(h; \boldsymbol{x}, y)$ denotes the non-negative individual loss of h incurred at $(\boldsymbol{x}, y)$. $R(h)$ denotes the regularization function for h (e.g., the ℓ_2-norm of the weight parameter in h) and $C_R > 0$ is the hyperparameter that balances the surrogate loss and the regularization term. Correspondingly, the finite-sample (or empirical) counterpart of the above optimization problem (also known as empirical risk minimization) can be similarly formulated as:

$$\hat{h}_c = \operatorname*{argmin}_{h \in \mathcal{H}} \Big[L(h; \mathcal{S}_c) + C_R \cdot R(h) \Big] \tag{2.2}$$

where $\mathcal{S}_c$ denotes the clean training data i.i.d. sampled from the clean distribution μ_c. We slightly abuse the notation and still use $L(h; \mathcal{S}) = \frac{1}{|\mathcal{S}|} \cdot \sum_{(\boldsymbol{x},y)\in\mathcal{S}} \big[\ell(h; \boldsymbol{x}, y)\big]$ to denote the empirical estimate of $L(h; \mu)$ using the dataset $\mathcal{S}$ sampled in an i.i.d. manner from μ. Throughout the dissertation, the usage of $L(\cdot)$ in the finite-sample and distributional settings will be distinguished using $\mathcal{S}$ (i.e., $L(h; \mathcal{S})$) or μ (i.e., $L(h; \mu)$). We mostly use h for clarity in presentation throughout this dissertation, but will explicitly write out the weight parameter $\boldsymbol{\theta}$ of h as $h_{\boldsymbol{\theta}}$ when the usage of the weight parameters $\boldsymbol{\theta}$ can improve the preciseness of the presentation. Related to the empirical risk minimization, we also define the attainable model using some training set $\mathcal{S}$:

Definition 2.1.1 (Attainable models)**.** We say a hypothesis $\hat{h} \in \mathcal{H}$ is C_R-attainable with respect to loss function L and regularization function R if there exists a training set $\mathcal{S}$ such that

$$\hat{h} = \operatorname*{argmin}_{h \in \mathcal{H}} L(h; \mathcal{S}) + C_R \cdot R(h)$$

and $\hat{h}$ is the unique minimizer for the above.

Linear learners. The formulation above is for any convex loss functions and is used in the theoretical analysis in Chapter 3. However, in Chapter 5, we also study the particular case of linear hypothesis class with hinge loss, which is the most common setting considered in prior works (Biggio et al., 2011; 2012; Steinhardt et al., 2017; Koh et al., 2022), but the theoretical results can also be easily extended to other linear methods such as logistic regression (LR). A *linear hypothesis* parameterized by a weight parameter $\boldsymbol{w} \in \mathbb{R}^d$ and a bias parameter $b \in \mathbb{R}$ (i.e., $\boldsymbol{\theta} = (\boldsymbol{w}, b)$) is defined as: $h_{\boldsymbol{w},b}(\boldsymbol{x}) = \operatorname{sgn}(\boldsymbol{w}^\top \boldsymbol{x} + b)$ for any $\boldsymbol{x} \in \mathbb{R}^d$, where $\operatorname{sgn}(\cdot)$

denotes the sign function. For any $\boldsymbol{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$, the *hinge loss* of a linear classifier $h_{\boldsymbol{w},b}$ (when the context is clear, we will still use h for simplicity) is defined as:

$$\ell(h_{\boldsymbol{w},b}; \boldsymbol{x}, y) = \max\{0, 1 - y(\boldsymbol{w}^\top \boldsymbol{x} + b)\} \tag{2.3}$$

and the regularization term for h parameterized by a weight vector $\boldsymbol{w}$ and a bias parameter b is the ℓ_2-norm regularization and is defined as:

$$C_R = \frac{\lambda}{2} \text{ and } R(h_{\boldsymbol{w}}) = \|\boldsymbol{w}\|_2^2 \tag{2.4}$$

where $\lambda \geq 0$ is the tuning hyperparameter and b is usually not regularized.

2.1.2 Threat Model

We consider injection-only poisoning attacks, which can be formulated as a game between an attacker and a victim in practice (i.e., finite-sample setting) (Steinhardt et al., 2017; Koh et al., 2022):

1. A clean training dataset $\mathcal{S}_c$ is produced, where each data point is i.i.d. sampled from μ_c.
2. The attacker generates a poisoned dataset $\mathcal{S}_p$ using some poisoning strategy $\mathcal{A}$, which aims to achieve some attack goals on compromising the model performance on the defined subset of the distribution (subpopulation setting) or the entire clean test data (indiscriminate setting) by injecting $\mathcal{S}_p$ into the training dataset.
3. The victim minimizes empirical surrogate loss $L(\cdot)$ on $\mathcal{S}_c \cup \mathcal{S}_p$ and produces a poisoned model $\hat{h}_p$ that performs in factor of the adversary.

Our game-theoretic formulation assumes insertion-only attacks, where the attacker is only able to add crafted poisoning points $\mathcal{S}_p$ into the clean training dataset $\mathcal{S}_c$. This assumption is reasonable for many practical scenarios where modifying existing training data points would require administrative access to the underlying system (Steinhardt et al., 2017; Koh et al., 2022), and many prior works are also limited to insertion-only attacks (Biggio et al., 2011; 2012; Steinhardt et al., 2017; Koh and Liang, 2017; Koh et al., 2022; Lu et al., 2022; 2023). Depending on the considered settings, the attacker's goal can be finding an effective poisoning strategy $\mathcal{A}$ such that it achieves the encoded attack objective in a particular target model $\hat{h}_{tar}$ (experiments in Chapter 3 and Chapter 4) or to compromise the model performance as much as possible under a fixed poisoning ratio ϵ (experiments in Chapter 5).

Following prior work (Steinhardt et al., 2017; Koh et al., 2022), we assume the attacker has full knowledge of the learning process, including the clean distribution μ_c (for theoretical analysis in Chapter 5) or the clean training dataset $\mathcal{S}_c$ and clean test data $\mathcal{S}_{te}$ (for empirical evaluations in Chapter 3 and Chapter 4), the hypothesis class $\mathcal{H}$, the surrogate loss function ℓ, regularization terms C_R, $R(h)$ and the learning algorithm adopted by the victim. This threat model is admittedly the strongest threat model in the injection-only setting and is generous to the attacker, but it is widely considered a bad practice to rely on secrecy for security (Kerckhoffs, 1883; Biggio et al., 2013; Steinhardt et al., 2017). Furthermore, as we will show in Chapter 4 and Chapter 5, attacks even under the strongest threat model still cannot effectively render the performance of poisoned models in certain settings and understanding the root causes for their ineffectiveness might provide better insights for the future defenses.

As for the poisoning attack, we assume $\mathcal{S}_p \subseteq \mathcal{C}$ where $\mathcal{C} \subseteq \mathcal{X} \times \mathcal{Y}$ is a bounded subset that captures the feasibility constraints for poisoned data. We assume that $\mathcal{C}$ is specified in advance with respect to different applications (e.g., normalized pixel values of images can only be in the range $[0, 1]$). Note that, in general, possible defenses the victim may choose (e.g., points that have larger Euclidean distance from the center will be removed) (Steinhardt et al., 2017; Koh et al., 2022) will also impact the choice of $\mathcal{C}$. However, in this dissertation, we mainly explore the limits of poisoning attacks in the no-defense setting and therefore, we will not consider the impact of defenses on $\mathcal{C}$ except in Section 5.6 when we discuss the implication of our results on future defenses. If the considered poisoning setting is to compromise the model performance as much as possible using a fixed poisoning ratio ϵ, as in the conventional indiscriminate poisoning setting, then we also constrain $|\mathcal{S}_p| \leq \epsilon \cdot |\mathcal{S}_c|$, where $\epsilon \in [0, 1]$ is the poisoning budget.

The above threat model is usually also formulated in the following bi-level optimization problem, and the existing poisoning attacks in the literature substantiate the general formulation in different forms (shown in Section 2.2):

Definition 2.1.2 (Worst Case Injection-only Poisoning Attacks)**.** Given the information about the victim regarding the clean training data $\mathcal{S}_c$ and clean test data $\mathcal{S}_{te}$ sampled in i.i.d. manner from a poison-free (but usually unknown) distribution μ_c, the learning algorithm $L(\cdot)$ and regularization related terms C_R and $R(h)$, we define a worst-case injection-only poisoning attack as an attack method $\mathcal{A}$ that leverages all information about the victim to maximize a certain attack objective $Obj(\cdot)$ by injecting a poisoning set $\mathcal{S}_p$ into the clean training set $\mathcal{S}_c$, and in the conventional indiscriminate poisoning attacks, only a maximum ϵ fraction of

poisoning points will be injected:

$$\mathcal{S}_p^* = \underset{\mathcal{S}_p}{\operatorname{argmax}}\ Obj(\hat{h}_p, \mathcal{S}_{te}, \mathcal{S}_c, \mathcal{S}_p, \ell, C_R, R) \quad \text{s.t. } \hat{h}_p = \underset{h\in\mathcal{H}}{\operatorname{argmin}}\ \big[L(h; \mathcal{S}_c \cup \mathcal{S}_p) + C_R \cdot R(h)\big],$$
$$\mathcal{S}_p \subseteq \mathcal{C},$$
$$|\mathcal{S}_p| \leq \epsilon \cdot |\mathcal{S}_c| \quad \text{(Optional for Conventional Indiscriminate Settings)}$$

Note that the bi-level optimization (due to the inter-dependence between $\mathcal{S}_p$ and $\hat{h}_p$) is in general NP-hard (Steinhardt et al., 2017; Bard, 2013) and is also the main reason why poisoning attacks (especially in indiscriminate settings) are still majorly limited to simple convex models (e.g., mostly linear models) compared to attacks against fixed static models such as adversarial examples (Szegedy et al., 2014), where state-of-the-art complex deep learning models can be easily evaded.

2.2 Indiscriminate Poisoning Attacks and Defenses

In this section, we provide details on the current indiscriminate poisoning attacks and representative data sanitization defenses in the literature. For the poisoning attacks, we only choose to present the indiscriminate attacks because state-of-the-art poisoning attacks for other attack goals majorly follow the ideas of attacks in the indiscriminate setting.

But before diving into specific attack methods, we first restate Theorem 1 in Koh et al. (2022) (with slight adaptation for our setting) so as to better illustrate the existing poisoning attacks to be introduced next. This result is also related to the lower bound on the number of poisoning points needed to induce a given target model when we introduce the MTP attack in Chapter 3.

Theorem 2.2.1 (2 points suffice for 2-class SVMs and logistic regression)**.** *Consider a learner that learns a 2-class SVM or logistic regression model by minimizing the average (regularized) training loss on $\mathcal{S}_c \cup \mathcal{S}_p$. Suppose that for each class $y = -1, +1$, the search space of poisoning points set $\mathcal{C}$ is a convex set. If a model $\hat{h}_p$ is C_R-attainable by any set of n poisoned points $\mathcal{S}_p = \{(\boldsymbol{x}_p^1, y_p^1), \ldots, (\boldsymbol{x}_p^n, y_p^n)\} \subseteq \mathcal{C}$, then there exists a set of at most n poisoned points $\tilde{\mathcal{S}}_p$ (possibly with fractional copies) that also attains $\hat{h}_p$ but only contains 2 distinct points, one from each class.*

More generally, the above statement is true for any margin-based model with loss of the form $\ell_M(h_{\boldsymbol{w},b}; \boldsymbol{x}, y) = c(-y(\boldsymbol{w}^\top \boldsymbol{x} + b))$*, where* $c : \mathbb{R} \to \mathbb{R}$ *is a convex, monotone increasing, and twice-differentiable function, and the ratio of second to first derivatives* c''/c' *is monotone non-increasing.*

Next, we will first show the canonical gradient attack that is based on the idea of influence function (Section 2.2.1), which is also the main go-to methods when designing attacks for other attack goals (Jagielski et al., 2021; Geiping et al., 2021). Then, we show the attacks that work with target models (Section 2.2.2), which often avoids the drawbacks of local optimization strategies as in the gradient attack. At last, we review some representative data sanitization defenses against indiscriminate poisoning attacks (Section 2.2.3).

2.2.1 Influence/Gradient Attacks

The influence (or gradient) attack is based on the idea of influence function (Rousseeuw et al., 2011) to convert the bi-level optimization problem (Definition 2.1.2) into a single-level optimization problem (Biggio et al., 2011; Mei and Zhu, 2015b;a; Koh and Liang, 2017). The influence attack usually aims to maximize the test loss $L(\hat{h}_p; \mathcal{S}_{te})$ (the $Obj(\cdot)$ in Definition 2.1.2) and iteratively updates the poisoning point $(\boldsymbol{x}_p, y_p) \in \mathcal{S}_p$ in the direction of the gradient $\frac{\partial L}{\partial \boldsymbol{x}_p}$ (with label y_p fixed in advance). The difficulty in computing the gradient of the test loss $L(\hat{h}_p; \mathcal{S}_{te})$ w.r.t. each $\boldsymbol{x}_p$ in $\mathcal{S}_p$ is that L depends on $\boldsymbol{x}_p$ only through the model parameters $\hat{\boldsymbol{\theta}}_p$ of $\hat{h}_p$ (also written as $\hat{h}_{\hat{\boldsymbol{\theta}}_p}$), which is a complicated function of $\mathcal{S}_p$. Below is the detail on approximately computing the gradients, which is slightly adapted from the description in Koh et al. (2022). We make a special note here regarding the work by Koh et al. (2022): the original manuscript is published in 2018, which is before all the proposed works in this dissertation. However, the official acceptance of the mentioned paper (with slight updates) is in 2022, which is after the main work discussed in Chapter 3. However, for the sake of consistency, throughout this dissertation, we will refer to the mentioned paper in its 2022 citation form (i.e., Koh et al. (2022), (Koh et al., 2022)), but treat it as a prior work to this dissertation.

Computing the gradient. The influence attacks compute the desired gradient $\frac{\partial L}{\partial \boldsymbol{x}_p}$ by the chain rule of $\frac{\partial L}{\partial \boldsymbol{x}_p} = \frac{\partial L}{\partial \hat{\boldsymbol{\theta}}_p} \frac{\partial \hat{h}_{\hat{\boldsymbol{\theta}}_p}}{\partial \boldsymbol{x}_p}$. The first term $\frac{\partial L}{\partial \hat{\boldsymbol{\theta}}_p}$ is the average gradient of the test loss with respect to the weight parameter $\hat{\boldsymbol{\theta}}_p$ for the model $\hat{h}_{\hat{\boldsymbol{\theta}}_p}$ and it can be straightforwardly computed as

$$g_{\hat{h}_p, \mathcal{S}_{te}} \stackrel{\text{def}}{=} \frac{\partial L}{\partial \hat{\boldsymbol{\theta}}_p} = \frac{1}{|\mathcal{S}_{te}|} \sum_{(\boldsymbol{x}, y) \in \mathcal{S}_{te}} \nabla_{\boldsymbol{\theta}} \ell(\hat{h}_{\hat{\boldsymbol{\theta}}_p}; \boldsymbol{x}, y). \tag{2.5}$$

The second term $\frac{\partial \hat{h}_{\hat{\boldsymbol{\theta}}_p}}{\partial \boldsymbol{x}_p}$ is complicated and cannot be directly computed due to inter-dependence between $\hat{h}_{\hat{\boldsymbol{\theta}}_p}$ and $\boldsymbol{x}_p$. Instead, it is estimated based on the influence function, which provides closed-form estimation for

the model weights in $\hat{h}_{\hat{\boldsymbol{\theta}}_p}$ when $\boldsymbol{x}_p$ is changed slightly (Koh and Liang, 2017). Specifically,

$$\frac{\partial \hat{h}_{\hat{\boldsymbol{\theta}}_p}}{\partial \boldsymbol{x}_p} = -H_{\hat{h}_p}^{-1} \frac{\partial^2 \ell(\hat{h}_{\hat{\boldsymbol{\theta}}_p}; \boldsymbol{x}_p, y_p)}{\partial \hat{\boldsymbol{\theta}}_p \, \partial \boldsymbol{x}_p}, \tag{2.6}$$

where $H_{\hat{h}_p}$ is the Hessian of the training loss at $\hat{h}_p$:

$$H_{\hat{h}_p} \stackrel{\text{def}}{=} C_R \cdot \frac{\partial R(\hat{h}_{\hat{\boldsymbol{\theta}}_p})}{\partial^2 \hat{\boldsymbol{\theta}}_p} + \frac{1}{|\mathcal{S}_c \cup \mathcal{S}_p|} \sum_{(\boldsymbol{x}, y) \in \mathcal{S}_c \cup \mathcal{S}_p} \frac{\partial^2 \ell(\hat{h}_{\hat{\boldsymbol{\theta}}_p}; \boldsymbol{x}, y)}{\partial \hat{\boldsymbol{\theta}}_p^2}. \tag{2.7}$$

For $\hat{h}_{\hat{\boldsymbol{\theta}}_p}$ with the commonly used ℓ_2-regularization with λ as the hyperparameter, we have $C_R \cdot \frac{\partial R(\hat{h}_{\hat{\boldsymbol{\theta}}_p})}{\partial^2 \hat{\boldsymbol{\theta}}_p} = \lambda \boldsymbol{I}$. Combining equations (2.5) to (2.7), the gradient of the test loss w.r.t. an attack point $\boldsymbol{x}_p$ is

$$\frac{\partial L(\hat{h}_p)}{\partial \boldsymbol{x}_p} = -g_{\hat{h}_p, \mathcal{S}_{te}}^{\top} H_{\hat{h}_p}^{-1} \frac{\partial^2 \ell(\hat{h}_{\hat{\boldsymbol{\theta}}_p}; \boldsymbol{x}_p, y_p)}{\partial \hat{\boldsymbol{\theta}}_p \, \partial \boldsymbol{x}_p}.$$

With the defined ways to compute the gradients, the attack runs iteratively as outlined in Algorithm 1. One major drawback of the influence attack is, because the non-convex bi-level optimization problem is solved iteratively, the returned solution can sometimes get stuck into bad local optima, leading to underperformance in some attack settings (Steinhardt et al., 2017; Koh et al., 2022). In addition, because of the involved inverse Hessian in computation and the necessity to optimize $\epsilon|\mathcal{S}_c|$ individual poisoning points, the gradient attacks are usually very slow, even for the simple linear models. Using the result in Theorem 2.2.1, the efficiency of the attack can be improved by only optimizing two distinct points with different fractional repetitions (Koh et al., 2022).

Algorithm 1 The Influence Attack.

Input: clean data set $\mathcal{S}_c$, poisoning fraction ϵ, learning rate lr.
Output: $\mathcal{S}_p$
Initialize poisoned data set $\mathcal{S}_p \leftarrow \{(\boldsymbol{x}_p^1, y_p^1), \ldots, (\boldsymbol{x}_p^{\epsilon n}, y_p^{\epsilon n})\}$.
for $t = 1, 2, \ldots$ **do**
    Compute model parameters of $\hat{h}_p \leftarrow \operatorname{argmin}_h L(h; \mathcal{S}_c \cup \mathcal{S}_p)$.
    Pre-compute $g_{\hat{h}_p, \mathcal{S}_{te}}, H_{\hat{h}_p}^{-1}$ as in (2.5) and (2.7).
    for $i = 1, \ldots, \epsilon n$ **do**
        Set $\boldsymbol{x}_p^i \leftarrow \boldsymbol{x}_p^i - lr \cdot g_{\hat{h}_p, \mathcal{S}_{te}}^{\top} H_{\hat{h}_p}^{-1} \frac{\partial^2 \ell(\hat{h}_{\hat{\boldsymbol{\theta}}_p}; \boldsymbol{x}_p^i, y_p^i)}{\partial \hat{\boldsymbol{\theta}}_p \, \partial \boldsymbol{x}_p^i}$.
    end for
end for
return $\mathcal{S}_p$.

2.2.2 Attacks with Target Models

These attacks convert the bi-level optimization into single-level optimization by utilizing additional target models and sometimes also utilize the min-max nature of the indiscriminate poisoning attacks (Koh et al., 2022). Since attacks in this section can work with a given target model, we first describe the heuristic approach to generate a target model and then introduce the individual attack methods that generate effective poisoning points with the target model as an input.

Generating target models with label-flipping. Target models can be generated using some simple heuristics such as label-flipping. With the generated target models, then different attacks can be designed to achieve the encoded attack objectives in the target models or to maximally compromise the model performance, potentially more efficiently using fewer poisoning points. The label-flipping method to generate a target model, proposed by Koh et al. (2022), is outlined in Algorithm 2, where the loss threshold γ and the number of repeats r are the two hyperparameters to tune. Koh et al. (2022) performs a grid search over a possible combination of γ and r from a list of values and generates a candidate set of target models and finally filters out target models that achieve similar or smaller test errors while having larger loss on the clean training data. A theoretical justification for the last step of filtering will be given in the implication of Theorem 3.3.3 in Section 3.3.1. If the attacker also has a requirement on the test error of the target models, then it can also additionally filter out target models that have smaller errors than the requirement, as these models will be unlikely to be useful. The label-flipping approach can also be applied to the subpopulation settings by simply replacing the $\mathcal{S}_{te}$ in Line 2 in Algorithm 2 with the test set $\mathcal{S}_{te}^{sub}$ belonging to the subpopulation. Then, the target classifier that satisfies the attacker goal (e.g., has similar or higher test errors than the requirement on the subpopulation) and has the lowest loss on the clean training data will be returned.

Algorithm 2 Finding Target Model $\hat{h}_{tar}$

Input: $\mathcal{S}_c$, $\mathcal{S}_{te}$, the loss functions L, l, regularization strength C_R, loss threshold γ, number of repeats r

Output: $\hat{h}_{tar}$

1: $\hat{h}_c = \operatorname{argmin} L(h; \mathcal{S}_c) + C_R \cdot R(h)$

2: $\mathcal{S}_{\text{flip}} = r$ copies of $\{(\boldsymbol{x}, -y) : (\boldsymbol{x}, y) \in \mathcal{S}_{te}, l(\hat{h}_c; \boldsymbol{x}, y) \geq \gamma\}$

3: $\mathcal{S}_{\text{comb}} = \mathcal{S}_c \cup \mathcal{S}_{\text{flip}}$

4: $\hat{h}_{tar} = \operatorname{argmin} L(h; \mathcal{S}_{\text{comb}}) + C_R \cdot R(h)$

5: **return** $\hat{h}_{tar}$

Next, we introduce different attach methods that generate the poisoning points with a target model as an additional input.

KKT attack. KKT attack (Koh et al., 2022; Lu et al., 2023) is based on the fact that, if a target model $\hat{h}_{tar}$ with weight parameter $\hat{\boldsymbol{\theta}}_{tar}$ ($\hat{h}_{\hat{\boldsymbol{\theta}}_{tar}}$ to be precise) is achieved using a poisoning set $\mathcal{S}_p$ by the ERM algorithm given in (2.2), then it should satisfy the first order optimality condition for the target model $\hat{h}_{tar}$. In particular, $\hat{h}_{tar}$ should satisfy the equality problem of

$$\hat{h}_{tar} = \underset{h}{\operatorname{argmin}}\, L(h; \mathcal{S}_c \cup \mathcal{S}_p) + C_R \cdot R(h) = \underset{h}{\operatorname{argmin}} \sum_{(\boldsymbol{x},y)\in\mathcal{S}_c} \ell(h; \boldsymbol{x}, y) + \sum_{(\boldsymbol{x}_p,y_p)\in\mathcal{S}_p} \ell(h; \boldsymbol{x}_p, y_p) + C_R \cdot R(h)$$

as well as the equivalent KKT optimality condition

$$\sum_{(x,y)\in\mathcal{S}_c} \nabla_{\boldsymbol{\theta}} \ell(\hat{h}_{\hat{\boldsymbol{\theta}}_{tar}}; \boldsymbol{x}, y) + \sum_{(\boldsymbol{x}_p,y_p)\in\mathcal{S}_p} \nabla_{\boldsymbol{\theta}} \ell(\hat{h}_{\hat{\boldsymbol{\theta}}_{tar}}; \boldsymbol{x}_p, y_p) + C_R \cdot \frac{R(\hat{h}_{\hat{\boldsymbol{\theta}}_{tar}})}{\partial \hat{\boldsymbol{\theta}}_{tar}} = \mathbf{0}. \tag{2.8}$$

Given a target model $\hat{h}_{\hat{\boldsymbol{\theta}}_{tar}}$, the KKT equality condition in (2.8) can be formulated as an optimization problem to minimize the norm of the gradient given in (2.8) (i.e., related to $-Obj(\cdot)$ in Definition 2.1.2), relying on the fact that the norm of a vector is minimized when the vector is $\mathbf{0}$. In particular, we have

$$\begin{aligned} \min_{\mathcal{S}_p} \quad & \Big\| \frac{1}{|\mathcal{S}_c|} \sum_{(x,y)\in\mathcal{S}_c} \nabla_{\boldsymbol{\theta}} \ell(\hat{h}_{\hat{\boldsymbol{\theta}}_{tar}}; \boldsymbol{x}, y) + \frac{1}{|\mathcal{S}_c|} \sum_{(\boldsymbol{x}_p,y_p)\in\mathcal{S}_p} \nabla_{\boldsymbol{\theta}} \ell(\hat{h}_{\hat{\boldsymbol{\theta}}_{tar}}; \boldsymbol{x}_p, y_p) + C_R(1+\epsilon) \cdot \frac{R(\hat{h}_{\hat{\boldsymbol{\theta}}_{tar}})}{\partial \hat{\boldsymbol{\theta}}_{tar}} \Big\|_2^2 \\ \text{s.t.} \quad & |\mathcal{S}_p| = \epsilon |\mathcal{S}_c| \\ & \mathcal{S}_p \subseteq \mathcal{C}, \end{aligned} \tag{2.9}$$

The above formulation is the original form of the KKT attack and is also successfully applied to neural networks very recently (Lu et al., 2022)). However, by leveraging Theorem 2.2.1 for convex models, the above optimization of the whole set of (possibly distinct) $\mathcal{S}_p$ points can be turned into the optimization of two distinct points from each class in the binary classification case. Then the problem in (2.9) is simplified to

$$\begin{aligned} \min_{\boldsymbol{x}_p^+, \boldsymbol{x}_p^-, \epsilon^+, \epsilon^-} \quad & \Big\| \frac{1}{|\mathcal{S}_c|} \sum_{(x,y)\in\mathcal{S}_c} \nabla_{\boldsymbol{\theta}} \ell(\hat{h}_{\hat{\boldsymbol{\theta}}_{tar}}; \boldsymbol{x}, y) + \epsilon^+ \nabla_{\boldsymbol{\theta}} \ell(\hat{h}_{\hat{\boldsymbol{\theta}}_{tar}}; \boldsymbol{x}_p^+, 1) + \epsilon^- \nabla_{\boldsymbol{\theta}} \ell(\hat{h}_{\hat{\boldsymbol{\theta}}_{tar}}; \boldsymbol{x}_p^-, -1) + C_R(1+\epsilon) \cdot \frac{R(\hat{h}_{\hat{\boldsymbol{\theta}}_{tar}})}{\partial \hat{\boldsymbol{\theta}}_{tar}} \Big\|_2^2 \\ \text{s.t.} \quad & \epsilon^+ + \epsilon^- = \epsilon \\ & (\boldsymbol{x}_p^+, 1), (\boldsymbol{x}_p^-, -1) \in \mathcal{C}. \end{aligned} \tag{2.10}$$

For general losses (e.g., logistic loss), this optimization problem is non-convex and requires local optimization methods such as gradient descent, but for hinge loss, the problem is indeed convex. The algorithmic detail is given in Algorithm 3, where $\hat{h}_p$ is obtained by training on the poisoned set $\mathcal{S}_c \cup \mathcal{S}_p$ and $\mathcal{S}_p$ only consists of two distinct points. We note that the KKT attack aims to exactly produce the target model $\hat{h}_{tar}$ and hence, can be applied beyond the indiscriminate attack goals by generating suitable target models, and we will additionally test the KKT attack in the subpopulation setting in Section 3.4.

Algorithm 3 KKT attack with grid search.

Input: clean data set $\mathcal{S}_c$, $\mathcal{S}_{te}$, feasible set $\mathcal{C}$, poisoning fraction ϵ, target model $\hat{h}_{tar}$, grid search size T.
Output: $\mathcal{S}_p$.
for $t = 0, \ldots, T$ **do**
  Set $\epsilon^+ \leftarrow t\epsilon/T$, $\epsilon^- \leftarrow \epsilon - \epsilon^+$.
  Obtain $\boldsymbol{x}_p^+$, and $\boldsymbol{x}_p^-$ by solving (2.10) with fixed values of ϵ^+, ϵ^-.
  Train $\hat{h}_p$ on the current poisoned dataset.
  Evaluate test error $L(\hat{h}_p; \mathcal{S}_{te})$.
end for
Pick $\boldsymbol{x}_p^+, \boldsymbol{x}_p^-, \epsilon^+, \epsilon^-$ corresponding to the highest test error $L(\hat{h}_p; \mathcal{S}_{te})$ found in grid search.
return $\mathcal{S}_p = \{\epsilon^+|\mathcal{S}_c| \text{ copies of } \boldsymbol{x}_p^+ \text{ and } \epsilon^-|\mathcal{S}_c| \text{ copies of } \boldsymbol{x}_p^-\}$.

Min-Max attacks. The Min-Max attack is tailored only for the indiscriminate setting because the indiscriminate poisoning attack can be naturally formulated as a min-max optimization problem. Specifically, one can approximate the average test loss with the average training loss ($L(h; \mathcal{S}_{te}) \approx L(h; \mathcal{S}_c)$), given that the model is properly regularized and does not have significant overfitting, while the training loss can be further upper bounded by the following

$$L(h; \mathcal{S}_c) \leq L(h; \mathcal{S}_c) + \epsilon L(h; \mathcal{S}_p) = (1+\epsilon) L(h; \mathcal{S}_c \cup \mathcal{S}_p) \leq (1+\epsilon)\big(L(h; \mathcal{S}_c \cup \mathcal{S}_p) + C_R \cdot R(h)\big), \tag{2.11}$$

At this step, instead of directly optimizing for $L(h; \mathcal{S}_{te})$ as the attacker, one can therefore optimize for $L(h; \mathcal{S}_c \cup \mathcal{S}_p) + C_R \cdot R(h)$ (i.e., the $Obj(\cdot)$ in Definition 2.1.2), which gives us

$$\begin{aligned} &\max_{\mathcal{S}_p \subseteq \mathcal{C}} \quad L(\hat{h}_p; \mathcal{S}_c \cup \mathcal{S}_p) + C_R \cdot R(h) \\ &\text{where} \quad \hat{h}_p \overset{\text{def}}{=} \operatorname*{argmin}_h L(h; \mathcal{S}_c \cup \mathcal{S}_p) + C_R \cdot R(h). \end{aligned}$$

The above formulation contains outer maximization and inner minimization over the same function $L(h; \mathcal{S}_c \cup \mathcal{S}_p) + C_R \cdot R(h)$, which leads to the saddle point problem

$$\max_{\mathcal{S}_p \subseteq \mathcal{C}} \min_h \quad L(h; \mathcal{S}_c \cup \mathcal{S}_p) + C_R \cdot R(h). \tag{2.12}$$

When the loss ℓ is convex, we can solve (2.12) by swapping min and max and solving the resulting min-max optimization (also how the name *Min-Max* comes from) problem $\min_h \max_{\mathcal{S}_p \subseteq \mathcal{C}} L(h; \mathcal{S}_c \cup \mathcal{S}_p) + C_R \cdot R(h)$, which expands out to

$$\min_h \Big[(1+\epsilon) C_R \cdot R(h) + L(h; \mathcal{S}_c) + \epsilon \max_{(\boldsymbol{x}_p, y_p) \in \mathcal{C}} \ell(h; \boldsymbol{x}_p, y_p) \Big]. \tag{2.13}$$

Still, for the commonly used ℓ_2-norm regularization, we can replace $(1+\epsilon) C_R \cdot R(h_{\boldsymbol{\theta}})$ with $\frac{\lambda(1+\epsilon)}{2} \|\boldsymbol{\theta}\|_2^2$. One can solve this problem by iteratively finding $(\boldsymbol{x}_p, y_p) \in \mathcal{C}$ that maximizes $\ell(h; \boldsymbol{x}_p, y_p)$, then taking the subgradient of the outer expression using the obtained $(\boldsymbol{x}_p, y_p)$. Note that, the original min-max attack shown above does not require target models. However, the drawback of the Min-Max attack without target models is, the generated poisoning points that maximize $\ell(h; \boldsymbol{x}_p, y_p)$ can be too extreme (especially in the case where there is no defense deployed) and the models trained on them fit poorly on the poisoning points (i.e., high loss) while fitting well on the clean points (i.e., low loss), leading to a poisoned model that still performs well on the clean data points, which is not aligned with the attacker goal. To circumvent this issue, one can add another term that constrains the loss of the poisoning points with respect to a given target model $\hat{h}_{tar}$ ($\leq \tau$, where τ is a preset threshold), so as to ensure the poisoning points are not too extreme. More formally, the maximization of $\ell(h; \boldsymbol{x}_p, y_p)$ now becomes:

$$\begin{aligned} \max_{\boldsymbol{x}_p} \quad & \ell(h; \boldsymbol{x}_p, y_p) \\ \text{s.t.} \quad & (\boldsymbol{x}_p, y_p) \in \mathcal{C}. \\ & \ell(\hat{h}_{tar}; \boldsymbol{x}_p, y_p) \leq \tau \end{aligned}$$

Algorithm 4 for the Min-Max attack is outlined below (using ℓ_2 regularization as an example), where we directly introduced the improved version of the Min-Max and call it *i-Min-Max* throughout this dissertation.

Upper bound on optimal poisoning. The formulation of the Min-Max attack (without target model) also provides an upper bound to the maximum loss on $\mathcal{S}_c$ achievable from any indiscriminate poisoning attacks. In particular, because the max-min problem is always upper bound by its min-max counterpart for convex

Algorithm 4 i-Min-Max attack.

Input: clean data $\mathcal{S}_c$, poisoned fraction ϵ, burn-in n_{burn}, feasible set $\mathcal{C}$, learning rate lr, target model $\hat{h}_{tar}$.
Output: $\mathcal{S}_p$.
Initialize: $\boldsymbol{\theta} \in \mathbb{R}^d$ for $h_{\boldsymbol{\theta}}$, $\mathcal{S}_p \leftarrow \emptyset$.
for $t = 1, \ldots, n_{\text{burn}} + \epsilon n$ **do**
  Pick$(\boldsymbol{x}_p, y_p) \in \text{argmax}_{(\boldsymbol{x},y)\in\mathcal{C}}\, \ell(h; \boldsymbol{x}, y)$, s.t. $\ell(\hat{h}_{tar}; \boldsymbol{x}_p, y_p) \leq \tau$.
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - lr \cdot ((1+\epsilon)\lambda\boldsymbol{\theta} + \nabla_{\boldsymbol{\theta}} L(h_{\boldsymbol{\theta}}) + \epsilon\nabla_{\boldsymbol{\theta}}\ell(h_{\boldsymbol{\theta}}; \boldsymbol{x}_p, y_p))$
  if $t > n_{\text{burn}}$ **then**
    $\mathcal{S}_p \leftarrow \mathcal{S}_p \cup \{(\boldsymbol{x}_p, y_p)\}$
  end if
end for
return $\mathcal{S}_p$.

losses, we have

$$\begin{aligned}
\max_{\mathcal{S}_p\subseteq\mathcal{C}} L(\hat{h}_p; \mathcal{S}_c) &\leq \max_{\mathcal{S}_p\subseteq\mathcal{C}} (1+\epsilon)\Big(L(\hat{h}_p; \mathcal{S}_c \cup \mathcal{S}_p) + C_R \cdot R(\hat{h}_p)\Big) \\
&= \max_{\mathcal{S}_p\subseteq\mathcal{C}} \min_h (1+\epsilon)\Big(L(h; \mathcal{S}_c \cup \mathcal{S}_p) + C_R \cdot R(h)\Big) \\
&\leq \min_h \max_{\mathcal{S}_p\subseteq\mathcal{C}} (1+\epsilon)\Big(L(h; \mathcal{S}_c \cup \mathcal{S}_p) + C_R \cdot R(h)\Big) \\
&= \min_h \Big(L(h; \mathcal{S}_c) + \epsilon \max_{(\boldsymbol{x}_p,y_p)\in\mathcal{C}} \ell(h; \boldsymbol{x}_p, y_p) + (1+\epsilon) C_R \cdot R(h)\Big) \qquad (2.14)
\end{aligned}$$

This upper bound also inspires us to establish the connection between the factors we identified for explaining dataset susceptibilities and the upper bound on the maximum risk inducible from the (best possible) optimal indiscriminate poisoning attacks in the distributional setting in Chapter 5. Note that (2.14) is for the finite-sample setting and the surrogate loss on $\mathcal{S}_c$ always upper bounds the training error on $\mathcal{S}_c$ and for well-regularized models, the training and test errors are similar. Therefore, by minimizing the upper bound above, one can get an (approximate) upper bound on the maximum test error achievable by any poisoning attacks, although this upper bound might be loose. The main idea behind the Min-Max attack (without target model) is to gradually minimize the upper bound, but halt the optimization when there are some number of poisoning points generated (to act as an empirical attack strategy). This can lead to a very loose estimation on the upper bound, especially in the no-defense setting. In Section 5.5.2, we run the optimization for significantly more number of iterations to achieve tighter upper bound on the (approximate) maximal test error after poisoning.

2.2.3 Background on Data Sanitization Defenses

In this section, we briefly introduce the representative data sanitization defenses that will be relevant when we discuss how understanding the limits of indiscriminate poisoning attacks can help future defenses. Two

representative defenses called *Slab* and *Sphere* focus on filtering out a small fraction of outlier points that are from their respective centroids. These attacks can be effective for datasets such as MNIST (LeCun et al., 1998) digit pairs of 1 and 7, especially if one knows the true centroid of the data points in each class (Steinhardt et al., 2017).

Slab and Sphere. The Slab measures the distance of a point $(\boldsymbol{x}, y), y \in \{-1, 1\}$ to its centroid (denoted as $\boldsymbol{\mu}_y$) by the projected distance. In particular, Slab filters out points that satisfy $\langle \boldsymbol{x} - \boldsymbol{\mu}_y, \boldsymbol{\mu}_y - \boldsymbol{\mu}_{-y} \rangle \geq s_y$, where s_y is set as the q-th quantile (e.g., 95-th quantile as in Steinhardt et al. (2017); Koh et al. (2022)). Sphere works by filtering out points that satisfy $\|\boldsymbol{x} - \boldsymbol{\mu}_y\|_2 \geq r_y$, where r_y is similarly set as the q-th quantile (still set as 95-th in Steinhardt et al. (2017); Koh et al. (2022)). We will use the combination of Slab and Sphere defenses as an example in Section 5.6 to show how the effectiveness of data sanitization defenses can be explained by the relevant factor we identified for poisoning effectiveness.

We choose Slab and Sphere as the representative sanitization defenses in this dissertation as they were most commonly studied in prior works (Steinhardt et al., 2017; Koh et al., 2022; Diakonikolas et al., 2019), but Koh et al. (2022) also discussed other sanitization defenses that filter out outliers based on certain criteria pre- or post-model training. These defenses include *loss defense* that filters out points of higher loss with respect to the model trained on the poisoned dataset; *SVD defense* that projects the data points onto the top-k right singular vectors of the data feature matrix and filters out the outliers; *k-NN defense* that removes points far away from their k nearest neighbors. A more detailed description of these defenses can be referred from the original paper (Koh et al., 2022).

Chapter 3

Model-Targeted Poisoning Attack[1]

3.1 Introduction

To assess the practical risks from poisoning, different types of poisoning strategies are proposed for indiscriminate, subpopulation, and targeted attacks. Among the three, indiscriminate attacks are investigated with the most versatile attack methods, ranging from the typical influence/gradient method that converts the original bi-level optimization in Definition 2.1.2 into a single-level optimization using influence functions (Rousseeuw et al., 2011), online convex optimization strategy that leverages the min-max nature of indiscriminate poisoning attacks (Steinhardt et al., 2017; Koh et al., 2022) to model-targeted attacks that additionally uses a given target model to generate effective poisoning points efficiently with convex optimization (Koh et al., 2022). Prior to our work in this chapter, the subpopulation attack only considers the weaker random label-flipping attack (Jagielski et al., 2019). Later, a computationally efficient relaxation of the influence attack that is adapted for the subpopulation setting is proposed (Jagielski et al., 2021). The targeted poisoning attacks still mostly (Koh and Liang, 2017; Huang et al., 2020; Geiping et al., 2021) follow the idea of influence functions with some computationally efficient relaxations.

The proposed poisoning attacks (see details of the representative indiscriminate attacks in Section 2.2) have drawbacks in some aspects. The influence/gradient attacks can be slow due to the computation of inverse Hessian, and can also easily get stuck into bad local optima and lead to poor performance because they rely on local optimization techniques for the non-convex poisoning problems (Steinhardt et al., 2017; Koh

[1]This chapter is largely based on Fnu Suya, Saeed Mahloujifar, Anshuman Suri, David Evans, Yuan Tian, *Model-Targeted Poisoning Attacks with Provable Convergence*, in the Thirty-eighth International Conference on Machine Learning (ICML 2021).

et al., 2022). Furthermore, adapting gradient attacks to different poisoning settings might require redesigning the loss functions (Jagielski et al., 2021). On the other hand, attacks that leverage the min-max nature of indiscriminate poisoning attacks cannot be applied to other poisoning settings that do not naturally admit the min-max formulation for poisoning. In contrast, model-targeted attacks are flexible because they can be applied for diverse attack objectives (and maybe even beyond security-related attack objectives) simply by generating the corresponding target models and typically avoid the problem of bad local optima because they no longer rely on local optimization techniques (Koh et al., 2022). In particular, model-targeted poisoning attacks can be divided into two stages: 1) generating the target model that satisfies a certain attack objective (e.g., a specific error rate on the entire population or some target subpopulation) using simple heuristics such as label-flipping attacks (Koh et al., 2022) (see details in Section 2.2.2); 2) with a given a target model, attackers attempt to recover the target model by generating poisoning points, which can now be easier to optimize as attackers have now additional leverage from the given target model. However, the current model-targeted attacks (Koh et al., 2022; Mei and Zhu, 2015a) do not have convergence guarantee to the target models, making it unclear how reliably the poisoning attacks can achieve the encoded attack objectives in those target models.

Driven by the drawbacks of existing data poisoning attacks, in this chapter, we focus on designing an attack strategy that naturally applies to different attack objectives, especially for the considered indiscriminate and subpopulation settings, and hence follow the procedure of the model-targeted attacks. However, different from the existing model-targeted poisoning attacks, we aim to design an attack that has guaranteed convergence to the given target model (i.e., addressing the convergence problem of existing attacks in the second stage described above), so as to reliably achieve the encoded attack objective in the target model.

This chapter is organized as follows: we first provide details on the proposed model-targeted poisoning attack (Section 3.2) and then prove its convergence to the target model (Section 3.3.1). Next, we also provide a lower bound on the number of poisoning points needed to induce a given target model and use this lower bound to check the optimality of our attack (Section 3.3.3). Then, we validate the effectiveness of our proposed attack in Section 3.4 by empirically demonstrating the convergence to the target models and also the effectiveness in achieving their encoded attack objectives. In Section 3.5, we discuss the limitations and possible extensions of our work and conclude the chapter in Section 3.6.

3.2 Model-Targeted Poisoning Attacks

In this section, we focus on studying the attack strategy in the second stage of model-targeted poisoning attacks. Therefore, when we compare our proposed attack to the state-of-the-art model-targeted poisoning attack (Koh et al., 2022) in Section 3.4, we will use the current target model generation method by Koh et al. (2022) for a fair comparison in terms of the attack strategy to generate effective poisoning points. At the end of the section (Section 3.4.4), we will present a simple way to improve the target model generation method in Koh et al. (2022), which then provides target models of higher quality to enable more efficient model-targeted poisoning attacks.

The main idea of our attack, shown in Algorithm 5, 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 3.3.1 why similar loss distribution implies convergence.

Algorithm 5 Model Targeted Poisoning (MTP)

Input: $\mathcal{S}_c$, the loss functions (L and ℓ), target model $\hat{h}_{tar}$, regularization strength C_R

Output: $\mathcal{S}_p$

1: $\mathcal{S}_p = \emptyset$
2: **while** stopping criteria not met **do**
3: $\quad \hat{h}_t = \operatorname{argmin} L(h; \mathcal{S}_c \cup \mathcal{S}_p) + C_R \cdot R(h)$
4: $\quad (\boldsymbol{x}^*, y^*) = \operatorname{argmax}_{(\boldsymbol{x},y)\in\mathcal{C}} \ell(\hat{h}_t; \boldsymbol{x}, y) - \ell(\hat{h}_{tar}; \boldsymbol{x}, y)$
5: $\quad \mathcal{S}_p = \mathcal{S}_p \cup \{(\boldsymbol{x}^*, y^*)\}$
6: **end while**
7: **return** $\mathcal{S}_p$

Algorithm 5 requires the input of clean training set $\mathcal{S}_c$, the Loss function (L for a set of points and ℓ for individual point), and the target model $\hat{h}_{tar}$. The output from Algorithm 5 will be the set of poisoning points $\mathcal{S}_p$. The algorithm is simple: first, adversaries train the intermediate model $\hat{h}_t$ on the mixture of clean and poisoning points $\mathcal{S}_c \cup \mathcal{S}_p$ with $\mathcal{S}_p$ an empty set in the first iteration (Line 3). Then, it searches for the point that maximizes the loss difference between $\hat{h}_t$ and $\hat{h}_{tar}$ at iteration t (Line 4). After the point of

maximum loss difference is found, it is added to the poisoning set $\mathcal{S}_p$ (Line 5). The whole process repeats until the stopping condition is satisfied (Line 2). Algorithm 5 selects poisoning points in $\mathcal{S}_p$ sequentially in order. However, we assume the adversary has no control over the training order, so when the victim trains a model on $\mathcal{S}_c \cup \mathcal{S}_p$ the training set is shuffled randomly. We next provide details on the 3 possible stopping conditions, which are used in later experiments in this dissertation.

Stopping conditions. The stopping condition in Line 2 is flexible and can take 3 different forms: 1) the intermediate classifier $\hat{h}_t$ is closer to the target classifier (below a preset threshold ζ) in terms of the maximum loss difference, and more details regarding this distance metric will be introduced in Section 3.3.1; 2) adversary has some requirement on the accuracy and the algorithm terminates when $\hat{h}_t$ satisfies the accuracy requirement; 3) adversary has a budget T on the number of poisoning points, and the algorithm halts when the algorithm runs for total T iterations (e.g., when attacker has a fixed poisoning budget of $T = \epsilon|\mathcal{S}_c|$); In this chapter, we focus on producing a classifier close to the target model so as to better achieve the encoded objective in the given target model, and hence adopt the first stopping criterion that measures the distance with respect to the maximum loss difference, and report results based on this criterion in Section 3.4. For experiments in Chapter 4, we use the second criterion for convenience in visualizing the poisoning process and comparing different subpopulations with a clear success metric. For experiments in Chapter 5, we use the third criterion to better compare with the existing indiscriminate poisoning attacks (including but not limited to model-targeted attacks) that are often evaluated at a given poisoning ratio ϵ (i.e., generates $\epsilon|\mathcal{S}_c|$ number of poisoning points).

A nice property of Algorithm 5 is that the classifier $\hat{h}_p$ trained on $\mathcal{S}_c \cup \mathcal{S}_p$ asymptotically converges to $\hat{h}_{tar}$. 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 $(\boldsymbol{x}^*, y^*)$ into $\mathcal{S}_p$ each time. This reduces the overall iteration number and hence reduces the number of times retraining models. For simplicity in interpreting the results (Section 3.4), we do not use this in our experiments and add only one copy of $(\boldsymbol{x}^*, y^*)$ each iteration. However, we also tested the performance by adding two copies of $(\boldsymbol{x}^*, y^*)$ and find that the attack results are nearly the same while the efficiency is improved significantly. For example, for experiments on the 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.

3.3 Theorerical Results of the Proposed Attack

We first show the convergence of the proposed attack in Section 3.3.1, followed by its proof in Section 3.3.2. We then present the lower bound on the minimum number of poisoning points needed to induce a target model in Section 3.3.3 and then provide the proof in Section 3.3.4. We use this lower bound to check the optimality of our attack in Section 3.4.3. Finally, we show the special properties of the loss-based distance (Definition 3.3.1) for the commonly studied hinge loss in Section 3.3.5.

3.3.1 Convergence to the Target Model

In this section, we show the convergence of our proposed attack to the target model. But before proving the convergence of Algorithm 5, we will first define a general closeness measure based on their prediction performance to measure the distance of the induced model $\hat{h}_p$ trained on $\mathcal{S}_c \cup \mathcal{S}_p$ to the target model $\hat{h}_{tar}$. We will also use the definition of attainable models in Definition 2.1.1. Both definitions will be used to state the convergence theorem in Theorem 3.3.3.

Definition 3.3.1 (Loss-based distance and ζ-close)**.** For two models h_1 and h_2, a space $\mathcal{X} \times \mathcal{Y}$ and a loss $\ell(h; \boldsymbol{x}, y)$, we define *loss-based distance* $D_{\ell,\mathcal{X},\mathcal{Y}} \colon \mathcal{H} \times \mathcal{H} \to R$ as

$$D_{\ell,\mathcal{X},\mathcal{Y}}(h_1, h_2) = \max_{(\boldsymbol{x},y)\in\mathcal{X}\times\mathcal{Y}} \ell(h_1; \boldsymbol{x}, y) - \ell(h_2; \boldsymbol{x}, y),$$

and we say model h_1 is ζ-*close* to model h_2 when the loss-based distance from h_1 to h_2 is upper bounded by ζ.

Measuring model distance. We use loss-based distance to capture the "behavioral" distance between two models. Namely, if h_1 is ζ-close (as measured by loss-based distance) to h_2 and vice versa, then h_1 and h_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 h is attainable using ERM, no model could have a negative distance to it. For special loss function ℓ (e.g., hinge loss, see details in Corollary 3.3.15), two models can be close to each other in both directions when measured by the loss-based distance, which can be useful in providing

further analysis on the vulnerabilities to poisoning attacks. We formally define such notion as *bi-directional closeness* as below.

Definition 3.3.2 (bi-directional closeness)**.** For two models h_1 and h_2, a space $\mathcal{X} \times \mathcal{Y}$ and a loss $\ell(h; \boldsymbol{x}, y)$, we define model h_1 and h_2 has *bi-directional closeness* as if:

$$D_{l,\mathcal{X},\mathcal{Y}}(h_1, h_2) \leq \zeta \quad \text{and} \quad D_{l,\mathcal{X},\mathcal{Y}}(h_2, h_1) \leq O(\zeta).$$

When measuring the distance between two models h_1 and h_2, the loss-based distance can also be related to the conventional distance metrics such as the Manhattan Distance for some special loss functions. In Section 3.3.5 (deferred to the end of the chapter for clarity in presentation), we demonstrate how the loss-based distance is an $O(\zeta)$ upper bound to the ℓ_1-norm of difference between two models that are ζ-close with respect to loss-based distance for the special case of hinge loss, indicating that the loss based distance not only reflects similar predictions between the induced and target models but also the parameter closeness in some special cases such as hinge loss.

In the rest of the paper, we will use the terms ζ-close or ζ-closeness to denote that a model is ζ away from another model based on the loss-based distance. Theorem 3.3.3 uses the loss-based distance to establish that the attack of Algorithm 5 produces a model that converges to the target classifier:

Theorem 3.3.3. *For $\upsilon > 0$, if $\hat{h}_{tar}$ is a $C_R(1+\upsilon)$-attainable model, after at most T steps, Algorithm 5 with $\mathcal{C} = \mathcal{X} \times \mathcal{Y}$ will produce the poisoning set $\mathcal{S}_p$ so that a classifier trained on $\mathcal{S}_c \cup \mathcal{S}_p$ using the empirical risk minimization algorithm shown in Eq. (2.2) is ζ-close to $\hat{h}_{tar}$, with respect to loss-based distance, $D_{l,\mathcal{X},\mathcal{Y}}$, for*

$$\zeta = \frac{\alpha(T) + |\mathcal{S}_c| \cdot \left(L(\hat{h}_{tar}; \mathcal{S}_c) - L(\hat{h}_c; \mathcal{S}_c)\right)}{T \cdot \upsilon/1+\upsilon}$$

where $\alpha(T)$ is the regret of the Follow the Leader algorithm for a series of loss functions $\ell_i(\cdot) = \ell(\cdot, \boldsymbol{x}_i, y_i) + C_R \cdot R(\cdot)$ and $(\boldsymbol{x}_i, y_i)$ is the i-th poisoning point.

Remark 3.3.4. 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 $\zeta \leq O(\frac{\log T}{T})$ when the loss function is Lipschitz continuous and the regularizer $R(h)$ is strongly convex and $\zeta \to 0$ 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. The condition $\mathcal{C} = \mathcal{X} \times \mathcal{Y}$ is also important because the induced model is required to be similar to the target model in predicting any input from the space $\mathcal{X} \times \mathcal{Y}$. This condition can be easily satisfied in practice when there are no defenses deployed. However, in the presence of defenses, $\mathcal{C}$ may exclude certain (clean) points and the induced model may never be able to have similar predictions with the target model on those excluded points and hence the convergence property might be broken.

Implications of Theorem 3.3.3. The theorem says that the loss-based distance of the model trained on $\mathcal{S}_c \cup \mathcal{S}_p$ to the target model correlates to the loss difference between the target model and the clean model $\hat{h}_c$ (trained on $\mathcal{S}_c$) on $\mathcal{S}_c$, and correlates inversely with the number of poisoning points. Therefore, it implies 1) if the target classifier $\hat{h}_{tar}$ has a lower loss on $\mathcal{S}_c$, 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 justifies the motivation behind the heuristic method in Koh et al. (2022) to select a target classifier with a lower loss on the clean training data. In addition, it also helps us find the subpopulation property that is related to the subpopulation susceptibility in Chapter 4. For the indiscriminate attack scenario, we also improve the current label-flipping method to generate a target model (Koh et al., 2022) by adaptively updating the intermediate models and producing a (final) target classifier with a much lower loss on $\mathcal{S}_c$, which then enables more effective model-targeted attacks. This helps to empirically validate our theorem. Details of the original and improved heuristic approach and their relevant experiments are in Section 3.4.4.

Another important (and broader) implication of our theorem is, the convergence to the target model in terms of the maximum loss difference ensures the induced model has similar prediction behavior to the target model. Therefore, in the broader sense, the proposed attack might be applicable to a wide set of attack objectives in other trustworthy aspects of machine learning such as privacy and fairness. The attackers only need to generate the target models that satisfy those broad sets of attack objectives.

3.3.2 Proof of the Convergence

In this section, we provide the proof of the convergence of the algorithm. Before proving the main theorem, we first prove the following two lemmas that characterize the relationship between the maximum loss difference and the difference of the regularization terms.

Lemma 3.3.5. *Let $\hat{h}_1$ be a C_R-attainable model for some $C_R > 0$, then for any model h_2 we have*

$$\sup_{\boldsymbol{x},y} \big(\ell(h_2;\boldsymbol{x},y) - \ell(\hat{h}_1;\boldsymbol{x},y)\big) > C_R \cdot \big(R(\hat{h}_1) - R(h_2)\big).$$

Proof. Consider an attainable model $\hat{h}_1$ and any model h_2, and let $\mathcal{S}_1$ to be the training set that leads the training algorithm to produce $\hat{h}_1$. Namely,

$$\hat{h}_1 = \operatorname*{argmin}_h L(h;\mathcal{S}_1) + C_R \cdot R(h)$$

Since $\hat{h}_1$ minimizes the loss on $\mathcal{S}_1$ uniquely, we have

$$L(h_2;\mathcal{S}_1) + C_R \cdot R(h_2) > L(\hat{h}_1;\mathcal{S}_1) + C_R \cdot R(\hat{h}_1)$$

By rearranging the above inequality and by an averaging argument, we have

$$\sup_{\boldsymbol{x},y} \big(\ell(h_2;\boldsymbol{x},y) - \ell(\hat{h}_1;\boldsymbol{x},y)\big) \geq L(h_2;\mathcal{S}_1) - L(\hat{h}_1;\mathcal{S}_1) > C_R \cdot \big(R(\hat{h}_1) - R(h_2)\big).$$

□

Lemma 3.3.6. *For $\upsilon > 0$, let F be the family of all $(C_R(1+\upsilon))$-attainable models. For any $\hat{h}_1 \in F$ and for all h_2 we have*

$$\sup_{\boldsymbol{x},y} \big(\ell(h_2;\boldsymbol{x},y) - \ell(\hat{h}_1;\boldsymbol{x},y)\big) + C_R(R(h_2) - R(\hat{h}_1)) > \frac{\upsilon}{1+\upsilon} \cdot \sup_{\boldsymbol{x},y} \big(\ell(h_2;\boldsymbol{x},y) - \ell(\hat{h}_1;\boldsymbol{x},y)\big).$$

Proof. By Lemma 3.3.5 we have

$$\sup_{\boldsymbol{x},y} \big(\ell(h_2;\boldsymbol{x},y) - \ell(\hat{h}_1;\boldsymbol{x},y)\big) + C_R(1+\upsilon)(R(h_2) - R(\hat{h}_1)) > 0.$$

Now by adding $\upsilon \sup_{\boldsymbol{x},y} \big(\ell(h_2;\boldsymbol{x},y) - \ell(\hat{h}_1;\boldsymbol{x},y)\big)$ to both sides we have

$$(1+\upsilon)\Big(\sup_{\boldsymbol{x},y} \big(\ell(h_2;\boldsymbol{x},y) - \ell(\hat{h}_1;\boldsymbol{x},y)\big) + C_R(R(h_2) - R(\hat{h}_1))\Big) > \upsilon \sup_{\boldsymbol{x},y} \big(\ell(h_2;\boldsymbol{x},y) - \ell(\hat{h}_1;\boldsymbol{x},y)\big)$$

which implies

$$\Big(\sup_{\boldsymbol{x},y}\big(\ell(h_2;\boldsymbol{x},y)-\ell(\hat{h}_1;\boldsymbol{x},y)\big)+C_R(R(h_2)-R(\hat{h}_1))\Big)>\frac{\upsilon}{1+\upsilon}\sup_{\boldsymbol{x},y}\big(\ell(h_2;\boldsymbol{x},y)-\ell(\hat{h}_1;\boldsymbol{x},y)\big)$$

□

With Definition 2.1.1 and the lemmas, we are ready to prove Theorem 3.3.3 in Section 3.3.1 (restated below for convenience):

Theorem 3.3.3. *For $\upsilon>0$, if $\hat{h}_{tar}$ is a $C_R(1+\upsilon)$-attainable model, after at most T steps Algorithm 5 with $\mathcal{C}=\mathcal{X}\times\mathcal{Y}$ will produce the poisoning set $\mathcal{S}_p$ so that the classifier trained on $\mathcal{S}_c\cup\mathcal{S}_p$ using Eq. (2.2) is ζ-close to $\hat{h}_{tar}$, with respect to loss-based distance, $D_{l,\mathcal{X},\mathcal{Y}}$, for*

$$\zeta=\frac{\alpha(T)+|\mathcal{S}_c|\cdot\big(L(\hat{h}_{tar};\mathcal{S}_c)-L(\hat{h}_c;\mathcal{S}_c)\big)}{T\cdot{}^{\upsilon}\!/_{1+\upsilon}}$$

where $\alpha(T)$ is the regret of the Follow the Leader algorithm for a series of loss functions $\ell_i(\cdot)=\ell(\cdot,\boldsymbol{x}_i,y_i)+C_R\cdot R(\cdot)$ and $(\boldsymbol{x}_i,y_i)$ is the ith poisoning point.

The goal of the adversary is to get ζ-close to $\hat{h}_{tar}$ (in terms of the loss-based distance) by injecting (potentially few) number of poisoned training data. The algorithm is, in essence, an online learning problem, and we transform Algorithm 5 into the form of a standard online learning problem. Specifically, we adopt the *Follow the Leader* (FTL) framework to describe Algorithm 5 in the language of a standard online learning problem. We first describe the online learning setting considered in this paper and the notion of regret.

Definition 3.3.7. Let $\mathcal{L}$ be a class of loss functions, $\mathcal{H}$ set of possible models, $A\colon(\mathcal{H}\times\mathcal{L})^*\to\mathcal{H}$ an online learner and $STG\colon(\mathcal{H}\times\mathcal{L})^*\times\mathcal{H}\to\mathcal{L}$ a strategy for picking loss functions in different rounds of online learning (adversarial environment in the context of online convex optimization). We use $\mathsf{Regret}(A,STG,T)$ to denote the regret of A against STG, in T rounds. Namely,

$$\mathsf{Regret}(A,STG,T)=\sum_{j=0}^{T}\ell_j(\hat{h}_j)-\min_{h\in\mathcal{H}}\sum_{j=0}^{T}\ell_j(h)$$

where

$$\hat{h}_i=A\big((\hat{h}_0,\ell_0),\ldots,(\hat{h}_{i-1},\ell_{i-1})\big)\quad\text{and}\quad l_i=STG\big((\hat{h}_0,\ell_0),\ldots,(\hat{h}_{i-1},\ell_{i-1}),\hat{h}_i\big).$$

With the online learning problem set up, we proceed to the main proof, which first describes Algorithm 5 in the FTL framework.

Proof of Theorem 3.3.3. The FTL framework proceeds by solving all the functions incurred during the previous online optimization steps, namely, $A_{\mathsf{FTL}}((\hat{h}_0, \ell_0), \ldots, (\hat{h}_i, \ell_i)) = \operatorname{argmin}_{h\in\mathcal{H}} \sum_{j=0}^{i} \ell_i(h)$.

Next, we describe how we design the i-th loss function ℓ_i in each round of the online optimization. For the first choice, A_{FTL} chooses a random model $\hat{h}_0 \in \mathcal{H}$. In the first round (round 0), $STG_{\hat{h}_{tar}}$ uses the clean training set $\mathcal{S}_c$ and the loss is set as

$$STG_{\hat{h}_{tar}}(\hat{h}_0) = \ell_0(h) = N \cdot L(h; \mathcal{S}_c) + NC_R \cdot R(h), \quad N = |\mathcal{S}_c|.$$

According to the FTL framework, A_{FTL} returns a model that minimizes the loss on the clean training set $\mathcal{S}_c$ using the structural empirical risk minimization. For the subsequent iterations ($i \geq 1$), the loss functions are defined as, given the latest model $\hat{h}_i$, $STG_{\hat{h}_{tar}}$ first finds $(\boldsymbol{x}_i^*, y_i^*)$ that maximizes the loss difference between $\hat{h}_i$ and a target model $\hat{h}_{tar}$. Namely,

$$(\boldsymbol{x}_i^*, y_i^*) = \operatorname*{argmax}_{(\boldsymbol{x},y)} \ell(\hat{h}_i; \boldsymbol{x}, y) - \ell(\hat{h}_{tar}; \boldsymbol{x}, y)$$

and then chooses the ith loss function as follows:

$$STG_{\hat{h}_{tar}}\big((\hat{h}_0, \ell_0), \ldots, (\hat{h}_{i-1}, \ell_{i-1}), \hat{h}_i\big) = \ell_i(h) = \ell(h; \boldsymbol{x}_i^*, y_i^*) + C_R \cdot R(h).$$

Now we will see how the FTL framework behaves when working on these loss functions at different iterations. We use $\mathcal{S}_p^i$ to denote the set $\{(\boldsymbol{x}_1^*, y_1^*), \ldots, (\boldsymbol{x}_i^*, y_i^*)\}$. We have

$$\begin{aligned}
\hat{h}_i = A_{\mathsf{FTL}}((\hat{h}_0, \ell_0), \ldots, (\hat{h}_{i-1}, \ell_{i-1})) &= \operatorname*{argmin}_{h\in\mathcal{H}} \sum_{j=0}^{i-1} l_j(h) \\
&= \operatorname*{argmin}_{h\in\mathcal{H}} \; N \cdot L(h; \mathcal{S}_c) + NC_R \cdot R(h) \\
&\quad + \sum_{j=1}^{i-1} \ell(h; \boldsymbol{x}_j^*, y_j^*) + C_R \cdot R(h) \\
&= \operatorname*{argmin}_{h\in\mathcal{H}} \; (N + i - 1) \cdot L(h; \mathcal{S}_c \cup \mathcal{S}_p^{i-1}) + (N + i - 1)C_R \cdot R(h) \\
&= \operatorname*{argmin}_{h\in\mathcal{H}} \; L(h; \mathcal{S}_c \cup \mathcal{S}_p^{i-1}) + C_R \cdot R(h)
\end{aligned}$$

This means that A_{FTL} algorithm, at each step, trains a new model over the combination of clean data and poison data so far ($i-1$ number of poisons). Now we want to see what is the translation of the $\mathsf{Regret}(A_{\mathsf{FTL}}, STG_{\hat{h}_{tar}}, T)$. If we can prove an upper bound on regret, namely if we show $\mathsf{Regret}(A_{\mathsf{FTL}}, STG_{\hat{h}_{tar}}, T) \leq \alpha(T)$ for some function α, then we have

$$\sum_{j=0}^{T} l_j(\hat{h}_j) - \sum_{j=0}^{T} \ell_j(\hat{h}_{tar}) \leq \sum_{j=0}^{T} \ell_j(\hat{h}_j) - \min_{h\in\mathcal{H}} \sum_{j=0}^{T} l_j(h) \leq \alpha(T)$$

which implies

$$\begin{aligned}
\sum_{j=0}^{T} l_j(\hat{h}_j) - \sum_{j=0}^{T} \ell_j(\hat{h}_{tar}) &= N \cdot \big(L(\hat{h}_c;\mathcal{S}_c) - L(\hat{h}_{tar};\mathcal{S}_c)\big) + NC_R \cdot (R(\hat{h}_c) - R(\hat{h}_{tar})) \\
&\quad + \sum_{j=1}^{T} l_j(\hat{h}_j) - \sum_{j=1}^{T} \ell_j(\hat{h}_{tar}) \\
&= N \cdot \big(L(\hat{h}_c;\mathcal{S}_c) - L(\hat{h}_{tar};\mathcal{S}_c)\big) + NC_R \cdot (R(\hat{h}_c) - R(\hat{h}_{tar})) \\
&\quad + \sum_{j=1}^{T} \big[\max_{\boldsymbol{x},y} \big(\ell(\hat{h}_j;\boldsymbol{x},y) - \ell(\hat{h}_{tar};\boldsymbol{x},y)\big) + C_R \cdot (R(\hat{h}_j) - R(\hat{h}_{tar}))\big] \\
&\leq \alpha(T).
\end{aligned}$$

Therefore we have

$$\begin{aligned}
\sum_{j=1}^{T} \big[\max_{\boldsymbol{x},y} \big(\ell(\hat{h}_j;\boldsymbol{x},y) - \ell(\hat{h}_{tar};\boldsymbol{x},y)\big) + C_R \cdot (R(\hat{h}_j) - R(\hat{h}_{tar}))\big] &\leq \alpha(T) + N \cdot \big(L(\hat{h}_{tar};\mathcal{S}_c) - L(\hat{h}_c;\mathcal{S}_c)\big) \\
&\quad + NC_R \cdot (R(\hat{h}_{tar}) - R(\hat{h}_c))
\end{aligned}$$

Based on Lemma 3.3.6, we further have

$$\sum_{j=1}^{T} \frac{\upsilon}{1+\upsilon} \cdot \big(\max_{\boldsymbol{x},y} \ell(\hat{h}_j;\boldsymbol{x},y) - \ell(\hat{h}_{tar};\boldsymbol{x},y)\big) \leq \alpha(T) + N \cdot \big(L(\hat{h}_{tar};\mathcal{S}_c) - L(\hat{h}_c;\mathcal{S}_c)\big) + NC_R \cdot (R(\hat{h}_{tar}) - R(\hat{h}_c))$$

The above inequality states that the average of the maximum loss difference in all previous rounds is bounded from above. Therefore, we know that among the T iterations, there exists an iteration $j^* \in [T]$ (with the lowest maximum loss difference) such that the maximum loss difference of $\hat{h}_{j^*}$ is ζ-close to $\hat{h}_{tar}$ with respect to the loss-based distance where

$$\zeta = \frac{\alpha(T) + N \cdot \big(L(\hat{h}_{tar};\mathcal{S}_c) - L(\hat{h}_c;\mathcal{S}_c)\big) + N \cdot C_R \cdot (R(\hat{h}_{tar}) - R(\hat{h}_c))}{T \cdot \upsilon/1+\upsilon}.$$

□

Theorem 3.3.3 characterizes the dependencies of ζ on $\alpha(T)$ and the constant term $N \cdot \big(L(\hat{h}_{tar}; \mathcal{S}_c) - L(\hat{h}_c; \mathcal{S}_c)\big) + NC_R \cdot (R(\hat{h}_{tar}) - R(\hat{h}_c))$. To show the convergence of Algorithm 5, we need to ensure $\zeta \to 0$ when $T \to +\infty$, which implies we need to show $\alpha(T) \leq O(T)$. The following remark (restating Remark 3.3.4 in Section 3.3.1) and its proof shows the desired convergence.

Remark 3.3.4. *Sublinear regret bounds for FTL 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 $\zeta \leq O(\frac{\log T}{T})$ when the loss function is Lipschitz continuous and the regularizer $R(h)$ is strongly convex and $\zeta \to 0$ 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.*

Our FTL framework formulation can utilize the existing logarithmic regret bound of the adaptive FTL algorithm when the objective functions are strongly convex with respect to some norm $\|\cdot\|$, as illustrated in Section 3.6 in McMahan (2017). For clarity in the presentation, we first restate their related results below.

Setting 1 (Setting 1 in McMahan (2017))**.** Given a sequence of objective loss functions $f_1, f_2, ..., f_i$ and a sequence of incremental regularization functions $r_0, r_1, ..., r_i$ we consider an algorithm that selects the response point based on

$$\boldsymbol{\theta}_1 = \operatorname*{argmin}_{\boldsymbol{\theta} \in \mathbb{R}^d} r_0(\boldsymbol{\theta})$$
$$\boldsymbol{\theta}_{i+1} = \operatorname*{argmin}_{\boldsymbol{\theta} \in \mathbb{R}^d} \sum_{j=1}^{i} f_j(\boldsymbol{\theta}) + r_j(\boldsymbol{\theta}) + r_0(\boldsymbol{\theta}), \text{for } i = 1, 2, ...$$

We simplify the summation notation with $f_{1:i}(\boldsymbol{\theta}) = \sum_{j=1}^{i} f_j(\boldsymbol{\theta})$. Assume that r_i is a convex function and satisfies $r_i(\boldsymbol{\theta}) \geq 0$ for $i \in \{0, 1, 2, ...\}$, against a sequence of convex loss functions $f_i : \mathbb{R}^d \to R \cup \{\infty\}$. Further, letting $l_{0:i} = r_{0:i} + f_{1:i}$ we assume dom $l_{0:i}$ is non-empty. Recalling $\boldsymbol{\theta}_i = \operatorname{argmin}_{\boldsymbol{\theta}} l_{0:i-1}(\boldsymbol{\theta})$, we further assume $\partial f_i(\boldsymbol{\theta}_i)$ is non-empty. We denote the dual norm of a norm $\|\cdot\|$ as $\|\cdot\|_*$.

Theorem 3.3.8 (Restatement of Theorem 1 in McMahan (2017))**.** *Consider Setting 1, and suppose the r_i are chosen such that $r_{0:i} + f_{1:i+1}$ is 1-strongly-convex w.r.t. some norm $\|\cdot\|_{(i)}$.. If we define the regret of the*

algorithm with respect to a selected point $\boldsymbol{\theta}^*$ *as*

$$\mathsf{Regret}_T(\boldsymbol{\theta}^*, f_i) \equiv \sum_{i=1}^{T} f_i(\boldsymbol{\theta}_i) - \sum_{i=1}^{T} f_i(\boldsymbol{\theta}^*).$$

Then, for any $\boldsymbol{\theta}^* \in \mathbb{R}^d$ *and for any* $T > 0$*, with* $g_i \in \partial f_i(\boldsymbol{\theta}_i)$*, we have*

$$\mathsf{Regret}_T(\boldsymbol{\theta}^*, f_i) \leq r_{0:T-1}(\boldsymbol{\theta}^*) + \frac{1}{2}\|g_i\|^2_{(i-1),*}$$

Here, the $\boldsymbol{\theta} \in \mathbb{R}^d$ corresponds to the weight parameter of the hypothesis $h_{\boldsymbol{\theta}}$ in our poisoning setting.

Corollary 3.3.9 (Formalization of FTL result in Section 3.6 in McMahan (2017))**.** *In the FTL framework (no individual regularizer is used in the optimization procedure), suppose each loss function* f_i *is 1-strongly convex w.r.t. a norm* $\|\cdot\|$*, then we have*

$$\mathsf{Regret}_T(\boldsymbol{\theta}^*, f_i) \leq \frac{1}{2}\sum_{i=1}^{T}\frac{1}{i}\|g_i\|^2_* \leq \frac{G^2}{2}(1 + \log T)$$

with $\|g_i\|_* \leq G$*.*

Proof. The following proof is a restatement of the proof in Section 3.6 in McMahan (2017). The proof follows from Theorem 3.3.8. Since we are considering the FTL framework, let $r_i(\boldsymbol{\theta}) = 0$ for all i and define $\|\boldsymbol{\theta}\|_{(i)} = \sqrt{i}\|\boldsymbol{\theta}\|$. Observe that $l_{0:i}$ (i.e., $f_{1:i}$) is 1-strongly convex with respect to $\|\boldsymbol{\theta}\|_{(i)}$ (Lemma 3 in McMahan (2017)), and we have $\|\boldsymbol{\theta}\|_{(i),*} = \frac{1}{\sqrt{i}}\|\boldsymbol{\theta}\|_*$. Then by applying Theorem 3.3.8, we have

$$\mathsf{Regret}_T(\boldsymbol{\theta}^*, f_i) \leq \frac{1}{2}\sum_{i=1}^{T}\|g_i\|^2_{(i),*} = \frac{1}{2}\sum_{i=1}^{T}\frac{1}{i}\|g_i\|^2_*$$

Based on the inequality of $\sum_{i=1}^{T} 1/i \leq 1 + \log T$ and if we further assume $\|g_i\|_* \leq G$, then we can have

$$\frac{1}{2}\sum_{i=1}^{T}\frac{1}{i}\|g_i\|^2_* \leq \frac{G^2}{2}(1 + \log T)$$

□

Proof of Remark 3.3.4. We will prove the logarithmic regret bound in Remark 3.3.4 utilizing Corollary 3.3.9. First of all, our online learning process fits into Setting 1 using the considered loss functions in our setting.

Specifically, we set $r_i(h) = 0$ for all i. For $f_i(h)$, when $1 \leq i \leq N$, we set $f_i(h) = L(h; \mathcal{S}_c) + C_R \cdot R(h)$ (evenly distributing the term $N \cdot L(h; \mathcal{S}_c) + NC_R \cdot R(h)$ across N iterations) and when $i \geq N+1$, we set $f_i(h) = \ell_{i-N}(h)$. Details of ℓ_i can be referred from the proof of Theorem 3.3.3. Therefore, f_i is 1-strongly convex with respect to a norm $\|\cdot\|$ (the norm is determined by the regularizer $R(h_{\boldsymbol{\theta}})$ and C_R). Further, $\ell_{0:i}(h) = f_{1:N+i}(h)$. In addition, the assumption that dom $l_{0:i}$ is non-empty in Setting 1 means if we train a classifier on the poisoned data set, we can always return a model and hence the assumption is satisfied. The assumption of the existence of subgradient $\partial f_i(h_i)$ in Setting 1 is also satisfied by the poisoning attack scenario.

The logarithmic regret of $\mathsf{Regret}(A_{\mathsf{FTL}}, STG_{\hat{h}_{tar}}, T)$ of our algorithm then follows from the result of $\mathsf{Regret}_T(h^*, f_i)$ in Corollary 3.3.9. Specifically, $\ell_{0:i}(h_{\boldsymbol{\theta}}) = f_{1:N+i}(h_{\boldsymbol{\theta}})$ is 1-strongly convex to norm $\|\cdot\|_i = \sqrt{N+i}\|\cdot\|$ and since we assume the loss function is G-Lipschitz, we have $\|g_i\|_* \leq G$. Therefore, we have the logarithmic regret bound as:

$$\mathsf{Regret}(A_{\mathsf{FTL}}, STG_{\hat{h}_{tar}}, T) \leq \alpha(T) = \frac{1}{2}\sum_{i=1}^{T}\frac{1}{i+N}\|g_i\|_*^2 \leq \frac{1}{2}\sum_{i=1}^{T}\frac{1}{i}\|g_i\|_*^2 \leq \frac{G^2}{2}(1+\log T) \leq O(\log T).$$

□

3.3.3 Lower Bound on Necessary Number of Poisoning Points

In this section, we show how to compute a lower bound on the number of poisoning points that are needed to induce a particular target model $\hat{h}_{tar}$.

We first provide the lower bound on the number of poisoning points required for producing the target classifier in the injection-only setting (Theorem 3.3.10) and then explain how the lower bound estimation can be incorporated into Algorithm 5. 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 h with lower loss compared to $\hat{h}_{tar}$ and hence the target classifier cannot be attained. The full proof of the theorem and some extensions are in Section 3.3.4.

Theorem 3.3.10 (Lower Bound)**.** *Given a target classifier $\hat{h}_{tar}$, clean training data $\mathcal{S}_c$ with $|\mathcal{S}_c| = N$, to reproduce $\hat{h}_{tar}$ by adding the poisoning set $\mathcal{S}_p$ into $\mathcal{S}_c$, the number of poisoning points $|\mathcal{S}_p|$ cannot be lower than*

$$\sup_h z(h) = \frac{N \cdot \big(L(\hat{h}_{tar}; \mathcal{S}_c) - L(h; \mathcal{S}_c)\big) + NC_R(R(\hat{h}_{tar}) - R(h))}{\sup_{\boldsymbol{x},y}\big(\ell(h; \boldsymbol{x}, y) - \ell(\hat{h}_{tar}; \boldsymbol{x}, y)\big) + C_R(R(h) - R(\hat{h}_{tar}))}.$$

Corollary 3.3.11. *If we further assume bi-directional closeness in the loss-based distance, we can also derive the lower bound on the number of poisoning points needed to induce models that are ζ-close to the target model. More precisely, if h_1 being ζ-close to h_2 implies that h_2 is also $k \cdot \zeta$ close to h_1, then we have,*

$$\sup_h z'(h) = \frac{N \cdot \big(L(\hat{h}_{tar}; \mathcal{S}_c) - L(h; \mathcal{S}_c)\big) - NC_R \cdot R^* - Nk\zeta}{\sup_{\boldsymbol{x},y} \big(\ell(h; \boldsymbol{x}, y) - \ell(\hat{h}_{tar}; \boldsymbol{x}, y)\big) + C_R \cdot R^* + k\zeta}.$$

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

The formula for the lower bound in Theorem 3.3.10 (and also the lower bound in Corollary 3.3.11) can be easily incorporated into Algorithm 5 to obtain a tighter theoretical lower bound. We simply need to check all the intermediate classifiers $\hat{h}_t$ produced during the attack process and replace h with $\hat{h}_t$, and the lower bound can be computed for the pair of $\hat{h}_t$ and $\hat{h}_{tar}$. Algorithm 5 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 3.1 and Table 3.2 in Section 3.4. We note that, for unbounded loss difference, the lower bounds of Theorem 3.3.10 and Corollary 3.3.11 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.

Our lower bound is generally applicable to any loss functions given that we are able to obtain the global minimizer (which can be hard for non-convex losses in practice), but requires identifying proper models h, which is usually obtained by iteratively training models in practice (e.g., using $\hat{h}_t$ in Algorithm 5 as mentioned above). If we further restrict the models to be linear models under special convex losses such as hinge loss and logistic loss (and also leverage the first-order optimality condition with respect to the target model), the lower bound can be computed in closed-form without repeatedly training models, which is demonstrated by the follow-up work (Lu et al., 2023) to ours.

Implications of our lower bound. First, our lower bound complements Theorem 2.2.1 (proven in Koh et al. (2022)), which states that any target model $\hat{h}_{tar}$ that is attainable using a training set $\mathcal{S}_p$ (Definition 2.1.1) can be similarly attained using a dataset consisting only two unique points with size less than or equal to $|\mathcal{S}_p|$. We complement this result by showing that the size of the training set that consists of only two

unique points cannot be arbitrarily small, in fact, cannot be smaller than the lower bound provided in Theorem 3.3.10.

This lower bound is also implicitly related to the effectiveness of data poisoning attacks that achieve a certain attack goal: our theorem describes the necessary amount of poisoning points that are needed to induce a target model and by checking all the target models that satisfy the attack goal, and taking the minimum over all the necessary numbers of poisoning points for the considered target models, one can obtain the necessary amount of poisoning points needed to achieve the attack goal.

3.3.4 Proof of the Lower Bound on Necessary Poisoning

In this section, we provide proof of the lower bound in Theorem 3.3.10 that characterizes the hardness in inducing a particular target model. The main intuition behind the theorem is, when the number of poisoning points added to the clean training set is lower than the certified lower bound, for the structural empirical risk minimization problem (in (2.2)), the target classifier will always have higher loss than another classifier and hence cannot be achieved.

Proof. We first show that for all models h, we can derive a lower bound on the number of poisoning points required to get $\hat{h}_{tar}$. Then since these lower bounds all hold, we can take the maximum over all of them and get a valid lower bound, which is also why in Algorithm 5, we return the highest lower bound computed. We first show that for any model h, the minimum number of poisoning points cannot be lower than

$$z(h) = \frac{N \cdot \big(L(\hat{h}_{tar}; \mathcal{S}_c) - L(h; \mathcal{S}_c)\big) + NC_R(R(\hat{h}_{tar}) - R(h))}{\sup_{\boldsymbol{x},y} \big(\ell(h; \boldsymbol{x}, y) - \ell(\hat{h}_{tar}; \boldsymbol{x}, y)\big) + C_R(R(h) - R(\hat{h}_{tar}))}.$$

Let us denote the point corresponding to the supremum of the loss difference between h and $\hat{h}_{tar}$ as $(\boldsymbol{x}^*, y^*)$ [2]. Namely, $\ell(h; \boldsymbol{x}^*, y^*) - \ell(\hat{h}_{tar}; \boldsymbol{x}^*, y^*) = \sup_{\boldsymbol{x},y} \big(\ell(h; \boldsymbol{x}, y) - \ell(\hat{h}_{tar}; \boldsymbol{x}, y)\big)$. Now, suppose we can obtain $\hat{h}_{tar}$ with a lower number of poisoning points $\underline{z} < z(h)$. Assume there is a poisoning set $\mathcal{S}_p$ with size $\underline{z}$ such that when added to $\mathcal{S}_c$ would result in $\hat{h}_{tar}$. Based on Lemma 3.3.5 we have

$$\sup_{\boldsymbol{x},y} \big(\ell(h; \boldsymbol{x}, y) - \ell(\hat{h}_{tar}; \boldsymbol{x}, y)\big) \geq L(h; \mathcal{S}_c \cup \mathcal{S}_p) - L(\hat{h}_{tar}; \mathcal{S}_c \cup \mathcal{S}_p) > C_R \cdot \big(R(\hat{h}_{tar}) - R(h)\big),$$

[2] In practice, the data space $\mathcal{X}$ is a closed convex set and hence, we can find (x^*, y^*) using convex optimization. In other words, as we will see in experiments in Section 3.4, calculating the lower bound is possible in practical scenarios.

implying $\sup_{\boldsymbol{x},y}\big(\ell(h;\boldsymbol{x},y)-\ell(\hat{h}_{tar};\boldsymbol{x},y)\big)+C_R\cdot(R(h)-R(\hat{h}_{tar}))>0$. Based on the assumption that $\underline{z}<z(h)$, and the fact that $\sup_{\boldsymbol{x},y}\big(\ell(h;\boldsymbol{x},y)-\ell(\hat{h}_{tar};\boldsymbol{x},y)\big)+C_R\cdot(R(h)-R(\hat{h}_{tar}))>0$, we have

$$\begin{aligned}
&\underline{z}\cdot\big(\ell(h;\boldsymbol{x}^*,y^*)-\ell(\hat{h}_{tar};\boldsymbol{x}^*,y^*)+C_R(R(h)-R(\hat{h}_{tar}))\big)\\
&<z(h)\cdot\big(\ell(h;\boldsymbol{x}^*,y^*)-\ell(\hat{h}_{tar};\boldsymbol{x}^*,y^*)+C_R(R(h)-R(\hat{h}_{tar}))\big)\\
&=N\cdot\big(L(\hat{h}_{tar};\mathcal{S}_c)-L(h;\mathcal{S}_c)\big)+NC_R(R(\hat{h}_{tar})-R(h)).
\end{aligned}$$

where the equality is based on the definition of $z(h)$. On the other hand, by definition of $(\boldsymbol{x}^*,y^*)$ for any $\mathcal{S}_p$ of size $\underline{z}$, we have

$$\begin{aligned}
\underline{z}\cdot\big(L(h;\mathcal{S}_p)-L(\hat{h}_{tar},\mathcal{S}_p)\big)+\underline{z}\cdot(C_R\cdot R(h)-C_R\cdot R(\hat{h}_{tar}))\leq\underline{z}\cdot\big(&\ell(h;\boldsymbol{x}^*,y^*)-\ell(\hat{h}_{tar};\boldsymbol{x}^*,y^*)\\
&+C_R(R(h)-R(\hat{h}_{tar}))\big).
\end{aligned}$$

The above two inequalities imply that for any set $\mathcal{S}_p$ with size $\underline{z}$ we have

$$L(h;\mathcal{S}_c\cup\mathcal{S}_p)+C_R\cdot R(h)<L(\hat{h}_{tar};\mathcal{S}_c\cup\mathcal{S}_p)+C_R\cdot R(\hat{h}_{tar}).$$

which indicates that adding $\mathcal{S}_p$ poisoning points into the training set $\mathcal{S}_c$, the model h has lower loss compared to $\hat{h}_{tar}$, which is a contradiction to the assumption that $\hat{h}_{tar}$ has the lowest loss on $\mathcal{S}_c\cup\mathcal{S}_p$ and can be achieved. Now, since $\hat{h}_{tar}$ needs to have lower loss on $\mathcal{S}_c\cup\mathcal{S}_p$ compared to any classifier $h\in\mathcal{H}$, the best lower bound is the supremum over all models in the hypothesis space $\mathcal{H}$. □

Corollary 3.3.11. *If we further assume bi-directional closeness in the loss-based distance, we can also derive the lower bound on the number of poisoning points needed to induce models that are ζ-close to the target model. More precisely, if h_1 being ζ-close to h_2 implies that h_2 is also $k\cdot\zeta$ close to h_1, then we have,*

$$\sup_h z'(h)=\frac{N\cdot\big(L(\hat{h}_{tar};\mathcal{S}_c)-L(h;\mathcal{S}_c)\big)-NC_R\cdot R^*-Nk\zeta}{\sup_{\boldsymbol{x},y}\big(\ell(h;\boldsymbol{x},y)-\ell(\hat{h}_{tar};\boldsymbol{x},y)\big)+C_R\cdot R^*+k\zeta}.$$

where R^* is an upper bound on the nonnegative regularizer $R(h)$.

Proof of Corollary 4.2.1. The lower bound for all ζ-close models to the target classifier is given exactly as follows:

$$\inf_{\|h'-\hat{h}_{tar}\|_{\mathcal{S}_{\ell,\mathcal{X},\mathcal{Y}}}\leq\zeta} \sup_{h} \left(z(h,h') = \frac{N\cdot\big(L(h';\mathcal{S}_c)-L(h;\mathcal{S}_c)\big)+NC_R(R(h')-R(h))}{\sup_{\boldsymbol{x},y}\big(\ell(h;\boldsymbol{x},y)-\ell(h';\boldsymbol{x},y)\big)+C_R(R(h)-R(h'))}\right),$$

where $\inf_{\|h'-\hat{h}_{tar}\|_{\mathcal{S}_{\ell,\mathcal{X},\mathcal{Y}}}\leq\zeta}$ denotes h' is ζ-close to $\hat{h}_{tar}$ in the loss-based distance. However, the formulation above is a min-max optimization problem and hard to analytically compute the lower bound by plugging the lower bound formula into Algorithm 5. Therefore, we need to make several relaxations such that the lower bound is computable. For any model h' that is ζ-close to $\hat{h}_{tar}$, based on the bi-directional assumption, then $\hat{h}_{tar}$ is $k\zeta$-close to h'. Therefore, we have

$$N\big(L(h';\mathcal{S}_c)-L(h;\mathcal{S}_c)\big)=N\big(L(h';\mathcal{S}_c)-L(\hat{h}_{tar};\mathcal{S}_c)+L(\hat{h}_{tar};\mathcal{S}_c)-L(h;\mathcal{S}_c)\big)\geq -Nk\zeta+N\big(L(\hat{h}_{tar};\mathcal{S}_c)-L(h;\mathcal{S}_c)\big)$$

and

$$\begin{aligned}\sup_{\boldsymbol{x},y}\big(\ell(h;\boldsymbol{x},y)-\ell(h',\boldsymbol{x},y)\big) &\leq \sup_{\boldsymbol{x},y}\big(\ell(h;\boldsymbol{x},y)-\ell(\hat{h}_{tar},\boldsymbol{x},y)\big)+\sup_{\boldsymbol{x},y}\big(\ell(\hat{h}_{tar},\boldsymbol{x},y)-\ell(h';\boldsymbol{x},y)\big)\\ &\leq \sup_{\boldsymbol{x},y}\big(\ell(h;\boldsymbol{x},y)-\ell(\hat{h}_{tar},\boldsymbol{x},y)+k\zeta\end{aligned}$$

and the inequalities are all based on $\hat{h}_{tar}$ being $k\zeta$-close to h'.

Plugging the above inequalities into the formula of $\sup_h z(h,h')$ for model h', and with the assumption that $0\leq R(h)\leq R^*,\forall h\in\mathcal{H}$, we immediately have

$$\begin{aligned}\sup_{h} z(h,h') &\geq \sup_{h}\frac{N\big(L(\hat{h}_{tar};\mathcal{S}_c)-L(h;\mathcal{S}_c)\big)-Nk\zeta+NC_R(R(h')-R(h))}{\sup_{\boldsymbol{x},y}\big(\ell(h;\boldsymbol{x},y)-\ell(\hat{h}_{tar};\boldsymbol{x},y)\big)+k\zeta+C_R(R(h)-R(h'))}\\ &\geq \sup_{h}\left(\frac{L(\hat{h}_{tar};\mathcal{S}_c)-L(h;\mathcal{S}_c)-Nk\zeta-NC_R\cdot R^*}{\sup_{\boldsymbol{x},y}\big(\ell(h;\boldsymbol{x},y)-\ell(\hat{h}_{tar};\boldsymbol{x},y)\big)+k\zeta+C_R\cdot R^*}=z'(h)\right).\end{aligned}$$

Since the inequality holds for any h', we have

$$\inf_{\|h'-\hat{h}_{tar}\|_{\mathcal{S}_{\ell,\mathcal{X},\mathcal{Y}}}\leq\zeta}\sup_{h} z(h,h')\geq\sup_{h} z'(h)$$

and hence $\sup_h z'(h)$ is a valid lower bound.

□

Remark 3.3.12 (Improving Results in Corollary 3.3.11). Assuming $0 \leq R(h) \leq R^*$ is not a strong assumption and actually can be satisfied by many common convex models. For example, for SVM model with ℓ_2-regularizer (in fact, applies to any regularizer $R(h)$ with $R(\mathbf{0}) = 0$), we have $R(h) \leq \frac{1}{C_R}$ and hence $R^* \leq \frac{1}{C_R}$. Moreover, we can further tighten the lower bound by better bounding the term $R(h') - R(h)$. Specifically, $R(h') - R(h) = R(h') - R(\hat{h}_{tar}) + R(\hat{h}_{tar}) - R(h)$ and we only need to have a tighter upper and lower bounds on $R(h') - R(\hat{h}_{tar})$ utilizing some special properties of the loss functions. For the constant k in the bi-directional closeness, we can also compute its value for some specific loss functions. For example, for hinge loss, we can compute the value based on Corollary 3.3.15 in Section 3.3.5.

3.3.5 Loss-based Distance for Hinge Loss

In this section, we provide additional properties of the loss-based distance for the commonly studied hinge loss. In Theorem 3.3.13, we show how one can relate the notion of ζ-closeness in Definition 3.3.1 to the closeness of parameters in the specific setting of hinge loss. We use this as an example to show that our notion of ζ-closeness can be tightly related to the closeness of the models in some cases. Then, in Theorem 3.3.14, we show how the closeness directly in the model parameters can imply the closeness in the loss-based distance. Finally, combining Theorem 3.3.13 and Theorem 3.3.14, In Corollary 3.3.15 we show how the ζ-closeness in the loss-based distance for hinge loss implies the bi-directional closeness.

Theorem 3.3.13. *Consider the hinge loss function $\ell(h_{\boldsymbol{\theta}}; \boldsymbol{x}, y) = \max(1 - y \cdot \langle \boldsymbol{x}, \boldsymbol{\theta} \rangle, 0)$ for $\boldsymbol{\theta} \in \mathbb{R}^d$ and $\boldsymbol{x} \in \mathbb{R}^d$ and $y \in \{-1, +1\}$. For $\boldsymbol{\theta}, \boldsymbol{\theta}' \in \mathbb{R}^d$, where $\boldsymbol{\theta}'$ is the weight parameter of the model $h_{\boldsymbol{\theta}'}$ such that $\|\boldsymbol{\theta}\|_1 \leq r$ and $\|\boldsymbol{\theta}'\|_1 \leq r$, if $\boldsymbol{\theta}$ is ζ-close to $\boldsymbol{\theta}'$ in the loss-based distance, then, $\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_1 \leq r \cdot \zeta$.*

Proof of Theorem 3.3.13. We construct a point $\boldsymbol{x}^*$ as follows:

$$\boldsymbol{x}_i^* = \begin{cases} -\frac{1}{r}, \text{ if } \boldsymbol{\theta}_i > \boldsymbol{\theta}'_i, i \in [d] \\ +\frac{1}{r} \text{ if } \boldsymbol{\theta}_i \leq \boldsymbol{\theta}'_i, i \in [d] \end{cases}$$

Then we have

$$\langle \boldsymbol{\theta}' - \boldsymbol{\theta}, \boldsymbol{x}^* \rangle = \frac{1}{r} \cdot \|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_1 \tag{3.1}$$

Since $\|\boldsymbol{\theta}\|_1 \leq r$ we have

$$\langle \boldsymbol{x}^*, \boldsymbol{\theta} \rangle \geq -1 \tag{3.2}$$

and similarly since $\|\boldsymbol{\theta}'\|_1 \leq r$ we have

$$\langle \boldsymbol{x}^*, \boldsymbol{\theta}' \rangle \geq -1. \tag{3.3}$$

Therefore by Inequalities (3.2) and (3.3) we have

$$\ell(h_{\boldsymbol{\theta}}; \boldsymbol{x}^*, -1) - \ell(h'_{\boldsymbol{\theta}'}; \boldsymbol{x}^*, -1) = \max(1 + \langle \boldsymbol{x}^*, \boldsymbol{\theta} \rangle, 0) - \max(1 + \langle \boldsymbol{x}^*, \boldsymbol{\theta}' \rangle, 0) = \langle \boldsymbol{\theta} - \boldsymbol{\theta}', \boldsymbol{x}^* \rangle$$

which by (3.1) implies

$$\ell(h_{\boldsymbol{\theta}}; \boldsymbol{x}^*, -1) - \ell(h'_{\boldsymbol{\theta}'}; \boldsymbol{x}^*, -1) = \frac{1}{r} \cdot \|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_1. \tag{3.4}$$

Now since we know that, $\forall \boldsymbol{x} \in \mathbb{R}^d$, the loss difference between h and h' is bounded by ζ, the bound should also hold for the point $(\boldsymbol{x}^*, -1)$, meaning that

$$\frac{1}{r} \cdot \|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_1 \leq \zeta.$$

which completes the proof. □

Theorem 3.3.14. *Consider the hinge loss function $\ell(h_{\boldsymbol{\theta}}; \boldsymbol{x}, y) = \max(1 - y \cdot \langle \boldsymbol{x}, \boldsymbol{\theta} \rangle, 0)$ for $\boldsymbol{\theta} \in \mathbb{R}^d$ and $\boldsymbol{x} \in \mathbb{R}^d$ and $y \in \{-1, +1\}$. For $\mathcal{X} = \{\boldsymbol{x} \in \mathbb{R}^d \colon \|\boldsymbol{x}\|_1 \leq q\}$ and $\mathcal{Y} = \{-1, +1\}$, For any two models $h_{\boldsymbol{\theta}}, h'_{\boldsymbol{\theta}'}$ if $\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_1 \leq \zeta$, then $h_{\boldsymbol{\theta}}$ is $q \cdot \zeta$-close to $h'_{\boldsymbol{\theta}'}$ in the loss-based distance. Namely,*

$$D_{\ell, \mathcal{X}, \mathcal{Y}}(h_{\boldsymbol{\theta}}, h'_{\boldsymbol{\theta}'}) \leq q \cdot \zeta.$$

Proof. For any given $h_{\boldsymbol{\theta}}$ and $h'_{\boldsymbol{\theta}'}$, by triangle inequality for maximum, we have

$$\ell(h_{\boldsymbol{\theta}}; \boldsymbol{x}, y) - \ell(h'_{\boldsymbol{\theta}'}, \boldsymbol{x}, y) = \max(1 - y \cdot \langle \boldsymbol{x}, \boldsymbol{\theta} \rangle, 0) - \max(1 - y \cdot \langle \boldsymbol{x}, \boldsymbol{\theta}' \rangle, 0) \leq \max(0, \langle y\boldsymbol{x}, \boldsymbol{\theta}' - \boldsymbol{\theta} \rangle).$$

Therefore, we have

$$\max_{(\boldsymbol{x}, y) \in \mathcal{X} \times \mathcal{Y}} \ell(h_{\boldsymbol{\theta}}; x, y) - \ell(h'_{\boldsymbol{\theta}'}; \boldsymbol{x}, y) \leq \max_{(\boldsymbol{x}, y) \in \mathcal{X} \times \mathcal{Y}} \max(0, \langle y\boldsymbol{x}, \boldsymbol{\theta}' - \boldsymbol{\theta} \rangle).$$

Our goal is then to obtain an upper bound of $O(\zeta)$ for $\max_{(\boldsymbol{x}, y) \in \mathcal{X} \times \mathcal{Y}} \langle y\boldsymbol{x}, \boldsymbol{\theta}' - \boldsymbol{\theta} \rangle$ when $\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_1 \leq \zeta$. To maximize $\langle y\boldsymbol{x}, \boldsymbol{\theta}' - \boldsymbol{\theta} \rangle$ by choosing $\boldsymbol{x}$ and y, we only need to ensure that $\operatorname{sign} y\boldsymbol{x}_i = \operatorname{sign}(\boldsymbol{\theta}'_i - \boldsymbol{\theta}_i), i \in [d]$.

Therefore, based on the assumption that $\frac{1}{q}\|\boldsymbol{x}\|_1 \leq 1$ (i.e.,$\frac{1}{q}|\boldsymbol{x}_i| \leq 1, i \in [d]$) we have

$$\max_{(\boldsymbol{x},y)\in\mathcal{X}\times\mathcal{Y}} \frac{1}{q}\langle y\boldsymbol{x}, \boldsymbol{\theta}' - \boldsymbol{\theta}\rangle = \sum_{i=1}^{d} \frac{1}{q}|\boldsymbol{x}_i||\boldsymbol{\theta}_i - \boldsymbol{\theta}'_i| \leq \sum_{i=1}^{d} |\boldsymbol{\theta}_i - \boldsymbol{\theta}'_i| = \|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_1 \leq \zeta,$$

which concludes the proof. □

Corollary 3.3.15. *For hinge loss, with Theorem 3.3.13 and Theorem 3.3.14, if h is ζ-close to h', then h' is $r \cdot q \cdot \zeta$-close to h.*

3.4 Experiments

In this section, we first show how well the proposed attack converges to the target model as more poisoning points are generated (Section 3.4.2). Then, we show how well the proposed attack performs in achieving the encoded attack objectives in the target models. The first two sections leverage the heuristic label-flipping method (Koh et al., 2022) to generate the target models so as to fairly compare with the existing baseline and highlight the major contribution of this work in designing model-targeted attacks for any achievable target models. In Section 3.4.4, we show our refined target model generation approach, which improves the performance of the proposed MTP attack compared to the case of running it with target models generated by the original method in Koh et al. (2022). This improved method also boosts the performance of other poisoning attacks that leverage target models and hence is used as the target model generation method in Chapter 5 when comparing against the (indiscriminate) influence attack that does not rely on target models.

3.4.1 Experimental Setup

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 and Zhu, 2015a; Koh et al., 2022; Steinhardt et al., 2017; Koh and Liang, 2017; Jagielski et al., 2019). For the subpopulation attack experiments, we use the Adult dataset (Dua and Graff, 2017), which was used for evaluation by (Jagielski et al., 2019; 2021). Following the prior work, 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 a dimension of 57 after one-hot encoding of the categorical attributes. For the indiscriminate setting, we use the Dogfish (Koh and Liang, 2017) and MNIST 1–7

datasets (LeCun, 1998)[3]. 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 and Liang (2017); Steinhardt et al. (2017); Koh et al. (2022) and each image is represented by a 2,048-dimensional vector. The MNIST 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 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. In Chapter 4, we will select semantically more meaningful subpopulations based on demographic characteristics for the Adult dataset to have a deeper investigation on subpopulation susceptibility.

Models. We conduct experiments on linear SVM and logistic regression (LR) models and set the hyperparameter $C_R = 0.09$ for all settings, but our attack performance is not sensitive to the choice of C_R. We use the heuristic approach from Koh et al. (2022) to generate target classifiers for both model types. To better assess the quality of induced models from different attacks in terms of their closeness to the target model and also the encoded attack objective, we consider attack settings where the attackers are interested in inducing a particular error rate on the whole test set or the defined subset for the indiscriminate and subpopulation settings respectively. 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 Dogfish, we aim to generate target models with overall test errors of 10%, 20%, and 30%.

To obtain target models of the desired errors, a range of candidate target classifiers are generated by trying different combinations of hyperparameters γ and repetitions r for Algorithm 2, where the γ is set as the q-th percentile when the clean margin for every point in $\mathcal{S}_{te}$ (Step 2 in Algorithm 2) is sorted from the lowest to the highest. In particular, for experiments on the MNIST 1–7 and Dogfish

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

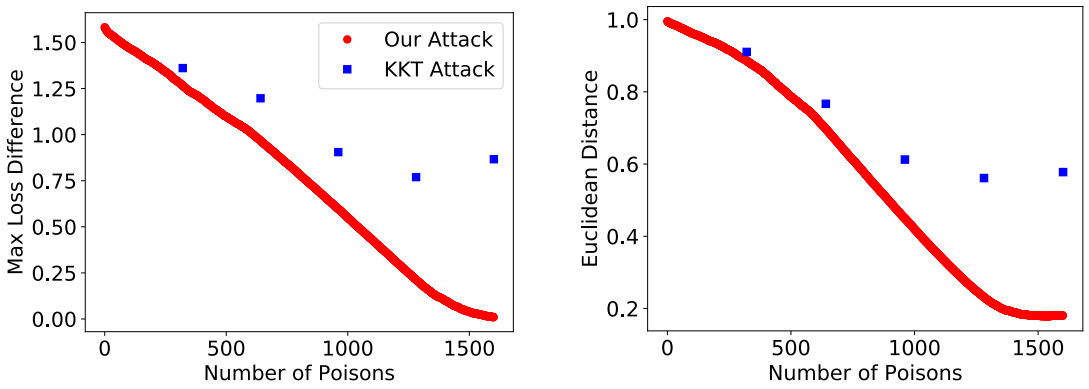
datasets, we use a range of values for q as $[0.05, 0.10, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40, 0.45, 0.50, 0.55, 0.6]$ and a range of values for r as $[1, 2, 3, 5, 8, 10, 12, 15, 20, 25, 30, 50, 60]$. For experiments on the Adult dataset, we use a range of values for q as $[0.40, 0.45, 0.50, 0.55, 0.8, 1.0]$ and a range of values for r as $[1, 2, 3, 5, 8, 10, 12, 15, 20, 25, 30, 40, 50, 80, 100]$.

With the above configurations and also returning the target model that satisfies the attack goal and has the lowest loss on $\mathcal{S}_c$, for linear SVM on MNIST 1–7, we obtained target models of test accuracies of 94.0%, 88.8%, and 83.3%, and for LR on MNIST 1–7, the target models are of test accuracies of 94.7%, 89.0%, and 84.5%. For linear SVM on Dogfish, we obtained target models of test accuracies of 89.3%, 78.3%, and 67.2% and for LR on Dogfish, 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, 98.8% on MNIST 1–7, and 99.2% on Dogfish. The test accuracy of the clean LR model is 79.9% on Adult, 97.8% on MNIST 1–7, and 98.3% on Dogfish.

Baseline attacks. We compare our model-targeted poisoning attack in Algorithm 5 to the state-of-the-art KKT attack (Koh et al., 2022) (details in Section 2.2.2). We omit the model-targeted attack from Mei and Zhu (2015b) because there is no open source implementation and this attack (similar in spirit to the influence attack (Koh et al., 2022)) is also reported underperforming the KKT attack (Koh et al., 2022). Our main focus here is to compare with other model-targeted attacks in terms of achieving the target models and in Chapter 5, we will compare with other data poisoning attacks that do not necessarily rely on target models and show that our attack still achieves competitive or better performance.

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 the 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. 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 ζ-close distance threshold (enables us to approach the target model closely when not knowing the required number of poisoning points in advance). The loss function is hinge loss for SVM and logistic loss for LR. For the SVM model, we set ζ as 0.01 on Adult, 0.1 on MNIST 1–7 and 2.0 on the Dogfish dataset. For the LR model, we set ζ as 0.05 on Adult, 0.1 on MNIST 1–7



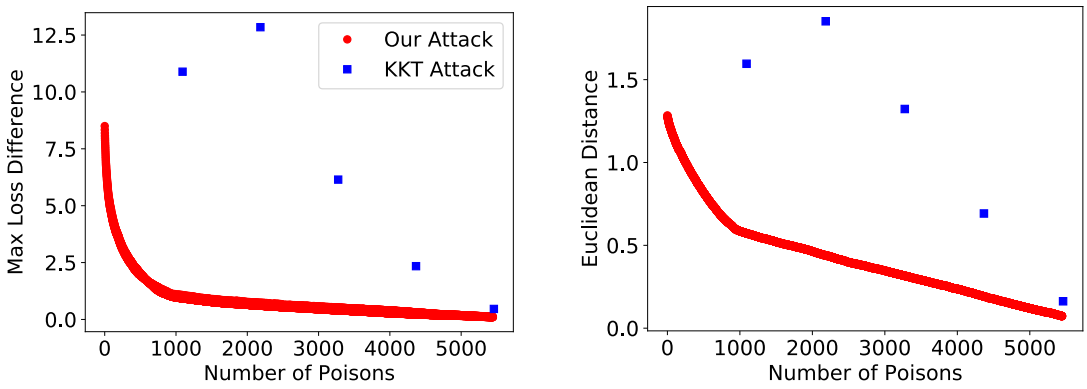
(a) Max Loss Difference (b) Euclidean Distance

Figure 3.1: Convergence to the target model. The results shown are for the first subpopulation (Cluster 0) of the Adult dataset, and the model is linear SVM. The maximum number of poisoning points is set using the 0.01-close threshold to the target classifier.

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 on 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, which means the performance of our attack can also be smoothly plotted in each attack iteration that incrementally adds a poisoning point.

3.4.2 Convergence to the Target Model

Figure 3.1 shows the convergence of Algorithm 5 using both the maximum loss difference and the 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 5 is 0.01-close to the target classifier. Our attack steadily reduces the maximum loss difference and the 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 poisoning points in Figure 3.1, 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. From Figure 3.2, we see a similar trend in the indiscriminate setting, and our attack still converges to the target model more reliably compared to the KKT



(a) Max Loss Difference (b) Euclidean Distance

Figure 3.2: Convergence to the target model. The results shown are for the target classifier of error rate 10% for the MNIST 1–7 dataset and the model is linear SVM. The maximum number of poisoning points is set using the 0.1-close threshold to the target classifier.

attack. Comparing the performances of the KKT attack in the indiscriminate and subpopulation settings, we can find that, it converges more reliably in the indiscriminate setting compared to the subpopulation setting, and might be worth exploring the possible reasons behind this interesting observation in the future.

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 $\mathcal{S}_c \cup \mathcal{S}_p$ with $|\mathcal{S}_p| = n$, the (exact) same classifier can also be obtained by training on the set $\mathcal{S}_c \cup \mathcal{S}'_p$ with $|\mathcal{S}'_p| \leq n$. This poisoning set $\mathcal{S}'_p$ 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 of points used to exactly produce the target model.

3.4.3 Achieving Encoded Objectives and the Attack Optimality

We first show how well the attack objectives can be achieved by the baseline KKT attack and the proposed MTP attack. Then, we show the optimality of our attack by comparing the number of poisoning points used by our attack and the lower bound empirically computed with Theorem 3.3.10.

Attack success. We compare the classifiers induced by the two attacks in terms of the attacker's goal. Table 3.1 summarize the results of the subpopulation attacks, where attack success is measured on the targeted

Model/ Dataset	Target Model	n_p	Lower Bound	$0.2n_p$ KKT	$0.2n_p$ Ours	$0.4n_p$ KKT	$0.4n_p$ Ours	$0.6n_p$ KKT	$0.6n_p$ Ours	$0.8n_p$ KKT	$0.8n_p$ Ours	n_p KKT	n_p Ours
SVM/ Adult	Cluster 0	1,866	1,667	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	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	1,863.0	94.9	**24.3**	15.9	20.3	34.3	**12.1**	21.6	**0.3**	20.3	**0.3**
LR/ Adult	Cluster 0	2,005	N/A	82.1	**75.6**	71.7	**42.2**	46.7	**15.9**	36.6	**1.9**	24.3	**0.4**
	Cluster 1	1,630	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	**5.8**	79.2	**0.6**	60.2	**0.6**

Table 3.1: Subpopulation attack on Adult: comparison of test accuracies on subpopulations (%). Target models for the 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 errors are omitted as they are almost negligible. We do not show the lower bound for LR because we can only compute an approximate maximum loss difference and the lower bound will no longer be valid.

cluster. At the maximum number of poisoning points, 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. Because of the better performance of MTP over the KKT attack, we will use MTP as the new state-of-the-art subpopulation attack when empirically measuring the susceptibilities of different subpopulations in Chapter 4.

Table 3.2 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 can be hard to optimize (see Section 2.2.2 for details). For logistic loss, our attack also needs to maximize a non-concave maximum loss difference[4] (Step 4 in Algorithm 5). However, this objective is much easier to optimize than that of the KKT attack. This might explain the observation why our attack is much more effective than the KKT attack when tested on LR models, compared to the original linear SVM models. We also make a note here that Dogfish can be easily attacked using our method, especially for linear SVM, to achieve significantly high test errors. For example, only using 1.8% (32/1800) of poisoning ratio, the test error can be increased from 0.8% to $> 10\%$. However, such a high test error is easily achieved because the target model overfits to the dataset, and it has much higher error on the test data compared to the training data. Therefore, in this experiment, we only use the Dogfish dataset to illustrate how well the encoded (and even overfitted) objective can be achieved in a given target model. Details on how to assess the dataset vulnerability correctly (in terms of the actual risk after poisoning) for the Dogfish dataset will be discussed in Section 5.2.1.

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

Model/ Dataset	Target Model	n_p	Lower Bound	$0.2n_p$		$0.4n_p$		$0.6n_p$		$0.8n_p$		n_p	
				KKT	Ours	KKT	Ours	KKT	Ours	KKT	Ours	KKT	Ours
SVM/ MNIST 1-7	5%	1,737	874	97.3	**97.1**	96.4	**96.1**	95.7	95.7	94.9	94.9	94.3	94.6
	10%	5,458	3,850.4	95.8	**95.5**	93.4	**92.1**	92.7	**90.9**	91.1	**90.7**	90.2	90.2
	15%	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%	32	15	97.0	**95.8**	94.0	**93.3**	92.2	**91.2**	90.7	**90.2**	90.3	**89.8**
	20%	89	45	95.5	95.7	92.5	**92.2**	90.3	**88.7**	84.7	84.7	82.3	**82.0**
	30%	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%	756	N/A	97.5	**96.9**	97.4	**96.5**	97.2	**96.0**	96.9	**95.7**	96.9	**95.2**
	10%	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%	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%	62	N/A	98.8	**93.0**	98.5	**89.7**	98.8	**89.2**	98.8	**89.2**	98.8	**89.0**
	20%	120	N/A	98.5	**93.2**	99.2	**88.2**	99.3	**85.3**	99.5	**83.0**	99.5	**80.7**
	30%	181	N/A	97.8	**92.3**	98.8	**85.7**	99.2	**81.3**	99.5	**75.7**	99.5	**72.5**

Table 3.2: 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 ζ-close to the target model, and we set ζ as 0.1 for MNIST 1–7 and 2.0 for the Dogfish dataset. All results are averaged over 4 runs and standard errors are omitted as they are negligible. We do not show the lower bound for LR because we can only compute an approximate maximum loss difference and the lower bound will no longer be valid.

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 3.3.10 (details in Section 3.3.3). Note that Theorem 3.3.10 provides a valid lower bound based on any intermediate model. To get tighter a lower bound on the number of poisoning points, we only need to use Theorem 3.3.10 on the encountered intermediate models and report the best (i.e., highest) one. We do this by running Algorithm 5 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 3.3.10. 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 3.1 (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 3.2, 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.

Target Models	Test Acc (%)		Loss on $\mathcal{S}_c$		# of Poisons	
	Original	Improved	Original	Improved	Original	Improved
5% Error	94.0	94.9	2,254.6	**1,767.1**	2,170	**1,340**
10% Error	88.8	88.9	4,941.0	**3,233.1**	5,810	**2,432**
15% Error	83.3	84.5	5,428.4	**4,641.6**	6,762	**3,206**

Table 3.3: SVM on MNIST 1–7: comparison of two target generation methods on the number of poisoning points used by MTP to reach 0.1-closeness to the target. *Original* indicates the original target generation process from Koh et al. (2022). *Improved* denotes our improved target generation process with adaptive model updating. Better results are highlighted in bold.

3.4.4 Improved Target Model Generation Process

We first show how to improve the quality of the generated target models significantly by slightly modifying the target generation method proposed in Koh et al. (2022), so that the returned target models can have much lower loss on $\mathcal{S}_c$. Then, compared to the original approach (Koh et al., 2022), we show that the proposed MTP attack requires significantly fewer poisoning points to achieve similar attacker objectives using target models generated from the modified approach.

Improved target model generation. In this section, we improve the quality of the generated target models by slightly adjusting the target model generation process by Koh et al. (2022) (details in Algorithm 2). In particular, we find that in the indiscriminate setting, when we are iteratively generating a list of target models with more and more aggressive hyperparameters (i.e., gradually decreasing γ and gradually increasing r) that lead to models with higher errors, at iteration i, if we replace the $\hat{h}_c$ in Line 2 in Algorithm 2 with the model $\hat{h}_{tar}$ that is returned from iteration $i-1$, then we are able to generate $\mathcal{S}_{\text{flip}}$ that helps to eventually produce a target model with a lower loss on the clean training data while still satisfying the desired error.

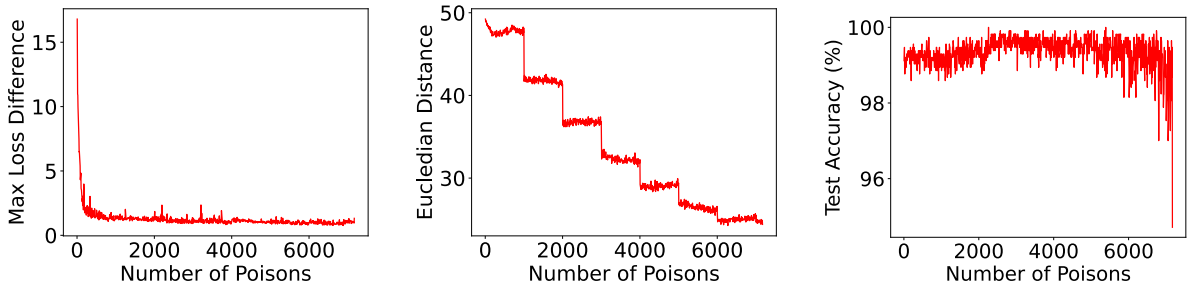
We speculate the possible reason is, this way of adaptively updating $\hat{h}_c$ in the iterative process helps to identify more "critical" data points in $\mathcal{S}_{\text{flip}}$ that achieve even lower loss when their labels are flipped. This slightly modified generation process can significantly reduce the number of poisoning points needed by our MTP attack to reach the same ζ-closeness (with respect to the loss-based distance) to the target classifier, consistent with the claims in Theorem 3.3.3 that using target models with a lower loss on the clean training data $\mathcal{S}_c$ will improve the performance of the proposed MTP attack. We also find similar improvements for the KKT attack and the i-Min-Max attack and hence use this improved target generation method when comparing these attacks in the indiscriminate setting thoroughly in Chapter 5. We did not find a significant difference in the subpopulation setting using the improved approach above, and hence leave its exploration as future work.

Results. We run experiments on the linear SVM and the MNIST 1–7 dataset. For both the original and improved target generation methods, we generate three target classifiers with error rates of 5%, 10% and 15%. The original target classifier generation method returns classifiers with test accuracies of 94.0%, 88.8%, and 82.3% respectively (these models are also used in earlier experiments in this section). The improved target generation process returns target classifiers with approximately the same test accuracies (94.9%, 88.9%, and 84.5%). However, for classifiers of (approximately) the same error rate returned from the two target generation processes, the improved generation method produces classifiers with significantly lower losses on $\mathcal{S}_c$ compared to the original one.

Table 3.3 compares the two target generation approaches by showing the number of poisoning points needed to get 0.1-close to the corresponding target model of the same error rate. For example, for target models of 15% error rate, the model from the original approach has a total clean loss of 5,428.4 on $\mathcal{S}_c$ while our improved method reduces it to 4641.6. With the reduced clean loss, getting 0.1-close to the target model generated from our improved process only requires 3,206 poisoning points, while reaching the same distance to the target model produced by the original method would require 6,762 poisoning points, which gives more than 50% reduction on the number of poisoning points.

3.5 Discussion

We first discuss the limitation of our work, and then the possible extension to other settings where poisoning is relevant.



(a) Maximum Loss Difference (b) Euclidean Distance (c) Test Accuracy on Subpopulation

Figure 3.3: Maximum loss difference, Euclidean distance and test accuracy on target subpopulation across iterations for our attack. Data is not batched, and the same weight-initializations for $\hat{h}_t, \hat{h}_p$ are used. The loss drops sharply within the first few iterations, but the accuracy fluctuates within a very small window, even when $|\mathcal{S}_p| \sim 0.5|\mathcal{S}_c|$ is added.

Limitation of our work. Our work focuses on inducing given target models but does not provide a principled way to generate target classifiers such that the attacker objectives can be achieved efficiently using our attack, other than slightly modifying the target model generation process in Section 3.4.4. A systematic investigation on the generation of better target classifiers is the important next step to fully characterize the limits of data poisoning attacks. A feasible approach is to generate the target models by directly formulating it as an optimization problem with respect to the model weights (Sun et al., 2021), instead of just generating the target model weights through training on a poisoned dataset obtained from simple heuristics. In fact, the very recent model-targeted attack against deep neural networks (DNNs) shows the potential of using optimization-based approaches to generate target models (Lu et al., 2023). We believe Theorem 3.3.3 may also provide additional insights when leveraging optimization-based approaches to generate the target models (e.g., the optimization should aim for minimizing the loss on $\mathcal{S}_c$). For example, Lu et al. (2023) find that scaling down the weight parameters of the target DNNs can help reduce the required number of poisoning points, which might be explained by Theorem 3.3.3 because the scaled-down weights will lead to a lower loss on $\mathcal{S}_c$ and hence, fewer poisoning points.

In terms of extending MTP to DNNs, our theoretical analysis only applies to convex losses, and it is unclear how to extend the theory to non-convex models (e.g., DNNs). As for the empirical performance of the MTP attack on DNNs, we tested it for the non-convex multilayer perceptrons (MLP) trained on MNIST 1–7, with non-linear activations and without convolutional layers. To generate this result, we unrealistically assumed the same model weight initialization will be used in the training of the target model $\hat{h}_{tar}$ and the intermediate model $\hat{h}_t$, and the model training also does not split the training data into multiple batches. The subpopulation is defined as all the points in class 1 and so, the adversary's goal is to have test images that would be correctly classified as 1s, classified as 7s. The target model has a 5% error on the selected subpopulation and is generated by adding a large number of poisoning points from the subpopulation with flipped labels.

The results are given in Figure 3.3 and we can see that the empirical performance of MTP is unstable and ineffective, despite using two unrealistic assumptions that help eliminate the randomness involved in the attack process, which otherwise will make the performance of the MTP attack even worse. It is clear that, although the maximum loss difference (and partially the Euclidean distance) seem to converge, the encoded attacker objective cannot be reliably achieved, as the test accuracy on the subpopulation after poisoning still largely fluctuates above a very high value (>98%) most of the time. We speculate the ineffectiveness of MTP on DNNs is because it uses a loss-based metric (i.e., maximum loss difference), which might not be suitable for non-convex models. Note that, the KKT attack is based on the first-order optimality condition

with respect to the target model (e.g., gradient-based metric), and the idea of KKT is successfully applied to neural networks very recently Lu et al. (2023). Therefore, we hypothesize that replacing the loss-based metric in our attack with a gradient-based metric might also improve the performance of MTP on DNNs. In particular, the gradient-based MTP might work by finding the poisoning points that minimize the gradient cosine similarity between the intermediate model $\hat{h}_t$ and the target model $\hat{h}_{tar}$, and update the model weights using the generated poisoning points. We leave the detailed exploration of this path as future work.

At last, in this work, we have not considered defenses, but it is an important and interesting direction to study the effectiveness of our attack against common defenses. As an example, data sanitization defenses may mainly impact the constraint set $\mathcal{C}$ and a significantly shrunk $\mathcal{C}$ will greatly hinder the convergence property of the proposed MTP attack in Theorem 3.3.3, and lead to a significantly increased number of poisoning points to achieve the given attack goal.

Extension of model-targeted attacks to other settings. We also believe the idea of model-targeted attack and the proposed MTP attack with provable convergence can be applied to attacker objectives related to other trustworthy aspects of machine learning, including privacy and fairness. In fact, the follow-up work that uses the model-targeted approach to break the fairness of machine learning demonstrates the feasibility of this extension (Jo et al., 2022).

3.6 Summary

We propose a general poisoning framework with provable guarantees to approach any attainable target classifier, along with a lower (Theorem 3.3.10) 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. Our attack quantifies a tighter lower bound on the limits of subpopulation and indiscriminate poisoning attacks and will also be used in Chapter 4 and Chapter 5 to measure the susceptibility variations empirically.

Chapter 4

Explaining Subpopulation Susceptibility[1]

4.1 Introduction

Subpopulation attacks might represent the most motivated poisoning attack goals, as these attacks can broadly impact the targeted subpopulation while minimally impacting the overall performance of the model (Jagielski et al., 2019; 2021). Therefore, understanding the limits of subpopulation poisoning attacks has important consequences in understanding the realistic risks of poisoning attacks. Jagielski et al. (2019) first demonstrated the feasibility of subpopulation attacks using a (relatively weak) random label-flipping attack on the Adult dataset, and later extended the analysis to other benchmark datasets (Jagielski et al., 2021). Both works also observe the existence of disparate vulnerabilities, measured by the absolute increase in test errors on the subpopulation, across a few selected subpopulations and some subpopulations are indeed very hard to poison. This seems to indicate that the effectiveness of subpopulation attacks might be inherently limited in some cases. However, the factors of subpopulations that impact the attack effectiveness were not explored. In addition, one might also question whether the empirical observation using the weaker random label-flipping attack can well reflect the limits of subpopulation poisoning attacks.

In order to better understand the limits of subpopulation attacks and better assess realistic risks from poisoning, in this chapter, we explore factors that are potentially related to subpopulation susceptibility by

[1]This chapter is largely based on Evan Rose, Fnu Suya, David Evans, *Poisoning Attacks and Subpopulation Susceptibility*, in the 5th Workshop on Visualization for AI Explainability (VISxAI 2022).

testing on a larger number of different subpopulations using stronger poisoning attack (i.e., the proposed MTP attack in Chapter 3). Findings in this chapter also motivate us to further explore different dataset susceptibility in Chapter 5. Another work that studies the inherent vulnerabilities of individual test samples under targeted poisoning attacks may also be applied for the subpopulation setting (Wang et al., 2022). However, a direct extension of targeted attacks to subpopulation settings might overestimate the power of subpopulation attacks and cannot well reflect the inherent vulnerabilities of subpopulations to poisoning. More detailed discussion on the relation of our work to Wang et al. (2022) is given in Section 5.7.

The published version (Rose et al., 2022) of the results in this chapter are presented in the form of an interactive data visualization (presented at the VISxAI workshop that focuses on explaining AI through visualizations) to help visually illustrate the data poisoning process and also demystify subpopulation properties that impact their vulnerabilities to strong empirical attacks. We are not able to include interactive visualizations in this format, and encourage readers to visit `https://uvasrg.github.io/visualizing-poisoning/` to explore the visualizations described here.

This chapter is organized as follows. In Section 4.2, we provide the details on the selected poisoning attack and the evaluation metric. In Section 4.3, we first present the findings (including the related subpopulation properties to susceptibility) on the synthetic dataset and then show that many findings also generalize to the benchmark Adult dataset (Section 4.4). We then discuss the limitation of our work in Section 4.5 and conclude the chapter in Section 4.6.

4.2 Poisoning Strategy and Evaluation Metrics

In this section, we first show how to measure the subpopulation susceptibility empirically given a poisoning attack. Then, we provide details on using the MTP attack as the given poisoning attack to obtain an empirical estimation of the subpopulation susceptibility.

Empirically measuring subpopulation susceptibility. Given a poisoning attack, we measure the vulnerability of a subpopulation to poisoning by the number of poisoning points needed to achieve a particular test error on the defined subpopulation (i.e., the attack difficulty), similar to the subpopulation experiments in Chapter 3 where attackers also aim to achieve certain test error on the subpopulation by inducing the corresponding target model. The only subtle difference is, in Chapter 3, we aim to well produce the attack objective encoded in a given target model, while for experiments in this chapter, the interested attack objective (e.g., 50% test error on the subpopulation) can be different from the exact encoded objective (e.g.,

100% test error on the subpopulation) in the target model. Also, there can be other ways to define the vulnerability, such as the poisoned test error on the subpopulation after injecting a fixed ratio of poisoning points with respect to the size of the subpopulation or the whole training data. However, from the perspective of visualizing poisoning attacks, compared to other forms of vulnerability measurement, measuring the subpopulation vulnerability (or the attack difficulty) based on the number of poisoning points needed can be visually clearer when comparing different subpopulations that are successfully attacked (i.e., test errors on the subpopulations exceed the set threshold). In addition, this susceptibility measurement also matches well with the chosen MTP attack from Chapter 3 (see justification below) that adds the poisoning point incrementally in each iteration, as we can clearly see how the poisoning points impact the learned model as more number of poisoning points are added, which is again beneficial for visualizing the poisoning process.

Choosing MTP over other baselines. We choose the proposed MTP attack in Chapter 3 as the poisoning strategy because it achieves state-of-the-art performance in achieving the attack objectives encoded in the target models in the subpopulation setting (Section 3.4). In addition, MTP does not require the number of poisoning points needed in advance, which suits our setting because we aim to induce similar test errors on different subpopulations, but do not know their needed number of poisoning points in advance. We also did not consider the random label-flipping attack by Jagielski et al. (2019; 2021) in the rest of the chapter because the attack is relatively weak due to only randomly flipping the labels of existing points, not optimizing both the data features and labels as in the MTP attack.

To see the performance gap between the random label-flipping and the MTP attacks, we compare their attack effectiveness in achieving 100% test error on the subpopulation using the same number of poisoning points, and we set this number by terminating our attack when the induced model has 100% test error on the subpopulation, and the clean model without poisoning has 0% test errors on all the selected subpopulations. Since the choice of target models can impact the performance of the MTP attack in achieving the target attack objective (e.g., 100% test error on the subpopulation), we heuristically generate the desired target models by ensuring that they 1) satisfy the attacker objective and 2) have larger losses on the training data from the subpopulation, and 3) have relatively lower losses on the entire clean training set. We simply generate the subpopulations by clustering, as in Section 3.4 to compare the two attacks.

Table 4.1 shows the result and across all settings, our attack is considerably more successful. However, the random label-flipping attack is also relatively successful in some cases (e.g., Cluster 1 in the SVM experiment). We believe the success of the label-flipping attack is due to the following two reasons: first, label-flipping in the subpopulation setting can be successful because smaller subpopulations show stronger degree of

	SVM			LR		
	Cluster 0	Cluster 1	Cluster 2	Cluster 0	Cluster 1	Cluster 2
MTP	**0.0**%	**0.0**%	**0.0**%	**0.0**%	**0.0**%	**0.0**%
Label-Flipping	31.4%	2.8%	15.5%	15.9%	14.0%	19.1%

Table 4.1: Comparison of our attack to the label-flipping based subpopulation attack. The table compares the test accuracy on subpopulation of Adult dataset under same number of poisoning points. The number of poisoning points is determined when our attack achieves 0% test accuracy on the subpopulation. Cluster 0-3 in the logistic regression and SVM models denote different clusters. For logistic regression, the number of poisoning points for Cluster 0-3 are 1,575, 1,336 and 1,649 respectively. For SVM, the number of poisoning points for Cluster 0-3 are 1,252, 1,268 and 1,179 respectively.

locality. Hence, injecting points (from the subpopulation) with flipped labels can also strongly impact the selected subpopulation. This is confirmed by the empirical evidence that increasing the subpopulation size (i.e., reducing its locality) gradually reduces the label-flipping effectiveness and the attack becomes almost ineffective in the indiscriminate setting (i.e., the subpopulation is the entire population). Second, the Adult dataset only contains 57 features, where 53 of them are binary features with additional constraints. Therefore, the benefit from optimizing the feature values is less significant, as the optimization search space of our attack is fairly limited. Nevertheless, our attack still achieves the best performance in the subpopulation settings and hence, we will empirically measure the subpopulation susceptibility by running the MTP attack next.

Details on running the MTP. To run the MTP attack, we first generate the target model of 100% error rate with the lowest loss on the clean training data (similar to the generation process in Section 3.4), but then terminate the MTP attack when the induced model misclassifies at least 50% of the target subpopulation measured on the test set (i.e., the second stopping condition in Section 3.2), instead of achieving the encoded objective of 100% test error in the target model. Then we record the set of poisoning points $\mathcal{S}_p$ generated during the attack process. The susceptibility metric (will be referred to as *difficulty* in the following discussion) will be set as the ratio between the size of the poisoning set $\mathcal{S}_p$ and the clean training set $\mathcal{S}_c$ as $|\mathcal{S}_p|/|\mathcal{S}_c|$. This 50% of threshold was chosen to mitigate the impact of outliers in the subpopulations. In earlier experiments requiring 100% attack success, we observed that attack difficulty was often determined by outliers in the subpopulation. By relaxing the attack success requirement, we are able to capture the more essential properties of an attack against each subpopulation. Since our eventual goal is to characterize attack difficulty in terms of the properties of the targeted subpopulation (which outliers do not necessarily satisfy), this is a reasonable relaxation.

Next, we will first show our findings on the correlated factors to subpopulation susceptibility with necessary visualizations for the 2-dimensional synthetic dataset (Section 4.3) and then show the identified factors also

generalize to the high-dimensional Benchmark Adult dataset (Section 4.4).

4.3 Synthetic Experiments

In this section, we first provide details on the experimental setup for the synthetic experiments (Section 4.3.1) and then present the findings on the subpopulation properties that impact their susceptibilities to poisoning attacks (Section 4.3.2).

4.3.1 Experiment Setup

Dataset generation. The synthetic datasets are generated using generation algorithms from Scikit-learn (Pedregosa et al., 2011), which are adapted from the techniques used to generate the "Madelon" dataset (Guyon et al., 2004). The global properties of each dataset are controlled by two dataset parameters. The first dataset parameter is a class separation parameter $\alpha \geq 0$ and controls the separation between class centers. Larger values of α correspond to an easier classification task. The second dataset parameter is a label noise parameter $\beta \in [0, 1]$ and controls the amount of label noise present in the data. Larger values of β result in a less well-posed classification task, as the input features correlate less strongly with the assigned label.

The exact generation process is analogous to the following procedure. First, two clusters in a 2-dimensional feature space are created by sampling from a Gaussian mixture with 2 components of parameters $\mathcal{N}(\gamma_1, \Sigma_1)$ and $\mathcal{N}(\gamma_2, \Sigma_2)$, respectively, with equal mixture weights. The 2×2 covariances Σ_1 and Σ_2 are determined independently at random and correspond to multiplying non-translated points by a random matrix with entries sampled uniformly at random. The distance between class centers $\|\gamma_1 - \gamma_2\|$ is proportional to the class separation parameter α. The label noise parameter β determines the fraction of points whose labels are assigned uniformly at random from $\{-1, +1\}$; the remaining labels are determined according to the corresponding component in the mixture: points sampled from $\mathcal{N}(\gamma_1, \Sigma_1)$ receive the label -1, and points sampled from $\mathcal{N}(\gamma_2, \Sigma_2)$ receive the label $+1$.

Datasets are generated over a grid of dataset parameters (α, β). The class separation parameter α ranges over 13 equally spaced values in the range $[0, 3]$, and the label noise parameter β ranges over 11 equally spaced values in the range $[0, 1]$. For each dataset parameter combination, 10 datasets are generated by feeding different random seeds into the generation algorithm. Seeds are reused between different dataset parameter combinations. In total, 1,430 synthetic datasets are generated with this method. The linear SVM models are

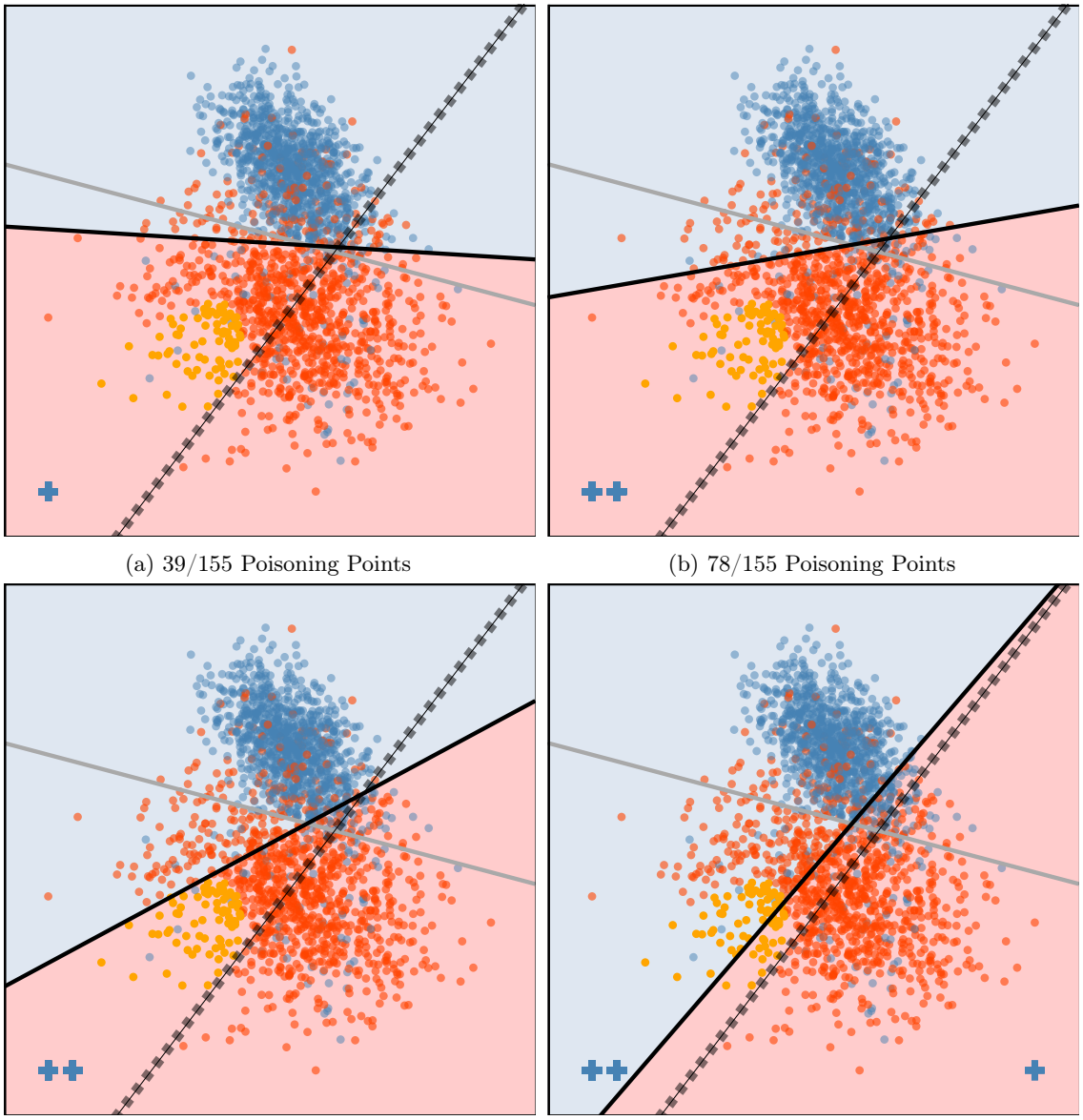
trained using the Scikit-learn package (Pedregosa et al., 2011) and the hyperparameter is set as $C_R = 5e - 4$ for the synthetic dataset.

Subpopulation generation. We use the K-Means clustering algorithm with $k = 16$ to generate the clusters and treat each cluster as a subpopulation, and further extract the negative-labeled instances from each cluster to form the final subpopulation, similar to the experiments in Section 3.4. In total, we generate $1430 \times 16 = 22,880$ subpopulations but ended up with 21,908 non-empty subpopulations that contain at least one point with the negative label. From these subpopulations, 9,591 subpopulations are trivial (i.e., the clean model already satisfies the attack goals and no attack is needed), leaving 12,317 non-trivial subpopulations where poisoning is needed to achieve the attack goal.

4.3.2 Visualizing Poisoning Attacks

In this section, we first show some visualizations that help understand poisoning attacks, especially the proposed MTP attack, better. Next, we show the drastic variation of the subpopulation susceptibility when tested on a large number of datasets and subpopulations. Then, we discuss our findings on the distributional and subpopulation properties that contribute to disparate susceptibilities under linear SVM models.

Visualizing the poisoning process. As a visualization work, one contribution of our work is to visualize the process of poisoning attacks and also some theoretical insights in an intuitive way. Towards this goal, Figure 4.1 shows how the decision boundary is moved as more poisoning points are added to the original training set to eventually misclassify 50% of the test points from the subpopulation (the training points are visualized while the attack stops when the test error exceeds 50%). We can observe that, with the existence of more positively labeled (i.e., blue color) poisoning points (marked as "+") near the region of the negative label (i.e., red color) points, the decision boundary is gradually shifted to fit those "wrongly" labeled poisoning points, which in turn misclassifies the original subpopulation (i.e., orange colored points) into (blue) positive label. The exact locations and the labels of the generated poisoning points are determined by the maximization of the loss difference, as given in Line 4 in Algorithm 5. Interestingly, many of the poisoning points reside on only 3 distinct locations, which roughly justifies Theorem 2.2.1 that for binary classification, poisoning points from two distinct locations is sufficient to achieve the attack goals. Although our MTP attack does not necessarily limit itself to finding only two distinct poisoning points, the poisoning points found are naturally concentrated. Future poisoning attacks, at least for convex models, may just focus on optimizing the poisoning points from only a few promising locations (in high dimensions), as optimization of many distinct points may not be necessary.



(a) 39/155 Poisoning Points

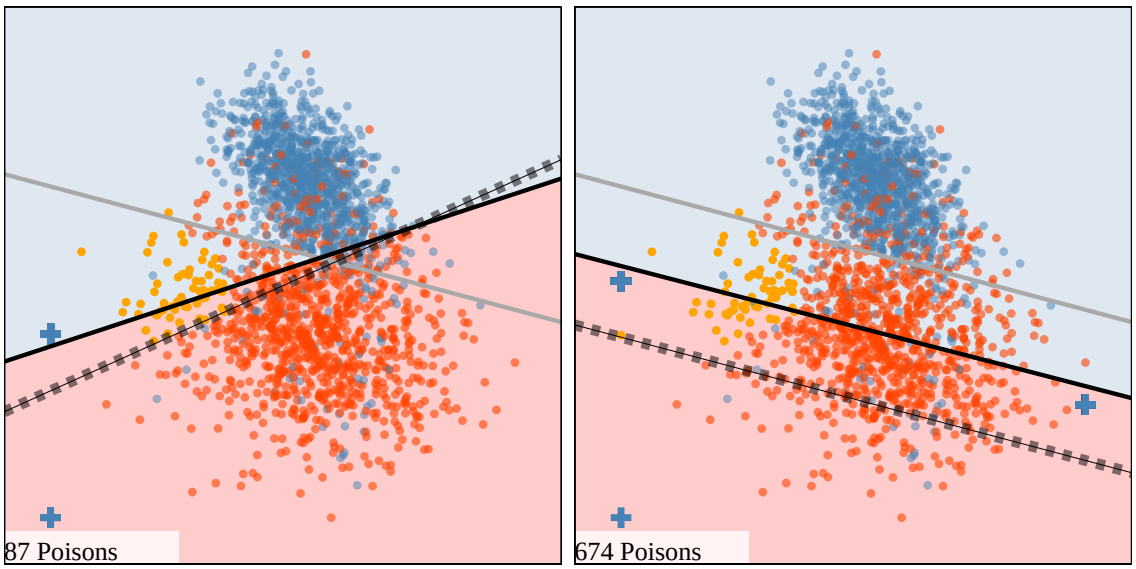
(b) 78/155 Poisoning Points

(c) 78/155 Poisoning Points

(d) 155/155 Poisoning Points

Figure 4.1: Visualization of the poisoning process to induce 50% test error on the selected subpopulation (better viewed in color). The selected subpopulation is denoted by the orange-colored area. The labels of the data points (including generated poisoning points marked as "+") are represented with given colors, and the blue color denotes the positive label and the red color denotes the negative label. The attack goal is to flip the original negative label (i.e., red color) of the orange-colored area into the positive label (i.e., blue color). The solid gray line denotes the clean decision boundary that is trained without poisoning. The dark solid line denotes the poisoned model after adding x out of the total 155 poisoning points into the training set. The dashed line denotes the target model that induces 100% test error on the selected subpopulation.

Importance of target model selection. Next, we provide the visualization on the importance of selecting the proper target model when running the MTP attack. In particular, in Section 3.3.1, we show that we desire a target model that satisfies the attack goal and also has a lower loss on $\mathcal{S}_c$. The drawback of choosing



(a) Target Model with Lower Loss on $\mathcal{S}_c$ (b) Target Model with Higher Loss on $\mathcal{S}_c$

Figure 4.2: Importance of choosing the proper target model for the MTP attack. The two figures compare the number of poisoning points needed to misclassify 50% of the same subpopulation (orange region), but using two different target models where Figure 4.2a shows the preferred model with a lower loss on $\mathcal{S}_c$. The solid gray line denotes the clean decision boundary that is trained without poisoning. The dark solid line denotes the poisoned model after adding poisoning points into the clean training set. The dashed line denotes the target model that induces 100% test error on the selected subpopulation.

a bad target model (i.e., a model with higher loss on $\mathcal{S}_c$) is that, it introduces other properties irrelevant to the attack goal on the subpopulation, while the proposed MTP attempts to induce the target model as closely as possible and unavoidably spends more poisoning points to achieve those undesired properties, which eventually increases the number of poisoning points used to achieve the relevant attack goal that matters. Visually, we can view the above argument as, using target models with a higher loss on $\mathcal{S}_c$, there will be higher resistance from the rest of the subpopulation (irrelevant to the attack goal on subpopulation) when we gradually move the decision boundary by adding poisoning points because the points from the rest of the population are originally correctly classified, but are now forced to be misclassified. Figure 4.2a shows the results of the target model that is generated by finding the model that has the lowest loss on $\mathcal{S}_c$ while also satisfying the attack goals, and the MTP attack only needs 87 poisoning points to misclassify 50% of the subpopulation. In comparison, a worse choice of target model as shown in Figure 4.2b will receive higher resistance from the rest of the poisoning points and leads to MTP using 647 poisoning points to induce 50% of test errors on the same subpopulation. Therefore, future exploration on model-targeted attacks should also focus on finding target models that satisfy the attack objectives while having the lowest possible loss on $\mathcal{S}_c$.

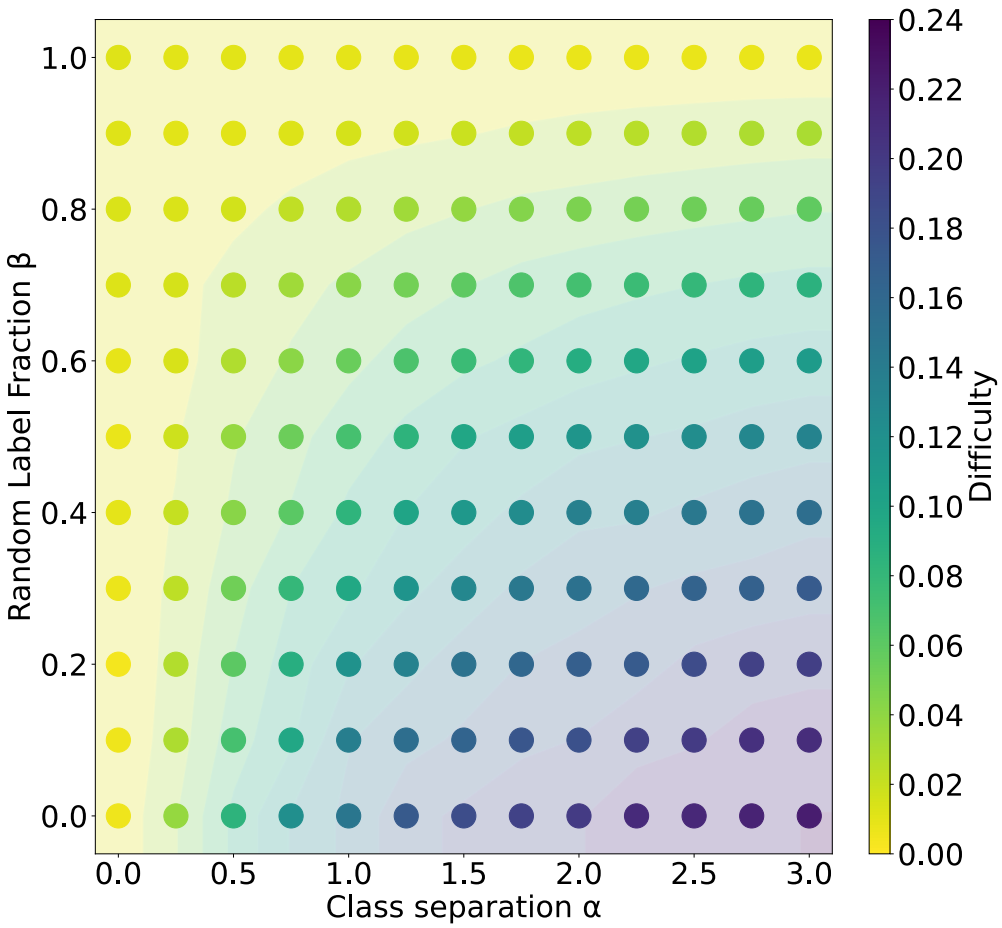


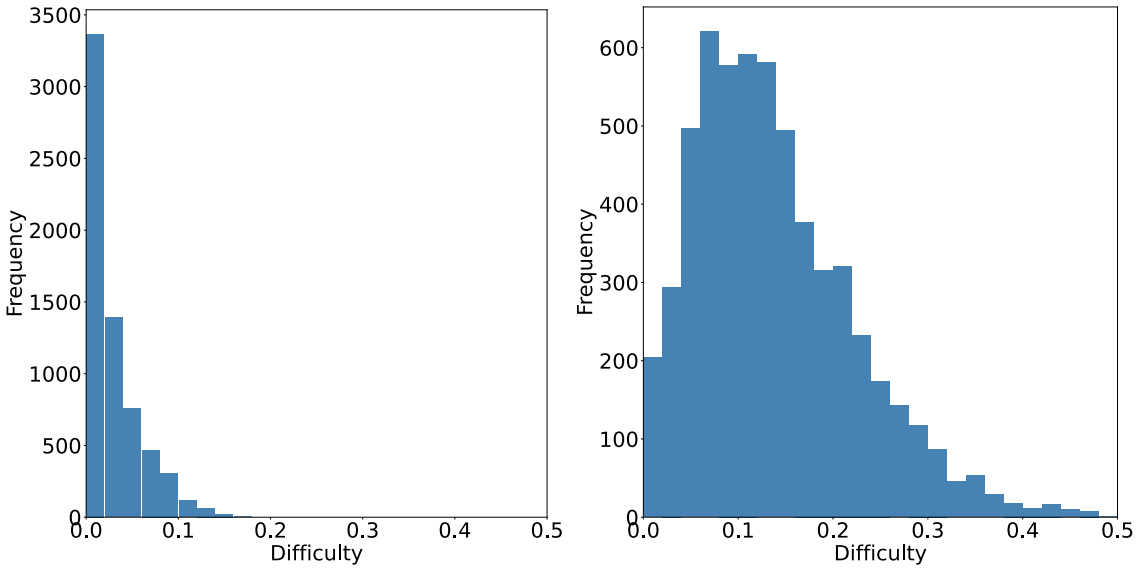
Figure 4.3: Comparison of the average subpopulation difficulty (of the total 16 subpopulations) of different synthetic datasets. The *difficulty* is measured as $|\mathcal{S}_p|/|\mathcal{S}_c|$, where $\mathcal{S}_p$ is generated to let the induced model misclassify 50% of the points in a subpopulation.

Next, we will first show the variation of subpopulation susceptibility and then characterize the subpopulation properties that impact their vulnerabilities to the MTP attack.

4.3.3 Subpopulation Susceptibility Variation

In this section, we describe the variation in subpopulation susceptibility by first showing the (dataset-level) average difficulty of subpopulations for each dataset and then providing a finer analysis of the variation across individual subpopulations for different datasets.

Average subpopulation susceptibility across datasets. We first explore the impact of the overall distributional properties on the vulnerabilities of the subpopulations, as these are high-level properties that might provide some general insights before digging deeper into particular subpopulation properties. Figure 4.3 shows the average difficulty of the total $k = 16$ subpopulations in each dataset for all the synthetic datasets

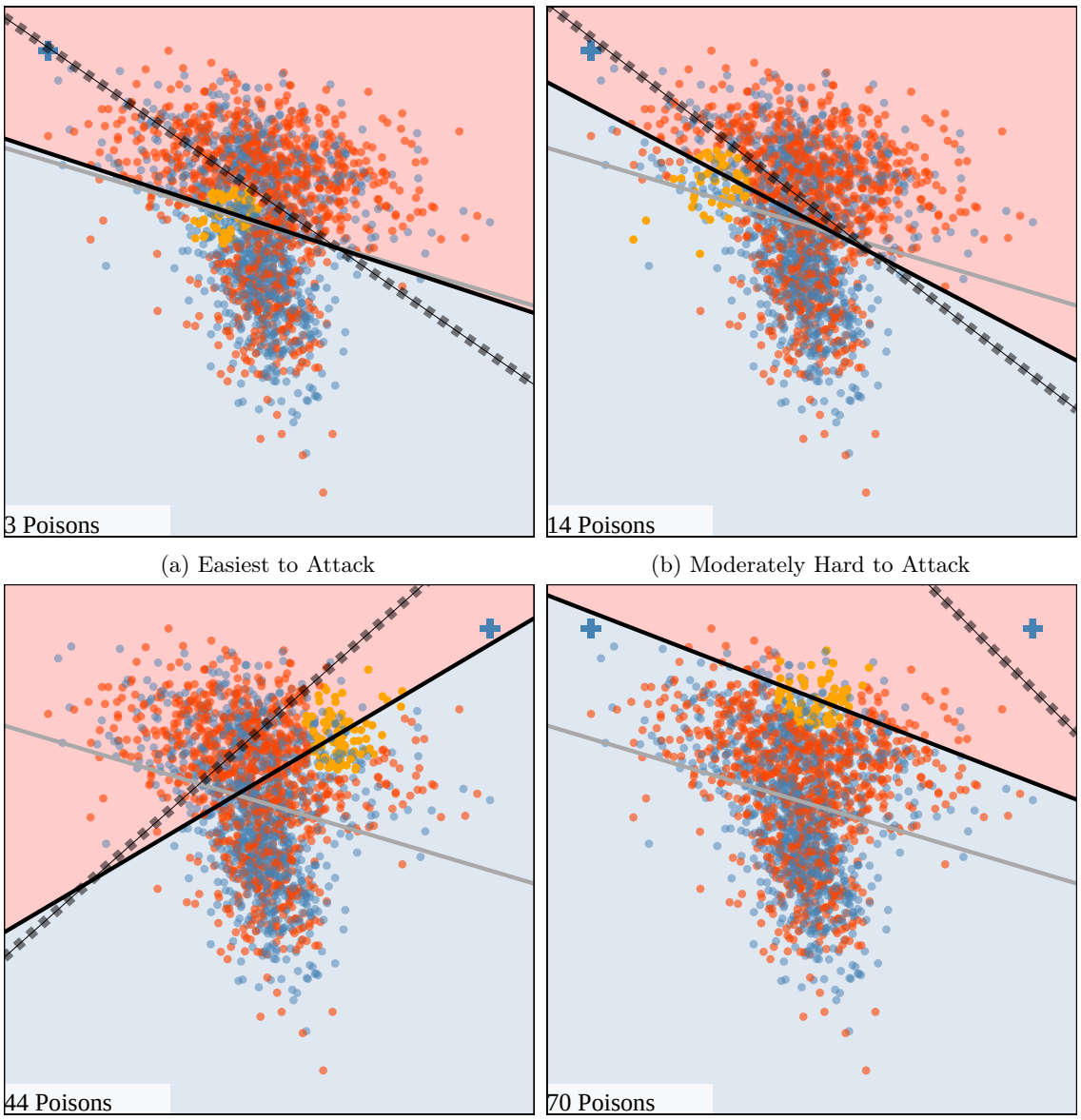


(a) Inaccurate Clean Models (b) Accurate Clean Models

Figure 4.4: Distribution of subpopulation susceptibilities for synthetic datasets of different clean accuracies under linear SVM. The x-axis denotes the measured attack difficulty and the y-axis denotes the frequency.

generated with different fractions of label noise β and class separation α. For datasets that are easier to classify (e.g., higher class separation α and lower fraction of label noise β from the distributional perspective), the average difficulty of the subpopulations increases, and vice versa. This observation is expected as the poorly separated datasets either already have high test errors on the subpopulations without poisoning or have points that are close to the decision boundary and are highly sensitive to misclassification when the decision boundary changes slightly, indicating that higher test errors on the subpopulations after poisoning can be easily achieved with a limited number of poisoning points. In contrast, for datasets that are well-separated by linear models, the subpopulations are far from the decision boundary with a lower number of misclassified points, and more poisoning points are needed to move the decision boundary (significantly) to incur the desired amount of test errors on the subpopulations. To conclude, the overall distributional properties of class separation and label noise indeed have a major impact on the vulnerabilities of the subpopulations, and subpopulations in less separable datasets are more vulnerable to poisoning.

Distribution of subpopulation susceptibilities. Once we have an understanding of the average difficulty for the subpopulations in each dataset, we further explore the variation of subpopulation susceptibility (measured by the attack difficulty) across different subpopulations for both the poorly- and well-separated datasets. We plot the frequencies of subpopulations with respect to the range of difficulty scores, as shown in Figure 4.4. From these experiments, we can clearly see that when the clean model accuracy is low (i.e., datasets are less separable under linear SVM), the majority of the subpopulations are easier to attack as the



(a) Easiest to Attack

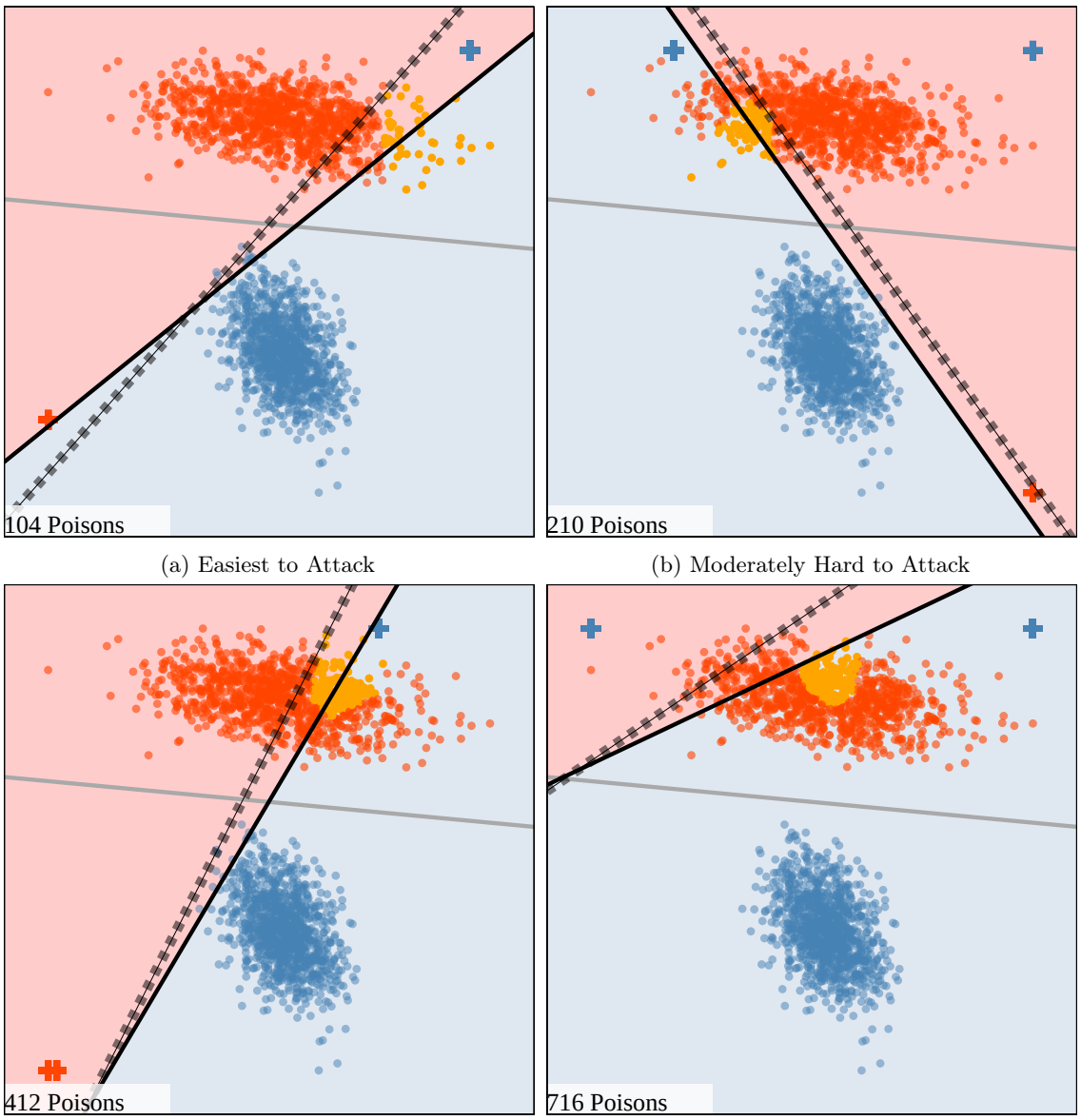
(b) Moderately Hard to Attack

(c) Moderately Hard to Attack

(d) Hardest to Attack

Figure 4.5: Variation of subpopulation susceptibilities in the non-linearly separable synthetic dataset. Each figure contains the number of poisoning points needed to have the induced model misclassify (as blue) 50% of the points in the subpopulation (colored in orange). The solid gray line denotes the clean decision boundary that is trained without poisoning. The dark solid line denotes the poisoned model after adding poisoning points into the clean training set. The dashed line denotes the target model that induces 100% test error on the selected subpopulation.

distributional properties dominate the subpopulation susceptibility in these cases. However, when the overall clean accuracy is high, the variation of the difficulty becomes more drastic and the impact of individual subpopulations matters more for the susceptibility, and further exploration of these properties is needed. Below, we provide initial insights on the possible subpopulation properties that impact the susceptibility by visualizing the attacked results of selected subpopulations for both the poorly- and well-separated datasets.



(a) Easiest to Attack

(b) Moderately Hard to Attack

(c) Moderately Hard to Attack

(d) Hardest to Attack

Figure 4.6: Variation of subpopulation susceptibilities in the linearly separable synthetic dataset. Each figure contains the number of poisoning points needed to have the induced model misclassify (as blue) 50% of the points in the subpopulation (colored in orange). The solid gray line denotes the clean decision boundary that is trained without poisoning. The dark solid line denotes the poisoned model after adding poisoning points into the clean training set. The dashed line denotes the target model that induces 100% test error on the selected subpopulation.

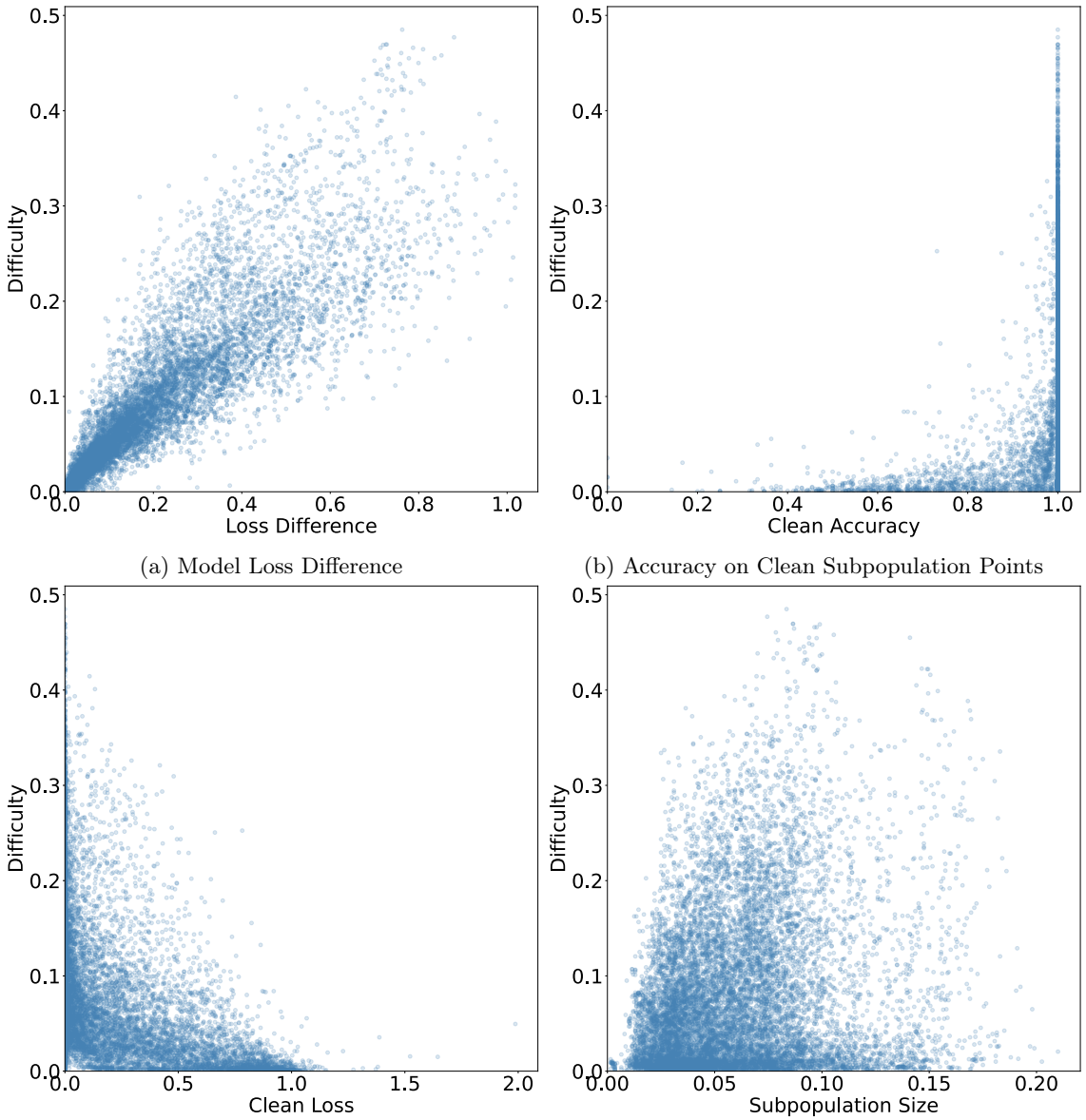
Understanding subpopulation susceptibilities through visualization. Figure 4.5 shows 4 different subpopulations that include the hardest, easiest, and moderately hard to attack subpopulations for a poorly-separated dataset under linear SVM. One obvious finding is, although the less separable dataset makes all of its subpopulations in general more vulnerable to poisoning (requires at most $70/2000 = 3.5\%$ of poisoning to attack all subpopulations), the individual subpopulation properties still lead to some minor variation in their

susceptibilities to poisoning. Expectedly, this variation across subpopulations is amplified for well-separated datasets, and Figure 4.6 shows 4 different subpopulations (hardest, easiest, and moderately hard to attack) in the well-separated case. In terms of the possible subpopulation properties that lead to this drastic variation, we can see visually from both Figure 4.5 and Figure 4.6 that, the relative position of the subpopulation to the rest of the population seems to be relevant, and the concrete subpopulation property that captures this "relative position" will be given next.

4.3.4 Characterization of Subpopulation Properties Impacting Susceptibility

So far, through extensive experiments, we have demonstrated that at the high level, the distributional properties matter more when the sampled datasets are poorly separated by linear models (while the individual subpopulations still contribute to minor variations) and the subpopulation properties that capture the "relative positions" of the subpopulations contribute more towards the susceptibility when the datasets are well-separated. In this section, we identify the relevant subpopulation properties. We tested four factors that are explicitly related to the properties of the subpopulations and also the underlying model (i.e., linear SVM in our case): 1) the model loss difference between the target model and the clean model on $\mathcal{S}_c$, where the target model has 100% error on the subpopulation with the lowest loss on $\mathcal{S}_c$; 2) the training accuracy of the clean model on the subpopulation; and 3) the training loss of the clean model on the subpopulation; 4) the size of the subpopulation.

Figure 4.7 shows the correlation between the four factors and the subpopulation susceptibility. We can see that only the model loss difference shows a strong correlation with the empirically observed susceptibility using the MTP attack and other factors of clean accuracy and clean loss on subpopulation and the size of the subpopulation all do not have a significant correlation. We believe the model loss difference is a reliable indicator because it implicitly captures the "relative position" of the subpopulation to the rest of the population using a target model that misclassifies the subpopulation, which otherwise might be hard to quantify directly. A smaller loss difference is likely to indicate that the subpopulation is more isolated compared to the rest and is closer to the clean decision boundary, which faces less resistance from the rest when moving the decision boundary and enables more efficient poisoning. Other factors such as the clean accuracy and clean loss are all related to the average margin of the clean points in the subpopulation to the decision boundary, but these factors do not capture the distribution of the rest of the subpopulation. If the subpopulation is very close to the decision boundary but is surrounded by the rest of the population, then misclassifying the subpopulation will unavoidably misclassify points from the rest of the population and require more number of poisoning points as there will be stronger resistance from other points. As for the



(a) Model Loss Difference

(b) Accuracy on Clean Subpopulation Points

(c) Loss on Clean Subpopulation Points

(d) Subpopulation Size

Figure 4.7: Correlation between the properties of the subpopulation and the susceptibilities for the synthetic dataset. Figure 4.7a shows the correlation of the model loss difference between the generated target model (has 100% error on the subpopulation and also the lowest loss on $\mathcal{S}_c$ among the generated candidate target models) and the clean model on $\mathcal{S}_c$. Figure 4.7b shows the correlation of the subpopulation clean accuracy. Figure 4.7c shows the correlation of the clean loss on the subpopulation. Figure 4.7d shows the correlation of the subpopulation size.

subpopulation size, expectedly, it also cannot characterize the susceptibility well as the subpopulation size may even fail to reflect the statistical properties of the subpopulation, not to mention the distribution of the rest of the population.

In these experiments, we only measure the correlation of the factors individually and observe the latter three

factors are not relevant. However, using more sophisticated ways to combine these irrelevant factors may still have strong predictive power on the subpopulation susceptibility. In addition, these irrelevant factors may still become relevant when we limit our scope to understanding the vulnerabilities of selected subpopulations. We leave these explorations as future work.

4.4 Experiments on Adult Dataset

In this section, we first show the experimental setup for the Adult dataset (Section 4.4.1). Then we show the generally descriptive subpopulation properties that are related to the subpopulation susceptibility (Section 4.4.2). Finally, we show some semantically meaningful subpopulation properties that are related to the susceptibility in selected settings (Section 4.4.3).

4.4.1 Experiment Setup

For the model training, as before, we use the Scikit-learn package to train the linear SVM models and set the hyperparameter $C_R = 0.09$. We use the FeatureMatch (Jagielski et al., 2019; 2021) approach that combines different attributes to generate the subpopulations for the Adult dataset, which are semantically more meaningful from the perspective of subpopulations that matter in practice and are also related to fairness in machine learning. In particular, to generate a semantic subpopulation, first, a subset of categorical features is selected, and specific values are chosen for those features. For example, the categorical features could be chosen to be "work class" and "education level", and the features' values could then be chosen to be "never worked" and "some college", respectively. Then, every negative label ("$\leq 50K$") instance in the training set matching all the (feature, label) pairs is extracted to form the subpopulation. The subpopulations for our experiments are chosen by considering every subset of categorical features and every combination of those features that is present in the training set. For simplicity, we only consider subpopulations with a maximum of three feature selections. In total, 4,338 subpopulations are formed using this method. Each of these subpopulations is attacked using the same attack as in the case of the synthetic dataset. Of these attacks, 1,602 are trivial (i.e., the clean model already satisfies the attacker objective), leaving 2,736 nontrivial attacks.

4.4.2 Variation of Subpopulation Susceptibility and Relevant Properties

We first show the drastic variation of subpopulation susceptibility in the Adult dataset in Figure 4.8. From the figure, we can clearly see the variation among subpopulations still exists for the high-dimensional benchmark dataset. Next, similar to the synthetic case studied in Section 4.3, we still proceed to explore the correlation of

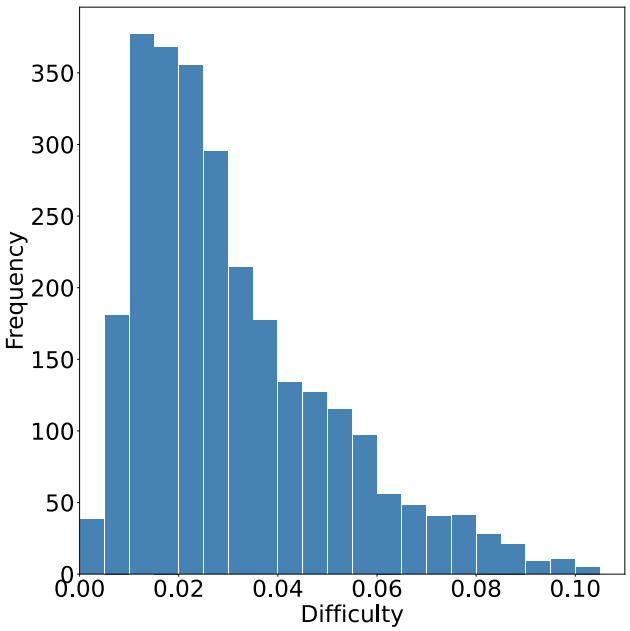


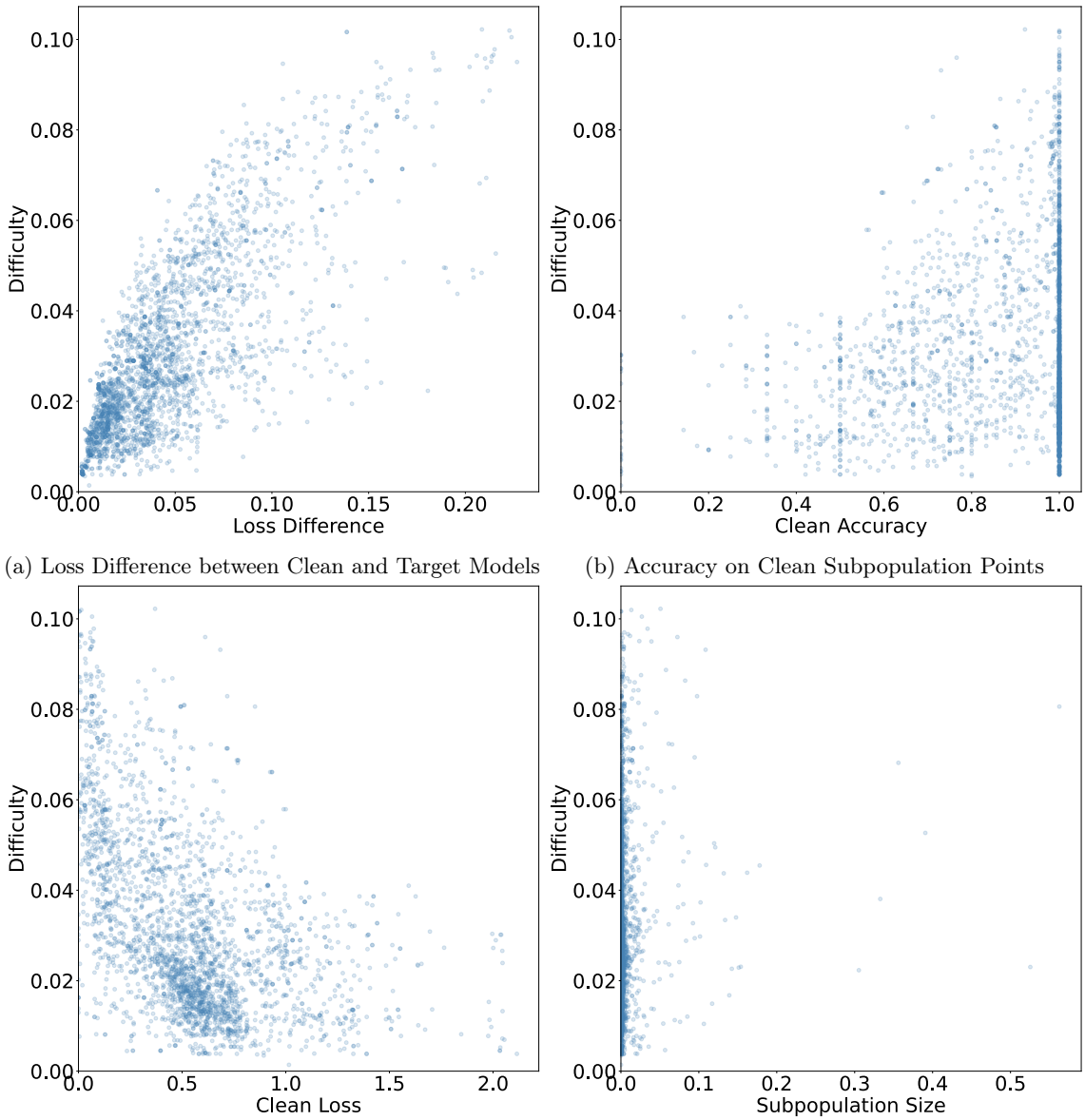
Figure 4.8: The distribution of the attack difficulties of subpopulations in the Adult dataset.

the 4 factors (i.e., model loss difference, accuracy on subpopulation, loss on subpopulation, and subpopulation size) to the subpopulation susceptibility in the Adult dataset. From Figure 4.9 we can see that, the model loss difference is a generalizable property and is still highly correlated to the susceptibility while other factors again fail to show a strong correlation for the Adult dataset. We believe the underlying reason for this observation is similar to the synthetic case, that the model loss difference well captures the relative position of the subpopulation while others don't.

4.4.3 Related Semantic Properties

Recall that for the Adult dataset, the subpopulations are generated using the FeatureMatch algorithm and hence, each subpopulation has a semantic meaning, and we can explore if there are identifiable semantic properties, rather than just the abstract description using the model loss difference, that can be related to the subpopulation susceptibility.

Through experiments, we find that the number of nearby points with different labels to the subpopulations can also be related to the subpopulation susceptibility. To show this, we first define *ambient positivity*. Specifically, for a subpopulation under binary classification with a given property P, we define the set of all points satisfying P the ambient subpopulation (since it also includes positive-label points), and call the fraction of points in the ambient subpopulation with a positive label the *ambient positivity* of the subpopulation. The relation of the ambient positivity and the subpopulation susceptibility is shown in Figure 4.10 for the



(a) Loss Difference between Clean and Target Models

(b) Accuracy on Clean Subpopulation Points

(c) Loss on Clean Subpopulation Points

(d) Subpopulation Size

Figure 4.9: Correlation between the properties of the Subpopulation and the susceptibilities for the Adult dataset. Model loss difference still denotes the loss difference between the generated target model (100% error on the subpopulation with the lowest loss on $\mathcal{S}_c$) and the clean model on $\mathcal{S}_c$.

selected subpopulations: these subpopulations are chosen to have 100% test accuracy by the clean model and are of similar sizes (ranges from 1% to 2% of the clean training set size $|\mathcal{S}_c|$). In the above attacks, attack difficulty is negatively correlated with the ambient positivity of the subpopulation. This makes sense since positive-label points near the subpopulation work to the advantage of the attacker when attempting to induce misclassification of the negative-label points (i.e., less resistance from the rest of the population). Stating in terms of the model-targeted attack, if the clean model classifies the ambient subpopulation as the negative

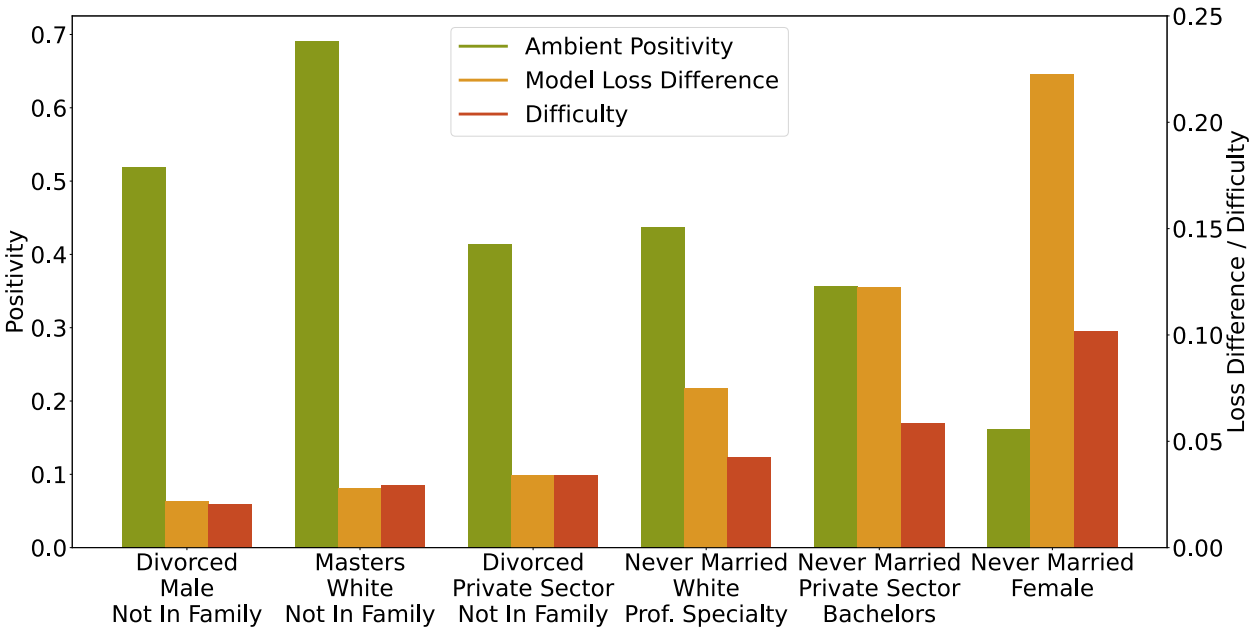


Figure 4.10: The correlation of ambient positivity on subpopulation susceptibility for selected subpopulations in the Adult dataset

label, then the loss difference between the target and clean models is smaller if there are positive-label points in that region. We did not test the ambient positivity for the synthetic case because subpopulations in the synthetic case are generated from (semantically less meaningful) clustering algorithms and hence, the majority of the subpopulations do not have many points with positive labels and the ambient positivity is also impacted significantly by the label-noise β when generating the synthetic dataset, making it a less significant factor related to the subpopulation susceptibility.

However, does the ambient positivity of a subpopulation necessarily determine attack difficulty for otherwise similar subpopulations? We find the answer is "No"—if we restrict our view to subpopulations with similar pre-poisoning ambient positivity (e.g., between 0.2 and 0.3), while still having 100% classification accuracy and similar subpopulation size, we still find a significant spread of attack difficulties as shown in Figure 4.11. This observation highlights the challenges in identifying generally applicable semantic properties for explaining the different subpopulation susceptibility. Related to this challenge in identifying related semantic properties, there are also subpopulations of different susceptibility that match on the same features but differ in the value of only a single feature. In addition, these subpopulations are similarly sized and also are perfectly classified by the clean model. For example, the two subpopulations that take similar values of "Clerical" for the feature "Occupation" and "Female" for the feature "Sex" only differ in the value of the feature "Relationship Status", and yet the one with the value "Not In Family" has an attack difficulty score of 0.07 while the other one with "Unmarried" has a difficulty score of only 0.02.

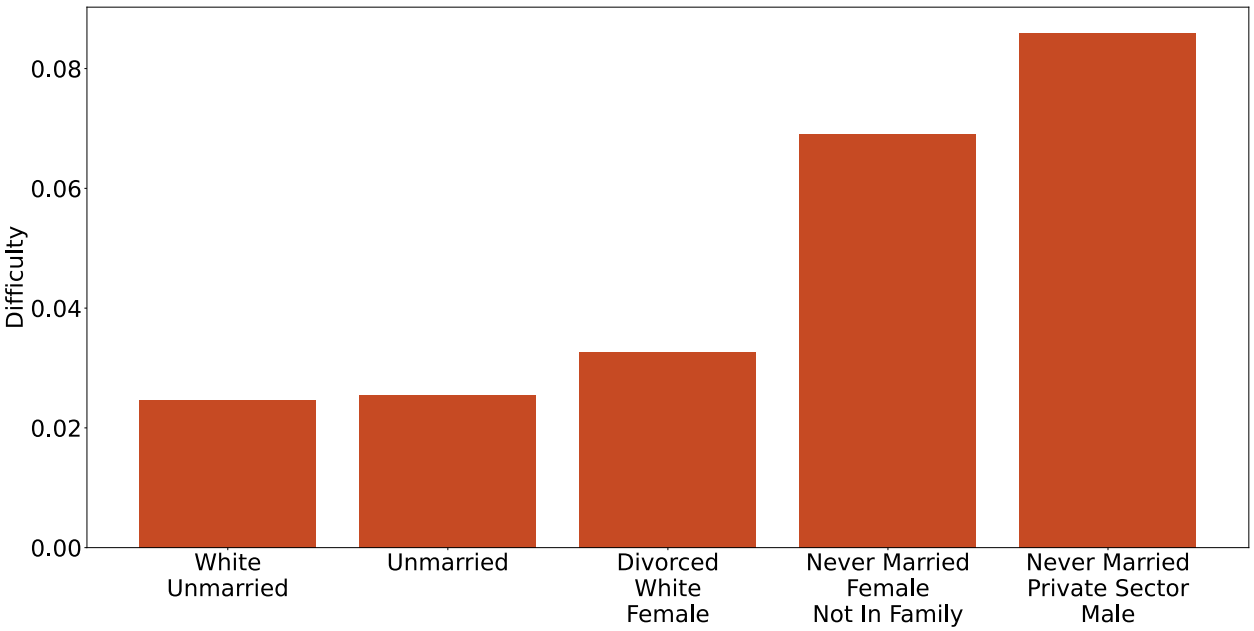


Figure 4.11: Variation of subpopulation susceptibility for subpopulations with similar ambient positivity in the Adult dataset.

4.5 Limitation and Discussion

Our results are limited because the subpopulation susceptibility is measured by the difficulty of the proposed MTP attack, while MTP is just a lower bound to the best possible poisoning attacks. Since our empirical observations are just proxies of the inherent susceptibility, future (stronger) poisoning attacks might also lead to some new insights in terms of the inherent susceptibility. Besides, our analysis is limited to simple datasets and a linear SVM model, and it is not yet clear how well they extend to more complex models. However, as a step towards better understanding of poisoning attacks and especially in understanding how attack difficulty varies with subpopulation characteristics, experiments in such a simplified setting are valuable and revealing. Further, simple and low-capacity models are still widely used in practice due to their ease of use, low computational cost, and effectiveness (Tramèr and Boneh, 2021; Ferrari Dacrema et al., 2019), and so our simplified analysis is still relevant in practice. Second, kernel methods or feature extraction layers in neural networks are powerful tools to handle non-linearly separable datasets by projecting them into a linearly separable high-dimensional or low-dimensional space and are widely adopted in practice. Therefore, if the important spatial relationships among the data points are still preserved after projection, then the same conclusions obtained in our simplified settings may still apply to the more complex cases by examining the spatial relationships in the transformed space. We leave these explorations as future work.

4.6 Summary

Through extensive experiments and visualizations, we show for the synthetic case that the overall distributional separability by linear models can dominate the subpopulation susceptibility when the sampled datasets are poorly separated, leading to mostly vulnerable subpopulations and the individual subpopulation properties only play some minor roles. In contrast, when the sampled datasets are well-separated, the subpopulation properties majorly determine the subpopulation susceptibility. In terms of the relevant subpopulation properties, we show the loss difference between the target model that misclassifies the subpopulation and the clean model on $\mathcal{S}_c$ is highly correlated with the susceptibility. The correlation of this property is generalizable and also holds for the benchmark Adult dataset. Other factors such as the subpopulation size and the average margin-related factors are not strongly correlated to the susceptibility in general for both the synthetic and benchmark datasets, due to the limitation in capturing the relative location of the subpopulation with respect to the rest of the population. For the Adult dataset, besides the general property of model loss difference, our experiments also identify a semantic property of the subpopulation that can be related to the susceptibility of a selected group of subpopulations, and finally highlight the challenges in finding generally relevant properties that are semantically meaningful. The observation of drastic variation of attack effectiveness across subpopulations also motivates us to explore the possibly different dataset susceptibility against indiscriminate attacks in Chapter 5, as the indiscriminate attacks can be treated as special subpopulation attacks that take the entire dataset as the subpopulation.

Chapter 5

Explaining Dataset Susceptibility[1]

5.1 Introduction

In Chapter 4, we observe the effectiveness of the state-of-the-art poisoning attack varies drastically across the subpopulations in a given dataset. This observation makes us wonder if the effectiveness of state-of-the-art indiscriminate poisoning attacks can also vary drastically across different datasets. This is because subpopulation attack is a very general concept and the indiscriminate attack can be interpreted as a special case of subpopulation attack where the entire dataset is the subpopulation. From this perspective, different datasets might correspond to different "subpopulations" and hence may still have drastically different susceptibility.

In fact, some prior works have demonstrated that state-of-the-art poisoning attacks can have drastically different effectiveness across datasets (Steinhardt et al., 2017; Koh et al., 2022; Lu et al., 2022; 2023), but these attacks focus on designing various indiscriminate attack methods that achieve empirically strong poisoning attacks in many settings (Steinhardt et al., 2017; Koh et al., 2022; Lu et al., 2022; 2023), but do not provide reasons on why these attacks are sometimes ineffective. In addition, the evaluations of these attacks can be deficient in some aspects (Biggio et al., 2011; 2012; Steinhardt et al., 2017; Billah et al., 2021; Koh et al., 2022; Lu et al., 2022) (see Section 5.2) and hence, may not be able to provide an accurate picture on the current progress of indiscriminate poisoning attacks on linear models. The goal of this chapter is to understand the properties of the learning tasks (more precisely, the distributional properties under the given

[1]This chapter is largely based on Fnu Suya, Xiao Zhang, Yuan Tian, David Evans, *When Can Linear Learners be Robust to Indiscriminate Poisoning Attacks?*, availabe online at: https://arxiv.org/abs/2307.01073.

learners) that help render attack effectiveness on linear models, as we will show in Section 5.2 that the current state-of-the-art attacks (including the MTP attack presented in Chapter 3) fail to effectively compromise the performance of the linear models trained on datasets such as MNIST 1–7. An attack is considered ineffective if the increased risk from poisoning is roughly equal to or smaller than the injected poisoning ratio (Lu et al., 2022; Koh et al., 2022).

This chapter is organized as follows: we first test the performance of state-of-the-art poisoning attacks (including the proposed MTP attack) on a variety of benchmark datasets (Section 5.2), and observe that all tested attacks are effective on some datasets while being fairly ineffective on others. To understand if there are data distributions that can be inherently robust to poisoning, we first define the optimal poisoning attack that attains maximum increased risk from poisoning (Section 5.3). Then, based on the definition of optimal poisoning attacks, we characterize the optimal attacks for the 1-D Gaussian distribution fully and partially for the general distributions and identify properties of the distributions under linear learners that impact the overall vulnerability to poisoning (Section 5.4). More importantly, we experimentally show that the identified factors are also correlated to the empirically observed vulnerabilities obtained by state-of-the-art poisoning attacks, on the synthetic (Section 5.5.1) and the benchmark datasets (Section 5.5.2). Then, we show how the insights on the limits of poisoning attacks might benefit future defenses (Section 5.6). In Section 5.7, we discuss the relationship of our work to other relevant works in the literature. Finally, we conclude the chapter and discuss the limitations of our work and possible future directions in Section 5.8.

5.2 Disparate Poisoning Vulnerability of Benchmark Datasets

Prior evaluations of poisoning attacks on *convex models* are inadequate in some aspects, either being tested on very small datasets (e.g., significantly subsampled MNIST 1–7 dataset) without competing baselines (Biggio et al., 2011; Demontis et al., 2019; Mei and Zhu, 2015a;b), generating invalid poisoning points (Steinhardt et al., 2017; Koh et al., 2022) or lacking diversity in the evaluated convex models/datasets (Lu et al., 2022; 2023). This motivates us to carefully evaluate representative attacks for linear models on various benchmark datasets without considering additional defenses, as considering defenses might obfuscate the results regarding the inherent vulnerabilities of the datasets under examination. By examining the state-of-the-art poisoning attacks, we mainly want to see if the current best indiscriminate attacks still have different effectiveness across datasets, similar to our observation across subpopulations in Chapter 4. The results in this section provide lower bounds on the limits of indiscriminate attacks across datasets, while in Section 5.5.2 we provide results on the upper bounds on the limits.

5.2.1 Experimental Setup

In this section, we provide details on the benchmark datasets, models, and attacks used for evaluations. Finally, we also discuss how to properly evaluate the vulnerability of the Dogfish (Koh and Liang, 2017) dataset, which interestingly "overfits" to the test data after poisoning.

Datasets and models. We test on various public benchmark datasets, including MNIST (LeCun, 1998) digit pairs of 1–7. 6–9, 4–9, Enron Metsis et al. (2006), Dogfish (Koh and Liang, 2017) Adult (Dua and Graff, 2017) and IMDB (Maas et al., 2011), which are all used in the evaluations of prior works (Steinhardt et al., 2017; Koh et al., 2022; Biggio et al., 2011; Jagielski et al., 2019; 2021) except for MNIST 6–9 and MNIST 4–9. MNIST 4–9 and MNIST 6–9 are picked to represent MNIST digit pairs that are relatively easier/harder to poison. We do not present the performance on IMDB in this section because running the existing state-of-the-art poisoning attacks on it is computationally too slow. Instead, we will directly quote the poisoned error of linear SVM from Koh et al. (2022) in Section 5.5.2 when demonstrating the correlation between the error increase and the identified vulnerability factors. We also construct a new dataset called *Filtered Enron*, which is obtained by filtering out 3% of near boundary points from Enron. The purpose is to construct a dataset with the lowest base test error (without poisoning) but the highest increased error after poisoning. For the Dogfish, Enron, and Filtered Enron datasets, we construct the constraint set $\mathcal{C}$ in the no-defense setting by finding the minimum ($u^i_{\min}$) and maximum ($u^i_{\max}$) values occurred in each feature dimension i for both the training and test data, which then forms a box constraint $[u^i_{\min}, u^i_{\max}]$ for each dimension. This way of construction is also used in the prior work (Koh et al., 2022).

For the victim models, we only consider linear models (linear SVM and LR) and the training (Pedregosa et al., 2011) of these models, and the attacks on them are stable (i.e., less randomness involved in the process) and so, we get almost identical results when feeding different random seeds. Therefore, we directly report the results in one run with one random seed. The regularization parameter C_R for training the linear models are configured as follows (unless specified next, the trained models include both the linear SVM and LR for the benchmark datasets): $C_R = 0.09$ for MNIST digit pairs, Adult, Dogfish, and SVM for Enron; $C_R = 0.01$ for IMDB, LR for Enron. Overall, the results and conclusions in this chapter are insensitive to the choice of C_R.

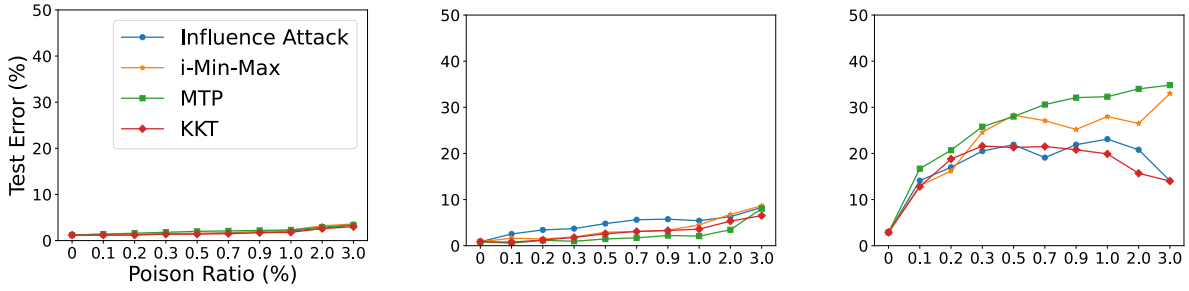
Attack details. We evaluate the state-of-the-art data poisoning attacks for linear models: *Influence Attack* (Koh and Liang, 2017; Koh et al., 2022), *KKT Attack* (Koh et al., 2022), *i-Min-Max Attack* (Steinhardt et al., 2017; Koh et al., 2022) (see details in Section 2.2), and our proposed MTP attack in Chapter 3. Many

Poison Ratio ϵ (%)	0.0	0.1	0.2	0.3	0.5	0.7	0.9	1.0	2.0	3.0
Train Error (%)	0.1	0.8	1.2	1.8	2.6	3.1	3.3	3.6	5.3	6.5
Test Error (%)	0.8	9.5	12.8	13.8	17.8	20.5	21.0	20.5	27.3	31.8

Table 5.1: Comparisons of the poisoned training and test errors for the Dogfish dataset. The poisoned errors are reported from the current best attacks.

of the attacks are evaluated by comparing the increased test errors at a fixed poisoning ratio ϵ and hence, the MTP attack in Chapter 3 will adopt the third stopping condition and terminate the attack when $\epsilon|\mathcal{S}_c|$ number of poisoning points are generated. We set $\epsilon = 3\%$ following previous works (Steinhardt et al., 2017; Koh et al., 2022; Lu et al., 2022; 2023). For these attacks, the KKT, MTP and i-Min-Max attacks require a target model as input and the target model is generated using the improved procedure described in Section 3.4.4 because the improved method generates better target models that help achieve higher errors on the clean test data after poisoning using the aforementioned three attacks, compared to the original method in Koh et al. (2022). Because the goal here is to achieve as high as possible test errors at the fixed poisoning budget $\epsilon = 3\%$, for attacks that require a target model as input, we first generate a set of candidate target models of different test errors (ranging from 5% to 70% with 5% increment) as it is hard to predetermine the best target model for the selected attack method. Then we run individual attacks separately to generate $\epsilon|\mathcal{S}_c|$ poisoning points for each of the generated target models and record the corresponding test errors after poisoning. Finally, for each tested attack, we set its poisoned test error as the highest one from all the poisoned errors recorded for the evaluated candidate target models.

Proper evaluation of the Dogfish dataset. Following the prior practice (Koh et al., 2022), in the threat model in 2.1.2, we considered adversaries that have access to both the clean training and test data, and therefore, adversaries can design attacks that can perform well on both the training and test data. This generally holds true for the tested benchmark datasets, except for the Dogfish dataset. For Dogfish, we find in our experiments that the attack "overfits" the test data heavily due to the small number of training and test data and also the high dimensionality. More specifically, we find that the poisoned model tends to incur significantly higher error rates on the clean test data compared to the clean training data. Since this high error cannot fully reflect the distributional risk, when we report the results of Dogfish in the rest of the chapter, we report the errors on both the training and the test data to give a better empirical sense of what the actual risk may look like. This also emphasizes the need to be cautious about the potential "overfitting" behavior when designing poisoning attacks. Table 5.1 shows the drastic differences between the errors of the clean training and test data after poisoning for the Dogfish dataset.



(a) MNIST 1-7 (b) Dogfish (c) Enron

Figure 5.1: Comparisons of the attack performance of existing data poisoning attacks on different benchmark datasets. Poisoning ratios are 0.1%, 0.2%, 0.3%, 0.5%, 0.7%, 0.9%, 1%, 2%, 3%.

5.2.2 Performance of the State-of-the-Art Poisoning Attacks

In this section, we first show the performance of all the selected state-of-the-art poisoning attacks on selective datasets and linear SVM for clarity in presentation, and then report the best attack result from the tested methods for all the benchmark datasets and linear models.

Performance of Different Attacks are Similar

We show the attack performance of different attacks on linear SVM models trained on the selected benchmark datasets of MNIST 1–7, Dogfish, and Enron. Other datasets also have similar patterns. Figure 5.1 summarizes the attack results of the tested attacks at different poisoning ratios of 0.1%, 0.2%, 0.3%, 0.5%, 0.7%, 0.9%, 1%, 2%, and 3%. The main observation is that different attacks perform mostly similarly for a given dataset, but their performance varies a lot across datasets. Our proposed MTP attack still performs comparably with or better than other attacks, especially for the more vulnerable Enron dataset.

Here, we tested different poisoning ratios instead of fixing the ϵ to 3% because, as can be seen from the results of Enron in Figure 5.1c, the KKT, Influence, and i-Min-Max attacks actually perform worse at higher poisoning ratios while we will prove in Theorem 5.3.8 that, under mild conditions (e.g., linear models), optimal poisoning attacks should always have non-decreasing risk as the poisoning budget increases. While there might be some chances that the attack performance can be improved by careful hyperparameter tuning, it may also be suggesting that these state-of-the-art attacks are suboptimal in some settings, as the test errors tend to decrease as the poisoning budget increases.

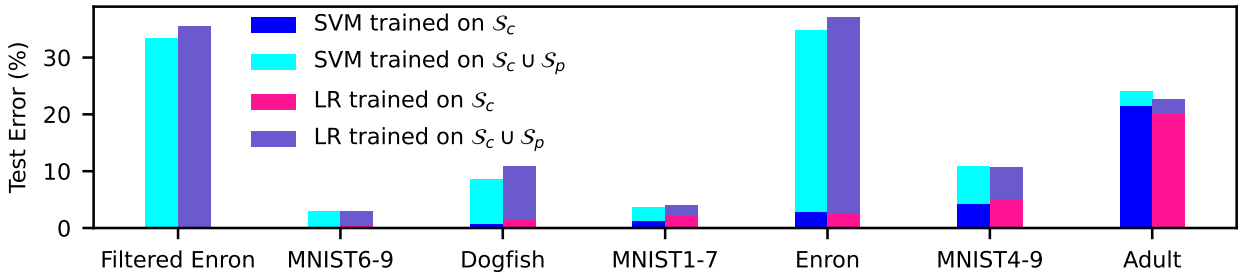


Figure 5.2: Performance of the best current indiscriminate poisoning attacks with $\epsilon = 3\%$ across different benchmark datasets. Datasets are sorted from lowest to highest base error rate (trained on $\mathcal{S}_c$) and the sorted order is the same for linear SVM and LR.

Reporting the Best Attack Results

Figure 5.2 shows the highest error from across the tested poisoning attacks (in most cases, all the attacks perform similarly). At the 3% poisoning ratio, the increased test errors of datasets such as MNIST 6–9 and MNIST 1–7 are less than 4% for both SVM and LR while for other datasets such as Dogfish, Enron, and Filtered Enron, the increased error is much higher than the injected poisoning ratio, indicating that these datasets are more vulnerable to poisoning. Dogfish is moderately vulnerable ($\approx 8\%$ increased error) while Enron and Filtered Enron are highly vulnerable with over 30% of increased error. Consistent with prior work (Steinhardt et al., 2017; Koh et al., 2022; Lu et al., 2023), throughout this paper, we measure the increased error to determine whether a dataset is vulnerable to poisoning attacks. However, in some security-critical applications, the ratio between the increased error and the initial error might matter more but leave its exploration for future work. These results reveal a drastic difference in the robustness of benchmark datasets to state-of-the-art indiscriminate data poisoning attacks which has not been explained in prior works. A natural question to ask from the above observation is *are datasets like MNIST digits inherently robust to poisoning attacks or just resilient to state-of-the-art attacks?* Since directly estimating the performance of optimal poisoning attacks for benchmark datasets is very challenging, we first explore and characterize optimal poisoning attacks for theoretical distributions and then study their partial characteristics for general distributions in Section 5.4.

5.3 Defining Optimal Poisoning Attacks

In this section, we lay out formal definitions of optimal poisoning attacks and study their general implications. For simplicity in presentation, for analysis in the remaining chapter, we incorporate the regularization term $C_R \cdot R(h)$ into the loss function $L(\cdot)$ (the loss and regularization terms were originally split as given in (2.1)

and (2.2)) and perform the risk or empirical risk minimization. Therefore, the individual loss of $(\boldsymbol{x}, y)$ with respect to h also incorporates an additional term $C_R \cdot R(h)$. For example, for any $\boldsymbol{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$, the *hinge loss* of a linear classifier $h_{\boldsymbol{w},b}$ now becomes:

$$\ell(h_{\boldsymbol{w},b}; \boldsymbol{x}, y) = \max\{0, 1 - y(\boldsymbol{w}^\top \boldsymbol{x} + b)\} + \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2. \tag{5.1}$$

With the setup above, we first introduce a notion of *finite-sample optimal poisoning* to formally define the optimal poisoning attack in the practical finite-sample setting with respect to our threat model:

Definition 5.3.1 (Finite-Sample Optimal Poisoning)**.** Consider input space $\mathcal{X}$ and label space $\mathcal{Y}$. Let μ_c be the underlying data distribution of clean inputs and labels. Let $\mathcal{S}_c$ be a set of examples sampled i.i.d. from μ_c. Suppose $\mathcal{H}$ is the hypothesis class and ℓ is the surrogate loss function that is used for learning. For any $\epsilon \geq 0$ and $\mathcal{C} \subseteq \mathcal{X} \times \mathcal{Y}$, a *finite-sample optimal poisoning adversary* $\hat{\mathcal{A}}_{\text{opt}}$ is defined to be able to generate some poisoned dataset $\mathcal{S}_p^*$ such that:

$$\mathcal{S}_p^* = \underset{\mathcal{S}_p}{\text{argmax}}\ \text{Risk}(\hat{h}_p; \mu_c) \quad \text{s.t. } \mathcal{S}_p \subseteq \mathcal{C} \text{ and } |\mathcal{S}_p| \leq \epsilon \cdot |\mathcal{S}_c|,$$

where $\hat{h}_p = \text{argmin}_{h \in \mathcal{H}}\ L(h; \mathcal{S}_c \cup \mathcal{S}_p)$ denotes the empirical loss minimizer.

Definition 5.3.1 suggests that no poisoning strategy can achieve a better attack performance than that achieved by $\hat{\mathcal{A}}_{\text{opt}}$. If we denote by $\hat{h}_p^*$ the hypothesis produced by minimizing the empirical loss on $\mathcal{S}_c \cup \mathcal{S}_p^*$, then $\text{Risk}(\hat{h}_p^*; \mu_c)$ can be regarded as the maximum achievable attack performance.

Next, we introduce a more theoretical notion of *distributional optimal poisoning*, which generalizes Definition 5.3.1 from finite-sample datasets to data distributions.

Definition 5.3.2 (Distributional Optimal Poisoning)**.** Consider the same setting as in Definition 5.3.1. A *distributional optimal poisoning adversary* $\mathcal{A}_{\text{opt}}$ is defined to be able to generate some poisoned data distribution μ_p^* such that:

$$(\mu_p^*, \delta^*) = \underset{(\mu_p, \delta)}{\text{argmax}}\ \text{Risk}(h_p; \mu_c) \quad \text{s.t. } \text{supp}(\mu_p) \subseteq \mathcal{C} \text{ and } 0 \leq \delta \leq \epsilon,$$

where $h_p = \text{argmin}_{h \in \mathcal{H}}\ \{L(h; \mu_c) + \delta \cdot L(h; \mu_p)\}$ denotes the population loss minimizer.

Similar to the finite-sample case, Definition 5.3.2 implies that there is no feasible poisoned distribution μ_p such that the risk of its induced hypothesis is higher than that attained by μ_p^*. Theorem 5.3.6 below connects

Definition 5.3.1 and Definition 5.3.2.

Before introducing the main theorem, we first introduce the formal definitions of strong convexity and Lipschitz continuity conditions with respect to a function, and the uniform convergence property with respect to a hypothesis class. These definitions are necessary for the proof of Theorem 5.3.6.

Definition 5.3.3 (Strong Convexity)**.** A function $f : \mathbb{R}^d \to \mathbb{R}$ is b-strongly convex for some $b > 0$, if $f(\boldsymbol{x}_1) \geq f(\boldsymbol{x}_2) + \nabla f(\boldsymbol{x}_2)^\top (\boldsymbol{x}_1 - \boldsymbol{x}_2) + \frac{b}{2}\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2^2$ for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d$.

Definition 5.3.4 (Lipschitz Continuity)**.** A function $f : \mathbb{R}^d \to \mathbb{R}$ is ρ-Lipschitz for some $\rho > 0$, if $|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)| \leq \rho\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2$ for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d$.

Definition 5.3.5 (Uniform Convergence)**.** Let $\mathcal{H}$ be a hypothesis class. We say that $\mathcal{H}$ satisfies the *uniform convergence property* with a loss function ℓ, if there exists a function $m_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ such that for every $\epsilon', \delta' \in (0,1)$ and for every probability distribution μ, if $\mathcal{S}$ is a set of examples with $m \geq m_{\mathcal{H}}(\epsilon', \delta')$ samples drawn i.i.d. from μ, then

$$\mathbb{P}_{\mathcal{S} \leftarrow \mu^m}\left[\sup_{h \in \mathcal{H}} \left|L(h; \hat{\mu}_{\mathcal{S}}) - L(h; \mu)\right| \leq \epsilon'\right] \geq 1 - \delta'.$$

Such a uniform convergence property, which can be achieved using the VC dimension or the Rademacher complexity of $\mathcal{H}$, guarantees that the learning rule specified by empirical risk minimization always returns a good hypothesis with high probability (Shalev-Shwartz and Ben-David, 2014). Similar to PAC learning, the function $m_{\mathcal{H}}$ measures the minimal sample complexity requirement that ensures uniform convergence.

Theorem 5.3.6. *Consider the same settings as in Definitions 5.3.1 and 5.3.2. Suppose $\mathcal{H}$ satisfies the uniform convergence property with function $m_{\mathcal{H}}(\cdot,\cdot)$. Assume ℓ is b-strongly convex and* $\mathrm{Risk}(h; \mu_c)$ *is ρ-Lipschitz continuous with respect to model parameters for some $b, \rho > 0$. Let $\hat{h}_p^* = \mathrm{argmin}_{h \in \mathcal{H}}\ L(h; \mathcal{S}_c \cup \mathcal{S}_p^*)$ and $h_p^* = \mathrm{argmin}_{h \in \mathcal{H}}\{L(h; \mu_c) + \delta^* \cdot L(h; \mu_p^*)\}$. For any $\epsilon', \delta' \in (0,1)$, if $|\mathcal{S}_c| \geq m_{\mathcal{H}}(\epsilon', \delta')$, then with probability at least $1 - \delta'$,*

$$\left|\mathrm{Risk}(\hat{h}_p^*; \mu_c) - \mathrm{Risk}(h_p^*; \mu_c)\right| \leq 2\rho\sqrt{\frac{\epsilon'}{b}}.$$

Remark 5.3.7. Theorem 5.3.6 assumes three regularity conditions to ensure the finite-sample optimal poisoning attack is a consistent estimator of the distributional optimal one (i.e., insights on poisoning from distributional settings can transfer to finite-sample settings): the uniform convergence property of $\mathcal{H}$ that guarantees empirical minimization of surrogate loss returns a good hypothesis, the strong convexity condition that

ensures a unique loss minimizer, and the Lipschitz condition that translates the closeness of model parameters to the closeness of risk. These conditions hold for most (properly regularized) convex problems and input distributions with bounded densities. The asymptotic convergence rate is determined by the function $m_{\mathcal{H}}$, which depends on the complexity of the hypothesis class $\mathcal{H}$ and the surrogate loss ℓ. For instance, if we choose the hyperparameter λ carefully for the regularization term of ℓ_2-norm, the sample complexity of the linear hypothesis class for a bounded hinge loss is $\Omega(1/(\epsilon')^2)$, where ϵ' is the error bound parameter for specifying the uniform convergence property (see Definition 5.3.5) and other problem-dependent parameters are hidden in the big-Ω notation (see Section 15 of Shalev-Shwartz and Ben-David (2014) for details). We note the generalization of optimal poisoning attack for the linear case is related to agnostic learning of halfspaces (Kalai et al., 2008), which also imposes assumptions on the underlying distribution such as anti-concentration assumption (Diakonikolas et al., 2020; Frei et al., 2021) similar to the Lipschitz continuity condition assumed in Theorem 5.3.6.

Moreover, we note that δ^* represents the ratio of injected poisoned data that achieves the optimal attack performance. In general, δ^* can be any value in $[0, \epsilon]$, but we show in Theorem 5.3.8 that optimal poisoning can always be achieved with ϵ-poisoning under mild conditions. Below, we first prove Theorem 5.3.6.

Proof of Theorem 5.3.6. First, we introduce the following notations to simplify the proof. Recall that, for any finite-sample set $\mathcal{S}$, denote by $\hat{\mu}_{\mathcal{S}}$ the empirical measure with respect to $\mathcal{S}$. For any $\mathcal{S}_p$, μ_p and $\delta \geq 0$, let

$$\hat{g}(\mathcal{S}_p, \mathcal{S}_c) = \operatorname*{argmin}_{h \in \mathcal{H}} \; L(h; \mathcal{S}_c \cup \mathcal{S}_p),$$

$$g(\delta, \mu_p, \mu_c) = \operatorname*{argmin}_{h \in \mathcal{H}} \{L(h; \mu_c) + \delta \cdot L(h; \mu_p)\}.$$

According to the definitions of $\hat{h}_p^*$ and h_p^*, we know $\hat{h}_p^* = \hat{g}(\mathcal{S}_p^*, \mathcal{S}_c)$ and $h_p^* = g(\delta^*, \mu_p^*, \mu_c)$.

Now we are ready to prove Theorem 5.3.6. For any $\mathcal{S}_c$ sampled from μ_c, consider the empirical loss minimizer $\hat{h}_p^* = \hat{g}(\mathcal{S}_p^*, \mathcal{S}_c)$ and the population loss minimizer $g(\delta_{\mathcal{S}_p^*}, \hat{\mu}_{\mathcal{S}_p^*}, \mu_c)$, where $\delta_{\mathcal{S}_p^*} = |\mathcal{S}_p^*|/|\mathcal{S}_c|$. Then $\mathcal{S}_p^* \cup \mathcal{S}_c$ can be regarded as the i.i.d. sample set from $(\mu_c + \delta_{\mathcal{S}_p^*} \cdot \hat{\mu}_{\mathcal{S}_p^*})/(1 + \delta_{\mathcal{S}_p^*})$. According to Definition 5.3.5, since $\mathcal{H}$ satisfies the uniform convergence property with respect to ℓ, we immediately know that the empirical loss minimization is close to the population loss minimization if the sample size is large enough (see Lemma 4.2 in Shalev-Shwartz and Ben-David (2014)). To be more specific, for any $\epsilon', \delta' \in (0, 1)$, if $|\mathcal{S}_c| \geq m_{\mathcal{H}}(\epsilon', \delta')$, then

with probability at least $1-\delta'$, we have

$$\begin{aligned} L\big(\hat{g}(\mathcal{S}_p^*, \mathcal{S}_c); \mu_c\big) + \delta_{\mathcal{S}_p^*} \cdot L\big(\hat{g}(\mathcal{S}_p^*, \mathcal{S}_c); \hat{\mu}_{\mathcal{S}_p^*}\big) &\leq \operatorname*{argmin}_{h \in \mathcal{H}} \{L(h; \mu_c) + \delta_{\mathcal{S}_p^*} \cdot L(h; \hat{\mu}_{\mathcal{S}_p^*})\} + 2\epsilon' \\ &= L\big(g(\delta_{\mathcal{S}_p^*}, \hat{\mu}_{\mathcal{S}_p^*}, \mu_c); \mu_c\big) + \delta_{\mathcal{S}_p^*} \cdot L\big(g(\delta_{\mathcal{S}_p^*}, \hat{\mu}_{\mathcal{S}_p^*}, \mu_c); \hat{\mu}_{\mathcal{S}_p^*}\big) + 2\epsilon'. \end{aligned}$$

In addition, since the surrogate loss ℓ is b-strongly convex and the population risk is ρ-Lipschitz, we further know the clean risk of $\hat{g}(\mathcal{S}_p^*, \mathcal{S}_c)$ and $g(\delta_{\mathcal{S}_p^*}, \hat{\mu}_{\mathcal{S}_p^*}, \mu_c)$ is guaranteed to be close. Namely, with probability at least $1-\delta'$, we have

$$\begin{aligned} \big|\mathrm{Risk}\big(\hat{g}(\mathcal{S}_p^*, \mathcal{S}_c); \mu_c\big) - \mathrm{Risk}\big(g(\delta_{\mathcal{S}_p^*}, \hat{\mu}_{\mathcal{S}_p^*}, \mu_c); \mu_c\big)\big| &\leq \rho \cdot \big\|\hat{g}(\mathcal{S}_p^*, \mathcal{S}_c) - g(\delta_{\mathcal{S}_p^*}, \hat{\mu}_{\mathcal{S}_p^*}, \mu_c)\big\|_2 \\ &\leq 2\rho\sqrt{\frac{\epsilon'}{b}}. \end{aligned}$$

Note that $\delta_{\mathcal{S}_p^*} \in [0, \epsilon]$ and $\mathrm{supp}(\hat{\mu}_{\mathcal{S}_p^*}) \subseteq \mathcal{C}$. Thus, according to the definition of $h_p^* = g(\delta^*, \mu_p^*, \mu_c)$, we further have

$$\begin{aligned} \mathrm{Risk}(h_p^*; \mu_c) \geq \mathrm{Risk}\big(g(\delta_{\mathcal{S}_p^*}, \hat{\mu}_{\mathcal{S}_p^*}, \mu_c); \mu_c\big) &\geq \mathrm{Risk}\big(\hat{g}(\mathcal{S}_p^*, \mathcal{S}_c); \mu_c\big) - 2\rho\sqrt{\frac{\epsilon'}{b}} \\ &= \mathrm{Risk}(\hat{h}_p^*; \mu_c) - 2\rho\sqrt{\frac{\epsilon'}{b}}. \end{aligned} \tag{5.2}$$

So far, we have proven one direction of the asymptotic for results Theorem 5.3.6.

On the other hand, we can always construct a subset $\tilde{\mathcal{S}}_p$ with size $|\tilde{\mathcal{S}}_p| = \delta^* \cdot |\mathcal{S}_c|$ by i.i.d. sampling from μ_p^*. Consider the empirical risk minimizer $\hat{g}(\tilde{\mathcal{S}}_p, \mathcal{S}_c)$ and the population risk minimizer $h_p^* = g(\delta^*, \mu_p^*, \mu_c)$. Similarly, since $\mathcal{H}$ satisfies the uniform convergence property, if $|\mathcal{S}_c| \geq m_{\mathcal{H}}(\epsilon', \delta')$, then with probability at least $1-\delta'$,we have

$$\begin{aligned} L\big(\hat{g}(\tilde{\mathcal{S}}_p, \mathcal{S}_c); \mu_c\big) + \delta^* \cdot L\big(\hat{g}(\tilde{\mathcal{S}}_p, \mathcal{S}_c); \mu_p^*\big) &\leq \operatorname*{argmin}_{h \in \mathcal{H}} \{L(h; \mu_c) + \delta^* \cdot L(h; \mu_p^*)\} + 2\epsilon' \\ &= L\big(g(\delta^*, \mu_p^*, \mu_c); \mu_c\big) + \delta^* \cdot L\big(g(\delta^*, \mu_p^*, \mu_c); \mu_p^*\big) + 2\epsilon'. \end{aligned}$$

According to the strong convexity of ℓ and the Lipschitz continuity of the population risk, we further have

$$\begin{aligned} \big|\mathrm{Risk}\big(\hat{g}(\tilde{\mathcal{S}}_p, \mathcal{S}_c); \mu_c\big) - \mathrm{Risk}\big(g(\delta^*, \mu_p^*, \mu_c); \mu_c\big)\big| &\leq \rho \cdot \big\|\hat{g}(\tilde{\mathcal{S}}_p, \mathcal{S}_c) - g(\delta^*, \mu_p^*, \mu_c)\big\|_2 \\ &\leq 2\rho\sqrt{\frac{\epsilon'}{b}}. \end{aligned}$$

Note that $\tilde{\mathcal{S}}_p \subseteq \mathcal{C}$ and $|\tilde{\mathcal{S}}_p| = \delta^* \cdot |\mathcal{S}_c| \leq \epsilon \cdot |\mathcal{S}_c|$. Thus, according to the definition of $\hat{h}_p^* = \hat{g}(\mathcal{S}_p^*, \mathcal{S}_c)$, we have

$$\begin{aligned}\text{Risk}(\hat{h}_p^*; \mu_c) &\geq \text{Risk}\big(\hat{g}(\tilde{\mathcal{S}}_p, \mathcal{S}_c); \mu_c\big) \geq \text{Risk}\big(g(\delta^*, \mu_p^*, \mu_c); \mu_c\big) - 2\rho\sqrt{\frac{\epsilon'}{b}} \\ &= \text{Risk}(h_p^*; \mu_c) - 2\rho\sqrt{\frac{\epsilon'}{b}}. \end{aligned} \tag{5.3}$$

Combining (5.2) and (5.3), we complete the proof of Theorem 5.3.6. □

Theorem 5.3.8. *The optimal poisoning attack performance defined in Definition 5.3.2 can always be achieved by choosing ϵ as the poisoning ratio, if either of the following conditions is satisfied:*

1. *The support of the clean distribution* $\text{supp}(\mu_c) \subseteq \mathcal{C}$.
2. *$\mathcal{H}$ is a convex hypothesis class, and for any $h_{\boldsymbol{\theta}} \in \mathcal{H}$, there always exists a distribution μ such that* $\text{supp}(\mu) \subseteq \mathcal{C}$ *and* $\frac{\partial}{\partial \boldsymbol{\theta}} L(h_{\boldsymbol{\theta}}; \mu) = \mathbf{0}$.

Remark 5.3.9. Theorem 5.3.8 characterizes the conditions under which the optimal performance is guaranteed to be achieved with the maximum poisoning ratio ϵ. Note that the first condition $\text{supp}(\mu_c) \subseteq \mathcal{C}$ is mild because it typically holds for poisoning attacks against undefended classifiers. When attacking classifiers that employ some defenses such as data sanitization, the condition $\text{supp}(\mu_c) \subseteq \mathcal{C}$ might not hold, due to the fact that the proposed defense may falsely reject some clean data points as outliers (i.e., related to false positive rates). The second condition complements the first one in that it does not require the victim model to be undefended, however, it requires $\mathcal{H}$ being convex. Following the proof of the main theorem, we also prove that for linear hypothesis with hinge loss, such a μ can be easily constructed. The theorem enables us to conveniently characterize the optimal poisoning attacks in Section 5.4.1 by directly using ϵ. When the required conditions are satisfied, this theorem also provides a simple sanity check on whether a poisoning attack is optimal. In particular, if a candidate attack is optimal, the risk of the induced model is monotonically non-decreasing with respect to the poisoning ratio.

Proof of Theorem 5.3.8. We prove Theorem 5.3.8 by construction.

We start with the first condition $\text{supp}(\mu_c) \subseteq \mathcal{C}$. Suppose $\delta^* < \epsilon$, since the theorem trivially holds if $\delta^* = \epsilon$. To simplify notations, define $h_p(\delta, \mu_p) = \text{argmin}_{h \in \mathcal{H}} \{L(h; \mu_c) + \delta \cdot L(h; \mu_p)\}$ for any δ and μ_p. To prove the statement in Theorem 5.3.8, it is sufficient to show that there exists some $\mu_p^{(\epsilon)}$ based on the first condition

such that

$$\text{Risk}\big(h_p(\epsilon, \mu_p^{(\epsilon)}); \mu_c\big) = \text{Risk}\big(h_p(\delta^*, \mu_p^*); \mu_c\big), \text{ and } \text{supp}(\mu_p^{(\epsilon)}) \subseteq \mathcal{C}. \tag{5.4}$$

The above equality means we can always achieve the same maximum risk after poisoning with the full poisoning budget ϵ. To proceed with the proof, we construct $\mu_p^{(\epsilon)}$ based on μ_c and μ_p^* as follows:

$$\begin{aligned}\mu_p^{(\epsilon)} &= \frac{\delta^*}{\epsilon} \cdot \mu_p^* + \frac{\epsilon - \delta^*}{\epsilon(1+\delta^*)} \cdot (\mu_c + \delta^* \cdot \mu_p^*) \\ &= \frac{\epsilon - \delta^*}{\epsilon(1+\delta^*)} \cdot \mu_c + \frac{\delta^*(1+\epsilon)}{\epsilon(1+\delta^*)} \cdot \mu_p^*.\end{aligned}$$

We can easily check that $\mu_p^{(\epsilon)}$ is a valid probability distribution and $\text{supp}(\mu_p^{(\epsilon)}) \subseteq \mathcal{C}$. In addition, we can show that

$$\begin{aligned}h_p(\epsilon, \mu_p^{(\epsilon)}) &= \operatorname*{argmin}_{h \in \mathcal{H}} \{L(h; \mu_c) + \epsilon \cdot L(h; \mu_p^{(\epsilon)})\} \\ &= \operatorname*{argmin}_{h \in \mathcal{H}} \left\{\mathbb{E}_{(\boldsymbol{x},y)\sim\mu_c} \ell(h; \boldsymbol{x}, y) + \epsilon \cdot \mathbb{E}_{(\boldsymbol{x},y)\sim\mu_p^{(\epsilon)}} \ell(h; \boldsymbol{x}, y)\right\} \\ &= \operatorname*{argmin}_{h \in \mathcal{H}} \left\{\frac{1+\epsilon}{1+\delta^*} \cdot \left(\mathbb{E}_{(\boldsymbol{x},y)\sim\mu_c} \ell(h; \boldsymbol{x}, y) + \delta^* \cdot \mathbb{E}_{(\boldsymbol{x},y)\sim\mu_p^*} \ell(h; \boldsymbol{x}, y)\right)\right\} \\ &= h_p(\delta^*, \mu_p^*)\end{aligned}$$

where the third equality holds because of the construction of $\mu_p^{(\epsilon)}$. Therefore, we have proven (5.4), which further implies the optimal attack performance can always be achieved with ϵ-poisoning as long as the first condition is satisfied.

Next, we turn to the second condition of Theorem 5.3.8. Similarly, it is sufficient to construct some $\mu_p^{(\epsilon)}$ for the setting where $\delta^* < \epsilon$ such that

$$\text{Risk}\big(h_p(\epsilon, \mu_p^{(\epsilon)}); \mu_c\big) = \text{Risk}\big(h_p(\delta^*, \mu_p^*); \mu_c\big), \text{ and } \text{supp}(\mu_p^{(\epsilon)}) \subseteq \mathcal{C}.$$

We construct $\mu_p^{(\epsilon)}$ based on μ_p^* and the assumed data distribution μ. More specifically, we construct

$$\mu_p^{(\epsilon)} = \frac{\delta^*}{\epsilon} \cdot \mu_p^* + \frac{\epsilon - \delta^*}{\epsilon} \cdot \mu. \tag{5.5}$$

By construction, we know $\mu_p^{(\epsilon)}$ is a valid probability distribution. In addition, according to the assumption of $\text{supp}(\mu) \subseteq \mathcal{C}$, we have $\text{supp}(\mu_p^{(\epsilon)}) \subseteq \mathcal{C}$. According to the assumption that for any $\boldsymbol{\theta}$, there exists a μ such that

$\frac{\partial}{\partial_{\boldsymbol{\theta}}} L(h_{\boldsymbol{\theta}}; \mu) = \mathbf{0}$, we know for any possible weight parameter $\boldsymbol{\theta}_p^*$ of $h_p(\delta^*, \mu_p^*)$, there also exists a corresponding μ such that the gradient is $\mathbf{0}$ and therefore, we have

$$\begin{aligned}
&\frac{\partial}{\partial_{\boldsymbol{\theta}_p^*}} \big(L(h_p(\delta^*, \mu_p^*); \mu_c) + \epsilon \cdot L(h_p(\delta^*, \mu_p^*); \mu_p^{(\epsilon)}) \big) \\
&\quad = \frac{\partial}{\partial_{\boldsymbol{\theta}_p^*}} \big(L(h_p(\delta^*, \mu_p^*); \mu_c) + \delta^* \cdot L(h_p(\delta^*, \mu_p^*); \mu_p^*) \big) \\
&\quad = \mathbf{0}
\end{aligned}$$

where the last equality is based on the first-order optimality condition of $h_p(\delta^*, \mu_p^*)$ for convex losses. For simplicity, we also assumed $h_p(\delta^*, \mu_p^*)$ is obtained by minimizing the loss on $\mu_c + \delta^* \cdot \mu_p^*$ while in the case of $\mathrm{supp}(\mu_c) \nsubseteq \mathcal{C}$, the victim usually minimizes the loss on $\bar{\mu}_c + \delta^* \cdot \mu_p^*$, where $\bar{\mu}_c$ is the "truncated" version of μ_c such that $\mathrm{supp}(\bar{\mu}_c) \subseteq \mathcal{C}$. To conclude, we know $h_p(\epsilon, \mu_p^{(\epsilon)}) = h_p(\delta^*, \mu_p^*)$ holds for any possible $h_p(\delta^*, \mu_p^*)$ and we complete the proof of Theorem 5.3.8.

□

Proof of the Statement about Linear Models in Remark 5.3.9. We provide the construction of μ with respect to the second condition of Theorem 5.3.8 for linear models and hinge loss. Since for any $h_{\boldsymbol{w},b} \in \mathcal{H}$ and any $(\boldsymbol{x}, y) \in \mathcal{X} \times \mathcal{Y}$, we have

$$\ell(h_{\boldsymbol{w},b}; \boldsymbol{x}, y) = \max\{0, 1 - y(\boldsymbol{w}^\top \boldsymbol{x} + b)\} + \frac{\lambda}{2} \|\boldsymbol{w}\|_2^2.$$

Let $\boldsymbol{\theta} = (\boldsymbol{w}, b)$, then the gradient with respect to $\boldsymbol{w}$ can be written as:

$$\frac{\partial}{\partial_{\boldsymbol{w}}} \ell(h_{\boldsymbol{w},b}; \boldsymbol{x}, y) = \begin{cases} -y \cdot \boldsymbol{x} + \lambda \boldsymbol{w} & \text{if } y(\boldsymbol{w}^\top \boldsymbol{x} + b) \leq 1, \\ \lambda \boldsymbol{w} & \text{otherwise.} \end{cases}$$

Similarly, the gradient with respect to b can be written as:

$$\frac{\partial}{\partial_b} \ell(h_{\boldsymbol{w},b}; \boldsymbol{x}, y) = \begin{cases} -y & \text{if } y(\boldsymbol{w}^\top \boldsymbol{x} + b) \leq 1, \\ 0 & \text{otherwise.} \end{cases}$$

Therefore, for large input space $\mathcal{X} \times \mathcal{Y}$, we can simply construct μ by constructing a two-point distribution (with equal probabilities of label +1 and -1) that cancels out each other's gradient (for both $\boldsymbol{w}$ and b), so that $y(\boldsymbol{w}^\top \boldsymbol{x} + b) \leq 1$ and $-y \cdot \boldsymbol{x} + \lambda \boldsymbol{w} = \mathbf{0}$.

We may also generalize the construction of μ from linear models with hinge loss to general convex models if the victim minimizes the loss on $\bar{\mu}_c + \delta^* \cdot \mu_p^*$ to obtain $h_p(\delta^*, \mu_p^*)$, which is common in practice (e.g., training models on filtered datasets from data sanitization defenses). In this case, we can simply set $\mu = \bar{\mu}_c + \delta^* \cdot \mu_p^*$, which guarantees

$$\frac{\partial}{\partial_{\boldsymbol{\theta}_p^*}} L(h_p(\delta^*, \mu_p^*); \mu) = \mathbf{0}.$$

□

5.4 Characterizing Optimal Poisoning Attacks

This section characterizes the distributional optimal poisoning attacks with respect to the linear hypothesis class. We first consider a theoretical 1-dimensional Gaussian mixture model and exactly characterize optimal poisoning attack, and then discuss the implications of the underlying factors that potentially cause the inherent vulnerabilities to poisoning attacks for general high-dimensional distributions.

5.4.1 One-Dimensional Gaussian Mixtures

Consider binary classification tasks with one-dimensional inputs, where $\mathcal{X} = \mathbb{R}$ and $\mathcal{Y} = \{-1, +1\}$. Let μ_c be the underlying clean data distribution, where each example (x, y) is assumed to be i.i.d. sampled according to the following Gaussian mixture model:

$$\begin{cases} y = -1, x \sim \mathcal{N}(\gamma_1, \sigma_1^2) & \text{with probability } p, \\ y = +1, x \sim \mathcal{N}(\gamma_2, \sigma_2^2) & \text{with probability } 1 - p, \end{cases} \tag{5.6}$$

where $\sigma_1, \sigma_2 > 0$ and $p \in (0, 1)$. Without loss of generality, we assume $\gamma_1 \leq \gamma_2$. Following our threat model, we let $\epsilon \geq 0$ be the maximum poisoning ratio and $\mathcal{C} = \mathcal{Q}(u) := [-u, u] \times \mathcal{Y}$ for some $u > 0$ be the constraint set. Let $\mathcal{H} = \{h_{w,b} : w \in \{-1, 1\}, b \in \mathbb{R}\}$ be the linear hypothesis class with normalized weights. Note that we consider a simplified setting where the weight parameter $w \in \{-1, 1\}$. Characterizing the optimal poisoning attack under the general setting of $w \in \mathbb{R}$ is more challenging (even in the 1-D setting) since we need to consider the effect of any possible choice of w and its interplay with the dataset and constraint set factors. We leave the theoretical analyses of $w \in \mathbb{R}$ to future work. Since $\|w\|_2$ is fixed, we also set $\lambda = 0$ in the hinge loss function (5.1). To begin with, we introduce two definitions that will be used when characterizing the optimal poisoning attacks.

Definition 5.4.1 (Two-point Distribution)**.** For any $\alpha \in [0,1]$, ν_α is defined as a *two-point distribution*, if for any (x, y) sampled from ν_α,

$$(x,y) = \begin{cases} (-u,+1) & \text{with probability } \alpha, \\ (u,-1) & \text{with probability } 1-\alpha. \end{cases} \tag{5.7}$$

Definition 5.4.2 (Weight-Flipping Condition)**.** Consider the assumed Gaussian mixture model (5.6) and the linear hypothesis class $\mathcal{H}$. Let g be an auxiliary function such that for any $b \in \mathbb{R}$,

$$g(b) = \frac{1}{2}\Phi\left(\frac{b+\gamma_1+1}{\sigma}\right) - \frac{1}{2}\Phi\left(\frac{-b-\gamma_2+1}{\sigma}\right),$$

where Φ is the cumulative distribution function (CDF) of standard Gaussian $\mathcal{N}(0,1)$. Let $\epsilon > 0$ be the poisoning budget and g^{-1} be the inverse of g, then the *weight-flipping condition* is defined as:

$$\max\{\Delta(-\epsilon), \Delta(g(0)), \Delta(\epsilon)\} \geq 0, \tag{5.8}$$

where $\Delta(s) = L(h_{1,g^{-1}(s)}; \mu_c) - \min_{b\in\mathbb{R}} L(h_{-1,b}; \mu_c) + \epsilon \cdot (1+u) - s \cdot g^{-1}(s)$.

Now we are ready to present our main theoretical results. The following theorem rigorously characterizes the behavior of the distributional optimal poisoning adversary $\mathcal{A}_{\text{opt}}$ under the Gaussian mixture model (5.6) and the corresponding optimal attack performance:

Theorem 5.4.3. *Suppose the clean distribution μ_c follows the Gaussian mixture model* (5.6) *with $p = 1/2$, $\gamma_1 \leq \gamma_2$, and $\sigma_1 = \sigma_2 = \sigma$. Assume $u \geq 1$ and $|\gamma_1 + \gamma_2| \leq 2(u-1)$. There always exists some $\alpha \in [0,1]$ such that the optimal attack performance defined in Definition 5.3.2 is achieved with $\delta = \epsilon$ and $\mu_p = \nu_\alpha$, where ν_α is defined by* (5.7)*. More specifically, if $h_p^* = \operatorname{argmin}_{h\in\mathcal{H}}\{L(h;\mu_c) + \epsilon \cdot L(h;\nu_\alpha)\}$ denotes the induced hypothesis with optimal poisoning, then*

$$\text{Risk}(h_p^*; \mu_c) = \begin{cases} \Phi\left(\frac{\gamma_2-\gamma_1}{2\sigma}\right) & \text{if condition (5.8) is satisfied,} \\ \frac{1}{2}\Phi\left(\frac{\gamma_1-\gamma_2+2s}{2\sigma}\right) + \frac{1}{2}\Phi\left(\frac{\gamma_1-\gamma_2-2s}{2\sigma}\right) & \text{otherwise,} \end{cases}$$

where $s = \max\{g^{-1}(\epsilon) - g^{-1}(0), g^{-1}(0) - g^{-1}(-\epsilon)\}$ and $g(\cdot)$ is defined in Definition 5.4.2.

The proof of Theorem 5.4.3 is given in Section 5.4.2. Below, we provide a high-level proof sketch. We first prove that in order to understand the optimal poisoning attacks, it is sufficient to study the family of

two-point distributions (Definition 5.4.1) as the poisoned data distribution. Based on this reduction and a specification of weight flipping condition (Definition 5.4.2), we then rigorously characterize the optimal attack performance with respect to different configurations of task-related parameters.

Remark 5.4.4. Theorem 5.4.3 characterizes the exact behavior of $\mathcal{A}_{\text{opt}}$ for typical combinations of hyperparameters under the considered model, including distribution-related parameters such as γ_1, γ_2, σ and poisoning related parameters such as ϵ, u. A larger u suggests the weight-flipping condition (5.8) is more likely to be satisfied, as an attacker can generate poisoned data with larger hinge loss to flip the weight parameter w. Class separability $|\gamma_1 - \gamma_2|$ and within-class variance σ also play an important role in affecting the optimal attack performance. If the ratio $|\gamma_1 - \gamma_2|/\sigma$ is large, then we know the initial risk $\text{Risk}(h_c; \mu_c) = \Phi(\frac{\gamma_1 - \gamma_2}{2\sigma})$ will be small. Consider the case where condition (5.8) is satisfied. Note that $\Phi(\frac{\gamma_2 - \gamma_1}{2\sigma}) = 1 - \Phi(\frac{\gamma_1 - \gamma_2}{2\sigma})$ implies an improved performance of optimal poisoning attack, thus a higher inherent vulnerability to data poisoning attacks. However, it is worth noting that there is an implicit assumption in condition (5.8) that the weight parameter can be flipped from $w = 1$ to $w = -1$. A large value of $|\gamma_1 - \gamma_2|/\sigma$ also implies that flipping the weight parameter becomes more difficult, since the gap between the hinge loss with respect to μ_c for a hypothesis with $w = -1$ and that with $w = 1$ becomes larger. Moreover, if condition (5.8) cannot be satisfied, then a larger ratio of $|\gamma_1 - \gamma_2|/\sigma$ suggests that it is more difficult to move the decision boundary to incur an increase in test error, because the number of correctly classified boundary points will increase at a faster rate. In summary, Theorem 5.4.3 suggests that a smaller value of u and a larger ratio of $|\gamma_1 - \gamma_2|/\sigma$ increases the inherent robustness to indiscriminate poisoning for typical configurations under our model (5.6). Empirical verification of the above theoretical results is given in Section 5.5.1.

Moreover, Theorem 5.4.3 suggests a specific method for producing the optimal poisoned distribution/dataset for the considered theoretical 1-D distribution/dataset, which we term as optimal poisoning (OPT) for the distributional setting and Empirical OPT for the finite-sample setting. For OPT attack, ν_α is the poisoning distribution with α chosen from $\{0, 1, \alpha_0\}$ depending on the property of μ_c and whether flipping w to -1 is feasible, where $\alpha_0 = 1/2 + g(0)/(2\epsilon)$. For the Empirical OPT attack, we just need to generate the i.i.d. samples from ν_α with the corresponding values of α to form the $\epsilon|\mathcal{S}_c|$ poisoned set and measure the test error after poisoning.

5.4.2 Proof of Theorem 5.4.3

To prove Theorem 5.4.3, we need to make use of the following three auxiliary lemmas, which are related to the maximum population hinge loss with $w = 1$ (Lemma 5.4.5), the weight-flipping condition (Lemma

5.4.6) and the risk behavior of any linear hypothesis under (5.6) (Lemma 5.4.7). For the sake of completeness, we present the full statements of Lemma 5.4.5 and Lemma 5.4.6 as follows. In particular, Lemma 5.4.5 characterizes the maximum achievable hinge loss with respect to the underlying clean distribution μ_c and some poisoned distribution μ_p conditioned on $w = 1$. The proofs of these technical lemmas are deferred to the Appendix for better flow in presentation.

Lemma 5.4.5. *Suppose the underlying clean distribution μ_c follows the Gaussian mixture model* (5.6) *with $p = 1/2$ and $\sigma_1 = \sigma_2 = \sigma$. Assume $|\gamma_1 + \gamma_2| \leq 2u$. For any $\epsilon \geq 0$, consider the following maximization problem:*

$$\max_{\mu_p \in \mathcal{Q}(u)} \big[L(h_{1,b_p}; \mu_c) + \epsilon \cdot L(h_{1,b_p}; \mu_p)\big], \tag{5.9}$$

where $b_p = \operatorname{argmin}_{b\in\mathbb{R}}[L(h_{1,b}; \mu_c) + \epsilon \cdot L(h_{1,b}; \mu_p)]$. There exists some $\alpha \in [0, 1]$ such that the optimal value of (5.9) *is achieved with $\mu_p = \nu_\alpha$, where ν_α is a two-point distribution with some parameter $\alpha \in [0, 1]$ defined according to* (5.7).

Lemma 5.4.5 suggests that it is sufficient to study the extreme two-point distributions ν_α with $\alpha \in [0, 1]$ to understand the maximum achievable population hinge loss conditioned on $w = 1$. Lemma 5.4.6, proven in Appendix 1.2, characterizes the sufficient and necessary conditions in terms of ϵ, u and μ_c, under which there exists a linear hypothesis with $w = -1$ that achieves the minimal value of population hinge loss with respect to μ_c and some μ_p, and is where the weight-flipping condition in Definition 5.4.2 stems from.

Lemma 5.4.6. *Suppose the underlying clean distribution μ_c follows the Gaussian mixture model* (5.6) *with $p = 1/2$ and $\sigma_1 = \sigma_2 = \sigma$. Assume $|\gamma_1 + \gamma_2| \leq 2(u - 1)$ for some $u \geq 1$. Let g be an auxiliary function such that for any $b \in \mathbb{R}$,*

$$g(b) = \frac{1}{2}\Phi\bigg(\frac{b + \gamma_1 + 1}{\sigma}\bigg) - \frac{1}{2}\Phi\bigg(\frac{-b - \gamma_2 + 1}{\sigma}\bigg),$$

where Φ is the CDF of standard Gaussian. For any $\epsilon > 0$, there exists some $\mu_p \in \mathcal{Q}(u)$ such that $\operatorname{argmin}_{h_{w,b}\in\mathcal{H}}[L(h_{w,b}; \mu_c) + \epsilon \cdot L(h_{w,b}; \mu_p)]$ outputs a hypothesis with $w = -1$, if and only if

$$\max\{\Delta(-\epsilon), \Delta(g(0)), \Delta(\epsilon)\} \geq 0,$$

where $\Delta(s) = L(h_{1,g^{-1}(s)}; \mu_c) - \min_{b\in\mathbb{R}} L(h_{-1,b}; \mu_c) + \epsilon(1 + u) - s \cdot g^{-1}(s)$, and g^{-1} denotes the inverse of g.

Lemma 5.4.6 identifies sufficient and necessary conditions when a linear hypothesis with flipped weight parameter is possible. Note that we assume $\gamma_1 \leq \gamma_2$, thus flipping the weight parameter of the induced model

from $w = 1$ to $w = -1$ is always favorable from an attacker's perspective. In particular, if the population hinge loss with respect to μ_c and some μ_p achieved by the loss minimizer conditioned on $w = 1$ is higher than that achieved by the loss minimizer with $w = -1$, then we immediately know that flipping the weight parameter is possible, which further suggests the optimal poisoning attack performance must be achieved by some poisoned victim model with $w = -1$.

Finally, we introduce Lemma 5.4.7, proven in Appendix 1.3, which characterizes the risk behavior of any linear hypothesis with respect to the assumed Gaussian mixture model (5.6).

Lemma 5.4.7. *Let μ_c be the clean data distribution, where each example is sampled i.i.d. according to the data generating process specified in* (5.6)*. For any linear hypothesis $h_{w,b} \in \mathcal{H}$, we have*

$$\text{Risk}(h_{w,b}; \mu_c) = p \cdot \Phi\left(\frac{b + w \cdot \gamma_1}{\sigma_1}\right) + (1 - p) \cdot \Phi\left(\frac{-b - w \cdot \gamma_2}{\sigma_2}\right),$$

where Φ denotes the CDF of the standard Gaussian distribution $\mathcal{N}(0, 1)$.

Now we are ready to prove Theorem 5.4.3 using Lemmas 5.4.5, 5.4.6 and 5.4.7.

Proof of Theorem 5.4.3. According to Theorem 5.3.8 and Remark 5.3.9, we note that the optimal poisoning performance in Definition 5.3.2 is always achieved with $\delta = \epsilon$. Therefore, we will only consider $\delta = \epsilon$ in the following discussions.

Since the optimal poisoning performance is defined with respect to clean risk, it will be useful to understand the properties of $\text{Risk}(h_{w,b}; \mu_c)$ such as monotonicity and range. According to Lemma 5.4.7, for any $h_{w,b} \in \mathcal{H}$, we have

$$\text{Risk}(h_{w,b}; \mu_c) = \frac{1}{2}\Phi\left(\frac{b + w \cdot \gamma_1}{\sigma}\right) + \frac{1}{2}\Phi\left(\frac{-b - w \cdot \gamma_2}{\sigma}\right).$$

To understand the monotonicity of risk, we compute its derivative with respect to b:

$$\frac{\partial}{\partial b}\text{Risk}(h_{w,b}; \mu_c) = \frac{1}{2\sigma\sqrt{2\pi}}\left[\exp\left(-\frac{(b + w \cdot \gamma_1)^2}{2\sigma^2}\right) - \exp\left(-\frac{(b + w \cdot \gamma_2)^2}{2\sigma^2}\right)\right].$$

If $w = 1$, then $\text{Risk}(h_{w,b}; \mu_c)$ is monotonically decreasing when $b \in (-\infty, -\frac{\gamma_1+\gamma_2}{2})$ and monotonically increasing when $b \in (-\frac{\gamma_1+\gamma_2}{2}, \infty)$, suggesting that minimum is achieved at $b = -\frac{\gamma_1+\gamma_2}{2}$ and maximum is achieved when b goes to infinity. To be more specific, $\text{Risk}(h_{1,b}; \mu_c) \in [\Phi(\frac{\gamma_1-\gamma_2}{2\sigma}), \frac{1}{2}]$. On the other hand, if $w = -1$, then $\text{Risk}(h_{w,b}; \mu_c)$ is monotonically increasing when $b \in (-\infty, \frac{\gamma_1+\gamma_2}{2})$ and monotonically decreasing

when $b \in (\frac{\gamma_1+\gamma_2}{2}, \infty)$, suggesting that maximum is achieved at $b = \frac{\gamma_1+\gamma_2}{2}$ and minimum is achieved when b goes to infinity. Thus, $\text{Risk}(h_{-1,b}; \mu_c) \in [\frac{1}{2}, \Phi(\frac{\gamma_2-\gamma_1}{2\sigma})]$.

Based on the monotonicity analysis of $\text{Risk}(h_{w,b}; \mu_c)$, we have the following two observations:

1. If there exists some feasible μ_p such that $h_{-1,b_p} = \text{argmin}_{h\in\mathcal{H}}\{L(h; \mu_c) + \epsilon L(h; \mu_p)\}$ can be achieved, then the optimal poisoning performance is achieved with $w = -1$ and b close to $\frac{\gamma_1+\gamma_2}{2}$ as much as possible.
2. If there does not exist any feasible μ_p that induces h_{-1,b_p} by minimizing the population hinge loss, then the optimal poisoning performance is achieved with $w = 1$ and b far from $-\frac{\gamma_1+\gamma_2}{2}$ as much as possible (conditioned that the variance σ is the same for the two classes).

Recall that we prove in Lemma 5.4.6 a sufficient and necessary condition for the existence of such h_{-1,b_p}, which is equivalent to the condition (5.8) presented in Definition 5.4.2. Note that according to Lemma 1.1, $b = \frac{\gamma_1+\gamma_2}{2}$ also yields the population loss minimizer with respect to μ_c conditioned on $w = -1$. Thus, if condition (5.8) is satisfied, then we know there exists some $\alpha \in [0, 1]$ such that the optimal poisoning performance can be achieved with $\mu_p = \nu_\alpha$. This follows from the assumption $|\gamma_1 + \gamma_2| \leq 2(u - 1)$, which suggests that for any $(x, y) \sim \nu_\alpha$, the individual hinge loss at (x, y) will be zero. In addition, we know that the poisoned hypothesis induced by $\mathcal{A}_{\text{opt}}$ is $h_{-1,\frac{\gamma_1+\gamma_2}{2}}$, which maximizes risk with respect to μ_c.

On the other hand, if condition (5.8) is not satisfied, we know that the poisoned hypothesis induced by any feasible μ_p has weight parameter $w = 1$. Based on our second observation, this further suggests that the optimal poisoning performance will always be achieved with either $\mu_p = \nu_0$ or $\mu_p = \nu_1$. According to the first-order optimality condition and Lemma 1.1, we can compute the closed-form solution regarding the optimal poisoning performance. Thus, we complete the proof. □

5.4.3 General Distributions

Recall that we have identified several key factors (i.e., u, $|\gamma_1 - \gamma_2|$ and σ) for 1-D Gaussian distributions in Section 5.4.1 which are highly related to the performance of an optimal distributional poisoning adversary $\mathcal{A}_{\text{opt}}$. In this section, we demonstrate how to generalize the definition of these factors to high-dimensional distributions and illustrate how they affect an inherent robustness upper bound on indiscriminate poisoning attacks for linear learners. In particular, we project the clean distribution μ_c and the constraint set $\mathcal{C}$ onto some vector $\boldsymbol{w}$, then compute those factors based on the projections.

Definition 5.4.8 (Projected Constraint Size)**.** Let $\mathcal{C} \subseteq \mathcal{X} \times \mathcal{Y}$ be the constraint set for poisoning. For any $\boldsymbol{w} \in \mathbb{R}^d$, the *projected constraint size* of $\mathcal{C}$ with respect to $\boldsymbol{w}$ is defined as:

$$\text{Size}_{\boldsymbol{w}}(\mathcal{C}) = \max_{(\boldsymbol{x},y)\in\mathcal{C}} \boldsymbol{w}^\top \boldsymbol{x} - \min_{(\boldsymbol{x},y)\in\mathcal{C}} \boldsymbol{w}^\top \boldsymbol{x}$$

According to Definition 5.4.8, $\text{Size}_{\boldsymbol{w}}(\mathcal{C})$ characterizes the size of the constraint set $\mathcal{C}$ when projected onto the (normalized) projection vector $\boldsymbol{w}/\|\boldsymbol{w}\|_2$ then scaled by $\|\boldsymbol{w}\|_2$, the ℓ_2-norm of $\boldsymbol{w}$. In theory, the constraint sets conditioned on $y = -1$ and $y = +1$ can be different, but for simplicity and practical considerations, we simply assume they are the same in the following discussions.

Definition 5.4.9 (Projected Separability and Standard Deviation)**.** Let $\mathcal{X} \subseteq \mathbb{R}^d$, $\mathcal{Y} = \{-1, +1\}$, and μ_c be the underlying distribution. Let μ_- and μ_+ be the input distributions with labels of -1 and $+1$ respectively. For any $\boldsymbol{w} \in \mathbb{R}^d$, the *projected separability* of μ_c with respect to $\boldsymbol{w}$ is defined as:

$$\text{Sep}_{\boldsymbol{w}}(\mu_c) = \left|\mathbb{E}_{\boldsymbol{x}\sim\mu_-}[\boldsymbol{w}^\top \boldsymbol{x}] - \mathbb{E}_{\boldsymbol{x}\sim\mu_+}[\boldsymbol{w}^\top \boldsymbol{x}]\right|.$$

In addition, the *projected standard deviation* of μ_c with respect to $\boldsymbol{w}$ is defined as:

$$\text{SD}_{\boldsymbol{w}}(\mu_c) = \sqrt{\text{Var}_{\boldsymbol{w}}(\mu_c)},\ \text{Var}_{\boldsymbol{w}}(\mu_c) = p_- \cdot \text{Var}_{\boldsymbol{x}\sim\mu_-}[\boldsymbol{w}^\top \boldsymbol{x}] + p_+ \cdot \text{Var}_{\boldsymbol{x}\sim\mu_+}[\boldsymbol{w}^\top \boldsymbol{x}],$$

where $p_- = \Pr_{(\boldsymbol{x},y)\sim\mu_c}[y = -1]$, $p_+ = \Pr_{(\boldsymbol{x},y)\sim\mu_c}[y = +1]$ denote the sampling probabilities.

For finite-sample settings, we simply replace the input distributions with their empirical counterparts to compute the sample statistics of $\text{Sep}_{\boldsymbol{w}}(\mu_c)$ and $\text{SD}_{\boldsymbol{w}}(\mu_c)$. Note that the above definitions are specifically for linear models, but the insights can still be partially applicable to non-linear models such as neural networks, and more discussion on this can be found in Section 5.7. Below, we provide justifications on how the three factors are related to the optimal poisoning attacks. Theorem 5.4.10 and the techniques used in its proof in Section 5.4.4 are inspired by the design of Min-Max Attack (Steinhardt et al., 2017).

Theorem 5.4.10. *Consider input space $\mathcal{X} \subseteq \mathbb{R}^d$, label space $\mathcal{Y}$, clean distribution μ_c and linear hypothesis class $\mathcal{H}$. For any $h_{\boldsymbol{w},b} \in \mathcal{H}$, $\boldsymbol{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$, let $\ell(h_{\boldsymbol{w},b}; \boldsymbol{x}, y) = \ell_{\text{M}}(-y(\boldsymbol{w}^\top \boldsymbol{x} + b)) + C_R \cdot R(h_{\boldsymbol{w},b})$ be a margin-based loss adopted by the victim, where ℓ_M is convex and non-decreasing. Let $\mathcal{C} \subseteq \mathcal{X} \times \mathcal{Y}$ be the constraint set and $\epsilon > 0$ be the poisoning budget. Suppose $h_c = \text{argmin}_{h\in\mathcal{H}} L(h; \mu_c)$ has weight $\boldsymbol{w}_c$ and h_p^* is*

the poisoned model induced by optimal adversary $\mathcal{A}_{\text{opt}}$, then we have

$$\text{Risk}(h_p^*; \mu_c) \leq \min_{h \in \mathcal{H}} \left[L(h; \mu_c) + \epsilon \cdot L(h; \mu_p^*) \right] \leq L(h_c; \mu_c) + \epsilon \cdot \ell_{\text{M}}(\text{Size}_{\boldsymbol{w}_c}(\mathcal{C})) + \epsilon C_R \cdot R(h_c). \tag{5.10}$$

Remark 5.4.11. Theorem 5.4.10 proves an upper bound on the inherent vulnerability to indiscriminate poisoning for linear learners, which can be regarded as a necessary condition for the optimal poisoning attack. A smaller upper bound likely suggests a higher inherent robustness to poisoning attacks. In particular, the right-hand side of (5.10) consists of two terms: the clean population loss of h_c and a term related to the projected constraint size. Intuitively, the projected separability and standard deviation metrics highly affect the first term, since a data distribution with a higher $\text{Sep}_{\boldsymbol{w}_c}(\mu_c)$ and a lower $\text{SD}_{\boldsymbol{w}_c}(\mu_c)$ implies a larger averaged margin with respect to h_c, which further suggests a smaller $L(h_c; \mu_c)$. The second term is determined by the poisoning budget ϵ and the projected constraint size, or more precisely, a larger ϵ and a larger $\text{Size}_{\boldsymbol{w}_c}(\mathcal{C})$ indicate a higher upper bound on $\text{Risk}(h_p^*; \mu_c)$. In addition, we set $h = h_c$ and the projection vector as $\boldsymbol{w}_c$ for the last inequality of (5.10), because h_c achieves the smallest population surrogate loss with respect to the clean data distribution μ_c. However, choosing $h = h_c$ may not always produce a tighter upper bound on $\text{Risk}(h_p^*; \mu_c)$ since there is no guarantee that the projected constraint size $\text{Size}_{\boldsymbol{w}_c}(\mathcal{C})$ (and $R(h_c)$) will be small. An interesting future direction is to select a more appropriate projection vector that returns a tight, if not the tightest, upper bound on $\text{Risk}(h_p^*; \mu_c)$ for any clean distribution μ_c.

Relation to the Min-Max attack. The upper bound on the risk of optimal poisoning attacks in (5.10) is, in essence, similar to the upper bound in the Min-Max attack shown below, which plugs h_c into (2.14) given in Section 2.2.2 and considers the upper bound in the distributional setting:

$$\text{Risk}(h_p^*; \mu_c) \leq \min_{h \in \mathcal{H}} \left[L(h; \mu_c) + \epsilon \cdot L(h; \mu_p^*) \right] \leq L(h_c; \mu_c) + \epsilon \cdot \ell_{\text{M}}(\max_{(\boldsymbol{x}, y) \in \mathcal{C}} |\boldsymbol{w}_c^\top \boldsymbol{x} + b_c|) + \epsilon C_R \cdot R(h_c). \tag{5.11}$$

Notably, the only difference between the upper bound in (5.10) (projected constraint size) and (5.11) (max-loss) is in the last term of $\ell_{\text{M}}(\text{Size}_{\boldsymbol{w}_c}(\mathcal{C}))$ or $\ell_{\text{M}}(\max_{(\boldsymbol{x}, y) \in \mathcal{C}} |\boldsymbol{w}_c^\top \boldsymbol{x} + b_c|)$. For these two terms, the max-loss term is upper-bounded by the term of the projected constraint size, and therefore, the provides a tighter upper bound on the risk of optimal poisoning attacks compared to the latter. However, the projected constraint size captures the two end points of the projected range of the candidate (optimal) poisoning distributions, while the max-loss only captures the higher end point of the projected range. Because of this, the projected constraint size may better reflect the (unknown) optimal poisoning distribution μ_p^* because μ_p^* does not necessarily contain the extreme poisoning point $(\boldsymbol{x}, y)$ that maximizes $\ell_{\text{M}}(|\boldsymbol{w}_c^\top \boldsymbol{x} + b_c|)$.

In Section 5.5.2, we minimize the tighter upper bound given in (5.11) (by replacing h_c with an optimizable h) to provide a non-trivial and tighter estimation of the upper bound of the risk of optimal poisoning attacks and demonstrate that these upper bounds also vary across benchmark datasets.

5.4.4 Proof of Theorem 5.4.10

The proof of Theorem 5.4.10 is inspired by the design of the Min-Max attack (Steinhardt et al., 2017) in the indiscriminate setting.

Proof of Theorem 5.4.10. Consider linear hypothesis class $\mathcal{H}$ and the poisoned distribution μ_p^* generated by the optimal poisoning adversary $\mathcal{A}_{\text{opt}}$ in Definition 5.3.1. Given clean distribution μ_c, poisoning ratio ϵ, and constraint set $\mathcal{C}$, the inherent vulnerability to indiscriminate poisoning is captured by the optimal attack performance $\text{Risk}(h_p^*; \mu_c)$, where h_p^* denotes the poisoned linear model induced by μ_p^*. For any $h \in \mathcal{H}$, we have

$$\text{Risk}(h_p^*; \mu_c) \leq L(h_p^*; \mu_c) \leq L(h_p^*; \mu_c) + \epsilon \cdot L(h_p^*; \mu_p^*) \leq L(h; \mu_c) + \epsilon \cdot L(h; \mu_p^*), \tag{5.12}$$

where the first inequality holds because the surrogate loss is defined to be not smaller than the 0-1 loss, the second inequality holds because the surrogate loss is always non-negative, and the third inequality holds because h_p^* minimizes the population loss with respect to both clean distribution μ_c and optimally generated poisoned distribution μ_p^*. Consider $h_c = \text{argmin}_{h \in \mathcal{H}} L(h; \mu_c)$ (with weight parameter $\boldsymbol{w}_c$ and bias parameter b_c), which is the linear model learned from the clean data. Therefore, plugging $h = h_c$ into the right-hand side of (5.12), we further obtain

$$\text{Risk}(h_p^*; \mu_c) \leq L(h_c; \mu_c) + \epsilon \cdot L(h_c; \mu_p^*) \leq L(h_c; \mu_c) + \epsilon \cdot \ell_{\text{M}}(\text{Size}_{\boldsymbol{w_c}}(\mathcal{C})) + \epsilon C_R \cdot R(h_c), \tag{5.13}$$

where the last inequality holds because for any poisoned data point $(\boldsymbol{x}, y) \sim \mu_p^*$, the surrogate loss at $(\boldsymbol{x}, y)$ with respect to h_c is $\ell_{\text{M}}\big(y \cdot (\boldsymbol{w}_c^\top \boldsymbol{x} + b_c)\big) + \epsilon C_R \cdot R(h_c)$, and $y \cdot (\boldsymbol{w}_c^\top \boldsymbol{x} + b_c) \leq \max_{(\boldsymbol{x}, y) \in \mathcal{C}} |\boldsymbol{w}_c^\top \boldsymbol{x} + b_c|$. Under the condition that $\min_{(\boldsymbol{x}, y) \in \mathcal{C}} \boldsymbol{w_c}^\top \boldsymbol{x} \leq -b_c \leq \max_{(\boldsymbol{x}, y) \in \mathcal{C}} \boldsymbol{w_c}^\top \boldsymbol{x}$ which means the decision boundary of h_c falls into the constraint set $\mathcal{C}$ when projected on to the direction of $\boldsymbol{w}_c$, we further have $\max_{(\boldsymbol{x}, y) \in \mathcal{C}} |\boldsymbol{w}_c^\top \boldsymbol{x} + b_c| \leq \text{Size}_{\boldsymbol{w_c}}(\mathcal{C})$, which implies the validity of (5.13). We remark that the condition $\min_{(\boldsymbol{x}, y) \in \mathcal{C}} \boldsymbol{w_c}^\top \boldsymbol{x} \leq -b_c \leq \max_{(\boldsymbol{x}, y) \in \mathcal{C}} \boldsymbol{w_c}^\top \boldsymbol{x}$ typically holds for margin-based loss in practice, since the support of the clean training data belongs to the constraint set for poisoning inputs (for either undefended victim models or models that employ some data sanitization defense). Therefore, we leave this condition out in the statement of Theorem 5.4.10 for simplicity. □

5.5 Experiments

We first introduce the experiments conducted on the synthetic dataset (Section 5.5.1) to study the impact of the identified factors on the performance of optimal poisoning attacks. Then, we show how the factors identified in theoretical settings are correlated to the empirical vulnerabilities observed for the benchmark datasets (Section 5.5.2).

5.5.1 Experiments on Synthetic Datasets

According to Remark 5.4.4, there are two important factors to be considered: (1) the ratio between class separability and within-class variance $|\gamma_1 - \gamma_2|/\sigma$; (2) the size of the constraint set u. We conduct synthetic experiments to study the impact of these factors on the performance of (optimal) data poisoning attacks.

For our experiments, we generate 10,000 training and 10,000 testing data points according to the Gaussian mixture model (5.6) with negative center $\gamma_1 = -10$ and positive center $\gamma_2 = 0$. Throughout our experiments, γ_1 and γ_2 are kept fixed, whereas we vary the variance parameter σ and the value of u. The default value of u is set as 20 if not specified. Evaluations of empirical poisoning attacks require training linear SVM models, where we choose $C_R = 0.01$. The poisoning ratio is still set as 3%, consistent with evaluations on the benchmark datasets in Section 5.2.

Impact of separability/within-class variance ratio ($|\gamma_1 - \gamma_2|/\sigma$). First, we show how the optimal attack performance changes as we increase the value of $|\gamma_1 - \gamma_2|/\sigma$. Here, we choose to report the actual risk achieved by the OPT attack based on Theorem 5.4.3, not the increased risk (or test error) that is used for the benchmark datasets in Section 5.2. This is because, the OPT attack is only characterized with respect to the actual risk after poisoning in Theorem 5.4.3, not the increased risk. Furthermore, for convenience in analysis, the OPT attack is only characterized for the restricted setting of $w = \{-1, 1\}$ instead of taking a real number in practice. We find that, in this restrictive setting, it is hard to find a dataset from the 1-D Gaussian mixture that has low base error and high poisoned error (i.e., high increased error). If we relax w to take real numbers, then finding such a dataset becomes easier and will be discussed in detail when measuring the impact of u below.

When computing the risk of the OPT attack, we can only obtain approximations of the inverse function g^{-1} using numerical methods, which may induce a small approximation error for evaluating the optimal attack performance. For the finite-sample setting, we also report the empirical test error of the poisoned models induced by the empirical OPT attack and the best current poisoning attack discussed in Section 5.2, where

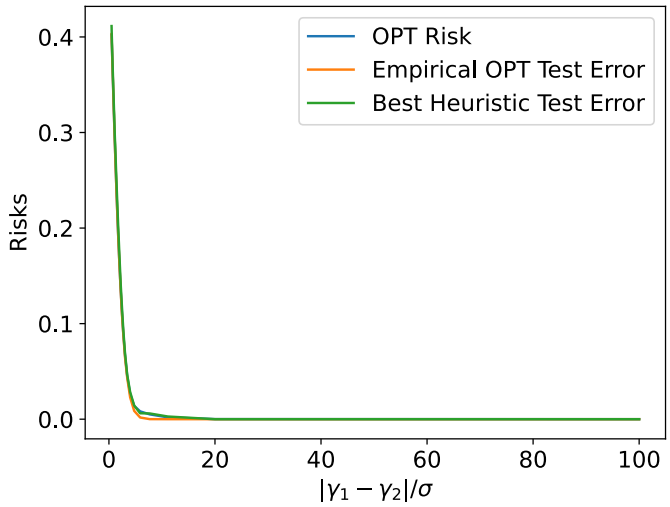


Figure 5.3: Measuring the performance of optimal poisoning attacks with different values of separability ratio $|\gamma_1 - \gamma_2|/\sigma$. *Best Heuristic* denotes the best-performing attack in the literature.

the latter is termed as *Best Heuristic* for simplicity. Since the poisoned models induced by these empirical attacks do not restrict $w \in \{-1, 1\}$, we normalize w to make the empirical results comparable with our theoretical results.

Figure 5.3 summarizes the attack performance as we vary the ratio $|\gamma_1 - \gamma_2|/\sigma$. As the ratio between class separability and within-class variance increases, the risk of the OPT attack and empirical test errors of the empirical OPT and best heuristic attacks gradually decrease. This is consistent with our theoretical results discussed in Section 5.4.1, and interestingly, for the simple 1-D Gaussian mixtures, the optimal poisoning attacks and the best heuristics have negligible gaps.

Impact of size of the constraint set (u). Our theoretical results assume the setting where $w \in \{-1, 1\}$. However, this restriction makes the impact of the constraint set size u less significant, as it is only helpful in judging whether flipping the sign of w (Condition 5.8 in Theorem 5.4.3) is feasible and becomes irrelevant to the maximum risk after poisoning when flipping is infeasible. In contrast, if w is not this restricted, the impact of u will be smoother and more significant. In particular, when w takes real numbers, it will be impacted more by larger u as the poisoning points generated can be very extreme and force the poisoned model to have reduced w (compared to clean model w_c) in the norm so as to minimize the large loss introduced by the extreme poisoning points. Figure 5.4 plots the relationship between u and w of the poisoned model and supports the statement above. In terms of the impact of u on the poisoning effectiveness, when the norm of w becomes smaller, the original clean data that are well-separated become less separable so that slight

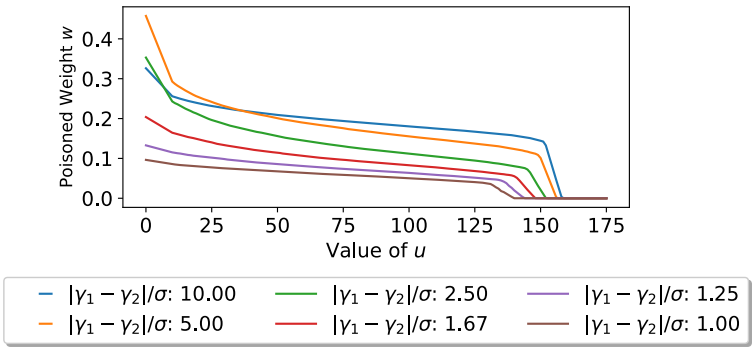


Figure 5.4: Impact of the (projected) constraint size u on the value of poisoned weight vector w after poisoning in 1-D Gaussian distributions.

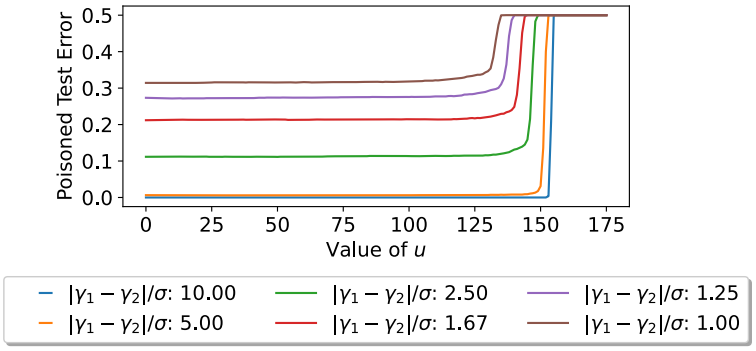


Figure 5.5: Impact of the (projected) constraint size u on poisoning effectiveness. $u = 0$ means the test error without poisoning.

movement in the decision boundary can cause significantly increased test errors. This makes the existence of datasets that have large risk gap before and after poisoning more likely.

Since relaxing w to take real numbers violates the assumption of our theory, the maximum risk after poisoning can no longer be characterized based on Theorem 5.4.3. Instead, we use the poisoning attack inspired by our theory to get an empirical lower bound on the maximum risk. Since $\gamma_1 + \gamma_2 < 0$, Theorem 5.4.3 suggests that optimal poisoning should place all poisoning points on u with label -1 when $w \in \{1, -1\}$. We simply use this approach even when the w can now take arbitrary values. We vary the value of u gradually and record the test error of the induced hypothesis, We repeat this procedure for different dataset configurations (i.e., fixing γ_1, γ_2, and varying σ). We still use the poisoned error (not increased error) to show poisoning effectiveness for consistency in the evaluation of the synthetic dataset, but additionally include the base test error without poisoning to provide a sense of the error increase after poisoning.

	Metric	F. Enron	MNIST 6–9	Dogfish	MNIST 1–7	Enron	MNIST 4–9	Adult
SVM	Upper Bound	64.5	21.4	34.9	20.1	64.7	40.8	65.1
	Lower Bound	33.3	3.0	8.7	3.6	34.8	10.9	24.1
	Base Error	0.2	0.3	0.8	1.2	2.9	4.3	21.5
LR	Upper Bound	64.8	35.1	46.9	35.9	64.9	54.8	64.1
	Lower Bound	35.6	2.9	10.9	4.0	37.1	10.7	22.6
	Base Error	0.3	0.6	1.7	2.2	2.5	5.1	20.1

Table 5.2: Non-trivial upper bounds on the limits of poisoning attacks across benchmark datasets. The *Upper Bound* on the limits of poisoning attacks is obtained by minimizing the upper bound Equation (5.11). The *Lower Bound* on the limit of poisoning attacks is obtained by reporting the highest poisoned error from the best empirical attacks. The datasets are sorted based on the lowest and highest empirical base error.

The results are summarized in Figure 5.5, where $u = 0$ denotes the setting without poisoning and hence the test error in this setting is the base test error. There are two key observations: (1) once w is no longer constrained, the test errors of all the considered datasets (even the datasets with very low base test errors) gradually increase (to a maximum of 50%) after poisoning when we gradually increase the value of u, and (2) for a given u, datasets with smaller ratio $|\gamma_1 - \gamma_2|/\sigma$ can be more vulnerable to poisoning and have larger increased test error. Although not backed by our theory, it makes sense as a smaller separability ratio[2] also means more points might be closer to the boundary (smaller margin) and hence small changes in the decision boundary can have significantly increased test errors.

5.5.2 Experiments on Benchmark Datasets

In this section, we first show the drastic differences in the upper bounds on the limits of any poisoning attacks, which complements the drastic variation observed for the lower bounds on the limits in Section 5.2. Then we show the correlation between the identified factors and the empirically observed vulnerabilities across benchmark datasets, using the clean model weight as the projection vector.

Minimization of the Upper Bound

We first provide details on the upper bound minimization and then present the computed non-trivial upper bounds. Finally, we provide reasons for the current large gap between the upper and lower bounds on the limits of poisoning attacks.

Variation of the non-trivial upper bounds. The upper-bound minimization of (5.11) (by replacing h_c with general h and then optimizing it) corresponds to a min-max optimization problem. We solve it using the

[2]Note that a smaller separability ratio does not necessarily indicate higher base test error. This will be more obvious for high-dimensional benchmark datasets such as Filtered Enron, which has the lowest base error and highest increased error. We will show in Table 5.3 that this high increased error is partially because Filtered Enron has a low separability ratio.

online gradient descent algorithm that alternatively updates the poisoning points and model weight, and the risk is empirically estimated on the training set. Note that, this approach is similar to the one used by Koh et al. for the i-Min-Max attack (Koh et al., 2022) without the target model $\hat{h}_{tar}$ (details in Algorithm 4). The subtle difference is, their method focuses on designing the practical attack and hence, stops the optimization after $\epsilon|\mathcal{S}_c|$ (or slightly more) poisoning points are generated in an online manner. These fewer iterations of online optimization can generate quite loose upper bounds. In this experiment, we focus on finding the tightest possible estimate using the online gradient descent and hence, choose to run the optimization for 30,000 iterations (which is much higher than the clean training set size for all the considered benchmark datasets in this chapter) with a learning rate of 0.03 for the weight vector update and report the lowest upper bound obtained in the process. It is important to note that, the empirically computed upper bound is estimated on the training data and hence only provides an approximate upper bound to the maximal test error after poisoning.

The minimized upper bounds at $\epsilon = 0.03$ poisoning are summarized in Table 5.2. From the table, we can easily see that the (approximate) upper bound on the limits of any poisoning attacks still vary significantly across datasets, complementing the drastic variations in terms of the lower bound on the limits observed in Section 5.2. In addition, the computed upper bounds are also mostly highly correlated with the observed empirical lower bounds, especially for the linear SVM, and the empirically less vulnerable datasets also have relatively smaller upper bounds. Here, we are presenting the correlation between the computed upper and lower bounds of the final test error (not the error increase that we use to empirically measure the dataset vulnerability in Section 5.2) after poisoning because the upper bound in (5.11) is for the maximum risk after optimal poisoning, not the maximum increased risk. While for a dataset with low enough base error, the final test error and the increased error are mostly similar, it is not the case for datasets with high base errors (e.g., the Adult dataset) because the provided upper bound on the final test error may not well capture the increased error. Nevertheless, we show that even the non-trivial upper bound on the limits of poisoning attacks can vary drastically across datasets, and in some cases (e.g., MNIST 6–9, MNIST 1–7), the (potentially loose) upper bound may still be limited in effectiveness to some degree ($\approx 20\%$ test error at 3% of poisoning).

Reasons for the large gap between the lower and upper bounds. Another side observation from Table 5.2 is, currently, the gaps between the lower and upper bounds for all the examined datasets are large, and more in-depth analysis is needed to shrink the gap. We speculate the large gap exists because of two reasons. The first is, we use the surrogate loss to upper bound the 0-1 loss and maximize the surrogate loss in

		Robust			Moderately Vulnerable		Highly Vulnerable		
	Metric	MNIST 6–9	MNIST 1–7	Adult	Dogfish	MNIST 4–9	F. Enron	Enron	IMDB
SVM	Error Increase	2.7	2.4	3.2	7.9	6.6	33.1	31.9	19.1†
	Base Error	0.3	1.2	21.5	0.8	4.3	0.2	2.9	11.9
	Sep/SD	6.92	6.25	9.65	5.14	4.44	1.18	1.18	2.57
	Sep/Size	0.24	0.23	0.33	0.05	0.14	0.01	0.01	0.002
LR	Error Increase	2.3	1.8	2.5	6.8	5.8	33.0	33.1	-
	Base Error	0.6	2.2	20.1	1.7	5.1	0.3	2.5	-
	Sep/SD	6.28	6.13	4.62	5.03	4.31	1.11	1.10	2.52
	Sep/Size	0.27	0.27	0.27	0.09	0.16	0.01	0.01	0.003

Table 5.3: Explaining disparate poisoning vulnerability under linear models by computing the metrics on the correctly classified clean test points. The top row for each model gives the increase in error rate due to the poisoning, over the base error rate in the second row. The error increase of IMDB (marked with †) is directly quoted from Koh et al. (2022) as running the existing poisoning attacks on IMDB is extremely slow. LR results are missing as they are not contained in the original paper. Sep/SD denotes the ratio between the projected separability and the projected variance onto the clean model weight. Sep/Size denotes the ratio between the projected separability and the projected constraint size onto the clean model weight.

Models	Metrics	Robust			Moderately Vulnerable		Highly Vulnerable		
		MNIST 6-9	MNIST 1–7	Adult	Dogfish	MNIST 4–9	F. Enron	Enron	IMDB
SVM	Error Increase	2.7	2.4	3.2	7.9	6.6	33.1	31.9	19.1†
	Base Error	0.3	1.2	21.5	0.8	4.3	0.2	2.9	11.9
	Sep/SD	6.70	5.58	1.45	4.94	3.71	1.18	1.15	1.95
	Sep/Size	0.23	0.23	0.18	0.05	0.13	0.01	0.01	0.001
LR	Error Increase	2.3	1.8	2.5	6.8	5.8	33.0	33.1	-
	Base Error	0.6	2.2	20.1	1.7	5.1	0.3	2.5	-
	Sep/SD	5.97	5.17	1.64	4.67	3.51	1.06	1.01	1.88
	Sep/Size	0.26	0.26	0.16	0.08	0.15	0.01	0.01	0.002

Table 5.4: Explaining the different vulnerabilities of benchmark datasets under linear models by computing metrics on the whole test data. The error increase of IMDB (marked with †) is directly quoted from Koh et al. (2022).

the poisoning setting, which unavoidably introduces a (potentially large) gap and this gap might be hard, if not impossible, to minimize. We may circumvent this issue if we no longer rely on the min-max formulation, as in (5.11), to upper bound the performance of any indiscriminate poisoning attacks. Second, the upper bound in (5.11) involves finding the (extreme) maximum loss for a given hypothesis h in the optimization process of h while the optimal poisoning distribution μ_p^* may not be that extreme to always maximize the loss with respect to h_p^*. A possible approach to handle this problem is to first identify the properties of μ_p^* (e.g., how extreme it can be). Then, when optimizing the h in (5.11), instead of finding the poisoning points that maximize the loss on h, find less extreme points that still provide a valid upper bound by leveraging the properties of μ_p^*.

Explaining the Variation in Empirical Attack Effectiveness

Recall from Theorem 5.3.6 and Remark 5.3.7 that the finite-sample optimal poisoning attack is a consistent estimator of the distributional one for linear learners. In this section, we demonstrate the theoretical insights

gained from Section 5.4, despite proven only for the distributional optimal attacks, still appear to largely explain the empirical performance of best attacks across benchmark datasets.

Given a clean training set $\mathcal{S}_c$, we empirically estimate the three distributional metrics defined in Section 5.4.3 on the clean test data with respect to the weight $\boldsymbol{w}_c$ of the clean model $h_{\boldsymbol{w}_c}$. Since $\|\boldsymbol{w}_c\|_2$ may vary across different datasets while the predictions of linear models (i.e., the classification error) are invariant to the scaling of $\|\boldsymbol{w}_c\|_2$, we use ratios to make their metrics comparable: $\text{Sep}_{\boldsymbol{w}_c}(\mu_c)/\text{SD}_{\boldsymbol{w}_c}(\mu_c)$ (denoted as Sep/SD in Table 5.3) and $\text{Sep}_{\boldsymbol{w}_c}(\mu_c)/\text{Size}_{\boldsymbol{w}_c}(\mathcal{C})$ (Sep/Size). According to our theoretical results, we expect datasets that are less vulnerable to poisoning to have higher values for both metrics. For the IMDB dataset, we directly quote the poisoned error on linear SVM from Koh et al. (2022) due to the extremely long computational time to run the current attacks and only (actually) compute the related metrics identified in this chapter.

Table 5.3 summarizes the results, showing that the Sep/SD and Sep/Size metrics can largely explain why datasets such as MNIST 1–7 and MNIST 6–9 are harder to poison than others. These datasets are more separable and impacted less by the poisoning points. In contrast, datasets such as Enron, Filtered Enron and IMDB are highly vulnerable because they are the least separable and also impacted the most by poisoning points. The empirical metrics are indeed highly correlated to the error increase (and also the final poisoned error) when the base error is small, which is the case for all tested benchmark datasets except Adult. The results of Filtered Enron (low base error, high increased error) and Adult (high base error, low increased error) demonstrate the poisoning vulnerability cannot be trivially inferred from the initial base error. When the base error becomes high as it is for Adult, the empirical metrics are highly correlated to the final poisoned error, but not the error increase. For the error increase, computing the metrics on clean test points that are correctly classified by $h_{\boldsymbol{w}_c}$ is more informative. Therefore, we report metrics based on correctly-classified test points in Table 5.3 and provide results of the whole test data in Table 5.4. For datasets except Adult, both ways of computing the metrics produce similar results. The Adult dataset is very interesting in that it is robust to poisoning (i.e., small error increase) despite having a very high base error.

5.6 Implication on Future Defenses

We show how our results on understanding the limits of indiscriminate poisoning attacks suggest future defenses by explaining why candidate defenses work and motivating future defenses to improve separability and reduce projected constraint size. We present two ideas: 1) current data sanitization defenses reduce the projected constraint size and future defenses should also focus on limiting the projected constraint size; 2)

Dataset	Error Increase		Base Error		Sep/SD		Sep/Size	
	w/o	w/	w/o	w/	w/o	w/	w/o	w/
MNIST 1–7 (10%)	7.7	1.0	1.2	2.4	6.25	6.25	0.23	0.43
Enron (3%)	31.9	25.6	2.9	3.2	1.18	1.18	0.01	0.11

Table 5.5: Understanding the impact of data sanitization defenses on poisoning attacks. *w/o* and *w/* denote *without defense* and *with defense* respectively. MNIST 1–7 is evaluated at a 10% poisoning ratio due to its strong robustness at $\epsilon = 3\%$ and Enron is still evaluated at $\epsilon = 3\%$ because it is highly vulnerable.

using better feature representations might improve separability (and potentially reduce projected constraint size) to resist poisoning with and without defenses.

Explaining the impact of data sanitization defenses. Common data sanitization defenses work by identifying and filtering out bad points. We speculate that such defenses work because they effectively limit the projected constraint size of $\mathcal{C}$. To test this, we picked the combination of Sphere and Slab defenses considered in prior works (Koh et al., 2022; Steinhardt et al., 2017) to protect the vulnerable Enron dataset at 3% poisoning ratio and the already robust MNIST 1–7 dataset at a higher 10% poisoning ratio. We considered a significantly higher poisoning ratio for MNIST 1–7 because at the original 3% poisoning rate, as shown in Section 5.2, the dataset can well resist known attacks and hence there is no point in protecting the dataset with sanitization defenses. This attack setting is just for an illustration purpose, and attackers in practice may not be able to manipulate such a large number of poisoning points.

The results are summarized in Table 5.5, and we can see that existing data sanitization defenses improve the robustness to poisoning by majorly limiting $\text{Size}_{\boldsymbol{w}_c}(\mathcal{C})$. Following the main result in the paper, we still compute the metrics based on the correctly classified samples in the clean test set, so as to better depict the relationship between the increased errors and the computed metrics. For Enron, with defense, the test error increases from 3.2% to 28.8% while without defense, the error can be increased from 2.9% to 34.8%. Although limited in effectiveness, the defense still mitigates the poisoning to some degree, mostly by shrinking the projected constraint size $\text{Size}_{\boldsymbol{w}_c}(\mathcal{C})$. This leads to a higher value for the Sep/Size metric: 0.11 with defense compared to 0.01 without defense. For MNIST 1–7, employing the data sanitization defense makes the dataset even more robust (preserving robustness even at the high 10% poisoning rate), which is consistent with the findings in prior work (Steinhardt et al., 2017), due to the reduced impact from poisoning (i.e., higher Sep/Size). To conclude, *future defenses against poisoning attacks should also focus on developing methods to effectively reduce the projected constraint size.*

Better feature representation to resist poisoning. We consider a transfer learning scenario where the

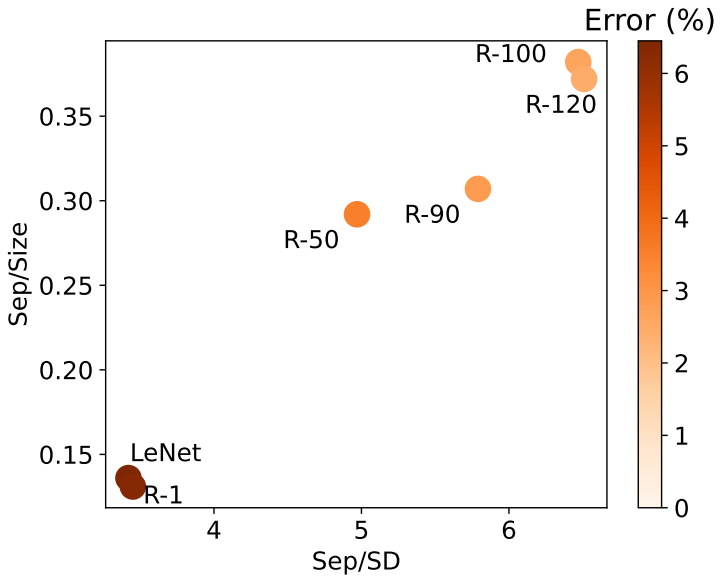


Figure 5.6: Impact of features on poisoning effectiveness. *R-X* denotes the feature representation of the dataset obtained from the feature extractor of the ResNet18 model trained for X epochs. *LeNet* denotes the feature representation of the dataset obtained from the feature extractor of the fully trained simple CNN model.

victim trains a linear model on a clean pre-trained model. As a preliminary experiment, we train LeNet and ResNet18 models on the CIFAR10 (Krizhevsky et al., 2009) dataset till convergence, but record the intermediate models of ResNet18 to produce models with different feature extractors (R-X denotes ResNet18 trained for X epochs). We then use the feature extraction layers of these models (including the fully trained LeNet) as the pre-trained models and obtain features of CIFAR10 images with labels "Truck" and "Ship", and train (binary) linear SVM models on them.

We evaluate the robustness of this dataset against poisoning attacks and set $\mathcal{C}$ as the dimension-wise box constraints, whose values are the minimum and maximum values of the clean data points for each dimension when fed to the feature extractors. This approach corresponds to the practical scenario where the victim has access to a small number of clean samples so that they can deploy a simple defense of filtering out inputs that do not fall into a dimension-wise box constraint that is computed from the available clean samples of the victim. Figure 5.6 shows that as the feature extractor becomes better (either using deep architecture or training it for more epochs), both the Sep/SD and Sep/Size metrics increase, leading to reduced error increase even without any additional defenses. This indicates that better feature representations (trained on the clean data) might help in resisting poisoning attacks even when there are no defenses deployed.

We believe the Sphere and Slab defenses mentioned above may also benefit from better representations because

Datasets	Base Error (%)	Increased Error (%)	Sep/SD	Sep/Size
MNIST	0.8%	1.1%	4.58	0.10
CIFAR-10	31.0%	4.3%	0.24	0.01

Table 5.6: Results on simple CNN models for MNIST and CIFAR-10 datasets using 3% poisoning ratio. The "Poisoned Error" of both datasets are directly quoted from Lu et al. (2022) as these are the only available state-of-the-art results in the literature. Under similar neural network structures, the vulnerability factors identified under linear models can also generalize to neural networks. The binary pair with the lowest Sep/SD score also has the lowest Sep/Size score.

Datasets	Base Error (%)	Increased Error (%)	Sep/SD	Sep/Size
MNIST	0.8%	9.6%	4.58	0.10
CIFAR-10	4.8%	13.7%	6.36	0.24

Table 5.7: Results on simple CNN model for MNIST and ResNet18 model for CIFAR-10 datasets using a 3% poisoning ratio. The "Poisoned Error" of both datasets are directly quoted from Lu et al. (2023), which represents the most recent progress on attacks on neural networks in the field. Under different neural network structures, the vulnerability factors identified under linear models cannot explain observations on neural networks. The binary pair with the lowest Sep/SD score also has the lowest Sep/Size score.

for the distributions with high Sep/SD and Sep/Size, the class-centroids will not be skewed much in the presence of poisoning points, and hence increase the resistance of these defenses to poisoning attacks.

Implication for defenses against subpopulation attacks. We believe learning better representations (e.g., compact representation with higher separability, or higher Sep/SD as in our analysis) might also help in improving the resistance to subpopulation poisoning attacks. This is because feature representations with higher separability might increase the average subpopulation difficulty, and densely clustered points in compact feature representation might indicate that misclassifying any subpopulation will unavoidably face strong resistance from the rest of the population and make the overall attack harder to succeed with a limited number of poisoning points.

5.7 Comparison with Related Work

In this section, we first provide a more detailed comparison to the related works on understanding the inherent vulnerabilities in targeted poisoning settings. All the aforementioned indiscriminate poisoning attacks on linear learners that inject poisoning points (Biggio et al., 2011; Mei and Zhu, 2015b;a; Koh and Liang, 2017; Steinhardt et al., 2017; Koh et al., 2022) focus on developing different indiscriminate poisoning algorithms, but did not explain why certain datasets are seemingly harder to poison than others. Our work leverages these attacks to empirically estimate the inherent vulnerabilities of benchmark datasets to poisoning but focuses on providing explanations for the disparate poisoning vulnerability across the datasets. For indiscriminate

poisoning attacks beyond linear learners (Lu et al., 2022; 2023), our work can also leverage these attacks to empirically estimate the inherent vulnerabilities of different datasets under neural networks and provide possible explanations, and some preliminary results on this are discussed below. Notably for Lu et al. (2023), it also characterizes the limits of model-targeted poisoning attacks by proving a lower bound on the number of poisoning points needed to induce a particular target model and can be related to our work in understanding the limits of indiscriminate attacks. However, the hardness in inducing a particular target model may not fully reflect the fundamental limits in maximizing the risk with ϵ fraction of poisoning points. Furthermore, their results on the vulnerability variation across datasets are implicitly reflected through particular target models while we can compute the related distributional properties under the given learner explicitly. Besides injecting poisoning points, some other works consider a different poisoning setup in the indiscriminate setting from ours by modifying up to the whole training data, also known as unlearnable examples (Huang et al., 2021; Yu et al., 2022; Fowl et al., 2021).

We then provide the preliminary results in extending our insights beyond linear models. Note that extending the insights from binary classification in linear models to multi-class classification in neural networks is non-trivial and a detailed description of this extension can be found in the online version of this chapter (Suya et al., 2023). At the high level, to extend binary classification to k-class classification ($k > 2$), we enumerate over all $k(k-1)/2$ binary pairs and use the pair(s) with the lowest Sep/SD and Sep/Size scores (report both pairs when there are two unique pairs for the lowest Sep/SD and Sep/Size scores). For a neural network, we view it as consisting of a fixed feature extractor and a linear classification layer. When computing the related two vulnerability metrics, we first feed the dataset through the feature extractor and obtain a "new" dataset in the transformed feature space, and then compute the metrics for the "new" dataset empirically. Table 5.6 shows that the distributional properties obtained for the linear models can also generalize to neural networks, given that the underlying learners are similar (simple CNN model in this experiment). However, when the underlying learners (e.g., neural network structures) are drastically different, then the factors cannot well-reflect the different effectiveness of the state-of-the-art poisoning attacks, as shown in Table 5.7. More detailed investigations are needed for this interesting observation.

We believe our approach to analyzing indiscriminate attacks might also be extended to *targeted* attacks (Shafahi et al., 2018; Zhu et al., 2019; Koh and Liang, 2017; Huang et al., 2020; Geiping et al., 2021) where the existing attacks (potentially also including our proposed MTP attack) will still be used as tools to empirically estimate the inherent vulnerability of a given test sample. As for the factors that contribute to the (potentially) different susceptibilities of different test samples, we believe the learning task properties (e.g., properties of the data distribution under the given learner) identified in this chapter may still be highly correlated, but the

Scale Factor c	Error Increase (%)	Sep/Size
1.0	2.2	0.27
2.0	3.1	0.15
3.0	4.4	0.10

Table 5.8: Impact of scale factor c on poisoning effectiveness for $\mathcal{C}$ in the form of dimension-wise box-constraint as $[0, c]$. The base error is 1.2%. The base error and the Sep/SD score will be the same for all settings because the support set of the clean distribution is a strict subset of $\mathcal{C}$, but Sep/Size will change as $\mathcal{C}$ changes.

additional factor(s) that describe the relative location of the individual test sample to the rest of the data points with respect to the clean decision boundary will also play an important role. We leave the exploration of the impact of individual test sample properties on the effectiveness of the optimal targeted attacks as future work.

In the targeted poisoning scenario, a related recent work that studies the inherent vulnerabilities of datasets to targeted data poisoning attacks proposed the Lethal Dose Conjecture (LDC) (Wang et al., 2022): given a dataset of size N, the tolerable amount of poisoning points from any targeted poisoning attack generated through insertion, deletion or modifications is $\Theta(N/n)$, where n is the sample complexity of the most data-efficient learner trained on the clean data to correctly predict a known test sample. Compared to our work, LDC is more general and applies to any dataset, any learning algorithm, and even different poisoning setups (e.g., deletion, insertion). In contrast, our work focuses on injection-only indiscriminate attacks for linear models. However, the general setting for LDC can result in overly pessimistic estimates on the power of injection-only indiscriminate poisoning attacks. Specifically, extending the results in targeted settings straightforwardly to indiscriminate (or subpopulation) settings will overestimate the power of these poisoning attacks—the direct adaptation assumes individual test samples are independently impacted by the poisoning points while in indiscriminate (or subpopulation) settings, the test samples are impacted by the same set of poisoning points, which is a much weaker attack setting compared to the former. In addition, the injection-only attack can also be much weaker than attacks that can delete existing points and inject new points. Note that the problem of overestimating the power of indiscriminate or subpopulation poisoning attacks also exists when adapting the optimal learning results in targeted poisoning scenario (Hanneke et al., 2022) to optimal learning in indiscriminate or subpopulation settings. These also highlight the importance of performing fine-grained analysis on the threat from poisoning attacks, as attacks in some restricted settings may not be that detrimental.

In addition, the key factor of the sample complexity n in LDC is usually unknown and difficult to determine. Our work complements LDC by making an initial step towards finding factors (projected separability and

projected variance in our case) that can be related to n under a particular attack scenario to better understand the power of indiscriminate data poisoning attacks. The projected constraint size identified in this paper can also be independent of n when the support of the clean distribution is a strict subset of the constraint set $\mathcal{C}$. In particular, in this case, if we further enlarge $\mathcal{C}$, it won't impact the clean distribution and therefore, the outcomes of learners trained on clean samples from the distribution will not change (including the most data-efficient learner) and hence n will remain the same for different permissible choices of $\mathcal{C}$, indicating that the vulnerability of the same dataset remains the same even when $\mathcal{C}$ changes drastically without impacting the clean distribution. However, changes in $\mathcal{C}$ (and subsequently, changes in the projected constraint size) will directly impact the attack effectiveness, as a larger $\mathcal{C}$ is likely to admit stronger poisoning attacks. To illustrate how much the attack power can change as $\mathcal{C}$ changes but without impacting the clean distribution, we conduct experiments on MNIST 1–7 and show that scaling up the original dimension-wise box-constraint from $[0, 1]$ to $[0, c]$ (where $c > 1$ is the scale factor) can significantly boost attack effectiveness. Table 5.8 summarizes the results, and we can observe that, as the scale factor c increases (enlarged $\mathcal{C}$, increased projected constraint size, and reduced Sep/Size), the attack effectiveness also increases significantly. Note that this experiment is an existence proof and MNIST 1–7 is used as a hypothetical example. In practice, for normalized images, the box constraint cannot be scaled beyond [0,1] as it will result in invalid images.

5.8 Summary

Motivated by the empirical observation that different datasets show disparate vulnerabilities to state-of-the-art poisoning attacks for linear learners, we rigorously characterized the optimal poisoning attacks for Gaussian distributions. The insights from the theoretical analysis can be used to explain the vulnerabilities of benchmark datasets. We made an initial but important step towards understanding the learning task properties that correlate with the inherent vulnerability to poisoning attacks. Our results also provide suggestions for building more robust systems.

Limitation and future work. Our work has several major limitations, which point out interesting future research directions. First, we only characterize the optimal poisoning attacks for theoretical distributions under linear models, but the extension to general distributions in high-dimensions and non-linear models is an important future work. Second, even for the linear models, the identified metrics cannot quantify the actual increased errors from optimal poisoning attacks, which itself is an interesting future work and one possible approach might be to tighten the upper bound in Theorem 5.4.10 using better optimization methods. Third, the metrics identified in this paper are learner dependent, depending on the properties of the learning

algorithm, dataset, and domain constraints (mainly reflected through $\mathcal{C}$). In certain applications, one might be interested in understanding the impact of learner-agnostic dataset properties on poisoning effectiveness—a desired dataset has such properties that any reasonable learners trained on the dataset can be robust to poisoning attacks. One likely application scenario is, the released data from the owner will be used by many different learners in various applications and these applications can be prone to poisoning.

Chapter 6

Conclusion

This dissertation analyzes the limits of poisoning attacks in the subpopulation and indiscriminate settings. We first quantify a tighter lower bound on the effectiveness of the best possible attacks by proposing a model-targeted poisoning attack that can be conveniently applied for indiscriminate and subpopulation settings (and beyond), and achieves comparable or better performance in comparison to the current state-of-the-art, especially in the practically-motivated subpopulation settings. Through extensive experiments on different subpopulations using our proposed attack, we observe drastically different susceptibilities across subpopulations, which further motivate us to explore learning task properties that contribute to the drastic variations. We then show that both the overall distributional separability and the loss difference between the clean model and the target model that misclassifies the subpopulation are highly correlated with the different attack effectiveness.

Driven by the fact that indiscriminate attacks are special forms of subpopulation attacks that take the entire datasets as the "subpopulations", we further test whether different datasets also have disparate vulnerabilities similar to the observations in the general subpopulation settings, and find that best empirical indiscriminate attacks indeed have drastically varied performances across datasets. We then explore distributional properties under the given learner that contribute to the observed variations and show that the projected separability, projected standard deviation, and projected constraint size onto the clean decision boundary can largely explain this variation. Moreover, these factors are also related to the upper bound on the performance of the (unknown) optimal indiscriminate poisoning attacks, and minimizing the upper bound can give a non-trivial estimate on the performance of the optimal attacks, which again drastically varies across datasets. Finally,

we discuss how our insights on the limits of poisoning attacks might help in designing better defenses against data poisoning attacks.

Although this dissertation, through a line of work, advances our understanding of why known poisoning attacks work in some settings and fail in others, our current understanding of the possible root causes of the different susceptibility is still obtained through quantitative correlation of the identified factors to the empirical attack performance. However, a more fundamental and perhaps practically more interesting question is to quantify the maximum risk (or test error) increase from any poisoning attacks on the defined subpopulation or the entire distribution by inspecting the related learning task properties. We made an initial step towards this goal in the indiscriminate setting by showing the connection between the identified learning task factors and the non-trivial upper bound on the performance of optimal poisoning attacks. However, a relatively large gap between the quantitative upper bound and the performance of the best empirical attacks remains. This raises the important question of whether the current upper bound estimation framework is limited or the existing empirical attacks are still far from optimal. We speculate that there is room for reducing the upper bound, as the current approach to compute the quantitative upper bound is limited. A tighter estimation might be obtained by finding the possible characteristics of the optimal poisoning distribution μ_p^*, which is again highly dependent on the learning task properties (Chapter 5). The more practically-motivated subpopulation setting is even more challenging, as it does not form a min-max formulation as in the indiscriminate setting to enable the computation of a non-trivial upper bound on the limits of poisoning attacks. We leave the exploration of tighter estimation on the optimal poisoning effectiveness for the indiscriminate and subpopulation settings (and beyond) as future work.

Finally, all of our explorations on the poisoning effectiveness eventually serve the purpose of designing more robust systems in an adversarial environment. This dissertation highlights the importance of feature representations in defending against data poisoning attacks. Future work should perform a more systematic exploration of better feature representations and their possible enhancements to data sanitization defenses. At an even higher level, our work also advocates for performing fine-grained security analysis for the underlying learning tasks so as to better understand the risks from poisoning in practice, as not all settings are lethally prone to poisoning attacks.

Appendix

1 Proofs of Technical Lemmas Used in Section 5.4.2

In this section, we provide the technical lemmas that are used to characterize the optimal poisoning attacks for the 1-D Gaussian distribution in Section 5.4.2.

1.1 Proof of Lemma 5.4.5

To prove Lemma 5.4.5, we need to make use of the following general lemma, which characterizes the population hinge loss and its derivative with respect to clean data distribution μ_c.

Lemma 1.1. *Let μ_c be data distribution generated according to* (5.6)*. For any $h_{w,b} \in \mathcal{H}$, the population hinge loss is:*

$$L(h_{w,b}; \mu_c) = p \int_{\frac{-b-w\cdot\gamma_1-1}{\sigma_1}}^{\infty} (b + w \cdot \gamma_1 + 1 + \sigma_1 z) \cdot \varphi(z) dz + (1-p) \int_{-\infty}^{\frac{-b-w\cdot\gamma_2+1}{\sigma_2}} (-b - w \cdot \gamma_2 + 1 - \sigma_2 z) \cdot \varphi(z) dz,$$

and its gradient with respect to b is:

$$\frac{\partial}{\partial b} L(h_{w,b}; \mu_c) = p \cdot \Phi\left(\frac{b + w \cdot \gamma_1 + 1}{\sigma_1}\right) - (1-p) \cdot \Phi\left(\frac{-b - w \cdot \gamma_2 + 1}{\sigma_2}\right),$$

where φ and Φ denote the PDF and CDF of the standard Gaussian distribution $\mathcal{N}(0,1)$, respectively.

Proof of Lemma 1.1. We use similar notations such as μ_1, μ_2, and φ as in Lemma 5.4.7. For any $h_{w,b} \in \mathcal{H}$ with $w = 1$, then according to the definition of population hinge loss, we have

$$
\begin{aligned}
&L(h_{w,b}; \mu_c) \\
&= \mathbb{E}_{(x,y)\sim\mu_c}\big[\max\{0, 1 - y(x+b)\}\big] \\
&= p\int_{-b-1}^{\infty}(1+b+z)\varphi(z;\gamma_1,\sigma_1)dz + (1-p)\int_{-\infty}^{-b+1}(1-b-z)\varphi(z;\gamma_2,\sigma_2)dz \\
&= p\int_{\frac{-b-1-\gamma_1}{\sigma_1}}^{\infty}(1+b+\gamma_1+\sigma_1 z)\varphi(z)dz + (1-p)\int_{-\infty}^{\frac{-b+1-\gamma_2}{\sigma_2}}(1-b-\gamma_2-\sigma_2 z)\varphi(z)dz \\
&= p(b+\gamma_1+1)\Phi\bigg(\frac{b+\gamma_1+1}{\sigma_1}\bigg) + p\sigma_1\frac{1}{\sqrt{2\pi}}\exp\bigg(-\frac{(b+\gamma_1+1)^2}{2\sigma_1^2}\bigg) \\
&\qquad + (1-p)(-b-\gamma_2+1)\Phi\bigg(\frac{-b-\gamma_2+1}{\sigma_2}\bigg) + (1-p)\sigma_2\frac{1}{\sqrt{2\pi}}\exp\bigg(-\frac{(-b-\gamma_2+1)^2}{2\sigma_2^2}\bigg).
\end{aligned}
$$

Taking the derivative with respect to parameter b and using simple algebra, we have

$$
\frac{\partial}{\partial b}L(h_{w,b};\mu_c) = p\cdot\Phi\bigg(\frac{b+\gamma_1+1}{\sigma_1}\bigg) - (1-p)\cdot\Phi\bigg(\frac{-b-\gamma_2+1}{\sigma_2}\bigg).
$$

Similarly, for any $h_{w,b} \in \mathcal{H}$ with $w = -1$, we have

$$
\begin{aligned}
&L(h_{w,b}; \mu_c) \\
&= \mathbb{E}_{(x,y)\sim\mu_c}\big[\max\{0, 1 - y(-x+b)\}\big] \\
&= p\cdot\int_{-\infty}^{b+1}(1+b-z)\varphi(z;\gamma_1,\sigma_1)dz + (1-p)\cdot\int_{b-1}^{\infty}(1-b+z)\varphi(z;\gamma_2,\sigma_2)dz \\
&= p\cdot\int_{-\infty}^{\frac{b+1-\gamma_1}{\sigma_1}}(1+b-\gamma_1-\sigma_1 z)\varphi(z)dz + (1-p)\cdot\int_{\frac{b-1-\gamma_2}{\sigma_2}}^{\infty}(1-b+\gamma_2+\sigma_2 z)\varphi(z)dz \\
&= p(b-\gamma_1+1)\Phi\bigg(\frac{b-\gamma_1+1}{\sigma_1}\bigg) + p\sigma_1\frac{1}{\sqrt{2\pi}}\exp\bigg(-\frac{(b-\gamma_1+1)^2}{2\sigma_1^2}\bigg) \\
&\qquad + (1-p)(-b+\gamma_2+1)\Phi\bigg(\frac{-b+\gamma_2+1}{\sigma_2}\bigg) + (1-p)\sigma_2\frac{1}{\sqrt{2\pi}}\exp\bigg(-\frac{(-b+\gamma_2+1)^2}{2\sigma_2^2}\bigg).
\end{aligned}
$$

Taking the derivative, we have

$$
\frac{\partial}{\partial b}L(h_{w,b};\mu_c) = p\cdot\Phi\bigg(\frac{b-\gamma_1+1}{\sigma_1}\bigg) - (1-p)\cdot\Phi\bigg(\frac{-b+\gamma_2+1}{\sigma_2}\bigg).
$$

Combining the two scenarios, we complete the proof. □

Next, let us summarize several key observations based on Lemma 1.1 (specifically for the setting considered in Lemma 5.4.5). For any $w \in \{-1, 1\}$, $\frac{\partial}{\partial b} L(h_{w,b}; \mu_c)$ is a monotonically increasing with b, which achieves minimum $-\frac{1}{2}$ when b goes to $-\infty$ and achieves maximum $\frac{1}{2}$ when b goes to ∞. If $w = +1$, then $L(h_{w,b}; \mu_c)$ is monotonically decreasing when $b \in (-\infty, -\frac{\gamma_1+\gamma_2}{2})$ and monotonically increasing when $b \in (-\frac{\gamma_1+\gamma_2}{2}, \infty)$, reaching the minimum at $b = b_c^*(1) := -\frac{\gamma_1+\gamma_2}{2}$. On the other hand, if $w = -1$, then $L(h_{w,b}; \mu_c)$ is monotonically decreasing when $b \in (-\infty, \frac{\gamma_1+\gamma_2}{2})$ and monotonically increasing when $b \in (\frac{\gamma_1+\gamma_2}{2}, \infty)$, reaching the minimum at $b = b_c^*(-1) := \frac{\gamma_1+\gamma_2}{2}$.

As for the clean loss minimizer conditioned on $w = 1$, we have

$$\begin{aligned} L(h_{1,b_c^*(1)}; \mu_c) &= \frac{1}{2} \int_{\frac{\gamma_2-\gamma_1-2}{2\sigma}}^{\infty} \left(\frac{\gamma_1 - \gamma_2}{2} + 1 + \sigma z \right) \cdot \varphi(z) dz \\ &\quad + \frac{1}{2} \int_{-\infty}^{\frac{\gamma_1-\gamma_2+2}{2\sigma}} \left(\frac{\gamma_1 - \gamma_2}{2} + 1 - \sigma z \right) \cdot \varphi(z) dz \\ &= \frac{(\gamma_1 - \gamma_2 + 2)}{2} \cdot \Phi\left(\frac{\gamma_1 - \gamma_2 + 2}{2\sigma} \right) + \frac{\sigma}{\sqrt{2\pi}} \cdot \exp\left(- \frac{(\gamma_1 - \gamma_2 + 2)^2}{8\sigma^2} \right), \end{aligned}$$

whereas as for the clean loss minimizer conditioned on $w = -1$, we have

$$\begin{aligned} L(h_{-1,b_c^*(-1)}; \mu_c) &= \frac{1}{2} \int_{\frac{\gamma_1-\gamma_2-2}{2\sigma}}^{\infty} \left(\frac{\gamma_2 - \gamma_1}{2} + 1 + \sigma z \right) \cdot \varphi(z) dz \\ &\quad + \frac{1}{2} \int_{-\infty}^{\frac{\gamma_2-\gamma_1+2}{2\sigma}} \left(\frac{\gamma_2 - \gamma_1}{2} + 1 - \sigma z \right) \cdot \varphi(z) dz \\ &= \frac{(\gamma_2 - \gamma_1 + 2)}{2} \cdot \Phi\left(\frac{\gamma_2 - \gamma_1 + 2}{2\sigma} \right) + \frac{\sigma}{\sqrt{2\pi}} \cdot \exp\left(- \frac{(\gamma_2 - \gamma_1 + 2)^2}{8\sigma^2} \right). \end{aligned}$$

Let $f(t) = t \cdot \Phi(\frac{t}{\sigma}) + \frac{\sigma}{\sqrt{2\pi}} \cdot \exp(-\frac{t^2}{2\sigma^2})$, we know $L(h_{1,b_c^*(1)}; \mu_c) = f(\frac{\gamma_1-\gamma_2+2}{2})$ and $L(h_{-1,b_c^*(-1)}; \mu_c) = f(\frac{\gamma_2-\gamma_1+2}{2})$. We can compute the derivative of $f(t)$: $f'(t) = \Phi(\frac{t}{\sigma}) \geq 0$, which suggests that $L(h_{1,b_c^*(1)}; \mu_c) \leq L(h_{-1,b_c^*(-1)}; \mu_c)$.

Now we are ready to prove Lemma 5.4.5.

Proof of Lemma 5.4.5. First, we prove the following claim: for any possible b_p, linear hypothesis h_{1,b_p} can always be achieved by minimizing the population hinge loss with respect to μ_c and $\mu_p = \nu_\alpha$ with some carefully-chosen $\alpha \in [0, 1]$ based on b_p.

For any $\mu_p \in \mathcal{Q}(u)$, according to the first-order optimality condition with respect to b_p, we have

$$\frac{\partial}{\partial b} L(h_{1,b_p}; \mu_c) = -\epsilon \cdot \frac{\partial}{\partial b} L(h_{1,b_p}; \mu_p) = -\epsilon \cdot \frac{\partial}{\partial b} \mathbb{E}_{(x,y)\sim\mu_p}\big[\ell(h_{1,b_p}; \mu_p)\big] \in [-\epsilon, \epsilon], \tag{1}$$

where the last inequality follows from $\frac{\partial}{\partial b}\ell(h_{w,b}; x, y) \in [-1, 1]$ for any (x, y). Let $\mathcal{B}_p$ be the set of any possible bias parameters b_p. According to (1), we have

$$\mathcal{B}_p = \left\{ b \in \mathbb{R} : \frac{\partial}{\partial b} L(h_{1,b}; \mu_c) \in [-\epsilon, \epsilon] \right\}.$$

Let $b_c^*(1) = \operatorname{argmin}_{b\in\mathbb{R}} L(h_{1,b}; \mu_c)$ be the clean loss minimizer conditioned on $w = 1$. According to Lemma 1.1 and the assumption $|\gamma_1 + \gamma_2| \leq 2u$, we know $b_c^*(1) = \frac{\gamma_1+\gamma_2}{2} \in [-u, u]$. For any $b_p \in \mathcal{B}_p$, we can always choose

$$\alpha = \frac{1}{2} + \frac{1}{2\epsilon} \cdot \frac{\partial}{\partial b} L(h_{1,b_p}; \mu_c) \in [0, 1], \tag{2}$$

such that

$$h_{1,b_p} = \operatorname*{argmin}_{b\in\mathbb{R}} [L(h_{1,b}; \mu_c) + \epsilon \cdot L(h_{1,b}; \nu_\alpha)],$$

where ν_α is defined according to (5.7). This follows from the first-order optimality condition for the convex function and the closed-form solution for the derivative of hinge loss with respect to ν_α:

$$\frac{\partial}{\partial b} L(h_{1,b_p}; \nu_\alpha) = \alpha \cdot \frac{\partial}{\partial b}\ell(h_{+1,b_p}; -u, +1) + (1 - \alpha) \cdot \frac{\partial}{\partial b}\ell(h_{+1,b_p}; u, -1) = 1 - 2\alpha.$$

Thus, we have proven the claim presented at the beginning of the proof of Lemma 5.4.5.

Next, we show that for any $b_p \in \mathcal{B}_p$, among all the possible choices of poisoned distribution μ_p that induces b_p, choosing $\mu_p = \nu_\alpha$ with α defined according to (2) is the optimal choice in terms of the maximization objective in (5.9). Let $\mu_p \in \mathcal{Q}(u)$ be any poisoned distribution that satisfies the following condition:

$$b_p = \operatorname*{argmin}_{b\in\mathbb{R}} [L(h_{1,b}; \mu_c) + \epsilon \cdot L(h_{1,b}; \mu_p)].$$

According to the aforementioned analysis, we know that by setting α according to (2), ν_α also yields b_p. Namely,

$$b_p = \operatorname*{argmin}_{b\in\mathbb{R}} [L(h_{1,b}; \mu_c) + \epsilon \cdot L(h_{1,b}; \nu_\alpha)].$$

Since the population losses with respect to μ_c are the same at the induced bias $b = b_p$, it remains to prove ν_α achieves a larger population loss with respect to the poisoned distribution than that of μ_p, i.e.,

$L(h_{1,b_p};\nu_\alpha) \geq L(h_{1,b_p};\mu_p)$.

Consider the following two probabilities with respect to b_p and μ_p:

$$p_1 = \mathbb{P}_{(x,y)\sim\mu_p}\left[\frac{\partial}{\partial b}\ell(h_{1,b_p};x,y) = -1\right], \quad p_2 = \mathbb{P}_{(x,y)\sim\mu_p}\left[\frac{\partial}{\partial b}\ell(h_{1,b_p};x,y) = 1\right].$$

Note that the derivative of hinge loss with respect to the bias parameter is $\frac{\partial}{\partial b}\ell(h_{w,b};x,y) \in \{-1,0,1\}$, thus we have

$$\mathbb{P}_{(x,y)\sim\mu_p}\left[\frac{\partial}{\partial b}\ell(h_{1,b_p};x,y) = 0\right] = 1-(p_1+p_2).$$

Moreover, according to the first-order optimality of b_p with respect to μ_p, we have

$$\frac{\partial}{\partial b}L(h_{1,b_p};\mu_c) = -\epsilon\cdot\frac{\partial}{\partial b}L(h_{1,b_p};\mu_p) = \epsilon\cdot(p_1-p_2),$$

If we measure the sum of the probability of input having a negative gradient and half of the probability of having zero gradient, we have:

$$p_1 + \frac{1-(p_1+p_2)}{2} = \frac{1}{2} + \frac{p_1-p_2}{2} = \frac{1}{2} + \frac{1}{2\epsilon}\cdot\frac{\partial}{\partial b}L(h_{1,b_p};\mu_c) = \alpha.$$

Therefore, we can construct a mapping g that maps μ_p to ν_α: by moving any $(x,y)\sim\mu_p$ that contributes p_1 (negative derivative) and any $(x,y)\sim\mu_p$ that contributes p_2 (positive derivative) to extreme locations $(-u,+1)$ and $(u,-1)$, respectively, and move the remaining (x,y) that has zero derivative to $(-u,+1)$ and $(u,-1)$ with equal probabilities (i.e., $\frac{1-p_1-p_2}{2}$), and we can easily verify that the gradient of b_p with respect to μ_p is the same as ν_α.

In addition, note that hinge loss is monotonically increasing with respect to the ℓ_2 distance of misclassified examples to the decision hyperplane, and the initial clean loss minimizer $b_c^*(1) \in [-u,u]$, we can verify that the constructed mapping g will not reduce the individual hinge loss. Namely, $\ell(h_{1,b_p};x,y) \leq \ell(h_{1,b_p};g(x,y))$ holds for any $(x,y)\sim\mu_p$. Therefore, we have proven Lemma 5.4.5. □

1.2 Proof of Lemma 5.4.6

Proof of Lemma 5.4.6. First, we introduce the following notations. For any $\mu_p \in \mathcal{Q}(u)$ and any $w \in \{-1, 1\}$, let

$$b_c^*(w) = \operatorname*{argmin}_{b\in\mathbb{R}} L(h_{w,b}; \mu_c), \quad b_p(w; \mu_p) = \operatorname*{argmin}_{b\in\mathbb{R}} [L(h_{w,b}; \mu_c) + \epsilon \cdot L(h_{w,b}; \mu_p)].$$

According to Lemma 5.4.5, we know that the maximum population hinge loss conditioned on $w = 1$ is achieved when $\mu_p = \nu_\alpha$ for some $\alpha \in [0, 1]$. To prove the sufficient and necessary condition specified in Lemma 5.4.6, we also need to consider $w = -1$. Note that different from $w = 1$, we want to specify the minimum loss that can be achieved with some μ_p for $w = -1$. For any $\mu_p \in \mathcal{Q}(u)$, we have

$$L(h_{-1,b_p(-1;\mu_p)}; \mu_c) + \epsilon \cdot L(h_{-1,b_p(-1;\mu_p)}; \mu_p) \geq \min_{b\in\mathbb{R}} L(h_{-1,b}; \mu_c) = L(h_{-1,b_c^*(-1)}; \mu_c). \tag{3}$$

According to Lemma 1.1, we know $b_c^*(-1) = \frac{\gamma_1+\gamma_2}{2}$, which achieves the minimum clean loss conditioned on $w = -1$. Since we assume $\frac{\gamma_1+\gamma_2}{2} \in [-u + 1, u - 1]$, according to the first-order optimality condition, the equality in (3) can be attained as long as μ_p only consists of correctly classified data that also incurs zero hinge loss with respect to $b_c^*(-1)$ (not all correctly classified instances incur zero hinge loss). It can be easily checked that choosing $\mu_p = \nu_\alpha$ based on (5.7) with any $\alpha \in [0, 1]$ satisfies this condition, which suggests that as long as the poisoned distribution μ_p is given in the form of ν_α and if the $w = -1$ is achievable (conditions on when this can be achieved will be discussed shortly), then the bias term that minimizes the distributional loss is equal to $b_c^*(-1)$, and is the minimum compared to other choices of $b_p(-1; \mu_p)$. According to Lemma 5.4.5, it further implies the following statement: there exists some $\alpha \in [0, 1]$ such that

$$\nu_\alpha \in \operatorname*{argmax}_{\mu_p\in\mathcal{Q}(u)} \Big\{ [L(h_{1,b_p(1;\mu_p)}; \mu_c) + \epsilon \cdot L(h_{1,b_p(1;\mu_p)}; \mu_p)] \\ - [L(h_{-1,b_p(-1;\mu_p)}; \mu_c) + \epsilon \cdot L(h_{-1,b_p(-1;\mu_p)}; \mu_p)] \Big\}.$$

For simplicity, let us denote by $\Delta L(\mu_p; \epsilon, u, \mu_c)$ the maximization objective regarding the population loss difference between $w = 1$ and $w = -1$. Thus, a necessary and sufficient condition such that there exists a $h_{-1,b_p(-1;\mu_p)}$ as the loss minimizer is that $\max_{\alpha\in[0,1]} \Delta L(\nu_\alpha; \epsilon, u, \mu_c) \geq 0$. This requires us to characterize the maximal value of loss difference for any possible configurations of ϵ, u and μ_c. According to Lemma 1.1

and the definition of ν_α, for any $\alpha \in [0,1]$, we denote the above loss difference as

$$\Delta L(\nu_\alpha; \epsilon, u, \mu_c) = \underbrace{L(h_{1,b_p(1;\nu_\alpha)}; \mu_c) + \epsilon \cdot L(h_{1,b_p(1;\nu_\alpha)}; \nu_\alpha)}_{I_1} - \underbrace{L(h_{-1,b_c^*(-1)}; \mu_c)}_{I_2}.$$

The second term I_2 is fixed (and the loss on ν_α is zero conditioned on $w=-1$), thus it remains to characterize the maximum value of I_1 with respect to α for different configurations. Consider the auxiliary function

$$g(b) = \frac{1}{2}\Phi\big(\frac{b+\gamma_1+1}{\sigma}\big) - \frac{1}{2}\Phi\big(\frac{-b-\gamma_2+1}{\sigma}\big).$$

We know $g(b) \in [-\frac{1}{2}, \frac{1}{2}]$ is a monotonically increasing function by checking with derivative to b. Let g^{-1} be the inverse function of g. Note that according to Lemma 1.1 and the first-order optimality condition of $b_p(1;\nu_\alpha)$, we have

$$\frac{\partial}{\partial b} L(h_{+1,b}; \mu_c)\big|_{b=b_p(1;\nu_\alpha)} = g\big(b_p(+1;\nu_\alpha)\big) = -\epsilon \cdot \frac{\partial}{\partial b} L(h_{+1,b_p(1;\nu_\alpha)}; \nu_\alpha) = \epsilon \cdot (2\alpha - 1), \tag{4}$$

where the first equality follows from Lemma 1.1, the second equality follows from the first-order optimality condition, and the last equality is based on the definition of ν_α. This suggests that $b_p(1;\nu_\alpha) = g^{-1}\big(\epsilon\cdot(2\alpha-1)\big)$ for any $\alpha \in [0,1]$.

Consider the following two configurations for the term I_1: $0 \notin [g^{-1}(-\epsilon), g^{-1}(\epsilon)]$ and $0 \in [g^{-1}(-\epsilon), g^{-1}(\epsilon)]$. Consider the first configuration, which is also equivalent to $g(0) \notin [-\epsilon, \epsilon]$. We can prove that if $\gamma_1 + \gamma_2 < 0$ meaning that $b_c^*(1) > 0$, choosing $\alpha = 0$ achieves the maximal value of I_1; whereas if $\gamma_1 + \gamma_2 > 0$, choosing $\alpha = 1$ achieves the maximum. Note that it is not possible for $\gamma_1 + \gamma_2 = 0$ under this scenario. The proof is straightforward since we have

$$\begin{aligned} I_1 &= L(h_{1,g^{-1}(2\epsilon\alpha-\epsilon)}; \mu_c) + \epsilon \cdot L(h_{1,g^{-1}(2\epsilon\alpha-\epsilon)}; \nu_\alpha) \\ &= L(h_{1,g^{-1}(2\epsilon\alpha-\epsilon)}; \mu_c) + \epsilon \cdot \big[1 + u + (1-2\alpha)\cdot g^{-1}(2\epsilon\alpha - \epsilon)\big] \\ &= L(h_{1,t}; \mu_c) + \epsilon \cdot (1+u) - t \cdot g(t), \end{aligned}$$

where $t = g^{-1}(2\epsilon\alpha - \epsilon) \in [g^{-1}(\epsilon), g^{-1}(\epsilon)]$. In addition, we can compute the derivative of I_1 with respect to t:

$$\frac{\partial}{\partial t} I_1 = g(t) - g(t) - t \cdot g'(t) = -t \cdot g'(t),$$

which suggests that I_1 is a concave function with respect to t. If $0 \in [g^{-1}(-\epsilon), g^{-1}(\epsilon)]$, we achieve the global

maximum of I_1 at $t = 0$ by carefully picking $\alpha_0 = \frac{1}{2} + \frac{1}{2\epsilon} \cdot g(0)$. If not (i.e., $g^{-1}(-\epsilon) > 0$ or $g^{-1}(\epsilon) < 0$), then we pick t that is closer to 0, which is either $g(-\epsilon)$ or $g(\epsilon)$ by setting $\alpha = 0$ or $\alpha = 1$ respectively. Therefore, we can specify the sufficient and necessary conditions when the weight vector w can be flipped from 1 to -1:

1. When $g(0) \notin [-\epsilon, \epsilon]$, the condition is

$$\max\{\Delta L(\nu_0; \epsilon, u, \mu_c), \Delta L(\nu_1; \epsilon, u, \mu_c)\} \geq 0.$$

2. When $g(0) \in [-\epsilon, \epsilon]$, the condition is

$$\Delta L(\nu_{\alpha_0}; \epsilon, u, \mu_c) \geq 0, \text{ where } \alpha_0 = \frac{1}{2} + \frac{1}{2\epsilon} \cdot g(0).$$

Plugging in the definition of g and ΔL, we complete the proof of Lemma 5.4.6. □

1.3 Proof of Lemma 5.4.7

Proof of Lemma 5.4.7. Let μ_1, μ_2 be the probability measures of the positive and negative examples assumed in (5.6), respectively. Let $\varphi(z; \gamma, \sigma)$ be the PDF of Gaussian distribution $\mathcal{N}(\gamma, \sigma^2)$. For simplicity, we write $\varphi(z) = \varphi(z; 0, 1)$ for standard Gaussian. For any $h_{w,b} \in \mathcal{H}$, we know w can be either 1 or -1. First, let's consider the case of $w = 1$. According to the definition of risk and the data generating process of μ_c, we have

$$\begin{aligned}
\text{Risk}(h_{w,b}; \mu_c) &= p \cdot \text{Risk}(h_{w,b}; \mu_1) + (1-p) \cdot \text{Risk}(h_{w,b}; \mu_2) \\
&= p \cdot \int_{-b}^{\infty} \varphi(z; \gamma_1, \sigma_1) dz + (1-p) \cdot \int_{-\infty}^{-b} \varphi(z; \gamma_2, \sigma_2) dz \\
&= p \cdot \int_{\frac{-b-\gamma_1}{\sigma_1}}^{\infty} \varphi(z) dz + (1-p) \cdot \int_{-\infty}^{\frac{-b-\gamma_2}{\sigma_2}} \varphi(z) dz \\
&= p \cdot \Phi\left(\frac{b+\gamma_1}{\sigma_1}\right) + (1-p) \cdot \Phi\left(\frac{-b-\gamma_2}{\sigma_2}\right).
\end{aligned}$$

Similarly, when $w = -1$, we have

$$\begin{aligned}
\text{Risk}(h_{w,b}; \mu_c) &= p \cdot \int_{-\infty}^{b} \varphi(z; \gamma_1, \sigma_1) dz + (1-p) \cdot \int_{b}^{\infty} \varphi(z; \gamma_2, \sigma_2) dz \\
&= p \cdot \int_{-\infty}^{\frac{b-\gamma_1}{\sigma_1}} \varphi(z) dz + (1-p) \cdot \int_{\frac{b-\gamma_2}{\sigma_2}}^{\infty} \varphi(z) dz \\
&= p \cdot \Phi\left(\frac{b-\gamma_1}{\sigma_1}\right) + (1-p) \cdot \Phi\left(\frac{-b+\gamma_2}{\sigma_2}\right).
\end{aligned}$$

Combining the two cases, we complete the proof. □

Bibliography

H. Aghakhani, D. Meng, Y.-X. Wang, C. Kruegel, and G. Vigna. Bullseye polytope: A scalable clean-label poisoning attack with improved transferability. In *IEEE European Symposium on Security and Privacy*, 2021.

Amazon, Inc. Amazon SageMaker. https://aws.amazon.com/sagemaker/, 2023. Accessed: 2023-05-01.

D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, K. Rieck, and C. Siemens. Drebin: Effective and explainable detection of android malware in your pocket. In *Network and Distributed System Security*, 2014.

J. F. Bard. *Practical bilevel optimization: algorithms and applications*, volume 30. Springer Science & Business Media, 2013.

P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 2006.

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

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

B. Biggio, G. Fumera, and F. Roli. Security evaluation of pattern classifiers under attack. *IEEE Transactions on Knowledge and Data Engineering*, 2013.

M. Billah, A. Anwar, Z. Rahman, and S. M. Galib. Bi-level poisoning attack model and countermeasure for appliance consumption data of smart homes. *Energies*, 2021.

J. Y. Chang and E. G. Im. Data poisoning attack on random forest classification model. *International Conference on Smart Media and Applications*, 2020.

Y. Chen, S. Wang, Y. Qin, X. Liao, S. Jana, and D. Wagner. Learning security classifiers with verified global robustness properties. In *ACM Conference on Computer and Communications Security*, 2021.

Y. Chen, Z. Ding, and D. Wagner. Continuous learning for Android malware detection. In *USENIX Security Symposium*, 2023.

E. G. Dada, J. S. Bassi, H. Chiroma, A. O. Adetunmbi, O. E. Ajibuwa, et al. Machine learning for email spam filtering: review, approaches and open research problems. *Heliyon*, 2019.

A. Demontis, P. Russu, B. Biggio, G. Fumera, and F. Roli. On security and sparsity of linear classifiers for adversarial settings. In *Structural, Syntactic, and Statistical Pattern Recognition: Joint IAPR International Workshop*, 2016.

A. Demontis, M. Melis, M. Pintor, M. Jagielski, B. Biggio, A. Oprea, C. Nita-Rotaru, and F. Roli. Why Do adversarial attacks transfer? Explaining transferability of evasion and poisoning attacks. In *USENIX Security Symposium*, 2019.

I. Diakonikolas, G. Kamath, D. Kane, J. Li, J. Steinhardt, and A. Stewart. Sever: A robust meta-algorithm for stochastic optimization. In *International Conference on Machine Learning*, 2019.

I. Diakonikolas, D. Kane, and N. Zarifis. Near-optimal SQ lower bounds for agnostically learning halfspaces and ReLUs under Gaussian marginals. In *Advances in Neural Information Processing Systems*, 2020.

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

M. Ferrari Dacrema, P. Cremonesi, and D. Jannach. Are we really making much progress? a worrying analysis of recent neural recommendation approaches. In *ACM Conference on Recommender Systems*, 2019.

L. Fowl, M. Goldblum, P.-y. Chiang, J. Geiping, W. Czaja, and T. Goldstein. Adversarial examples make strong poisons. In *Advances in Neural Information Processing Systems*, 2021.

S. Frei, Y. Cao, and Q. Gu. Agnostic learning of halfspaces with gradient descent via soft margins. In *International Conference on Machine Learning*, 2021.

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

J. Guo and C. Liu. Practical poisoning attacks on neural networks. In *European Conference on Computer Vision*, 2020.

I. Guyon, S. R. Gunn, A. Ben-Hur, and G. Dror. Result analysis of the nips 2003 feature selection challenge. In *Advances in Neural Information Processing Systems*, 2004.

S. Hanneke, A. Karbasi, M. Mahmoody, I. Mehalel, and S. Moran. On optimal learning under targeted data poisoning. In *Advances in Neural Information Processing Systems*, 2022.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016.

H. Huang, X. Ma, S. M. Erfani, J. Bailey, and Y. Wang. Unlearnable examples: Making personal data unexploitable. In *International Conference on Learning Representations*, 2021.

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

W. R. Huang, J. Geiping, L. Fowl, G. Taylor, and T. Goldstein. MetaPoison: Practical general-purpose clean-label data poisoning. In *Advances in Neural Information Processing Systems*, 2020.

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

M. Jagielski, G. Severi, N. Pousette Harger, and A. Oprea. Subpopulation data poisoning attacks. In *ACM Conference on Computer and Communications Security*, 2021.

C. Jo, J.-y. Sohn, and K. Lee. Breaking fair binary classification with optimal flipping attacks. In *IEEE International Symposium on Information Theory*, 2022.

A. T. Kalai, A. R. Klivans, Y. Mansour, and R. A. Servedio. Agnostically learning halfspaces. *SIAM Journal on Computing*, 2008.

A. Kerckhoffs. La cryptographie militaire. journal des sciences militaries. *IX (38)*, 1883.

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

P. W. Koh and P. Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, 2017.

P. W. Koh, J. Steinhardt, and P. Liang. Stronger data poisoning attacks break data sanitization defenses. *Machine Learning*, 2022.

A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. *Technical report*, 2009.

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

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, 1998.

Y. Liu, C. Tantithamthavorn, L. Li, and Y. Liu. Explainable AI for Android malware detection: Towards understanding why the models perform so well? In *International Symposium on Software Reliability Engineering*, 2022.

Y. Lu, G. Kamath, and Y. Yu. Indiscriminate data poisoning attacks on Neural Networks. *Transactions on Machine Learning Research*, 2022.

Y. Lu, G. Kamth, and Y. Yu. Exploring the limits of model-targeted indiscriminate data poisoning attacks. In *International Conference on Machine Learning*, 2023.

A. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. Learning word vectors for sentiment analysis. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011.

H. B. McMahan. A Survey of Algorithms and Analysis for Adaptive Online Learning. *Journal of Machine Learning Research*, 2017.

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

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

V. Metsis, I. Androutsopoulos, and G. Paliouras. Spam filtering with Naive Bayes—Which Naive Bayes? In *Conference on Email and Anti-Spam*, 2006.

B. Nelson, M. Barreno, F. J. Chi, A. D. Joseph, B. I. Rubinstein, U. Saini, C. A. Sutton, J. D. Tygar, and K. Xia. Exploiting machine learning to subvert your spam filter. In *USENIX Workshop on Large Scale Exploits and Emergent Threats*, 2008.

OpenAI. Gpt-4 technical report. Technical report, OpenAI, 2023.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 2011.

A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

M. T. Ribeiro, S. Singh, and C. Guestrin. "Why should I trust you?" Explaining the predictions of any classifier. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

E. Rose, F. Suya, and D. Evans. Poisoning attacks and subpopulation susceptibility. In *5th Workshop on Visualization for AI Explainability.*, 2022.

P. J. Rousseeuw, F. R. Hampel, E. M. Ronchetti, and W. A. Stahel. *Robust statistics: the approach based on influence functions*. John Wiley & Sons, 2011.

A. Salah, E. Shalabi, and W. Khedr. A lightweight Android malware classifier using novel feature selection methods. *Symmetry*, 2020.

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

S. Shalev-Shwartz. Online learning and online Convex Optimization. *Foundations and Trends in Machine Learning*, 2012.

S. Shalev-Shwartz and S. Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014.

N. Šrndic and P. Laskov. Detection of malicious PDF files based on hierarchical document structure. In *Network and Distributed System Security*, 2013.

J. Steinhardt, P. W. W. Koh, and P. S. Liang. Certified defenses for data poisoning attacks. In *Advances in Neural Information Processing Systems*, 2017.

X. Sun, Z. Zhang, X. Ren, R. Luo, and L. Li. Exploring the vulnerability of deep neural networks: A study of parameter corruption. In *AAAI Conference on Artificial Intelligence*, 2021.

F. Suya, X. Zhang, Y. Tian, and D. Evans. When can linear learners be robust to indiscriminate poisoning attacks? *arXiv preprint arXiv:2307.01073*, 2023.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

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

F. Tramèr and D. Boneh. Differentially private learning needs better features (or much more data). *International Conference on Learning Representations*, 2021.

W. Wang, A. Levine, and S. Feizi. Lethal Dose Conjecture on data poisoning. In *Advances in Neural Information Processing Systems*, 2022.

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

D. Yu, H. Zhang, W. Chen, J. Yin, and T.-Y. Liu. Availability attacks create shortcuts. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022.

Z. Zhang, Y. Zhang, D. Guo, L. Yao, and Z. Li. Secfednids: Robust defense for poisoning attack against federated learning-based network intrusion detection system. *Future Generation Computer Systems*, 2022.

C. Zhu, W. R. Huang, A. Shafahi, H. Li, G. Taylor, C. Studer, and T. Goldstein. Transferable clean-label poisoning attacks on Deep Neural Nets. In *International Conference on Machine Learning*, 2019.