Understanding the Risk Factors of Learning in Adversarial Environments

Blaine Nelson
Wilhelm Schickard Institute for Computer Science
University of Tübingen
Sand 1, 72076 Tübingen, Germany
blaine.nelson@wsii.uni-tuebingen.de

Battista Biggio
Department of Electrical and Electronic Engineering
University of Cagliari
Piazza d'Armi
09123 Cagliari, Italy
battista.biggio@diee.unica.it

Pavel Laskov
Wilhelm Schickard Institute for Computer Science
University of Tübingen
Sand 1, 72076 Tübingen, Germany
pavel.laskov@uni-tuebingen.de

ABSTRACT

Learning for security applications is an emerging field where adaptive approaches are needed but are complicated by changing adversarial behavior. Traditional approaches to learning assume benign errors in data and thus may be vulnerable to adversarial errors. In this paper, we incorporate the notion of adversarial corruption directly into the learning framework and derive a new criteria for classifier robustness to adversarial contamination.

Categories and Subject Descriptors

D.4.6 [**Security and Protection**]: Invasive software (e.g., viruses, worms, Trojan horses); G.3 [**Probability and Statitics**]: Statistical computing; I.5.1 [**Models**]: Statistical; I.5.2 [**Design Methodology**]: Classifier design and evaluation

General Terms

Security

Keywords

Adversarial Learning, Computer Security, Machine Learning, Statistical Learning, Robust Classification

1. INTRODUCTION

Applying learning algorithms to detection problems in security domains has received a great deal of recent attention. Learning approaches are well-suited to these tasks because of the rapidly-changing nature of adversaries and the need for high-speed automated analysis of complex data. However, as has been witnessed in spam-detection, adversaries adapt their approaches based on the detection algorithms. A clever adversary can change their behavior either to evade or mislead a learning algorithm. When exposed to an adversary, learning-based detection algorithms may not only be ineffective, they can in fact become a hindrance or liability to the system. Because of these threats, careful choice and analysis of learning algorithms is critical.

In this paper, we propose a framework for analyzing and selecting learning algorithms for security-sensitive learning tasks. Based on a decomposition of the expected risk, we propose a metric for classifier stability under contamination. Our metric adds a new dimension to the process of selecting a learning algorithm, which incorporates the security concern of adversarial contamination.

Prior Work.

A great deal of prior work has addressed different aspects of secure learning; *e.g.*, [1, 9, 10, 11]. However, to our knowledge, the only systematic approach to learning with contaminated data comes from robust statistics [7, 8]. Under this framework, estimators are designed to be robust to contamination. The major tool used to assess an estimator's robustness is its influence function—a measure of the asymptotic bias due to an infinitesimal contamination. While having a bounded influence function is important robustness property, it does not completely describe a classifier's stability. Under contamination, the change to a classifier's parameters may be bounded but its classification performance may still be dramatically decrease. To see this, consider that a hyperplane that classifies well need only rotate 180° to classify poorly. In this paper, we derive a measure of classifier stability that directly captures how contamination affects classification performance; we call this *classification robustness*.

2. CLASSIFICATION

In this section, we briefly introduce the classification problem and the framework of risk minimization. Data comes from an input space denoted by $\mathcal{X}$ and the prediction space is the set $\mathcal{Y} = \{-1, +1\}$. These spaces are paired as the space $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ and the training data points are paired instances from this space: $\mathbb{D} = \{(\mathbf{x}_i, y_i) \in \mathcal{Z}\}_{i=1}^{N}$. We assume the dataset is drawn independently and identically distrubuted from a joint distribution $P_{\mathcal{Z}}$ when there is no adversarial influence.

A hypothesis (or classifier) is a function f mapping $f : \mathcal{X} \to \mathcal{Y}$. Of course, there are many such hypotheses belong-

AISec'11, October 21, 2011, Chicago, Illinois, USA.

ing to the hypothesis space $\mathcal{F}$ of all possible hypotheses. For instance, learners often only consider the space of linear classifiers of the form $f_{\mathbf{w},b}(\mathbf{x}) = 2 \cdot \mathrm{I}\left[\mathbf{w}^\top \mathbf{x} \geq b\right] - 1$ where $\mathrm{I}[\cdot]$ is the indicator function for an event, $\mathbf{w}$ is the hyperplane's normal vector, and b is its bias. The rest of this section discusses how a hypothesis is selected.

2.1 Statistical Learning

We formalize the learning process in terms of a two-step process, in which a dataset is first produced (according to some natural distribution) and a learning algorithm $\mathcal{L}$ uses that dataset to produce a hypothesis as follows:

1. **Nature:** Produce dataset, $\mathbb{D} \sim P_{\mathcal{Z}}$
2. **Learner:** Produce hypothesis, $f = \mathcal{L}(\mathbb{D})$

The learned hypothesis f produced by the learning algorithm is a predictor that, when given a new instance $\mathbf{x} \in \mathcal{X}$ produces a prediction $\hat{y} = f(\mathbf{x})$. This hypothesis is selected by the learning algorithm $\mathcal{L}$ from its hypothesis space $\mathcal{F}$ (presumably in accordance with the data $\mathbb{D}$).

A performance measure is used to assess the ability of the hypothesis to predict the label of a given data point. In statistical learning theory, this performance measure is called a *loss function* ℓ and assigns a non-negative cost for making the prediction $\hat{y} = f(\mathbf{x})$ when the true label is y; that is, $\ell : \mathcal{Y} \times \mathcal{Y} \mapsto \Re^{0+}$. For classification, the zero-one loss captures the standard notion of loss, which is zero when $\hat{y} = y$ and one otherwise.[1]

For a given data distribution, $P_{\mathcal{Z}}$, the expected loss for the hypothesis, f, assesses its prediction capability. The expected loss is called the *risk* of the hypothesis and is

$$R(f) \quad = \quad \mathrm{E}_{(x,y)\sim P_{\mathcal{Z}}}\left[\ell\left(y, f\left(x\right)\right)\right] \quad .$$

For a fixed hypothesis, f, the risk is a quantity that can be computed or estimated. However, notice that when the hypothesis itself is selected by a procedure that depends on data, the risk is a random variable. Hence, the expected risk (over a random dataset $\mathbb{D}$ drawn from $P_{\mathcal{Z}}$) yields a measure of how well a learning algorithm $\mathcal{L}$ performs over a particular data distribution and will be the primary focus of this paper.

2.2 Risk Minimization Framework

Above, we introduced the standard risk approach to assessing a learning algorithm and we now describe how risk is used to compare and design learning algorithms. The learner $\mathcal{L}$ is tasked with selecting a *hypothesis* f that, when given a new data point $\mathbf{x}$, best predicts its label y and its performance is assessed by its risk $R(f)$. Since the hypothesis is assessed according to its ability to minimize expected risk, it is natural to consider learning algorithms that select hypotheses that minimize risk. That is, the learning algorithm should select a hypothesis $f^\dagger \in \mathcal{F}$ that minimizes risk according to the data's distribution $P_{\mathcal{Z}}$; *i.e.*,

$$f^\dagger \in \operatorname*{argmin}_{f \in \mathcal{F}} R(f) \quad .$$

This procedure is the *risk minimizer*, but is generally not possible to find since the distribution $P_{\mathcal{Z}}$ is not known.

[1] In most settings, minimizing the zero-one loss is intractable and a surrogate lost function is used instead. However, for our purposes, we work with the zero-one loss directly.

2.2.1 Empirical Risk Minimization

While directly minimizing risk is infeasible since $P_{\mathcal{Z}}$ is unknown, the dataset $\mathbb{D}$ provides information about this distribution. Under the usual *stationarity assumption* the training data is drawn from *same* distribution $P_{\mathcal{Z}}$ as the test data. Thus, the learner can instead select a hypothesis f_N to minimize the empirical risk $\mathbb{D} \sim P_{\mathcal{Z}}$

$$R_N(f) = \frac{1}{N} \sum_{(x,y)\in\mathbb{D}} \ell\left(y, f\left(x\right)\right)$$

where $N = |\mathbb{D}|$. The practice of minimizing this surrogate for the true risk is known as *empirical risk minimization* [13].

2.2.2 Regularization

Since the dataset is of limited size, the learner must also restrict the space of hypotheses, $\mathcal{F}$. If the space of hypotheses is too expressive, some hypothesis may fit the training data exactly but may not make accurate predictions about unseen instances. This phenomenon is known as *overfitting*. To avoid overfitting the learner may use a small or restricted space of hypotheses; *e.g.*, linear classifiers. Alternatively, one could allow for a large space of hypotheses, but penalize hypothesis complexity—a practice known as *regularization*. The learner selects hypothesis f_N that minimizes the modified objective

$$R_N(f) + \lambda \cdot \rho(f)$$

where the function $\rho : \mathcal{F} \to \Re$ is a measure of the complexity of a hypothesis and $\lambda > 0$ controls the trade-off.

2.2.3 Risk Analysis

We now reexamine the expected risk of a learning algorithm and break it into two components: one that captures the cost of learning on a finite dataset and the second that captures the cost inherent to the choice of hypothesis space. Suppose that f^* is the hypothesis (not necessarily in $\mathcal{F}$) that minimizes the expected risk (over all datasets) and that $f^\dagger \in \mathcal{F}$ is the minimizer in the hypothesis space. We want to compare the expected risk of the hypothesis f_N based on the dataset to the expected risk of the optimal classifier. The expected risk can be decomposed into two components:

$$\begin{aligned}
\mathrm{E}_{\mathbb{D}}\left[R(f_N) - R(f^*)\right] &= \underbrace{\mathrm{E}_{\mathbb{D}}\left[R(f_N) - R\left(f^\dagger\right)\right]}_{\varepsilon_{\text{est}}} \\
&\quad + \underbrace{\mathrm{E}_{\mathbb{D}}\left[R\left(f^\dagger\right) - R(f^*)\right]}_{\varepsilon_{\text{apprx}}} \\
&= \varepsilon_{\text{est}} + \varepsilon_{\text{apprx}}
\end{aligned}$$

where the outer expectation is over the training data $\mathbb{D}$ drawn from the distribution $P_{\mathcal{Z}}$ and the risks are expectations over the test data also drawn from the same distribution. This breakdown is the classical decomposition of error in learning theory. The term $\varepsilon_{\text{apprx}}$ captures the *approximation error* due to the limitations of the hypothesis class and the term ε_{est} captures the *estimation error* due to limitations of learning on a finite dataset. Motivated by an extension to this decomposition of expected risk proposed by Bottou and Bousquet [2], we now introduce our own third component to the expected risk, which captures the error due to adversarial contamination.

3. DATA CONTAMINATION

For the purposes of security analysis, we now assume that, in addition to the training dataset being finite, it has also been contaminated or altered by an adversary. We model this contamination event as a transformation $\mathcal{A} : \mathcal{Z}^N \to \mathcal{Z}^N$ on the training data that produces the altered data as in Dalvi *et al.* [4]. We view this as a game between the adversary and the learner that proceeds in the following three steps:

1. **Nature:** Produce true data, $\mathbb{D} \sim P_{\mathcal{Z}}$
2. **Adversary:** Contaminate dataset, $\hat{\mathbb{D}} = \mathcal{A}(\mathbb{D})$
3. **Learner:** Produce hypothesis, $\hat{f}_N = \mathcal{L}\left(\hat{\mathbb{D}}\right)$

Importantly, in this setting, there is some clean dataset $\mathbb{D}$ drawn from the underlying distribution $P_{\mathcal{Z}}$ and has been transformed. The adversary presumably chooses his transformation to maximize

$$\mathrm{E}_{\mathbb{D}}\left[R\left(\hat{f}_N\right)\right]$$

under some constraints but we are agnostic at this point to the choice of $\mathcal{A}$. We simply assume $\hat{f}_N$ is the result of learning on contaminated data and again examine the difference between the risk of the tainted hypothesis and of best hypothesis. To this end we introduce, as intermediates, the risk of the best hypothesis in $\mathcal{F}$ and the risk of a hypothesis learned by the same algorithm on the *clean* dataset $\mathbb{D}$. We thus break this expected difference into three components:

$$\begin{aligned}\mathrm{E}_{\mathbb{D}}\left[R\left(\hat{f}_N\right) - R\left(f^*\right)\right] &= \underbrace{\mathrm{E}_{\mathbb{D}}\left[R\left(\hat{f}_N\right) - R\left(f_N\right)\right]}_{\varepsilon_{\text{rbst}}} \\ &+ \underbrace{\mathrm{E}_{\mathbb{D}}\left[R\left(f_N\right) - R\left(f^\dagger\right)\right]}_{\varepsilon_{\text{est}}} + \underbrace{\mathrm{E}_{\mathbb{D}}\left[R\left(f^\dagger\right) - R\left(f^*\right)\right]}_{\varepsilon_{\text{apprx}}} \\ &= \varepsilon_{\text{rbst}} + \varepsilon_{\text{est}} + \varepsilon_{\text{apprx}}\end{aligned}$$

Under this extended decomposition, we introduce a classification robustness term, $\varepsilon_{\text{rbst}}$, which measures the error caused by the (adversarial) transformation $\mathcal{A}$.

3.1 Analyzing Classification Robustness

The error term $\varepsilon_{\text{rbst}}$ captures a notion of stability for the learning algorithm, $\mathcal{L}$. Clearly, if a learning algorithm is stable under a reasonable level of (worst-case) noise, this term will be small and we can expect nearly the same risk from function $\hat{f}_N$ learned with the contamination as we would obtain from a hypothesis f_N learned from clean data. Naturally, there are trade-offs between classification robustness and the other forms of error; *i.e.*, learning procedures that are highly stable may not generalize as quickly (and hence have a higher estimation error) or may come from an overly restricted hypothesis class (and hence have a higher approximation error). However, in this paper we focus solely on the classification robustness and how it can be incorporated to the design and selection of learning algorithms in contamination settings.

Under certain assumptions, the stability error can be rewritten in several ways. Firstly, noting that the risk itself is an expectation, we can merge the two risk terms as follows:

$$\begin{aligned}\varepsilon_{\text{rbst}} &= \mathrm{E}_{\mathbb{D}}\left[R\left(\hat{f}_N\right) - R\left(f_N\right)\right] \\ &= \mathrm{E}_{\mathbb{D}}\left[\mathrm{E}_{(\mathbf{x},y)}\left[\ell\left(y, \hat{f}_N(\mathbf{x})\right)\right] - \mathrm{E}_{(\mathbf{x},y)}\left[\ell\left(y, f_N(\mathbf{x})\right)\right]\right] \\ &= \mathrm{E}_{\mathbb{D}}\left[\mathrm{E}_{(\mathbf{x},y)}\left[\ell\left(y, \hat{f}_N(\mathbf{x})\right) - \ell\left(y, f_N(\mathbf{x})\right)\right]\right]\end{aligned}$$

Examining this difference between losses, we can further refine it under the assumption that the loss function obeys on the following triangle inequality:

$$\begin{aligned}\ell(x,y) &\leq \ell(x,z) + \ell(z,y) \\ \ell(x,y) - \ell(x,z) &\leq \ell(z,y)\end{aligned}$$

This triangle inequality is exhibited by the zero-one loss since it is the discrete metric and yields the following bound:

$$\varepsilon_{\text{rbst}} \leq \mathrm{E}_{\mathbb{D}}\left[\mathrm{E}_{(\mathbf{x},y)}\left[\ell\left(f_N(\mathbf{x}), \hat{f}_N(\mathbf{x})\right)\right]\right]$$

This bound is particularly interesting because it compares two hypotheses not on how well they predict the true label y but rather on how well the two hypotheses agree. In this sense, this approach is similar to the notion of regret used in online prediction games [3].

We can further expand the inner expectation for the classification functions with a label space $\mathcal{Y} = \{-1, +1\}$, for which the zero-one loss can be rewritten as

$$\ell\left(f_N(\mathbf{x}), \hat{f}_N(\mathbf{x})\right) = \tfrac{1}{2}\left[1 - f_N(\mathbf{x}) \cdot \hat{f}_N(\mathbf{x})\right] \quad . \tag{1}$$

This result is derived from the fact that

$$f_N(\mathbf{x}) \cdot \hat{f}_N(\mathbf{x}) = \begin{cases} +1 & \text{if } f_N \text{ and } \hat{f}_N \text{ agree on } \mathbf{x} \\ -1 & \text{otherwise} \end{cases} .$$

3.2 A Measure of Classification Robustness

Here we further expand our bound on the classification robustness for certain families of classifiers. Namely, we consider the family, $\mathcal{F}^{\text{ind}}$, of classifiers that can be expressed as a threshold on a real-valued decision function g in the form

$$f(\mathbf{x}) = 2 \cdot \mathrm{I}\left[g(\mathbf{x}) \geq 0\right] - 1$$

where $g : \mathcal{X} \to \Re$. Based on the properties of the indicator, $\mathrm{I}[\cdot]$, for any pair of classifiers $f_1, f_2 \in \mathcal{F}^{\text{ind}}$, we have that

$$f_1(\mathbf{x}) \cdot f_2(\mathbf{x}) = 2 \cdot \mathrm{I}\left[g_1(\mathbf{x}) \cdot g_2(\mathbf{x}) \geq 0\right] - 1 \quad .$$

From this, the zero-one loss from Eq. (1) simplifies for classifiers f_N and $\hat{f}_N$ in this family to

$$\ell\left(f_N(\mathbf{x}), \hat{f}_N(\mathbf{x})\right) = 1 - \mathrm{I}\left[g_N(\mathbf{x}) \cdot \hat{g}_N(\mathbf{x}) \geq 0\right] \quad .$$

Thus, to minimize the expected risk under the zero-one loss, we can instead minimize

$$\mathrm{P}_{\mathbf{x} \sim P_{\mathcal{X}}}\left(g_N(\mathbf{x}) \cdot \hat{g}_N(\mathbf{x}) < 0\right)$$

where $P_{\mathcal{X}}$ is the marginal distribution over $\mathcal{X}$. Taking the expectation over the dataset of this probability for the corresponding decision functions $\hat{g}_N$ and g_N yields our upper bound on classification robustness,

$$\varepsilon_{\text{rbst}} \leq \mathrm{E}_{\mathbb{D}}\left[\mathrm{P}_{\mathbf{x} \sim P_{\mathcal{X}}}\left(g_N(\mathbf{x}) \cdot \hat{g}_N(\mathbf{x}) < 0\right)\right] \quad .$$

Unfortunately, this notion of stability is highly dependent on the unknown marginal distribution $P_{\mathcal{X}}$ of the test data. At issue is the fact that, unless $\hat{f}_N$ and f_N everywhere agree

or disagree in sign, this probability can be made arbitrarily high or low if the distribution $P_{\mathcal{X}}$ is concentrated in the classifiers' region of agreement or disagreement, respectively. We desire a measure of robustness that is distribution independent and thus choose an alternative measure of classification robustness that is *agnostic* to the distribution of the test data. We therefore measure the classifiers' agreement against a uniform distribution U on the data and thus define the surrogate measure[2]

$$\hat{\varepsilon}_{\text{rbst}} \triangleq \mathrm{E}_{\mathbb{D}}\left[\mathrm{P}_{\mathbf{x}\sim U}\left(g_N\left(\mathbf{x}\right)\cdot\hat{g}_N\left(\mathbf{x}\right)<0\right)\right] \quad . \tag{2}$$

This quantity directly measures the stability of a learning algorithm's predictions due to a transformation on its training data (as to be discussed in the next section) whereas prior robustness criteria assessed robustness indirectly by measuring the sensitivity of learned parameters.

3.3 Application to Linear Classifiers

A special case of the family of thresholded real-valued functions occurs when the function g is a linear function that can be expressed in terms of a normal vector $\mathbf{w}$ and displacement b as $g\left(\mathbf{x}\right)=\mathbf{w}^{\top}\mathbf{x}-b$. Here we consider the special case when $b=0$ (non-zero displacements can be accounted for by appending it to the normal vector). For this family, $\mathcal{F}^{\text{lin}}$, the poison and clean hypotheses are parameterized by $\hat{\mathbf{w}}_N$ and $\mathbf{w}_N$ respectively and from Eq. (2) we want to minimize the expected value of

$$\mathrm{P}_{\mathbf{x}\sim U}\left(\mathbf{w}_N{}^{\top}\mathbf{x}\cdot\hat{\mathbf{w}}_N^{\top}\mathbf{x}<0\right) \quad .$$

We can consider the normal vectors $\mathbf{w}_N$ and $\hat{\mathbf{w}}_N$ to have unit length (their norm can be absorbed into the 0). Dasgupta *et al.* [5] note that the above probability over the uniform distribution is simply given by

$$\tfrac{1}{\pi}\arccos\left(\frac{\mathbf{w}_N{}^{\top}\hat{\mathbf{w}}_N}{\left\|\mathbf{w}_N\right\|\left\|\hat{\mathbf{w}}_N\right\|}\right)$$

and thus, the classification robustness of the linear classifiers is given simply by $\frac{1}{\pi}$ times the expected angle of the normals for the hyperplane learned on transformed and non-transformed data (varying from 0 when the normals are aligned to 1 when they are oppositely aligned). This measure of stability corresponds to our previously stated mentioned notion in that (1) it accounts for hyperplane rotation in measuring error and (2) it doesn't consider the magnitude of the normal (which does not affect the classifier's decision performance).

Support Vector Machines.

Support vector machines and other kernel-based classifiers can also be assessed under this analysis because they are linear in their feature space and their normal vector can be expressed simply as a linear combination of the training data. Thus, computing the above angle is feasible using the kernel matrix.

[2]In some settings, it may be possible to bound $\varepsilon_{\text{rbst}}$ directly. However, $\hat{\varepsilon}_{\text{rbst}}$ is more desirable in a sense, because the uniform distribution distributes its mass over $\mathcal{X}$ equally. Thus, $\hat{\varepsilon}_{\text{rbst}}$ is a more distribution-independent measure of the learning algorithm's stability since all discrepancies between the clean and contaminated hypotheses are equally weighted.

4. FROM THEORY TO PRACTICAL ALGORITHMS

In the preceding section, we derived the notion of classification robustness but the quantity we derived, $\hat{\varepsilon}_{\text{rbst}}$, can not be directly calculated in practice unless both the clean and contaminated classifiers are known or can be inferred. In practice, we believe exact computation of $\hat{\varepsilon}_{\text{rbst}}$ may not generally be possible but it could be bounded (as with ε_{est} and $\varepsilon_{\text{apprx}}$) for a particular family of adversarial transformations; *i.e.*, the set of contamination actions available to the adversary. By bounding the classification robustness under a particular contamination model, a worst-case bound is obtained and can be used as a criteria for selecting algorithms for a particular setting where that particular contamination model is realistic. Thus far, no non-trivial bounds on $\hat{\varepsilon}_{\text{rbst}}$ have been obtained for any algorithm and we leave this task to future work. However, below we discuss a set of different contamination models that may arise in security-sensitive applications to motivate further research.

Thus far, our analysis assumed that there is some transformation, $\mathcal{A}:\mathcal{Z}^N\rightarrow\mathcal{Z}^N$, that alters the training data to a limited extent. Now we briefly explore realistic scenarios for data corruption/transformation and we discuss the security implications of these settings. For each model, we describe how the adversarial contamination changes the data and what limitations are placed on the extent of contamination. The task that lays ahead is bounding $\hat{\varepsilon}_{\text{rbst}}$ for algorithms under the following contamination models.

4.1 Outliers

In many applications, it is appropriate to assume that some of the training data are outliers (as in the usual contamination model for robust statistics [7, 8]); that is, some fraction of the data points are from an altogether different source than the data distribution $P_{\mathcal{Z}}$. These outliers may be due benign mistakes such as mislabeled data points or can be maliciously chosen by an adversary. Naturally, for learning to be successful, we must assume that only a fraction ϵ of the data points are outliers. The transformation selects a fraction ϵ of the dataset and replaces it with data points drawn from some distribution $Q_{\mathcal{Z}}$.

4.2 Data Perturbation

An alternative model for data corruption involves noise among all the data points $\mathbf{x}_i\in\mathbb{D}$. These models for noise have previously been used to design more robust kernel learning algorithms [12, 14]. In this scenario, all of the data is perturbed by either a natural source of noise or by a malicious one. Of course, the level of noise must be bounded and we can think of $\mathcal{A}$ as adding some noise σ_i the data points with the limitation that the total noise is less than ϵ.

4.3 Label Flipping

Yet another model for data corruption instead assumes that noise occurs amongst the labels; that is, some process causes labels in the dataset to be flipped: $y_i\rightarrow-1\cdot y_i$. As with the outlier model, the size of the contamination is measured the fraction ϵ of labels that are flipped.

4.4 Feature-Constrained Outliers

The last model of corruption we consider in this paper is one in which the adversary can potentially alter any data points but can only change a subset of the features. This

model of contamination was used previously by Globerson and Roweis to model feature deletion [6]. Obviously, the power of the adversary in this model is measured in terms of the fraction of features ϵ that can be influenced by the adversary.

5. DISCUSSION

In this paper, we introduce a measure of classifier stability under adversarial contamination to the training data, which we call classification robustness, $\hat{\varepsilon}_{\text{rbst}}$. This quantity is designed to assess the stability of an algorithm's prediction performance for a particular realistic model of adversarial contamination. Further, for linear classifiers, we showed that this measure intuitively corresponds to the expected rotation angle in the hyperplane due to the contamination. While we believe that $\hat{\varepsilon}_{\text{rbst}}$ is the appropriate measure for assessing the threat posed to a classifier, it remains to be seen which classifiers perform well under this measure. Designing classifiers to trade-off the traditional measures of generalization performance and classifier robustness is a potentially lucrative area for future work because we believe that assessing algorithms' classifier robustness under an appropriate threat model is a critical objective for security applications.

6. ACKNOWLEDGEMENTS

The authors wish to acknowledge the Alexander von Humboldt Foundation and the Heisenberg Fellowship of the Deutsche Forschungsgemeinschaft (DFG) for providing financial support to carry out this research. This work was also partly supported by a grant awarded to B. Biggio by Regione Autonoma della Sardegna, PO Sardegna FSE 2007-2013, L.R. 7/2007 "Promotion of the scientific research and technological innovation in Sardinia". The opinions expressed in this paper are solely those of the authors and do not necessarily reflect the opinions of any sponsor.

7. REFERENCES

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

[2] L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In *Advances in Neural Information Processing Systems (NIPS)*, volume 20, pages 161–168, 2008.

[3] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, New York, NY, USA, 2006.

[4] N. Dalvi, P. Domingos, Mausam, S. Sanghai, and D. Verma. Adversarial classification. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 99–108, 2004.

[5] S. Dasgupta, A. T. Kalai, and C. Monteleoni. Analysis of perceptron-based active learning. *Journal of Machine Learning Research*, 10:281–299, 2009.

[6] A. Globerson and S. Roweis. Nightmare at test time: Robust learning by feature deletion. In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, pages 353–360, 2006.

[7] F. R. Hampel, E. M. Ronchetti, P. J. Rousseeuw, and W. A. Stahel. *Robust Statistics: The Approach Based on Influence Functions*. John Wiley and Sons, 1986.

[8] P. Huber. *Robust Statistics*. John Wiley & Sons, 1981.

[9] M. Kearns and M. Li. Learning in the presence of malicious errors. *SIAM Journal on Computing*, 22(4):807–837, 1993.

[10] P. Laskov and M. Kloft. A framework for quantitative security analysis of machine learning. In *Proceedings of the* 2^{nd} *ACM Workshop on Security and Artificial Intelligence (AISec)*, pages 1–4, 2009.

[11] D. Lowd and C. Meek. Adversarial learning. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 641–647, 2005.

[12] C. H. Teo, A. Globerson, S. T. Roweis, and A. J. Smola. Convex learning with invariances. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

[13] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., 1995.

[14] H. Xu, C. Caramanis, and S. Mannor. Robustness and regularization of support vector machines. *Journal of Machine Learning Research*, 10:1485–1510, 2009.