Optimal Randomized Classification in Adversarial Settings

Yevgeniy Vorobeychik and Bo Li
Electrical Engineering and Computer Science
Vanderbilt University
Nashville, TN
{yevgeniy.vorobeychik, bo.li.2}@vanderbilt.edu

ABSTRACT

The problem of learning to distinguish good inputs from malicious has come to be known as *adversarial classification* emphasizing the fact that, unlike traditional classification, the adversary can manipulate input instances to avoid being so classified. We offer the first general theoretical analysis of the problem of adversarial classification, resolving several important open questions in the process. First, we significantly generalize previous results on adversarial classifier reverse engineering (ACRE), showing that if a classifier can be efficiently learned, it can subsequently be efficiently reverse engineered with arbitrary precision. We extend this result to randomized classification schemes, but now observe that reverse engineering is imperfect, and its efficacy depends on the defender's randomization scheme. Armed with this insight, we proceed to characterize optimal randomization schemes in the face of adversarial reverse engineering and classifier manipulation. What we find is quite surprising: in all the model variations we consider, the defender's optimal policy tends to be either to randomize uniformly (ignoring baseline classification accuracy), which is the case for targeted attacks, or not to randomize at all, which is typically optimal when attacks are indiscriminate.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed artificial intelligence—*Intelligent agents*

Keywords

Adversarial classification, game theory

1. INTRODUCTION

Machine learning has become a mainstay in a wide variety of prediction tasks. Among these, there has been considerable interest in using machine learning techniques for security, for example as a means for detecting intrusions, separating spam email from good, and distinguishing benign and malicious files [21]. However, applying machine learning in cyber (or physical) security has a particular caveat: unlike traditional machine learning tasks, the adversary in security domains can, and will, actively attempt to undermine the classifier, either through evasion (trying to change the patterns of behavior so as to be misclassified as benign) or attempted sabotage (for example, manipulating the training data or the classification software to mislabel instances). The study of machine learning in such settings has come to be known as *adversarial machine learning* [8, 16, 3, 5, 15, 2, 6, 14, 10, 7, 18]. Most of the work to date can be grouped into two categories: approaches that attempt to modify the learning algorithms to be robust to adversarial manipulation, typically by leveraging a game theoretic model of the learner-attacker interaction (e.g., [8, 5]), and studies of the algorithmic complexity of the adversary's classifier evasion problem (e.g., [16, 18]). Moreover, with the notable exceptions of Biggio et al. [3] (who propose adding noise to the classification boundary) and Colbaugh and Glass [7] (who address randomization among two equally good classifiers), the vast majority of the work considers only deterministic classification schemes. When possible, however, randomization is widely recognized to be of great value in security, and it has been leveraged in many other domains [20, 11].

We study the problem of adversarial classification by drawing on ideas from machine learning theory, with the aim of offering fundamental insights into the nature of the problem. To begin, we consider the question of *adversarial reverse engineering*, asking in what circumstances the adversary can efficiently learn the classifier used by the defender with arbitrary precision. In previous work, Lowd and Meek [16] showed that a similar problem of reverse engineering a classifier *through queries* (an easier problem) is, in general, hard, but this task can be efficiently done for linear classifiers. Recently, Nelson et al. [18] have extended these results to show that query-based reverse engineering can be efficiently done (approximately) for arbitrary convex-inducing classifiers, but also note that without the ability to query the classifier, the problem becomes NP-Hard in general. In other work, Nelson et al. [19] state as an open question a full characterization of the classes of classifiers that are easy to evade. Our first result shows that *an arbitrary* classifier can be reverse engineered if we impose a natural restriction that it came from a learnable class. Thus, for practical purposes, we address the open question posed by Nelson et al. Additionally, we prove an analogous result for randomized classification schemes, showing that in this case the scheme can be efficiently, but imperfectly, reverse engineered, answering another open question from Nelson et al.

As we identify a clear (theoretical) advantage of randomized classification scheme, we proceed to study the problem of *optimal* randomization schemes, given a pair of classifiers

Appears in: *Alessio Lomuscio, Paul Scerri, Ana Bazzan, and Michael Huhns (eds.), Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014), May 5-9, 2014, Paris, France.*

that may be used as a part of it. Colbaugh and Glass [7] showed under restrictive assumptions on adversarial behavior that, armed with two equally accurate classifiers, the learner would optimally choose a uniform randomization scheme. We study the general setting in which two classifiers have arbitrary accuracy, and consider a number of settings with varying adversary capabilities and goals. A surprising outcome of our endeavor is that, by and large, optimal randomization schemes take on one of two modes: either the defender randomizes uniformly among the two classifiers, or chooses the one with higher accuracy. Specifically, when attacks are highly targeted, the learner will always randomize, while in the case of indiscriminate attacks, the tendency is for the learner to choose the better classifier.

2. REVERSE ENGINEERING AS LEARNABILITY

2.1 Deterministic Classification

Consider a learning system (a "defender") which has been trained to distinguish good and bad instances, for some notion of good and bad (for example, malware and goodware, spam and ham, etc). We begin by assuming that this learning system uses a single classifier, and consider the attacker's problem of *reverse engineering* this classifier, or using the data of past classifications (collected, for example, from repeatedly querying the classifier) to determine with high precision the nature of the classifier.

Formally, let $\mathcal{H}$ be the class of possible classifiers which the defender considers (i.e., the defender's *hypothesis class*). Let $(x, y) \sim P$ be instances, where $x \in X$ is an input feature vector, and y is a label in $\{0, 1\}$. To simplify exposition, we assume below that $y = \bar{h}(x)$, i.e., a deterministic function of input x which is the true classification of x as either benign or malicious. For any $h \in \mathcal{H}$, let $e(h) = \Pr_{x\sim P}[h(x) \neq \bar{h}(x)]$ the expected error of h w.r.t. P, and we define $e_{\mathcal{H}} = \inf_{h\in\mathcal{H}} e(h)$ as the optimal (smallest) error achievable by any function $h \in \mathcal{H}$. Let $z^m = \{(x_1, y_1), \dots, (x_m, y_m)\}$ be data generated according to P and let Z^m be the set of all possible z^m.

Definition 2.1. *Let $\mathcal{H}$ be a class of functions mapping x to $\{0, 1\}$. A* learning algorithm *is a function $L : \cup_{m\geq1} Z^m \to \mathcal{H}$. We say that $\mathcal{H}$ is* learnable *if there is a learning algorithm for $\mathcal{H}$ with the property that for any $\epsilon,\delta \in (0, 1)$ and any P, there exists $m_0(\epsilon,\delta)$, such that for all $m \geq m_0(\epsilon,\delta)$, $\Pr_{z^m\sim P}\{e(L(z^m)) \leq e_{\mathcal{H}} + \epsilon\} \geq 1-\delta$. We say it is efficiently learnable if $m_0(\epsilon,\delta)$ is polynomial in $1/\epsilon$ and $1/\delta$ and there exists a learning algorithm for $\mathcal{H}$ which runs in time polynomial in m, $1/\epsilon$, and $1/\delta$.*[1]

In words, the definition of learnability above says that a hypothesis class $\mathcal{H}$ is learnable if we can compute a nearly-optimal candidate from this class for an arbitrary distribution P over data. In our context, learning will be performed at two levels: first, by the "defender", who is trying to distinguish between good and bad instances, and second, by an "attacker", who is trying to infer the resulting classifier. We call the *attacker's* learning task as the *reverse engineering* problem, with an additional restriction: we require the attacker to be within a small error, γ, from the *actual* classification behavior by the defender, *rather than merely from the best candidate in a hypothesis class.*

Definition 2.2. *We say that a distribution P over data (x, y) can be efficiently γ*-reverse engineered *using a hypothesis class $\mathcal{F}$ if there is an efficient learning algorithm $L(\cdot)$ for $\mathcal{F}$ with the property that for any $\epsilon,\delta \in (0, 1)$, there exists $m_0(\epsilon,\delta)$, such that for all $m \geq m_0(\epsilon,\delta)$, $\Pr_{z^m\sim P}\{e(L(z^m)) \leq \gamma + \epsilon\} \geq 1 - \delta$.*

As the following result demonstrates, efficient learning directly implies efficient 0-reverse engineering.

Theorem 2.1. *Suppose that $\mathcal{H}$ is polynomially learnable, and let $\hat{h} \in \mathcal{H}$ be the best candidate in $\mathcal{H}$ for some distribution over instances P. Then the distribution over (x, y) with $x \sim P$ and $y = \hat{h}(x)$ can be efficiently 0-reverse engineered.*

Proof. Suppose that we have learned the classifier $\hat{h} \in \mathcal{H}$. Consider the new distribution of instances $(x, \hat{h}(x))$ where $x \sim P$. The task of the attacker is to reverse engineer this new distribution. In other words, the attacker wishes to *efficiently learn* h_{re} which is arbitrarily close to $\hat{h}$ w.r.t. $x \sim P$. Since $\hat{h} \in \mathcal{H}$, $e_{\mathcal{H}} = 0$, and since $\mathcal{H}$ is efficiently learnable (it was chosen so by the defender), it follows directly that there is an efficient learning algorithm for the attacker that outputs h_{re} which makes an error w.r.t. $x \in P$ and $\hat{h}$ of at most ϵ with probability at least $1 - \delta$. That is, the attacker can consequently efficiently 0-reverse engineer $\hat{h}$. □

In essence, this result tells us that if the defender can learn to distinguish good from bad, the adversary can reverse engineer the resulting decision rule. This result is a significant generalization of previous work on ACRE learnability of classifiers [16, 18]. For example, Nelson et al. [18] prove that all convex-inducing classifiers are ACRE-learnable, but not all can be efficiently reverse engineered (a stronger requirement). We show, in contrast, that all classifiers, convex-inducing or not, can be efficiently reverse engineered, *as long as they are learnable by the defender* for his classification task. Insofar as we view the efficiency of the defender's learning algorithm as a practical prerequisite, our result suggests that reverse engineering is easy *in practice.*

2.2 Randomized Classification Schemes

A fundamental assumption that we have made so far is that the defender learns, and uses, a single, deterministic classifier. Recently, there has been a significant push in the arena of cyber security towards *moving target* or *dynamic* defense [13, 12]. In our setting, a natural interpretation of dynamic defense is to learn multiple classifiers, and use a randomized policy which switches among these on an instance-by-instance basis.

As before, suppose that we have a distribution $\mathcal{P}$ over inputs x and let $\bar{h}(x)$ be some fixed ground truth classifying x as malicious or benign. Suppose that we have n hypothesis classes, $\mathcal{H}_i$, all polynomially learnable. Let $h_i \in \mathcal{H}_i$ be the best fits to $\bar{h}$ (w.r.t. $\mathcal{P}$) from these classes, respectively, and let the corresponding error rates be ϵ_i. Additionally, let$\Delta_{ij} = \Pr_{x\sim\mathcal{P}}\{h_i(x) \neq h_j(x)\}$ for all i, j; that is, Δ_{ij} is the measure of how different h_i and h_j are from one another. We assume that for all $i \neq j$, $\Delta_{ij} > 0$. Consider a space of policies parametrized by $p_i \in [0, 1]$ with $\sum_i p_i = 1$,

[1]This definition is taken, in a slightly extended form, from Anthony and Bartlett [1] (Definition 2.1).

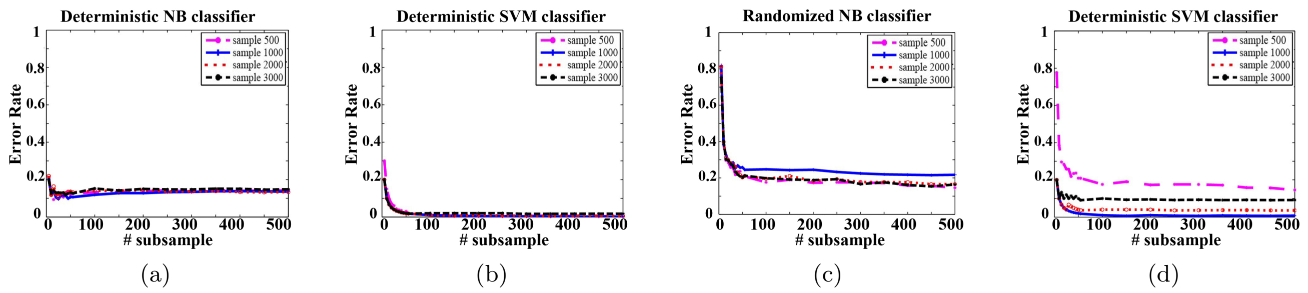

Figure 1: Effectiveness of adversarial reverse engineering (a) deterministic Naive Bayes classifier, (b) deterministic SVM, (c) randomization over two Naive Bayes classifiers (each using a subset of features), (d) randomization over two SVM classifiers (each using a subset of features).

where we choose h_i with probability p_i; we denote the corresponding probability vector by $\vec{p}$. Then, a policy $\vec{p}$ induces a distribution $\mathcal{Q}(\vec{p})$ over (x, z), where $x \sim \mathcal{P}$ and $z = h_i(x)$ with probability p_i. Suppose that the attacker observes an infinite sequence of such data points, and will fit the best function h from $\mathcal{H} = \cup_i \mathcal{H}_i$ to these. Define$\Sigma_A(\vec{p})$ to be the expected error of the best such fit measured against the true randomized policy $\vec{p}$ by the defender. The defender, in turn, will incur a baseline error rate of$\Sigma_D(\vec{p}) = \sum_i p_i \epsilon_i$ if there are no attacks.

Theorem 2.2. *Suppose that each $\mathcal{H}_i$ is polynomially learnable, and let $h_i \in \mathcal{H}_i$ be the best candidate in $\mathcal{H}_i$. Then, any distribution $\mathcal{Q}(\vec{p})$ over (x, z) can be efficiently $\Sigma_A(\vec{p})$-reverse engineered, where $\Sigma_A(\vec{p}) = \min_i \sum_{j \neq i} p_j \Delta_{ij}$.*

This and other missing proofs are in the online appendix (`http://appendices.webs.com/proofs.pdf`). A direct consequence of this result is that if each $\mathcal{H}_i$ admits a good fit to the ground truth, the randomized classification scheme can be approximately reverse-engineered for any $\vec{p}$.

Corollary 2.3. *Suppose that each $\mathcal{H}_i$ is polynomially learnable, and let $h_i \in \mathcal{H}_i$ be the best candidate in $\mathcal{H}_i$. Then, any distribution $\mathcal{Q}(\vec{p})$ over (x, z) can be efficiently $2\bar{\epsilon}$-reverse engineered, where $\bar{\epsilon} = \max_i \epsilon_i$.*

There are two ways to think about this result. On the one hand, it makes clear that even with randomization, reverse engineering is easy as long as all classifiers among which we randomize accurately predict the target. On the other hand, clearly the attacker does incur an error, and this error depends directly on the difference among the classifiers, which can be significant if at least some of the classifiers are not very accurate. This suggests tradeoff between classification accuracy and susceptibility to being reverse engineered: using low-accuracy classifiers allows the defender to induce a higher learning error rate on the attacker, but will also degrade performance against the target. In the sequel, we explore this tradeoff, considering the problem of choosing an *optimal randomized classification scheme*. Before we can do this, however, we need a tighter bound on the reverse engineering error than that provided by Theorem 2.2. We now show that in the special case when the defender only randomizes between two classifiers (i.e., $n = 2$), this bound is tight. Since there are only two classifiers, let p be the probability of choosing h_1, with $1 - p$ the probability of choosing h_2. The bound in Theorem 2.2 is then equivalent to $\Delta \min\{p, 1-p\}$, where $\Delta = \Delta_{12} = \Delta_{21}$.

Proposition 2.4. *Suppose that there are two classifiers, h_1 and h_2, and p is the probability of choosing h_1. Then, $\Sigma_A(p) = \Delta \min\{p, 1-p\}$.*

This result is critical for our analysis of specific adversarial reverse engineering vignettes below, which we restrict to two classifiers. As it turns out, this result does not hold for more than two classifiers (see online appendix).

2.3 Experiments

The theoretical results above have two practical limitations: first, they are asymptotic, assuming large amounts of data, and second, they assume that the algorithms employed actually satisfy the theoretical assumptions (for example, the particular algorithm used may end up finding a local, rather than a global, minimum). We consider, therefore, two algorithms commonly used for spam classification: Naive Bayes and SVM. In our experiments, we first learn a classifier on Enron data, then use a subset of samples of input vectors of a given size, evaluated using the learned classifier, as input to the adversary's reverse engineering problem. The adversary then uses the same algorithm to attempt to identify the classifier based on this data. In Figure 1 (a) and (b) we offer the results for Naive Bayes and SVM, respectively, for different numbers of initial samples used to train the classifier, as well as number of samples taken by the attacker. The theoretical result is essentially borne out in both cases, although it is much stronger in the case of SVM, in large part because it is a more effective algorithm. This is a recurrent theme in our paper: a good learning algorithm is a double-edge sword, since it can equally well be applied to reverse engineer the resulting classifier. Notice that this is quite unlike traditional over-fitting: here, we actually mean performance on *test* data; the issue is that effective algorithms are effective also when used by the attacker.

In the next set of experiments, we consider what happens if the learner can randomize among two classifiers. We set this up by splitting the set of features into two non-overlapping subsets, learning corresponding classifiers, and then uniformly randomizing between these. Figure 1 (b) and (c) shows the reverse engineering error rates when both classifiers are Naive Bayes and SVM respectively. As our results predict, randomization can be quite effective in reducing the accuracy of reverse engineering attempts, but there is a caveat: when the learner uses a larger sample size, the advantage of randomization fades away. The reason is that when the learner uses relatively few samples, the learned classifiers are typically less accurate and, consequently, more unlike (i.e., higher Δ), and recall that it is the difference between the two classifiers that is pivotal in determining reverse engineering effectiveness. With many training samples, however, both learners are likely quite similar, so randomization offers little advantage.

3. OPTIMAL RANDOMIZED DEFENSE: BASELINE MODEL

We start our study of optimal randomized classification by considering an objective which explicitly trades off minimizing the error against the underling target $\bar{h}$ and maximizing reverse engineering error. We assume that there are only two candidate classifiers, h_1 and h_2, with errors ϵ_1 and ϵ_2 w.r.t. $\bar{h}$, and consequently the defender's decision amounts to choosing p, which we define as the probability of classifying an input according to h_1. Without loss of generality we assume from now on that $\epsilon_2 \leq \epsilon_1$. The defender's objective function is then $\beta\Sigma_A(p) - (1-\beta)\Sigma_D(p)$, where $\beta \in [0,1]$ is an exogenous measure of importance of the two factors in the objective. We now fully characterize the optimal solutions to this problem.

Theorem 3.1. *The optimal solution of the problem* $\max_{p\in[0,1]} \beta\Sigma_A(p) - (1-\beta)\Sigma_D(p)$ *is:*

$$p^* = \begin{cases} 1/2 & \text{if} \quad \beta > \Delta(1-\beta)(\epsilon_1-\epsilon_2) \\ 0 & \text{if} \quad \beta < (\Delta-\beta)(\epsilon_1-\epsilon_2) \\ \text{any } p \in [0,1/2] & \text{o.w.} \end{cases}$$

The characterization in Theorem 3.1 suggests that in typical cases the defender will either choose one of the two classifiers uniformly at random, or choose the better classifier with probability 1. Which of these is optimal depends on the relative importance of the adversarial ability to reverse engineer the classification scheme (captured by the parameter β), the relative quality of the two classifiers $\epsilon_1 - \epsilon_2$, and how different the two classifiers are (captured byΔ). Other things being equal, the more different the two classifiers, the more value is gained by randomizing between them; on the other hand, the better h_2 is compared to h_1, the stronger the pull towards choosing this classifier with probability 1.

The reason that the result above is surprising is that intuitively we may expect that while uniformly random classification is optimal for equally good classifiers, probabilities would gradually shift to favor the better classifier as its relative superiority increases. The latter is, indeed, more typical of other randomized security schemes [20, 11], and is also the observation previously made in adversarial machine learning [7]. One substantial difference between our baseline optimization problem and the treatment of related problems in the literature is that we only consider the error that the attacker incurs in reverse engineering our decision rule, and do not have a model of how this error affects actual attacks on the classifiers. In the remainder, we consider several models that capture both the reverse engineering problem, and the corresponding "attacks" on the classifiers.

4. MODELS OF CLASSIFIER MANIPULATION

We have so far specified one aspect of our attacker model: the attacker chooses the best fit h to the defensive classification scheme among all functions from a hypothesis class $\mathcal{H} = \cup_i \mathcal{H}_i$, that is, considering all possible candidate classifiers the defender could be using (in doing so, we rely on the assumption that $\mathcal{H}_i$ are learnable and we can therefore approximate h arbitrarily well with enough data). Given such a fit h to the defender's decision process, the main remaining question is how to model the impact of adversarial manipulation which uses h.

We denote by A_h the event that the attacker is manipulating (attacking) a classifier h, and $A_h(x)$ to mean that the manipulation affects an input x (that is, whatever manipulation is deployed by the adversary, it has an effect on input x), whereas $\neg A_h(x)$ will mean that manipulation of h does not affect x. For any classifier h_i used by the defender, the effect of adversarial manipulation is to change the classification accuracy of h_i relative to ground truth, $\bar{h}$, on a subset of inputs x. This can happen in two ways: either the actual classification scheme is impacted directly, where the adversary changes something about the the code implementing the classification (for example, by flipping the outcome of classification for select inputs x using embedded malware), or indirectly, with the adversary actually changing the ground truth $\bar{h}$ (for example, by changing a spam template so that spam is now classified as benign). Generically, we let $\{h_i^A(x) \neq \bar{h}(x)\}$ be the event that the classifier i predicts the true classification of x incorrectly after adversarial manipulation (which may have targeted another classifier h); although we only use a modifier on h_i, we mean this to capture both kinds of manipulation above.

Another important modeling element as yet left open is how the attacker chooses the classifier h to manipulate. We consider two options. The first is that the attacker chooses to manipulate the function that is his best fit to the randomized classification scheme; we refer to this model as *attacking the best fit*. This is a natural, indeed, almost obvious choice. However, in this model the attacker is acting suboptimally from another perspective: in his choice of which classifier to attack, he does not consider his ultimate objective, which is to maximize the expected error incurred by the defender. In our second model, therefore, the attacker chooses a classifier to manipulate so as to maximize the resulting expected classification error by the defender; we refer to this alternative as *attacking to maximize the defender's error*. This latter model carries with it a complication which makes it potentially less practically plausible. While we observed above that when there are only two classifiers, the attack error is minimized when choosing one of them, when we consider that an adversary aims to maximize the defender's error, this restriction may now be suboptimal for the attacker, making analysis intractable at the level of generality we consider here. We deal with this complication by restricting attention to attacks in which the attacker still only chooses one of the classifiers used by the defender. This restriction can be justified by assuming that the attacker knows the true target and all hypothesis classes the defender learns from. In any case, it will provide us with initial insights, and we leave improved treatments of this model for future work.

4.1 Targeted (Classifier Evasion) Attacks

One widely studied class of attacks on classifiers is what is known as *classifier evasion* [19, 2, 18] or *adversarial classifier reverse engineering (ACRE)* attacks [16]. In such attacks, an attacker is assumed to choose a single input x which is misclassified by the learning agent (defender); typically, the focus is on inducing a false negative, thereby allowing the adversary to get malicious input past the filters.

There are, indeed, two variants of this model which we can consider. In the first, the adversary knows the actual target of learning, $\bar{h}$, as well as $\mathcal{H}_i$ for all i used by the defender. If this is the case, the adversary's job is easy: merely find an instance x that will be misclassified by all h_i (which the attacker can infer given above information). This attack will succeed with probability 1, no matter what $\vec{p}$ is.

The limitation of the above model, from the attacker's perspective, is that it is not at all clear how difficult it would be for the attacker to find a desirable instance x that bypasses *all* of the classifiers. We therefore consider an alternative, in which the attacker will either only choose to evade the best fit, or the best alternative of all i that maximizes the defender's error. In this model, if the defender chooses h_i which is manipulated by the attacker, the attacker always succeeds. However, if the defender chooses another classifier h_j, the attacker only succeeds with probability $1 - \Delta_{ij}$.[2]

4.2 Indiscriminate Attacks

Many attacks on machine learning systems are not carefully targeted, either due to the difficulty of identifying precisely what target one should aspire to, or because of limited ability to manipulate specific training or evaluation instances [17, 9]. We now consider several abstract models of attacks that have this nature.

Our first, quite natural, assumption, on the nature of indiscriminate manipulation is that when x is not affected by manipulation, the correctness of its previous classification by h_i is not affected. Formally, we assume that

$$\forall\, i, h: \ \Pr_x\{h_i^A(x) \neq \bar{h}(x) | \neg A_h(x), h_i(x) \neq \bar{h}(x)\} = 1 \quad (1)$$

$$\forall\, i, h: \ \Pr_x\{h_i^A(x) \neq \bar{h}(x) | \neg A_h(x), h_i(x) = \bar{h}(x)\} = 0. \quad (2)$$

In addition, we assume that the impact of manipulation on the accuracy of the classifier i is independent of which classifier is actually being manipulated by the adversary, if we know that x is affected by manipulation. Thus, we introduce the following notation:

$$\forall\, i, h: \ \Pr_x\{h_i^A(x) \neq \bar{h}(x) | A_h(x), h_i(x) \neq \bar{h}(x)\} = r_i \quad (3)$$

$$\forall\, i, h: \ \Pr_x\{h_i^A(x) \neq \bar{h}(x) | A_h(x), h_i(x) = \bar{h}(x)\} = s_i. \quad (4)$$

In words, r_i is the probability that an input misclassified by h_i remains misclassified after manipulation, and s_i the probability that a correctly classified input becomes misclassified. Thus, for highly effective manipulations we expect both r_i and s_i to be close to 1. Finally, we model manipulation as an indicator whether x is affected or not, as described above, which means that, in general, all that matters is the probability that manipulation has an effect on input x. For our purposes, the following variables capture all the relevant information about the effectiveness of manipulation A_h:

$$\Pr_x\{A_i(x) | h(x) \neq \bar{h}(x)\} = f_{ih}, \quad \text{and} \quad (5)$$

$$\Pr_x\{A_i(x) | h(x) = \bar{h}(x)\} = \bar{f}_{ih}, \quad (6)$$

[2]We are effectively assuming here that the attacker evaluates the probability that h_i and h_j are different according to the same distribution as that generating instances x. In fact, this caveat will make no difference for our analysis, and we only keep it to simplify notation.

where we use i as a shorthand for h_i (as we do below as well). Intuitively, f_{ih} is the probability the an instance misclassified by a classifier h is affected by the attack on classifier i. Similarly, $\bar{f}_{ih}$ is the corresponding probability of successfully affecting an instance correctly classified by h through an attack on i. Put differently, f_{ih} and $\bar{f}_{ih}$ capture the externalities of manipulating i on another classifier h when $h \neq i$, or simply the expected impact of manipulation when $h = i$ (i.e., how well it works). When a function being manipulated is one of the defender's classifiers indexed by j, we simply use notation f_{ij} and $\bar{f}_{ij}$ for the above quantities. For a highly successful (targeted) manipulation, for example, we expect $f_{ii} \approx 0$ while $\bar{f}_{ii} \approx 1$, that is, the attacker aims to affect only the correctly classified instances. Finally, we define $f_i := f_{ii}$.

We now introduce some more notation that we will need for the analysis below. Define $\epsilon_{ih} = \epsilon_{hi} = \Pr_x\{h_i(x) \neq \bar{h}(x) \wedge h_i(x) = h(x)\} = \Pr_x\{h(x) \neq \bar{h}(x) \wedge h_i(x) = h(x)\}$. Then, for all i, h, $\epsilon_i = \epsilon_{ih} + \gamma_{ih}$, where $\gamma_{ih} = \Pr_x\{h_i(x) \neq \bar{h}(x) \wedge h_i(x) \neq h(x)\}$, and $\epsilon_h = \epsilon_{ih} + \gamma_{hi}$. Observe that for all i, $\epsilon_{ii} = \epsilon_i$, and $\gamma_{ii} = 0$. Moreover, it is not difficult to verify that $\forall\, i, h \gamma \quad {}_{ih} + \gamma_{hi} = \Delta_{ih} = \Delta_{hi}$. Armed with this framework, we can derive the expression for the probability that a classifier i used by the defender makes a mistake if another classifier h is used as the object of adversarial manipulation.

Theorem 4.1.

$$\Pr_x\{h_i^A(x) \neq \bar{h}(x) | A_j\}$$
$$= (1 - f_{ji})\epsilon_i + r_i f_{ji} \epsilon_i + s_i \bar{f}_{ji}(1 - \epsilon_i). \quad (7)$$

The model of manipulation above, while generic, has a lot of moving parts that make it difficult to obtain clean insights. Consequently, for our analyses below we make another restriction that the probability that an input x is affected depends only on how it is classified by the classifier h targeted by the adversary. Formally, we assume that

$$\Pr_x\{A_h(x) | h(x) \neq \bar{h}(x), h'(x) \neq \bar{h}(x)\}$$
$$= \Pr_x\{A_h(x) | h(x) \neq \bar{h}(x)\} = f_h \quad (8)$$

and

$$\Pr_x\{A_h(x) | h(x) = \bar{h}(x), h'(x) \neq \bar{h}(x)\}$$
$$= \Pr_x\{A_h(x) | h(x) = \bar{h}(x)\} = \bar{f}_h. \quad (9)$$

Note that this is not an innocuous restriction, as it implies that manipulation has to be, in general, uniform across inputs x. This would capture manipulations in which the attacker targets the inputs randomly, but then decides whether, say, to flip a classification output bit depending on whether it is classified correctly or not (or create a new template corresponding to that feature vector if it is classified as benign). Given the restrictions 8 and 9, we can obtain expressions for the probability that a classifier i is incorrect.

Theorem 4.2. *Under the assumptions 8 and 9,*

$$\Pr_x\{h_i^A(x) \neq \bar{h}(x) | A_h\}$$
$$= (1 - \bar{f}_h)\gamma_{ih} + (1 - f_h)\epsilon_{ih} + r_i(\bar{f}_h \gamma_{ih} + f_h \epsilon_{ih})$$
$$+ s_i(\bar{f}_h(1 - \epsilon_i) - (\bar{f}_h - f_h)\gamma_{hi}). \quad (10)$$

Below, we analyze two specific parametric variations of this restricted manipulation model in increasing order of generality:

Model I assumes that $r_i = s_i = 1$ for all i, and $f_h = \bar{f}_h$ for all h. As a consequence, we obtain a much simplified expression for the probability of error due to manipulation:

$$\Pr_x\{h_i^A(x) \neq \bar{h}(x)|A_h\} = \epsilon_i + f_h(1-\epsilon_i) \quad \forall\, i, h.$$

Model II generalizes Model I, allowing $f_h \neq \bar{f}_h$. In this model, the probability of error due to manipulation is

$$\Pr_x\{h_i^A(x) \neq \bar{h}(x)|A_h\} = \epsilon_i + \bar{f}_h(1-\epsilon_i) - (\bar{f}_h - f_h)\gamma_{hi} \quad \forall\, i, h.$$

5. TARGETED ATTACKS

We now proceed to analyze optimal randomized policies under our model of targeted attacks. A key step in this analysis is to quantify the expected error incurred by the defender when h_i is targeted by the adversary, which we do in the following lemma.

LEMMA 5.1. *If the attacker attacks h_1, the defender's objective value is $O(h_1) = (1-\Delta)+p\Delta$, while if h_2 is attacked, the defender's objective value is $O(h_2) = 1 - p\Delta$.*

Armed with this simple characterization, we proceed to describe optimal defense for the two models of attacker choice, one in which the attacker first obtains the best fit h to the data, and then exploits h, and another in which the attacker is actually trying to exploit a function which maximizes the defender's error. Surprisingly, we find below that these two models result in exactly the same prescription for the defender: uniform randomization between the two classifiers.

5.1 Attacking the Best Fit

When $p \geq 1/2$, the attacker will attack h_1, and will attack h_2 when $p < 1/2$. By Lemma 5.1, therefore, and the fact that$\Delta > 0$, the optimal decision of the defender is to randomize uniformly between h_1 and h_2, since the coefficient on p is positive when $p \geq 1/2$ and negative when $p \leq 1/2$. We thus obtain the following result.

THEOREM 5.1. *When the attacker manipulates best fit in a targeted attack, $p^* = 1/2$.*

This result is surprising because it is independent either of the baseline error rates, or of the difference between the classifiers. The objective value, however, does depend on how different the classifiers are, which suggests a very counter-intuitive defensive strategy of choosing two arbitrary classifiers that are as dissimilar as possible, and uniformly randomizing between them. Such a strategy is clearly unreasonable in practice, and this result rather suggests a limitation of considering only targeted attacks: many attacks (spam, etc) are, in fact, not so directly targeted. Often, malware slowly evolves in response to filtering techniques, and our abstraction of this evolution process into a single stage is clearly limiting here. The main insight, however, is that, given highly targeted attacks (e.g., malware generated by criminals targeting, for example, proprietary information), the most the defender can do in this limited defense space is to maximize classification entropy, making the attacker's reverse engineering problem more challenging and uncertain.

5.2 Attacking to Maximize the Defender's Error

Suppose now that the attacker anticipates the result of manipulation on the defender's error, and attempts to make a choice of which h_i to manipulate based on this objective. By Lemma 5.1, therefore, the attacker will prefer h_1 iff $O(h_1) \geq O(h_2)$, or, equivalently, iff $(1-\Delta)+p\Delta \geq 1-p\Delta \Leftrightarrow p \geq 1/2$, since$\Delta > 0$. Consequently, we get the following surprising result:

THEOREM 5.2. *When the attacker maximizes the defender's error in a targeted attack, $p^* = 1/2$.*

Remarkably, therefore, whether the attacker targets the best fit, or attempts to account for the resulting defense classification error, the defender's optimal policy remains the same, which is to classify according to each h_i with equal probability. The baseline error does not enter the equation, since in this model the attacker always finds an input which is misclassified by the targeted function; thus, baseline errors do not even enter the objective. It may still be surprising that the quantity Δ does not play a role, but note that it is symmetric, and the attacker will therefore confer the maximal error on the defender by attacking the most likely h_i under the defender's policy. Consequently, the defender's only recourse is to choose h_i uniformly at random.

5.3 Discussion

Randomization is often noted to endow a defender with substantial power over a deterministic choice of defense [3, 4]. Indeed, we observe this to be very much the case with targeted attacks: either the attacker must now subvert multiple classifiers, or he can be forced by the defender to incur some probability of failure even in a highly targeted attack. When we get down to specifics, however, our results offer several extremely counter-intuitive findings. The first is that the results or the error rates after the attack do not directly depend on the baseline error rate of the classifiers. This is, of course, natural once we consider that the very nature of targeted attacks, through exploiting misclassification, makes baseline error rates irrelevant. But there is a far more subtle, and far more surprising consequence of our results. Consider the optimal expected error rate achievable by the defender given a targeted attack, $1 - \Delta/2$ and note that $\Delta \leq \epsilon_1 + \epsilon_2$ (shown in the proof of Corollary 2.3 in the online appendix). Consequently, we reach the following remarkable conclusion: *the better the baseline performance of classifiers, the worse the performance after a targeted attack*! The reason is that good accuracy of both classifiers implies high similarity, and, consequently, higher likelihood that adversarial tampering will affect both of them. As we mentioned before, we certainly do not use this result to advocate in favor of lousy classifiers for malware detection. Rather, this result offers us insight into some of the tradeoffs between performance on current data and future exploitability that arise in the adversarial classification settings.

6. INDISCRIMINATE ATTACKS

The second important class of attacks on classifiers that we consider involves indiscriminate attacks, implying a threat model in which an attacker does not have the ability to hand-pick a fixed input, but is rather tampering with a broad class of instances (we do allow for some very limited targeting based on whether the classifier is accurately classifying instances or not in *Model II* described earlier).

As above, the first step is to characterize the impact of attacks on defender's classification error, which we do below for Models I and II of consequences of indiscriminate attacks, respectively.

LEMMA 6.1. *Under* Model I*, if the attacker attacks h_i, the defender's objective value is*

$$O(h_i) = \epsilon_2 + f_i(1-\epsilon_2) + p(1-f_i)(\epsilon_1-\epsilon_2).$$

LEMMA 6.2. *Under* Model II*, if the attacker attacks h_1, the defender's objective value is*

$$\begin{aligned} O(h_1) = \epsilon_2 &+ \bar{f}_1(1-\epsilon_2) - (\bar{f}_1 - f_1)\gamma_{12} \\ &+ p\left((1-\bar{f}_1)(\epsilon_1-\epsilon_2) + (\bar{f}_1 - f_1)\gamma_{12}\right). \end{aligned}$$

while when h_2, the defender's objective value is

$$\begin{aligned} O(h_2) = \epsilon_2 &+ \bar{f}_2(1-\epsilon_2) - (\bar{f}_2 - f_2)\gamma_{21} \\ &+ p\left((1-\bar{f}_2)(\epsilon_1-\epsilon_2) - (\bar{f}_2 - f_2)\gamma_{21}\right). \end{aligned}$$

6.1 Attacking the Best Fit

6.1.1 Model I

We now characterize optimal randomization under the first model of adversarial manipulation.

THEOREM 6.1. *The optimal solution when the attacker exploits h_i which is the best fit to the randomized strategy p under* Model I *is:*

$$p^* = \begin{cases} 0 & \text{if} \quad \frac{2(f_2-f_1)}{1-f_1} \le \frac{\epsilon_1-\epsilon_2}{1-\epsilon_2} \\ 1/2 & \text{o.w.} \end{cases}$$

Let us make a few simple observations based on the result in Theorem 6.1. First, if $f_2 < f_1$, $p^* = 0$, since $\epsilon_2 \le \epsilon_1$. This is highly intuitive: if h_2 is better than h_1 in *both* baseline error and susceptibility to manipulation, it is surely the dominant choice. Second, it may be surprising that under no condition do we always (with probability 1) choose h_1 in this setting. To see why, it is helpful to rearrange $O(h_i)$ in Lemma 6.1, getting $O(h_i) = f_i + (1-f_i)(p\epsilon_1 + (1-p)\epsilon_2)$. Since the attacks do not condition on the classifier at all, and are indiscriminate, f_i represents, in essence, an attacker's budget in attacking a classifier h_i; given that, the success of attacks is not affected by the defensive posture p, and only when the attacks are not successful does defense enter into play. However, in the latter case, the only thing that the defender can impact is the baseline accuracy, hence the tendency towards low p^*. The reason the defender will still randomize is entirely to persuade the attacker to attack the classifier h_1 if it yields a significantly lower success rate f_1 (for example, because it is easier to hack the computer that executes that classifier). In practice, it seems likely that $f_1 \approx f_2$ and, therefore, $p^* = 0$ in most interesting cases.

6.1.2 Model II

The characterization for the second model, when the attacker manipulates the best fit, takes the following, rather more complex form.

THEOREM 6.2. *Suppose that $\bar{f}_i \ge f_i$. Then the optimal solution when the attacker exploits h_i which is the best fit to the randomized strategy p under* Model II *is:*

$$p^* = \begin{cases} 0 & \text{if} \quad \frac{\epsilon_2-\epsilon_1}{\gamma_{21}} \ge \frac{\bar{f}_2-f_2}{1-f_2} \quad \& \\ & 2\bar{f}_1(1-\epsilon_2) + (1-\bar{f}_1)(\epsilon_1-\epsilon_2) - D_1\gamma_{12} \\ & \qquad \ge 2\bar{f}_2(1-\epsilon_2) - 2D_2\gamma_{21} \\ 1/2 & \text{o.w.} \end{cases}$$

where $D_i = \bar{f}_i - f_i$.

One high-level observation we can make about the characterization in Theorem 6.2 is that there is now a stronger push towards randomization when the attacker actively discriminates manipulations based on observed (or inferred) classification correctness of the defender. Nevertheless, for a substantial range of problem parameters, the tendency is for the defender to choose the classifier with lower baseline error with probability 1.

6.2 Attacking to Maximize the Defender's Error

6.2.1 Model I

LEMMA 6.3. *If $f_1 > f_2$, the attacker always exploits h_1. If $f_1 < f_2$, the attacker always exploits h_2. If $f_1 = f_2$, the attacker is indifferent.*

The implication of this result is that the attacker's optimal decision is *independent of the learner's choice of p*. Since both $O(h_1)$ and $O(h_2)$ are increasing in p, the defender's optimal strategy is to set $p^* = 0$. We state this in the following theorem.

THEOREM 6.3. *In an optimal solution, the defender chooses h_2 with probability* 1*, that is, $p^* = 0$.*

This result echoes what we observed in the case when the attacker simply attacks the best fit. In the latter case, the characterization was somewhat more nuanced, but practically speaking, the bottom-line is that $p^* = 0$ is a robust optimal strategy for the defender when attacks are indiscriminate: since such attacks do not strongly depend on the choice of a classifier the defender uses, he may as well use the one with the best baseline performance. Randomization adds little value in this context.

6.2.2 Model II

Below, we define $\bar{H} = (\bar{f}_1 - \bar{f}_2)(1-\epsilon_2) - [(\bar{f}_1 - f_1)\gamma_{12} - (\bar{f}_2 - f_2)\gamma_{21}]$, $\underline{H} = (\bar{f}_1 - \bar{f}_2)(\epsilon_1 - \epsilon_2) - [(\bar{f}_1 - f_1)\gamma_{12} + (\bar{f}_2 - f_2)\gamma_{21}]$, and $H = \bar{H}/\underline{H}$.

LEMMA 6.4. *If $\underline{H} > 0$, the attacker exploits h_1 iff $p \le H$. Otherwise, the attacker exploits h_1 iff $p \ge H$.*

For the sequel, assume that $\bar{f}_i \ge f_i$, in which case $\underline{H} \ge \bar{H}$. We distinguish three cases.

Case I: $\bar{H} > 0$ and $\underline{H} > 0$. In this case, $H \ge 1$, which implies that the attacker chooses h_1 for any p. Since $O(h_1)$ is increasing in p when $\bar{f}_i \ge f_i$, $p^* = 0$ in this case.

Case II: $\bar{H} > 0$ and $\underline{H} < 0$. In this case, the attacker will choose h_1 iff $p \ge H$. Since $H < 0$, this implies that the attacker will always exploit h_1 and, consequently, the defender will choose $p^* = 0$, just as in Case I.

Case III: $\bar{H} < 0$ and $\underline{H} < 0$, which implies that $H \ge 0$. Additionally, since $\bar{H} \ge \underline{H}$, $-\bar{H} \le -\underline{H}$, which implies that $H \le 1$. By Lemma 6.4, then, the attacker will exploit h_1 iff $p \ge H$. Since by our assumption above, $O(h_1)$ is increasing in p, it means that if the defender is trying to incentivize the attacker to attack h_1, his optimal choice is $p^*_{h_1} = H$. Matters are slightly more subtle if the defender wishes to incentivize an attack on h_2. Here, it is not difficult to derive that $O(h_2)$ is decreasing in p iff $\frac{\epsilon_1-\epsilon_2}{\gamma_{21}} \ge \frac{\bar{f}_2-f_2}{1-f_2}$. If $O(h_2)$ is decreasing in p, $p^*_{h_2} = p^*_{h_1} = H$, which implies that $p^* = H$. Otherwise, the optimal decision depends on the relative objective values in these two instances. The optimal objective value for the defender if the attacker attacks h_1 (i.e., when $p = H$) is $\epsilon_2 + \bar{f}_1(1-\epsilon_2) - (\bar{f}_1 - f_1)\gamma_{12} + H[(1-\bar{f}_1(\epsilon_1-\epsilon_2)+(\bar{f}_1-f_1)\gamma_{12})]$, while the optimal setting of p when $O(h_2)$ is increasing in p is $p = 0$, yielding the objective value of $\epsilon_2 + \bar{f}_2(1-\epsilon_2) - (\bar{f}_2 - f_2)\gamma_{21}$. Consequently, $p^* = 0$ in this case i ff

$$(\bar{f}_2 - f_2)\gamma_{21} - (\bar{f}_2 - \bar{f}_1)(1-\epsilon_2) \\ \leq (1-H)(\bar{f}_1 - f_1)\gamma_{12} - H(1-\bar{f}_1)(\epsilon_1 - \epsilon_2). \tag{11}$$

To help understand the implication of Condition 11 it is useful to consider what is implied by Case III: $\bar{H} < 0$ and $\underline{H} < 0$ suggests that the dominant term in these expressions is $(\bar{f}_1 - f_1)\gamma_{12}$. Thus, the attacker, when he manipulates h_1, can rather effectively distinguish instances which are classified correctly, and those which are not. As a result, we can expect the right-hand-side to be relatively large, and the condition to be typically true. Since h_2 is already better (by our assumption) in terms of baseline (non-adversarial) performance, it is in such a case a clear preference for the defender. Thus, for Case III, just as in Cases I and II, we can conclude that *typically* we expect $p^* = 0$.

6.3 Discussion

Despite considering several different models of manipulation in the context of indiscriminate attacks, our results are relatively consistent: facing this class of attacks, the defender will often eschew randomization entirely, and simply use the classifier with better baseline performance. This is rather dramatically in contrast with the situation we observed when the attacks are targeted, in which case baseline performance of the classifiers had no impact at all, and the defender would simply choose among classifiers uniformly at random.

7. CONCLUSION

We studied optimal randomized classification both when attacks are targeted and when they are indiscriminate, and obtained almost completely different answers: in the former, the defender will randomize uniformly between the classifiers, while in the latter an optimal scheme would rarely randomize at all. In reality, attacks do not neatly fall into one of these stylized classes. In some settings, such as spam, attacks tend towards indiscriminate, while in spear phishing the attacks are more targeted. System designers using machine learning tools in adversarial settings should therefore consider carefully which stylized model is best aligned with their predicament, and use our insights as guidelines to best determine operational security posture.

8. REFERENCES

[1] Martin Anthony and Peter L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999.

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

[3] Battista Biggio, Giorgio Fumera, and Fabio Roli. Adversarial pattern classification using multiple classifiers and randomisation. In *Lecture Notes in Computer Science*, pages 500–509, 2008.

[4] Battista Biggio, Giorgio Fumera, and Fabio Roli. Multiple classifier systems for adversarial classification tasks. In *Eighth International Workshop on Multiple Classifier Systems*, pages 132–141, 2009.

[5] Michael Brückner and Tobias Scheffer. Nash equilibria of static prediction games. In *Advances in Neural Information Processing Systems*, pages 171–179, 2009.

[6] Michael Brückner and Tobias Scheffer. Stackelberg games for adversarial prediction problems. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 547–555, New York, NY, USA, 2011. ACM.

[7] Richard Colbaugh and Kristin Glass. Predictive defense against evolving adversaries. In *IEEE International Conference on Intelligence and Security Informatics*, pages 18–23, 2012.

[8] Nilesh Dalvi, Pedro Domingos, Mausam, Sumit Sanghai, and Deepak Verma. Adversarial classification. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 99–108, 2004.

[9] P. Fogla and W. Lee. Evading network anomaly detection systems: Formal reasoning and practical techniques. In *ACM Conference on Computer and Communications Security*, pages 59–68, 2006.

[10] Ling Huang, Anthony D. Joseph, Blaine Nelson, Benjamin I.P. Rubinstein, and J. D. Tygar. Adversarial machine learning. In *Fourth ACM workshop on Security and artificial intelligence*, pages 43–58, 2011.

[11] Manish Jain, Jason Tsai, James Pita, Christopher Kiekintveld, S. Rathi, Milind Tambe, and Fernando Ordonez. Software assistants for randomized patrol planning for the lax airport police and the federal air marshals service. *Interfacs*, 40:267–290, 2010.

[12] S. Jajodia, A.K. Ghosh, V.S. Subrahmanian, V. Swarup, C. Wang, and X.S. Wang, editors. *Moving Target Defense II*. Springer, 2013.

[13] S. Jajodia, A.K. Ghosh, V. Swarup, C. Wang, and X.S. Wang, editors. *Moving Target Defense*. Springer, 2011.

[14] Murat Kantarcioğlu, Bowei Xi, and Chris Clifton. Classifier evaluation and attribute selection against active adversaries. *Data Min. Knowl. Discov.*, 22(1-2):291–335, January 2011.

[15] Wei Liu and Sanjay Chawla. Mining adversarial patterns via regularized loss minimization. *Machine Learning*, 81:69–83, 2010.

[16] Daniel Lowd and Christopher Meek. Adversarial learning. In *ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pages 641–647, 2005.

[17] Daniel Lowd and Christopher Meek. Good word attacks on statistical spam filters. In *Conference on Email and Anti-Spam*, 2005.

[18] Blaine Nelson, Benjamin I.P. Rubinstein, Ling Huang, Anthony D. Joseph, Steven J. Lee, Satish Rao, and J.D. Tyger. Query strategies for evading convex-inducing classifiers. *Journal on Machine Learning Research*, 13:1293–1332, 2012.

[19] Blaine Nelson, Benjamin I.P. Rubinstein, Ling Huang, Anthony D. Joseph, and J.D. Tygar. Classifier evasion: Models and open problems. In *Workshop on Privacy and Security Issues in Data Mining and Machine Learning*, pages 92–98, 2010.

[20] P. Paruchuri, J. Pearce, Janus Marecki, and Milind Tambe. Playing games for security: An efficient exact algorithm for solving Bayesian Stackelberg games. In *Seventh International Conference on Autonomous Agents and Multiagent Systems*, pages 895–902, 2008.

[21] Robin Sommer and Vern Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *IEEE Symposium on Security and Privacy*, pages 305–316, 2010.