Classifier Evasion: Models and Open Problems

Blaine Nelson[1], Benjamin I.P. Rubinstein[2], Ling Huang[3],
Anthony D. Joseph[1,3], and J.D. Tygar[1]

[1] UC Berkeley
[2] Microsoft Research
[3] Intel Labs Berkeley

Abstract. As a growing number of software developers apply machine learning to make key decisions in their systems, adversaries are adapting and launching ever more sophisticated attacks against these systems. The near-optimal evasion problem considers an adversary that searches for a low-cost negative instance by submitting a minimal number of queries to a classifier, in order to effectively evade the classifier. In this position paper, we posit several open problems and alternative variants to the near-optimal evasion problem. Solutions to these problems would significantly advance the state-of-the-art in secure machine learning.

Keywords: Query Algorithms, Evasion, Reverse Engineering, Adversarial Learning.

1 Introduction

A number of systems and security engineers have proposed the use of machine learning techniques for detecting or filtering miscreant activities in a variety of applications, *e.g.*, spam, intrusion, virus, and fraud detection. All known detection algorithms have *blind spots*: classes of miscreant activity that fail to be detected. While learning enables the detector to adapt, adversaries can still exploit blind spots to evade detection. A significant challenge is to quantify how effectively an adversary can discover blind spots by querying the detector.

Consider, for example, a spammer who wishes to minimally modify a spam message so it is not classified as spam by a public webmail system's spam classifier. His cost is measured by the change in the value of the modified message. The spammer can observe the classifier's behavior by creating a dummy account on the webmail system. By observing the responses of the spam detector to queries, the spammer can search for a minimal modification. Similarly for host-based intrusion detection, an intruder may be forced to obfuscate an attack to avoid detection. The attacker can alter their exploit by inserting *no-ops*, using synonymous system calls, or even choosing between one of several possible exploits. Successful intrusions can also be observed by the attacker.

While instances have associated costs to the adversary, he also incurs a cost for the queries needed to search for evading instances. Sending a large number of queries requires additional resources and may arouse suspicion to the malicious

C. Dimitrakakis et al. (Eds.): PSDML 2010, LNAI 6549, pp. 92–98, 2011.

behavior. In practice, adversaries employ a number of techniques to mitigate this second cost, such as opening multiple webmail accounts or rate-limiting spam or worm probing attacks; these strategies require the attacker to use increased sophistication and computational resources.

Here we revisit the near-optimal evasion problem—a theoretical formulation that quantifies how effectively a classifier can be evaded through querying. We outline open problems and introduce novel variants of the original formulation.

1.1 The Near-Optimal Evasion Problem

The *Near-Optimal Evasion Problem* is a formulation of adversarial evasion that quantifies the hardness of search for low-cost negative instances in terms of the number of queries used. As first posed by Lowd and Meek [6] under the name *adversarial classifier reverse-engineering*, the near-optimal evasion problem quantifies the query complexity required by an adversary to find a negative instance with near-minimal cost in terms of the size of the feature space and the desired accuracy. By analyzing the query complexity for a family of classifiers, near-optimal evasion provides a notion of how hard that family is to evade; *e.g.*, if a spammer must send exponentially-many queries in the dimensionality of the feature space, it is difficult for the spammer to successfully improve his spam.

In this setting, we assume instances are represented in a D-dimensional feature space (*e.g.*, $\mathcal{X} \subseteq \mathbb{R}^D$) and the target classifier f belongs to a family $\mathcal{F}$ of binary classifiers where each classifier $f \in \mathcal{F}$ is a mapping from feature space $\mathcal{X}$ to a label in $\{\text{'}-\text{'}, \text{'}+\text{'}\}$. A deterministic $f \in \mathcal{F}$ partitions $\mathcal{X}$ into two sets—the positive class $\mathcal{X}_f^+ = \{\mathbf{x} \in \mathcal{X} \mid f(\mathbf{x}) = \text{'}+\text{'}\}$ and the analogous negative class $\mathcal{X}_f^-$, which we take to be the *normal* instances. We assume that the adversary is aware of at least one instance in each class, $\mathbf{x}^- \in \mathcal{X}_f^-$ and $\mathbf{x}^A \in \mathcal{X}_f^+$, knows $\mathcal{F}$ and not f, and can observe $f(\mathbf{x})$ for any $\mathbf{x} \in \mathcal{X}$ by issuing a *membership query.*

Adversarial Cost. We assume the adversary has a cost function $A : \mathcal{X} \to \mathbb{R}^{0+}$; *e.g.*, for a spammer this could be edit distance on messages. The adversary wishes to optimize A over the negative class $\mathcal{X}_f^-$; *e.g.*, the spammer wants to send spam that will be classified as normal email ('$-$'). Typically the cost function is a distance to a target instance $\mathbf{x}^A \in \mathcal{X}_f^+$ that is most desirable to the adversary; *e.g.*, an ℓ_p distance induces cost $A_p(\mathbf{x}) = \left\|\mathbf{x} - \mathbf{x}^A\right\|_p$.

Lowd and Meek [6] define *minimal adversarial cost (MAC)* of a classifier f to be the best lower bound on the cost that any negative instance obtains

$$MAC(f, A) \triangleq \inf_{\mathbf{x} \in \mathcal{X}_f^-} [A(\mathbf{x})] \quad . \tag{1}$$

They further define a data point to be an ϵ-approximate *instance of minimal adversarial cost (ϵ-IMAC)* if it is a negative instance with cost no more than a factor of $(1 + \epsilon)$ times the *MAC*; *i.e.*, every ϵ-*IMAC* is a member of the set

$$\epsilon\text{-}IMAC(f, A) \triangleq \left\{\mathbf{x} \in \mathcal{X}_f^- \;\middle|\; A(\mathbf{x}) \leq (1 + \epsilon) \cdot \mathrm{MAC}(f, A)\right\} \quad . \tag{2}$$

The goal of the *Near-Optimal Evasion Problem* is to quantify the worst-case query complexity required for an adversary to infer an ϵ-*IMAC*, depending on the richness of the family of classifiers. Finding an ϵ-*IMAC* for a singleton family can be achieved offline without any queries, whereas if the family is large, many more queries will be necessary to find an ϵ-*IMAC*. Formally,

> A family of classifiers $\mathcal{F}$ is ϵ-*IMAC searchable* under a family of cost functions $\mathcal{A}$ if for all $f \in \mathcal{F}$ and $A \in \mathcal{A}$, there is an algorithm that finds $\mathbf{x} \in \epsilon\text{-}IMAC\,(f, A)$ using polynomially many membership queries in D and $L_\epsilon = \log \frac{1}{\epsilon}$. We will refer to such an algorithm as *efficient.*

1.2 Security and the Near-Optimal Evasion Problem

The near-optimal evasion problem sheds light on several real-world security issues. The problem abstracts the scenario of an adversary who wishes to launch a specific attack that is blocked by a classifier-based defense. The attacker has a limited number of probing opportunities after which he must send an attack as close as possible to his originally intended attack—a *near-optimal attack.*

In the case of email spam, the spammer may originally have a message that will be detected as spam. He probes, finds a near-optimal message that evades the filter, and sends this message instead. In the case of an intruder, he has a preferred sequence of system calls that will be detected as intrusions. He probes, finds and executes a near-optimal sequence that evades the detector.

With this framework in mind, we now clearly see the role of a defender: to provide a classifier that resists near-optimal evasion. Practical implementation requires careful selection of costs and realistic bounds on the number of probes an adversary can perform. Resulting lower-bounds on the number of probes required for near-optimal evasion provide significant evidence of effective security.

1.3 Previous Work

Lowd and Meek [6] first introduced near-optimal evasion and developed efficient methods that reverse-engineer linear classifiers in both real-valued and Boolean feature spaces for ℓ_1 costs. Nelson *et al.* [7] generalized the result from linear classifiers to the family of *convex-inducing classifiers* that partition the space of instances into two sets one of which is convex. In generalizing to this family, they showed that near-optimal evasion does not require an estimate of the classifier's decision boundary or state. Nelson *et al.* [8] further explored general ℓ_p costs and found not all are ϵ-*IMAC* searchable for convex-inducing classifiers.

Dalvi *et al.* use a cost-sensitive game-theoretic approach to preemptively patch a classifier's blind spots [5]. They construct a modified classifier designed to detect optimally modified instances. Biggio *et al.* [3] extend this game-theoretic approach and propose hiding information or randomization as additional defense mechanisms for this setting. However, they do not explore near-optimal evasion for randomized classifiers as we propose in Section 3.3.

2 Open Problems in the Theory of Near-Optimal Evasion

A number of unanswered questions remain about the near-optimal evasion problem. Here we motivate these problems and suggest potential directions.

Question 1: *Can we find matching upper and lower bounds for evasion algorithms? Is there a deterministic strategy with polynomial query complexity for all convex-inducing classifiers?* In previous work, linear and convex-inducing classifiers were shown to be ϵ-*IMAC* searchable for ℓ_1 costs by demonstrating algorithms with polynomial query complexity. Currently, it is known that for convex positive class, $\mathcal{O}\left(L_\epsilon + \sqrt{L_\epsilon} D\right)$ queries are sufficient to find a near-optimal instance, although the tightest known lower bound in this case is $\mathcal{O}\left(L_\epsilon + D\right)$. In the case of convex $\mathcal{X}_f^-$, the best known algorithm is a randomized ellipsoid approach (*cf.* [2]) that finds a near-optimal instance with high probability using $\mathcal{O}^*\left(D^5\right)$ queries (ignoring logarithmic terms).

Question 2: *Is there some family larger than the convex-inducing classifiers that is ϵ-IMAC searchable? Are there families outside of the convex-inducing classifiers for which near-optimal evasion is efficient?* Existing approaches to near-optimality have built on the machinery of convex optimization. However, many interesting classifiers are not convex-inducing classifiers. Currently, the only such known result (due to Lowd and Meek) is that linear classifiers on Boolean instance space are 2-*IMAC* searchable.

Question 3: *Is some family of SVMs (*e.g.*, with a known kernel) ϵ-IMAC searchable for some ϵ? Can an adversary incorporate the structure of a non-convex classifier into the ϵ-IMAC search?* Consider SVMs with non-linear kernels. The classifier is non-convex in the original feature space, while the classifier is linear in its Reproducing Kernel Hilbert Space but the cost function may no longer be easy to minimize in that space. However SVMs have other properties that may facilitate near-optimal evasion. For instance, in cases where there are few support vectors, one only needs to find these instances to reconstruct the classifier.

Question 4: *Are there characteristics of non-convex, contiguous bodies that are indicative of the hardness of the body for near-optimal evasion? What about non-contiguous bodies?* It appears that the family of contiguous bodies (*i.e.*, the set of all classifiers for which either $\mathcal{X}_f^+$ or $\mathcal{X}_f^-$ is a contiguous set) cannot be generally ϵ-*IMAC* searchable since this family includes members with many locally minimal cost regions which are hard for local search or binary search procedures to avoid, but perhaps some subsets of this family are ϵ-*IMAC* searchable. For families of non-contiguous bodies, ϵ-*IMAC* searchability seems impossible to achieve (disconnected components could be arbitrarily close to $\mathbf{x}^A$) unless the classifiers' structure can be exploited; *e.g.*, as we discuss for SVMs above.

Question 5: *For what classes of classifiers is reverse-engineering as easy as evasion?* Reverse-engineering is the process of querying to learn the decision boundary, and is sufficient for solving the evasion problem. It is now known that the query complexity of reverse-engineering linear classifiers is identical to that of evasion, while reverse-engineering is strictly more difficult for general convex-inducing classifiers [8]. It is unknown whether there exists a class in between linear and convex-inducing classifiers on which the two tasks are equivalent.

QUESTION 6: *Is there a relationship between the query complexity required for evasion and traditional measures of classifier complexity such as the* VC-dimension*?* Currently, there is no known relationship between these complexity measures for a family of classifiers. Finding such a relationship could answer many of the previous questions about what classifiers can be evaded.

3 Alternative Models for Evasion

Here we suggest a number of variants of near-optimal evasion that generalize or reformulate the original problem to capture new aspects of the overall challenge.

3.1 Additional Information about Training Data Distribution

Consider an adversary that knows the training algorithm and obtains samples drawn from a natural distribution. A few interesting settings include:

1. The adversary's samples are a subset of the training data.
2. The adversary's samples are from the same distribution as the training data.
3. The adversary's samples are from a perturbation of the training distribution.

With this additional information, the adversary may estimate their own classifier $\tilde{f}$ and analyze it offline. Some open questions include:

QUESTION 7: *What can be learned from $\tilde{f}$ about f? How can $\tilde{f}$ best be used to guide search? Can the sample data be directly incorporated into ϵ-IMAC-search?* Relationships between between f and $\tilde{f}$ can build on existing results in learning theory. A possibility is to bound the difference between $MAC(f, A)$ and $MAC(\tilde{f}, A)$ in one of the above settings. If the difference is sufficiently small with high probability, then a search for an *ϵ-IMAC* could use $MAC(\tilde{f}, A)$ to initially lower bound $MAC(f, A)$. This should reduce query complexity since lower bounds on the MAC are typically harder to obtain than upper bounds.

3.2 Beyond the Membership Oracle

QUESTION 8: *What types of additional feedback may be available to the adversary and how do they impact the query complexity of ϵ-IMAC-search?* In this scenario, the adversary receives more from the classifier than just a '+'/'−' label. For instance, suppose the classifier is defined as $f(\mathbf{x}) = \mathbb{I}\{g(\mathbf{x}) > 0\}$ for some real-valued function g (as is the case for SVMs) and the adversary receives $g(\mathbf{x})$ for every query instead of $f(\mathbf{x})$. If g is linear, the adversary can use $D + 1$ queries and solve a linear regression problem to reverse-engineer g. This additional information may also be useful for approximating the support of an SVM.

3.3 Evading Randomized Classifiers

In this variant of near-optimal evasion, we consider randomized classifiers that generate random responses from a distribution conditioned on the query $\mathbf{x}$. To analyze the query complexity of such a classifier, we must first generalize our concept of the *MAC*. We propose the following candidate generalization:

$$RMAC\left(f, A\right) = \inf_{\mathbf{x} \in \mathcal{X}_f^-} \left\{ A\left(\mathbf{x}\right) + \lambda \mathbb{P}\left(f\left(\mathbf{x}\right) = \text{'}-\text{'}\right) \right\} \ .$$

If f is deterministic, we need $\lambda \geq MAC\left(f, A\right)$ for this definition to be equivalent to Eq. (1) (*e.g.*, $\lambda = A\left(\mathbf{x}^A\right) + 1$ is sufficient); otherwise, a trivial minimizer for $\lambda < MAC\left(f, A\right)$ is $\mathbf{x}^A$. For a randomized classifier, λ balances cost with probability of success.

Question 9: *Given access to the membership oracle only, how difficult is near-optimal evasion of randomized classifiers? Are there families of randomized classifiers that are ϵ-IMAC searchable?* Potential randomized families include:

1. Classifiers with fuzzy boundary of width δ around a deterministic boundary
2. Classifiers based on the class-conditional densities for a pair of Gaussians, a logistic regression model, or other members of the exponential family.

Evasion of randomized classifiers seems to be more difficult than for deterministic classifiers as each query provides limited information about the query probabilities. Based on this argument, Biggio *et al.* promote randomized classifiers as a defense against evasion [3]. However, the query complexity for evasion of randomized classifiers is currently unknown.

3.4 Querying with Real-World Objects

Question 10: *How can the feature mapping be inverted to design real-world instances to map to desired queries? How can query algorithms be adapted for approximate querying?* In the original model of evasion, it was assumed that the attacker could observe $f\left(\mathbf{x}\right)$ for any $\mathbf{x} \in \mathcal{X}$. This capability implicitly assumes that the attacker has knowledge of the feature mapping from real-world objects (*e.g.*, emails or packets) into the learner's feature space. Not only is this mapping often unknown, it also need not be one-to-one or onto: multiple emails may map to the same bag-of-words vector, and some instances in feature space may not correspond to *any* real-world object. This significantly complicates evasion for real-world adversaries.

3.5 Evading an Adaptive Classifier

Finally we consider a classifier that periodically retrains on queries. This variant is a multi-fold game between the attacker and learner, with the adversary now able to issue queries that degrade the learner's performance. Techniques from game-theoretic online learning should be well-suited to this setting [4].

Question 11: *Given a set of adversarial queries (and possibly additional innocuous data) will the learning algorithm converge to the true boundary or can* *the adversary deceive the learner and evade it simultaneously? If the algorithm does converge, at what rate?* To properly analyze retraining, it is important to have an oracle that labels the points sent by the adversary since the adversary can also query in $\mathcal{X}_f^-$. If all points sent by the adversary are labeled '+', the classifier may prevent effective evasion, but with a large number of false positives due to the adversary's queries in $\mathcal{X}_f^-$; this itself would constitute an attack against the learner [1].

4 Conclusion

The intersection of security, systems, and machine learning research has yielded significant advances in decision-making for complex systems, but has also introduced new challenges in protecting against malicious users. While earlier research laid the groundwork for understanding the near-optimal evasion problem, fundamental problems remain unaddressed. In this paper, we discussed several of these problems and variants, and proposed potential avenues for future research.

References

1. Barreno, M., Nelson, B., Joseph, A.D., Tygar, J.D.: The security of machine learning. Machine Learning 81(2), 121–148 (2010)
2. Bertsimas, D., Vempala, S.: Solving convex programs by random walks. J. ACM 51(4), 540–556 (2004)
3. Biggio, B., Fumera, G., Roli, F.: Multiple classifier systems under attack. In: El Gayar, N., Kittler, J., Roli, F. (eds.) MCS 2010. LNCS, vol. 5997, pp. 74–83. Springer, Heidelberg (2010)
4. Cesa-Bianchi, N., Lugosi, G.: Prediction, Learning, and Games. Cambridge University Press, Cambridge (2006)
5. Dalvi, N., Domingos, P., Sanghai, S., Verma, D.: Adversarial classification. In: Proc. KDD 2004, pp. 99–108 (2004)
6. Lowd, D., Meek, C.: Adversarial learning. In: Pro. KDD 2005, pp. 641–647 (2005)
7. Nelson, B., Rubinstein, B.I.P., Huang, L., Joseph, A.D., Lau, S., Lee, S., Rao, S., Tran, A., Tygar, J.D.: Near-optimal evasion of convex-inducing classifiers. In: Proc. AISTATS 2010, pp. 549–556 (2010)
8. Nelson, B., Rubinstein, B.I.P., Huang, L., Joseph, A.D., Lee, S.J., Rao, S., Tygar, J.D.: Query strategies for evading convex-inducing classifiers (2010) report arXiv:1007.0484v1 available at, `http://arxiv.org/abs/1007.0484`