GAME ANALYSIS OF DEFENCES AGAINST POISONING ATTACK

GAME THEORETIC ANALYSIS OF DEFENCE ALGORITHMS AGAINST DATA POISONING ATTACK

BY

YIFAN OU

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTING AND SOFTWARE

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

Master of Science (2020) McMaster University
(computing and software) Hamilton, Ontario, Canada

TITLE:	Game Theoretic Analysis of Defence Algorithms Against Data Poisoning Attack
AUTHOR:	Yifan Ou McMaster University, Hamilton, Canada
SUPERVISOR:	Dr. Reza Samavi
NUMBER OF PAGES:	x, 57

Lay Abstract

As Machine Learning (ML) algorithms are deployed to solve a wide variety of tasks in today's world, data poisoning attack poses a significant threat to ML applications. In this work, we study the defence against poisoning attack scenario as a competitive game between the defender and the adversary and analyze the game characteristics for several defence algorithms. Our goal is to identify the optimal defence strategy against poisoning attacks, even when the adversary responds optimally to the defence strategy. We propose a game model for the poisoning attack scenario, and develop an efficient algorithm to approximate for the Nash Equilibrium defence strategy. Our approach does not only provide insights about the effectiveness of the analyzed algorithms under optimal poisoning attacks, but also serves as a method for the modellers to determine capable defence algorithms and optimal strategies to employ on their ML models.

Abstract

As Machine Learning (ML) algorithms are deployed to solve a wide variety of tasks in today's world, data poisoning attack poses a significant threat to ML applications. Although numerous defence algorithms against data poisoning attack have been proposed and shown to be effective, most defence algorithms are analyzed under the assumption of fixed attack strategies, without accounting for the strategic interactions between the attacker and the defender. In this work, we perform game theoretic analysis of defence algorithms against data poisoning attacks on Machine Learning. We study the defence strategy as a competitive game between the defender and the adversary and analyze the game characteristics for several defence algorithms. We propose a game model for the poisoning attack scenario, and prove the characteristics of the Nash Equilibrium (NE) defence strategy for all distance-based defence algorithms. Based on the NE characteristics, we develop an efficient algorithm to approximate for the NE defence strategy. Using fixed attack strategies as the benchmark, we then experimentally evaluate the impact of strategic interactions in the game model. Our approach does not only provide insights about the effectiveness of the analyzed algorithms under optimal poisoning attacks, but also serves as a method for the modellers to determine capable defence algorithms and optimal strategies to employ on their ML models.

Acknowledgements

I would like to express my very great appreciation to Dr. Samavi for his valuable and constructive suggestions during the planning and development of this research work. His willingness to give his time so generously has been very much appreciated. I would also like to to thank my family for the support and encouragement throughout my study.

Contents

List of Figures

List of Tables

Abbreviations

AI	Artificial Intelligence
ML	Machine Learning
NE	Nash Equilibrium
DNN	Deep Neural Network
SVM	Support Vector Machine
KKT	Karush-Kuhn-Tucker

Chapter 1

Introduction

1.1 Motivation

Artificial Intelligence (AI) has a wide range of applications in our lives. Modern AI applications usually embed Machine Learning (ML) algorithms to solve a wide range of complex tasks from navigation and personal recommendation systems for consumer use to a larger scale decision making systems such as speech recognition and computer vision. While effectiveness and application diversity of ML-embedded AIs improve, the incentive to compromise AI by attack the underlying ML algorithms also rises. In safety-critical applications such as image recognition on self-driving cars or tumor detection in diagnostic imaging, serious consequences may occur if the ML model is compromised by adversaries. Hence it is imperative for the modeller to ensure that the ML model is robust even with the presence of potential attacks.

There are variety of attack vectors that the adversaries can employ to compromise the outcome of ML models. An adversary may try to fool an ML classifier in the inference phase with *adversarial examples* Lin *et al.* (2017), where the attacker attempts to craft malicious inputs with an imperceptible perturbation from genuine inputs (e.g., putting stickers on certain positions of a stop sign to cause an autonomous vehicle to classify it as a speed limit sign (Eykholt *et al.*, 2017)). Other attackers may try to alter the ML model during its training phase by corrupting the ML model directly and injecting relatively undetectable malicious data points into the training sample, such that when the ML model is trained on the contaminated dataset, it will misbehave in the favour of the attacker. This type of attack is commonly called *poisoning attack* (Papernot, 2016) and considered an emerging security threat to machine learning models, particularly when training the model relies on the collection of large amounts of data in the wild (Muñoz González *et al.*, 2017; Joseph *et al.*, 2013). In application domains such as malware detection, a small amount of carefully crafted malicious datapoints is sufficient to significantly degrade the performance of ML algorithms (Huang *et al.*, 2011). From an adversary's perspective, in the worst case, the poisoning attack compromises model availability by maximizing the overall classification error (Biggio *et al.*, 2012) and in the best case, the attack compromises model integrity to cause the attacker's desired misclassification on a subset of datapoints (Liu *et al.*, 2018b).

One approach to reduce the effect of poisoning attacks is sanitizing the input dataset prior to being used for training. Sanitization algorithms involve constructing a feasible domain, and reject all datapoints that are outside of the feasible domain (Koh *et al.*, 2018). While sanitization algorithms are typically effective against outlying poisoning points, a sophisticated attacker can adjust his attack strategy to make the injected datapoints less detectable while still delivering a less successful attack even with the presence of defence. To counteract such attempt, the defender may also adjust the strength of the filter to mitigate a less conceivable attack, at the price of removing genuine datapoints. Since both the attacker and the defender can change their strategies, the poisoning attack scenario can be modelled as a competitive game. Studying the game theoretic characteristics of poisoning attacks is a necessary step for modellers to devise an optimal defence strategy against a strategical adversary.

1.2 Current State-of-the-art Defences

Numerous defence algorithms are analyzed in the field of ML security (Paudice *et al.*, 2018; Nelson *et al.*, 2009; Jagielski *et al.*, 2018; Rubinstein *et al.*, 2009; Laishram and Phoha, 2016). However, during their analysis, the attacker's strategy is chosen under the assumption that no defender is present, hence malicious points are detected by the defence algorithm with ease, leading to over-optimistic conclusions. Therefore to truly evaluate the performance of a defence algorithm, the *strategic interaction* between the attacker and the modeller must be accounted for.

1.3 Methodology

The goal of the game analysis is to identify the Nash Equilibrium (NE), from which the result of the game and the behaviours of each player can be predicted when all players try to selfishly maximize their own payoff. A part of our work is published in (Ou and Samavi, 2019), in which we formulated the game model for the poisoning attack and defence scenario and analyzed the L2 (Centroid) defence algorithm using the game model. We analyze multiple defence algorithms using the game model we propose, and experimentally analyze the performance of the optimal strategy in comparison with the state-of-the-art pure strategy defences.

1.4 Contribution

In this work, we make three contributions. First, we develop a game model which simulates the *strategic interaction* between the adversary and the modeller for all distance-based defence algorithms. Second, we prove the characteristics of NE defence strategy, and derive an efficient algorithm to approximate for the NE strategy. Third, we experimentally evaluate the effectiveness of proposed defence methods.

1.5 Thesis Structure

The rest of this thesis is organized as follows: In Chapter 2 we describe the related work. In Chapter 3 we introduce our game model and present the theoretic analysis on the game model. In Chapter 4 we describe the analyzed defence algorithms. In Chapter 5 we show our experimental results of defence algorithms against optimal attacks. Finally, in Chapter 6 we conclude our work and discuss the potential future work.

Chapter 2

Related Work

2.1 Poisoning Attacks

In the poisoning attack scenario, the attacker aims to find the set of malicious points which maximizes the attacker's objective function. Authors in (Mei and Zhu, 2015) formulated poisoning attack as a bi-level optimization problem:

$$\begin{aligned} &D_c \in \underset{D_c \in \phi(D)}{\operatorname{argmax}} \; O_A(D_{val}, w') \\ &\text{subject to } w' \in \underset{w' \in W}{\operatorname{argmin}} \; L(D_T \cup D_c, w') \end{aligned} \tag{2.1.1}$$

where D_c is the set of malicious data points, $\phi(D)$ is a function that maps the malicious points onto the feasible domain, O_A is the attacker's objective function, D_{val} is the validation dataset used by the attacker, w' is the trained classifier, D_T is the original training dataset, and L is the loss function of the ML model. By solving this optimization problem, the attacker is able to identify the best possible malicious set which maximizes his objective O_A.

The challenge of conducting poisoning attack is that, the bi-level optimization problem of poisoning attack (Equation. 2.1.1) is generally non-convex and NP hard to solve. Therefore attacking with brute-force will be resource-demanding, especially for complex ML models like Deep Neural Networks (DNN). More efficient approaches for conducting poisoning attack are proposed and analyzed in recent proposals. For instance, authors in (Muñoz González *et al.*, 2017) apply back-gradient optimization method to compute the gradient more efficiently; authors in (Koh and Liang, 2017) use *influence functions* to approximate the gradients; authors in (Koh *et al.*, 2018) proposed KKT attack in which the attacker has a desired resulting classifier $w*$ in mind, and places malicious points such that the classifier trained using $D_T \cup D_c$ is close to $w*$. There are also heuristic attack methods which perform well under certain circumstances. For example, when the training dataset D_T and the attacker's validation dataset D_{val} have similar distribution, the *minimax* attack proposed in (Steinhardt *et al.*, 2017) reduces the bi-level optimization to a *saddle point problem* which can be solved with ease via gradient descent. Also, authors in (Jagielski *et al.*, 2018) design an attack method that specifically targets linear Support Vector Machines (SVM). There are other attack algorithms that focus on decreasing the computation time by sacrificing performance. For example, in *label-flipping attacks* (Xiao *et al.*, 2015), the attacker generates the malicious points by inverting the label of a subset of genuine points instead of directing the malicious points to the optimal positions.

To conduct a successful poisoning attack, knowledge to the original training dataset D_T and the training algorithm is crucial for solving the optimization problem shown in Equation. 2.1.1. Attackers with access to such information are called *white-box* attackers. Attacks with incomplete information are called *black-box* attacks. Although *white-box attacks* are less common in practice, as attackers usually have limited knowledge to the ML model, analyzing *white-box attacks* can provide upper bounds on the impact of the attack on the ML model. In addition, authors in (Papernot *et al.*, 2016) demonstrated that *black-box* attacks can be converted to *white-box* easily, as attacks are **transferable** across different ML models and training datasets. That is, although the attackers may not have access to the training dataset D_T nor the model internals directly, they can acquire an auxiliary ML model and training dataset D'_T with a similar distribution to D_T, then the attacker can perform the attack to the auxiliary model trained with D'_T and use the resulting set of malicious data points to successfully attack the original ML model. Therefore, in this research, we will focus on the more general problem of *white-box attacks*.

2.2 Defence Algorithms Against Poisoning Attacks

There exists a wide range of data sanitization algorithms against poisoning attacks. The majority of proposed sanitization algorithms are distance-based. Distance-based sanitization defence involves computing the distance value d_i for every datapoint i, and remove points with $d_i > \theta_d$, where θ_d is the threshold value chosen by the defender. Notice that each defence algorithm can have a unique measurement for distance. For instance, L2 (Centroid) defence (Kloft and Laskov, 2012) measures the Euclidean (L2) distance from each point to the centroid; loss defence (Koh *et al.*, 2018) uses the loss value of the datapoints as the measurement for distance; PCA anomaly detection (Rubinstein *et al.*, 2009) measures the Euclidean distance of points in the low-variance subspace; and Curie (Laishram and Phoha, 2016) defines its own measurement of distance. Non-distance-based filters, such as Reject on Negative Impact (RONI) (Nelson *et al.*, 2009) and TRIM (Jagielski *et al.*, 2018), filter by repetitively training the model and retaining points that maximizes accuracy. Regularization is also used as a defence algorithm against poisoning attacks (Farokhi, 2020). Instead of filtering out the suspicious datapoints, regularization protects the model by altering the loss function in training time to reduce the impact of malicious points.

All of these defence algorithms demonstrate capability in mitigating poisoning attack. However, one significant drawback in their evaluation is that, the results could be over-optimistic, when it's assumed that the attacker uses a fixed attack strategy during their evaluation instead of adjusting attack strategy to avoid detection. In fact, authors in (Paudice *et al.*, 2018; Koh *et al.*, 2018) demonstrate that several defence algorithms with fixed defence strategy failed to mitigate coordinated weaker attacks despite being able to remove outlying malicious points. The result is over-pessimistic, because the defender is allowed to choose his defence strategy in reality to maximize his objective. None of these proposals have considered the strategic interaction between the attacker and the defender, which we will investigate in this paper for several well-known distance-based defence algorithms.

2.3 Game Analysis of ML Security

Game theoretic analysis in poisoning attack is first studied in (Liu and Chawla, 2009), which models the poisoning attack scenario as a zero-sum Stackelberg game between the attacker and an **undefended** ML model, and proposed an algorithm to identify the Stackelberg equilibrium attack strategy. The game model in (Liu and Chawla, 2009) is further extended in the work of (Zhang and Zhu, 2017; Ma *et al.*, 2018) to analyze the impact of poisoning attacks on different ML applications like Distributed Support Vector Machines (DSVM) and Data-driven Learning Games. Game model for poisoning **defended** ML models is proposed in (Koh *et al.*, 2018). One major limitation to the state-of-the-art game models for poisoning attack is that, the defence algorithm is either absent in the game analysis, or assumed the defence algorithms is using a fixed pure strategy, allowing the attacker to subvert the defence easily. We will address this limitation by designing a game model which allows the defender to choose his strategy freely, as presented in the next section.

Game models are also being used to analyze a variety of other ML security related problems. For instance, authors in (Brückner and Scheffer, 2011; Brückner *et al.*, 2012) studied ML evasion attack as a game between the adversary and the modeller, who is allowed to deploy any arbitrary classifier to thwart the attack instead of automatically using the classifier that maximizes test accuracy. Authors in (Wang *et al.*, 2019) proposed a Stackelberg Cybersecurity Investment Game (SCIG) which aims to identify the optimal allocation of resources to prevent cyber attacks.

Chapter 3

Game Model

In this chapter, we first provide the preliminary definitions of game theories. Then, we formulate the poisoning attack with defence scenario as an optimization problem. The minimax property of such optimization problem tell us that game analysis is valid. Then we introduce our game model, as well as any assumptions made in game analysis. Lastly, we prove the characteristics of NE defence strategies in our game model.

3.1 Preliminaries Game Theory

In a two player game, denote X_i as the *pure strategy set* with cardinality n_i of player i, X_i represents the set of all legal moves that player i can perform, $i \in \{1, 2\}$. The cardinality of X_i can be infinite, such games are called *continuous games* (Glicksberg, 1952); a game is a *discrete game* if both X_1 and X_2 has finite cardinality. Let $X = X_1 \times X_2$ represent all possible combination of choices made by both players. Then each player's *payoff function* is denoted as $u_i : x \in X \to \mathbf{R}$, which the player wish to maximize.

In a simultaneous game (also referred as *Nash game*), each player chooses a legal move $x_{i_j} \in X_i$ to play, $1 \leq j \leq n_i$, the choices are made simultaneously. Then each player is able to evaluate his payoff function as $u_i(x \in X)$. A two player game is *zero-sum* if $u_1(x) = -u_2(x)$. Each player chooses their legal move based on their own *strategy.* A (mixed) *strategy* of any player i is the set of his legal moves X_i, each associated with a probability to play the corresponding move, forming a vector P_i with same cardinality as X_i. Hence a strategy for player i can be defined as $S_i = \{X_i, P_i\}$, with $\sum_{S_i} P_i = 1$. A strategy is a *pure strategy* if any move is associated with probability of 1 ($\exists p_j \in P_i | p_j = 1$). Denote $S = \{X_1 \times X_2, P_1 \otimes P_2\}$ as the combined mixed strategies used by both players, the mixed extension of the payoff function $M_i(S) = \sum_{\{x,p\} \in S} u_i(x) * p$ can be used to represent the expected payoff of mixed strategies.

A Nash Equilibrium (NE) is the equilibrium state of all players in a simultaneous game. When all players in the game are using the NE strategies, no players can secure a better payoff by deviating from his NE strategy. In another word, using a NE strategy maximizes a player's payoff, if the other player is playing rationally to maximize his own payoff. Hence in a competitive game, a NE strategy is a optimal strategy against a rational opponent. There may be multiple different NE strategies in a game, resulting different NEs, but the payoff of all NEs are always identical.

3.2 Problem Formulation

To derive the attack-defence scenario as an optimization problem, recall from Section. 2.1 that the bi-level optimization problem for poisoning attack is formulated as

Formula. 2.1.1. Notice that no defence algorithm is considered in Formula. 2.1.1, the model will be trained with all incoming datapoints in attempt to minimize training loss.

Now, when defence algorithm is taken into consideration, the defender seeks to minimize the effect of the attack by identifying the best defence strategy. In distance-based defence algorithms, the defender chooses a filtering threshold θ_d, and remove datapoints with distance parameter $\geq \theta_d$. Denote M_d as the set of probability measure on the Borel subset of all possible values of θ_d(M_d represents the defender's *mixed strategy set*), F as the feasible set constructed by the defender using M_d (datapoints not in F will be removed by the filter). The defence objective can be formulated as the following 3-level optimization:

$$\begin{aligned}
&\min_{M_d} E[O_A(D_{val}, w')] \\
&\text{subject to} D_c \in \operatorname*{argmax}_{D_c \in \phi(D)} E[O_A(D_{val}, w')] \\
&\text{subject to } w' \in \operatorname*{argmin}_{w' \in W} E[L((D_T \cup D_c) \cap F, w')]
\end{aligned} \tag{3.2.1}$$

where $E[O_A(D_{val}, w')]$ is the expected value of O_A. In Formula. 3.2.1, the defender aims to identify the best defence strategy to thwart the attack. The validity of Formula. 3.2.1 requires the following assumption:

Assumption 1. All damages done by the filter (e.g. removing genuine points) benefits the attacker.

Assumption 1 is valid because removing genuine points will cause the ML model to be trained with less genuine information, and be more dependant on malicious information which survived from the filter. Therefore Assumption 1 implies that the defender must balance between the impact of the attack and the loss of genuine points, preventing the defender from filtering aggressively. Most existing literature that propose effective defence algorithms (Paudice *et al.*, 2018; Lakhina *et al.*, 2004; Laishram and Phoha, 2016; Jagielski *et al.*, 2018) implicitly formulate the attack scenario as Formula. 3.2.1, solving it through experimental analysis. Notice that Formula. 3.2.1 only describe the scenario of defending against a uninformed attacker, who is unaware of the presence of the defence algorithm. An informed adversary may be able to adjust the attack strategy accordingly and subvert the defence algorithm. Therefore while analyzing the effectiveness of defence algorithms, it is imperative to also consider the optimal response from the attacker.

To model the attacker's response to the defence algorithm, an additional optimization level is required. Let R represent the maximum distance restriction on all malicious points. Setting $R = \theta_d - \epsilon$ will allow all malicious points to survive from the filter, although choosing a small value for R may reduce the effect of the malicious points. Denote M_a as the probability measure on the Borel subset of all R (M_a is the attacker's *mixed strategy sets*). Now we can formulate the attack-defence scenario as follows:

$$\begin{aligned} &\max_{M_a}\min_{M_d} E[O_A(D_{val}, w')] \\ &\text{subject to} D_c \in \operatorname*{argmax}_{D_c \in \phi(D, M_a)} E[O_A(D_{val}, w')] \qquad (3.2.2) \\ &\text{subject to } w' \in \operatorname*{argmin}_{w' \in W} E[L((D_T \cup D_c) \cap F, w')] \end{aligned}$$

where the feasible attack domain $\phi(D, M_a)$ is further restricted by the attack strategy M_a. The first two level of Formula. 3.2.2 is a minimax optimization problem, which corresponds the formulation of a two player zero-sum game in game theoretic analysis. Therefore, the resulting optimization problem provides justification for the use of a two player zero-sum game model for poisoning attack.

3.3 Game Model

We will formulate the attack-defence scenario as a two player competitive zero-sum game between the attacker and the defender. In this game model, the attacker (denoted by a) chooses a set $S_a = \{[r_1, n_1], [r_2, n_2], ...[r_m, n_m]\}$, where for each radius $r_i \in S_a$, n_i poisoning points will be placed optimally within r_i distance ($\sum_{i=1}^{m} n_i = N_c$, where N_c is the total number of maliciously injected points). Denote $R = \max(r_i \in S_a)$, hence all malicious points have distance $\leq R$. The attacker will therefore position all poisoning points to maximize his payoff within R distance. The defender (denoted by d) chooses a defence threshold value θ_d, such that the data points with distance greater than θ_d will be removed. The measurement of distance is dependant on the defence algorithm (as described in Section. 2.1), and both parties will be using the distance measurement of the defence algorithm.

In game analysis, the objective functions of both players are assumed to be public information (known to both players). Since the defender aims to minimize the adversarial impact on the ML model, the defender will always use the same objective function as the attacker, while attempting to minimize the objective instead. Therefore, the game is always zero-sum. However, if the attacker's objective function is unknown to the defender (e.g. in the context of Trojaning attack (Liu *et al.*, 2018b)) and needs to be approximated instead, the game may not be zero-sum. The validity of game analysis for this non-zero-sum scenario is heavily dependant on how well the approximation of the attacker's objective function is, and is therefore considered beyond the scope of this work.

During a poisoning attack scenario, the adversary acts first by injecting malicious points into the training dataset. The defender then inspects the poisoned dataset and decides which datapoints to retain. However, the defender cannot possibly infer the attacker's choice of S_a from the potentially contaminated dataset, and the attacker cannot foresee the choice of θ_d. Therefore instead of using a Stackelberg game model (Von Stackelberg, 2010), a simultaneous (Nash) game model is more appropriate for poisoning attack and defence.

The outcome of the game is determined using two functions $E(S_a, \theta_d)$ and $\Gamma(\theta_d)$. $E(S_a, \theta_d)$ represents the attacker's gain from the injected malicious points, and $\Gamma(\theta_d)$ represents the defender's loss for removing genuine datapoints. Since R serves as a constraint on the attacker, increasing the value of R will give the attacker more freedom to position the poisoning points, hence we can expect that $E(S_a, \theta_d)$ is non-decreasing as R increases. The defence threshold θ_d restricts the maximum value of R, therefore reduces the maximum value of $E(S_a, \theta_d)$. However, as θ_d decreases, the filter will remove more genuine points, leading to a higher defence loss $\Gamma(\theta_d)$. The utility functions $U(S_a, \theta_d)$ of the game model can be represented as:

$$\begin{aligned} U(S_a, \theta_d) &= U_a(S_a, \theta_d) = -U_d(S_a, \theta_d) \\ U(S_a, \theta_d) &= E(S_a, \theta_d) + \Gamma(\theta_d) \end{aligned} \tag{3.3.1}$$

Although the most straightforward measurement for $E(S_a, \theta_d)$ and $\Gamma(\theta_d)$ is the classification error of the ML model caused by the attack and defence, $E(S_a, \theta_d)$ and $\Gamma(\theta_d)$ can be measured differently. For instance, in the online learning scenario, loss due to removing incoming genuine points are often not reflected in the degradation of model performance. Because the filter can at worst remove all incoming datapoints to retain classification accuracy of the previous classifier, which violates the purpose of online learning. Therefore, online learning modellers can measure $\Gamma(\theta_d)$ in terms of the amount of genuine points removed instead, with a customized conversion function to match the measurement of $E(S_a, \theta_d)$(e.g. $\Gamma(\theta_d) = 0.2\%$ classification error for every 1% of genuine points removed). The Nash Equilibrium (NE) defence strategy of the game is dependant on the choice of metrics of $E(S_a, \theta_d)$ and $\Gamma(\theta_d)$, but the resulting NE strategy always maximizes the customized payoff function of the modeller.

3.4 Characteristics of NE Defence Strategy

3.4.1 Pure Strategy NE

Proposition 1: In our game model, a pure strategy NE exists only when $T_a = T_d$ where,

- T_a is the minimum distance where poisoning points yield benefit for the attacker: $E(r_i, n_i) \leq 0, \forall r_i <= T_a$.
- T_d is the value of distance where reducing the filtering threshold θ_d below T_d causes the defender's payoff to be strictly lower than $U_d(S_a, T_d)$. $(U_d(R, i) < U_d(R, T_d), \forall i < T_d)$

Proof. The best respond functions (BRF) (Nash, 1950) of the attacker and the defender are:

$$\beta_a(\theta_d) = \begin{cases} [\theta_d, N] & \theta_d \geq T_a \quad (3.4.1a) \\ \text{all } r_i \geq T_a & \text{otherwise} \quad (3.4.1b) \end{cases}$$

$$\beta_d(S_a) = \begin{cases} B & \text{if } r_i \leq T_d, \forall r_i \in S_a \quad (3.4.2a) \\ r_i - \epsilon & r_i = \min\{r_i \in S_a | r_i > T_d\} \quad (3.4.2b) \end{cases}$$

where B is the boundary, aka maximum possible distance in the scenario.

Suppose that (S_a^*, θ_d^*) are the NE strategies. Then

$$\begin{cases} (S_a^*, \theta_d^*) \in \beta_d(S_a^*) \\ (\theta_d^*, S_a^*) \in \beta_a(\theta_d^*) \end{cases}$$

In such condition, the NE strategy set (S_a^*, θ_d^*) must satisfy one of 3.4.1a and 3.4.1b, as well as one of 3.4.2b and 3.4.2a. Denote r_{min} as the minimum radius in S_a such that $r_{min} > T_d$. In the remainder of the proof, we relax the condition above into the following:

$$\begin{cases} (r_{min}^*, \theta_d^*) \in \beta_d(S_a^*) \\ (\theta_d^*, r_{min}^*) \in \beta_a(\theta_d^*) \end{cases}$$

Note that each choice of $r_{min} \in \beta_a(\theta_d)$ may represent multiple possible attacker's responses S_a. Therefore, if the NE (r_{min}^*, θ_d^*) does not exist for this relaxed problem, the NE for the original problem will not exist either. Clearly 3.4.1a and 3.4.2b cannot be satisfied simultaneously, as $\beta_a(\theta_d) = r_{min}$ and $\beta_d(r_{min}) = r_{min} - \epsilon$ does not intersect. Similarly 3.4.1b and 3.4.2a does not intersect because 3.4.1b has condition $\theta_d < T_a$, but $\beta_d(S_a) = B$ and $B >> T_a$. 3.4.1a and 3.4.2a also does not intersect, as $\beta_a(\theta_d) = r_{min}$ intersects $\beta_d(r_{min}) = B$ at (B, B), which violates the condition of 3.4.2a.

The only possible intersection of the BRFs is 3.4.1b and 3.4.2b. This will occur when $T_a \geq T_d$, at $(\theta_d = T_a, r_{min} = T_a + \epsilon)$. However, it is impossible to have $T_a > T_d$, because the attacker will not place poisoning points inside T_a (doing so yields no profit). If the defender move θ_d from T_a towards T_d, he will lose from $\Gamma(\theta_d)$ and gain nothing from $E(S_a)$, thus violates the definition of T_d. Therefore, a pure strategy NE only exists when $T_a = T_d$ □

In other words, an optimal pure strategy defence exists only when the attacker and the defender lose incentive to inject/defend simultaneously. By the definition of T_a and T_d we can infer that, if a pure strategy NE exists and is used by the defender (by setting $\theta_d = T_a$), then the defender essentially forced the attacker to lose attack incentive (as it is impossible for the attacker to gain via injecting malicious points in any way). The defence will therefore mitigate all poisoning attacks with an acceptable loss from removing genuine points. Therefore if a pure strategy NE exists for a defence algorithm, then the defence algorithm must be very effective against poisoning attacks.

3.4.2 Mixed Strategy NE

Proposition 2: In our game model, a mixed strategy NE exists.

Proof. According to (Reny, 1999), if the mixed-extension of the game is *reciprocally upper semi-continuous* and *payoff secure*, then it has a mixed strategy NE. A zero-sum game is always reciprocally upper semi-continuous in its mixed extension (Reny, 1999). A game is payoff secure if for every $x \in X$ and every $\epsilon > 0$, each player can secure a payoff of $U(X) - \epsilon$ at x. That is, each player has a way to guarantee his payoff at every $x \in X$, even if the other player deviates from x slightly. All continuous games (infinite games with continuous payoff functions) possesses this property, but many discontinuous games also do.

Our game model is payoff secure, because decreasing θ_d slightly will at worst decrease U_d slightly, if we make a continuous approximation on $\Gamma(\theta_d)$, as long as the attacker's mixed strategy does not change too much. Similarly, decreasing a radius in S_a slightly will at worst decrease U_a slightly. Therefore, our game possesses a mixed strategy NE. $\square$

Notice that any pure strategy is a degenerate case of a mixed strategy, in which that particular pure strategy is selected with probability 1 and every other strategy with probability 0. Therefore a NE strategy always exists for our game model, regardless of which defence algorithm is being analyzed.

Proposition 3: The NE defence strategy must meet the following condition: for every θ_d in the defender's mixed strategy $m \in M_d$ with $pdf_m(\theta_d) > 0$, the product of $E(\theta_d)$ and $cdf_m(\theta_d)$ must be equal, where *cdf* is the cumulative density function of m.

Proof. Suppose that a defender's strategy m^* is a NE strategy that does not meet the condition, then:

$$\exists(\theta_x, \theta_y) \in m^* | cdf_{m^*}(\theta_x) * E(\theta_x) > cdf_{m^*}(\theta_y) * E(\theta_y) \wedge pdf_{m^*}(\theta_x) > 0, pdf_{m^*}(\theta_y) > 0$$

Then, the attacker will not place any malicious points on θ_y, because doing so yields less profit than placing on θ_x. Then, the defender may increase the value of θ_y until $cdf_{m^*}(\theta_x) * E(\theta_x) = cdf_{m^*}(\theta_y) * E(\theta_y)$ (or if $\theta_y = B$, we can shift probability from θ_x to θ_y). This will reduce $\Gamma(m)$ and hence increase the defender's profit without altering the attacker's choice of strategy (θ_x remains no less attractive than θ_y

throughout this process). This contradicts with the assumption that m^* is the NE strategy.

We have shown that any strategy that does not meet the above condition cannot be NE strategies, therefore a NE strategy must possess the property above. □

Notice that when the condition is met, malicious points placed on each θ_d with $pdf_m(\theta_d) > 0$ yields the same profit for the attacker. Therefore the attacker is *indifferent* among his strategies (as long as all malicious points are placed on some θ_d with $pdf_m(\theta_d) > 0$ in any combination). The NE strategy of the defender is simply the strategy which minimizes the attacker's profit while satisfying the condition above. Using these properties, we can derive an algorithm to approximate the defender's NE strategy (Algorithm. 1). The algorithm starts with a set of initial filter radii. In every iteration, the algorithm calculates the probability that satisfies the conditions described in the proof above for every radius in the set. Then it performs gradient descent on the set of radius to minimize the defender's loss function. Notice that placing all poisoning points within the strongest filtering radius r_{min} is one of the optimal attack strategy, therefore we can safely use its resulting loss $N * E(r_{min})$ to represent the loss from an optimal attack.

Computing an exact NE strategy may be time consuming and infeasible due to the unbounded number of radius that the defender can include in his mixed strategy. However, computing the NE strategy which uses a fixed number of radius is possible and is usually sufficient in practice.

INPUT:

1. $\Gamma(p)$ - the estimated loss for removing genuine points(p = fraction of points to remove)
2. $E(p)$ - the maximum effect of a poisoning point placed in that percentile
3. n - number of radius in mixed strategy
4. ϵ - convergence threshold
5. N - expected number of poisoning data points in the dataset

OUTPUT:

1. M_d - the NE mixed strategy of defender
2. $U_d(M_d, *)$ - the resulting impact to the ML model

```
$\{r_1, r_2, ..., r_n\}$ = chooseInitialRadius($n$) $S_r = \{r_1, r_2, ..., r_n\}$
$t = 0$
while $f(S_r)^t - f(S_r)^{t-1} < \epsilon$ do
    $pdf$ = findPercentage($S_r$)
    $r_{min} = \min(S_r)$
    $f = N * E(r_{min}) + \int_0^1 pdf(p_i) * \Gamma(p_i) dp_i$
    Compute $\nabla(f(S_r)) = \frac{df}{dS_r}$
    $S_r = S_r - \nabla(f(S_r))$
    $t = t + 1$
end
return $\{S_r, pdf\}$, $f(S_r)$
```

Algorithm 1: Compute Optimal Defense

Chapter 4

Defence Algorithms

In this chapter, we will describe several defence algorithms in online learning settings. We will first introduce the concept of *online learning* and *online defence algorithms*, then discuss the defence algorithms that we analyze in this work.

4.1 Online Learning & Offline Learning

In the *online learning* scenario (Sugiyama, 2015), the model is initially developed and trained under the supervision of the modeller on day 0. During this process, the training dataset may be gathered internally and is unexposed to attacks. Furthermore the dataset can be sanitized manually to prevent any tampering. Hence we can assume that the training dataset is clean when the ML model is deployed on day 0.

After the ML model is deployed, it is constantly retrained with newly gathered data to improve accuracy and to capture the new trends of data. After gathering dataset D_i from day i, the new ML classifier w_i is trained using $\bigcup_{j=0}^{i} D_j$, with the assumption that the ML model has infinite memory for the datasets.

It is easy for an adversary to inject malicious datapoints during the data gathering process on any day $i > 0$ to contaminate the training dataset. After gathering data from day i, the ML developer will have dataset $D_i = D_{T_i} \cup D_{c_i}$, where D_{T_i} is the genuine datapoints from D_i and D_{c_i} is the malicious dataset. In addition, the defender has the previous classifier w_{i-1}, as well as datasets $\bigcup_{j=0}^{i-1} D_j$, starting with the uncontaminated dataset D_0. This information may be utilized in detecting anomalies in the new dataset D_i.

In the offline learning attack-defence scenario, the attacker starts by injecting malicious dataset D_c into the training set D_T, so that $D_T \cup D_c$ is received by the modeller. The modeller chooses a defence algorithm and derives a defence strategy based on $D_T \cup D_c$, in order to remove suspicious points in $D_T \cup D_c$. Defending in offline learning is harder, as the defence algorithm will be affected by D_c. We will call the defence algorithms that requires uncontaminated D_0 as *online defence algorithms*, and defence algorithm that does not require D_0 as *offline defence algorithms*.

However, if the offline learning modeller is able to curate small fraction of datapoints $D_s \subset D_T \cup D_c$ to ensure that D_s is clear from contamination, then the offline learning scenario can be effectively converted to online learning by setting $D_0 = D_s$ and separating all remaining points as multiple incoming datasets. *Online defence algorithm* can thus be applied to the offline model. Some existing defence algorithms require a "trusted" subset of data to be functional, despite being proposed in offline learning (e.g. RONI (Nelson *et al.*, 2009) and the anomaly detector in (Paudice *et al.*, 2018)). These algorithms belong to the group of *online defence algorithms*.

Although all *offline defence algorithms* can be directly applied in online learning settings without taking advantage of a clean D_0 (other than the benefit of not needing to remove points from D_0), slight modifications to some *offline defence algorithms* can be made to improve the performance of the defence algorithm in online learning.

Game analysis for online learning is particularly beneficial, as the modeller can perform the game analysis himself to derive the optimal defence strategy, prior to deploying the model. During the game analysis, the modeller can estimate for $E(S_a, \theta_d)$ and $\Gamma(\theta_d)$ by attacking his own model, and use the resulting optimal strategy against real attackers. Furthermore, in online learning, the attack-defence game is played repetitively. Hence the modeller will have incentive to minimize his expected loss instead of any one-time loss. Therefore we will focus on analyzing online defence algorithms using our game model.

4.2 L2 (Centroid) Defence

L2 defence is a straightforward offline defence algorithm. To sanitize dataset $D = D_T \cup D_c$, L2 defence computes the centroids of each class in D, then measure the Euclidean (L2) distance d_i of each datapoint i to its respective class centroid and remove points with $d_i > \theta_d$. L2 defence can be considered as fitting a hypersphere to datasets of each class centered at the centroid of the dataset, removing all datapoints that are outside of the hyperspheres. Analysis of L2 defence in (Kloft and Laskov, 2012; Steinhardt *et al.*, 2017) show that L2 defence is ineffective against poisoning attacks, because the position of the centroids can be significantly influenced by the malicious points D_c. Therefore, online L2 defence will benefit from the initial clean dataset D_0, which the defender can use to compute the clean centroids and use them to sanitize all incoming sets.

4.3 Loss Defence

Loss defence is an offline defence algorithm. In loss defence, a defence classifier w' is first trained using $D_T \cup D_c$. Then, every datapoint $p_i \in D_T \cup D_c$ with loss value $L(p_i, w') > \theta_d$ will be removed. Notice that defence algorithm relies on w', and in turn, relies on D_c. Therefore the malicious points have the opportunity shift the defence classifier w' before it is used to determine the loss value of the malicious points. In fact, authors in (Koh *et al.*, 2018) demonstrated that poisoning points that survive the filter in this manner can still significantly degrade performance of the ML model.

However, when loss defence is used in online learning, w' can be trained using the initial clean dataset D_0, to sanitize any incoming datasets D_i. The advantage of online loss defence is that, it prevents the attacker from altering the defence classifier w', compared to offline loss defence. The attacker's only remaining option to attack is to place malicious points with $L(p_i \in D_c, w') <= \theta_d$.

4.4 PCA Defence

4.4.1 Existing PCA Detection Algorithms

Existing PCA based defence mechanisms are used to detect abnormal network traffic (Lakhina *et al.*, 2004) and to mitigate evasion attacks (Bhagoji *et al.*, 2017). The success of PCA defence relies on the observation of that anomaly datapoints typically result a large spike on the low-variance principle component axis. To apply PCA defence, the principal components of the dataset must be computed. The subspace spanned by first K principal components capturing the most variance is the **normal subspace**, and the remaining $(N-K)$ low-variance components forms the **abnormal**

subspace. Then, any datapoint $\vec{x}_P$ in PCA space can be decomposed into $\vec{x}_P = \vec{x}_n + \vec{x}_a$, where $\vec{x}_n$ and $\vec{x}_a$ corresponds to the projection of $\vec{x}_P$ onto the **normal subspace** by and the **abnormal subspace** respectively. After that, the approach of (Lakhina *et al.*, 2004) and (Bhagoji *et al.*, 2017) differs slightly: (Lakhina *et al.*, 2004) chooses a threshold θ_d on the **abnormal subspace**, such that points with $||\vec{x}_a||^2 > \theta_d$ (Euclidean distance from the PCA origin) are identified as anomaly data; while (Bhagoji *et al.*, 2017) uses dimension reduction technique to remove the **abnormal subspace** spanned by low-variance components prior to training the ML model, such that the resulting model is robust against evasion attack in the direction of low-variance principal components.

Both methods does not have any defence on the **normal subspace**, which can be exploited by an informed attacker. Therefore PCA detector needs to be modified to become effective against poisoning attacks.

4.4.2 PCA for Poisoning Attack

PCA can be considered as fitting an ellipsoid to a target dataset, where each axis of the principal components corresponds to the direction that captures the most variance of the dataset. The ellipsoid can be used as the feasible domain of PCA defence algorithm, such that all PCA subspace are defended. The main advantage of embedding PCA into filtering mechanism is that, the filter can be more restrictive on the low-variance direction while being more receptive on the high-variance direction. Since the malicious points are out-of-distribution, they are likely to have abnormal statistical properties compared to genuine points, which PCA defence can take advantage of for anomaly detection. Notice that L2 defence is a special case of PCA defence, where the filtering strength is identical along all principal component directions.

To apply PCA filter, the first step is to perform PCA for datasets of each class. This will produce the eigenvectors $\vec{E} = [\vec{e_1}, \vec{e_2}, ..., \vec{e_N}]$ and its corresponding eigenvalues $\vec{\lambda} = [\lambda_1, \lambda_2, ..., \lambda_N]$ of each principal components. The eigenvectors indicate the directions of the principal component axis, while the eigenvalues represent the magnitude of the eigenvectors. Then, choose a multiplier η on the eigenvalues, such that the equation of the ellipsoid can be written as:

$$\frac{x_{1_p}^2}{(\eta\lambda_1)^2} + \frac{x_{2_p}^2}{(\eta\lambda_2)^2} + ... + \frac{x_{N_p}^2}{(\eta\lambda_N)^2} \leq 1 \tag{4.4.1}$$

Simplify the equation, we get

$$\vec{x_p}^{\circ 2} \cdot \vec{\lambda}^{\circ -2} \leq \eta^2 \tag{4.4.2}$$

where $\vec{x_p} = [x_{1_p}, x_{2_p}, ..., x_{N_p}]$ is a vector representing the coordinate of the **projection** of a datapoint $\vec{x}$ onto the PCA space, $\vec{x_p}^{\circ 2}$ and $\vec{\lambda}^{\circ -2}$ are Hadamard (element-wise) Products. In simple terms, $\vec{E}$ represents the axes of the ellipsoid, and $\eta * \vec{\lambda}$ represents the radii of the ellipsoid on each axis.

To check whether a datapoint $\vec{x}$ is malicious, we first acquire its projection on the PCA space $\vec{x}_{pca} = (\vec{E}^{-1}\vec{x}^T)^T$. Then, we simply substitute $\vec{x}_{pca}$ into Equation 4.4.2 and classify the point as

$$\vec{x} = \begin{cases} \text{malicious} & \vec{x}_{pca}^{\circ 2} \cdot \vec{\lambda}^{\circ -2} > \eta^2 \quad (4.4.3a) \\ \text{innocuous} & \vec{x}_{pca}^{\circ 2} \cdot \vec{\lambda}^{\circ -2} \leq \eta^2 \quad (4.4.3b) \end{cases}$$

4.5 Curie Defence

Curie defence (Laishram and Phoha, 2016) is also a distance-based defence algorithm. The intuition of the algorithm is that for any poisoning point $[\vec{x_p}, y_p]$, its feature values $\vec{x_p}$ are likely to follow the statistical distribution of feature values of a different class (other than y_p). This is true especially when Label Flipping attack is performed on the training dataset. Therefore after running a clustering algorithm to separate points from each class, the malicious points will be distinguishable as the minority with label y_p in the cluster. These points are assigned a high anomaly score, and points with anomaly score higher than the threshold θ_d will be removed.

The steps to run Curie is the followings:

1. Run clustering algorithm on the potentially contaminated training dataset (DB-SCAN is used in (Laishram and Phoha, 2016))

2. For each datapoint i, compute the sum of the squared Euclidean distance e_i to all other datapoints in the same cluster C, with one special rule:

    - If the two datapoints have different labels, an additional ω^2 distance is added to e_i (ω is also a parameter chosen by the defender)

3. Compute the Curie distance d_i as $d_i = e_i/N_C$, where N_C is the amount of datapoints in cluster C.

4. Choose a filtering threshold value θ_d, and remove all points with $d_i > \theta_d$.

In online learning, notice that when Curie sanitize an incoming dataset D_i, it has to rerun the clustering algorithm with $\bigcup_{j=0}^{i} D_j$ in order to compute the Curie distance. Therefore the result of clustering can still be altered by incoming poisoning points, despite having a clean D_0 at disposal. This implies that Curie defence is unable to take advantage of a initial clean dataset in online learning.

4.5.1 Optimal Attack against Curie

If the attacker is aware of Curie, there are two optimal methods to poison the training set. The first method is to concentrate the malicious points around a certain region, such that the malicious points form their own cluster during the clustering process of Curie(Koh *et al.*, 2018). We will call this the **aggressive** attack. Since malicious points can be placed with small Euclidean distance to each other and all have identical labels (hence not penalized by ω^2), even the extremely outlying poisoning points will be able to subvert Curie. In this case, Curie defence will be unable to distinguish malicious points from genuine points, regardless of the choice of defence parameters ω and θ. Hence under **aggressive** attack, Curie cannot be improved via game theoretic analysis, which goal is to identify the best choice of ω and θ_d.

The other attack method is that: the attacker will not intentionally form poison clusters. Instead, he poison the model with malicious points that have a Curie-score $\leq \theta_d$ to one of the existing clusters. These poisoning points can still subvert Curie, but may have smaller impact on the ML model compared to *aggressive* attacks. We will call this the **sneaky** attack. The defender can in turn adjust ω and θ_d to counteract the attacker, leading to a competitive game. In this scenario, it is of the defender's interest to identify the NE defence strategy, which yields the best performance when the attacker is playing optimally.

We will perform game analysis on Curie, assuming that the attacker performs the **sneaky** attack. We will be using the attack algorithm in (Steinhardt *et al.*, 2017), which place and move malicious points of target class T to maximize classification error. And when malicious point has Curie distance $> \theta_d$, then the point will be projected onto the boundary of the filter with Curie distance $= \theta_d$. The detailed projection rule for Curie will be described in the Section 4.5.2

4.5.2 Projection Rule for Attacking Curie

Before describing the details of the projection rules, we provide some necessary background knowledge of statistics:

Theorem 1. *The centroid (mean) $\vec{m}$ of a cluster C has the minimum sum of the squared Euclidean distance $e_{\vec{m}}$ to all other points in C, with $e_{\vec{m}} = N_C * Var(C)$, where N_C is the amount of points in C , and $Var(C)$ is the variance of C.*

Theorem 2. *For any datapoint $\vec{p} = \vec{m} + \vec{\epsilon}$ in cluster C with centroid $\vec{m}$, the sum of its squared Euclidean distance $e_{\vec{p}} = e_{\vec{m}} + N_c * \sum_{i=0}^{f} \epsilon_i^2$, where f is the number of elements (feature values) in $\vec{\epsilon}$.*

Based on the properties of the centroid in Theorem. 1 and 2, we derive the constraints of the attack on Curie:

Theorem 3. *Malicious points of class T placed in cluster C incurs at least $\frac{n_d}{N_c}\omega^2 + Var(C)$ Curie distance, where n_d is the amount of points in C which label is different from T.*

Proof. By placing malicious points at the centroid of the cluster, the **average** squared Euclidean distance is minimized with $d_{\vec{m}} = Var(C)$ (Theorem 1). An additional $\frac{n_d}{N_c}\omega^2$ distance is added to the malicious point for having different class labels. □

Combining Theorem. 1, 2 and 3, we get the next corollary.

Corollary 3.1. *To avoid detection, malicious points $\vec{p_m} = \vec{m} + \vec{\epsilon}$ of class T placed in cluster C must satisfy* $\sum_{i=0}^{f} \epsilon_i \leq \sqrt{\theta_d - (\frac{n_d}{N_c}\omega^2 + Var(C))}$

By identifying a projection centroid $\vec{m_p}$ and restricting the value of $\sum_{i=0}^{f} \epsilon_i$ in Corollary. 3.1, we can conveniently use the spherical projection method in (Steinhardt *et al.*, 2017) and (Koh *et al.*, 2018) to develop projection rule for attacking Curie. To ensure that all malicious points have Curie distance $\leq \theta_d$, the attacker inspects all clusters that contain points opposite to class T, and determine the projection centroid $\vec{m}$ as the centroid of the cluster with the largest number of points, and projection variance $Var(C)$ as the largest variance amoung these clusters. Notice that this method only guarantee the malicious points to subvert the filter while maximizing classification error, it may not prevent the malicious points from creating its own cluster (like the **aggressive** attack). Furthermore, if $\theta_d < (\frac{n_d}{N_c}\omega^2 + Var(C))$, then the attack algorithm will assume that no malicious points can subvert the filter, and therefore does not perform the attack.

Chapter 5

Experimental Evaluation of the Game Model

In this chapter, we present the experimental results of online defence algorithms against optimal poisoning attacks. We also include some additional experiments performed in offline defence scenario.

5.1 Online Defence Experiments

The goal of the experiments is to evaluate the performance of L2 defence, loss defence, PCA defence and Curie against an informed attacker in online learning. To compute the optimal defence strategy, $E(S_a, \theta_d)$ and $\Gamma(\theta_d)$ need to be calculated experimentally first. Therefore our experiments consist of two stages: (1) perform pure strategy attack-defence experiments for different values of R and θ_d, and compute $E(S_a, \theta_d)$ and $\Gamma(\theta_d)$ using the results; (2) run Algorithm. 1 to acquire the NE defence strategy, and assess the performance of the defence algorithm when the NE defence strategy is used.

We use the Spambase dataset (Dua and Graff, 2017) and a sample of 50% datapoints from MNIST-17 (MNIST data with only written digits of ”1”s and ”7”s) (LeCun, 2017) to train and test our ML models. We separate Spambase and MNIST-17 into 30% of test data (1381 instances for Spambase, 2163 for MNIST-17), 21% of day-zero training data D_0 (clean initial dataset) (676 instances for Spambase, 1366 for MNIST-17) and 49% of incoming data over the course of 10 days (2254 instances for Spambase, 4552 for MNIST-17 in total). The attacker can manipulate 20% of the entire training data (644 instances for Spambase, 1300 for MNIST-17) in total over the course of 10 days (injects 2% each day). We use a Support Vector Machine (SVM) with hinge loss as our ML model and trained the model for 5000 epoch in every training iteration.

In online L2 defence, the class centroids of D_0 are computed and used to measure the L2 distance of all incoming datapoints. In loss defence, the defence classifier w' is trained using D_0, and used to sanitize each incoming dataset. For PCA defence, the result of PCA for D_0 is used to construct the elliptical feasible domain, which determines the datapoints to retain from all incoming sets. Notice that all three defence algorithm are not affected by malicious points, therefore injecting malicious points over 10 days is essentially the same as injecting all points in one day. Hence we report the results for 1-day attack instead for L2, loss and PCA defence, treating all poisoned incoming datasets as a single incoming set.

For Curie defence, the defender uses D_0 to determine the value of ω and θ_d, and uses it to sanitize the incoming datasets. We start by computing the maximum intra-cluster distance icd_{max}, and set $\omega = \sqrt{icd_{max} * 4}$, so that ω^2 contributes at least 80% of Curie distance to all datapoints. At the end of each day, the sanitized dataset is added to the training dataset, and the defence algorithm is updated with the new training set. Such update is inevitable for Curie, as explained in Section 4.5. For PCA and Curie, we perform the experiment on different percentile value of P, which determines the value of θ_d in the following way: denote $Dist_C$ as the distance of each datapoint sorted in ascending order, N_0 the number of points in D_0, then $\theta_d = Dist_C[(100 - P) * N_0])$.

The performance of the defence algorithm is measured with two factors: 1)the accuracy loss incurred from the attack $E(S_a, \theta_d)$, and 2) the amount of genuine points rejected. When computing the optimal defence strategy, modellers need to choose an **aggregated** accuracy/loss metrics that accounts for both factors. In our experiments, we let $\Gamma(\theta_d)$ contribute 0.2% classification error for each percent of genuine points removed. Thus $E(S_a, \theta_d) + \Gamma(\theta_d)$ is the **aggregated** loss in our experiments.

The attacker computes the optimal attack strategy against the defence algorithm, based on the defender's strategy. In pure strategy attack-defence, the attacker will coordinate the poisoning points in S_a such that $R \leq \theta_d$ (none of the malicious points will be removed by the filter) while maximizing classification error of the model. The results for L2 defence, loss defence, PCA defence and Curie are reported in Fig. 5.1, Fig. 5.2, Fig. 5.3, Fig. 5.4, Fig. 5.5, Fig. 5.6, Fig. 5.7 and Fig. 5.8 respectively. For Curie defence, $\theta_d < (\frac{n_d}{N_c}\omega^2 + Var(C))$ occurred at 40 percentile in Spambase and at 10 percentile in MNIST-17, hence no successful attack is performed after 30 percentile in Spambase and after 0 percentile in MNIST-17. The changes in accuracy of the **attacked** model over the 10-day interval are shown in Fig.5.9.

As we can see from Fig. 5.1 and Fig. 5.2, online L2 defence is not very effective

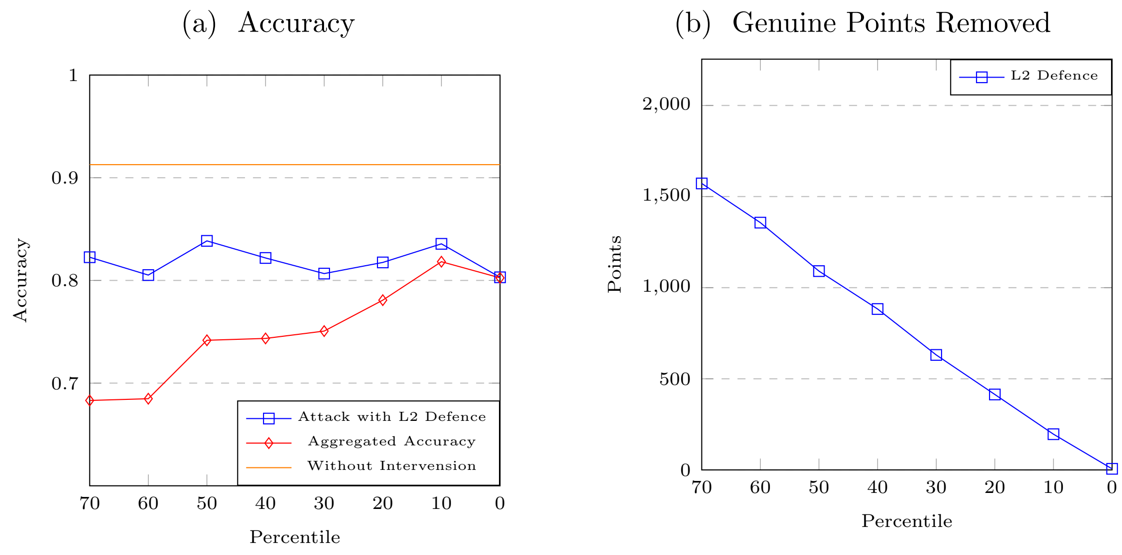


Figure 5.1: Performance of L2 Defence Under Optimal Attack - Spambase

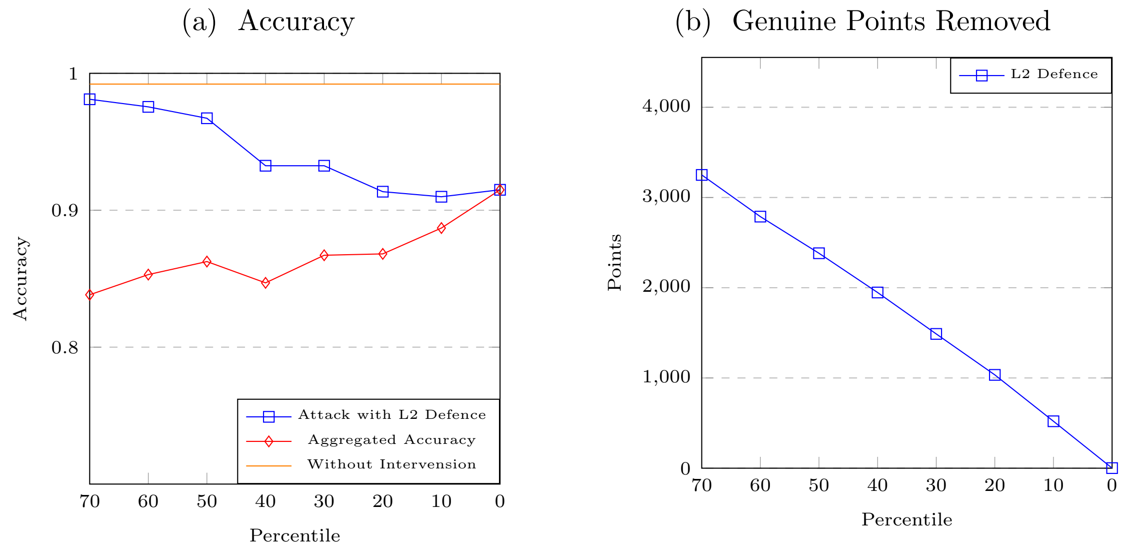


Figure 5.2: Performance of L2 Defence Under Optimal Attack - MNIST-17

against informed poisoning attack, even when the centroids are not affected by the malicious points. For Spambase dataset (Fig. 5.1a), using a stricter filter does not prevent accuracy from going down. This is because the size of the initial clean dataset is small in our experiments, rejecting a large amount of incoming genuine points can significantly increase the proportion of the remaining malicious points within the sanitized dataset, hence the poisoning points can have a large impact on the model, despite the restrictions on the L2 distance. L2 defence is able to thwart poisoning attack in the MNIST-17 dataset with a strict filter (Fig. 5.2a), retaining the accuracy of the ML model after 50 percentile. However, more than 50% of genuine points are rejected during this process (Fig. 5.2b), such loss is deemed to outweigh the gain mitigating poisoning attack, as reflected in the **aggregated** accuracy (red line) in Fig. 5.2a.

Notice that our experiment for L2 defence is similar to the experiment in (Steinhardt *et al.*, 2017), which demonstrated that if the defender can acquire the centroid of the **uncontaminated** dataset, L2 defence is effective against poisoning attack for the MNIST-17 dataset, a contradiction to our findings. However, experiments in (Steinhardt *et al.*, 2017) focuses on analyzing the loss incurred by the attacker given the constraints on the L2 distance, therefore assuming that no genuine points are removed during the filtering process. In fact, our results demonstrate that when taking the removal of genuine points into consideration, online L2 defence may not be desirable in terms of retaining both the classification accuracy the incoming genuine points.

In online loss defence, from Fig. 5.3a and Fig. 5.4a we can see that malicious points with high loss value can significantly degrade the classification accuracy of the ML model. However, as the defender restricts the maximum loss value of malicious points by reducing θ_d, the impact of the attack decreases drastically, becoming insignificant at $\theta_d = 1$. Furthermore, the amount of genuine points are removed during

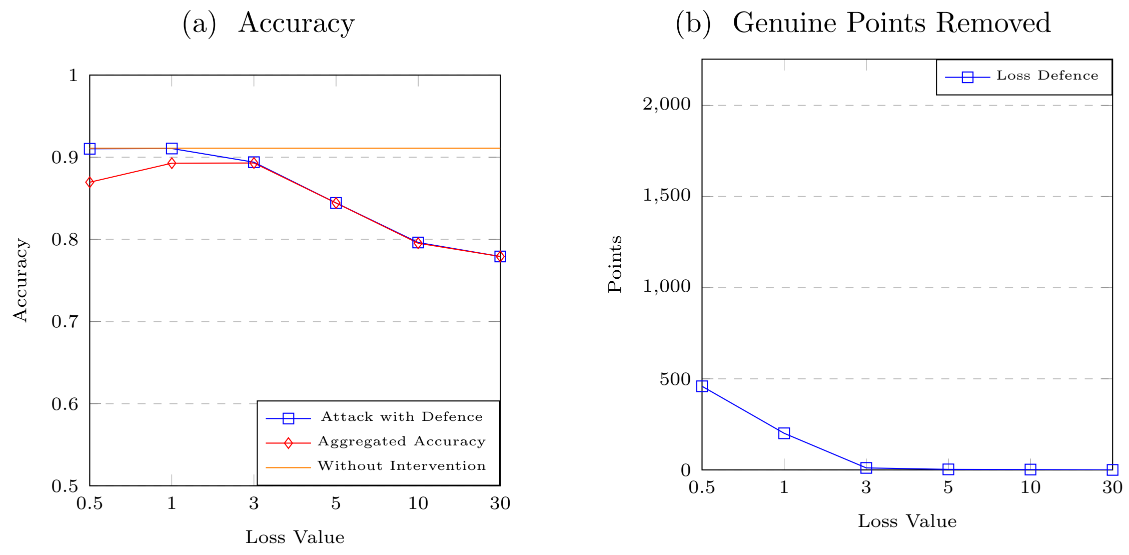


Figure 5.3: Performance of Loss Defence Under Optimal Attack - Spambase

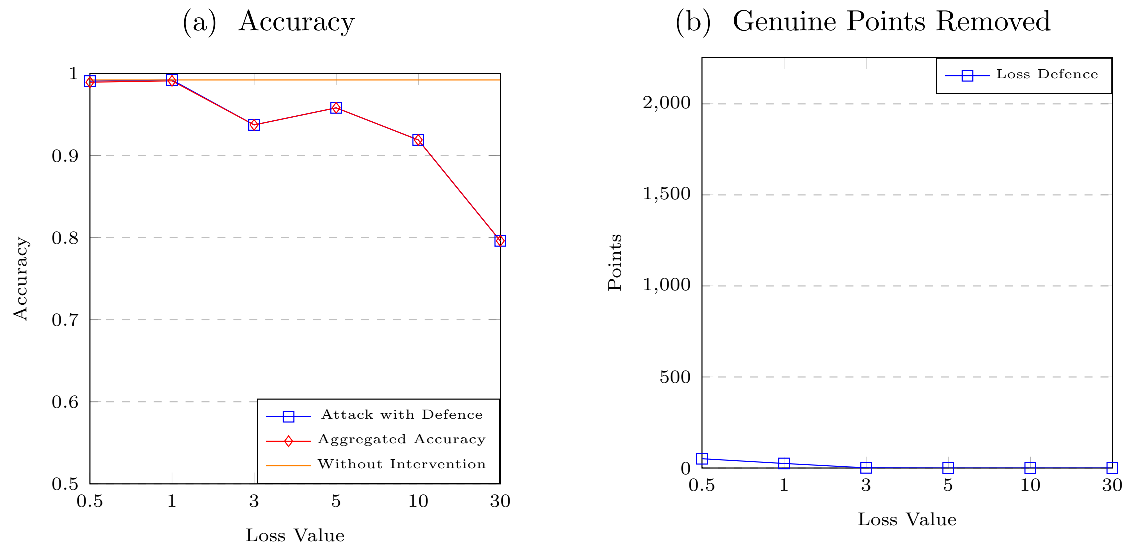


Figure 5.4: Performance of Loss Defence Under Optimal Attack - MNIST-17

the sanitization process is 9% for Spambase, and 0.6% for MNIST-17, as shown in Fig. 5.3b and Fig. 5.4b. Our results demonstrates that loss defence can mitigate poisoning attacks while removing a small amount of genuine points, and therefore is

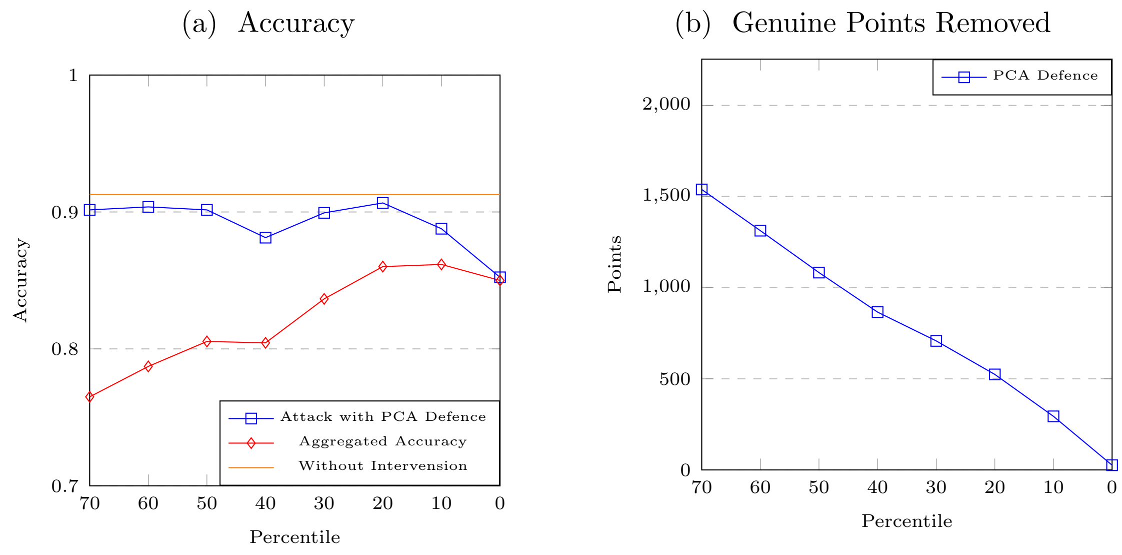


Figure 5.5: Performance of PCA Defence Under Optimal Attack - Spambase

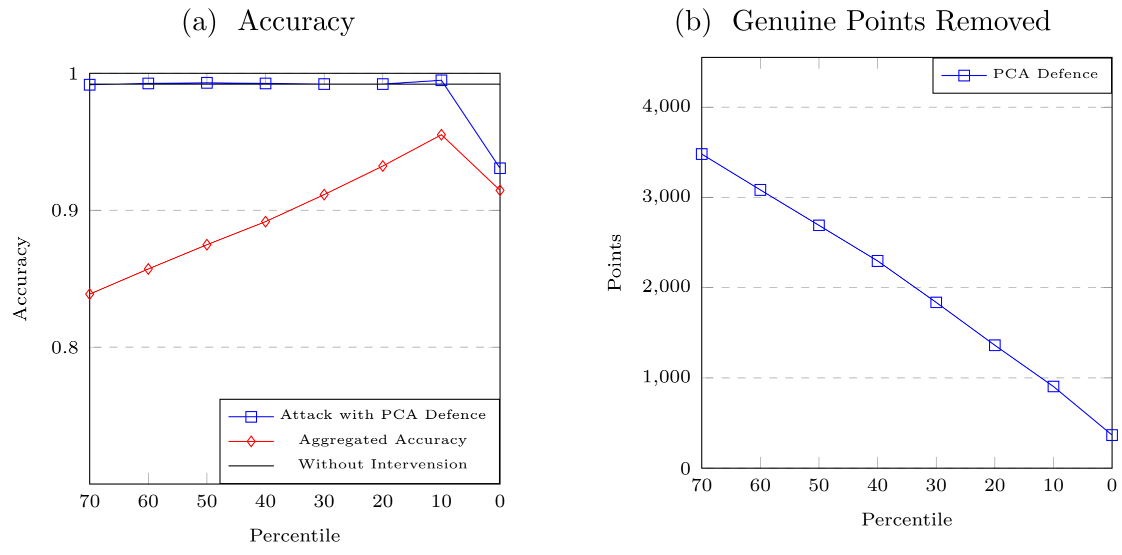


Figure 5.6: Performance of PCA Defence Under Optimal Attack - MNIST-17

an effective defence algorithm in online learning. Online PCA defence is also effective against poisoning attacks on both datasets. As we can see from Fig. 5.5a and Fig. 5.6a, a 10-20 percentile filter is sufficient to minimize the accuracy loss incurred by the attacker, while losing around 15-20% genuine points for both datasets (Fig. 5.5b and Fig. 5.6b).

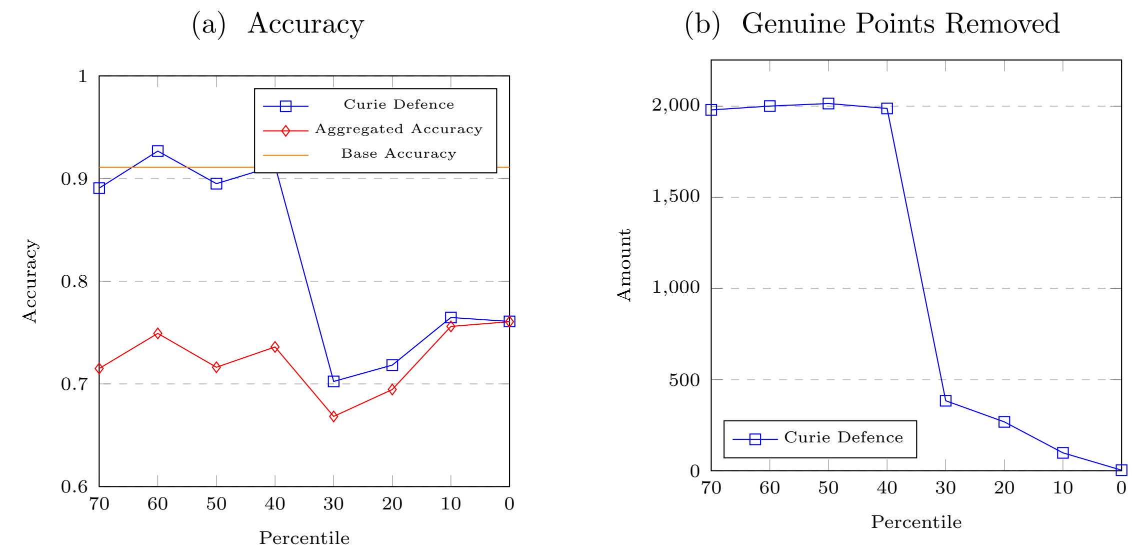


Figure 5.7: Performance of Curie Defence Under Optimal Attack - Spambase

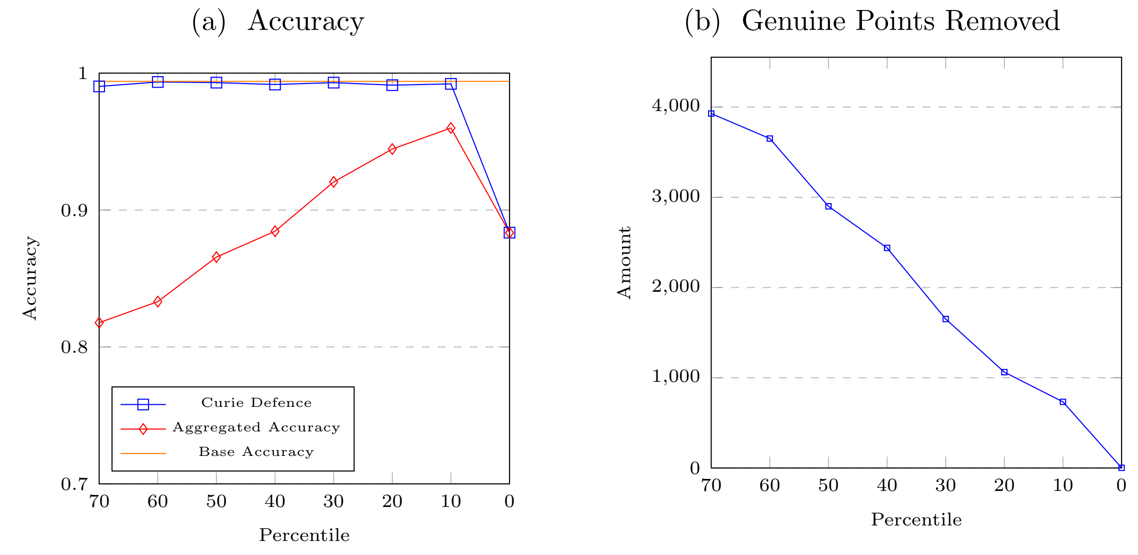


Figure 5.8: Performance of Curie Defence Under Optimal Attack - MNIST-17

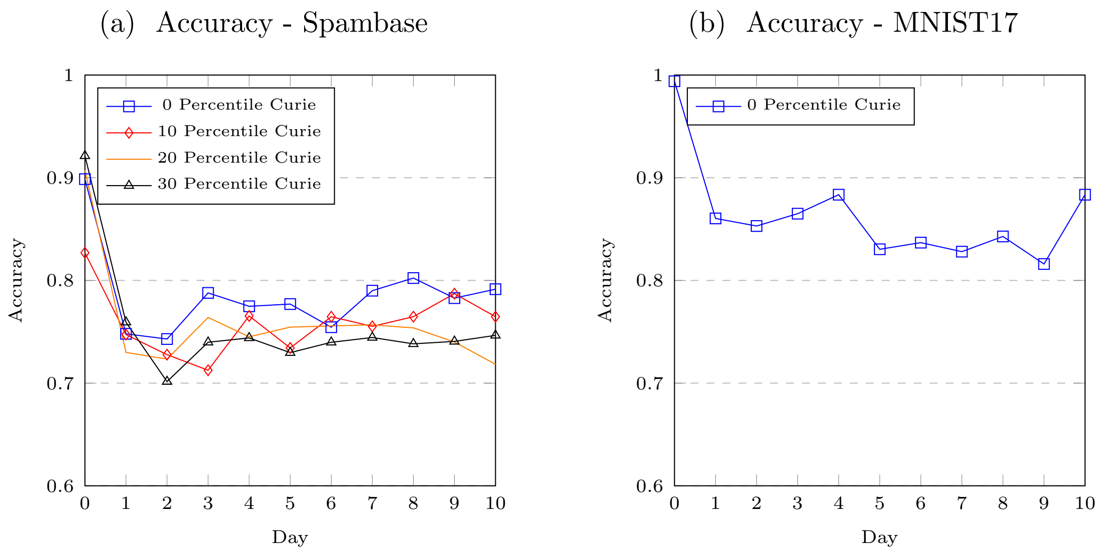
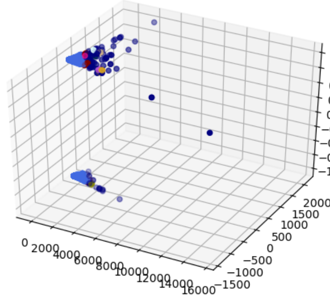


Figure 5.9: Overtime accuracy impact on ML model under attack

As for Curie defence, we can see that an informed attacker can easily subvert Curie defence, incurring high accuracy loss before 30 percentile for Spambase, and 10 percentile for MNIST-17 from Fig.5.7a and Fig. 5.8a. Furthermore, from Fig. 5.9 we can see that most accuracy is lost on the first iteration of attack instead of degrading overtime. The same observations can be made for MNIST-17 (from Fig. 5.8a and Fig. 5.9b), before 10 percentile. This demonstrates that Curie defence is ineffective when $\theta_d \geq (\frac{n_d}{N_c}\omega^2 + Var(C))$. At the price of rejecting a large amount of genuine points(as shown in Fig. 5.7b, after 40 percentile), the defender can set $\theta_d < (\frac{n_d}{N_c}\omega^2 + Var(C))$ to mitigate all **sneaky** poisoning attack in Spambase. In fact, 88% of the incoming genuine datapoints are removed by a 40 percentile Curie filter. However, for MNIST-17, Curie only remove a small amount of genuine points while mitigating the attack. The difference in terms of performance can be explained by the data distribution of the two datasets in Fig. 5.10.

In Spambase (Fig. 5.10a), datapoints in the $+1$ class and -1 have similar values in

(a) Spambase



(b) MNIST-17

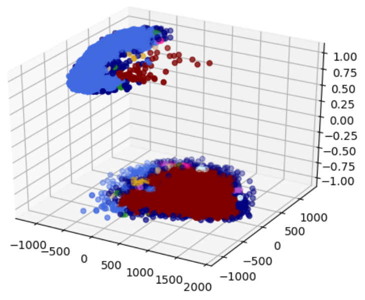


Figure 5.10: Clustering results of clean Spambase and MNIST-17 datasets. The x and z axis represents the first two principal components of the dataset, and the y axis indicates the labels of datapoints.

the first two principal components, hence the clustering algorithm groups the majority of points from both class into a single cluster. As a result, all incoming datapoints with feature values following the same distribution will inevitably suffer the ω^2 penalty in Curie distance, regardless of the class label of the datapoint. Therefore Curie will fail to distinguish malicious points from genuine points, and is incapable of defending the Spambase ML model. For MNIST-17 dataset (Fig. 5.10b), distribution of feature values of the $+1$ class data is nicely separated from the distribution of the -1 class. Hence incoming genuine points are much less likely to suffer the ω^2 penalty, while malicious points that follows the distribution of the opposite class are much more identifiable. Therefore Curie is expected to perform well against **sneaky** poisoning attacks.

In summary, our result demonstrates that the effectiveness of Curie defence is dependant on the distribution of the dataset, in contrast to the optimistic results in (Laishram and Phoha, 2016). Therefore modellers will need to examine the data distribution in order to determine the validity of applying Curie defence.

Using the findings from above, we compute the optimal defence strategy for each defence algorithm and compare its performance with pure strategy defences. Notice that the maxima of the **aggregated** accuracy plots correspond to the performance of the optimal pure strategy. We first make a continuous approximation of $E(S_a, \theta_d)$ and $\Gamma(\theta_d)$ with the experiment results, using non-linear regression analysis, then run Algorithm. 1 with $E(S_a, \theta_d)$ and $\Gamma(\theta_d)$ to acquire the NE defence strategy. We test the resulting defence strategy against poisoning attack, in which the attacker also computes the optimal attack strategy against the chosen defence strategy. The results are reported in Table. 5.1, Table. 5.2, Table. 5.3 and Table. 5.4. For loss defence, the algorithm produces the pure strategy $\theta_d = 1$ as the optimal defence strategy for both Spambase and MNIST-17. Curie is shown to be incapable to defend the Spambase dataset regardless of the defence strategy, therefore we omit its mixed strategy analysis on Spambase dataset. As for Curie on MNIST-17 datatset, the result shows that the 10 percentile filter is optimal. As shown in the tables, the **aggregated** accuracy of using the optimal mixed strategy defence is strictly higher than the **aggregated** accuracy of the best pure strategy. This validates the effectiveness of the optimal defence strategy generated through game analysis. Notice that the optimal strategy only guarantees higher **aggregated** accuracy, it does not have to guarantee better performance in reducing accuracy degradation $E(R, \theta_d)$ and retaining more genuine points $\Gamma(\theta_d)$ **simultaneously**.

Dataset & Strategy	Spambase Mixed			Best Spambase Pure
Percentile	0.05%	4.6%	8.1%	10.0%
Probability	86.6%	12.1%	1.3%	100%
Expected Aggregated Accuracy	82.6%			81.8%
Expected Accuracy	82.7%			83.5%
Expected % of Genuine Points Removed	0.89%			8.7%

Table 5.1: Mixed strategy L2 defence under optimal attack - Spambase

Dataset & Strategy	MNIST-17 Mixed			Best Mnist-17 Pure
Percentile	0.0%	26.5%	30.7%	0.0%
Probability	83.8%	7.4%	8.8%	100%
Expected Aggregated Accuracy	91.8%			91.5%
Expected Accuracy	92.8%			91.5%
Expected % of Genuine Points Removed	10.6%			0.02%

Table 5.2: Mixed strategy L2 defence under optimal attack - MNIST-17

Dataset & Strategy	Spambase Mixed			Best Spambase Pure
Percentile	0.4%	3.3%	7.1%	10.0%
Probability	43.9%	29.7%	26.4%	100%
Expected Aggregated Accuracy	86.3 %			86.2%
Expected Accuracy	87.3%			88.8%
Expected % of Genuine Points Removed	5.0%			8.7%

Table 5.3: Mixed strategy PCA defence under optimal attack - Spambase

Dataset & Strategy	MNIST-17 Mixed			Best Mnist-17 Pure
Percentile	1.3%	2.5%	3.4%	10.0%
Probability	23.4%	33.9%	42.7%	100%
Expected Aggregated Accuracy	96.0 %			95.5%
Expected Accuracy	98.9%			99.5%
Expected % of Genuine Points Removed	14.6%			19.9%

Table 5.4: Mixed strategy PCA defence under optimal attack - MNIST-17

5.2 Additional Experiments for Offline Defence

We also experimented offline L2 defence on Spambase dataset, and PCA defence on both Spambase and MNIST-17 dataset. We separate Spambase dataset into 70%

of training data (3220 instances) and 30% of test data (1381 instances), MNIST-17 have separated training dataset (13007 instances) and test dataset (2163 instances) before processing. The rest of the experimental setup is identical to the experiments in Section 5.1. The accuracy of the ML model are reported in Fig. 5.11 and Fig. 5.12 [1]. Notice that since malicious points are coordinated so that none will be removed by the filter, only genuine points can be removed during the filtering process (a 20 percentile filter will always remove 20% genuine points).

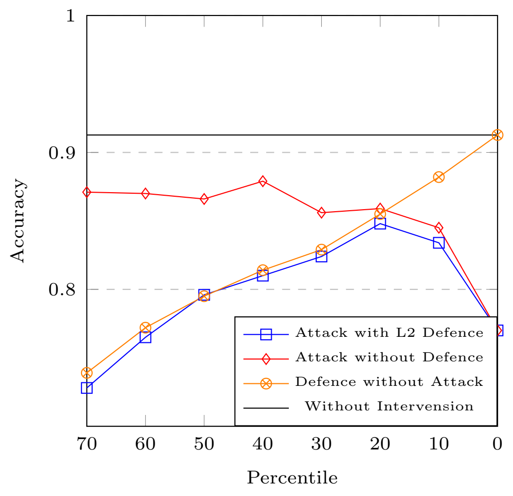


Figure 5.11: Accuracy of pure strategy L2 under optimal attack

As shown in Fig. 5.11, applying L2 filter reduces the accuracy of the ML model, regardless of the presence of the attack. However, aborting the filter will enable the attacker to perform more threatening attacks. Also, we can see that for this model, the defender loses incentive to increase filter strength at some point around 20 percentile, as filtering with more than 20 percentile yields strictly less accuracy than filtering with 20 percentile, while the attacker always have incentive to inject,

[1]Offline PCA defence experiment is run 10 times and the medians are reported, see Appendix A for box and whisker plots.

regardless of the percentile, as malicious points are able to degrade accuracy for all percentile. These facts indicate that no pure strategy NE exists for offline L2 defence in Spambase dataset.

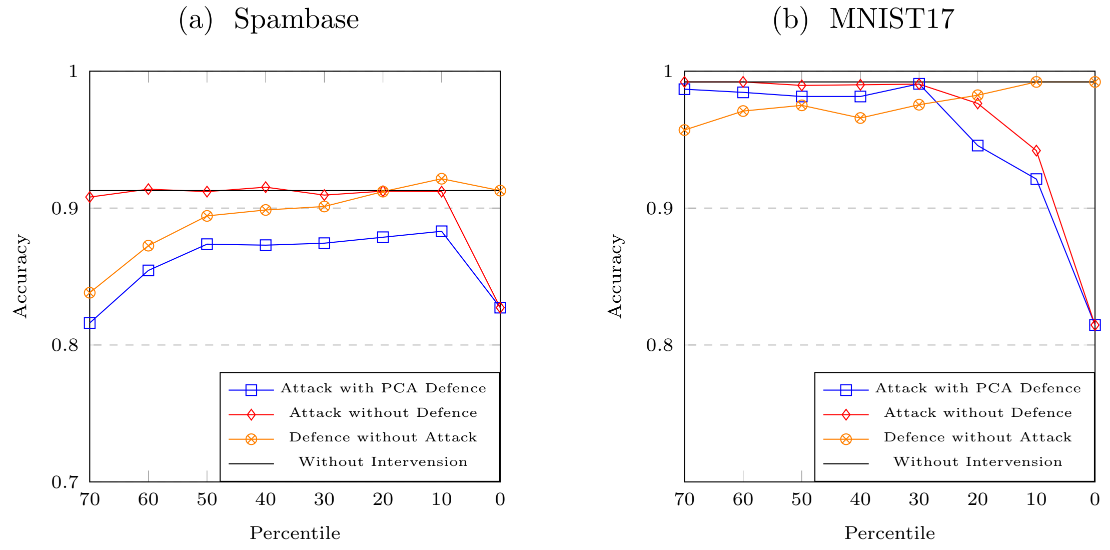


Figure 5.12: Accuracy of pure strategy PCA under optimal attack

As for offline PCA defence (Fig. 5.12), by comparing the accuracy difference between the unaffected model (black line) and the defended model without attack (the orange line), we can see that surprisingly PCA defence does not degrade classification accuracy by a lot, even when large potion of datapoints are removed. However, as we can see from the defended model under attack (blue line) in Spambase dataset (Figure 5.12a), when the model is under attack, filtering large amount of datapoints significantly degrades accuracy. This is because when a large quantity of genuine points are removed, the malicious points will start to outnumber the genuine points and cause high accuracy loss. The difference of accuracy between the defended model without attack (orange line) and the defended model under attack (blue line) indicates that the attacker always have incentive to inject in Spambase, while losing incentive

Dataset & Strategy	Spambase Mixed			Best Spambase Pure
Radius	5.8%	9.4%	16.3%	20.0%
Probability	33.3%	33.3%	33.4%	100%
Accuracy	86.1%			84.8%

Table 5.5: Mixed strategy offline L2 defence under optimal attack - Spambase

at around 30 percentile in MNIST17 (where attacker's gain is insignificant in Figure 5.12b). In Spambase the accuracy of the defended model under attack (blue line) is maximum from 10-50 percentile and drastically decrease after 50 percentile, showing that the defender would certainly lose incentive after 50 percentile. In MNIST-17 the accuracy of the defended model under attack (blue line) never drastically decreases after 30 percentile, showing that the earliest point of losing defending incentive is at 30 percentile. The results indicate that a pure strategy NE exists for PCA defence in MNIST17, and does not exist for Spambase.

We then ran Algorithm 1 to generate the mixed defence strategies. The input of the algorithm, $E(p)$ and $\Gamma(p)$, are approximated using the results in Fig. 5.11 and Fig.5.12. We tested the accuracy of the ML model when the generated defence strategy is used. The results of the mixed strategy defence are reported in Table. 5.5 and Table. 5.6. For PCA defence on MNIST-17 dataset, the algorithm produces a pure strategy of 30 percentile as the optimal strategy.

As shown in Table. 5.5 and Table. 5.6, the accuracy of the ML model using mixed defence strategy is strictly higher than the accuracy of all pure defense strategies. This further validates the effectiveness of mixed strategy defence in poisoning attacks, even in the context of offline defence.

Dataset & Strategy	Spambase Mixed				Best Spambase Pure
Radius	3.0%	15.4%	28.4%	40.8%	10.0%
Probability	42.6%	26.6%	18.4%	12.4%	100%
Accuracy	88.8%				88.3%

Table 5.6: Mixed strategy offline PCA defence under optimal attack - Spambase

Chapter 6

Conclusions & Future Work

In this work, we performed game theoretic analysis in the poisoning attack and defence scenario. We proposed the game model which simulates the strategic interaction between the attacker and the defender, and identified a key property of the NE defence strategy. Using the property of the NE strategy, we developed an efficient algorithm to approximate for the NE defence strategy. Finally, we experimentally evaluated the performance of the NE defence strategy for several distance-based defence algorithms. Compared to the state-of-the-art analysis on poisoning attacks and countermeasures, which are either over-optimistic or over-pessimistic, our game analysis reveals the true performance of the analyzed algorithms under optimal poisoning attacks. Furthermore, our methodology can be replicated by any online modellers to determine the optimal defence strategies to employ on their ML models. One limitation in our work is that, although finding the optimal defence strategy via game analysis can enhance the performance of defence algorithms, the optimal strategy may not be useful if the defence strategy itself is incapable of defending the dataset. For example, consider a dataset which consists of images of cats and images without cats. The distribution of the genuine "no-cat" image are likely scattered and have large variance along most feature spaces, making it difficult for L2 and PCA defence to detect malicious "no-cat" samples. Therefore, modellers must first identify a capable defence algorithm for their dataset before solving for an optimal defence strategy. Furthermore, the attacker's objective function is assumed to be known by the defender in our game analysis. Hence our game model does not accurately simulate the attack-defence scenarios with unknown attacker's objective functions.

There are several future research avenues. For instance, effectiveness of *combinational* defence algorithm, which is formed by combining two or more defence (e.g. combining L2 defence and slab defence (Steinhardt *et al.*, 2017)), can be analyzed against optimal attacks. Also, as mentioned in Section 3.3, the non-zero-sum condition of the game can be analyzed. It may be challenging to find a good approximation on the true attacker's objective, which can be chosen to be any arbitrary function by the attacker. However, the existence of effective defence algorithms against Trojaning attack (Liu *et al.*, 2018a) implies that the approximation is possible. Hence extending our game model to include the unknown objective scenario will be an interesting area of research. Optimal defence strategies for non-distance based defence algorithms can be analyzed using game theories in future works. Finally, defence mechanisms against targeted attacks using optimal control (Lessard *et al.*, 2018) can be analyzed using game theories.

Appendix A

Box and Whisker Plots for Offline Defence Experiments

The box and whisker plots for offline PCA defence analysis are included in this appendix.

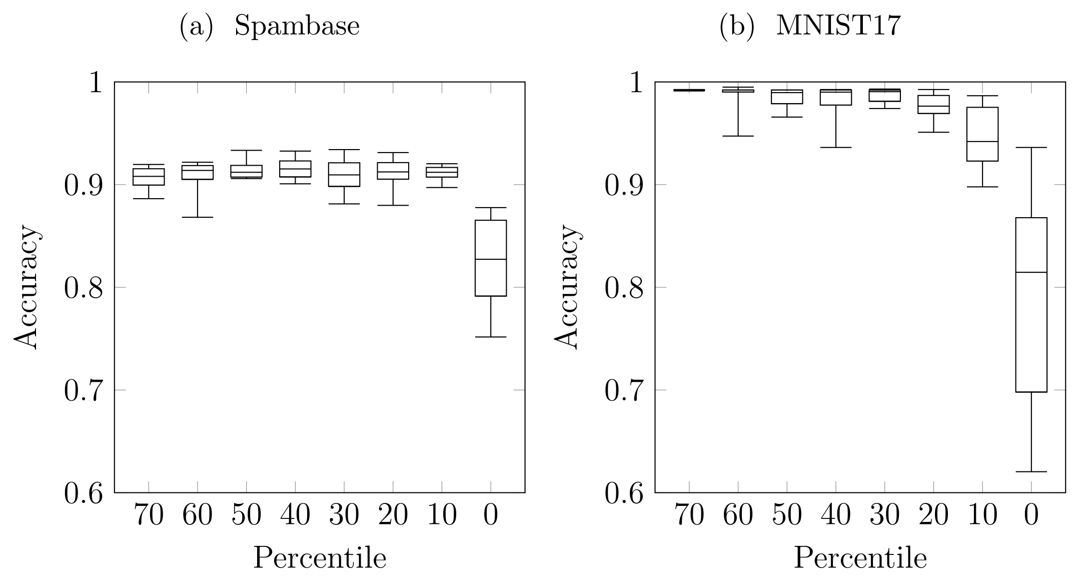


Figure A.1: Box and Whisker Chart for PCA-Aware Attack on Undefended Model, used in figure 5.12a

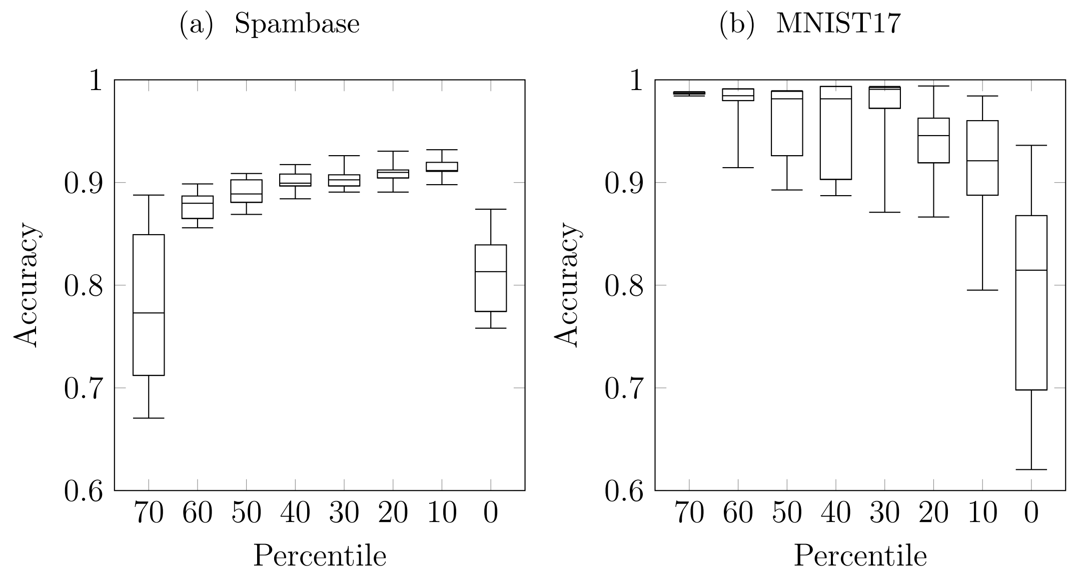


Figure A.2: Box and Whisker Chart for PCA-Aware Attack on Defended Model, used in figure 5.12b

Bibliography

Bhagoji, A. N., Cullina, D., and Mittal, P. (2017). Dimensionality reduction as a defense against evasion attacks on machine learning classifiers. *arXiv preprint arXiv:1704.02654*.

Biggio, B., Nelson, B., and Laskov, P. (2012). Poisoning attacks against support vector machines. *Computer Science*, pages 1467–1474.

Brückner, M. and Scheffer, T. (2011). Stackelberg games for adversarial prediction problems. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 547–555.

Brückner, M., Kanzow, C., and Scheffer, T. (2012). Static prediction games for adversarial learning problems. *Journal of Machine Learning Research*, **13**(Sep), 2617–2654.

Dua, D. and Graff, C. (2017). UCI machine learning repository - Spambase Dataset. https://archive.ics.uci.edu/ml/machine-learning-databases/spambase/spambase.data.

Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T., and Song, D. (2017). Robust physical-world attacks on deep learning models.

Farokhi, F. (2020). Regularization helps with mitigating poisoning attacks: Distributionally-robust machine learning using the wasserstein distance. *arXiv preprint arXiv:2001.10655*.

Glicksberg, I. L. (1952). A further generalization of the kakutani fixed point theorem, with application to nash equilibrium points. *Proceedings of the American Mathematical Society*, **3**(1), 170–174.

Huang, L., Joseph, A. D., Nelson, B., Rubinstein, B. I., and Tygar, J. D. (2011). Adversarial machine learning. In *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence*, AISec '11, pages 43–58, New York, NY, USA. ACM.

Jagielski, M., Oprea, A., Biggio, B., Liu, C., Nita-Rotaru, C., and Li, B. (2018). Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 19–35. IEEE.

Joseph, A. D., Laskov, P., Roli, F., Tygar, J. D., and Nelson, B. (2013). Machine learning methods for computer security (dagstuhl perspectives workshop 12371). In *Dagstuhl Manifestos*, volume 3. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Kloft, M. and Laskov, P. (2012). Security analysis of online centroid anomaly detection. *Journal of Machine Learning Research*, **13**(Dec), 3681–3724.

Koh, P. W. and Liang, P. (2017). Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1885–1894. JMLR. org.

Koh, P. W., Steinhardt, J., and Liang, P. (2018). Stronger data poisoning attacks break data sanitization defenses. *arXiv preprint arXiv:1811.00741*.

Laishram, R. and Phoha, V. V. (2016). Curie: A method for protecting svm classifier from poisoning attack. *arXiv preprint arXiv:1606.01584*.

Lakhina, A., Crovella, M., and Diot, C. (2004). Diagnosing network-wide traffic anomalies. In *ACM SIGCOMM computer communication review*, volume 34, pages 219–230. ACM.

LeCun, Y. (2017). The MNIST database of handwritten digits. http://yann.lecun.com/exdb/mnist/.

Lessard, L., Zhang, X., and Zhu, X. (2018). An optimal control approach to sequential machine teaching. *arXiv preprint arXiv:1810.06175*.

Lin, Y.-C., Hong, Z.-W., Liao, Y.-H., Shih, M.-L., Liu, M.-Y., and Sun, M. (2017). Tactics of adversarial attack on deep reinforcement learning agents. *arXiv preprint arXiv:1703.06748*.

Liu, K., Dolan-Gavitt, B., and Garg, S. (2018a). Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pages 273–294. Springer.

Liu, W. and Chawla, S. (2009). A game theoretical model for adversarial learning.

In *2009 IEEE International Conference on Data Mining Workshops*, pages 25–30. IEEE.

Liu, Y., Ma, S., Aafer, Y., Lee, W. C., and Zhang, X. (2018b). Trojaning attack on neural networks. In *Network and Distributed System Security Symposium*.

Ma, Y., Jun, K.-S., Li, L., and Zhu, X. (2018). Data poisoning attacks in contextual bandits. In *International Conference on Decision and Game Theory for Security*, pages 186–204. Springer.

Mei, S. and Zhu, X. (2015). Using machine teaching to identify optimal training-set attacks on machine learners. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Muñoz González, L., Biggio, B., Demontis, A., Paudice, A., Wongrassamee, V., Lupu, E. C., and Roli, F. (2017). Towards poisoning of deep learning algorithms with back-gradient optimization. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, AISec '17, pages 27–38, New York, NY, USA. ACM.

Nash, J. F. (1950). Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, **36**(1), 48–49.

Nelson, B., Barreno, M., Jack Chi, F., Joseph, A. D., Rubinstein, B. I. P., Saini, U., Sutton, C., Tygar, J. D., and Xia, K. (2009). *Misleading Learners: Co-opting Your Spam Filter*, pages 17–51. Springer US, Boston, MA.

Ou, Y. and Samavi, R. (2019). Mixed strategy game model against data poisoning attacks. In *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, pages 39–43.

Papernot, N., McDaniel, P. D., and Goodfellow, I. J. (2016). Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *CoRR*, **abs/1605.07277**.

Papernot, McDaniel, S. W. (2016). Sok: Towards the science of security and privacy in machine learning. *arXiv:1611.03814*.

Paudice, A., Muñoz-González, L., Gyorgy, A., and Lupu, E. C. (2018). Detection of adversarial training examples in poisoning attacks through anomaly detection. *arXiv:1802.03041*.

Reny, P. J. (1999). On the existence of pure and mixed strategy nash equilibria in discontinuous games. *Econometrica*, **67**(5), 1029–1056.

Rubinstein, B. I., Nelson, B., Huang, L., Joseph, A. D., Lau, S.-h., Rao, S., Taft, N., and Tygar, J. D. (2009). Antidote: understanding and defending against poisoning of anomaly detectors. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement*, pages 1–14. ACM.

Steinhardt, J., Koh, P. W. W., and Liang, P. S. (2017). Certified defenses for data poisoning attacks. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3517–3529. Curran Associates, Inc.

Sugiyama, M. (2015). *Introduction to Statistical Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Von Stackelberg, H. (2010). *Market structure and equilibrium*. Springer Science & Business Media.

Wang, X., An, B., and Chan, H. (2019). Who should pay the cost: a game-theoretic model for government subsidized investments to improve national cybersecurity. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 6020–6027. AAAI Press.

Xiao, H., Biggio, B., Nelson, B., Xiao, H., Eckert, C., and Roli, F. (2015). Support vector machines under adversarial label contamination. *Neurocomputing*, **160**, 53–62.

Zhang, R. and Zhu, Q. (2017). A game-theoretic defense against data poisoning attacks in distributed support vector machines. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 4582–4587. IEEE.