Evasion-Robust Classification on Binary Domains

Bo Li, UC, Berkeley
Yevgeniy Vorobeychik, Vanderbilt University

The success of classification learning has led to numerous attempts to apply it in adversarial settings such as spam and malware detection. The core challenge in this class of applications is that adversaries are not static, but make a deliberate effort to evade the classifiers. We investigate both the problem of modeling the objectives of such adversaries, as well as the algorithmic problem of accounting for rational, objective-driven adversaries. We first present a general approach based on mixed-integer linear programming (MILP) with constraint generation. This approach is the first to compute an optimal solution to adversarial loss minimization for two general classes of adversarial evasion models in the context of binary feature spaces. To further improve scalability and significantly generalize the scope of the MILP-based method, we propose a principled iterative retraining framework, which can be used with arbitrary classifiers and essentially arbitrary attack models. We show that the retraining approach, when it converges, minimizes an upper bound on adversarial loss. Extensive experiments demonstrate that the mixed-integer programming approach significantly outperforms several state-of-the-art adversarial learning alternatives. Moreover, the retraining framework performs nearly as well, but scales significantly better. Finally, we show that our approach is robust to misspecifications of the adversarial model.

Categories and Subject Descriptors: D.4.6 [**Software**]: Security and Protection

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Adversarial classification, classifier evasion, robust learning, mixed-integer linear programming, adversarial examples

ACM Reference Format:
Bo Li and Yevgeniy Vorobeychik. *ACM Trans. Knowl. Discov. Data.* 9, 4, Article 39 (February 2018), 30 pages.
DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

The success of machine learning has led to its widespread use as a workhorse in a wide variety of domains, from text and language recognition to trading agent design. It has also made significant inroads into security applications, such as fraud detection, computer intrusion detection, and web search [Fawcett and Provost 1997; Mahoney and Chan 2002]. The use of machine (classification) learning in security settings has especially piqued the interest of the research community in recent years because traditional learning algorithms are highly susceptible to a number of attacks [Barreno et al. 2010; Barreno et al. 2008; Biggio et al. 2014; Laskov and Lippmann 2010; Nelson et al. 2011]. The class of attacks that is of interest to us are *evasion* attacks, in which an intelligent adversary attempts to adjust its behavior so as to evade a classifier that is expressly designed to detect it [Barreno et al. 2010; Lowd and Meek 2005; Karlberger et al. 2007].

Machine learning has been an especially important tool for filtering spam and phishing email, which we treat henceforth as our canonical motivating domain. To date, there has been extensive research investigating spam and phishing detection strategies using machine learning, most without considering adversarial modification [Sahami et al. 1998; Ying and Jie 2012; Metsis et al. 2006]. Failing to consider an adversary, however, exposes spam and phishing detection systems to evasion attacks. Typically, the predicament of adversarial evasion is dealt with by repeatedly relearning the classifier. This is a weak solution, however, since evasion tends to be rather quick, and relearning is a costly task as it requires one to label a large number of instances (in crowdsourced labeling, one also exposes the system to deliberate corruption of the *training* data). Therefore, several efforts have focused on proactive approaches of modeling the learner and adversary as players in a game in which the learner chooses a classifier or a learning algorithm, and the attacker modifies either the training or test data [Dalvi et al. 2004; El Ghaoui et al. 2003; Liu and Chawla 2009; Fawcett 2003; Lowd and Meek 2005; Brückner and Scheffer 2011; Androutsopoulos et al. 2005].

While there has been considerable prior research on adversarial classifier evasion, there are surprisingly few approaches for evasion-robust classification when feature spaces are binary. For ex-

ample, Brückner and Scheffer [2009] and Brückner and Scheffer [2011] require unrestricted feature spaces (in addition, they impose strong restrictions on the loss function and the form of adversarial evasion cost). The few approaches that do consider binary features, such as Dalvi et al. [2004], either assume that the adversary does not optimally respond to the defender's robust classifier, or restrict attention to zero-sum interactions where the adversary maximizes the defender's loss [Teo et al. 2007], which are both distinct from typical adversarial models of evasion in the literature specifically focusing on classifier evasion attacks, and overly conservative, particularly when loss functions are upper bounds on the zero-one loss (such as a hinge loss). The existence of this gap is particularly remarkable given the importance of binary feature spaces in numerous actual adversarial classification problems, such as spam filtering, where binary bag-of-words features are typical [Hinde 2003; Gyongi and Garcia-Molina 2005; Goodman et al. 2007; Rao and Reiley 2012], and malware classification, where best performing classifiers often rely on binary features (e.g., in pdf malware classification [Srndic and Laskov 2013]).

We bridge this gap in the context of a general adversarial modeling framework in which the adversary trades off evasion success and cost of modifying an original malicious instance. This framework generalizes most of the prior evasion modeling approaches, and we illustrate it using two special cases that are closely connected to common adversarial models in prior literature. We formalize the evasion robustness problem of the defender as adversarial loss minimization, which computes the defender's optimal classifier in the associated classifier-evader Stackelberg game, and propose an exact solution using bi-level mixed-integer linear programming when the classifier is linear and uses l_1 regularization. We term the resulting approach a *Stackelberg game multi-adversary model (SMA)*. The baseline formulation is quite intractable, and we offer two techniques for making it tractable: first, we cluster adversarial objectives, and second, we use constraint generation to iteratively converge upon an optimal solution. The principal merits of our proposed bi-level optimization approach over the state-of-the-art are: a) it is able to capture a very general class of adversary models, including the model proposed by [Lowd and Meek 2005], as well as a novel cost function which allows feature cross-substitution; in contrast, state-of-the-art approaches are specifically tailored to their highly restrictive threat models; and b) it makes an implicit tradeoff between feature selection through the use of sparse (l_1) regularization and adversarial evasion (through the adversary model), thereby solving the problem of adversarial feature selection.

To provide a more general scalable robust learning framework we then propose an iterative retraining with adversarial examples approach, *RAD*, which can boost evasion robustness of *arbitrary learning algorithms using arbitrary evasion attack models*. We show that *RAD* minimizes an upper bound on optimal adversarial risk. This is significant: whereas adversarial risk minimization is a hard bi-level optimization problem with poor scalability properties (indeed, no method exists to solve it for general attack models), *RAD* itself is extremely scalable in practice, as our experiments show. We develop *RAD* for a more specific, but very broad class of adversarial models, offering a theoretical connection to adversarial risk minimization even when the adversarial model is only an approximation. Perhaps the most appealing aspect of the proposed approach is that *it requires no modification of learning algorithms*: rather, it can wrap any learning algorithm "out-of-the-box."

RAD closely relates to prior retraining approaches in machine learning [Goodman et al. 2007; Smutz and Stavrou 2012], especially recent retraining methods proposed specifically in adversarial learning [Teo et al. 2007; Goodfellow et al. 2014; Kantchelian et al. 2015; Kurakin et al. 2017]. Traditional retraining in machine learning is typically one-shot, either periodically ingesting new data as the ground truth evolves (e.g., in spam detection [Goodman et al. 2007]), or by adding synthetic (e.g., adversarial) instances into a data set and retraining once [Teo et al. 2007; Smutz and Stavrou 2012]. Neither idea offers significant adversarial robustness. Approaches recently introduced specifically for adversarial learning settings have proposed iterative retraining (by repeatedly adding adversarial examples into data and retraining the classifier), but as an ad hoc procedure, for example, interleaved with stochastic gradient descent [Goodfellow et al. 2014], with no theoretical guarantees.

This work significantly extends our prior publication in *Neural Information Processing Systems* [Li and Vorobeychik 2014]. Specifically, our contributions are:

(1) A general adversarial evasion framework on binary feature spaces based on two specific evasion models and a novel equivalence-based cost function that explicitly accounts for feature cross-substitution attacks, such as substitution of words by synonyms (Section 4.4; in particular, we consider a new general framework for modeling adversarial evasion, and two specific adversarial evasion models, whereas Li and Vorobeychik [2014] focused on only one of these, and did not consider the more general modeling framework),
(2) *SMA*, a bi-level optimization framework and solution methods that make a principled tradeoff between feature selection and adversarial evasion (Section 5; Li and Vorobeychik [2014] only developed a MILP approach for one of the two adversarial evasion models),
(3) *RAD*, the first systematic framework for adversarial retraining with provable guarantees (not considered by Li and Vorobeychik [2014]), and
(4) extensive experimental evaluation of *SMA* and *RAD*, including evaluation of robustness to misspecification of adversarial behavior (significantly extending the experimental analysis performed by Li and Vorobeychik [2014]).

We illustrate the effectiveness of our methods on both spam filtering and handwritten digit recognition tasks, where evasion attacks are extremely salient [Klimt and Yang 2004; LeCun and Cortes 2010].

2. RELATED WORK

Several streams of research have investigated the use of machine learning in adversarial settings in general, as well as the design of spam filtering systems in particular. Spam detection, of course, has received a great deal of attention (see, e.g., [Hinde 2003; Gyongi and Garcia-Molina 2005; Goodman et al. 2007; Rao and Reiley 2012]). A common approach to spam detection involves the use of classification learning, whereby spam and non-spam instances are labeled and a standard classification algorithm is run to obtain a classifier that would predict a label on future observed emails [Carreras and Marquez 2001; Androutsopoulos et al. 2000]. While typically features of email text are used, other approaches make use of additional characteristics, such as source addresses [Ramachandran and Feamster 2006; Ramachandran et al. 2007]. More generally, machine learning systems have been used in other malware and intrusion detection settings. For example, Lakhina et al. used principal component analysis (PCA) for network anomaly detection [Lakhina et al. 2004].

2.1. Classifier Evasion

The problem of classifier evasion has been considered from an algorithmic perspective by casting it as an optimization problem in which the attacker chooses an instance (a feature vector) to minimize a cost function, penalizing deviations from an ideal attack, subject to a constraint that the new instance is classified as benign [Dalvi et al. 2004; Lowd and Meek 2005; Nelson et al. 2011; Nelson et al. 2012a]. Formally, this evasion problem, termed *adversarial classifier reverse engineering* (ACRE), has been cast in terms of a query model where the adversary has query access to the classifier "oracle." ACRE has been shown to be NP-Hard even when linear classifiers are used (if features are binary), although algorithms with provable approximation guarantees have been developed first in the context of a linear classifier [Lowd and Meek 2005] and then for general convex-inducing classifiers [Nelson et al. 2012a]. Vorobeychik and Li [2014] studied the general problem of black-box attacks on classifiers, showing that classifiers which can be learned in polynomial time can also be reverse engineered to arbitrary precision in polynomial time. In much prior literature on evasion attacks, the cost function which captures the cost to an attacker of changing features of an instance has taken the form of an l_p norm difference between an "ideal" instance and the instance chosen by the attacker. One of our contributions is to describe the limitations of this cost function, and propose a generalization that addresses these limitations by considering feature cross-substitution attacks.

Evasion attacks have recently received considerable attention in the context of deep learning systems [Goodfellow et al. 2014; Papernot et al. 2016c; Nguyen et al. 2015]. Tabacof and Valle [2015] analyzed the adversarial image space and showed that adversarial images appear in large regions in the pixel space. Papernot et al. [2016c] studied the limitation of adversarial evasion examples and showed that some instances are more difficult to manipulate than others. Sabour et al. [2015] demonstrated that the attacker can change classification to an arbitrary class by malicious manipulations. Even without knowing exactly the learning algorithm, several black-box attacks have been proposed [Papernot et al. 2016a; Papernot et al. 2016b].

2.2. Evasion-Robust Classification

A number of efforts have attempted to address the issue of evasion and data poisoning attacks on classifiers through game theoretic modeling and analysis [Parameswaran et al. 2010; Pita et al. 2011; Dalvi et al. 2004].

In one of the earliest such efforts, Dalvi et al. [2004] played out the first two steps of best response dynamics in this game: first, the adversary best responds to a baseline learner by computing an optimal set of modifications, and subsequently the learner computes optimal parameters using the adversarial model. Androutsopoulos et al. [2005] developed a two-player game between spammers and email users to predict equilibrium strategies that could be used to tune spam filters. Vassilakis et al. [2007] extended this model to account for human interactive proofs in conjunction with spam filers. Similarly, Reshef and Solan [2006] consider the optimal behavior of spammers in response to three specific strategies that can be used to combat spam: increasing email delivery costs, filtering, and a do-not-spam registry. Globerson and Roweis [2006] considered a problem where features of a classifier are deleted in an adversarial way at test time, and develop a learning algorithm which is robust to such feature deletion. Brückner and Scheffer [2009] focused on single-shot prediction games, where the utility functions of learner and adversary are not necessarily antagonistic, and propose algorithms to find the equilibria, including equilibrium learning algorithm parameters. Brückner and Scheffer [2011] suggested an alternative game model, a Stackelberg game in which the learner first sets the algorithm parameters, and the follower (attacker) would best respond by optimizing its utility (which is connected to algorithm performance on data). Zhang et al. [2015] proposed a general feature selection algorithm to optimize the generalization capability of both the linear and non-linear wrapped classifier, as well as its security against evasion attacks. Liu and Chawla [2010] formulated the interaction between a data miner and a adversary as a zero-sum Stackelberg game, where the adversary (and not the learner) is the leader and the data miner is the follower. Zhou et al. [2012] introduced an extension of Support Vector Machine optimization that considers attacks that involve adding a displacement vector to each malicious instance to maximize the associated loss, and Zhou and Kantarcioglu [2014] presented a similar extension to a Bayesian hierarchical mixtures of experts model. Torkamani and Lowd [2013] leveraged similar ideas in developing an adversarial learning algorithm that considers associations among labels for different objects (instances). In the deep learning literature, a common approach to evasion defense has been to insert synthetic adversarial evasion instances into training data and retraining [Goodfellow et al. 2014; Kurakin et al. 2017]. This general approach has been shown empirically to be effective, but has been integrated into learning methods in an ad hoc way. Our proposed iterative retraining approach, in contrast, is systematic and theoretically grounded.

2.3. Data Poisoning in Adversarial Machine Learning

In addition to the evasion attacks on classifiers, much work has focused on data poisoning/contamination attacks [Kearns and Li 1993; Newsome et al. 2006; Venkataraman et al. 2008; Rubinstein et al. 2009; Huber 2011; Tyler 2008; Wagner 2004; Kloft and Laskov 2012]. Some of the earliest treatments consider the robustness of learning algorithms to noise, including the extension of the probably approximately correct (PAC) model by [Kearns and Li 1993], as well as the general literature on robust statistics (developing algorithms that are robust to data contamination in a worst-case sense) [Huber 2011; Tyler 2008]. More recently, work has emerged to character-

ize specific classes of deliberate attacks on machine learning systems. One class of attacks is *red herring attacks*, which add words (features) that reduce the maliciousness score [Newsome et al. 2006; Venkataraman et al. 2008]. For example, Newsome et al. [2006] analyzed the red herring attacks against conjunction learners. The attack introduces spurious features during training for the learning systems. The true malicious instances, however, will lack the spurious features and thereby bypass the filter. Rubinstein et al. [2009] have examined how an attacker can exploit the sensitivity of PCA. Specifically, they showed that an attacker can systematically inject traffic to increase variance along the links of their target flow and mislead the anomaly detection system to require a high computational expense (and, consequently, to significantly reduce its usability). Kloft and Laskov [2012] demonstrated another class of attacks called *boiling frog attacks* on centroid anomaly detection. These attacks involve incremental contamination of systems that involve iterative re-training (a common paradigm in machine learning applied to intrusion/spam detection) so that each incremental change is sufficiently small to escape detection, but over time the attack can significantly move the centroid. While data poisoning attacks are an important consideration, they are outside the scope of this work.

3. PROBLEM DEFINITION

Let $X \subseteq R^n$ be the feature space, with n the number of features. For a feature vector $x_i \in X$, we let x_{ij} denote the jth feature. Suppose that the training set (x_i, y_i) is comprised of feature vectors $x_i \in X$ generated according to some unknown distribution $x_i \sim D$, with $y_i \in \{-1, +1\}$ the corresponding binary labels, where -1 means the instance x_i is benign, while $+1$ indicates a malicious instance. The learner aims to learn a classifier with parameters w, $g_w : X \to \{-1, +1\}$, to label instances as malicious or benign, using a training data set of labeled instance $D = \{(x_1, y_1), ..., (x_m, y_m)\}$. Let I_{bad} be the subset of datapoints i with $y_i = +1$; abusing notation, we also use I_{bad} to correspond to the set of malicious *feature vectors* x in the training dataset. Finally, we assume that $g_w(x) = \text{sgn}(f_w(x))$ for some real-valued function $f_w(x)$. Henceforth, we omit the subscript w where clear from context.

Traditionally, machine learning algorithms commonly minimize regularized empirical risk:

$$\min_w \mathcal{L}(w) \equiv \sum_i l(f(x_i), y_i) + \delta \|w\|_p^p, \tag{1}$$

where $l(a, y)$ is the loss associated with a prediction score $a \in \mathbb{R}$ when true classification is y. An important issue in adversarial settings is that instances classified as malicious (in our convention, corresponding to $g(x) = +1$) are associated with malicious agents who subsequently modify such instances in order to evade the classifier (and be classified as benign). Conceptually, we capture such adversarial evasion behavior as an oracle $\mathcal{O}(w, x)$, which returns, for a given parameter vector w and original feature vector (in the training data) x, an alternative feature vector x'. For the moment, the nature of this oracle, which captures adversary's evasion behavior, is generic. Below, we consider the issue of adversarial modeling in greater detail.

When the adversary modifies malicious instances according to a behavior oracle $\mathcal{O}(w, x)$, the resulting effective risk for the defender is no longer captured by Equation 1, but must account for adversarial response. Consequently, the defender would seek to minimize the following *adversarial risk* (on training data):

$$\min_w \mathcal{L}_A(w; \mathcal{O}) = \sum_{i:y_i=-1} l(f(x_i), -1) + \sum_{i:y_i=+1} l(f(\mathcal{O}(w, x_i)), +1) + \delta \|w\|_p^p. \tag{2}$$

We make several observations about the adversarial risk function. First, note that adversarial behavior depends on the original malicious instance x_i in the training data: effectively, we are modeling a collection of adversaries, each behaving quite distinctly, but their behavior is completely captured by x_i (their current malicious action) and w (how they respond to the classifier). In other words, we suppose that every instance $x \sim \mathcal{D}$ corresponds to a fixed label $y \in \{-1, +1\}$, where a label of

$+1$ indicates that this instance x was generated by an adversary. In the context of a threat model, therefore, we take this malicious x to be an expression of *revealed preferences* of the adversary: that is, x is an "ideal" instance that the adversary would generate if it were not marked as malicious (e.g., filtered) by the classifier. Second, adversarial risk function is the direct analog of empirical risk in adversarial settings. In reality, this is a proxy for the expected risk which is what the learner is truly trying to minimize. Third, we assume here, and throughout, that adversary's response behavior is known to the defender. Typically, this assumption is captured by using a model of the adversary's behavior. Essentially all prior literature in classifier evasion has made a far stronger assumption of a *particular* model of adversary behavior, whereas our goal is to ultimately admit a broad class of adversary models within this general framework. Later, we evaluate the question of robustness of adversarial learning against mistakes in adversary modeling.

4. ADVERSARY MODELING

Generally, in prior literature evasion attacks have almost universally been modeled as optimization problems in which attackers balance the objective of evading the classifier (by changing the label from $+1$ to -1) and the cost of such evasion. We now define a very general adversarial modeling framework which extends most of the specific models studied in prior literature [Lowd and Meek 2005; Biggio et al. 2014; Brückner and Scheffer 2011; Li and Vorobeychik 2014]. We then specialize this framework to two important general models. Both are fundamentally optimization problems aiming to trade off two conflicting goals: evading the classifier (trying to find an instance x' such that $g(x') = -1$ and, perhaps, as far from the classification boundary as possible) and making minimal changes to the original malicious instance x, as captured by the associated cost function $c(x', x)$. The first model imposes a strict constraint that evasion is successful without being concerned about precisely how benign the new malicious instance x' appears, but instead imposes additionally a budget constraint on the amount of change to the original feature vector x the adversary can tolerate. In the case where no evasion is found which does not violate the cost budget, the attacker does not undertake evasion.cover The second model explicitly trades-off evasion success and associated cost. Our adversarial models are *white-box*, that is, the attacker knows the classifier, including the score function.

Recall that we treat malicious instances in the dataset, $x \in I_{bad}$ as *ideal* feature vectors capturing behavior that the malicious actor would continue to perform if it were not for the classifier which marks these as malicious. In both models, each such malicious feature vector x is treated as a distinct adversary, and we call this the *ideal* instance x^A for this adversary, and the adversary is assumed to aspire to remain as close to this instance as possible while evading the classifier.

4.1. General Framework for Adversarial Evasion Modeling

We begin with a rather general framework for adversarial evasion modeling, formalized as Problem (3).

$$z = \underset{x' | h(x') \leq 0}{\arg\min}\ l_A(x', x^A) \equiv \beta r(f(x')) + \eta c(x', x^A) \tag{3}$$

$$x^* = \begin{cases} z & z \in \mathcal{C} \\ x & otherwise, \end{cases}$$

In this problem, $\beta, \eta \geq 0$ are exogenously specified parameters which allow us to specialize the model to specific sub-classes (we illustrate two general examples below). $h(x') \leq 0$ represents certain constraint for the modified instance x'. Moreover, $r(a)$ is a non-decreasing function of a, which captures the key adversarial objective of appearing more benign (having a smaller $f(x)$). Finally, the adversary may be constrained in the kinds of modifications they can make, and we express this constraint as $\mathcal{C}$; in the model, if the optimal evasion z is infeasible, the attacker will stay with the original malicious instance x_i.

4.2. Cost Minimization with Budget Constraint (*CMBC*)

Our first specific model, which we refer to as *CMBC*, is a generalization of the adversarial evasion model proposed by [Lowd and Meek 2005]. To formalize this model, consider an attacker who in the original training data uses an ideal feature vector from $x_i \sim D$ denoted as x^A ($x^A \in I_{bad}$). This attacker aims to solve the following optimization problem:

$$\min_{x' \in X : g(x') = -1} c(x', x^A) \tag{4a}$$

$$\text{s.t.}: \quad c(x', x^A) \leq B_c, \tag{4b}$$

where B_c is the highest cost (deviation from x^A) the adversary is willing to tolerate. If no evasion instance within the cost budget is found, the adversary is assumed to continue with the original feature vector x^A. Note that this model specializes Problem (3) when $\beta = 0$, $\eta = 1$, $h(x') \equiv f(x') \leq 0$, and $\mathcal{C} = \{x' | c(x', x^A) \leq B_c\}$.

Lowd and Meek [2005] proposed an iterative approximation algorithm to solve the optimization problem 4 which yields a 2-approximation when the cost function is a weighted l_1 distance, $g(x)$ is linear, and feature space is binary. Moreover, they proposed a polynomial time exact algorithm for continuous feature spaces and linear classifiers. In principle, the approximation algorithm by Lowd and Meek [2005] can be applied with non-linear classifiers as well, although with no guarantees, and we use a generalization of it discussed in Appendix C as a general heuristic, after making the modifications to account for the cost constraint in our model.

4.3. Balancing Evasion Cost and Success (*BECS*)

In our second specific model, referred to henceforth as *BECS*, the adversary has two competing objectives: 1) appear as benign as possible to the classifier, and 2) minimize modification cost. Just as in the first model, we assume that the attacker obtains no value from a modification to the original feature vector if the result is still classified as malicious. Formally, the adversary is solving the following optimization problem:

$$\min_{x' \in X} \min\{0, f(x')\} + c(x', x^A). \tag{5}$$

We assume that $c(x', x^A) \geq 0$, $c(x', x^A) = 0$ iff $x' = x^A$, and c is strictly increasing in $\|x' - x^A\|_2$ and strictly convex in x'. Observe that this second model is, too, a special case of Problem (3), by setting $\beta = 1$, $\eta = 1$, $r(f(x')) = \min\{0, f(x')\}$, $h(x') \equiv 0$, and $\mathcal{C} = \emptyset$.

Because Problem (5) is non-convex, we instead minimize an upper bound:

$$\min_{x'} Q(x') \equiv f(x') + c(x', x^A). \tag{6}$$

In addition, if $f(x^A) < 0$, we return x^A before solving Problem (6). If Problem (6) returns an optimal solution x^* with $f(x^*) \geq 0$, we return x^A; otherwise, return x^*. Problem (6) has two advantages. First, if $f(x)$ is convex and x is real-valued, this is a (strictly) convex optimization problem, which has a unique solution, and we can solve it in polynomial time. An important special case is when $f(x) = w^T x$. The second one we formalize in the following lemma.

LEMMA 4.1. *Suppose x^* is the optimal solution to Problem* (5)*, x_i is suboptimal, and $f(x^*) < 0$. Let $\bar{x}$ be the optimal solution to Problem* (6)*. Then $f(\bar{x}) + c(\bar{x}, x_i) = f(x^*) + c(x^*, x_i)$, and $f(\bar{x}) < 0$.*

The following corollary then follows by uniqueness of optimal solutions for strictly convex objective functions over a real vector space.

COROLLARY 4.2. *If $f(x)$ is convex and x continuous, x^* is the optimal solution to Problem* (5)*, $\bar{x}$ is the optimal solution to Problem* (6)*, and $f(x^*) < 0$, then $\bar{x} = x^*$.*

A direct consequence of this corollary is that when we use Problem (6) to approximate Problem (5) and this approximation is convex, we always return either the optimal evasion, or x_i if no cost-effective evasion is possible. An oracle $\mathcal{O}$ constructed on this basis will therefore return a unique solution.

4.3.1. Coordinate Greedy. Many learning problems do not feature a convex $g(x)$, or a continuous feature space, so that the optimization problems which capture the attacker model above cannot be efficiently solved. To address this, we propose to use *CoordinateGreedy (CG)* (Algorithm 1) to approximate optimal attacker evasion. The key advantage of coordinate greedy is that it does not need any specific information about the nature of the classifier or the cost function, although specific variations, such as coordinate descent, can make use of this information. The high-level idea is to

ALGORITHM 1: CoordinateGreedy(*CG*): $\mathcal{O}(\beta, x)$

Input: Parameter vector β, malicious instance x
Set $k \leftarrow 0$ and let $x^0 \leftarrow x$
repeat
  Randomly choose index $i_k \in \{1, 2, ..., n\}$
  $x^{k+1} \leftarrow i_k + \epsilon$
  $k \leftarrow k + 1$
until $\frac{\ln Q(x^k)}{\ln Q(x^{k-1})} \leq \epsilon$
if $f(x^k) \geq 0$ **then**
  $x^k \leftarrow x$
end if
Output: Adversarially optimal instance x^k.

iteratively choose a feature, and greedily update this feature to incrementally improve the attacker's utility (as defined by Problem (6)).

In general, this algorithm will only converge to a locally optimal solution. Indeed, the issue of only being able to compute an attack heuristically with respect to a model is a fundamental problem in most prior adversarial evasion modeling efforts, excepting several which make strong assumptions about adversarial cost and loss functions (e.g., Brückner and Scheffer [2011]). The concern is that using such a heuristic approach as a means for making a classifier robust will fail as it underestimates the true attacks (which may achieve a better quality solution to the associated optimization problem).

We address this fundamental limitation in two ways. First, we propose a version with random restarts: run *CG* from L random starting points in feature space. As long as a global optimum has a basin of attraction with positive Lebesgue measure, or the feature space is finite, this process will asymptotically converge to a globally optimal solution as we increase the number of random restarts. Thus, as we increase the number of random restarts, we expect to increase the frequency that we actually return the global optimum.

In general, however, asymptotic convergence to a global optimum through random restarts can be exponentially slow and, consequently, of limited help. We therefore consider empirically how effective it is in our setting. Let p_L denote the probability that the oracle based on coordinate greedy with L random restarts returns a suboptimal solution to Problem (6). In Figure 1 we investigate how fast p_L converges to zero. A key observation from this figure is that this convergence tends to be remarkably fast, necessitating relatively few random restarts.

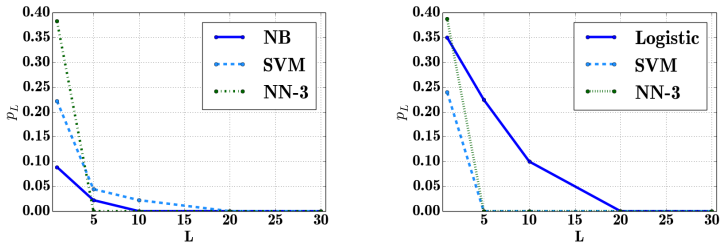


Fig. 1. The convergence of p_L based on different number of starting points for (a) Binary, (b) Continuous feature space.

Consequently, it is unlikely that we underestimate the attacker capability using 20 or fewer random restarts.

4.4. Cost Function Models

Within the threat model for adversary, the cost function $c(x', x^A)$ can have multiple formulations in different situations. Here we will briefly discuss two general categories: the distance-based and equivalence-based cost functions.

4.4.1. Distance-Based Cost Function. In one of the first adversarial classification models, [Lowd and Meek 2005] proposed a natural l_1 distance-based cost function which penalizes for deviations from the ideal feature vector x^A:

$$c(x', x^A) = \sum_i a_i |x'_i - x^A_i|, \tag{7}$$

where a_i is a relative importance of feature i to the adversary. All follow-up work in the adversarial classification domain has used either this cost function or l_p-norm generalizations [Barreno et al. 2010; Barreno et al. 2008; Nelson et al. 2011; Nelson et al. 2012a].

4.4.2. Equivalence-Based Cost Function. While distance-based cost functions seem natural models of adversarial objective, they miss an important phenomenon of feature cross-substitution. In spam or phishing, this phenomenon is most obvious when an adversary substitutes words for their synonyms or substitutes similar-looking letters in words. These words can contain features with the similar meaning or effect (e.g. *money* and *cash*) or differ in only a few letters (e.g *clearance* and *claerance*). The impact is that the adversary can achieve a much lower cost of transforming an ideal instance x^A using similarity-based feature substitutions than simple distance would admit.

To model feature cross-substitution attacks, we introduce for each feature i an equivalence class of features, F_i, which includes all admissible substitutions (e.g., k-letter word modifications or synonyms), and generalize (7) to account for such cross-feature equivalence:

$$c(x', x^A) = \sum_i \min_{j \in F_i | x^A_j \oplus x'_j = 1} a_i |x'_j - x^A_i|, \tag{8}$$

where $\oplus$ is the exclusive-or, so that $x^A_j \oplus x'_j = 1$ ensures that we only substitute between different features rather than simply adding features. The point here is that the equivalence-based cost function significantly may reduce attack costs compared to the distance-based cost function, with the difference increasing in the size of the equivalence class. The practical importance of this observation is that the adversary will far more frequently come under cost budget when he is able to use such substitution attacks. Failure to capture this phenomenon therefore results in a threat model that significantly underestimates the adversary's ability to evade a classifier.

In order to solve the optimization problem 4 in the context of equivalence-based cost function, we generalize the algorithm proposed by Lowd and Meek to l_1 cost and linear classifiers. The generalized algorithm termed $FindBooleanIMAC$ is presented in Appendix C. Note that the algorithm becomes identical to Lowd and Meek's when equivalence classes F_i are singletons (i.e., the cost function reduces to l_1 cost).

5. STACKELBERG GAME MULTI-ADVERSARY MODEL (*SMA*)

We now offer a principled and general approach to adversarial classification in the context of evasion attacks modeled above. For now, we restrict attention to linear classifiers where $f(x) = w^T x$ and l_1 regularization is used. These restrictions are necessary for our *exact* optimization algorithms, but will be subsequently relaxed as we propose an approximate but far more general and scalable approach in Section 6. Because the resulting classifier choice is formally a Stackelberg equilibrium in which there is a single defender (classifier) and multiple followers (evaders), we term this approach *Stackelberg game multi-adversary model (SMA)*.

Since adversaries correspond to feature vectors x_i which are malicious (and which we interpret as the "ideal" instances x^A of these adversaries), we henceforth refer to a given adversary by the associated index i of a malicious instance in the data. We now rewrite the optimization problem (2) for the general SMA model as a bi-level program in which the learner first chooses the weights w and the attackers modify malicious instances x_i into alternatives, $\tilde{x}_i$, in response:

$$\begin{aligned}
&\min_w \sum_{i|y_i=-1} l(w^T x_i, -1) + \sum_{i|y_i=1} l(w^T \widetilde{x_i}, 1) + \delta||w||_1 \qquad (9)\\
&\text{s.t.}: \quad \forall i : y_i = 1,\\
&z_i = \operatorname*{arg\,min}_{x|h(x;w)\leq 0} l_A(x, x_i; w)\\
&\widetilde{x_i} = \begin{cases} z_i & z_i \in \mathcal{C} \\ x_i & otherwise, \end{cases}
\end{aligned}$$

where $l_A(x, , x_i; w)$ is an adversarial loss function that the attacker wishes to minimize (which may depend on the learning parameters w), subject to constraints $h(x; w) \leq 0$. An example of these constraints is $h(x; w) = w^T x \leq 0$, that is, the attacker wishes to ensure that they are classified as benign. The decision of the attackers also depends on whether or not their budget constraints are satisfied by the optimal adversarial instance (for example, whether it's so far from the original malicious instance that malicious utility is largely compromised). This is represented by the constraint that $\tilde{x}_i = z_i$ if $z_i \in \mathcal{C}$, and otherwise the attacker does not change their original feature vector x_i. A natural example of a budget constraint is $\mathcal{C} = \{z|c(z, x_i) \leq B_c\}$.

The power of our approach and the formulation (9) is that it admits, in principle, *an arbitrary adversarial loss function* $l_A(x, x^A; w)$, and, consequently, an arbitrary cost function, unlike prior approaches. The methods described below will generalize as long as we have an algorithm for optimizing the adversary's loss given a classifier.

In order to solve the optimization problem (9) we now describe how to formulate it as a (very large) mathematical program, and then propose several heuristic methods for making it tractable. The first step is to observe that the hinge loss function and $\|w\|_1$ can both be easily linearized using standard methods. We therefore focus on the more challenging task of expressing the adversarial decision in response to a classification choice w as a collection of linear constraints.

We begin by representing the adversary's optimization problem using a collection of linear constraints. Define an auxiliary matrix T in which each column corresponds to a particular attack feature vector x', which we index using variables a; thus T_{ja} corresponds to the value of feature j in the attack feature vector with index a. Define another auxiliary binary matrix Q where $Q_{ai} = 1$ iff the attack strategy $a \in \mathcal{C}$ for the attacker i.

Next, define a matrix L where L_{ai} is the loss of the strategy a to adversary i Finally, let z_{ai} be a binary variable that selects exactly one feature vector a for the adversary i. First, we must have a constraint that $z_{ai} = 1$ for exactly one strategy a: $\sum_a z_{ai} = 1 \; \forall \; i$. Now, suppose that the strategy a that is selected is the best available option for the attacker i; it may be below the cost budget, in which case this is the strategy used by the adversary, or above budget, in which case x_i is used. We can calculate the resulting value of $w^T \tilde{x}_i$ inside the loss function corresponding to adversarial instances using

$$w^T \tilde{x}_i = e_i = \sum_a z_{ai} w^T (Q_{ai} T_a + (1 - Q_{ai}) x_i). \tag{10}$$

This expression introduces bilinear terms $z_{ai} w^T$, but since z_{ai} are binary, these terms can be linearized using McCormick inequalities [McCormick 1976].

To ensure that z_{ai} selects the strategy which minimizes the adversary's loss $l_A(\cdot)$ among all feasible options, captured by the matrix L, we introduce constraints

$$\sum_a z_{ai} L_{ai} \leq L_{a'i} + M(1 - r_{a'}),$$

where M is a large constant and $r_{a'}$ is an indicator variable which is 1 iff $h(T_a; w) \leq 0$ (that is, if feature vector x associated with the attack a, satisfies the constraint $h(x; w) \leq 0$). We calculate r_a for all a using constraints

$$(1 - 2r_a) h(T_a; w) \leq 0.$$

The resulting full mathematical programming formulation is shown below.

$$\min_{w,z,r} \sum_{i|y_i=0} \max\{0, 1 - w^T x_i\} + \sum_{i|y_i=1} \max\{0, 1 + e_i\} + \delta \|w\|_1 \tag{11}$$

$$\text{s.t.:} \quad \forall a, i, j : z_{ai}, r_a \in \{0, 1\} \tag{12}$$

$$\sum_a z_i(a) = 1 \tag{13}$$

$$\forall i : e_i = \sum_a m_{ai} (Q_{ai} T_a + (1 - Q_{ai}) x_i) \tag{14}$$

$$\forall a, i, j : -M z_{ai} \leq m_{aij} \leq M z_{ai} \tag{15}$$

$$\forall a, i, j : w_j - M(1 - z_{ai}) \leq m_{aij} \leq w_j + M(1 - z_{ai}) \tag{16}$$

$$\forall a', i : \sum_a z_{ai} L_{ai} \leq L_{a'i} + M(1 - r_{a'}) \tag{17}$$

$$\forall a : (1 - 2r_a) h(T_a; w) \leq 0. \tag{18}$$

Variables m_{ai} allow us to linearize the Constraints (10), replacing them with Constraints (14)-(16). Constraint 18 is the only non-linear constraint remaining (the hinge loss and l_1 terms in the objective can be linearized using standard methods), and depends on the specific form of the function $h(T_a; w)$; we deal with it below in the two special cases of attack models we consider.

As is, the resulting mathematical program is intractable for two reasons: first, the best response must be computed (using a set of constraints above) for each adversary i, of which there could be many, and second, we need a set of constraints for each feasible attack action (feature vector) $x \in X$ (which we index by a). We tackle the first problem by clustering the "ideal" attack vectors x_i into a set of 100 clusters and using the mean of each cluster as x^A for the representative attacker. This dramatically reduces the number of adversaries and, therefore, the size of the problem. To tackle the second problem, we use constraint generation to iteratively add strategies a into the above program by computing optimal, or approximately optimal, attack strategy to add in each iteration.

ALGORITHM 2: *SMA*(X)

T =randStrats() // initial set of attacks
$X' \leftarrow$ cluster(X)
$w_0 \leftarrow$ MILP(X', T)
$w \leftarrow w_0$
while T changes **do**
  for $x^A \in I'_{bad}$ **do**
    t =computeAttack(x^A, w)
    $T \leftarrow T \cup t$
  end for
  $w \leftarrow$ MILP(X', T)
end while
return w

The full *SMA* iterative algorithm using clustering and constraint generation is shown in Algorithm 2. Here we can apply any clustering algorithm for the cluster() function, an we use K-nearest neighbor algorithm to cluster the data [Peterson 2009]. The matrices Q and L in the mathematical program can be pre-computed in each iteration using the matrix of strategies and corresponding T, as well as the set of constraints $\mathcal{C}$. The computeAttack() function generates an optimal attack by solving (often approximately) the optimization problem $z_i = \arg\min_{x \in \mathcal{C}_1} l_A(x, x_i)$.

In the next several sections we instantiate this approach for two adversarial models described above: *CMBC* and *BECS*. As we show below, both can be formulated as mixed-integer linear programs.

5.1. *SMA* for *CMBC*

Recall that the *CMBC* model minimizes the adversary's cost $c(x, x^A)$ subject to the constraint that $w^T x \leq 0$, that is, that the adversarial instance is classified as benign. Additionally, it uses the cost constraint $\mathcal{C} = \{x | c(x, x^A) \leq B_c\}$, which can be handled directly by the SMA mathematical program described above.

The loss function in this case becomes $l_A(x, x^A; w) = c(x, x^A)$. The non-linear constraint (18), on the other hand, now becomes $(1 - 2r_a) w^T T_a \leq 0$. While this constraint introduces bilinear terms, these can be linearized since r_a are binary. In particular, we can replace it with the following constraints:

$$\forall a : \sum_j w_j T_{aj} \leq 2 \sum_j T_{aj} t_{aj}$$
$$\forall a, j : -M r_a \leq t_{aj} \leq M r_a$$
$$\forall a, j : w_j - M(1 - r_a) \leq t_{aj} \leq w_j + M(1 - r_a),$$

where we introduce a new variable t_{aj} to assist in linearization. The full SMA mathematical program thus becomes a mixed-integer linear program in the context of the CMBC attack model.

Finally, we can implement the iterative constraint generation approach by executing a variant of the Lowd and Meek algorithm in each iteration in response to the classifier w computed in previous iteration. Specifically, Algorithm 3 computes the attacker's best response, which in turn makes use of Algorithm 6 *FindBooleanIMAC* to compute approximately optimal attack strategies in response to a given classifier w, subject to the cost budget constraint.

5.2. *SMA* for *BECS*

In the context of the *BECS* evasion attack model, the attacker's loss becomes $l_A(x, x^A; w) = w^T x + c(x, x^A)$ (the upper bound of the original BECS adversarial loss function). As mentioned in Section 4.3, there is no constraint $\mathcal{C}$, and $h(x; w) \equiv 0$, which also eliminates the non-linear con-

ALGORITHM 3: computeAttack(*CMBC*) (x^A, w)

Get matrix T
Generate matrix C, L based on T, B_c
Randomly select x^- from I_{good}
$t \leftarrow$*FindBooleanIMAC*(x^A, x^-, w)
return t

straint (18). the attacker's best response computation computeAttack() can be calculated by using Algorithm 1 for the *BECS* adversary model.

ALGORITHM 4: computeAttack(*BECS*) (x, w)

Input: Parameter vector w, malicious instance x
Set $k \leftarrow 0$ and let $x^0 \leftarrow x$
repeat
  Randomly choose index $i_k \in \{1, 2, ..., n\}$
  $x^{k+1} \leftarrow i_k + \epsilon$
  $k \leftarrow k + 1$
until $\frac{\ln Q(x^k)}{\ln Q(x^{k-1})} \leq \epsilon$
if $f(x^k) \geq 0$ **then**
  $x^k \leftarrow x$
end if
Output: Adversarially optimal instance x^k.

6. SCALING UP: ADVERSARIAL LEARNING THROUGH RETRAINING

The *SMA* optimization approach is exact, but it suffers from three limitations: 1) it assumes specific structure of the attack models which must be *embedded* in the optimization approach itself, 2) it requires substantial modifications of the learning algorithm, and is restricted to linear classification with l_1 regularization, and 3) it suffers from significant limitations in scalability as shown in the experiments below. Indeed, these are very general issues exhibited by a number of approaches have been proposed for making learning algorithms more robust to adversarial evasion attacks [Dalvi et al. 2004; Li and Vorobeychik 2014; Li and Vorobeychik 2015; Teo et al. 2007; Brückner and Scheffer 2011]. Recently, retraining with adversarial data has been proposed as a means to increase robustness of learning [Goodfellow et al. 2014; Kantchelian et al. 2015; Teo et al. 2007].[1] However, to date, such approaches have not been systematic and have not been formally connected to adversarial risk minimization formalized in Section 3.

Thus, we present a systematic retraining algorithm, *RAD*, for retraining with adversarial data (Algorithm 5). Our key observation is that *RAD* is a principled approximation to *SMA*: specifically, it minimizes an upper bound on adversarial loss. *RAD* systematizes some of the prior insights involving adversarial examples and retraining, and enables us to provide a formal connection between retraining with adversarial data, and adversarial risk minimization in the sense of Equation 2.

The *RAD* algorithm is general in terms of the adversarial models as well as the malicious instances. At the high level, it starts with the original training data X and iterates between computing a classifier and adding adversarial instances to the training data that evade the previously computed classifier, if they are not already a part of the data. A significant enhancement in terms of the speed of the approach can be obtained by clustering malicious instances as done for *SMA*: this would reduce

[1] Indeed, neither [Teo et al. 2007] nor [Kantchelian et al. 2015] focuses on retraining as a main contribution, but observes its effectiveness.

ALGORITHM 5: *RAD*: Retraining with ADversarial Examples

Input: training data X
$N_i \leftarrow \emptyset \; \forall \; i \in I_{bad}$
repeat
  $w \leftarrow$ Train($X \cup_i N_i$)
  $new \leftarrow \emptyset$
  for $i \in I_{bad}$ **do**
    $x' = \mathcal{O}(w, x_i)$
    if $x' \notin N_i$ **then**
      $new \leftarrow new \cup x'$
    end if
    $N_i \leftarrow N_i \cup x'$
  end for
until $new = \emptyset$
Output: Parameter vector w

both the number of iterations, as well as the number of data points added per iteration. Experiments (in the appendix E) show that this is indeed quite effective.

A baseline termination condition for *RAD* is that no new adversarial instances can be added (either because instances generated by $\mathcal{O}$ have already been previously added, or because the adversary's can no longer benefit from evasion). If the range of $\mathcal{O}$ is finite (e.g., if the feature space is finite), *RAD* with this termination condition would always terminate. In practice, our experiments demonstrate that when termination conditions are satisfied, the number of *RAD* iterations is quite small (between 5 and 20). Moreover, while *RAD* effectively increases the importance of malicious instances in training, this does not appear to significantly harm classification performance in a non-adversarial setting. In general, we can also control the number of rounds directly, or use an additional termination condition, such as that the parameter vector w changes little between successive iterations. However, we assume henceforth that there is no fixed iteration limit or convergence check.

6.1. Theoretical Analysis

To analyze what happens if the algorithm terminates, we define the regularized empirical risk in the last iteration of *RAD* as:

$$\mathcal{L}_N^R(w, \mathcal{O}) = \sum_{i \in D \cup N} l(f_w(x_i), y_i) + \delta ||w||_p^p, \tag{19}$$

where a set $N = \cup_i N_i$ of data points has been added by the algorithm (we omit its dependence on $\mathcal{O}$ to simplify notation). We now characterize the relationship between $\mathcal{L}_N^R(w, \mathcal{O})$ and $\mathcal{L}_A^*(\mathcal{O}) = \min_w \mathcal{L}_A(w, \mathcal{O})$, where $\mathcal{L}_A(w, \mathcal{O})$ represents the loss of adversary based on any model parameters.

PROPOSITION 6.1. *$\mathcal{L}_A^*(\mathcal{O}) \leq \mathcal{L}_N^R(w, \mathcal{O})$ for all $w, \mathcal{O}$.*

PROOF. Let $\bar{w} \in \arg\min_w \mathcal{L}_N^R(w, \mathcal{O})$. Consequently, for any w,

$$\begin{aligned}
\mathcal{L}_N^R(w, \mathcal{O}) &\geq \mathcal{L}_N^R(\bar{w}, \mathcal{O}) \\
&= \sum_{i: y_i = -1} l(f_{\bar{w}}(x_i), -1) + \sum_{i: y_i = +1} \sum_{j \in N_i \cup x_i} l(f_{\bar{w}}(x_j), +1) + \delta ||\bar{w}||_p^p \\
&\geq \sum_{i: y_i = -1} l(f_{\bar{w}}(x_i), -1) + \sum_{i: y_i = +1} l(f_{\bar{w}}(\mathcal{O}(\bar{w}, x_i)), +1) + \delta ||\bar{w}||_p^p \\
&\geq \min_w \mathcal{L}_A(w; \mathcal{O}) = \mathcal{L}_A^*(\mathcal{O}),
\end{aligned}$$

where the second inequality follows because in the last iteration of the algorithm, $new = \emptyset$ (since it must terminate after this iteration), which means that $\mathcal{O}(w, x_i) \in N_i$ for all $i \in I_{bad}$. $\square$

In general, retraining, systematized in the *RAD* algorithm, effectively minimizes an upper bound on optimal adversarial risk.[2] This offers a conceptual explanation for the previously observed effectiveness of such algorithms in boosting robustness of learning to adversarial evasion. Formally, however, the result above is limited for several reasons. First, for many adversarial models in prior literature, adversarial evasion is NP-Hard. While some effective approaches exist to compute optimal evasion for specific learning algorithms [Kantchelian et al. 2015], this is not true in general. Although approximation algorithms for these models exist, using them as oracles in *RAD* is problematic, since actual attackers may compute better solutions, and Proposition 6.1 no longer applies. Second, we assume that $\mathcal{O}$ returns a unique result, but when evasion is modeled as optimization, optima need not to be unique. Third, there does not exist effective general-purpose adversarial evasion algorithms the use of which in *RAD* would allow reasonable theoretical guarantees.

These challenges were partially addressed by our general-purpose coordinate greedy algorithms for computing optimal adversarial evasion: coupled with random restarts, we can naturally ensure, with enough restarts, that we eventually obtain an optimal solution with high probability. Moreover, we showed empirically that probability p_L of computing suboptimal instances essentially vanishes with relatively few restarts. We now generalize the above result to offer guarantees in this case as well.

PROPOSITION 6.2. *Let $B = |I_{bad}|$. $\mathcal{L}^*_{A,01}(\mathcal{O}) \leq \mathcal{L}^R_N(w, \mathcal{O}_L) + \delta(p)$ with probability at least $1 - p$, where $\delta(p) = B\left(p_L + \frac{\sqrt{\log^2 p - 8Bp_L \log p} - \log p}{2B}\right)$, and $\mathcal{L}^R_N(w, \mathcal{O}_L)$ uses any loss function $l(f_w(x), y)$ which is an upper bound on the $0/1$ loss.*

PROOF. Let $\bar{w} \in \arg\min_w \mathcal{L}^R_N(w, \mathcal{O}_L)$. Consequently, for any w,

$$\begin{aligned}\mathcal{L}^*_{A,01}(\mathcal{O}_L) &= \min_w \mathcal{L}_{A,01}(w; \mathcal{O}_L) \\ &\leq \sum_{i:y_i=-1} l_{01}(f_{\bar{w}}(x_i), -1) + \sum_{i:y_i=+1} l_{01}(f_{\bar{w}}(\mathcal{O}(\bar{w}, x_i)), +1) + \alpha||\bar{w}||^p_p.\end{aligned}$$

Now,

$$\sum_{i:y_i=+1} l_{01}(f_{\bar{w}}(\mathcal{O}(\bar{w}, x_i)), +1) \leq \sum_{i:y_i=+1} l_{01}(f_{\bar{w}}(\mathcal{O}_L(\bar{w}, x_i)), +1) + \delta(p)$$

with probability at least $1 - p$, where $\delta(p) = Bp_L + \frac{\sqrt{\log^2 p - 8Bp_l \log p} - \log p}{2}$, by the Chernoff bound, and Lemma 4.1, which assures that an optimal solution to Problem 6 can only over-estimate mistakes. Moreover,

$$\sum_{i:y_i=+1} l_{01}(f_{\bar{w}}(\mathcal{O}_L(\bar{w}, x_i)), +1) \leq \sum_{i:y_i=+1} \sum_{j \in N_i} l(f_{\bar{w}}(x_j), +1),$$

since $\mathcal{O}_L(\bar{w}, x_i) \in N_i$ for all i by construction, and l is an upper bound on l_{01}. Putting everything together, we get the desired result. $\square$

6.2. *RAD* with Stochastic Gradient Descent

RAD works particularly well with online methods, such as stochastic gradient descent. Indeed, in this case we need only to make gradient descent steps for newly added malicious instances, which

[2]Note that the bound relies on the fact that we are only adding adversarial instances, and terminate once no more instances can be added. In particular, natural variations, such as removing or re-weighing added adversarial instances to retain original malicious-benign balance lose this guarantee.

can be added one at a time until convergence. Note that this is different from interleaving adversarial example generation and stochastic gradient descent steps, as suggested in prior work [Goodfellow et al. 2014; Kurakin et al. 2017]. In fact, the latter approaches lose the theoretical guarantees described above, *particularly* when original instances are replaced with synthetic adversarial examples, as suggested by Kurakin et al. [2017].

6.3. *RAD* and Multi-Class Classification

Discussion so far dealt entirely with binary classification. We now observe that extending it to multi-class problems is quite direct. Specifically, while previously the attacker aimed to make an instance classified as $+1$ (malicious) into a benign instance (-1), for a general label set Y, we can define a malicious set $M \subset Y$ and a target set $T \subset Y$, with $M \cap T = \emptyset$, where every entity represented by a feature vector x with a label $y \in M$ aims to transform x so that its label is changed to T. In this setting, let $g(x) = \arg\max_{y \in Y} f(x, y)$. We can then use the following empirical risk function:

$$\sum_{i:y_i \notin M} l(f(x_i), y_i) + \sum_{i:y_i \in M} l(f(\mathcal{O}(w, x_i)), y_i) + \lambda ||w||_p^p, \tag{20}$$

where $\mathcal{O}$ aims to transform instances x_i so that $g(\mathcal{O}(w, x_i)) \in T$. The relaxed version of the *BECS* adversarial model can then be generalized to

$$\min_{x, y \in T} -f(x, y) + c(x, x_i).$$

Similar generalization is possible for the *CMBC* model.

7. EXPERIMENTS

In this section we investigate the effectiveness of the two proposed methods: the Stackelberg game multi-adversary model (*SMA*) solved using mixed-integer linear programming and the adversarial retraining framework *RAD*.

We consider four data sets for our evaluation: the Enron dataset [Cohen 2009], Ling-spam dataset [Androutsopoulos et al. 2000], UCI dataset [Lichman 2013], and MNIST dataset [LeCun and Cortes 2010]. In particular, the Enron email dataset contains approximately 500,000 emails generated by employees of the Enron Corporation. The Ling-spam dataset includes 2412 Linguist messages, obtained by randomly downloading digests from the archives, separating their messages, and removing text added by the lists server. There are 481 spam messages. Attachments, HTML tags, and duplicate spam messages received on the same day were not included. The UCI dataset contains 4601 email instances, and about 30% of them are spam messages. MNIST is a handwritten digits dataset containing 28×28 images, which represent digit 0-9 for multi-class classification problem.

7.1. Evaluation of *SMA*

We draw a comparison to three baselines: 1) "traditional" machine learning algorithms (we report the results in comparison with standard SVM; comparisons to Naive Bayes and Neural Network classifiers proved similar), 2) Stackelberg prediction game (SPG) algorithm with linear loss [Brückner and Scheffer 2011], and 3) SPG with logistic loss [Brückner and Scheffer 2011]. Both (2) and (3) are state-of-the-art alternative methods developed specifically for adversarial classification problems. Xu et al. [2009] demonstrate a connection between robustness to evasion attacks and regularization. Since we consider adversarial cost functions based on l_1 distance, the relevant regularization in our setting is l_∞. Our first set of results, shown in Figure 2, is a performance comparison of *SMA* based on the *CMBC* threat model to four baselines, evaluated with respect to an adversary striving to evade the classifier, subject to cost budget constraints. The four baselines include SVM, l_∞ regularized SVM (SVM-reg), and SPG with different loss functions. The results demonstrate that *SMA* approaches significantly outperforms the baselines, including l_∞ regularized SVM. The intuition is two-fold: first, the connection between robustness regularization assumes

both a zero-sum game between the learner and the attacker, and a specific l_1-based adversarial cost function, whereas our adversarial models are more general, and not zero-sum.

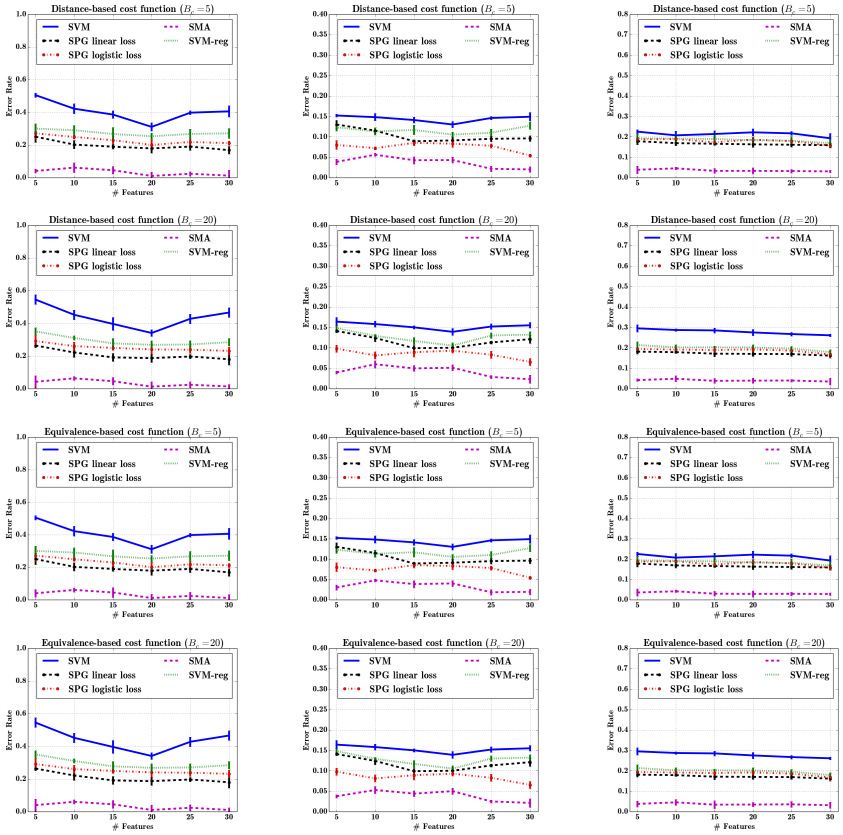


Fig. 2. Comparison of *SMA* based on *CMBC* threat model to baseline alternatives SVM, l_∞ regularized SVM, and SPG on Enron data (a), Ling-spam data (b), and UCI data(c). Row 1: distance-based cost function, $B_c = 5$. Row 2: distance-based cost function, $B_c = 20$. Row 3: equivalence-based cost function, $B_c = 5$. Row 4: equivalence-based cost function, $B_c = 20$.

Here we allow unlimited queries for adversaries and bound the cost budget as 5 and 20. In the case of the Enron data, we can see, remarkably, *SMA* based on the *CMBC* threat model (purple lines in Figure 2) exhibits dramatic performance improvement compared to alternatives in all instances. Moreover, *SMA* is significantly better than the alternatives whether the equivalence-based or distance-based cost function is used.

Figure 3 considers the impact of the number of clusters used in solving the *SMA* based on *CMBC* problem on running time and error. The key observation is that with relatively few (80-100) clusters we can achieve near-optimal performance, with significant savings in running time.

Figure 4 evaluates the *SMA* based on the *BECS* threat model by applying the coordinate descent algorithm to generate multiple attacker strategies as a function of the cost sensitivity λ and then use these attacker strategies to solve *SMA*. It is clear that with larger λ the attacker has higher cost to modify the instances. Therefore, when λ is relatively small, there are more various attacker strategies and *SMA* still outperforms the baselines. Note that while the differences appear smaller here, they

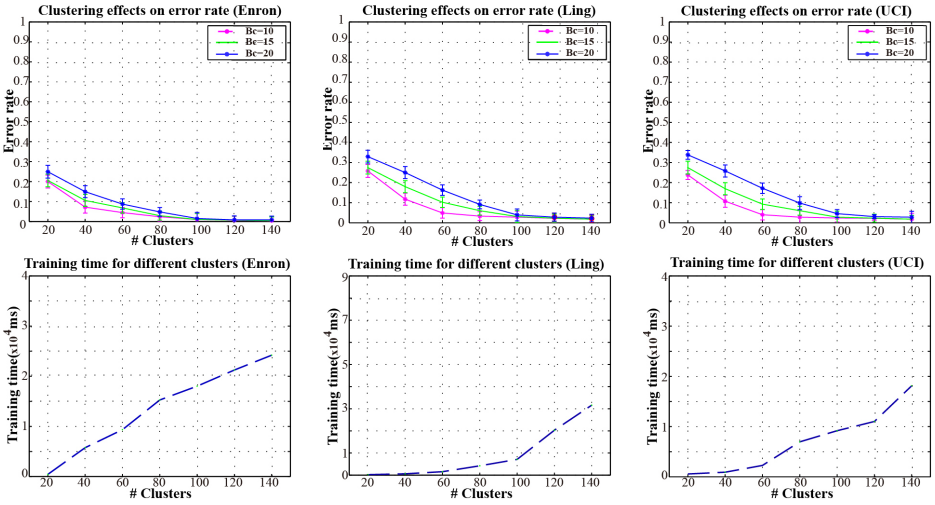


Fig. 3. Row 1: Error rates of *SMA* based on *CMBC*, and Row 2: *SMA* running time as a function of the number of clusters. Results are based on (a) Enron data (b) Ling data, and (c) UCI data.

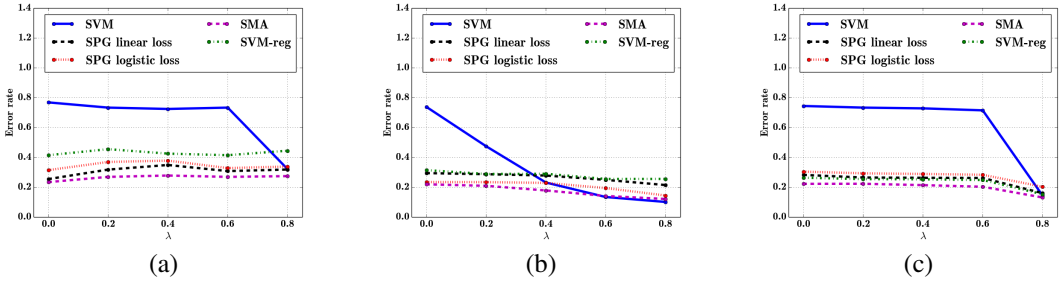


Fig. 4. Comparison of *SMA* based on the *BECS* threat model to baseline alternatives as a function of cost sensitivity λ on Enron data (a), Ling-spam data (b), and UCI data(c).

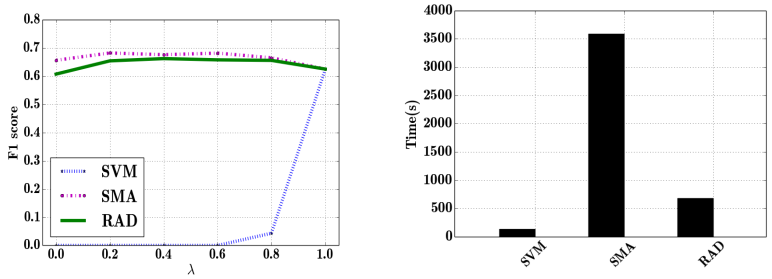
are substantial in percentage terms (in some cases, as much as $\sim$50% improvement shown by *SMA* compared to state-of-the-art adversarial classifiers).

7.2. Evaluation of *RAD*

The theoretical results above for *RAD* suggest that this systematic retraining algorithm is likely to be effective at increasing resilience to adversarial evasion. So we first offer an experimental evaluation to compare the results of *RAD* and *SMA*, which computes an optimal defense against evasion (modulo the slight approximation induced by clustering attacks). We then simulated attacks and present the results for the *RAD* framework based on the *CMBC* and *BECS* threat models, respectively. We compare the results with SPG based on logistic loss and SVM with L_∞ regularize (SVM-reg) to demonstrate the robustness of *RAD*. We also show that the *RAD* works for both discrete and continuous features. Moreover, we also show that the approach is robust to non-adversarial environments, and the cost function and parameter misspecification for defenders.

7.2.1. Comparison of RAD *to Optimal.* The first comparison we draw is to *SMA* and *RAD* based on the *BECS* threat model, for which we can apply the scalable Algorithm 1 to generate adversary strategies. The main limitation of *SMA* is scalability. Because retraining methods use out-of-the-

box learning tools and does not involve non-convex bi-level optimization, it is considerably more scalable.



(a) (b)

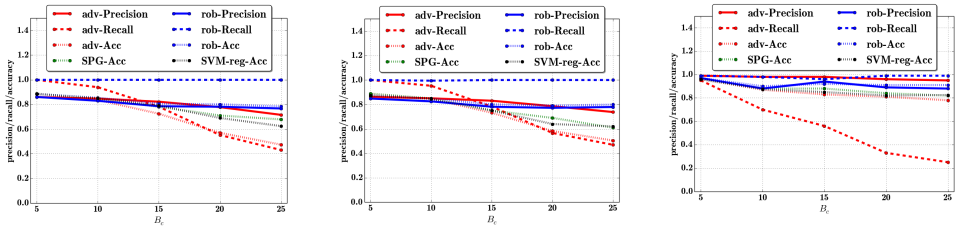
Fig. 5. Comparison between *RAD* and *SMA* based on the *BECS* threat model for Enron dataset with 30 binary features. (a) The F_1 score of different algorithms corresponding to various λ; (b) the average runtime for each algorithm.

We compared *SMA* and *RAD* using Enron data [Klimt and Yang 2004]. As Figure 5(a) demonstrates, retraining solutions of *RAD* are nearly as good as *SMA*, particularly for a non-trivial adversarial cost sensitivity λ. In contrast, a baseline implementation of SVM is significantly more fragile to evasion attacks. However, the runtime comparison for these algorithms in Figure 5(b) shows that *RAD* is much more scalable than *SMA*.

7.2.2. Effectiveness of RAD . In this section we use the Enron [Klimt and Yang 2004] and MNIST [LeCun and Cortes 2010] datasets to evaluate the robustness of three common algorithms in their standard implementation, and in *RAD* for both the *CMBC* and *BECS* threat models: logistic regression, SVM (using a linear kernel), and neural networks (NN) with 3 hidden layers. In Enron data, features correspond to relative word frequencies. 2000 features were used for the Enron and 784 for MNIST datasets. Throughout, we use precision, recall, and accuracy as metrics (in the plots, we indicate accuracy measures by "-Acc"; for example, SVM-reg-Acc refers to accuracy results for l_∞ regularized SVM). We present the results for both continuous and binary feature spaces here.

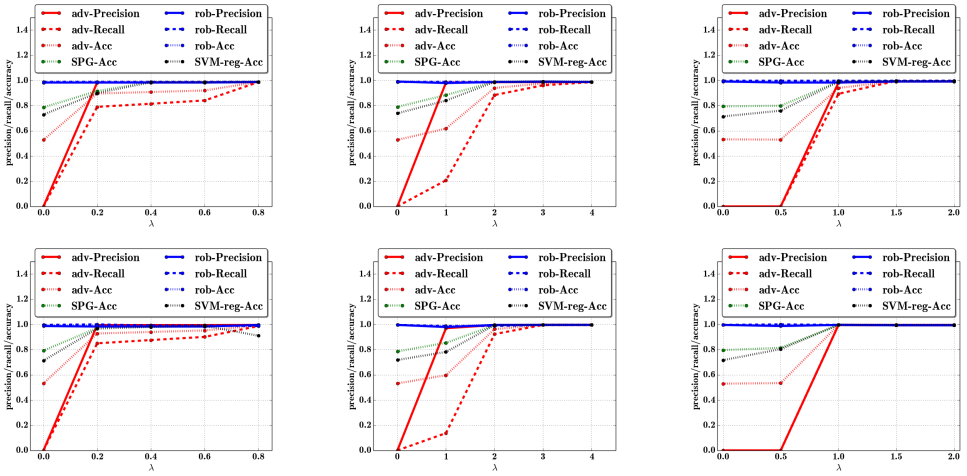
Figure 6 shows the performance of *RAD* based on the *CMBC* threat model as a function of the adversarial cost budget based on the distance-based cost function (we present the results based on the equivalence-based cost function in appendix D). In general, the *RAD* framework based on *CMBC* performs robustly compared with other baseline methods even there are high cost budgets; while the traditional learning algorithms or optimized algorithms, which have taken adversarial strategies into account, will fail to detect rapidly when the cost budget increases.

Figure 7(a) shows the performance of logistic regression, with and without retraining, on Enron and MNIST based on continuous feature. The increased robustness of *RAD* based on the *BECS* threat model is immediately evident: performance of *RAD* is essentially independent of λ on all three measures, and substantially exceeds baseline algorithm performance for small λ. Interestingly, we observe that the baseline algorithms are significantly more fragile to evasion attacks on Enron data compard to MNIST: benign and malicious classes seem far easier to separate on the latter than the former. This qualitative comparison between the Enron and MNIST datasets is consistent with other classification methods as well (SVM, NN). These results also illustrate that the neural-network classifiers, in their baseline implementation, are significantly more robust to evasion attacks than the (generalized) linear classifiers (logistic regression and SVM): even with a relatively small attack cost attacks become ineffective relatively quickly, and the differences between the performance on Enron and MNIST data are far smaller. Throughout, however, *RAD* significantly improves robustness to evasion, maintaining extremely high accuracy, precision, and recall essentially independently of λ, datasets, adversarial models, and algorithms used.



(a) (b) (c)

Fig. 6. Performance of baseline (*adv-*) and *RAD* (*rob-*) based on the *CMBC* threat model as a function of cost budget B_c based on the distance-cost function for Enron dataset based on 2000 binary features testing on adversarial instances. (a) logistic regression, (b) SVM, (c) 3-layer NN.



(a) (b) (c)

Fig. 7. Performance of baseline (*adv-*) and *RAD* (*rob-*) based on the *BECS* threat model as a function of cost sensitivity λ for Enron (top) and MNIST (bottom) datasets with continuous features testing on adversarial instances. (a) logistic regression, (b) SVM, (c) 3-layer NN.

Fig. 8. Example modification of digit images (MNIST data) as λ decreases (left-to-right) for logistic regression, SVM, 1-layer NN, and 3-layer NN (rows 1-4 respectively).

In Figure 8 we visualize the relative vulnerability of the different classifiers, as well as effectiveness of our general-purpose evasion methods based on coordinate greedy. Each row corresponds

to a classifier, and moving right within a row represents decreasing λ (allowing attacks to make more substantial modifications to the image in an effort to evade correct classification). We can observe that NN classifiers require more substantial changes to the images to evade, ultimately making these entirely unlike the original. In contrast, logistic regression is quite vulnerable: the digit remains largely recognizable even after evasion attacks.

Considering now datasets with binary features, we use the Enron data with a bag-of-words feature representation, for a total of 2000 features. We compare Naive Bayes (NB), logistic regression, SVM, and a 3-layer neural networks. Our comparison involves both the baseline, and *RAD* implementations based on the *BECS* threat model. Figure 9 confirms the effectiveness of *RAD* based on discrete features: every algorithm is substantially more robust to evasion with retraining, compared to baseline implementation. Most of the algorithms can obtain extremely high accuracy on this data with the bag-of-words feature representation. However, a 3-layer neural network is now *less* robust than the other algorithms, unlike in the experiments with continuous features. Indeed, Goodfellow et al. [2014] similarly observe the relative fragility of NN to evasion attacks.

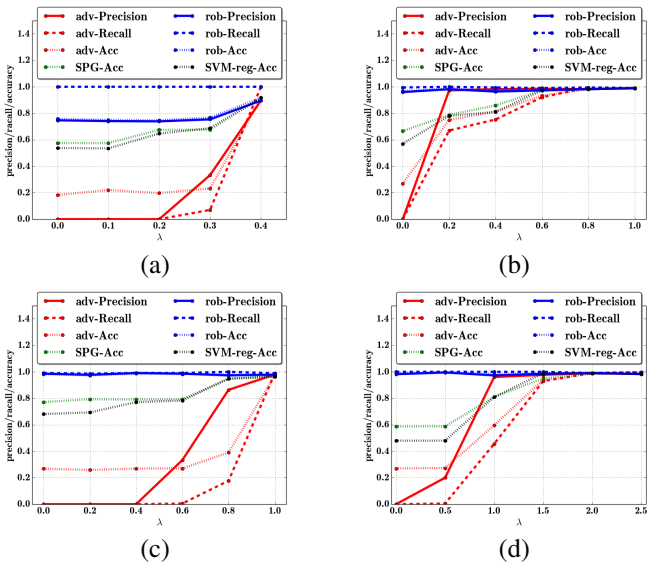


Fig. 9. Performance of baseline (*adv-*) and *RAD* (*rob-*) based on *BECS* implementations of (a) Naive Bayes, (b) logistic regression, (c) SVM, and (d) 3-layer NN, using binary features testing on adversarial instances.

7.2.3. Experiments with Multi-class Classification. To evaluate the effectiveness of *RAD*, and resilience of baseline algorithms, in multi-class classification settings, we use the MNIST dataset and aim to correctly identify digits based on their images. Our comparison involves SVM and 3-layer neural network (results for NN-1 are similar). We use $M = \{1, 4\}$ as the malicious class (that is, instances corresponding to digits 1 and 4 are malicious), and $T = \{2, 7\}$ is the set of benign labels (what malicious instances wish to be classified as). The results, shown in Figure 10 are largely consistent with our previous observations: both SVM and 3-layer NN perform well when retrained with *RAD*, with near-perfect accuracy despite adversarial evasion attempts. Moreover, *RAD* significantly boosts robustness to evasion, particularly when λ is small (adversary who is not very sensitive to evasion costs).

Figure 11 offers a visual demonstration of the relative effectiveness of attacks on the baseline implementation of SVM and 1- and 3-layer neural networks. Here, we can observe that a significant

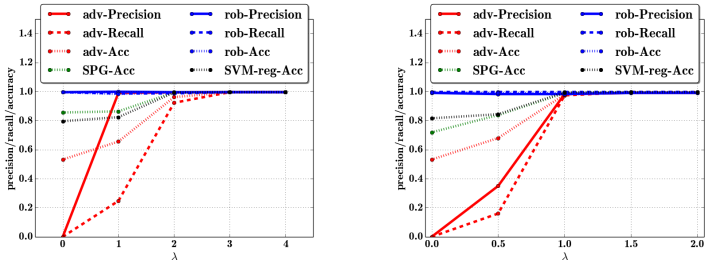


Fig. 10. Performance of baseline (*adv-*) and RAD (*rob-*) based on *BECS* implementations of (a) multi-class SVM and (b) multi-class 3-layer NN, using MNIST dataset testing on adversarial instances.

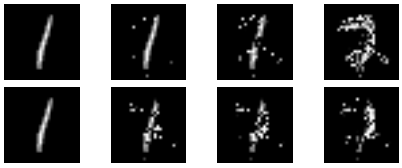


Fig. 11. Visualization of modification attacks with decreasing the cost sensitivity parameter λ (from left to right), to change 1 to the set $\{2,7\}$. The rows correspond to SVM and 3-layer NN, respectively.

change is required to evade the linear SVM, with the digit having to nearly resemble a 2 after modification. In contrast, significantly less noise is added to the neural networks in effecting evasion.

7.2.4. Oracles based on Human Evasion Behavior for RAD. To evaluate the considerable generality of *RAD*, we now use instances both generated by a non-optimization-based threat model, and from the observed human evasion behavior *in human subject experiments*. The data for this evaluation was obtained from the human subject experiment by [Ke et al. 2016] in which subjects were tasked with the goal of evading an SVM-based spam filter, manipulating 10 spam/phishing email instances in the process. In these experiments, Ke et al. developed a model of human subject evasion behavior. We now adopt this model as the evasion oracle, $\mathcal{O}$, injected in our *RAD* retraining framework, executing the synthetic model for 0-10 iterations to obtain evasion examples.

Figure 12(a) shows the recall results for the dataset of 10 malicious emails (the classifiers are trained on Enron data, but evaluated on these 10 emails, including evasion attacks). Figure 12(b) shows the classifier performance for the Enron dataset by applying the synthetic adversarial model as the oracle. We can make two high-level observations. First, notice that human adversaries appear

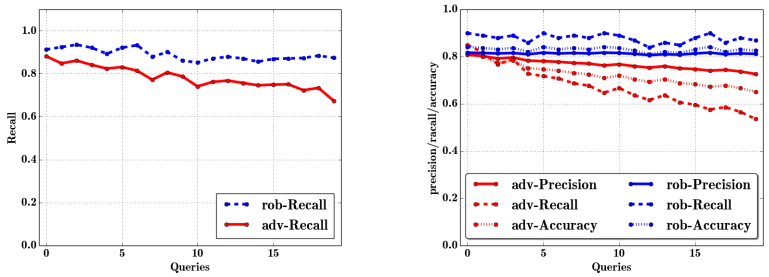


Fig. 12. *RAD* (*rob-*) and baseline SVM (*adv-*) performance based on human subject behavior data over 20 queries, (a) using experimental data with actual human subject experiment submissions, (b) using Enron data and a synthetic model of human evader.

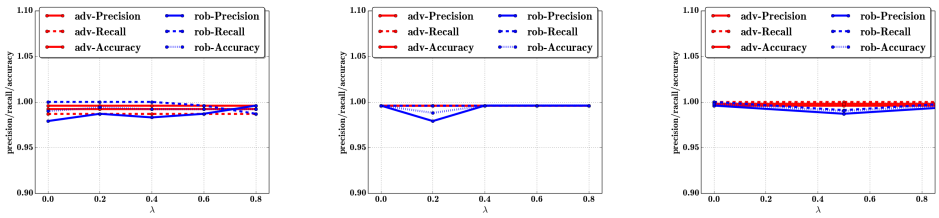


Fig. 13. Performance of baseline (*adv-*) and *RAD* (*rob-*) based on *BECS* as a function of cost sensitivity λ for MNIST dataset with continuous features testing on non-adversarial instances. (a) logistic regression, (b) SVM, (c) 3-layer NN.

significantly less powerful in evading the classifier than the automated optimization-based attacks we previously considered. This is a testament to both the effectiveness of our general-purpose adversarial evaluation approach, and the likelihood that such automated attacks likely significantly overestimate adversarial evasion risk in many settings. Nevertheless, we can observe that the synthetic model used in *RAD* leads to a significantly more robust classifier. Moreover, as our evaluation used actual evasions, while the synthetic model was used only in training the classifier as a part of *RAD*, this experiment suggests that the synthetic model can be relatively effective in modeling behavior of human adversaries. Figure 12(b) shows a more systematic study using the synthetic model of adversarial behavior on the Enron dataset. The findings are consistent with those only considering the 10 spam instances: retraining significantly boosts robustness to evasion, with classifier effectiveness essentially independent of the number of queries made by the oracle.

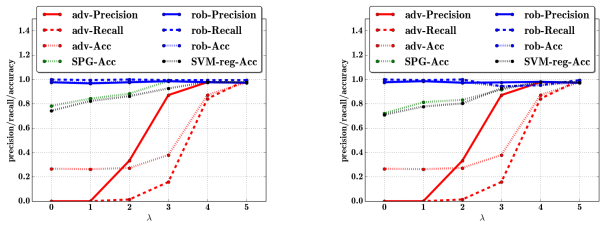
7.2.5. Evaluation of Robustness of RAD . While *RAD* is quite general and admits nearly arbitrary adversary models, it still requires a *specific* adversary model as an oracle. All models are ultimately approximation to real adversarial behavior, which can be complex and is not well understood. A key question, therefore, is how robust *RAD* is to inaccurate assumptions about adversarial models it uses. An orthogonal but equally important question is how well *RAD* performs when in fact no evasion attacks are present: in other words, how much classification performance do we sacrifice by trying to be robust to evasion attacks? We now explore these questions.

Performance of *RAD* in Non-Adversarial Environments: In order to explore whether *RAD* sacrifices accuracy when no adversary is present, Figure 13 shows the performance of the baseline algorithms and *RAD* on a test dataset sans evasions. Surprisingly, *RAD* is never significantly worse, and in some cases better than non-adversarial baselines: adding malicious instances appears to increase overall generalization ability. This is also consistent with the observation by [Kantchelian et al. 2015].

Robustness to Cost Function and Parameter Misspecification: Our evaluation of RAD robustness to adversarial model misspecification consists of two parts: first, we consider a case in which the defender and attacker use different cost functions, and second, when the defender assumes that the attacker has a lower tolerance for evasion cost than the attacker actually does. In both cases, we use the *BECS* model.

Figure 14 shows the evaluation results based on the Enron dataset with binary features. This figure compares efficacy of *RAD* when it correctly assumes that the attacker uses a quadratic cost function (a), and in the case when it assumes the attacker's cost to be exponential, while the attacker actually uses a quadratic cost (b). We can see that the two plots look quite similar, with *RAD* nearly equally effective in both cases.

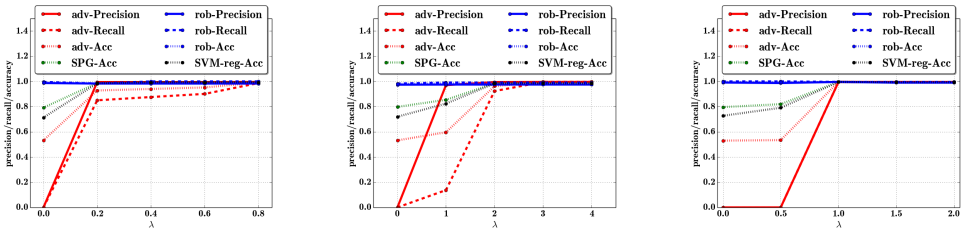
Next, we evaluate robustness of *RAD* when the defender's estimate of the parameter λ which determines the tradeoff the attacker makes between evading the classifier and evasion cost when under-estimates attacker's evasion aggressiveness. In Figure 15 we show the classification results as a function of the attacker's value of λ when the defender uses the immediately larger λ in the



(a) (b)

Fig. 14. Performance of baseline (*adv-*) and *RAD* (*rob-*) based on the *BECS* adversarial model as a function of cost sensitivity λ for Enron dataset with binary features, after evasion attacks. (a) both defender and adversary use the quadratic distance cost function, and (b) adversary uses quadratic cost function while defender estimates it based on exponential cost.

discretized space. We can see that even when the defender underestimate the attacker's evasion aggressiveness, *RAD* retains most of its efficacy in being robust to evasion attacks.



(a) (b) (c)

Fig. 15. Classification performance when defender and attacker use different values of λ for (a) logistic regression, (b) SVM, (c) 3-layer NN.

8. CONCLUSIONS

We consider an adversarial evasion problem, formalized as *adversarial loss minimization* in which adversaries, corresponding to maliciously labeled instances, modify the corresponding feature vectors so as to evade being classified as malicious. We follow the general approach to adversarial evasion introduced by Lowd and Meek [2005] in which evasion behavior represents a tension between replacing an "ideal" feature vector (representing behavior the adversary actually wishes to follow) with another that is misclassified as benign, and reducing the total modification made to this ideal feature vector. We distill a number of models that have been proposed to capture this tradeoff into a very general framework, which we specialize into two general optimization problems for the adversary, one following closely the Lowd and Meek [2005] framework in which deviation cost from the ideal feature vector is minimized subject to a hard evasion constraint, and another in which the optimization problem involves an explicit tradeoff between evasion success (in terms of distance from classification boundary, representing how "benign" the instance appears to the classifier) and cost.

We offer two solutions to the evasion problem for both adversarial models. The first is a principled and general Stackelberg game multi-adversary model (*SMA*), solved using mixed-integer linear programming, under the assumption of linear classifiers and l_1 regularization. The second is *RAD*, a general-purpose systematic retraining algorithm against evasion attacks for arbitrary classifiers (used as a "black-box") and arbitrary oracle-based (or "black-box") evasion models. We show that *RAD* minimizes an upper bound on optimal adversarial risk. Our experiments demonstrate that the first solution outperforms state-of-the-art adversarial classification methods, often notably. Experimentally, we showed that the performance of *RAD* is nearly indistinguishable from optimal, whereas

scalability is dramatically improved compared to *SMA*: indeed, with *RAD*, we are able to easily scale the approach to thousands of features, whereas *SMA* scales only to dozens of features.

An important challenge in all adversarial learning approaches to date is the specific assumptions they make on adversarial behavior. While *RAD* makes few specific requirements on the adversarial model, it still requires *some* adversarial model to be used in training. We therefore experimentally evaluate how robust *RAD* is when this model does not represent actual adversarial behavior. Our experiments indeed demonstrate considerable robustness of *RAD* to several model misspecifications. We believe that the most significant strength of *RAD* is that it can make use of arbitrary learning algorithms essentially "out-of-the-box", and effectively and quickly boost their robustness to nearly arbitrary evasion attack models, in contrast to most prior adversarial learning methods which are algorithm-specific.

ACKNOWLEDGMENTS

This research was supported in part by the National Science Foundation (IIS-1649972, IIS-1526860, CNS-1640624, CNS-1238959), Army Research Office (W911NF-16-1-0069), Office of Naval Research (N00014-15-1-2621), Air Force Research Laboratory (FA8750- 14-2-0180), the National Institutes of Health (R01LM10207), Symantec Research Labs Graduate Fellowship, and Sandia National Laboratories.

REFERENCES

Ion Androutsopoulos, John Koutsias, Konstantinos V Chandrinos, George Paliouras, and Constantine D Spyropoulos. 2000. An evaluation of naive bayesian anti-spam filtering. *arXiv preprint cs/0006013* (2000).

Ion Androutsopoulos, Evangelos F Magirou, and Dimitrios K Vassilakis. 2005. A Game Theoretic Model of Spam E-Mailing.. In *CEAS*.

Marco Barreno, Peter L Bartlett, Fuching Jack Chi, Anthony D Joseph, Blaine Nelson, Benjamin IP Rubinstein, Udam Saini, and J Doug Tygar. 2008. Open problems in the security of learning. In *Proceedings of the 1st ACM workshop on Workshop on AISec*. ACM, 19–26.

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

Battista Biggio, Giorgio Fumera, and Fabio Roli. 2014. Security evaluation of pattern classifiers under attack. *Knowledge and Data Engineering, IEEE Transactions on* 26, 4 (2014), 984–996.

Stephen Boyd and Lieven Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press.

Michael Brückner and Tobias Scheffer. 2009. Nash equilibria of static prediction games. In *Advances in neural information processing systems*. 171–179.

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

Xavier Carreras and Lluis Marquez. 2001. Boosting trees for anti-spam email filtering. *arXiv preprint cs/0109015* (2001).

William W Cohen. 2009. Enron email dataset. (2009).

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

Laurent El Ghaoui, Gert René Georges Lanckriet, Georges Natsoulis, and others. 2003. *Robust classification with interval data*. Computer Science Division, University of California.

Tom Fawcett. 2003. In vivo spam filtering: a challenge problem for KDD. *ACM SIGKDD Explorations Newsletter* 5, 2 (2003), 140–148.

Tom Fawcett and Foster Provost. 1997. Adaptive fraud detection. *Data mining and knowledge discovery* 1, 3 (1997), 291–316.

Amir Globerson and Sam Roweis. 2006. Nightmare at test time: robust learning by feature deletion. In *Proceedings of the 23rd international conference on Machine learning*. ACM, 353–360.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

Joshua Goodman, Gordon V. Cormack, and David Heckerman. 2007. Spam and the ongoing battle for the inbox. *Commun. ACM* 50, 2 (2007), 25–33.

Zoltan Gyongi and Hector Garcia-Molina. 2005. Spam: It's not just for inboxes anymore. *Computer* 38, 10 (2005), 28–34.

Stephen Hinde. 2003. Spam: the evolution of a nuisance. *Computers & Security* 22, 6 (2003), 474–478.

Peter J Huber. 2011. *Robust statistics*. Springer.

A. Kantchelian, J. D. Tygar, and A. D. Joseph. 2015. Evasion and Hardening of Tree Ensemble Classifiers. arXiv pre-print. (2015).

Christoph Karlberger, Günther Bayler, Christopher Kruegel, and Engin Kirda. 2007. Exploiting Redundancy in Natural Language to Penetrate Bayesian Spam Filters. *WOOT* 7 (2007), 1–7.

Liyiming Ke, Bo Li, and Yevgeniy Vorobeychik. 2016. Behavioral Experiments in Email Filter Evasion. In *AAAI Conference on Artificial Intelligence*.

Michael Kearns and Ming Li. 1993. Learning in the presence of malicious errors. *SIAM J. Comput.* 22, 4 (1993), 807–837.

Bryan Klimt and Yiming Yang. 2004. The enron corpus: A new dataset for email classification research. In *Machine learning: ECML 2004*. Springer, 217–226.

Marius Kloft and Pavel Laskov. 2012. Security analysis of online centroid anomaly detection. *The Journal of Machine Learning Research* 13, 1 (2012), 3681–3724.

Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. 2017. Adversarial Machine Learning at Scale. In *International Conference on Learning Representations*.

Anukool Lakhina, Mark Crovella, and Christophe Diot. 2004. Diagnosing network-wide traffic anomalies. In *ACM SIGCOMM Computer Communication Review*, Vol. 34. ACM, 219–230.

Pavel Laskov and Richard Lippmann. 2010. Machine learning in adversarial environments. *Machine learning* 81, 2 (2010), 115–119.

Yann LeCun and Corinna Cortes. 2010. MNIST handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist* (2010).

Bo Li and Yevgeniy Vorobeychik. 2014. Feature cross-substitution in adversarial classification. In *Advances in Neural Information Processing Systems*. 2087–2095.

Bo Li and Yevgeniy Vorobeychik. 2015. Scalable Optimization of Randomized Operational Decisions in Adversarial Classification Settings. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*. 599–607.

M. Lichman. 2013. UCI Machine Learning Repository. (2013). http://archive.ics.uci.edu/ml

Wei Liu and Sanjay Chawla. 2009. A game theoretical model for adversarial learning. In *Data Mining Workshops, 2009. ICDMW'09. IEEE International Conference on*. IEEE, 25–30.

Wei Liu and Sanjay Chawla. 2010. Mining adversarial patterns via regularized loss minimization. *Machine Learning* 81, 1 (2010), 69–83.

Daniel Lowd and Christopher Meek. 2005. Adversarial learning. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 641–647.

Matthew V Mahoney and Philip K Chan. 2002. Learning nonstationary models of normal network traffic for detecting novel attacks. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 376–385.

Garth P McCormick. 1976. Computability of global solutions to factorable nonconvex programs: Part IConvex underestimating problems. *Mathematical Programming* 10, 1 (1976), 147–175.

Vangelis Metsis, Ion Androutsopoulos, and Georgios Paliouras. 2006. Spam filtering with naive bayes-which naive bayes?. In *CEAS*. 27–28.

B. Nelson, B. Rubinstein, L. Huang, A. Joseph, S. Lee, S. Rao, and J. D. Tygar. 2012a. Query strategies for evading convex-inducing classifiers. *Journal of Machine Learning Research* 13 (2012), 1293–1332.

Blaine Nelson, Benjamin IP Rubinstein, Ling Huang, Anthony D Joseph, Steven J Lee, Satish Rao, and JD Tygar. 2012b. Query strategies for evading convex-inducing classifiers. *The Journal of Machine Learning Research* 13, 1 (2012), 1293–1332.

Blaine Nelson, Benjamin IP Rubinstein, Ling Huang, Anthony D Joseph, and JD Tygar. 2011. Classifier evasion: Models and open problems. In *Privacy and Security Issues in Data Mining and Machine Learning*. Springer, 92–98.

James Newsome, Brad Karp, and Dawn Song. 2006. Paragraph: Thwarting signature learning by training maliciously. In *Recent advances in intrusion detection*. Springer, 81–105.

Anh Nguyen, Jason Yosinski, and Jeff Clune. 2015. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 427–436.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. 2016a. Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples. *arXiv preprint arXiv:1605.07277* (2016).

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 2016b. Practical Black-Box Attacks against Deep Learning Systems using Adversarial Examples. *arXiv preprint arXiv:1602.02697* (2016).

Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. 2016c. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 372–387.

Manoj Parameswaran, Huaxia Rui, and S Sayin. 2010. A game theoretic model and empirical analysis of spammer strategies. In *Collaboration, Electronic Messaging, AntiAbuse and Spam Conf*, Vol. 7.

Leif E Peterson. 2009. K-nearest neighbor. *Scholarpedia* 4, 2 (2009), 1883.

James Pita, Milind Tambe, Chris Kiekintveld, Shane Cullen, and Erin Steigerwald. 2011. GUARDS: game theoretic security allocation on a national scale. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems, 37–44.

Anirudh Ramachandran and Nick Feamster. 2006. Understanding the network-level behavior of spammers. *ACM SIGCOMM Computer Communication Review* 36, 4 (2006), 291–302.

Anirudh Ramachandran, Nick Feamster, and Santosh Vempala. 2007. Filtering spam with behavioral blacklisting. In *Conference on Computer and Communications Security*. 342–351.

Justin M. Rao and David H. Reiley. 2012. The Economics of Spam. *Journal of Economic Perspectives* 26, 3 (2012), 87–110.

Eran Reshef and Eilon Solan. 2006. The effects of anti-spam methods on spam mail. In *Conference on Email and Anti-Spam*.

Benjamin IP Rubinstein, Blaine Nelson, Ling Huang, Anthony D Joseph, Shing-hon Lau, Satish Rao, Nina Taft, and JD Tygar. 2009. Antidote: understanding and defending against poisoning of anomaly detectors. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*. ACM, 1–14.

Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J Fleet. 2015. Adversarial manipulation of deep representations. *arXiv preprint arXiv:1511.05122* (2015).

Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. 1998. A Bayesian approach to filtering junk e-mail. In *Learning for Text Categorization: Papers from the 1998 workshop*, Vol. 62. 98–105.

C. Smutz and A. Stavrou. 2012. Malicious PDF detection using metadata and structural features. In *Annual Computer Security Applications Conference*. 239–248.

Nedim Srndic and Pavel Laskov. 2013. Detection of Malicious PDF Files Based on Hierarchical Document Structure. In *Annual Network & Distributed System Security Symposium*.

Pedro Tabacof and Eduardo Valle. 2015. Exploring the space of adversarial images. *arXiv preprint arXiv:1510.05328* (2015).

Choon Hui Teo, Amir Globerson, Sam T Roweis, and Alex J Smola. 2007. Convex Learning with Invariances.. In *NIPS*, Vol. 20. 1489–1496.

MohamadAli Torkamani and Daniel Lowd. 2013. Convex Adversarial Collective Classification. In *Proceedings of The 30th International Conference on Machine Learning*. 642–650.

David E Tyler. 2008. Robust statistics: Theory and methods. *J. Amer. Statist. Assoc.* 103, 482 (2008), 888–889.

Dimitrios K Vassilakis, Ion Androutsopoulos, and Evangelos F Magirou. 2007. A Game-Theoretic Investigation of the Effect of Human Interactive Proofs on Spam E-mail.. In *Conference on Email and Anti-Spam*.

Shobha Venkataraman, Avrim Blum, and Dawn Song. 2008. Limits of learning-based signature generation with adversaries. (2008).

Yevgeniy Vorobeychik and Bo Li. 2014. Optimal randomized classification in adversarial settings. In *International Conference on Autonomous Agents and Multiagent Systems*.

David Wagner. 2004. Resilient aggregation in sensor networks. In *Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*. ACM, 78–87.

Huan Xu, Constantine Caramanis, and Shie Mannor. 2009. Robustness and regularization of support vector machines. *Journal of Machine Learning Research* 10, Jul (2009), 1485–1510.

KONG Ying and ZHAO Jie. 2012. Learning to Filter Unsolicited Commercial E-Mail. *International Proceedings of Computer Science & Information Technology* 49 (2012).

Fei Zhang, Patrick PK Chan, Battista Biggio, Daniel S Yeung, and Fabio Roli. 2015. Adversarial feature selection against evasion attacks. (2015).

Yan Zhou and Murat Kantarcioglu. 2014. Adversarial Learning with Bayesian Hierarchical Mixtures of Experts. In *SIAM International Conference on Data Mining*. 929–937.

Yan Zhou, Murat Kantarcioglu, Bhavani Thuraisingham, and Bowei Xi. 2012. Adversarial support vector machine learning. In *SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1059–1067.

APPENDIX

B. COMPARISON BASED ON DIFFERENT EQUIVALENCE CLASS SIZES

To demonstrate the impact of feature cross-substitution attacks, we show comparisons for NB, SVM with linear kernel, SVM with rbf kernel and Neural Network classifiers based on the baseline Distance-based 16 (a) and the Equivalence-based 16 (b)-(d) cost function with Enron data.

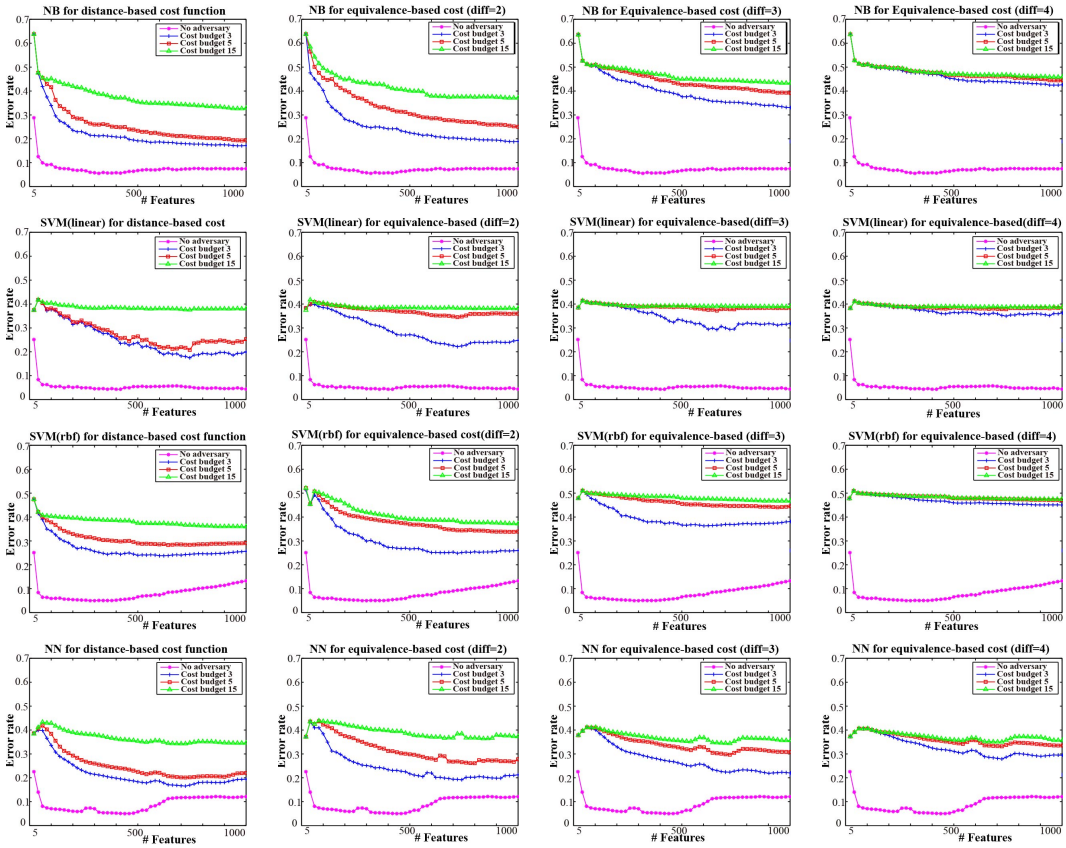


Fig. 16. Impacts of different equivalence class sizes for (a) Distance-based cost function, (b) Equivalence-based cost function with max-2-letter substitution, (c) Equivalence-based cost function with max-3-letter substitution, (d) Equivalence-based cost function with max-4-letter substitution.

For the equivalence-based cost function, we applied max-2,3,4-letter substitution respectively to form equivalence classes with increasing sizes. From the comparison results in Figure 16, it is obvious that the feature cross-substitution attacks elevate the test error on a large scale, and such attack gains more power when the equivalence class size increases.

C. GENERAL INSTANCE SUBSTITUTION ALGORITHM FOR EQUIVALENCE-BASED COST FUNCTION

Here we simulate the behavior of an adversary as running an algorithm $FindBooleanIMAC(x^A, x^-)$ to substitute features from the "ideal" instance x^A based on

ALGORITHM 6: $FindBooleanIMAC(x^A, x^-)$

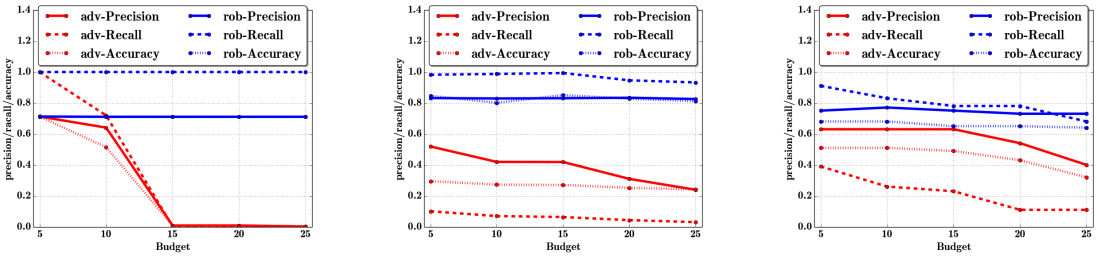
```
y ← x^-
flag ← false
repeat
  y^prev ← y
  for all i ∈ C_y do
    if F_i ∩ C_y = ∅ or MatchClass(i, C_y) ≤ 0 then
      toggle i in y
      if c(y) = 1 then
        toggle i in y
      end if
    end if
  end for
  count ← 0
  for all i_1 ∉ C_y, i_2, i_3 ∈ C_y do
    randomly choose i_1 ∉ C_y, i_2, i_3 ∈ C_y and i_2 ≠ i_3
    if F_{i_2} ∩ C_y = ∅ and F_{i_3} ∩ C_y = ∅; or MatchClass(i_2, C_y) ≤ 0 and MatchClass(i_3, C_y) ≤ 0
    then
      toggle i_1, i_2, i_3 in y
      count ← count + 1
      if c(y) = 1 then
        toggle i_1, i_2, i_3 in y
        count ← count − 1
      end if
    end if
  end for
  if flag and count > 0 then
    flag ← false
  end if
  if count = 0 and flag = false then
    flag ← true
    for all i_1 ∉ C_y, i_2 ∈ C_y, i_3 ∈ C_y do
      toggle i_1, i_2, i_3 in y
      if c(y) = 1 then
        toggle i_1, i_2, i_3 in y
      end if
    end for
  end if
until y^prev = y
return y
```

an arbitrary ham instance x^- to generate the alternative instance x' for the adversary. This is a generalization of the one proposed by Lowd and Meek, which is run only based on the distance-based cost function, to support our proposed equivalence-based cost function.

Here $c(y)$ represents the classifier, which maps the input to malicious (1) or benign (0). Within the algorithm, function $MatchClass(i, C_v)$ is used to help decide whether it is possible for a feature $i \in C_v$ to be substituted by the others from its class F_i, which leads to no cost. Here C_v denotes the vector contains features with different values in v and x^A. We employ $MatchClass(i, C_v)$ to guarantee that the number of original substitutable pairs from x^A would not decrease, which leads to cost as 0. This means we would only change features in C_y that cannot be substituted by features within its class. $MatchClass(i, C_v) = \sum_{j \in F_i \cap C_v} \mathbb{1}\{f_i \oplus f_j = 1\} - \sum_{j \in F_i \cap C_v} \mathbb{1}\{f_i \oplus f_j = 0\}$.

D. *RAD* BASED ON EQUIVALENCE-BASED COST FUNCTION FOR *CMBC*

If the feature space is continuous, the *RAD* framework based on *CMBC* can be solved optimally using standard convex optimization methods [Boyd and Vandenberghe 2004]. If the feature space is binary and $f(x)$ is linear or convex-inducing, while algorithms proposed by [Lowd and Meek 2005] and [Nelson et al. 2012b]. Figure 17 shows the performance of *RAD* based on the optimized adversarial strategies based on binary features for various learning models, respectively.

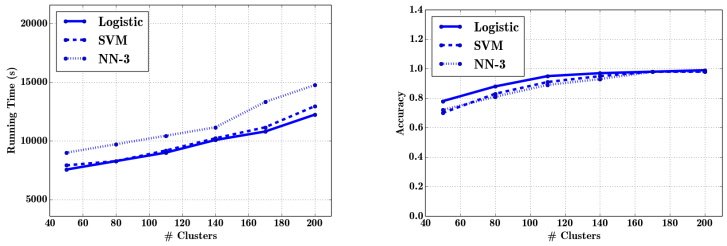


(a) (b) (c)

Fig. 17. Performance of baseline (*adv-*) and *RAD* (*rob-*) based on the *CMBC* threat model as a function of cost budget B based on the equivalence-cost function for Enron dataset based on 2000 binary features testing on adversarial instances. (a) logistic regression, (b) SVM, (c) 3-layer NN.

E. EXPERIMENTS FOR CLUSTERING MALICIOUS INSTANCES

To efficiently speed up the proposed algorithm, here we cluster the malicious instances and use the center of each cluster to generate the potential "evasion" instances for the retraining framework. Figure 18 shows that the running time can be reduced by applying the clustering algorithm to the original malicious instances and the classification performance stays pretty stable for different learning models.



(a) (b)

Fig. 18. Performance of different learning models based on the number of clusters for Enron dataset testing on adversarial instances. (a) Running time, (b) classification accuracy of *RAD*.