Adversarial Active Learning

Brad Miller
UC Berkeley

Alex Kantchelian
UC Berkeley

Sadia Afroz
UC Berkeley

Rekha Bachwani
Intel Labs

Edwin Dauber
Drexel University

Ling Huang
DataVisor

Michael Carl Tschantz
UC Berkeley

Anthony D. Joseph
UC Berkeley

J. D. Tygar
UC Berkeley

ABSTRACT

Active learning is an area of machine learning examining strategies for allocation of finite resources, particularly human labeling efforts and to an extent feature extraction, in situations where available data exceeds available resources. In this open problem paper, we motivate the necessity of active learning in the security domain, identify problems caused by the application of present active learning techniques in adversarial settings, and propose a framework for experimentation and implementation of active learning systems in adversarial contexts. More than other contexts, adversarial contexts particularly need active learning as ongoing attempts to evade and confuse classifiers necessitate constant generation of labels for new content to keep pace with adversarial activity. Just as traditional machine learning algorithms are vulnerable to adversarial manipulation, we discuss assumptions specific to active learning that introduce additional vulnerabilities, as well as present vulnerabilities that are amplified in the active learning setting. Lastly, we present a software architecture, Security-oriented Active Learning Testbed (SALT), for the research and implementation of active learning applications in adversarial contexts.

Categories and Subject Descriptors

K.6 [**Management of Computing and Information Systems**]: Security and Protection; I.2 [**Artificial Intelligence**]: Learning; H.1 [**Models and Principles**]: User/Machine Systems

Keywords

Secure Machine Learning; Active Learning; Human in the Loop

1. INTRODUCTION

The scale and diversity of attacks demands that humans have machine assistance to identify malicious content in a timely fashion. For example, the Google anti-phishing platform received over 500 million URLs in four months [56]. Likewise, VirusTotal receives approximately 700,000 submissions of new, unique binaries each day[1] [50]. Further complicating matters, constantly evolving adversarial attempts to evade or corrupt current detectors require prompt and accurate labeling of new content to maintain accurate detection.

Active learning is a useful tool when limited resources are available to label training data and extract features. In this open problem paper, we argue that we must study the application of active learning techniques in an adversarial context. In real world applications the scale of malicious content prevents labeling in a timely fashion; but systems proposed in the academic literature frequently assume that accurate labels will be available for all data in a timely fashion. This divergence with reality presents challenges to the deployment of these systems and diminishes general faith in their efficiency. Industry had addressed this gap through introduction of active learning techniques, as evidenced in an area survey stating: "software companies and large-scale research projects such as CiteSeer, Google, IBM, Microsoft, and Siemens are increasingly using active learning technologies in a variety of real-world applications" [42]. Google has published at least two papers identifying its use of active learning approaches to detect malicious advertisements and phishing pages [41, 56]. Because the real world has adversaries, each of these organizations has developed its own proprietary techniques for dealing with adversaries. However we believe that it is important to explore large-scale adversarial active machine learning techniques in open academic research so that all users of active learning may benefit.

There is limited research considering active learning approaches in the presence of an adversary. The introduction of active learning techniques may introduce or exacerbate vulnerabilities in the following ways:

- **Selection of Instances for Labeling** To achieve accurate classification, active learning approaches require ongoing selection of instances which will improve performance when added to training data. We posit that an adversary may be able to introduce instances which appear appealing to the selection algorithm but have little impact on, or even degrade, classification accuracy. Additionally, we believe the adversary may be capable of accomplishing this manipulation even when the adversarial instances are correctly labeled.

[1]Based on a series of observations in July and August 2014.

AISec'14, November 7, 2014, Scottsdale, Arizona, USA.

ACM 978-1-4503-3153-1/14/11 ...$15.00.
http://dx.doi.org/10.1145/2666652.2666656.

- **Oracle Maliciousness** Active learning traditionally models the human as an oracle that can provide a correct label for any given instance in a constant amount of time. In an adversarial context, a human may be able to corrupt the labeling process by not always providing accurate labels.
- **Exacerbation of Present Vulnerabilities** The introduction of active learning, in recognition of the reality that not all samples can be expertly labeled, creates additional opportunity for attackers to exploit presently known machine learning vulnerabilities. By structuring samples to either appeal to or avoid the query algorithm, the attacker may either control a larger portion of the training data or decrease the likelihood of attack instances being included in training data.

We examine each of these vulnerabilities in greater depth, as well as the benefits and challenges of applying active learning. In Section 2 we describe the background and basic setup of active learning and in Section 3 we motivate the application of active learning in security contexts. Section 4 discusses potential vulnerabilities of active learning to adversarial activity, and Section 5 presents experimental topics and a scalable software framework for studying and developing active learning systems for security applications. In Section 6 we discuss related work and in Section 7 we conclude.

2. BACKGROUND: ACTIVE LEARNING

Machine learning operates over a set of *instances*, such as software binaries. A *classifier* is an algorithm that predicts a label, such as benign or malicious, assigned to an instance. A classifier is accurate when it predicts the correct label.

Traditionally, a classifier is trained using a learning algorithm that starts by consuming a *training set* of labeled instances (instances with their correct label supplied). The algorithm then trains a classifier based on information in the training set.

Active learning is a form of machine learning in which the learning algorithm actively engages an *oracle*, such as a human labeler, to request information in addition to the original training set. (See [42] for a survey.) Most commonly, it requests that the oracle label an instance of its selection. The learner employs a *query strategy* to select the instance for labeling. In some cases, the active learner may be free to select any instance, even ones that do not occur in the data set. In other cases, the learner is limited to some pool of observed but unlabeled instances.

Active learning is useful in cases where there are many unlabeled examples, but human labeling is expensive. This situation is common in many security applications. For example, machine learning could automate malware detection and other attack detection, but training a model with sufficient accuracy requires a large number of labeled instances and labeling is expensive. Active learning supports the use of various strategies to prioritize human labeling.

2.1 Query Strategies

The approach we focus on in our examples is *uncertainty sampling*. This strategy is applicable every time a model returns a meaningful real-valued score along with the predicted label. For such models, the most uncertain sample is the one which receives the most uncertain score. For example, in linear support vector machines, the score is taken to be the distance of the instance to the separating hyperplane, also called the margin. In this context, the most uncertain scores are the smallest in absolute value.

Lewis *et al.* first introduced the uncertainty sampling method in the context of text classification [28] but it has since been successfully applied to information extraction tasks [13, 43], a domain with abundant unlabeled data. Overall, it is both a popular and easy to implement method.

Other notable query strategies include *density-based* [43], *query-by-committee* [44] and *variance reduction* methods [12]. We refer the interested reader to [42] for a complete overview.

2.2 Extensions

Here we consider extensions to the simple model of active learning just presented.

Oracle Accuracy. Using active learning in an adversarial setting raises the concern that the adversary might compromise the oracle. For example, consider a system for flagging inappropriate images using crowdsourcing for labeling the instances. Such a system could be vulnerable to some of the labelers adversarially mislabeling the instances. While these types of questions have been examined in reputation systems [23, 32, 58] and crowdsourcing [27], they remain an open issue in the context of active learning.

Noisy oracles, which may be present in both adversarial and non-adversarial settings, present a related challenge for active learning. While traditional active learning techniques are not designed to be robust against noisy labeling, *agnostic active learning* is designed with this in mind [3, 5, 16]. However, these works make assumptions, such as independent and identically distributed data, that may not hold in an adversarial setting. Although these works acknowledge an adversary as a potential source of noise, along with fundamental randomness in the data set or a misfit in labeling, none of them test agnostic active learning as a defense against noise based attacks on an active learning system.

Feature Extraction Costs. The cost of altering and measuring features is an additional consideration in the adversarial setting. Adversaries may attempt to disguise their malicious instances in manners that do not decrease their effectiveness or incur other costs. Defenders prefer to identify malicious instances features that can be measured with low cost. Prior research on adversarial machine learning has studied these issues using game theory [8, 14, 24, 51] or as a problem of reverse engineering classifiers [30, 36]. Other works have studied the cost of features in the setting of active learning during either training [33] or during testing [9, 20, 29, 62].

Batch Learning. Traditional active learning research uses a sequential model in which each iteration selects a single unlabeled sample to be labeled by the oracle and added to the training data, followed by retraining after every iteration. In applications where training is expensive this model is impractical. Recent research has been interested in a *batch mode* model in which each iteration selects a subset of the unlabeled samples to have labeled by the oracle and added to the training data, with retraining occurring after all instances in the subset have been labeled and entered into the training set [2, 10, 11, 21].

Feedback on Features. Active learners can query oracles for information other than just labels or feature values.

Raghavan, Madani, and Jones proposed an active learning system which in each iteration queries the oracle on both an instance and a list of features [37, 38]. The oracle needs to provide feedback both on what class the instance belongs to and the relevance of each of the listed features. They found that humans can rate the relevance of features more quickly than they can instances, and that rating features can improve the accuracy of the system.

3. ACTIVE LEARNING FOR SECURITY

Active learning offers an approach to handle a range of common phenomena in security applications, including large amounts of data, unlabeled or poorly labeled data and adversarial drift.

Machine learning systems in security deal with a large amount of data. For example, in the malware domain, different malware creation toolkits like Zeus [54], SpyEye [47], Dendroid [48], make the process of creating, customizing and re-packaging malware binaries easy. There has been a dramatic proliferation of new malware in the wild. In the first quarter of 2014 alone, two million mobile malware and high-risk apps were found [49]. Labeling new instances as quickly as possible is important for the security of a system but labeling every instance manually is impractical. We need ways to prioritize instances to be labeled that have the most impact on the dataset and classifier's performance.

The accuracy of a machine learning system depends on the quality of the training and test data. If the training data is polluted, the system will be unable to distinguish between malicious and benign instances. If the test data is polluted, a poorly performing model might be chosen. Collecting high quality labels can be costly as it requires expertise and time for a human labeler. For example, high quality labeling is one of bottlenecks of the malware classification. In 2009 the typical time window between a malware's release and its detection by AV software was 54 days and 15% of samples remain undetected after 180 days [15]. Active learning provides systematic strategies to better allocate human resources by identifying instances that have the most impact on the system's performance.

Unlike other domains, data in security applications suffer from adversarial drift. Drift refers to the non-stationarity of data where the data distribution changes over time. This drift can be natural gradual drift, for example, changes to a user's preference over time, or adversarial drift where an adversary changes the data to purposefully decrease the classification accuracy [14, 25, 53]. For example, using malware re-packaging toolkits, known as Fully Un-Detectable or FUD crypters, malware vendors repackage malware to evade anti-virus tools [7]. To handle drift and novel attacks the system needs to be periodically retrained, a process that requires labeling and model validation. Current active learning methods can handle regular drift by using randomization with uncertainty sampling [61], can handle noisy labels generated by nonadaptive adversaries [3], and by adaptive adversaries [18]. Yang considered such drift theoretically in a covariate shift setting [57]. Yang assumes that data has pre-fixed dimensions and goes through a fixed sequence of distribution changes with a bounded amount of drift. While theoretically appealing, it is unclear how these assumptions are relevant in the practical context of intelligent adversaries.

Several industrial classification systems are already using active learning approaches to handle these issues. For example, Google uses active learning approaches for labeling malicious advertisements and phishing pages [41, 56]. The Google anti-phishing system evaluates millions of potential phishing pages everyday. Google prioritizes potential phishing pages for human review based in part on PageRank, claiming that this type of active learning minimizes the instance labeling task.

While effective, active learning has its own limitations, especially in the presence of an adversary. We discuss these limitations in Section 4 and present a research agenda to improve and augment active learning methods for security applications in Section 5.

4. ADVERSARIAL ACTIVE LEARNING

Traditional machine learning developed in a setting which did not account for an adversary, and consequently suffers from a range of vulnerabilities [4]. Likewise, active learning has not traditionally considered adversarial activity, and hence is likely to have a range of vulnerabilities. In this section, we discuss several classes of likely vulnerabilities in active learning which must be better understood and addressed to robustly deploy active learning in security contexts. As these have received limited attention in the literature [60], they remain poorly understood and important areas for future research.

We explore three types of attacks on active learning. First, we consider attacks on how the active learner selects instances for querying the oracle. Second, we consider settings where an oracle may act maliciously. Finally, we consider how active learning can exacerbate known attacks on machine learning techniques.

4.1 Query Strategy Robustness

We first demonstrate on a simple example how an attacker can take considerable advantage of the query strategy.

Model setup. For visualization purposes, we work in a continuous two dimensional feature space where the aim is to learn a binary classifier. We suppose that the data is generated by the following underlying process: let $d > 0$, $\mathbf{X} \in \mathbb{R}^2$ an instance and $Y \in \{-1; 1\}$ its label. Let the data distribution be

$$p(\mathbf{X}, Y) = p(\mathbf{X}|Y)p(Y)$$

where $Y \sim \text{Bernoulli}(1/2)$ and $\mathbf{X} \sim (Yd/2, 0) + \mathcal{N}_2$. $\mathcal{N}_2$ is the bivariate normal distribution centered at the origin, with the identity covariance. In other words, this is a perfectly label-balanced task, where the positive instances are normally distributed around $(d/2, 0)$ and the negatives around $(-d/2, 0)$.

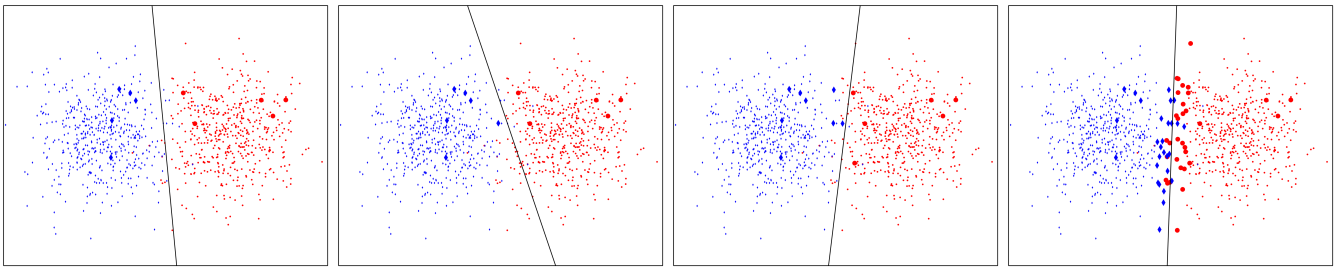
The aim is to learn a decision function $f : \mathbb{R}^2 \to \{-1, +1\}$ that performs well on the task of predicting the label Y of a given sample $\mathbf{X}$. Here, we will quantify the quality of f by its misclassification rate, or average number of misclassified samples. If the distribution $p(\mathbf{X}, Y)$ is available, one can compute this number as

$$R(f) = \mathbb{E}[1_{f(\mathbf{X}) \neq Y}]$$

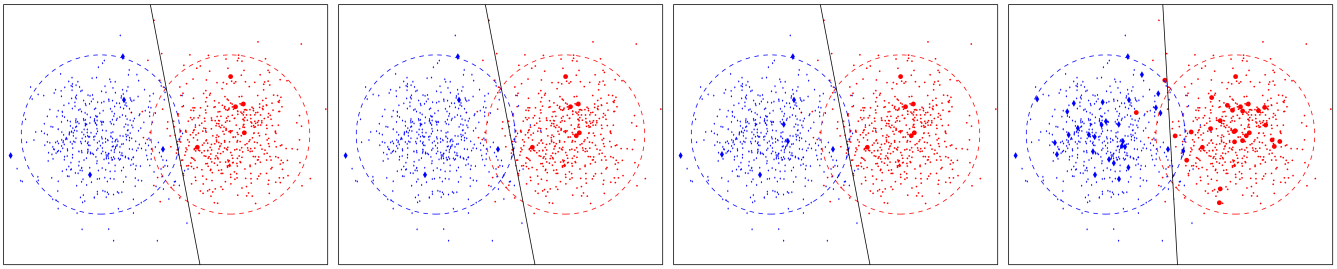
where R is known as the expected 0/1-loss, or risk.

In our case, it is easy to prove that the optimal decision rule f^* which minimizes R is a "vertical" linear separator

(a) Uncertainty query strategy, without attack. Models shown after initial sampling, 1, 4, and 50 queries.



(b) Random query strategy, with attack. Models shown after initial sampling, 1, 4, and 50 queries.



(c) Uncertainty query strategy, with attack. Models shown after initial sampling, 1, 2, and 50 queries.

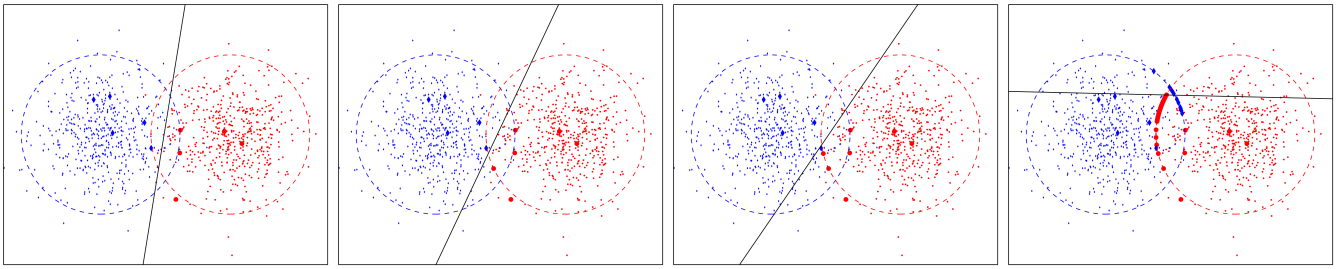


Figure 1: Model evolution during active learning. Selected points for training are drawn with larger blue diamond and red circle marks, for respectively negative and positive instances. For the attack, p is set to 5% and the corresponding 95% probability disks are shown.

coinciding with the Y axis:

$$f^*(\mathbf{X}) = \begin{cases} -1 & \text{if } \mathbf{X}_1 < 0 \\ +1 & \text{else} \end{cases}$$

$$R(f^*) = 1 - \Phi(d/2)$$

where Φ is the cumulative distribution function of the standard normal distribution.

In what follows, we chose d such that we know a good separator exists. Namely, we set $d = 4$, giving a best case misclassification rate $R(f^*) \approx 2.3\%$. We use a linear SVM for the underlying learning algorithm, so that the class of separators in which we perform our model search actually contains f^*.

We initially randomly draw both 500 positive and negative samples according to $p(\mathbf{X}|Y = \pm 1)$ and randomly reveal 5 positive and 5 negative labels, that is, we label 1% of the data set. This data set of 10 labeled instances serves to build the initial model. The learner then chooses an unlabeled data point according to the query strategy and reveals the true label. The new point is added to the training set and a new model is computed. The process continues until we have revealed the labeling of 5% of the data set; equivalent to 50 queries.

In deployed systems, 1% of initially labeled samples is not an unrealistic upper-bound when faced with hundreds of thousands of samples per day. Similarly, our choice of a 1,000 sample-size data set is in accordance with the low-dimensionality of the data, where both theory and practice show it is possible to approach excellent performance with very limited data. We further note that in our experiments, the size of the underlying data set has no importance, except in the case of the randomized strategy in the presence of an attacker.

We compare three active learning query strategies:

- a randomized strategy where an instance is selected uniformly at random among all unlabeled instances,
- an uncertainty strategy where the unlabeled point closest to the decision boundary is selected,
- a mixed strategy where we alternate between picking a point at random and selecting the closest point from round to round.

Attack setup. We suppose the attacker has the following capabilities:

1. The attacker is able to estimate with arbitrary precision the decision function at every round of the active learning.

2. The attacker knows the process generating the data, namely $p(\mathbf{X}, Y)$, but is not revealed the actual data set, nor the randomly chosen initial 10 points.

3. At every round, the attacker can inject at most one instance in the training set. The true label of the instance is consistent with $p(\mathbf{X}, Y)$ in the following sense. For a given $0 < p < 1$, the injected instance lies in the disk of probability $1 - p$ around the center of the normal distribution $p(\mathbf{X}|Y)$ corresponding to its label.

Assumption (1) reflects the fact that in practice, an adversary can repeatedly probe the system and obtain increasingly good descriptions of the decision boundary. Assumption (2) reflects the fact that the adversary has some general knowledge about how benign and malicious instances typically look like, without specifically knowing the data set used by the defender. Assumption (3) reflects the non-stationarity of the learning process in real life, where the attacker can forge new, either positive or negative samples every day. The p value effectively constrains the attacker's capacity to forge very outlying instances of a given label.

Assumptions (1) and (2) are beneficial in terms of model simplicity, but are potentially advantageous for the attacker. We however notice that one can arbitrarily weaken both by introducing exogenous parameters describing how much information the attacker possess about both the decision function (limited number of queries) and the probability distribution (limited number of randomly drawn samples) without changing the particular mechanics of the attack.

Attack strategy. Our attack is specifically designed for the maximum uncertainty (closest point) strategy. We suppose that the aim of the attacker is to maximally increase the learned model risk, under the previous assumptions. For the attacker to have any consequent influence on the active learning process, he must inject points that lie on, or arbitrarily close to, the decision boundary so that they get selected. This is necessarily so because the attacker is unaware of the defender's pool of samples, which might contain points that are already very close to the boundary.

Let $\mathcal{D}_t$ be the set of all labeled instances available for training at iteration t. Let a be the training algorithm which maps a labeled dataset to a classifier in $\{-1, +1\}^{\mathbb{R}^2}$. Formally, the attacker is interested in finding the instance $(\mathbf{x}, y)$ which maximizes the risk after retraining with it at iteration $t + 1$:

$$\max_{(\mathbf{x},y) \text{ as assum. (3)}} R\left(a\left(\mathcal{D}_t \cup \{(\mathbf{x}, y)\}\right)\right)$$

As the exact training set $\mathcal{D}_t$ is hidden, the attacker must approximate the effect of adding a new training instance on the risk of the model. To do so, the attacker is able to compute R for any model as per assumption (2), and knows f_t by assumption (1). A simple heuristic to estimate $R(f_{t+1})$ is then to perturb the current model f_t with the given training instance, and compute the risk of the resulting model. If $\mathbf{w}_t, \mathbf{w}_{t+1}$ denote the weight vectors corresponding to f_t, f_{t+1}, the attacker can approximate the new decision f_{t+1} by

$$\mathbf{w}_{t+1} \approx \mathbf{w}_t + \delta y \mathbf{x}$$

for some $\delta > 0$. Hence, to chose which point to inject, the attacker can now solve a simpler problem. Letting $\tilde{f}$ be the classifier associated with $\mathbf{w}_t + \delta y \mathbf{x}$, the attacker now solves

$$\max_{(\mathbf{x},y) \text{ as assum. (3)}} R(\tilde{f})$$

In our case of concern, we see that the possible solutions lie at the extremal points of the probability disks. If we further require the points to lie closest to the decision boundary, we have at most 4 candidate solutions to examine (the intersection of a line with two circles) and can do so efficiently. In our attack, we use the $\delta = 0.01$ and any small enough value of δ will give similar results.

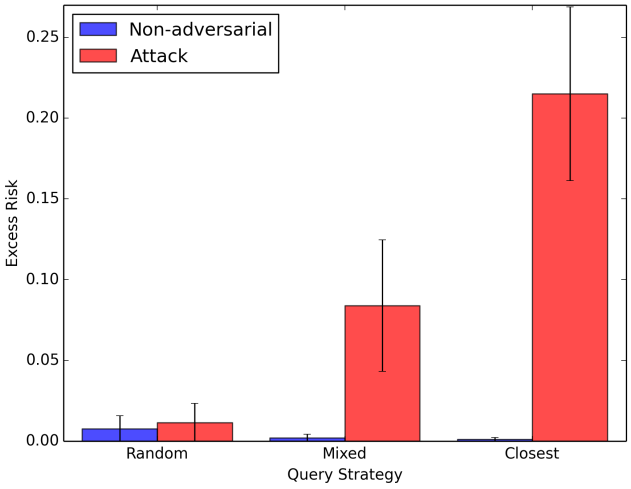


Figure 2: Excess Risk $R(f) - R(f^*)$ of learned models with different query strategies, with and without attack ($p = 5\%$). Average and standard errors for 50 runs per experiment are shown.

Results. For each configuration of the query strategy, we simulate the behavior of the system free of and under attack. We summarize our findings in Figure 2, which lists the excess risk of the final model, that is, the amount of risk that is in excess of the minimal achievable risk on the task. For the system which is not under attack, the maximum uncertainty strategy performs significantly better than random choice. When the system is under attack, randomization of training samples becomes the best strategy, while the maximum uncertainty choice suffers severe degradation. For both of these, the mixed query strategy achieves middle ground scores, but still incurs very high excess risk under attack. Figure 1 helps visualize how the attack unfolds when compared to a system free of attacks in the first row, and a system using randomized query strategy in the second row.

4.2 Trusted Oracle

Active learning research traditionally assumes an oracle that is able to label each submitted sample accurately and in a constant amount of time. With the rise of crowdsourcing, some work has relaxed the assumption of accurate labeling and considered noisy oracle models of increasing complexity. Initially, noisy oracle research began with a model in which

each oracle is correct with some uniform probability [45], and expanded to include models where each oracle has a different accuracy [55]. Prior work has also addressed practices for assigning labels under the assumption that oracles vary in accuracy and ground truth is unknown [17, 46].

Unfortunately, none of these models appropriately fit the adversarial context. In the adversarial context, we must consider that a labeler may be well behaved much of the time, yet make certain well placed mistakes designed to cause specific classification errors after a model is trained. With the rise of crowdsourcing, it is increasingly feasible for such malicious labelers to enter the system. For example, spam detection relies in part on users labeling messages as spam. Similarly, many blackhat SEO techniques effectively rely on crowdsourcing to manipulate the page rank algorithm.

4.3 ML Vulnerability Exacerbation

In Sections 1 and 3 we recognize the reality that not all available data can be labeled for training in a timely fashion. We advocate active learning techniques as a principled way of selecting the data which will be labeled and used for training. The explicit incorporation of the training data selection process into system design exposes heightened risk from presently known vulnerabilities in machine learning techniques. Traditional machine learning vulnerabilities are divided into two categories: *exploratory* vulnerabilities which focus on modifying samples once a model has been trained, and *causative* vulnerabilities which focus on modification of training data [22]. In this section, we discuss how active learning is likely to exacerbate both causative and exploratory vulnerabilities.

Exploratory Attacks. Lowd *et al.* present the *good words* attack against spam filters, in which words indicative of non-spam emails are added to spam emails to change classification results [31]. Although the good words attack was originally developed in the context of spam, the attack can be implemented in other contexts which have high-dimensional, sparse feature vectors. Lowd demonstrates that the good words attack is highly effective against a fixed model, but also finds that retraining the model with data including updated spam messages that contain the good words diminishes attack effectiveness.

The unfortunate reality, which is recognized by active learning, that timely labels will not be available for all possible training data complicates this otherwise effective defense. Effective retraining depends on the inclusion of spam messages continuing good words in the training corpus. However, if active learning fails to select the modified spam messages for labeling during the retraining process, defenses based on retraining will be ineffective.

Causative Attacks. Many attacks based on training data manipulation have been proposed in previous literature. The *dictionary* attack operates against spam filters by adding a broad range of words to spam emails to degrade the overall quality of classification and make the spam filter unusable [35]. The *focused* or *correlated outlier* attack also operates against spam filters by adding specific words expected to appear in specific legitimate emails to spam emails so that the legitimate emails are misclassified [22, 35]. Both of these attacks can easily be extended to non-spam settings. Lastly, the *red herring* attack introduces spurious features to malicious instances that are designed to induce the classifier to strongly associate the spurious features with maliciousness. Once the classifier has learned to associate the spurious features with malicious content, the attacker can remove the spurious features to prevent detection of malicious content [22].

Although the specifics of each of these attacks varies, each attack depends on the attacker controlling some portion of the data used to train the classifier. For many production scale applications of learning for security, such as spam filtering, PDF filtering or malware detection, it may not be feasible for an individual attacker to control a significant portion of the training data. However, as the unrealistic assumption that labels will be instantly available for all data is removed, the attacker becomes able to control a larger portion of the data through manipulation of the selection process. Active learning formalizes the selection process and will allow researches to better understand the viability of and defend against causative attacks.

5. REALIZATION OF ACTIVE LEARNING

This section describes a set of experimental topics that we believe are necessary for building an active learning system for the security domain. It also describes a software framework, SALT (Security-oriented Active Learning Testbed), that we are building to conduct those experiments. Table 1 lists the experimental topics, under the categories of drift, human integration, and query strategies.

5.1 Experimental Topics

Drift

Experimental Topic 1: Real-time Accuracy Monitoring. Perhaps the most important task surrounding a practical machine learning system is constantly monitoring its performance, not only in computational terms but also, and more importantly, in its classification accuracy. One can effectively do so by estimating the mean misclassification cost given a representative set of labeled instances. Unfortunately, since active learning incrementally builds a labeled data set, such a data set is unsuitable for evaluation purposes as it is dependently defined by the classifier to be evaluated. A naive solution would be to build a second labeled data set in parallel, which would be unbiased. Such an approach discards all information from the algorithm-defined labeled data set. Alternatively, the evaluator could create an all-new sample designed to efficiently estimate the system's performance [39, 40]. It would be interesting to explore middle grounds where this information is partially recycled for evaluation purposes in a safe, that is, unbiased manner. This may involve "uniformizing" the initially biased sampling in under-represented regions of the observation space, by querying for those labels as in [26].

Experimental Topic 2: Accuracy Degradation Over Time. The detailed investigation of the effects of non-stationary time-ordered data may bring a better understanding of the behavior of machine learning in our setting. In particular, fundamental questions about the nature of drift are related to the amount of information past data contains about yet to come observations. While it is established that classification performance degrades with time for a fixed model in adversarial applications [52], a fundamental question is to measure the system's confidence in its mistakes. In particular, low-confidence misclassifications may be a sign of

Drift	1. **Real-time Accuracy Monitoring.** The presence of drift combined with limited label availability complicates accuracy measurement on fresh data. We need parsimonious sampling strategies to provide both the labels requested by the active learning itself and those necessary for a sound evaluation of system performance. 2. **Accuracy Degradation Over Time.** By receiving data as an ordered stream, researchers should be able to observe the rate of decrease in model accuracy as training data and models become stale. 3. **Retraining Strategy.** Given that models will become stale with time, researchers should be able to experiment with different regiments for updating models. Policies could include retraining in response to changes in the data or elapsed time.
Human Integration	4. **Return on Human Effort.** Given the expense of human effort, we need a system to allow experimentation on both the return on fixed investments of human effort and policies for varied human resource allocation. 5. **Coping with Malicious Labels.** For some applications, plentiful, low-quality labels may be available through untrusted techniques such as crowdsourcing that may admit malicious labelers. We need a system to allow experimentation with attacks and defenses against malicious labelers. 6. **Identifying Malicious Labelers.** Given that malicious labelers may perform accurately in the vast majority of cases and make very selective errors, standard noisy models may be poorly suited to capture their malicious behavior.
Query Strategies	7. **Active Feature Acquisition.** In many cases, multiple forms of feature extraction with varied costs may be available (e.g. static and dynamic analysis of malware). We need a system to allow experimentation with varied policies for active feature acquisition. 8. **Query Strategy Performance.** Query strategies are vital to the system's ability to learn a concept and react to drift. We need a system to allow experimentation with varied query strategies. 9. **Query Strategy Robustness.** In addition to being effective, query strategies must be robust against adversarial manipulation. We need a system to allow researchers to experiment with different attack strategies.

Table 1: Specific experimental topics relating to active learning for security applications and in adversarial contexts.

mild drift, whereas many high-confidence misclassifications may indicate fundamentally unpredictable novelties.

Experimental Topic 3: Retraining Strategy. Because of the performance degradation incurred by fixed models, ongoing retraining is necessary. The best strategy is a function of many factors, including the amount of drift and the amount of available human work. In this regard, we are interested in comparing retraining strategies that are combinations of time-based and data-driven. For instance, periodic retraining might occur as a background process, while daily data volume and classification accuracy measures might drive additional retraining phases as necessary. Another interesting problem is to estimate the quality of the new model in absence of labeled future data. For example, it may be possible to predict the model's performance during its lifetime using data selected from recently observed instances.

Human Integration

Experimental Topic 4: Return on Human Effort. In a typical machine learning task, when we are progressively given more and more labeled training data, we often observe the accuracy of the trained model increases rapidly at first but then the accuracy flattens — and we see diminishing returns [6]. (We define the return as the amount of accuracy improvement per human effort, measured either by time or money.) Since human effort for labeling instances is costly, we do not want to waste human effort when the return is small. In our research agenda, we plan to develop query strategies that buck the trend of diminishing returns, so that we can improve model accuracy as much as possible before diminishing return occurs. We also develop methods that detect diminishing returns and stop acquiring costly labels as soon as possible.

Experimental Topic 5: Coping with Malicious Labels. Using crowdsourcing in the place of a trusted oracle creates the possibility of maliciously labeled instances. For example, product review on Amazon.com admits low-quality (and possibly malicious) labels [34]. To deal with such data, we need to develop robust active learning algorithms that are resilient to malicious labels, and to design active learning systems to cope with both benign errors and malicious attacks.

Experimental Topic 6: Identifying Malicious Labelers. Intelligent malicious labelers may collude together, perform accurately in the vast majority of cases, and make very selective errors. In such a case, standard noisy models are poorly suited to capture such adversarial behavior. We need to define new metrics to quantify the "harmfulness" of malicious labelers, and develop a framework and efficient

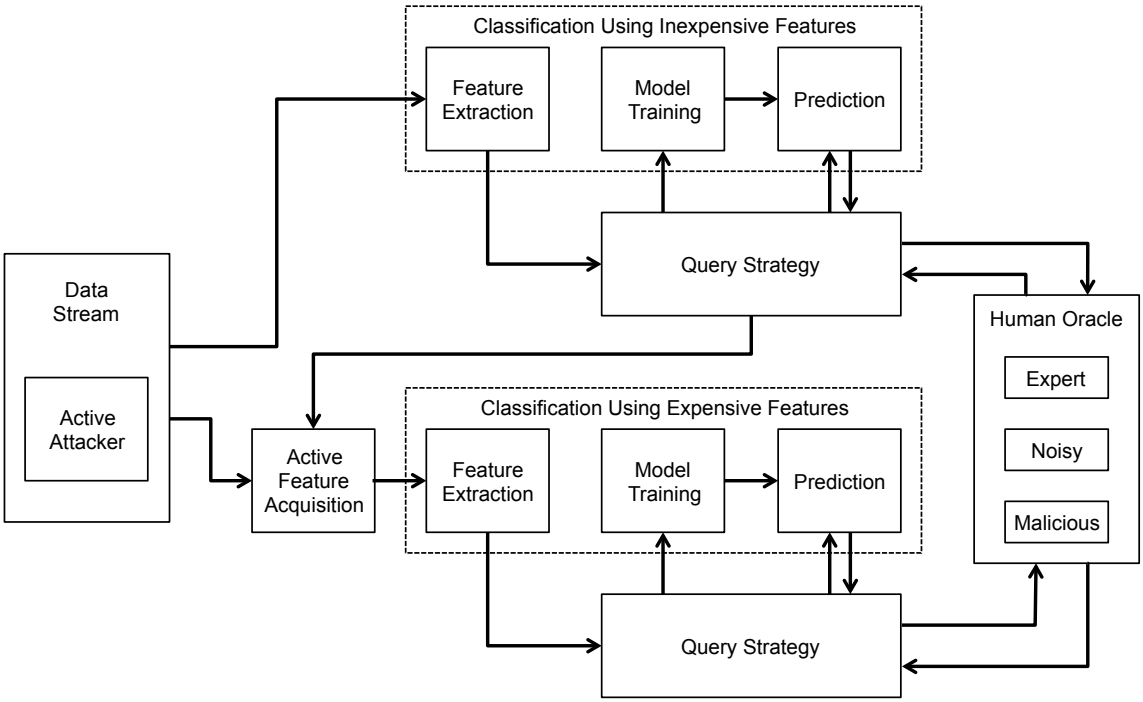


Figure 3: Security-oriented Active Learning Testbed (SALT): A software framework for classification of a stream of samples, including samples generated by an attacker in response to model state. SALT supports classification using both cheap and expensive features, with resource allocation governed by a query strategy that mediates access to and usage of the labeling oracle. We model the human oracle by appealing to the labels in the data set with perturbations added to model the effects of expert, noisy and malicious labelers.

algorithms to accurately detect malicious labelers. The system must be robust against malicious and strategic labelers, with performance slowly and gracefully degrading with the number of malicious agents.

Query Strategies

Experimental Topic 7: Active Feature Acquisition. Extraction of some features can be expensive. For example, computing the call graph of a binary may require dynamic analysis. Thus, we must be able to perform cost-benefit analyses to determine whether extracting a feature is worth the cost. By knowing how each feature impacts accuracy, a system could determine the most cost-effective features for each instance and utilize a query strategy to improve feature acquisition.

Experimental Topic 8: Query Strategy Performance. As discussed in Section 2, the active learner can select instances for labeling using numerous different methods, such as uncertainty sampling, density base strategies, query-by-committee, and variance reduction methods. The choice of a query strategy can impact the system's ability to find an accurate decision boundary and to quickly respond to drift. We must be able to test and monitor multiple query strategies to compare their effectiveness.

Experimental Topic 9: Query Strategy Robustness. In addition to responding to attacks against traditional learning algorithms, the query strategies themselves must be robust against manipulation, such as the attack discussed in Section 4.1. We must be able to stress-test query strategies under simulated attacks.

5.2 Security-oriented Active Learning Testbed

In this section, we present a software framework for engaging in the experimental topics discussed in Section 5.1. Our framework is the Security-oriented Active Learning Testbed, or SALT. Figure 3 shows a block diagram of the SALT architecture. At a high level, SALT provides for experimentation on time series data including modeling limited availability of labeling resources and feature acquisition options with varied costs. We discuss the role of each component of SALT in investigating the topics in Section 5.1, and then provide a brief discussion of a potential realization of SALT in a distributed computing environment.

Data Stream. This component houses the data set used for analysis. Preferably, a data set should have the following properties:

- Timestamp for each instance
- Distribution of instances reflecting density as observed in the real-world

Although these two properties can be difficult to achieve, they are not without significant benefit, particularly in the context of active learning. For example, note that a data set that is simply a collection of unique malware binaries will not have timestamp or distribution information. Timestamps are critical for the exploration of concepts related to drift, such as real-time accuracy monitoring (Experimental Topic 1), accuracy degradation (Experimental Topic 2), and retraining strategy (Experimental Topic 3). Likewise, a sample distribution reflective of the real world is necessary for

evaluating the effectiveness of query strategies that strive to place the highest emphasis on the most common threats.

Active Attacker. The attacker is located within the data stream to allow simulation of an attacker capable of fabricating samples designed to attack either the query selection processes or the traditional machine learning components of the system. We allow the attacker to insert attack instances into the data stream as the analysis progresses. As a matter of experimental design, the attacker could be given access to the present model state, the ability to request classification of specific samples, or no ability to interact with the present model at all. The active attacker module is useful for the study of query strategy robustness (Experimental Topic 9).

Classification Using Inexpensive/Expensive Features. These sections of the SALT system contain the basic components of a machine learning classification workflow, including feature extraction, model training and prediction. Multiple levels of feature extraction may improve resource utilization by reserving more expensive forms of analysis for instances with the greatest anticipated benefit. For example, in the context of malware a "cheap" application of static analysis may help to identify instances that represent known classes of malware. Then, a more expensive application of dynamic analysis can be applied to remaining instances to obtain classifications of greater certainty for less well-known malware. SALT's design containing two classification workflows supports active feature acquisition (Experimental Topic 7).

Query Strategy. The query strategy manages scarce resources in the system, including queries to the human oracle and active feature acquisition. The query strategy is responsible for selecting samples for both training and evaluation, and must balance resources between noisy and expert oracles. The query strategy also controls active feature acquisition by determining instances to prioritize for expensive feature extraction. By instrumenting the query strategy module and varying query behavior, SALT supports research on the return on human effort (Experimental Topic 4), the vulnerability to malicious oracles (Experimental Topic 5), the identification of malicious labelers (Experimental Topic 6), active feature acquisition (Experimental Topic 7), query strategy performance (Experimental Topic 8), and query strategy robustness (Experimental Topic 9).

Oracle Models. The oracle represents the involvement of a human to label data. SALT presents three oracle profiles corresponding to varied human resources; each profile may be instantiated multiple times with varied parameters (e.g. to vary accuracy as a function of oracle). We describe each of the profiles below.

- **Expert.** This oracle represents the highly accurate labeling supplied by expensive, technical experts. These experts may be used to check the work of and develop reputations for noisy oracles, to handle difficult instances, or to identify malicious oracles.
- **Noisy.** This oracle represents crowdsourced information, as may be obtained from user reviews, submissions (e.g. via a "spam" button), etc. A SALT implementation could allow multiple noisy oracles with accuracy varying as a function of oracle or instance.
- **Malicious.** This oracle represents labels supplied by a strategically malicious party. The malicious labeler could intentionally provide accurate labels for the majority of data while mislabeling specific, targeted samples. The malicious oracle component allows development of techniques for identifying and copying with malicious labelers (Experimental Topics 5 & 6).

We have begun implementation of the SALT framework with a design prioritizing scalability and interaction with data. We leverage the capabilities Spark, a system for distributed computation built on top of HDFS [59]. Spark offers several services critical to our design and not available in other distributed computation platforms. MLlib provides distributed implementations of common learning algorithms for rapid model generation over large amounts of data, as required by successive oracle queries and model retraining [1]. Spark also implements map-reduce functionality on data held in memory. Computation over the entire data (held in memory) can be initiated in an interactive fashion, allowing researchers quantify attack and drift behaviors using aggregate measurements over all instances.

6. RELATED WORK

While there has been much work on using machine learning in the face of an adversary (see [4] for an overview) and on active learning in general (see [42] for a survey), there has been relatively little work explicitly focused on active learning and security. Here, we focus on prior work related to using active learning in an adversarial setting.

Active Learning for Security. Active learning has appeared in high-impact production systems. Sculley et al. discuss Google's use of active learning to detect adversarial advertisements [41]. Google's system stratifies instances by language of the ad and whether the ad is new, recently blocked, or neither. They bin the ads from each strata by the probability of being adversarial. They randomly sample from each bin of each strata adjusting the number of samples from each to obtain a representative sampling across all assigned scores. For example, to behave similarly to uncertainty sampling, they may favor bins with probabilities near to 0.5.

Despite its use in practice, we have found little research using active learning for security applications. Görnitz et al. employ active learning in the setting of network intrusion detection [19]. They use active learning to select points to label with the goal of improving the performance of a SVM-based anomaly detection system. They find that active learning improves the performance of the system when facing malicious traffic that attempts to appear normal by including common HTTP headers. Neither they nor Sculley et al. study attacks specifically directed at active learning.

Adversarial Sample Creation. Using active learning in an adversarial setting raises new challenges for the learning system. The one with which this work is primarily concerned is that the adversary might craft its instances to affect the sample the learner requests labels for. Zhao et al. study an adversary's ability to attack active learning in this manner [60]. They examine how an adversary inserting or removing clusters of instances can decrease the quality of the instances sampled by the active learner for labeling.

To select the decoy instances to add they simulate a learning algorithm and add instances that have a high entropy (uncertainty) under their classifier. Assuming that the attacked active learner uses a similar learning algorithm and

opts to label instances with high uncertainty, the active learner will select the attacker's decoy instances. Since the decoy instances are crafted to be misleading by the adversary, they may lead the algorithm astray. They also select instances to delete by maximizing entropy. Zhao et al. run experiments to find that adding and deleting points in this manner degrades the quality of an active learner.

7. CONCLUSION

Involving humans in the learning process is always expensive. Humans, even in crowdsourcing scenarios, are a limited resource. Humans have limited capacity. Humans make errors, sometimes maliciously. Active learning is an approach to prioritizing issues that are presented to humans in an attempt to best use people — a finite and sometimes erroneous resource.

In this open problem paper, we discuss how the use of active learning in security raises particular adversarial concerns. These concerns lead to a number of open areas for further investigation (see Table 1). We are proposing a particular software framework, SALT, to allow these experiments. We believe that the availability of SALT will allow the security community to conduct experiments that might otherwise be impractical. In particular, SALT will allow an experimenter to detect drift, to model humans as noisy oracles and to evaluate different query strategies. SALT will allow experiments that otherwise would be impossible to fine tune prioritization algorithms and predict optimal levels of human resources.

We hope that this paper will help provoke discussion in the AISec community about the role of active learning in security, adversarial active learning, experiments in security and active learning, and suggestions and critiques of our proposed SALT software framework.

Acknowledgements

This research is supported in part by NSF CISE Expeditions Award CCF-1139158, LBNL Award 7076018, and DARPA XData Award FA8750-12-2-0331, and gifts from Amazon Web Services, Google, SAP, The Thomas and Stacey Siebel Foundation, Adobe, Apple Inc., Bosch, C3Energy, Cisco, Cloudera, EMC, Ericsson, Facebook, GameOnTalis, Guavus, HP, Huawei, Intel, Microsoft, NetApp, Pivotal, Splunk, Virdata, VMware, and Yahoo!. We also gratefully acknowledge funding support from the Freedom 2 Connect Foundation, Intel, the National Science Foundation, Open Technology Fund, the TRUST Science and Technology Center, the US Department of State Bureau of Democracy, Human Rights, and Labor. The opinions in this paper are those of the authors and do not necessarily reflect the opinions of any funding sponsor or the United States Government.

References

[1] Apache Spark. Machine learning library (mllib), 2014. `http://spark.apache.org/docs/latest/mllib-guide.html`.

[2] J. Azimi, A. Fern, X. Z. Fern, G. Borradaile, and B. Heeringa. Batch active learning via coordinated matching. In *Proceedings of the 29th International Conference on Machine Learning (ICML 2012)*. icml.cc / Omnipress, 2012.

[3] M.-F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 65–72, New York, NY, USA, 2006. ACM.

[4] M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar. The security of machine learning. *Mach. Learn.*, 81(2):121–148, Nov. 2010.

[5] A. Beygelzimer, J. Langford, Z. Tong, and D. J. Hsu. Agnostic active learning without constraints. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 199–207, 2010.

[6] M. Bloodgood and C. Callison-Burch. Bucking the trend: large-scale cost-focused active learning for statistical machine translation. In *Proceedings of ACL*, 2010.

[7] Brian Krebs. Antivirus is dead: Long live antivirus!, 2014. `http://krebsonsecurity.com/2014/05/antivirus-is-dead-long-live-antivirus/#more-25861`.

[8] M. Brückner and T. Scheffer. Nash equilibria of static prediction games. In Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 171–179. Curran Associates, Inc., 2009.

[9] X. Chai, L. Deng, Q. Yang, and C. X. Ling. Test-cost sensitive naive bayes classification. In *Proceedings of the Fourth IEEE International Conference on Data Mining*, ICDM '04, pages 51–58, Washington, DC, USA, 2004. IEEE Computer Society.

[10] S. Chakraborty, V. N. Balasubramanian, and S. Panchanathan. Dynamic batch mode active learning via l1 regularization. In W. Burgard and D. Roth, editors, *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2011)*. AAAI Press, 2011.

[11] Y. Chen and A. Krause. Near-optimal batch mode active learning and adaptive submodular optimization. In *Proceedings of the 30th International Conference on Machine Learning (ICML 2013)*, volume 28 of *JMLR Proceedings*, pages 160–168. JMLR.org, 2013.

[12] D. A. Cohn. Neural network exploration using optimal experiment design. *Neural Netw.*, 9(6):1071–1083, Aug. 1996.

[13] A. Culotta and A. McCallum. Reducing labeling effort for structured prediction tasks. In *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 2*, AAAI'05, pages 746–751. AAAI Press, 2005.

[14] N. Dalvi, P. Domingos, S. Sanghai, D. Verma, et al. Adversarial classification. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 99–108. ACM, 2004.

[15] Damballa, Inc. 3% to 5% of enterprise assets are compromised by bot-driven targeted attack malware, 2009. `http://www.prnewswire.com/news-releases/3-to-5-of-enterprise-assets-are-compromised-by-bot-driven-targeted-attack-malware-61634867.html`.

[16] S. Dasgupta, C. Monteleoni, and D. J. Hsu. A general agnostic active learning algorithm. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 353–360. Curran Associates, Inc., 2008.

[17] A. P. Dawid and A. M. Skene. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied Statistics*, 28(1):20–28, 1979.

[18] O. Dekel, C. Gentile, and K. Sridharan. Robust selective sampling from single and multiple teachers. In *Conference on Learning Theory (COLT)*, pages 346–358, 2010.

[19] N. Görnitz, M. Kloft, K. Rieck, and U. Brefeld. Active learning for network intrusion detection. In *Proceedings of the Second ACM Workshop on Security and Artificial Intelligence*, AISec '09, pages 47–54, New York, NY, USA, 2009. ACM.

[20] R. Greiner, A. J. Grove, and D. Roth. Learning cost-sensitive active classifiers. *Artif. Intell.*, 139(2):137–174, Aug. 2002.

[21] Y. Guo and D. Schuurmans. Discriminative batch mode active learning. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*. Curran Associates, Inc., 2007.

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

[23] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in P2P networks. In *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, pages 640–651, New York, NY, USA, 2003. ACM.

[24] M. Kantarcıoğlu, B. Xi, and C. Clifton. Classifier evaluation and attribute selection against active adversaries. *Data Min. Knowl. Discov.*, 22(1-2):291–335, Jan. 2011.

[25] A. Kantchelian, S. Afroz, L. Huang, A. C. Islam, B. Miller, M. C. Tschantz, R. Greenstadt, A. D. Joseph, and J. Tygar. Approaches to adversarial drift. In *Proceedings of the 2013 ACM Workshop on Artificial Intelligence and Security*, pages 99–110. ACM, 2013.

[26] A. Kantchelian, J. Ma, L. Huang, S. Afroz, A. Joseph, and J. Tygar. Robust detection of comment spam using entropy rate. In *Proceedings of the 5th ACM Workshop on Security and Artificial Intelligence*, pages 59–70. ACM, 2012.

[27] A. Kittur, E. H. Chi, and B. Suh. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 453–456, New York, NY, USA, 2008. ACM.

[28] D. D. Lewis and W. A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '94, pages 3–12, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[29] S. Lomax and S. Vadera. A survey of cost-sensitive decision tree induction algorithms. *ACM Comput. Surv.*, 45(2):16:1–16:35, Mar. 2013.

[30] D. Lowd and C. Meek. Adversarial learning. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, KDD '05, pages 641–647, New York, NY, USA, 2005. ACM.

[31] D. Lowd and C. Meek. Good word attacks on statistical spam filters. In *Proceedings of the Second Conference on Email and Anti-Spam (CEAS)*, 2005.

[32] S. Marti and H. Garcia-Molina. Taxonomy of trust: Categorizing P2P reputation systems. *Comput. Netw.*, 50(4):472–484, Mar. 2006.

[33] P. Melville, M. Saar-Tsechansky, F. Provost, and R. Mooney. Active feature-value acquisition for classifier induction. In *Fourth IEEE International Conference on Data Mining (ICDM '04)*, pages 483–486, Nov 2004.

[34] A. Mukherjee, B. Liu, and N. Glance. Spotting fake reviewer groups in consumer reviews. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 191–200, New York, NY, USA, 2012. ACM.

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

[36] B. Nelson, L. Huang, A. D. Joseph, S. hon Lau, S. J. Lee, S. Rao, A. Tran, J. D. Tygar, and B. I. Rubinstein. Near-optimal evasion of convex-inducing classifiers. In Y. W. Teh and D. M. Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS-10)*, volume 9, pages 549–556, 2010.

[37] H. Raghavan, O. Madani, and R. Jones. InterActive feature selection. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, IJCAI'05, pages 841–846, San Francisco, CA, USA, 2005. Morgan Kaufmann Publishers Inc.

[38] H. Raghavan, O. Madani, and R. Jones. Active learning with feedback on features and instances. *J. Mach. Learn. Res.*, 7:1655–1686, Dec. 2006.

[39] C. Sawade. *Active Evaluation of Predictive Models*. PhD thesis, Universität Potsdam, 2013.

[40] C. Sawade, N. Landwehr, S. Bickel, and T. Scheffer. Active risk estimation. In J. Fürnkranz and T. Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 951–958, Haifa, Israel, June 2010. Omnipress.

[41] D. Sculley, M. E. Otey, M. Pohl, B. Spitznagel, J. Hainsworth, and Y. Zhou. Detecting adversarial advertisements in the wild. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 274–282, New York, NY, USA, 2011. ACM.

[42] B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[43] B. Settles and M. Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 1070–1079, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[44] H. S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 287–294, New York, NY, USA, 1992. ACM.

[45] V. S. Sheng, F. Provost, and P. G. Ipeirotis. Get another label? Improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 614–622, New York, NY, USA, 2008. ACM.

[46] P. Smyth, U. Fayyad, M. Burl, P. Perona, and P. Baldi. Inferring ground truth from subjective labelling of venus images. *Advances in Neural Information Processing Systems 7*, pages 1085–1092, 1995.

[47] A. K. Sood, R. J. Enbody, and R. Bansal. Dissecting SpyEye – Understanding the design of third generation botnets. *Comput. Netw.*, 2013.

[48] G. Suarez-Tangil, J. E. Tapiador, P. Peris-Lopez, and J. Blasco. Dendroid: A text mining approach to analyzing and classifying code structures in Android malware families. *Expert Systems with Applications*, 41:1104–1117, 2014.

[49] Trend Micro. Cybercrime hits the unexpected, 2014. `http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/reports/rpt-cybercrime-hits-the-unexpected.pdf`.

[50] VirusTotal. `https://www.virustotal.com/en/statistics/`. Retrieved on July 30, 2014.

[51] Y. Vorobeychik and J. R. Wallrabenstein. Using machine learning for operational decisions in adversarial environments. To appear in *International Joint Workshop on Optimisation in Multi-Agent Systems and Distributed Constraint Reasoning (OptMAS-DCR)*, 2014.

[52] N. Šrndic and P. Laskov. Detection of Malicious PDF Files Based on Hierarchical Document Structure. In *Proceedings of the 20th Annual Network & Distributed Systems Symposium*. The Internet Society, 2013.

[53] N. Šrndic and P. Laskov. Practical evasion of a learning-based classifier: A case study. In *Proceedings of the 2014 IEEE Symposium on Security and Privacy*, SP '14, pages 197–211, Washington, DC, USA, 2014. IEEE Computer Society.

[54] R. S. Westmoreland. Zeus, 2010. `http://www.antisource.com/article.php/zeus-botnet-summary`.

[55] J. Whitehill, P. Ruvolo, T. Wu, J. Bergsma, and J. R. Movellan. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in Neural Information Processing Systems 22 (NIPS 2009)*, pages 2035–2043, 2009.

[56] C. Whittaker, B. Ryner, and M. Nazif. Large-scale automatic classification of phishing pages. In *Network and Distributed System Security Symposium (NDSS)*. The Internet Society, 2010.

[57] L. Yang. Active learning with a drifting distribution. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2079–2087. Curran Associates, Inc., 2011.

[58] B. Yu and M. P. Singh. Detecting deception in reputation management. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '03, pages 73–80, New York, NY, USA, 2003. ACM.

[59] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, and I. Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, NSDI'12, pages 2–2, Berkeley, CA, USA, 2012. USENIX Association.

[60] W. Zhao, J. Long, J. Yin, Z. Cai, and G. Xia. Sampling attack against active learning in adversarial environment. In *Proceedings of the 9th International Conference on Modeling Decisions for Artificial Intelligence*, MDAI'12, pages 222–233, Berlin, Heidelberg, 2012. Springer-Verlag.

[61] I. Žliobaitė, A. Bifet, G. Holmes, and B. Pfahringer. MOA concept drift active learning strategies for streaming data. In *The Second Workshop on Applications of Pattern Analysis (WAPA)*, pages 48–55, 2011.

[62] V. B. Zubek and T. G. Dietterich. Pruning improves heuristic search for cost-sensitive learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, ICML '02, pages 19–26, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.